



2 **The Application Level Events (ALE) Specification,**  
3 **Version 1.1**  
4 **Part II: XML and SOAP Bindings**

5 EPCglobal Recommended Specification of 21 December 2007

6

7 Copyright © 2007, 2004–2006 [EPCglobal](#)<sup>®</sup>, All Rights Reserved.

## 8 Abstract

9 This document specifies an interface through which clients may interact with filtered,  
10 consolidated EPC data and related data from a variety of sources. The design of this  
11 interface recognizes that in most EPC processing systems, there is a level of processing  
12 that reduces the volume of data that comes directly from EPC data sources such as RFID  
13 readers into coarser “events” of interest to applications. It also recognizes that  
14 decoupling these applications from the physical layers of infrastructure offers cost and  
15 flexibility advantages to technology providers and end-users alike. The interface  
16 described herein, and the functionality it implies, is called “Application Level Events,” or  
17 ALE.

18 The role of the ALE interface within the EPCglobal Network Architecture is to provide  
19 independence between the infrastructure components that acquire the raw EPC data, the  
20 architectural component(s) that filter & count that data, and the applications that use the  
21 data. This allows changes in one without requiring changes in the other, offering  
22 significant benefits to both the technology provider and the end-user. The ALE interface  
23 described in the present specification achieves this independence through five means:

- 24 • It provides a means for clients to specify, in a high-level, declarative way, what data  
25 they are interested in or what operations they want performed, without dictating an  
26 implementation. The interface is designed to give implementations the widest  
27 possible latitude in selecting strategies for carrying out client requests; such strategies  
28 may be influenced by performance goals, the native abilities of readers or other  
29 devices which may carry out certain filtering or counting operations at the level of  
30 firmware or RF protocol, and so forth.
- 31 • It provides a standardized format for reporting accumulated, filtered data and results  
32 from carrying out operations that is largely independent of where the data originated  
33 or how it was processed.
- 34 • It abstracts the channels through which data carriers are accessed into a higher-level  
35 notion of “logical reader,” often synonymous with “location,” hiding from clients the  
36 details of exactly what physical devices were used to interact with data relevant to a  
37 particular logical location. This allows changes to occur at the physical layer (for  
38 example, replacing a 2-port multi-antenna reader at a loading dock door with three  
39 “smart antenna” readers) without affecting client applications. Similarly, it abstracts  
40 away the fine-grained details of how data is gathered (*e.g.*, how many individual tag  
41 read attempts were carried out). These features of abstraction are a consequence of  
42 the way the data specification and reporting aspects of the interface are designed.
- 43 • It abstracts the addressing of information stored on Tags and other data carriers into a  
44 higher-level notion of named, typed “fields,” hiding from clients the details of how a  
45 particular data element is encoded into a bit-level representation and stored at a  
46 particular address within a data carrier’s memory. This allows application logic to  
47 remain invariant despite differences between the memory organization of different  
48 data carriers (for example, differences between Gen 1 and Gen 2 RFID Tags), and

49 also shields application logic from having to understand complex layout or data  
 50 parsing rules.

- 51 • It provides a security mechanism so that administrators may choose which operations  
 52 a given application may perform, as a policy that is decoupled from application logic  
 53 itself.

54 Part I [ALE1.1Part1] specifies at an abstract level all interfaces that are part of the ALE  
 55 specification, using UML notation. This Part II specifies XML-based wire protocol  
 56 bindings of the interfaces, including XSD schemas for all data types, WS-I compliant  
 57 WSDL definitions of SOAP bindings of the service interfaces, and several XML-based  
 58 bindings of callback interfaces used in certain modes of reading and writing data.  
 59 Implementations may provide additional bindings of the API, including bindings to  
 60 particular programming languages.

## 61 Audience for this document

62 The target audience for this specification includes:

- 63 • EPC Middleware vendors
- 64 • Reader vendors
- 65 • Application developers
- 66 • System integrators

## 67 Status of this document

68 This section describes the status of this document at the time of its publication. Other  
 69 documents may supersede this document. The latest status of this document series is  
 70 maintained at EPCglobal. See [www.epcglobalinc.org](http://www.epcglobalinc.org) for more information.

71 This draft is an EPCglobal **Recommended Specification** submitted for ratification to the  
 72 EPCglobal Board of Governors. This draft was reviewed and approved by the EPCglobal  
 73 Business Steering Committee on 17 December 2007 and by the Technical Steering  
 74 Committee on 21 December 2007. It is a draft document and may be updated, replaced  
 75 or made obsolete by other documents at any time. It is inappropriate to use EPCglobal  
 76 Working Drafts as reference material or to cite them as other than “work in progress.”  
 77 This is work in progress and does not imply endorsement by the EPCglobal membership.

78 Comments on this document should be sent to the EPCglobal Software Action Group  
 79 Filtering and Collection Working Group mailing list  
 80 [sag\\_fc1\\_1\\_wg@lists.epcglobalinc.org](mailto:sag_fc1_1_wg@lists.epcglobalinc.org).

## 81 Table of Contents

82 1 Introduction ..... 6

83 2 Terminology and Typographical Conventions..... 6

84 3 XML Schemas..... 6

85 3.1 Organization of the XML Schemas ..... 7

86 3.2 Extensibility Mechanism ..... 7

87 3.2.1 Extensibility Design Rules – Classes..... 8

88 3.2.2 Extensibility Design Rules – Interfaces..... 10

89 3.2.3 Extensibility Design Rules – Enumerations ..... 10

90 3.3 EPCglobal Base Schema ..... 11

91 3.4 Schema for Datatypes Common to the Reading API and Writing API..... 12

92 3.5 ALE Reading API Schema ..... 13

93 3.5.1 ECSpec – Example 1 (non-normative) ..... 21

94 3.5.2 ECSpec – Example 2 (non-normative) ..... 22

95 3.5.3 ECReports – Example 1 (non-normative)..... 23

96 3.5.4 ECReports – Example 2 (non-normative)..... 24

97 3.6 ALE Writing API Schema..... 25

98 3.7 ALE Tag Memory API Schema ..... 33

99 3.8 ALE Logical Reader API Schema..... 35

100 3.9 ALE Access Control API Schema..... 36

101 4 SOAP Bindings ..... 38

102 4.1 Organization of the SOAP Bindings ..... 38

103 4.2 Link Level Security ..... 38

104 4.3 ALE Reading API SOAP Binding..... 38

105 4.4 ALE Writing API SOAP Binding ..... 48

106 4.5 ALE Tag Memory API SOAP Binding..... 73

107 4.6 ALE Logical Reader API SOAP Binding ..... 78

108 4.7 ALE Access Control API SOAP Binding ..... 90

109 5 Bindings for the Reading and Writing Callback APIs ..... 112

110 5.1 HTTP Bindings..... 113

111 5.2 TCP Bindings ..... 114

112 5.3 FILE Bindings ..... 114

113 5.4 HTTPS Bindings..... 115

114 6 Appendix: Schema and WSDL Differences from ALE 1.0 (non-normative)..... 116

|     |     |                                      |     |
|-----|-----|--------------------------------------|-----|
| 115 | 6.1 | Fully Compatible Changes .....       | 116 |
| 116 | 6.2 | Non-Forward Compatible Changes ..... | 117 |
| 117 | 6.3 | Corrections.....                     | 117 |
| 118 | 6.4 | Changes in Idiom.....                | 118 |
| 119 | 7   | References .....                     | 118 |
| 120 | 8   | Credits .....                        | 119 |
| 121 |     |                                      |     |

## 122 **1 Introduction**

123 This document is Part II of the ALE 1.1 specification. This Part II specifies bindings of  
124 the classes and interfaces specified in Part I [ALE1.1Part1] that are based on XML  
125 [XML1.0]. In particular, this document specifies XML schemas for all classes defined in  
126 Part I, WSDL specifications of WS-I compliant SOAP interfaces of all interfaces defined  
127 in Part I, and XML-based bindings for the callback interfaces of the Reading and Writing  
128 APIs.

129 Implementations may provide additional bindings of the API, including bindings to  
130 particular programming languages.

## 131 **2 Terminology and Typographical Conventions**

132 Within this specification, the terms SHALL, SHALL NOT, SHOULD, SHOULD NOT,  
133 MAY, NEED NOT, CAN, and CANNOT are to be interpreted as specified in Annex G of  
134 the ISO/IEC Directives, Part 2, 2001, 4th edition [ISODir2]. When used in this way,  
135 these terms will always be shown in ALL CAPS; when these words appear in ordinary  
136 typeface they are intended to have their ordinary English meaning.

137 All sections of this document, with the exception of Section 1, are normative, except  
138 where explicitly noted as non-normative.

139 The following typographical conventions are used throughout the document:

- 140 • ALL CAPS type is used for the special terms from [ISODir2] enumerated above.
- 141 • Monospace type is used to denote programming language, UML, and XML  
142 identifiers, as well as for the text of XML documents.
- 143 ➤ Placeholders for changes that need to be made to this document prior to its reaching  
144 the final stage of approved EPCglobal specification are prefixed by a rightward-  
145 facing arrowhead, as this paragraph is.

## 146 **3 XML Schemas**

147 This section defines the standard XML representation for all data types used by the five  
148 ALE APIs, including ECSpec instances and ECRports instances of the Reading API,  
149 CCSpec instances and CCReports instances of the Writing API, and other types used  
150 by the Writing API, the Tag Memory API, the Logical Reader API, and the Access  
151 Control API. All schemas are specified using the W3C XML Schema language [XSD1,  
152 XSD2]. Samples are also shown.

153 The schemas defined herein conform to EPCglobal standard schema design rules. The  
154 schema below imports the EPCglobal standard base schema, as mandated by the design  
155 rules.

156 **3.1 Organization of the XML Schemas**

157 Because an implementation may not implement all five ALE APIs, the schemas are  
 158 divided into separate files, one per API. In addition, types shared by the Reading API  
 159 and Writing API are defined in a separate file that is shared by the main file for the  
 160 Reading and Writing APIs, and the five main schema files import the standard EPCglobal  
 161 schema file.

162 All schemas are defined in the same XML namespace, which is  
 163 urn:epcglobal:ale:xsd:1.

164 The seven schema files are summarized in the table below:

| Schema     | Section | Imported Schemas        | Top-level Elements   |
|------------|---------|-------------------------|--|
| EPCglobal  | 3.3     | (none)                  | (none)   |
| ale-common | 3.4     | EPCglobal               | (none)   |
| ale        | 3.5     | EPCglobal<br>ale-common | ECSpec<br>ECReports  |
| alecc      | 3.6     | EPCglobal<br>ale-common | CCSpec<br>CCParameterList<br>CCReports<br>EPCCacheSpec<br>EPCPatternList<br>AssocTableSpec<br>AssocTableEntryList<br>RNGSpec |
| aletm      | 3.7     | EPCglobal               | TMSpec   |
| alelr      | 3.8     | EPCglobal               | LRSpec<br>LRProperty   |
| aleac      | 3.9     | EPCglobal               | ACPermission<br>ACRole<br>ACClientIdentity   |

165

166 **3.2 Extensibility Mechanism**

167 The XML schema in this section implements the <<extension point>> given in  
 168 the UML using a methodology described in [XMLVersioning]. This methodology  
 169 provides for both vendor extension, and for extension by EPCglobal in future versions of  
 170 this specification or in supplemental specifications. Extensions introduced through this  
 171 mechanism will be *backward compatible*, in that documents conforming to older versions  
 172 of the schema will also conform to newer versions of the standard schema and to schema  
 173 containing vendor-specific extensions. Extensions will also be *forward compatible*, in  
 174 that documents that contain vendor extensions or that conform to newer versions of the  
 175 standard schema will also conform to older versions of the schema.

176 When a document contains extensions (vendor-specific or standardized in newer versions  
177 of schema), it may conform to more than one schema. For example, a document  
178 containing vendor extensions to the ALE 1.1 schema will conform both to the EPCglobal  
179 ALE 1.1 schema and to a vendor-specific schema that includes the vendor extensions. In  
180 this example, when the document is parsed using the standard schema there will be no  
181 validation of the extension elements and attributes, but when the document is parsed  
182 using the vendor-specific schema the extensions will be validated. Similarly, a document  
183 containing new features introduced in the ALE 1.1 schema will conform both to the ALE  
184 1.0 schema and to the ALE 1.1 schema, but validation of the new features will only be  
185 available using the ALE 1.1 schema.

186 The UML for ALE consists of three kinds of definitions: classes, interfaces, and  
187 enumerations. Extensibility for each of these is provided via a different XSD-level  
188 mechanism, as defined below.

### 189 3.2.1 Extensibility Design Rules – Classes

190 The design rules for the extensibility pattern used for classes are based on  
191 [XMLVersioning]. The design rules are as follows:

- 192 • For each UML class in which <<extension point>> occurs, include an  
193 `xsd:anyAttribute` declaration in the corresponding XSD type. This declaration  
194 provides for the addition of new attributes, either in subsequent versions of the  
195 standard schema or in vendor-specific schema.
- 196 • For each UML class in which <<extension point>> occurs, include an  
197 optional (`minOccurs = 0`) element named `extension` in the corresponding XSD  
198 type. The type declared for the `extension` element will always be as follows:

```
199     <xsd:sequence>  
200         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"  
201             namespace="##local"/>  
202     </xsd:sequence>  
203     <xsd:anyAttribute processContents="lax"/>
```

204 This declaration provides for forward-compatibility with new elements introduced  
205 into subsequent versions of the standard schema.

- 206 • For each UML class in which <<extension point>> occurs, include at the end  
207 of the element list within the corresponding XSD type a declaration

```
208     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"  
209         namespace="##other"/>
```

210 This declaration provides for forward-compatibility with new elements introduced in  
211 vendor-specific schema.

212 The rules for adding vendor-specific extensions to the schema are as follows:

- 213 • Vendor-specific attributes may be added to any XSD type corresponding to a UML  
214 class in which <<extension point>> occurs. Vendor-specific attributes  
215 SHALL NOT be in the EPCglobal ALE namespace  
216 (`urn:epcglobal:ale:xsd:1`). Vendor-specific attributes SHALL be in a  
217 namespace whose namespace URI has the vendor as the owning authority. (In

218 schema parlance, this means that all vendor-specific attributes must have  
219 qualified as their form.) For example, the namespace URI may be an HTTP  
220 URL whose authority portion is a domain name owned by the vendor, a URN having  
221 a URN namespace identifier issued to the vendor by IANA, an OID URN whose  
222 initial path is a Private Enterprise Number assigned to the vendor, etc. Declarations  
223 of vendor-specific attributes SHALL specify use="optional".

- 224 • Vendor-specific elements may be added to any XSD type corresponding to a UML  
225 class in which <<extension point>> occurs. Vendor-specific elements  
226 SHALL NOT be in the EPCglobal ALE namespace  
227 (urn:epcglobal:ale:xsd:1). Vendor-specific elements SHALL be in a  
228 namespace whose namespace URI has the vendor as the owning authority (as  
229 described above). (In schema parlance, this means that all vendor-specific elements  
230 must have qualified as their form.)

231 To create a schema that contains vendor extensions, replace the <xsd:any ...  
232 namespace="##other" /> declaration with a content group reference to a group  
233 defined in the vendor namespace; e.g., <xsd:group  
234 ref="vendor:VendorExtension">. In the schema file defining elements for  
235 the vendor namespace, define a content group using a declaration of the following  
236 form:

```
237 <xsd:group name="VendorExtension">  
238   <xsd:sequence>  
239     <!--  
240       Definitions or references to vendor elements  
241       go here. Each SHALL specify minOccurs="0".  
242     -->  
243     <xsd:any processContents="lax"  
244       minOccurs="0" maxOccurs="unbounded"  
245       namespace="##other" />  
246   </xsd:sequence>  
247 </xsd:group>
```

248 (In the foregoing illustrations, vendor and VendorExtension may be any  
249 strings the vendor chooses.)

250 *Explanation (non-normative): Because vendor-specific elements must be optional,*  
251 *including references to their definitions directly into the ALE schema would violate the*  
252 *XML Schema Unique Particle Attribution constraint, because the <xsd:any ...>*  
253 *element in the ALE schema can also match vendor-specific elements. Moving the*  
254 *<xsd:any ...> into the vendor's schema avoids this problem, because ##other in*  
255 *that schema means "match an element that has a namespace other than the vendor's*  
256 *namespace." This does not conflict with standard elements, because the element form*  
257 *default for the standard ALE schema is unqualified, and hence the ##other in the*  
258 *vendor's schema does not match standard ALE elements, either.*

259 The rules for adding attributes or elements to future versions of the ALE standard schema  
260 are as follows:

- 261 • Standard attributes may be added to any XSD type corresponding to a UML class in  
262 which <<extension point>> occurs. Standard attributes SHALL NOT be in  
263 any namespace, and SHALL NOT conflict with any existing standard attribute name.
- 264 • Standard elements may be added to any XSD type corresponding to a UML class in  
265 which <<extension point>> occurs. New elements are added using the  
266 following rules:
- 267 • Find the innermost extension element type.
- 268 • Replace the <xsd:any ... namespace="##local"/> declaration with (a)  
269 new elements (which SHALL NOT be in any namespace); followed by (b) a new  
270 extension element whose type is constructed as described before. In  
271 subsequent revisions of the standard schema, new standard elements will be added  
272 within this new extension element rather than within this one.

273 *Explanation (non-normative): the reason that new standard attributes and elements are*  
274 *specified above not to be in any namespace is to be consistent with the ALE schema's*  
275 *attribute and element form default of unqualified.*

276 As noted in Section 6.1, in the schema for the ALE Reading API there are already several  
277 instances of this pattern being applied to introduce new features in ALE 1.1.

### 278 **3.2.2 Extensibility Design Rules – Interfaces**

279 All five ALE APIs include an <<extension point>> in the UML for the respective  
280 interfaces. The extension point for interfaces is implemented simply by noting that it is  
281 possible to add additional methods to the WSDL without affecting the existing methods.

282 A vendor implementation MAY add additional methods to an ALE API, provided that the  
283 name of a vendor extension method SHALL NOT conflict with existing methods.

284 Future versions of the ALE specification may add additional methods to the APIs. For  
285 back-compatibility, no incompatible changes will be made to existing methods.

### 286 **3.2.3 Extensibility Design Rules – Enumerations**

287 Enumerated types having an <<extension point>> in their UML are made  
288 extensible in the following manner. In the schema, such types are declared as equivalent  
289 to xsd:string. This provides for forward compatibility, as any vendor extension  
290 value or value defined in a future version of the specification will be valid according to  
291 the schema. A comment in the schema specifies the values that are defined in this  
292 version of the specification.

293 An implementation MAY extend an enumeration by allowing additional values beyond  
294 those defined in the specification. Vendor extension values SHALL take the form of  
295 absolute URIs [URI], where the URI has the vendor as the owning authority. For  
296 example, the URI for a vendor extension enumeration value may be an HTTP URL  
297 whose authority portion is a domain name owned by the vendor, a URN having a URN

298 namespace identifier issued to the vendor by IANA, an OID URN whose initial path is a  
299 Private Enterprise Number assigned to the vendor, etc.

300 Future versions of the ALE specification may add additional values to enumerations.  
301 Any enumeration value specified in this or future versions of the specification will be  
302 strings not containing a colon (:) character, and therefore will never clash with vendor  
303 extension values that take the form of absolute URIs.

### 304 3.3 EPCglobal Base Schema

305 The XML schemas for all ALE APIs make reference to the EPCglobal Base Schema.  
306 This schema is reproduced below.

```
307 <xsd:schema targetNamespace="urn:epcglobal:xsd:1"  
308           xmlns:epcglobal="urn:epcglobal:xsd:1"  
309           xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
310           elementFormDefault="unqualified"  
311           attributeFormDefault="unqualified"  
312           version="1.0">  
313   <xsd:annotation>  
314     <xsd:documentation>  
315       <epcglobal:copyright>Copyright (C) 2004 Epcglobal Inc., All Rights  
316       Reserved.</epcglobal:copyright>  
317       <epcglobal:disclaimer>EPCglobal Inc., its members, officers, directors, employees,  
318       or agents shall not be liable for any injury, loss, damages, financial or otherwise,  
319       arising from, related to, or caused by the use of this document. The use of said  
320       document shall constitute your express consent to the foregoing  
321       exculpation.</epcglobal:disclaimer>  
322       <epcglobal:specification>EPCglobal common components Version  
323       1.0</epcglobal:specification>  
324     </xsd:documentation>  
325   </xsd:annotation>  
326   <xsd:complexType name="Document" abstract="true">  
327     <xsd:annotation>  
328       <xsd:documentation xml:lang="en">  
329         EPCglobal document properties for all messages.  
330       </xsd:documentation>  
331     </xsd:annotation>  
332     <xsd:attribute name="schemaVersion" type="xsd:decimal" use="required">  
333       <xsd:annotation>  
334         <xsd:documentation xml:lang="en">  
335           The version of the schema corresponding to which the instance conforms.  
336         </xsd:documentation>  
337       </xsd:annotation>  
338     </xsd:attribute>  
339     <xsd:attribute name="creationDate" type="xsd:dateTime" use="required">  
340       <xsd:annotation>  
341         <xsd:documentation xml:lang="en">  
342           The date the message was created. Used for auditing and logging.  
343         </xsd:documentation>  
344       </xsd:annotation>  
345     </xsd:attribute>  
346   </xsd:complexType>  
347   <xsd:complexType name="EPC">  
348     <xsd:annotation>  
349       <xsd:documentation xml:lang="en">  
350         EPC represents the Electronic Product Code.  
351       </xsd:documentation>  
352     </xsd:annotation>  
353     <xsd:simpleContent>  
354       <xsd:extension base="xsd:string"/>  
355     </xsd:simpleContent>  
356   </xsd:complexType>  
357 </xsd:schema>
```

### 3.4 Schema for Datatypes Common to the Reading API and Writing API

The following is an XML Schema (XSD) defining common data types used in both the Reading API and the Writing API.

*Explanation (non-normative): Because an implementation MAY support only the Reading API, only the Writing API, or both, it is more convenient to have the common data types in a separate schema file that is imported by the main schema files for the Reading and Writing APIs. In this way, an implementation supporting only the Reading API need only concern itself with the schema file in this section together with the main Reading API schema file (similarly for an implementation that supports only the Writing API), and an implementation that implements both the Reading API and the Writing API can use all three files with no duplication.*

```
370 <?xml version="1.0" encoding="UTF-8"?>
371 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
372           xmlns:ale="urn:epcglobal:ale:xsd:1"
373           targetNamespace="urn:epcglobal:ale:xsd:1" elementFormDefault="unqualified"
374           attributeFormDefault="unqualified" version="1.0">
375   <!-- COMMON ELEMENTS -->
376
377   <!-- COMMON TYPES -->
378
379   <xsd:complexType name="ECFieldSpec">
380     <xsd:sequence>
381       <xsd:element name="fieldname" type="xsd:string"/>
382       <xsd:element name="datatype" type="xsd:string" minOccurs="0"/>
383       <xsd:element name="format" type="xsd:string" minOccurs="0"/>
384       <xsd:element name="extension" type="ale:ECFieldSpecExtension" minOccurs="0"/>
385       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
386             namespace="##other"/>
387     </xsd:sequence>
388     <xsd:anyAttribute processContents="lax"/>
389   </xsd:complexType>
390
391   <xsd:complexType name="ECFieldSpecExtension">
392     <xsd:sequence>
393       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
394             namespace="##local"/>
395     </xsd:sequence>
396     <xsd:anyAttribute processContents="lax"/>
397   </xsd:complexType>
398
399   <xsd:complexType name="ECFilterListMember">
400     <xsd:sequence>
401       <xsd:element name="includeExclude" type="ale:ECIncludeExclude"/>
402       <xsd:element name="fieldspec" type="ale:ECFieldSpec"/>
403       <xsd:element name="patList">
404         <xsd:complexType>
405           <xsd:sequence>
406             <xsd:element name="pat" type="xsd:string" maxOccurs="unbounded"/>
407           </xsd:sequence>
408         </xsd:complexType>
409       </xsd:element>
410       <xsd:element name="extension" type="ale:ECFilterListMemberExtension"
411             minOccurs="0"/>
412       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
413             namespace="##other"/>
414     </xsd:sequence>
415     <xsd:anyAttribute processContents="lax"/>
416   </xsd:complexType>
417
418   <xsd:complexType name="ECFilterListMemberExtension">
419     <xsd:sequence>
```

```

420     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
421         namespace="##local" />
422   </xsd:sequence>
423   <xsd:anyAttribute processContents="lax" />
424 </xsd:complexType>
425
426 <!-- The ECIncludeExclude type is an enumerated type.
427       The following strings are legal values for this type:
428         INCLUDE
429         EXCLUDE
430       -->
431 <xsd:simpleType name="ECIncludeExclude">
432   <xsd:restriction base="xsd:string" />
433 </xsd:simpleType>
434
435 <xsd:complexType name="ECReaderStat">
436   <xsd:sequence>
437     <xsd:element name="readerName" type="xsd:string" />
438     <xsd:element name="sightings" minOccurs="0">
439       <xsd:complexType>
440         <xsd:sequence>
441           <xsd:element name="sighting" type="ale:ECsightingStat" minOccurs="0"
442             maxOccurs="unbounded" />
443         </xsd:sequence>
444       </xsd:complexType>
445     </xsd:element>
446   </xsd:sequence>
447 </xsd:complexType>
448
449 <xsd:complexType name="ECsightingStat" />
450
451 <xsd:complexType name="ECTime">
452   <xsd:simpleContent>
453     <xsd:extension base="xsd:long">
454       <xsd:attribute name="unit" type="ale:ECTimeUnit" use="required" />
455     </xsd:extension>
456   </xsd:simpleContent>
457 </xsd:complexType>
458
459 <!-- The ECTimeUnit type is an enumerated type.
460       The following strings are legal values for this type:
461         MS
462       An implementation may also recognize additional strings as extensions.
463       -->
464 <xsd:simpleType name="ECTimeUnit">
465   <xsd:restriction base="xsd:string" />
466 </xsd:simpleType>
467
468 <xsd:simpleType name="ECTrigger">
469   <xsd:restriction base="xsd:string" />
470 </xsd:simpleType>
471 </xsd:schema>

```

### 472 3.5 ALE Reading API Schema

473 The following is an XML Schema (XSD) for the ALE Reading API, defining both  
474 ECSpec and ECReports as top level elements.

```

475 <?xml version="1.0" encoding="UTF-8"?>
476 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
477     xmlns:ale="urn:epcglobal:ale:xsd:1"
478     targetNamespace="urn:epcglobal:ale:xsd:1"
479     xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
480     attributeFormDefault="unqualified" version="1.0">
481   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd" />
482   <xsd:include schemaLocation="EPCglobal-ale-1_1-common.xsd" />
483   <!-- ALE ELEMENTS -->
484   <xsd:element name="ECSpec" type="ale:ECSpec" />

```

```

485 <xsd:element name="ECReports" type="ale:ECReports"/>
486
487 <!-- ALE TYPES -->
488
489 <xsd:complexType name="ECBoundarySpec">
490   <xsd:sequence>
491     <xsd:element name="startTrigger" type="ale:ECTrigger" minOccurs="0"/>
492     <xsd:element name="repeatPeriod" type="ale:ECTime" minOccurs="0"/>
493     <xsd:element name="stopTrigger" type="ale:ECTrigger" minOccurs="0"/>
494     <xsd:element name="duration" type="ale:ECTime" minOccurs="0"/>
495     <xsd:element name="stableSetInterval" type="ale:ECTime" minOccurs="0"/>
496     <xsd:element name="extension" type="ale:ECBoundarySpecExtension" minOccurs="0"/>
497     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
498       namespace="##other"/>
499   </xsd:sequence>
500   <xsd:anyAttribute processContents="lax"/>
501 </xsd:complexType>
502
503 <xsd:complexType name="ECBoundarySpecExtension">
504   <xsd:sequence>
505     <xsd:element name="startTriggerList" minOccurs="0">
506       <xsd:complexType>
507         <xsd:sequence>
508           <xsd:element name="startTrigger" type="ale:ECTrigger" minOccurs="0"
509             maxOccurs="unbounded"/>
510         </xsd:sequence>
511       </xsd:complexType>
512     </xsd:element>
513     <xsd:element name="stopTriggerList" minOccurs="0">
514       <xsd:complexType>
515         <xsd:sequence>
516           <xsd:element name="stopTrigger" type="ale:ECTrigger" minOccurs="0"
517             maxOccurs="unbounded"/>
518         </xsd:sequence>
519       </xsd:complexType>
520     </xsd:element>
521     <xsd:element name="whenDataAvailable" type="xsd:boolean" minOccurs="0"/>
522     <xsd:element name="extension" type="ale:ECBoundarySpecExtension2" minOccurs="0"/>
523   </xsd:sequence>
524   <xsd:anyAttribute processContents="lax"/>
525 </xsd:complexType>
526
527 <xsd:complexType name="ECBoundarySpecExtension2">
528   <xsd:sequence>
529     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
530       namespace="##local"/>
531   </xsd:sequence>
532   <xsd:anyAttribute processContents="lax"/>
533 </xsd:complexType>
534
535 <xsd:complexType name="ECFilterSpec">
536   <xsd:sequence>
537     <xsd:element name="includePatterns" minOccurs="0">
538       <xsd:complexType>
539         <xsd:sequence>
540           <xsd:element name="includePattern" type="xsd:string" minOccurs="0"
541             maxOccurs="unbounded"/>
542         </xsd:sequence>
543       </xsd:complexType>
544     </xsd:element>
545     <xsd:element name="excludePatterns" minOccurs="0">
546       <xsd:complexType>
547         <xsd:sequence>
548           <xsd:element name="excludePattern" type="xsd:string" minOccurs="0"
549             maxOccurs="unbounded"/>
550         </xsd:sequence>
551       </xsd:complexType>
552     </xsd:element>
553     <xsd:element name="extension" type="ale:ECFilterSpecExtension" minOccurs="0"/>
554     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"

```

```

555         namespace="##other" />
556     </xsd:sequence>
557 <xsd:anyAttribute processContents="lax" />
558 </xsd:complexType>
559
560 <xsd:complexType name="ECFilterSpecExtension">
561     <xsd:sequence>
562         <xsd:element name="filterList" minOccurs="0">
563             <xsd:complexType>
564                 <xsd:sequence>
565                     <xsd:element name="filter" type="ale:ECFilterListMember" minOccurs="0"
566                         maxOccurs="unbounded" />
567                 </xsd:sequence>
568             </xsd:complexType>
569         </xsd:element>
570         <xsd:element name="extension" type="ale:ECFilterSpecExtension2" minOccurs="0" />
571     </xsd:sequence>
572     <xsd:anyAttribute processContents="lax" />
573 </xsd:complexType>
574
575 <xsd:complexType name="ECFilterSpecExtension2">
576     <xsd:sequence>
577         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
578             namespace="##local" />
579     </xsd:sequence>
580     <xsd:anyAttribute processContents="lax" />
581 </xsd:complexType>
582
583 <xsd:complexType name="ECGroupSpec">
584     <xsd:sequence>
585         <xsd:element name="pattern" type="xsd:string" minOccurs="0" maxOccurs="unbounded" />
586         <xsd:element name="extension" type="ale:ECGroupSpecExtension" minOccurs="0" />
587         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
588             namespace="##other" />
589     </xsd:sequence>
590     <xsd:anyAttribute processContents="lax" />
591 </xsd:complexType>
592
593 <xsd:complexType name="ECGroupSpecExtension">
594     <xsd:sequence>
595         <xsd:element name="fieldspec" type="ale:ECFieldSpec" minOccurs="0" />
596         <xsd:element name="extension" type="ale:ECGroupSpecExtension2" minOccurs="0" />
597     </xsd:sequence>
598     <xsd:anyAttribute processContents="lax" />
599 </xsd:complexType>
600
601 <xsd:complexType name="ECGroupSpecExtension2">
602     <xsd:sequence>
603         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
604             namespace="##local" />
605     </xsd:sequence>
606     <xsd:anyAttribute processContents="lax" />
607 </xsd:complexType>
608
609 <!-- The ECInitiationCondition type is an enumerated type.
610     The following strings are legal values for this type:
611         TRIGGER
612         REPEAT_PERIOD
613         REQUESTED
614         UNDEFINE
615     An implementation may also recognize additional strings as extensions.
616     -->
617 <xsd:simpleType name="ECInitiationCondition">
618     <xsd:restriction base="xsd:string" />
619 </xsd:simpleType>
620
621 <xsd:complexType name="ECReport">
622     <xsd:sequence>
623         <xsd:element name="group" type="ale:ECReportGroup" minOccurs="0"
624             maxOccurs="unbounded" />

```

```

625     <xsd:element name="extension" type="ale:ECReportExtension" minOccurs="0"/>
626     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
627         namespace="##other"/>
628 </xsd:sequence>
629 <xsd:attribute name="reportName" type="xsd:string" use="required"/>
630 <xsd:anyAttribute processContents="lax"/>
631 </xsd:complexType>
632
633 <xsd:complexType name="ECReportExtension">
634 <xsd:sequence>
635     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
636         namespace="##local"/>
637 </xsd:sequence>
638 <xsd:anyAttribute processContents="lax"/>
639 </xsd:complexType>
640
641 <xsd:complexType name="ECReportGroup">
642 <xsd:sequence>
643     <xsd:element name="groupList" type="ale:ECReportGroupList" minOccurs="0"/>
644     <xsd:element name="groupCount" type="ale:ECReportGroupCount" minOccurs="0"/>
645     <xsd:element name="extension" type="ale:ECReportGroupExtension" minOccurs="0"/>
646     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
647         namespace="##other"/>
648 </xsd:sequence>
649 <!-- The groupName attribute SHALL be omitted to indicate the default group. -->
650 <xsd:attribute name="groupName" type="xsd:string" use="optional"/>
651 <xsd:anyAttribute processContents="lax"/>
652 </xsd:complexType>
653
654 <xsd:complexType name="ECReportGroupExtension">
655 <xsd:sequence>
656     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
657         namespace="##local"/>
658 </xsd:sequence>
659 <xsd:anyAttribute processContents="lax"/>
660 </xsd:complexType>
661
662 <xsd:complexType name="ECReportGroupCount">
663 <xsd:sequence>
664     <xsd:element name="count" type="xsd:int"/>
665     <xsd:element name="extension" type="ale:ECReportGroupCountExtension"
666         minOccurs="0"/>
667     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
668         namespace="##other"/>
669 </xsd:sequence>
670 <xsd:anyAttribute processContents="lax"/>
671 </xsd:complexType>
672
673 <xsd:complexType name="ECReportGroupCountExtension">
674 <xsd:sequence>
675     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
676         namespace="##local"/>
677 </xsd:sequence>
678 <xsd:anyAttribute processContents="lax"/>
679 </xsd:complexType>
680
681 <xsd:complexType name="ECReportGroupList">
682 <xsd:sequence>
683     <xsd:element name="member" type="ale:ECReportGroupListMember" minOccurs="0"
684         maxOccurs="unbounded"/>
685     <xsd:element name="extension" type="ale:ECReportGroupListExtension" minOccurs="0"/>
686     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
687         namespace="##other"/>
688 </xsd:sequence>
689 <xsd:anyAttribute processContents="lax"/>
690 </xsd:complexType>
691
692 <xsd:complexType name="ECReportGroupListExtension">
693 <xsd:sequence>
694     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"

```

```

695         namespace="##local" />
696     </xsd:sequence>
697     <xsd:anyAttribute processContents="lax" />
698 </xsd:complexType>
699
700 <xsd:complexType name="ECReportGroupListMember">
701     <xsd:sequence>
702         <!-- Each of the following four elements SHALL be omitted if null. -->
703         <xsd:element name="epc" type="epcglobal:EPC" minOccurs="0" />
704         <xsd:element name="tag" type="epcglobal:EPC" minOccurs="0" />
705         <xsd:element name="rawHex" type="epcglobal:EPC" minOccurs="0" />
706         <xsd:element name="rawDecimal" type="epcglobal:EPC" minOccurs="0" />
707         <xsd:element name="extension" type="ale:ECReportGroupListMemberExtension"
708             minOccurs="0" />
709         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
710             namespace="##other" />
711     </xsd:sequence>
712     <xsd:anyAttribute processContents="lax" />
713 </xsd:complexType>
714
715 <xsd:complexType name="ECReportGroupListMemberExtension">
716     <xsd:sequence>
717         <xsd:element name="fieldList" minOccurs="0">
718             <xsd:complexType>
719                 <xsd:sequence>
720                     <xsd:element name="field" type="ale:ECReportMemberField" minOccurs="0"
721                         maxOccurs="unbounded" />
722                 </xsd:sequence>
723             </xsd:complexType>
724         </xsd:element>
725         <xsd:element name="stats" minOccurs="0">
726             <xsd:complexType>
727                 <xsd:sequence>
728                     <xsd:element name="stat" type="ale:ECTagStat" minOccurs="0"
729                         maxOccurs="unbounded" />
730                 </xsd:sequence>
731             </xsd:complexType>
732         </xsd:element>
733         <xsd:element name="extension" type="ale:ECReportGroupListMemberExtension2"
734             minOccurs="0" />
735     </xsd:sequence>
736     <xsd:anyAttribute processContents="lax" />
737 </xsd:complexType>
738
739 <xsd:complexType name="ECReportGroupListMemberExtension2">
740     <xsd:sequence>
741         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
742             namespace="##local" />
743     </xsd:sequence>
744     <xsd:anyAttribute processContents="lax" />
745 </xsd:complexType>
746
747 <xsd:complexType name="ECReportMemberField">
748     <xsd:sequence>
749         <xsd:element name="name" type="xsd:string" />
750         <xsd:element name="value" type="xsd:string" minOccurs="0" />
751         <xsd:element name="fieldspec" type="ale:ECFieldSpec" minOccurs="0" />
752         <xsd:element name="extension" type="ale:ECReportMemberFieldExtension"
753             minOccurs="0" />
754         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
755             namespace="##other" />
756     </xsd:sequence>
757     <xsd:anyAttribute processContents="lax" />
758 </xsd:complexType>
759
760 <xsd:complexType name="ECReportMemberFieldExtension">
761     <xsd:sequence>
762         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
763             namespace="##local" />
764 </xsd:sequence>

```

```

765     <xsd:anyAttribute processContents="lax" />
766 </xsd:complexType>
767
768 <xsd:complexType name="ECReportOutputFieldSpec">
769   <xsd:sequence>
770     <xsd:element name="fieldspec" type="ale:ECFieldSpec" />
771     <xsd:element name="name" type="xsd:string" minOccurs="0" />
772     <xsd:element name="includeFieldSpecInReport" type="xsd:boolean" minOccurs="0" />
773     <xsd:element name="extension" type="ale:ECReportOutputFieldSpecExtension"
774       minOccurs="0" />
775     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
776       namespace="##other" />
777   </xsd:sequence>
778   <xsd:anyAttribute processContents="lax" />
779 </xsd:complexType>
780
781 <xsd:complexType name="ECReportOutputFieldSpecExtension">
782   <xsd:sequence>
783     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
784       namespace="##local" />
785   </xsd:sequence>
786   <xsd:anyAttribute processContents="lax" />
787 </xsd:complexType>
788
789 <xsd:complexType name="ECReportOutputSpec">
790   <xsd:sequence>
791     <xsd:element name="extension" type="ale:ECReportOutputSpecExtension"
792       minOccurs="0" />
793     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
794       namespace="##other" />
795   </xsd:sequence>
796   <xsd:attribute name="includeEPC" type="xsd:boolean" default="false" />
797   <xsd:attribute name="includeTag" type="xsd:boolean" default="false" />
798   <xsd:attribute name="includeRawHex" type="xsd:boolean" default="false" />
799   <xsd:attribute name="includeRawDecimal" type="xsd:boolean" default="false" />
800   <xsd:attribute name="includeCount" type="xsd:boolean" default="false" />
801   <xsd:anyAttribute processContents="lax" />
802 </xsd:complexType>
803
804 <xsd:complexType name="ECReportOutputSpecExtension">
805   <xsd:sequence>
806     <xsd:element name="fieldList" minOccurs="0">
807       <xsd:complexType>
808         <xsd:sequence>
809           <xsd:element name="field" type="ale:ECReportOutputFieldSpec" minOccurs="0"
810             maxOccurs="unbounded" />
811         </xsd:sequence>
812       </xsd:complexType>
813     </xsd:element>
814     <xsd:element name="extension" type="ale:ECReportOutputSpecExtension2"
815       minOccurs="0" />
816   </xsd:sequence>
817   <xsd:anyAttribute processContents="lax" />
818 </xsd:complexType>
819
820 <xsd:complexType name="ECReportOutputSpecExtension2">
821   <xsd:sequence>
822     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
823       namespace="##local" />
824   </xsd:sequence>
825   <xsd:anyAttribute processContents="lax" />
826 </xsd:complexType>
827
828 <xsd:complexType name="ECReports">
829   <xsd:complexContent>
830     <xsd:extension base="epcglobal:Document">
831       <xsd:sequence>
832         <xsd:element name="reports">
833           <xsd:complexType>
834             <xsd:sequence>

```

```

835         <xsd:element name="report" type="ale:ECReport" minOccurs="0"
836             maxOccurs="unbounded" />
837     </xsd:sequence>
838 </xsd:complexType>
839 </xsd:element>
840 <xsd:element name="extension" type="ale:ECReportsExtension" minOccurs="0" />
841 <xsd:element name="ECSpec" type="ale:ECSpec" minOccurs="0" />
842 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
843     namespace="##other" />
844 </xsd:sequence>
845 <xsd:attribute name="specName" type="xsd:string" use="required" />
846 <xsd:attribute name="date" type="xsd:dateTime" use="required" />
847 <xsd:attribute name="ALEID" type="xsd:string" use="required" />
848 <xsd:attribute name="totalMilliseconds" type="xsd:long" use="required" />
849 <xsd:attribute name="initiationCondition" type="ale:ECInitiationCondition"
850     use="optional" />
851 <xsd:attribute name="initiationTrigger" type="ale:ECTrigger" use="optional" />
852 <xsd:attribute name="terminationCondition" type="ale:ECTerminationCondition"
853     use="required" />
854 <xsd:attribute name="terminationTrigger" type="ale:ECTrigger" use="optional" />
855 <xsd:attribute name="schemaURL" type="xsd:string" use="optional" />
856 <xsd:anyAttribute processContents="lax" />
857 </xsd:extension>
858 </xsd:complexContent>
859 </xsd:complexType>
860
861 <xsd:complexType name="ECReportsExtension">
862     <xsd:sequence>
863         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
864             namespace="##local" />
865     </xsd:sequence>
866     <xsd:anyAttribute processContents="lax" />
867 </xsd:complexType>
868
869 <!-- The ECReportSetEnum type is an enumerated type.
870     The following strings are legal values for this type:
871     CURRENT
872     ADDITIONS
873     DELETIONS
874     An implementation may also recognize additional strings as extensions.
875     -->
876 <xsd:simpleType name="ECReportSetEnum">
877     <xsd:restriction base="xsd:string" />
878 </xsd:simpleType>
879
880 <xsd:complexType name="ECReportSetSpec">
881     <xsd:attribute name="set" type="ale:ECReportSetEnum" use="required" />
882 </xsd:complexType>
883
884 <xsd:complexType name="ECReportSpec">
885     <xsd:sequence>
886         <xsd:element name="reportSet" type="ale:ECReportSetSpec" />
887         <xsd:element name="filterSpec" type="ale:ECFilterSpec" minOccurs="0" />
888         <xsd:element name="groupSpec" type="ale:ECGroupSpec" minOccurs="0" />
889         <xsd:element name="output" type="ale:ECReportOutputSpec" />
890         <xsd:element name="extension" type="ale:ECReportSpecExtension" minOccurs="0" />
891         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
892             namespace="##other" />
893     </xsd:sequence>
894     <xsd:attribute name="reportName" type="xsd:string" use="required" />
895     <xsd:attribute name="reportIfEmpty" type="xsd:boolean" default="false" />
896     <xsd:attribute name="reportOnlyOnChange" type="xsd:boolean" default="false" />
897     <xsd:anyAttribute processContents="lax" />
898 </xsd:complexType>
899
900 <xsd:complexType name="ECReportSpecExtension">
901     <xsd:sequence>
902         <xsd:element name="statProfileNames" minOccurs="0">
903             <xsd:complexType>
904                 <xsd:sequence>

```

```

905         <xsd:element name="statProfileName" type="ale:ECStatProfileName"
906             minOccurs="0" maxOccurs="unbounded" />
907     </xsd:sequence>
908 </xsd:complexType>
909 </xsd:element>
910 <xsd:element name="extension" type="ale:ECReportSpecExtension2" minOccurs="0" />
911 </xsd:sequence>
912 <xsd:anyAttribute processContents="lax" />
913 </xsd:complexType>
914
915 <xsd:complexType name="ECReportSpecExtension2">
916     <xsd:sequence>
917         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
918             namespace="##local" />
919     </xsd:sequence>
920     <xsd:anyAttribute processContents="lax" />
921 </xsd:complexType>
922
923 <xsd:complexType name="ECSpec">
924     <xsd:complexContent>
925         <xsd:extension base="epcglobal:Document">
926             <xsd:sequence>
927                 <xsd:element name="logicalReaders">
928                     <xsd:complexType>
929                         <xsd:sequence>
930                             <xsd:element name="logicalReader" type="xsd:string"
931                                 maxOccurs="unbounded" />
932                         </xsd:sequence>
933                     </xsd:complexType>
934                 </xsd:element>
935                 <xsd:element name="boundarySpec" type="ale:ECBoundarySpec" />
936                 <xsd:element name="reportSpecs">
937                     <xsd:complexType>
938                         <xsd:sequence>
939                             <xsd:element name="reportSpec" type="ale:ECReportSpec"
940                                 maxOccurs="unbounded" />
941                         </xsd:sequence>
942                     </xsd:complexType>
943                 </xsd:element>
944                 <xsd:element name="extension" type="ale:ECSpecExtension" minOccurs="0" />
945                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
946                     namespace="##other" />
947             </xsd:sequence>
948             <xsd:attribute name="includeSpecInReports" type="xsd:boolean" default="false" />
949             <xsd:anyAttribute processContents="lax" />
950         </xsd:extension>
951     </xsd:complexContent>
952 </xsd:complexType>
953
954 <xsd:complexType name="ECSpecExtension">
955     <xsd:sequence>
956         <xsd:element name="primaryKeyFields" minOccurs="0">
957             <xsd:complexType>
958                 <xsd:sequence>
959                     <xsd:element name="primaryKeyField" type="xsd:string" minOccurs="0"
960                         maxOccurs="unbounded" />
961                 </xsd:sequence>
962             </xsd:complexType>
963         </xsd:element>
964         <xsd:element name="extension" type="ale:ECSpecExtension2" minOccurs="0" />
965     </xsd:sequence>
966     <xsd:anyAttribute processContents="lax" />
967 </xsd:complexType>
968
969 <xsd:complexType name="ECSpecExtension2">
970     <xsd:sequence>
971         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
972             namespace="##local" />
973     </xsd:sequence>
974     <xsd:anyAttribute processContents="lax" />

```

```

975     </xsd:complexType>
976
977     <!-- The ECStatProfileName type is an enumerated type.
978         The following strings are legal values for this type:
979         TagTimestamps
980         An implementation may also recognize additional strings as extensions.
981         -->
982     <xsd:simpleType name="ECStatProfileName">
983         <xsd:restriction base="xsd:string"/>
984     </xsd:simpleType>
985
986     <xsd:complexType name="ECTagStat">
987         <xsd:sequence>
988             <xsd:element name="profile" type="ale:ECStatProfileName"/>
989             <xsd:element name="statBlocks" minOccurs="0">
990                 <xsd:complexType>
991                     <xsd:sequence>
992                         <xsd:element name="statBlock" type="ale:ECReaderStat" minOccurs="0"
993                             maxOccurs="unbounded"/>
994                     </xsd:sequence>
995                 </xsd:complexType>
996             </xsd:element>
997         </xsd:sequence>
998     </xsd:complexType>
999
1000     <xsd:complexType name="ECTagTimestampStat">
1001         <xsd:complexContent>
1002             <xsd:extension base="ale:ECTagStat">
1003                 <xsd:sequence>
1004                     <xsd:element name="firstSightingTime" type="xsd:dateTime"/>
1005                     <xsd:element name="lastSightingTime" type="xsd:dateTime"/>
1006                 </xsd:sequence>
1007             </xsd:extension>
1008         </xsd:complexContent>
1009     </xsd:complexType>
1010
1011     <!-- The ECTerminationCondition type is an enumerated type.
1012         The following strings are legal values for this type:
1013         TRIGGER
1014         DURATION
1015         STABLE_SET
1016         DATA_AVAILABLE
1017         UNREQUEST
1018         UNDEFINE
1019         An implementation may also recognize additional strings as extensions.
1020         -->
1021     <xsd:simpleType name="ECTerminationCondition">
1022         <xsd:restriction base="xsd:string"/>
1023     </xsd:simpleType>
1024 </xsd:schema>

```

### 1025 3.5.1 ECSpec – Example 1 (non-normative)

1026 Here is an example ECSpec rendered into XML [XML1.0]:

```

1027 <?xml version="1.0" encoding="UTF-8"?>
1028
1029 <ale:ECSpec xmlns:ale="urn:epcglobal:ale:xsd:1"
1030     xmlns:epcglobal="urn:epcglobal:xsd:1"
1031     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1032     schemaVersion="1.0"
1033     creationDate="2003-08-06T10:54:06.444-05:00">
1034     <logicalReaders>
1035         <logicalReader>dock_1a</logicalReader>
1036         <logicalReader>dock_1b</logicalReader>
1037     </logicalReaders>
1038     <boundarySpec>
1039         <startTrigger>http://example.com/trigger1</startTrigger>
1040         <repeatPeriod unit="MS">20000</repeatPeriod>

```

```

1041         <stopTrigger>http://example.com/trigger2</stopTrigger>
1042         <duration unit="MS">3000</duration>
1043     </boundarySpec>
1044     <reportSpecs>
1045         <reportSpec reportName="report1">
1046             <reportSet set="CURRENT"/>
1047             <output includeTag="true"/>
1048         </reportSpec>
1049         <reportSpec reportName="report2">
1050             <reportSet set="ADDITIONS"/>
1051             <output includeCount="true"/>
1052         </reportSpec>
1053         <reportSpec reportName="report3">
1054             <reportSet set="DELETIONS"/>
1055             <groupSpec>
1056                 <pattern>urn:epc:pat:sgtin-96:X.X.X.*</pattern>
1057             </groupSpec>
1058             <output includeCount="true"/>
1059         </reportSpec>
1060     </reportSpecs>
1061 </ale:ECSpec>

```

## 1062 3.5.2 ECSpec – Example 2 (non-normative)

1063 Here is a second example ECSpec rendered into XML [XML1.0]:

```

1064 <?xml version="1.0" encoding="UTF-8"?>
1065 <ale:ECSpec xmlns:ale="urn:epcglobal:ale:xsd:1"
1066             xmlns:epcglobal="urn:epcglobal:xsd:1"
1067             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1068             schemaVersion="1.0"
1069             creationDate="2007-05-14T10:10:10+02:00">
1070     <logicalReaders>
1071         <logicalReader>dock_1a</logicalReader>
1072         <logicalReader>dock_1b</logicalReader>
1073     </logicalReaders>
1074     <boundarySpec>
1075         <repeatPeriod unit="MS">20000</repeatPeriod>
1076         <duration unit="MS">3000</duration>
1077         <extension>
1078             <startTriggerList>
1079                 <startTrigger>http://example.com/trigger1</startTrigger>
1080             </startTriggerList>
1081             <stopTriggerList>
1082                 <stopTrigger>http://example.com/trigger2</stopTrigger>
1083             </stopTriggerList>
1084         </extension>
1085     </boundarySpec>
1086     <reportSpecs>
1087         <reportSpec reportName="report1">
1088             <reportSet set="CURRENT"/>
1089             <filterSpec>
1090                 <extension>
1091                     <filterList>
1092                         <filter>
1093                             <includeExclude>INCLUDE</includeExclude>
1094                             <fieldspec>
1095                                 <fieldname>epc</fieldname>
1096                             </fieldspec>
1097                             <patList>
1098                                 <pat>urn:epc:pat:gid-96:*.*. *</pat>
1099                             </patList>
1100                         </filter>
1101                     </filterList>
1102                 </extension>
1103             </filterSpec>
1104             <output includeTag="true">
1105                 <extension>

```

```

1107         <fieldList>
1108             <field>
1109                 <fieldspec>
1110                     <fieldname>LotCode</fieldname>
1111                 </fieldspec>
1112             </field>
1113         </fieldList>
1114     </extension>
1115 </output>
1116 </reportSpec>
1117 <reportSpec reportName="report2">
1118     <reportSet set="ADDITIONS"/>
1119     <output includeCount="true"/>
1120 </reportSpec>
1121 <reportSpec reportName="report3">
1122     <reportSet set="DELETIONS"/>
1123     <groupSpec>
1124         <pattern>urn:epc:pat:sgtin-96:X.X.X.*</pattern>
1125     </groupSpec>
1126     <output includeCount="true"/>
1127 </reportSpec>
1128 </reportSpecs>
1129 </ale:ECSpec>

```

### 1130 3.5.3 ECREports – Example 1 (non-normative)

1131 Here is an example ECREports rendered into XML [XML1.0]: This ECREports  
1132 instance corresponds to the ECSpec example in Section 3.5.1, assuming that the ECSpec  
1133 was defined using the name “embarcadère”.

```

1134 <?xml version="1.0" encoding="UTF-8"?>
1135 <ale:ECREports xmlns:ale="urn:epcglobal:ale:xsd:1"
1136     xmlns:epcglobal="urn:epcglobal:xsd:1"
1137     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1138     schemaVersion="1.0"
1139     creationDate="2003-08-06T10:54:06.444-05:00"
1140     specName="embarcad&#232;re"
1141     date="2003-08-06T10:54:06.444-05:00"
1142     ALEID="Edge34"
1143     totalMilliseconds="3034"
1144     initiationCondition="TRIGGER"
1145     initiationTrigger="http://example.com/trigger1"
1146     terminationCondition="DURATION">
1147     <reports>
1148         <report reportName="report1">
1149             <group>
1150                 <groupList>
1151                     <member><tag>urn:epc:tag:gid-96:10.50.1000</tag></member>
1152                     <member><tag>urn:epc:tag:gid-96:10.50.1001</tag></member>
1153                 </groupList>
1154             </group>
1155         </report>
1156         <report reportName="report2">
1157             <group><groupCount><count>6847</count></groupCount></group>
1158         </report>
1159         <report reportName="report3">
1160             <group name="urn:epc:pat:sgtin-96:3.0037000.012345.*">
1161                 <groupCount><count>2</count></groupCount>
1162             </group>
1163             <group name="urn:epc:pat:sgtin-96:3.0037000.055555.*">
1164                 <groupCount><count>3</count></groupCount>
1165             </group>
1166             <group>
1167                 <groupCount><count>6842</count></groupCount>
1168             </group>
1169         </report>
1170     </reports>
1171 </ale:ECREports>

```

### 1172 3.5.4 ECReports – Example 2 (non-normative)

1173 Here is an example ECReports rendered into XML [XML1.0]: This ECReports  
1174 instance corresponds to the ECSpec example in Section 3.5.2, assuming the ECSpec was  
1175 defined using the name “Spec1”.

```
1176 <?xml version="1.0" encoding="UTF-8"?>
1177
1178 <ale:ECReports xmlns:ale="urn:epcglobal:ale:xsd:1"
1179               xmlns:epcglobal="urn:epcglobal:xsd:1"
1180               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1181               schemaVersion="1.0"
1182               creationDate="2007-05-14T11:11:11+02:00"
1183               specName="Spec1"
1184               date="2007-05-14T11:11:11+02:00"
1185               ALEID="Edge34"
1186               totalMilliseconds="3034"
1187               initiationCondition="TRIGGER"
1188               initiationTrigger="http://example.com/trigger1"
1189               terminationCondition="TRIGGER"
1190               terminationTrigger="http://example.com/trigger2">
1191   <reports>
1192     <report reportName="report1">
1193       <group>
1194         <groupList>
1195           <member>
1196             <tag>urn:epc:tag:gid-96:10.50.1000</tag>
1197             <extension>
1198               <fieldList>
1199                 <field>
1200                   <name>LotCode</name>
1201                   <value>12345</value>
1202                 </field>
1203               </fieldList>
1204             </extension>
1205           </member>
1206           <member>
1207             <tag>urn:epc:tag:gid-96:10.50.1001</tag>
1208             <extension>
1209               <fieldList>
1210                 <field>
1211                   <name>LotCode</name>
1212                   <value>12345</value>
1213                 </field>
1214               </fieldList>
1215             </extension>
1216           </member>
1217         </groupList>
1218       </group>
1219     </report>
1220     <report reportName="report2">
1221       <group><groupCount><count>6847</count></groupCount></group>
1222     </report>
1223     <report reportName="report3">
1224       <group name="urn:epc:pat:sgtin-96:3.0037000.012345.*">
1225         <groupCount><count>2</count></groupCount>
1226       </group>
1227       <group name="urn:epc:pat:sgtin-96:3.0037000.055555.*">
1228         <groupCount><count>3</count></groupCount>
1229       </group>
1230       <group>
1231         <groupCount><count>6842</count></groupCount>
1232       </group>
1233     </report>
1234   </reports>
1235 </ale:ECReports>
```

## 1236 3.6 ALE Writing API Schema

1237 The following is an XML Schema (XSD) for the ALE Writing API, defining CCSpec,  
1238 CCParameterList, CCReports, EPCCacheSpec, EPCPatternList,  
1239 AssocTableSpec, AssocTableEntryList, and RNGSpec as top level elements.

```
1240 <?xml version="1.0" encoding="UTF-8"?>
1241 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1242           xmlns:ale="urn:epcglobal:ale:xsd:1"
1243           targetNamespace="urn:epcglobal:ale:xsd:1"
1244           xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
1245           attributeFormDefault="unqualified" version="1.0">
1246   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd"/>
1247   <xsd:include schemaLocation="EPCglobal-ale-1_1-common.xsd"/>
1248   <!-- ALECC ELEMENTS -->
1249   <xsd:element name="CCSpec" type="ale:CCSpec"/>
1250   <xsd:element name="CCReports" type="ale:CCReports"/>
1251   <xsd:element name="EPCCacheSpec" type="ale:EPCCacheSpec"/>
1252   <xsd:element name="EPCPatternList" type="ale:EPCPatternList"/>
1253   <xsd:element name="AssocTableSpec" type="ale:AssocTableSpec"/>
1254   <xsd:element name="AssocTableEntryList" type="ale:AssocTableEntryList"/>
1255   <xsd:element name="RNGSpec" type="ale:RNGSpec"/>
1256
1257   <!-- ALECC TYPES -->
1258
1259   <xsd:complexType name="AssocTableEntry">
1260     <xsd:sequence>
1261       <xsd:element name="key" type="xsd:string"/>
1262       <xsd:element name="value" type="xsd:string"/>
1263     </xsd:sequence>
1264   </xsd:complexType>
1265
1266   <xsd:complexType name="AssocTableEntryList">
1267     <xsd:sequence>
1268       <xsd:element name="entries" minOccurs="0">
1269         <xsd:complexType>
1270           <xsd:sequence>
1271             <xsd:element name="entry" type="ale:AssocTableEntry" minOccurs="0"
1272                       maxOccurs="unbounded"/>
1273           </xsd:sequence>
1274         </xsd:complexType>
1275       </xsd:element>
1276     </xsd:sequence>
1277   </xsd:complexType>
1278
1279   <xsd:complexType name="AssocTableSpec">
1280     <xsd:complexContent>
1281       <xsd:extension base="epcglobal:Document">
1282         <xsd:sequence>
1283           <xsd:element name="datatype" type="xsd:string"/>
1284           <xsd:element name="format" type="xsd:string"/>
1285           <xsd:element name="extension" type="ale:AssocTableSpecExtension"
1286                     minOccurs="0"/>
1287           <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1288                 namespace="##other"/>
1289         </xsd:sequence>
1290       <xsd:anyAttribute processContents="lax"/>
1291     </xsd:extension>
1292   </xsd:complexContent>
1293 </xsd:complexType>
1294
1295   <xsd:complexType name="AssocTableSpecExtension">
1296     <xsd:sequence>
1297       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1298             namespace="##local"/>
1299     </xsd:sequence>
1300   <xsd:anyAttribute processContents="lax"/>
1301 </xsd:complexType>
1302
```

```

1303 <xsd:complexType name="CCBoundarySpec">
1304   <xsd:sequence>
1305     <xsd:element name="startTriggerList" minOccurs="0">
1306       <xsd:complexType>
1307         <xsd:sequence>
1308           <xsd:element name="startTrigger" type="ale:ECTrigger" minOccurs="0"
1309             maxOccurs="unbounded"/>
1310         </xsd:sequence>
1311       </xsd:complexType>
1312     </xsd:element>
1313     <xsd:element name="repeatPeriod" type="ale:ECTime" minOccurs="0"/>
1314     <xsd:element name="stopTriggerList" minOccurs="0">
1315       <xsd:complexType>
1316         <xsd:sequence>
1317           <xsd:element name="stopTrigger" type="ale:ECTrigger" minOccurs="0"
1318             maxOccurs="unbounded"/>
1319         </xsd:sequence>
1320       </xsd:complexType>
1321     </xsd:element>
1322     <xsd:element name="duration" type="ale:ECTime" minOccurs="0"/>
1323     <xsd:element name="noNewTagsInterval" type="ale:ECTime" minOccurs="0"/>
1324     <xsd:element name="tagsProcessedCount" type="xsd:int" minOccurs="0"/>
1325     <xsd:element name="afterError" type="xsd:boolean" minOccurs="0"/>
1326     <xsd:element name="extension" type="ale:CCBoundarySpecExtension" minOccurs="0"/>
1327     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1328       namespace="##other"/>
1329   </xsd:sequence>
1330   <xsd:anyAttribute processContents="lax"/>
1331 </xsd:complexType>
1332
1333 <xsd:complexType name="CCBoundarySpecExtension">
1334   <xsd:sequence>
1335     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1336       namespace="##local"/>
1337   </xsd:sequence>
1338   <xsd:anyAttribute processContents="lax"/>
1339 </xsd:complexType>
1340
1341 <xsd:complexType name="CCCmdReport">
1342   <xsd:sequence>
1343     <xsd:element name="tagReports" minOccurs="0">
1344       <xsd:complexType>
1345         <xsd:sequence>
1346           <xsd:element name="tagReport" type="ale:CCTagReport" minOccurs="0"
1347             maxOccurs="unbounded"/>
1348         </xsd:sequence>
1349       </xsd:complexType>
1350     </xsd:element>
1351     <xsd:element name="extension" type="ale:CCCmdReportExtension" minOccurs="0"/>
1352     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1353       namespace="##other"/>
1354   </xsd:sequence>
1355   <xsd:attribute name="cmdSpecName" type="xsd:string" use="required"/>
1356   <xsd:anyAttribute processContents="lax"/>
1357 </xsd:complexType>
1358
1359 <xsd:complexType name="CCCmdReportExtension">
1360   <xsd:sequence>
1361     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1362       namespace="##local"/>
1363   </xsd:sequence>
1364   <xsd:anyAttribute processContents="lax"/>
1365 </xsd:complexType>
1366
1367 <xsd:complexType name="CCCmdSpec">
1368   <xsd:sequence>
1369     <xsd:element name="filterSpec" type="ale:CCFilterSpec" minOccurs="0"/>
1370     <xsd:element name="opSpecs" minOccurs="0">
1371       <xsd:complexType>
1372         <xsd:sequence>

```

```

1373         <xsd:element name="opSpec" type="ale:CCOpSpec" minOccurs="0"
1374             maxOccurs="unbounded" />
1375     </xsd:sequence>
1376 </xsd:complexType>
1377 </xsd:element>
1378 <xsd:element name="statProfileNames" minOccurs="0">
1379     <xsd:complexType>
1380         <xsd:sequence>
1381             <xsd:element name="statProfileName" type="ale:CCStatProfileName"
1382                 minOccurs="0" maxOccurs="unbounded" />
1383         </xsd:sequence>
1384     </xsd:complexType>
1385 </xsd:element>
1386 <xsd:element name="extension" type="ale:CCCmdSpecExtension" minOccurs="0" />
1387 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1388     namespace="##other" />
1389 </xsd:sequence>
1390 <xsd:attribute name="name" type="xsd:string" use="required" />
1391 <xsd:attribute name="reportIfEmpty" type="xsd:boolean" default="false" />
1392 <xsd:anyAttribute processContents="lax" />
1393 </xsd:complexType>
1394
1395 <xsd:complexType name="CCCmdSpecExtension">
1396     <xsd:sequence>
1397         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1398             namespace="##local" />
1399     </xsd:sequence>
1400     <xsd:anyAttribute processContents="lax" />
1401 </xsd:complexType>
1402
1403 <xsd:complexType name="CCFilterSpec">
1404     <xsd:sequence>
1405         <xsd:element name="filterList" minOccurs="0">
1406             <xsd:complexType>
1407                 <xsd:sequence>
1408                     <xsd:element name="filter" type="ale:ECFilterListMember" minOccurs="0"
1409                         maxOccurs="unbounded" />
1410                 </xsd:sequence>
1411             </xsd:complexType>
1412         </xsd:element>
1413         <xsd:element name="extension" type="ale:CCFilterSpecExtension" minOccurs="0" />
1414         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1415             namespace="##other" />
1416     </xsd:sequence>
1417     <xsd:anyAttribute processContents="lax" />
1418 </xsd:complexType>
1419
1420 <xsd:complexType name="CCFilterSpecExtension">
1421     <xsd:sequence>
1422         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1423             namespace="##local" />
1424     </xsd:sequence>
1425     <xsd:anyAttribute processContents="lax" />
1426 </xsd:complexType>
1427
1428 <!-- The CCInitiationCondition type is an enumerated type.
1429     The following strings are legal values for this type:
1430     TRIGGER
1431     REPEAT_PERIOD
1432     REQUESTED
1433     UNDEFINE
1434     An implementation may also recognize additional strings as extensions.
1435     -->
1436 <xsd:simpleType name="CCInitiationCondition">
1437     <xsd:restriction base="xsd:string" />
1438 </xsd:simpleType>
1439
1440 <!-- The CCLockOperation type is an enumerated type.
1441     The following strings are legal values for this type:
1442     UNLOCK

```

```

1443         PERMAUNLOCK
1444         LOCK
1445         PERMALOCK
1446     An implementation may also recognize additional strings as extensions.
1447     -->
1448 <xsd:simpleType name="CCLockOperation">
1449     <xsd:restriction base="xsd:string"/>
1450 </xsd:simpleType>
1451
1452 <xsd:complexType name="CCOpDataSpec">
1453     <xsd:sequence>
1454         <xsd:element name="specType" type="ale:CCOpDataSpecType"/>
1455         <xsd:element name="data" type="xsd:string"/>
1456         <xsd:element name="extension" type="ale:CCOpDataSpecExtension" minOccurs="0"/>
1457         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1458             namespace="##other"/>
1459     </xsd:sequence>
1460     <xsd:anyAttribute processContents="lax"/>
1461 </xsd:complexType>
1462
1463 <xsd:complexType name="CCOpDataSpecExtension">
1464     <xsd:sequence>
1465         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1466             namespace="##local"/>
1467     </xsd:sequence>
1468     <xsd:anyAttribute processContents="lax"/>
1469 </xsd:complexType>
1470
1471 <!-- The CCOpDataSpecType type is an enumerated type.
1472     The following strings are legal values for this type:
1473     LITERAL
1474     PARAMETER
1475     CACHE
1476     ASSOCIATION
1477     RANDOM
1478     An implementation may also recognize additional strings as extensions.
1479     -->
1480 <xsd:simpleType name="CCOpDataSpecType">
1481     <xsd:restriction base="xsd:string"/>
1482 </xsd:simpleType>
1483
1484 <xsd:complexType name="CCOpReport">
1485     <xsd:sequence>
1486         <xsd:element name="data" type="xsd:string" minOccurs="0"/>
1487         <xsd:element name="opStatus" type="ale:CCStatus"/>
1488         <xsd:element name="opName" type="xsd:string" minOccurs="0"/>
1489         <xsd:element name="extension" type="ale:CCOpReportExtension" minOccurs="0"/>
1490         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1491             namespace="##other"/>
1492     </xsd:sequence>
1493     <xsd:anyAttribute processContents="lax"/>
1494 </xsd:complexType>
1495
1496 <xsd:complexType name="CCOpReportExtension">
1497     <xsd:sequence>
1498         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1499             namespace="##local"/>
1500     </xsd:sequence>
1501     <xsd:anyAttribute processContents="lax"/>
1502 </xsd:complexType>
1503
1504 <xsd:complexType name="CCOpSpec">
1505     <xsd:sequence>
1506         <xsd:element name="opType" type="ale:CCOpType"/>
1507         <xsd:element name="fieldspec" type="ale:ECFieldSpec" minOccurs="0"/>
1508         <xsd:element name="dataSpec" type="ale:CCOpDataSpec" minOccurs="0"/>
1509         <xsd:element name="opName" type="xsd:string" minOccurs="0"/>
1510         <xsd:element name="extension" type="ale:CCOpSpecExtension" minOccurs="0"/>
1511         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1512             namespace="##other"/>

```

```

1513     </xsd:sequence>
1514     <xsd:anyAttribute processContents="lax" />
1515 </xsd:complexType>
1516
1517 <xsd:complexType name="CCOpSpecExtension">
1518   <xsd:sequence>
1519     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1520       namespace="##local" />
1521   </xsd:sequence>
1522   <xsd:anyAttribute processContents="lax" />
1523 </xsd:complexType>
1524
1525 <!-- The CCOpType type is an enumerated type.
1526     The following strings are legal values for this type:
1527     READ
1528     CHECK
1529     INITIALIZE
1530     ADD
1531     WRITE
1532     DELETE
1533     PASSWORD
1534     KILL
1535     LOCK
1536     An implementation may also recognize additional strings as extensions.
1537     -->
1538 <xsd:simpleType name="CCOpType">
1539   <xsd:restriction base="xsd:string" />
1540 </xsd:simpleType>
1541
1542 <xsd:complexType name="CCParameterList">
1543   <xsd:sequence>
1544     <xsd:element name="entries" minOccurs="0">
1545       <xsd:complexType>
1546         <xsd:sequence>
1547           <xsd:element name="entry" type="ale:CCParameterListEntry" minOccurs="0"
1548             maxOccurs="unbounded" />
1549         </xsd:sequence>
1550       </xsd:complexType>
1551     </xsd:element>
1552   </xsd:sequence>
1553 </xsd:complexType>
1554
1555 <xsd:complexType name="CCParameterListEntry">
1556   <xsd:sequence>
1557     <xsd:element name="name" type="xsd:string" />
1558     <xsd:element name="value" type="xsd:string" />
1559   </xsd:sequence>
1560 </xsd:complexType>
1561
1562 <xsd:complexType name="CCReports">
1563   <xsd:complexContent>
1564     <xsd:extension base="epcglobal:Document">
1565       <xsd:sequence>
1566         <xsd:element name="CCSpec" type="ale:CCSpec" minOccurs="0" />
1567         <xsd:element name="cmdReports" minOccurs="0">
1568           <xsd:complexType>
1569             <xsd:sequence>
1570               <xsd:element name="cmdReport" type="ale:CCCmdReport" minOccurs="0"
1571                 maxOccurs="unbounded" />
1572             </xsd:sequence>
1573           </xsd:complexType>
1574         </xsd:element>
1575         <xsd:element name="extension" type="ale:CCReportsExtension" minOccurs="0" />
1576         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1577           namespace="##other" />
1578       </xsd:sequence>
1579       <xsd:attribute name="specName" type="xsd:string" use="required" />
1580       <xsd:attribute name="date" type="xsd:dateTime" use="required" />
1581       <xsd:attribute name="ALEID" type="xsd:string" use="required" />
1582       <xsd:attribute name="totalMilliseconds" type="xsd:long" use="required" />

```

```

1583     <xsd:attribute name="initiationCondition" type="ale:CCInitiationCondition"
1584         use="required"/>
1585     <xsd:attribute name="initiationTrigger" type="ale:ECTrigger" use="optional"/>
1586     <xsd:attribute name="terminationCondition" type="ale:CCTerminationCondition"
1587         use="required"/>
1588     <xsd:attribute name="terminationTrigger" type="ale:ECTrigger" use="optional"/>
1589     <xsd:anyAttribute processContents="lax"/>
1590 </xsd:extension>
1591 </xsd:complexContent>
1592 </xsd:complexType>
1593
1594 <xsd:complexType name="CCReportsExtension">
1595     <xsd:sequence>
1596         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1597             namespace="##local"/>
1598     </xsd:sequence>
1599     <xsd:anyAttribute processContents="lax"/>
1600 </xsd:complexType>
1601
1602 <xsd:complexType name="CCSpec">
1603     <xsd:complexContent>
1604         <xsd:extension base="epcglobal:Document">
1605             <xsd:sequence>
1606                 <xsd:element name="logicalReaders">
1607                     <xsd:complexType>
1608                         <xsd:sequence>
1609                             <xsd:element name="logicalReader" type="xsd:string"
1610                                 maxOccurs="unbounded"/>
1611                         </xsd:sequence>
1612                     </xsd:complexType>
1613                 </xsd:element>
1614                 <xsd:element name="boundarySpec" type="ale:CCBoundarySpec"/>
1615                 <xsd:element name="cmdSpecs">
1616                     <xsd:complexType>
1617                         <xsd:sequence>
1618                             <xsd:element name="cmdSpec" type="ale:CCCmdSpec" maxOccurs="unbounded"/>
1619                         </xsd:sequence>
1620                     </xsd:complexType>
1621                 </xsd:element>
1622                 <xsd:element name="extension" type="ale:CCSpecExtension" minOccurs="0"/>
1623                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1624                     namespace="##other"/>
1625             </xsd:sequence>
1626             <xsd:attribute name="includeSpecInReports" type="xsd:boolean" default="false"/>
1627             <xsd:anyAttribute processContents="lax"/>
1628         </xsd:extension>
1629     </xsd:complexContent>
1630 </xsd:complexType>
1631
1632 <xsd:complexType name="CCSpecExtension">
1633     <xsd:sequence>
1634         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1635             namespace="##local"/>
1636     </xsd:sequence>
1637     <xsd:anyAttribute processContents="lax"/>
1638 </xsd:complexType>
1639
1640 <!-- The CCStatProfileName type is an enumerated type.
1641     The following strings are legal values for this type:
1642     An implementation may also recognize additional strings as extensions.
1643     -->
1644 <xsd:simpleType name="CCStatProfileName">
1645     <xsd:restriction base="xsd:string"/>
1646 </xsd:simpleType>
1647
1648 <!-- The CCStatus type is an enumerated type.
1649     The following strings are legal values for this type:
1650     SUCCESS
1651     MISC_ERROR_TOTAL
1652     MISC_ERROR_PARTIAL

```

```

1653     PERMISSION_ERROR
1654     PASSWORD_ERROR
1655     FIELD_NOT_FOUND_ERROR
1656     OP_NOT_POSSIBLE_ERROR
1657     OUT_OF_RANGE_ERROR
1658     FIELD_EXISTS_ERROR
1659     MEMORY_OVERFLOW_ERROR
1660     MEMORY_CHECK_ERROR
1661     ASSOCIATION_TABLE_VALUE_INVALID
1662     ASSOCIATION_TABLE_VALUE_MISSING
1663     EPC_CACHE_DEPLETED
1664     An implementation may also recognize additional strings as extensions.
1665     -->
1666     <xsd:simpleType name="CCStatus">
1667     <xsd:restriction base="xsd:string"/>
1668     </xsd:simpleType>
1669
1670     <xsd:complexType name="CCTagReport">
1671     <xsd:sequence>
1672     <xsd:element name="id" type="xsd:string" minOccurs="0"/>
1673     <xsd:element name="opReports" minOccurs="0">
1674     <xsd:complexType>
1675     <xsd:sequence>
1676     <xsd:element name="opReport" type="ale:CCOpReport" minOccurs="0"
1677     maxOccurs="unbounded"/>
1678     </xsd:sequence>
1679     </xsd:complexType>
1680     </xsd:element>
1681     <xsd:element name="stats" minOccurs="0">
1682     <xsd:complexType>
1683     <xsd:sequence>
1684     <xsd:element name="stat" type="ale:CCTagStat" minOccurs="0"
1685     maxOccurs="unbounded"/>
1686     </xsd:sequence>
1687     </xsd:complexType>
1688     </xsd:element>
1689     <xsd:element name="extension" type="ale:CCTagReportExtension" minOccurs="0"/>
1690     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1691     namespace="##other"/>
1692     </xsd:sequence>
1693     <xsd:anyAttribute processContents="lax"/>
1694     </xsd:complexType>
1695
1696     <xsd:complexType name="CCTagReportExtension">
1697     <xsd:sequence>
1698     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1699     namespace="##local"/>
1700     </xsd:sequence>
1701     <xsd:anyAttribute processContents="lax"/>
1702     </xsd:complexType>
1703
1704     <xsd:complexType name="CCTagStat">
1705     <xsd:sequence>
1706     <xsd:element name="profile" type="ale:CCStatProfileName"/>
1707     <xsd:element name="statBlocks" minOccurs="0">
1708     <xsd:complexType>
1709     <xsd:sequence>
1710     <xsd:element name="statBlock" type="ale:ECReaderStat" minOccurs="0"
1711     maxOccurs="unbounded"/>
1712     </xsd:sequence>
1713     </xsd:complexType>
1714     </xsd:element>
1715     </xsd:sequence>
1716     </xsd:complexType>
1717
1718     <!-- The CCTerminationCondition type is an enumerated type.
1719     The following strings are legal values for this type:
1720     TRIGGER
1721     DURATION
1722     NO_NEW_TAGS

```

```

1723         COUNT
1724         ERROR
1725         UNREQUEST
1726         UNDEFINE
1727     An implementation may also recognize additional strings as extensions.
1728     -->
1729 <xsd:simpleType name="CCTerminationCondition">
1730     <xsd:restriction base="xsd:string"/>
1731 </xsd:simpleType>
1732
1733 <xsd:complexType name="EPCCacheSpec">
1734     <xsd:complexContent>
1735         <xsd:extension base="epcglobal:Document">
1736             <xsd:sequence>
1737                 <xsd:element name="extension" type="ale:EPCCacheSpecExtension" minOccurs="0"/>
1738                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1739                     namespace="##other"/>
1740             </xsd:sequence>
1741             <xsd:anyAttribute processContents="lax"/>
1742         </xsd:extension>
1743     </xsd:complexContent>
1744 </xsd:complexType>
1745
1746 <xsd:complexType name="EPCCacheSpecExtension">
1747     <xsd:sequence>
1748         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1749             namespace="##local"/>
1750     </xsd:sequence>
1751     <xsd:anyAttribute processContents="lax"/>
1752 </xsd:complexType>
1753
1754 <xsd:complexType name="EPCPatternList">
1755     <xsd:sequence>
1756         <xsd:element name="patterns" minOccurs="0">
1757             <xsd:complexType>
1758                 <xsd:sequence>
1759                     <xsd:element name="pattern" type="xsd:string" minOccurs="0"
1760                         maxOccurs="unbounded"/>
1761                 </xsd:sequence>
1762             </xsd:complexType>
1763         </xsd:element>
1764     </xsd:sequence>
1765 </xsd:complexType>
1766
1767 <xsd:complexType name="RNGSpec">
1768     <xsd:complexContent>
1769         <xsd:extension base="epcglobal:Document">
1770             <xsd:sequence>
1771                 <xsd:element name="length" type="xsd:int"/>
1772                 <xsd:element name="extension" type="ale:RNGSpecExtension" minOccurs="0"/>
1773                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1774                     namespace="##other"/>
1775             </xsd:sequence>
1776             <xsd:anyAttribute processContents="lax"/>
1777         </xsd:extension>
1778     </xsd:complexContent>
1779 </xsd:complexType>
1780
1781 <xsd:complexType name="RNGSpecExtension">
1782     <xsd:sequence>
1783         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1784             namespace="##local"/>
1785     </xsd:sequence>
1786     <xsd:anyAttribute processContents="lax"/>
1787 </xsd:complexType>
1788 </xsd:schema>

```

### 1789 3.7 ALE Tag Memory API Schema

1790 The following is an XML Schema (XSD) for the ALE Tag Memory API, defining  
1791 TMSpec as a top level element.

```
1792 <?xml version="1.0" encoding="UTF-8"?>
1793 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1794             xmlns:ale="urn:epcglobal:ale:xsd:1"
1795             targetNamespace="urn:epcglobal:ale:xsd:1"
1796             xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
1797             attributeFormDefault="unqualified" version="1.0">
1798   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd"/>
1799   <!-- ALETM ELEMENTS -->
1800   <xsd:element name="TMSpec" type="ale:TMSpec"/>
1801
1802   <!-- ALETM TYPES -->
1803
1804   <xsd:complexType name="TMFixedFieldListSpec">
1805     <xsd:complexContent>
1806       <xsd:extension base="ale:TMSpec">
1807         <xsd:sequence>
1808           <xsd:element name="fixedFields" minOccurs="0">
1809             <xsd:complexType>
1810               <xsd:sequence>
1811                 <xsd:element name="fixedField" type="ale:TMFixedFieldSpec"
1812                             minOccurs="0" maxOccurs="unbounded"/>
1813               </xsd:sequence>
1814             </xsd:complexType>
1815           </xsd:element>
1816           <xsd:element name="extension" type="ale:TMFixedFieldListSpecExtension"
1817                       minOccurs="0"/>
1818           <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1819                 namespace="##other"/>
1820         </xsd:sequence>
1821         <xsd:anyAttribute processContents="lax"/>
1822       </xsd:extension>
1823     </xsd:complexContent>
1824   </xsd:complexType>
1825
1826   <xsd:complexType name="TMFixedFieldListSpecExtension">
1827     <xsd:sequence>
1828       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1829             namespace="##local"/>
1830     </xsd:sequence>
1831     <xsd:anyAttribute processContents="lax"/>
1832   </xsd:complexType>
1833
1834   <xsd:complexType name="TMFixedFieldSpec">
1835     <xsd:sequence>
1836       <xsd:element name="fieldname" type="xsd:string"/>
1837       <xsd:element name="bank" type="xsd:int"/>
1838       <xsd:element name="length" type="xsd:int"/>
1839       <xsd:element name="offset" type="xsd:int"/>
1840       <xsd:element name="defaultDatatype" type="xsd:string"/>
1841       <xsd:element name="defaultFormat" type="xsd:string"/>
1842       <xsd:element name="extension" type="ale:TMFixedFieldSpecExtension" minOccurs="0"/>
1843       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1844             namespace="##other"/>
1845     </xsd:sequence>
1846     <xsd:anyAttribute processContents="lax"/>
1847   </xsd:complexType>
1848
1849   <xsd:complexType name="TMFixedFieldSpecExtension">
1850     <xsd:sequence>
1851       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1852             namespace="##local"/>
1853     </xsd:sequence>
1854     <xsd:anyAttribute processContents="lax"/>
1855   </xsd:complexType>
```

```

1856
1857 <xsd:complexType name="TMSpec" abstract="true">
1858   <xsd:complexContent>
1859     <xsd:extension base="epcglobal:Document">
1860       <xsd:sequence>
1861         <xsd:element name="baseExtension" type="ale:TMSpecExtension" minOccurs="0"/>
1862       </xsd:sequence>
1863       <xsd:anyAttribute processContents="lax"/>
1864     </xsd:extension>
1865   </xsd:complexContent>
1866 </xsd:complexType>
1867
1868 <xsd:complexType name="TMSpecExtension">
1869   <xsd:sequence>
1870     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1871       namespace="##local"/>
1872   </xsd:sequence>
1873   <xsd:anyAttribute processContents="lax"/>
1874 </xsd:complexType>
1875
1876 <xsd:complexType name="TMVariableFieldListSpec">
1877   <xsd:complexContent>
1878     <xsd:extension base="ale:TMSpec">
1879       <xsd:sequence>
1880         <xsd:element name="variableFields" minOccurs="0">
1881           <xsd:complexType>
1882             <xsd:sequence>
1883               <xsd:element name="variableField" type="ale:TMVariableFieldSpec"
1884                 minOccurs="0" maxOccurs="unbounded"/>
1885             </xsd:sequence>
1886           </xsd:complexType>
1887         </xsd:element>
1888         <xsd:element name="extension" type="ale:TMVariableFieldListSpecExtension"
1889           minOccurs="0"/>
1890         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1891           namespace="##other"/>
1892       </xsd:sequence>
1893       <xsd:anyAttribute processContents="lax"/>
1894     </xsd:extension>
1895   </xsd:complexContent>
1896 </xsd:complexType>
1897
1898 <xsd:complexType name="TMVariableFieldListSpecExtension">
1899   <xsd:sequence>
1900     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1901       namespace="##local"/>
1902   </xsd:sequence>
1903   <xsd:anyAttribute processContents="lax"/>
1904 </xsd:complexType>
1905
1906 <xsd:complexType name="TMVariableFieldSpec">
1907   <xsd:sequence>
1908     <xsd:element name="fieldname" type="xsd:string"/>
1909     <xsd:element name="bank" type="xsd:int"/>
1910     <xsd:element name="oid" type="xsd:string"/>
1911     <xsd:element name="extension" type="ale:TMVariableFieldSpecExtension"
1912       minOccurs="0"/>
1913     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1914       namespace="##other"/>
1915   </xsd:sequence>
1916   <xsd:anyAttribute processContents="lax"/>
1917 </xsd:complexType>
1918
1919 <xsd:complexType name="TMVariableFieldSpecExtension">
1920   <xsd:sequence>
1921     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1922       namespace="##local"/>
1923   </xsd:sequence>
1924   <xsd:anyAttribute processContents="lax"/>
1925 </xsd:complexType>

```

1926 </xsd:schema>

### 1927 3.8 ALE Logical Reader API Schema

1928 The following is an XML Schema (XSD) for the ALE Logical Reader API, defining  
1929 LRSpec, and LRProperty as top level elements.

```
1930 <?xml version="1.0" encoding="UTF-8"?>
1931 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1932           xmlns:ale="urn:epcglobal:ale:xsd:1"
1933           targetNamespace="urn:epcglobal:ale:xsd:1"
1934           xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
1935           attributeFormDefault="unqualified" version="1.0">
1936   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd"/>
1937   <!-- ALELR ELEMENTS -->
1938   <xsd:element name="LRSpec" type="ale:LRSpec"/>
1939   <xsd:element name="LRProperty" type="ale:LRProperty"/>
1940
1941   <!-- ALELR TYPES -->
1942
1943   <xsd:complexType name="LRProperty">
1944     <xsd:sequence>
1945       <xsd:element name="name" type="xsd:string"/>
1946       <xsd:element name="value" type="xsd:string" minOccurs="0"/>
1947     </xsd:sequence>
1948   </xsd:complexType>
1949
1950   <xsd:complexType name="LRSpec">
1951     <xsd:complexContent>
1952       <xsd:extension base="epcglobal:Document">
1953         <xsd:sequence>
1954           <xsd:element name="isComposite" type="xsd:boolean" minOccurs="0"/>
1955           <xsd:element name="readers" minOccurs="0">
1956             <xsd:complexType>
1957               <xsd:sequence>
1958                 <xsd:element name="reader" type="xsd:string" minOccurs="0"
1959                             maxOccurs="unbounded"/>
1960               </xsd:sequence>
1961             </xsd:complexType>
1962           </xsd:element>
1963           <xsd:element name="properties" minOccurs="0">
1964             <xsd:complexType>
1965               <xsd:sequence>
1966                 <xsd:element name="property" type="ale:LRProperty" minOccurs="0"
1967                             maxOccurs="unbounded"/>
1968               </xsd:sequence>
1969             </xsd:complexType>
1970           </xsd:element>
1971           <xsd:element name="extension" type="ale:LRSpecExtension" minOccurs="0"/>
1972           <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1973                 namespace="##other"/>
1974         </xsd:sequence>
1975       <xsd:anyAttribute processContents="lax"/>
1976     </xsd:extension>
1977   </xsd:complexContent>
1978 </xsd:complexType>
1979
1980   <xsd:complexType name="LRSpecExtension">
1981     <xsd:sequence>
1982       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1983             namespace="##local"/>
1984     </xsd:sequence>
1985     <xsd:anyAttribute processContents="lax"/>
1986   </xsd:complexType>
1987 </xsd:schema>
```

### 1988 3.9 ALE Access Control API Schema

1989 The following is an XML Schema (XSD) for the ALE Access Control API, defining  
1990 ACPermission, ACRole, and ACClientIdentity as top level elements.

```
1991 <?xml version="1.0" encoding="UTF-8"?>
1992 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1993           xmlns:ale="urn:epcglobal:ale:xsd:1"
1994           targetNamespace="urn:epcglobal:ale:xsd:1"
1995           xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
1996           attributeFormDefault="unqualified" version="1.0">
1997   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd"/>
1998   <!-- ALEAC ELEMENTS -->
1999   <xsd:element name="ACPermission" type="ale:ACPermission"/>
2000   <xsd:element name="ACRole" type="ale:ACRole"/>
2001   <xsd:element name="ACClientIdentity" type="ale:ACClientIdentity"/>
2002
2003   <!-- ALEAC TYPES -->
2004
2005   <!-- The ACClass type is an enumerated type.
2006        The following strings are legal values for this type:
2007        Method
2008        An implementation may also recognize additional strings as extensions.
2009        -->
2010   <xsd:simpleType name="ACClass">
2011     <xsd:restriction base="xsd:string"/>
2012   </xsd:simpleType>
2013
2014   <xsd:complexType name="ACClientCredential">
2015     <xsd:sequence>
2016       <xsd:element name="extension" type="ale:ACClientCredentialExtension"
2017                 minOccurs="0"/>
2018       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
2019             namespace="##other"/>
2020     </xsd:sequence>
2021     <xsd:anyAttribute processContents="lax"/>
2022   </xsd:complexType>
2023
2024   <xsd:complexType name="ACClientCredentialExtension">
2025     <xsd:sequence>
2026       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
2027             namespace="##local"/>
2028     </xsd:sequence>
2029     <xsd:anyAttribute processContents="lax"/>
2030   </xsd:complexType>
2031
2032   <xsd:complexType name="ACClientIdentity">
2033     <xsd:complexContent>
2034       <xsd:extension base="epcglobal:Document">
2035         <xsd:sequence>
2036           <xsd:element name="credentials" minOccurs="0">
2037             <xsd:complexType>
2038               <xsd:sequence>
2039                 <xsd:element name="credential" type="ale:ACClientCredential"
2040                       minOccurs="0" maxOccurs="unbounded"/>
2041             </xsd:sequence>
2042           </xsd:complexType>
2043         </xsd:element>
2044         <xsd:element name="roleNames" minOccurs="0">
2045           <xsd:complexType>
2046             <xsd:sequence>
2047               <xsd:element name="roleName" type="xsd:string" minOccurs="0"
2048                       maxOccurs="unbounded"/>
2049             </xsd:sequence>
2050           </xsd:complexType>
2051         </xsd:element>
2052         <xsd:element name="extension" type="ale:ACClientIdentityExtension"
2053               minOccurs="0"/>
2054       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded">
```

```

2055         namespace="##other" />
2056     </xsd:sequence>
2057     <xsd:anyAttribute processContents="lax" />
2058 </xsd:extension>
2059 </xsd:complexContent>
2060 </xsd:complexType>
2061
2062 <xsd:complexType name="ACClientIdentityExtension">
2063     <xsd:sequence>
2064         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
2065             namespace="##local" />
2066     </xsd:sequence>
2067     <xsd:anyAttribute processContents="lax" />
2068 </xsd:complexType>
2069
2070 <xsd:complexType name="ACPermission">
2071     <xsd:complexContent>
2072         <xsd:extension base="epcglobal:Document">
2073             <xsd:sequence>
2074                 <xsd:element name="permissionClass" type="ale:ACClass" />
2075                 <xsd:element name="instances" minOccurs="0">
2076                     <xsd:complexType>
2077                         <xsd:sequence>
2078                             <xsd:element name="instance" type="xsd:string" minOccurs="0"
2079                                 maxOccurs="unbounded" />
2080                         </xsd:sequence>
2081                     </xsd:complexType>
2082                 </xsd:element>
2083                 <xsd:element name="extension" type="ale:ACPermissionExtension" minOccurs="0" />
2084                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
2085                     namespace="##other" />
2086             </xsd:sequence>
2087             <xsd:anyAttribute processContents="lax" />
2088         </xsd:extension>
2089     </xsd:complexContent>
2090 </xsd:complexType>
2091
2092 <xsd:complexType name="ACPermissionExtension">
2093     <xsd:sequence>
2094         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
2095             namespace="##local" />
2096     </xsd:sequence>
2097     <xsd:anyAttribute processContents="lax" />
2098 </xsd:complexType>
2099
2100 <xsd:complexType name="ACRole">
2101     <xsd:complexContent>
2102         <xsd:extension base="epcglobal:Document">
2103             <xsd:sequence>
2104                 <xsd:element name="permissionNames" minOccurs="0">
2105                     <xsd:complexType>
2106                         <xsd:sequence>
2107                             <xsd:element name="permissionName" type="xsd:string" minOccurs="0"
2108                                 maxOccurs="unbounded" />
2109                         </xsd:sequence>
2110                     </xsd:complexType>
2111                 </xsd:element>
2112                 <xsd:element name="extension" type="ale:ACRoleExtension" minOccurs="0" />
2113                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
2114                     namespace="##other" />
2115             </xsd:sequence>
2116             <xsd:anyAttribute processContents="lax" />
2117         </xsd:extension>
2118     </xsd:complexContent>
2119 </xsd:complexType>
2120
2121 <xsd:complexType name="ACRoleExtension">
2122     <xsd:sequence>
2123         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
2124             namespace="##local" />

```

```

2125     </xsd:sequence>
2126     <xsd:anyAttribute processContents="lax" />
2127   </xsd:complexType>
2128 </xsd:schema>

```

## 2129 **4 SOAP Bindings**

2130 This section defines SOAP 1.1 [SOAP1.1] bindings for the five ALE APIs, using WSDL.  
 2131 All SOAP bindings are WS-I Basic Profile 1.0 [WSI] compliant for greatest  
 2132 interoperability.

### 2133 **4.1 Organization of the SOAP Bindings**

2134 Because an implementation may not implement all five ALE APIs, each API has a  
 2135 separate WSDL file. The request, response, and exception message types are defined in a  
 2136 separate XML namespace for each API. Complex data types referred to in requests and  
 2137 responses are all imported from the XML schemas defined in Section 2, which are all in a  
 2138 common XML namespace..

2139 The five WSDL files are summarized in the table below:

| WSDL  | Section | Imported Schema | XML Namespace              |
|-------|---------|-----------------|----------------------------|
| ale   | 4.3     | ale             | urn:epcglobal:ale:wSDL:1   |
| alecc | 4.4     | alecc           | urn:epcglobal:alecc:wSDL:1 |
| aletm | 4.5     | aletm           | urn:epcglobal:aletm:wSDL:1 |
| alelr | 4.6     | alelr           | urn:epcglobal:alelr:wSDL:1 |
| aleac | 4.7     | aleac           | urn:epcglobal:aleac:wSDL:1 |

2140

### 2141 **4.2 Link Level Security**

2142 ALE implementations that wish to provide a measure of security between themselves and  
 2143 their clients MAY implement an HTTPS (“HTTP Over TLS”) [RFC2818] transport for  
 2144 SOAP messages.

### 2145 **4.3 ALE Reading API SOAP Binding**

2146 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
 2147 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Reading  
 2148 API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0 [WSI].

```

2149 <?xml version="1.0" encoding="UTF-8"?>
2150 <!-- ALESERVICE DEFINITIONS -->
2151 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2152     xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
2153     xmlns:wSDLsoap="http://schemas.xmlsoap.org/wSDL/soap/"
2154     xmlns:ale="urn:epcglobal:ale:xsd:1"
2155     xmlns:impl="urn:epcglobal:ale:wSDL:1"
2156     targetNamespace="urn:epcglobal:ale:wSDL:1">
2157   <!-- ALESERVICE TYPES -->
2158   <wSDL:types>

```

```

2159 <xsd:schema targetNamespace="urn:epcglobal:ale:wSDL:1">
2160   <xsd:import namespace="urn:epcglobal:ale:xsd:1"
2161     schemaLocation="EPCglobal-ale-1_1-ale.xsd"/>
2162   <!-- ALESERVICE MESSAGE WRAPPERS -->
2163
2164   <xsd:element name="Define" type="impl:Define"/>
2165   <xsd:complexType name="Define">
2166     <xsd:sequence>
2167       <xsd:element name="specName" type="xsd:string"/>
2168       <xsd:element name="spec" type="ale:ECSpec"/>
2169     </xsd:sequence>
2170   </xsd:complexType>
2171
2172   <xsd:element name="Undefine" type="impl:Undefine"/>
2173   <xsd:complexType name="Undefine">
2174     <xsd:sequence>
2175       <xsd:element name="specName" type="xsd:string"/>
2176     </xsd:sequence>
2177   </xsd:complexType>
2178
2179   <xsd:element name="GetECSpec" type="impl:GetECSpec"/>
2180   <xsd:complexType name="GetECSpec">
2181     <xsd:sequence>
2182       <xsd:element name="specName" type="xsd:string"/>
2183     </xsd:sequence>
2184   </xsd:complexType>
2185   <xsd:element name="GetECSpecResult" type="ale:ECSpec"/>
2186
2187   <xsd:element name="GetECSpecNames" type="impl:EmptyParms"/>
2188   <xsd:element name="GetECSpecNamesResult" type="impl:ArrayOfString"/>
2189
2190   <xsd:element name="Subscribe" type="impl:Subscribe"/>
2191   <xsd:complexType name="Subscribe">
2192     <xsd:sequence>
2193       <xsd:element name="specName" type="xsd:string"/>
2194       <xsd:element name="notificationURI" type="xsd:string"/>
2195     </xsd:sequence>
2196   </xsd:complexType>
2197
2198   <xsd:element name="Unsubscribe" type="impl:Unsubscribe"/>
2199   <xsd:complexType name="Unsubscribe">
2200     <xsd:sequence>
2201       <xsd:element name="specName" type="xsd:string"/>
2202       <xsd:element name="notificationURI" type="xsd:string"/>
2203     </xsd:sequence>
2204   </xsd:complexType>
2205
2206   <xsd:element name="Poll" type="impl:Poll"/>
2207   <xsd:complexType name="Poll">
2208     <xsd:sequence>
2209       <xsd:element name="specName" type="xsd:string"/>
2210     </xsd:sequence>
2211   </xsd:complexType>
2212   <xsd:element name="PollResult" type="ale:ECReports"/>
2213
2214   <xsd:element name="Immediate" type="impl:Immediate"/>
2215   <xsd:complexType name="Immediate">
2216     <xsd:sequence>
2217       <xsd:element name="spec" type="ale:ECSpec"/>
2218     </xsd:sequence>
2219   </xsd:complexType>
2220   <xsd:element name="ImmediateResult" type="ale:ECReports"/>
2221
2222   <xsd:element name="GetSubscribers" type="impl:GetSubscribers"/>
2223   <xsd:complexType name="GetSubscribers">
2224     <xsd:sequence>
2225       <xsd:element name="specName" type="xsd:string"/>
2226     </xsd:sequence>
2227   </xsd:complexType>
2228   <xsd:element name="GetSubscribersResult" type="impl:ArrayOfString"/>

```

```

2229
2230 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
2231 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
2232
2233 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
2234 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
2235
2236 <xsd:element name="ALEException" type="impl:ALEException"/>
2237 <xsd:complexType name="ALEException">
2238   <xsd:sequence>
2239     <xsd:element name="reason" type="xsd:string"/>
2240   </xsd:sequence>
2241 </xsd:complexType>
2242
2243 <xsd:element name="SecurityException" type="impl:SecurityException"/>
2244 <xsd:complexType name="SecurityException">
2245   <xsd:complexContent>
2246     <xsd:extension base="impl:ALEException"/>
2247   </xsd:complexContent>
2248 </xsd:complexType>
2249
2250 <xsd:element name="DuplicateNameException" type="impl:DuplicateNameException"/>
2251 <xsd:complexType name="DuplicateNameException">
2252   <xsd:complexContent>
2253     <xsd:extension base="impl:ALEException"/>
2254   </xsd:complexContent>
2255 </xsd:complexType>
2256
2257 <xsd:element name="ECSpecValidationException"
2258   type="impl:ECSpecValidationException"/>
2259 <xsd:complexType name="ECSpecValidationException">
2260   <xsd:complexContent>
2261     <xsd:extension base="impl:ALEException"/>
2262   </xsd:complexContent>
2263 </xsd:complexType>
2264
2265 <xsd:element name="InvalidURIException" type="impl:InvalidURIException"/>
2266 <xsd:complexType name="InvalidURIException">
2267   <xsd:complexContent>
2268     <xsd:extension base="impl:ALEException"/>
2269   </xsd:complexContent>
2270 </xsd:complexType>
2271
2272 <xsd:element name="NoSuchNameException" type="impl:NoSuchNameException"/>
2273 <xsd:complexType name="NoSuchNameException">
2274   <xsd:complexContent>
2275     <xsd:extension base="impl:ALEException"/>
2276   </xsd:complexContent>
2277 </xsd:complexType>
2278
2279 <xsd:element name="NoSuchSubscriberException"
2280   type="impl:NoSuchSubscriberException"/>
2281 <xsd:complexType name="NoSuchSubscriberException">
2282   <xsd:complexContent>
2283     <xsd:extension base="impl:ALEException"/>
2284   </xsd:complexContent>
2285 </xsd:complexType>
2286
2287 <xsd:element name="DuplicateSubscriptionException"
2288   type="impl:DuplicateSubscriptionException"/>
2289 <xsd:complexType name="DuplicateSubscriptionException">
2290   <xsd:complexContent>
2291     <xsd:extension base="impl:ALEException"/>
2292   </xsd:complexContent>
2293 </xsd:complexType>
2294
2295 <xsd:element name="ImplementationException" type="impl:ImplementationException"/>
2296 <xsd:complexType name="ImplementationException">
2297   <xsd:complexContent>
2298     <xsd:extension base="impl:ALEException">

```

```

2299         <xsd:sequence>
2300             <xsd:element name="severity" type="impl:ImplementationExceptionSeverity"/>
2301         </xsd:sequence>
2302     </xsd:extension>
2303 </xsd:complexContent>
2304 </xsd:complexType>
2305
2306     <xsd:complexType name="ArrayOfString">
2307         <xsd:sequence>
2308             <xsd:element name="string" type="xsd:string" minOccurs="0"
2309                 maxOccurs="unbounded"/>
2310         </xsd:sequence>
2311     </xsd:complexType>
2312
2313     <!-- The ImplementationExceptionSeverity type is an enumerated type.
2314         The following strings are legal values for this type:
2315         ERROR
2316         SEVERE
2317     -->
2318     <xsd:simpleType name="ImplementationExceptionSeverity">
2319         <xsd:restriction base="xsd:string"/>
2320     </xsd:simpleType>
2321
2322     <xsd:element name="VoidHolder" type="impl:VoidHolder"/>
2323     <xsd:complexType name="VoidHolder"/>
2324
2325     <xsd:complexType name="EmptyParms"/>
2326 </xsd:schema>
2327 </wsdl:types>
2328 <!-- ALESERVICE MESSAGES -->
2329
2330 <wsdl:message name="defineRequest">
2331     <wsdl:part name="parms" element="impl:Define"/>
2332 </wsdl:message>
2333 <wsdl:message name="defineResponse">
2334     <wsdl:part name="defineReturn" element="impl:VoidHolder"/>
2335 </wsdl:message>
2336
2337 <wsdl:message name="undefineRequest">
2338     <wsdl:part name="parms" element="impl:Undefine"/>
2339 </wsdl:message>
2340 <wsdl:message name="undefineResponse">
2341     <wsdl:part name="undefineReturn" element="impl:VoidHolder"/>
2342 </wsdl:message>
2343
2344 <wsdl:message name="getECSpecRequest">
2345     <wsdl:part name="parms" element="impl:GetECSpec"/>
2346 </wsdl:message>
2347 <wsdl:message name="getECSpecResponse">
2348     <wsdl:part name="getECSpecReturn" element="impl:GetECSpecResult"/>
2349 </wsdl:message>
2350
2351 <wsdl:message name="getECSpecNamesRequest">
2352     <wsdl:part name="parms" element="impl:GetECSpecNames"/>
2353 </wsdl:message>
2354 <wsdl:message name="getECSpecNamesResponse">
2355     <wsdl:part name="getECSpecNamesReturn" element="impl:GetECSpecNamesResult"/>
2356 </wsdl:message>
2357
2358 <wsdl:message name="subscribeRequest">
2359     <wsdl:part name="parms" element="impl:Subscribe"/>
2360 </wsdl:message>
2361 <wsdl:message name="subscribeResponse">
2362     <wsdl:part name="subscribeReturn" element="impl:VoidHolder"/>
2363 </wsdl:message>
2364
2365 <wsdl:message name="unsubscribeRequest">
2366     <wsdl:part name="parms" element="impl:Unsubscribe"/>
2367 </wsdl:message>
2368 <wsdl:message name="unsubscribeResponse">

```

```

2369     <wsdl:part name="unsubscribeReturn" element="impl:VoidHolder"/>
2370 </wsdl:message>
2371
2372 <wsdl:message name="pollRequest">
2373     <wsdl:part name="parms" element="impl:Poll"/>
2374 </wsdl:message>
2375 <wsdl:message name="pollResponse">
2376     <wsdl:part name="pollReturn" element="impl:PollResult"/>
2377 </wsdl:message>
2378
2379 <wsdl:message name="immediateRequest">
2380     <wsdl:part name="parms" element="impl:Immediate"/>
2381 </wsdl:message>
2382 <wsdl:message name="immediateResponse">
2383     <wsdl:part name="immediateReturn" element="impl:ImmediateResult"/>
2384 </wsdl:message>
2385
2386 <wsdl:message name="getSubscribersRequest">
2387     <wsdl:part name="parms" element="impl:GetSubscribers"/>
2388 </wsdl:message>
2389 <wsdl:message name="getSubscribersResponse">
2390     <wsdl:part name="getSubscribersReturn" element="impl:GetSubscribersResult"/>
2391 </wsdl:message>
2392
2393 <wsdl:message name="getStandardVersionRequest">
2394     <wsdl:part name="parms" element="impl:GetStandardVersion"/>
2395 </wsdl:message>
2396 <wsdl:message name="getStandardVersionResponse">
2397     <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult"/>
2398 </wsdl:message>
2399
2400 <wsdl:message name="getVendorVersionRequest">
2401     <wsdl:part name="parms" element="impl:GetVendorVersion"/>
2402 </wsdl:message>
2403 <wsdl:message name="getVendorVersionResponse">
2404     <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult"/>
2405 </wsdl:message>
2406
2407 <wsdl:message name="SecurityExceptionResponse">
2408     <wsdl:part name="fault" element="impl:SecurityException"/>
2409 </wsdl:message>
2410
2411 <wsdl:message name="DuplicateNameExceptionResponse">
2412     <wsdl:part name="fault" element="impl:DuplicateNameException"/>
2413 </wsdl:message>
2414
2415 <wsdl:message name="ECSpecValidationExceptionResponse">
2416     <wsdl:part name="fault" element="impl:ECSpecValidationException"/>
2417 </wsdl:message>
2418
2419 <wsdl:message name="InvalidURIExceptionResponse">
2420     <wsdl:part name="fault" element="impl:InvalidURIException"/>
2421 </wsdl:message>
2422
2423 <wsdl:message name="NoSuchNameExceptionResponse">
2424     <wsdl:part name="fault" element="impl:NoSuchNameException"/>
2425 </wsdl:message>
2426
2427 <wsdl:message name="NoSuchSubscriberExceptionResponse">
2428     <wsdl:part name="fault" element="impl:NoSuchSubscriberException"/>
2429 </wsdl:message>
2430
2431 <wsdl:message name="DuplicateSubscriptionExceptionResponse">
2432     <wsdl:part name="fault" element="impl:DuplicateSubscriptionException"/>
2433 </wsdl:message>
2434
2435 <wsdl:message name="ImplementationExceptionResponse">
2436     <wsdl:part name="fault" element="impl:ImplementationException"/>
2437 </wsdl:message>
2438 <!-- ALESERVICE PORTTYPE -->

```

```

2439
2440 <wsdl:portType name="ALEServicePortType">
2441
2442   <wsdl:operation name="define">
2443     <wsdl:input message="impl:defineRequest" name="defineRequest" />
2444     <wsdl:output message="impl:defineResponse" name="defineResponse" />
2445     <wsdl:fault message="impl:DuplicateNameExceptionResponse"
2446       name="DuplicateNameExceptionFault" />
2447     <wsdl:fault message="impl:ECSpecValidationExceptionResponse"
2448       name="ECSpecValidationExceptionFault" />
2449     <wsdl:fault message="impl:SecurityExceptionResponse"
2450       name="SecurityExceptionFault" />
2451     <wsdl:fault message="impl:ImplementationExceptionResponse"
2452       name="ImplementationExceptionFault" />
2453   </wsdl:operation>
2454
2455   <wsdl:operation name="undefine">
2456     <wsdl:input message="impl:undefineRequest" name="undefineRequest" />
2457     <wsdl:output message="impl:undefineResponse" name="undefineResponse" />
2458     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2459       name="NoSuchNameExceptionFault" />
2460     <wsdl:fault message="impl:SecurityExceptionResponse"
2461       name="SecurityExceptionFault" />
2462     <wsdl:fault message="impl:ImplementationExceptionResponse"
2463       name="ImplementationExceptionFault" />
2464   </wsdl:operation>
2465
2466   <wsdl:operation name="getECSpec">
2467     <wsdl:input message="impl:getECSpecRequest" name="getECSpecRequest" />
2468     <wsdl:output message="impl:getECSpecResponse" name="getECSpecResponse" />
2469     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2470       name="NoSuchNameExceptionFault" />
2471     <wsdl:fault message="impl:SecurityExceptionResponse"
2472       name="SecurityExceptionFault" />
2473     <wsdl:fault message="impl:ImplementationExceptionResponse"
2474       name="ImplementationExceptionFault" />
2475   </wsdl:operation>
2476
2477   <wsdl:operation name="getECSpecNames">
2478     <wsdl:input message="impl:getECSpecNamesRequest" name="getECSpecNamesRequest" />
2479     <wsdl:output message="impl:getECSpecNamesResponse" name="getECSpecNamesResponse" />
2480     <wsdl:fault message="impl:SecurityExceptionResponse"
2481       name="SecurityExceptionFault" />
2482     <wsdl:fault message="impl:ImplementationExceptionResponse"
2483       name="ImplementationExceptionFault" />
2484   </wsdl:operation>
2485
2486   <wsdl:operation name="subscribe">
2487     <wsdl:input message="impl:subscribeRequest" name="subscribeRequest" />
2488     <wsdl:output message="impl:subscribeResponse" name="subscribeResponse" />
2489     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2490       name="NoSuchNameExceptionFault" />
2491     <wsdl:fault message="impl:InvalidURIExceptionResponse"
2492       name="InvalidURIExceptionFault" />
2493     <wsdl:fault message="impl:DuplicateSubscriptionExceptionResponse"
2494       name="DuplicateSubscriptionExceptionFault" />
2495     <wsdl:fault message="impl:SecurityExceptionResponse"
2496       name="SecurityExceptionFault" />
2497     <wsdl:fault message="impl:ImplementationExceptionResponse"
2498       name="ImplementationExceptionFault" />
2499   </wsdl:operation>
2500
2501   <wsdl:operation name="unsubscribe">
2502     <wsdl:input message="impl:unsubscribeRequest" name="unsubscribeRequest" />
2503     <wsdl:output message="impl:unsubscribeResponse" name="unsubscribeResponse" />
2504     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2505       name="NoSuchNameExceptionFault" />
2506     <wsdl:fault message="impl:NoSuchSubscriberExceptionResponse"
2507       name="NoSuchSubscriberExceptionFault" />
2508     <wsdl:fault message="impl:InvalidURIExceptionResponse"

```

```

2509         name="InvalidURIExceptionFault" />
2510     <wsdl:fault message="impl:SecurityExceptionResponse"
2511         name="SecurityExceptionFault" />
2512     <wsdl:fault message="impl:ImplementationExceptionResponse"
2513         name="ImplementationExceptionFault" />
2514 </wsdl:operation>
2515
2516 <wsdl:operation name="poll">
2517     <wsdl:input message="impl:pollRequest" name="pollRequest" />
2518     <wsdl:output message="impl:pollResponse" name="pollResponse" />
2519     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2520         name="NoSuchNameExceptionFault" />
2521     <wsdl:fault message="impl:SecurityExceptionResponse"
2522         name="SecurityExceptionFault" />
2523     <wsdl:fault message="impl:ImplementationExceptionResponse"
2524         name="ImplementationExceptionFault" />
2525 </wsdl:operation>
2526
2527 <wsdl:operation name="immediate">
2528     <wsdl:input message="impl:immediateRequest" name="immediateRequest" />
2529     <wsdl:output message="impl:immediateResponse" name="immediateResponse" />
2530     <wsdl:fault message="impl:ECSpecValidationExceptionResponse"
2531         name="ECSpecValidationExceptionFault" />
2532     <wsdl:fault message="impl:SecurityExceptionResponse"
2533         name="SecurityExceptionFault" />
2534     <wsdl:fault message="impl:ImplementationExceptionResponse"
2535         name="ImplementationExceptionFault" />
2536 </wsdl:operation>
2537
2538 <wsdl:operation name="getSubscribers">
2539     <wsdl:input message="impl:getSubscribersRequest" name="getSubscribersRequest" />
2540     <wsdl:output message="impl:getSubscribersResponse" name="getSubscribersResponse" />
2541     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2542         name="NoSuchNameExceptionFault" />
2543     <wsdl:fault message="impl:SecurityExceptionResponse"
2544         name="SecurityExceptionFault" />
2545     <wsdl:fault message="impl:ImplementationExceptionResponse"
2546         name="ImplementationExceptionFault" />
2547 </wsdl:operation>
2548
2549 <wsdl:operation name="getStandardVersion">
2550     <wsdl:input message="impl:getStandardVersionRequest"
2551         name="getStandardVersionRequest" />
2552     <wsdl:output message="impl:getStandardVersionResponse"
2553         name="getStandardVersionResponse" />
2554     <wsdl:fault message="impl:ImplementationExceptionResponse"
2555         name="ImplementationExceptionFault" />
2556 </wsdl:operation>
2557
2558 <wsdl:operation name="getVendorVersion">
2559     <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest" />
2560     <wsdl:output message="impl:getVendorVersionResponse"
2561         name="getVendorVersionResponse" />
2562     <wsdl:fault message="impl:ImplementationExceptionResponse"
2563         name="ImplementationExceptionFault" />
2564 </wsdl:operation>
2565 </wsdl:portType>
2566 <!-- ALESERVICE BINDING -->
2567
2568 <wsdl:binding name="ALEServiceBinding" type="impl:ALEServicePortType">
2569     <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
2570
2571     <wsdl:operation name="define">
2572         <wsdlsoap:operation soapAction="" />
2573         <wsdl:input name="defineRequest">
2574             <wsdlsoap:body use="literal" />
2575         </wsdl:input>
2576         <wsdl:output name="defineResponse">
2577             <wsdlsoap:body use="literal" />
2578         </wsdl:output>

```

```

2579     <wsdl:fault name="DuplicateNameExceptionFault">
2580       <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
2581     </wsdl:fault>
2582     <wsdl:fault name="ECSpecValidationExceptionFault">
2583       <wsdlsoap:fault name="ECSpecValidationExceptionFault" use="literal"/>
2584     </wsdl:fault>
2585     <wsdl:fault name="SecurityExceptionFault">
2586       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2587     </wsdl:fault>
2588     <wsdl:fault name="ImplementationExceptionFault">
2589       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2590     </wsdl:fault>
2591   </wsdl:operation>
2592
2593   <wsdl:operation name="undefine">
2594     <wsdlsoap:operation soapAction=""/>
2595     <wsdl:input name="undefineRequest">
2596       <wsdlsoap:body use="literal"/>
2597     </wsdl:input>
2598     <wsdl:output name="undefineResponse">
2599       <wsdlsoap:body use="literal"/>
2600     </wsdl:output>
2601     <wsdl:fault name="NoSuchNameExceptionFault">
2602       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2603     </wsdl:fault>
2604     <wsdl:fault name="SecurityExceptionFault">
2605       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2606     </wsdl:fault>
2607     <wsdl:fault name="ImplementationExceptionFault">
2608       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2609     </wsdl:fault>
2610   </wsdl:operation>
2611
2612   <wsdl:operation name="getECSpec">
2613     <wsdlsoap:operation soapAction=""/>
2614     <wsdl:input name="getECSpecRequest">
2615       <wsdlsoap:body use="literal"/>
2616     </wsdl:input>
2617     <wsdl:output name="getECSpecResponse">
2618       <wsdlsoap:body use="literal"/>
2619     </wsdl:output>
2620     <wsdl:fault name="NoSuchNameExceptionFault">
2621       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2622     </wsdl:fault>
2623     <wsdl:fault name="SecurityExceptionFault">
2624       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2625     </wsdl:fault>
2626     <wsdl:fault name="ImplementationExceptionFault">
2627       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2628     </wsdl:fault>
2629   </wsdl:operation>
2630
2631   <wsdl:operation name="getECSpecNames">
2632     <wsdlsoap:operation soapAction=""/>
2633     <wsdl:input name="getECSpecNamesRequest">
2634       <wsdlsoap:body use="literal"/>
2635     </wsdl:input>
2636     <wsdl:output name="getECSpecNamesResponse">
2637       <wsdlsoap:body use="literal"/>
2638     </wsdl:output>
2639     <wsdl:fault name="SecurityExceptionFault">
2640       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2641     </wsdl:fault>
2642     <wsdl:fault name="ImplementationExceptionFault">
2643       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2644     </wsdl:fault>
2645   </wsdl:operation>
2646
2647   <wsdl:operation name="subscribe">
2648     <wsdlsoap:operation soapAction=""/>

```

```

2649     <wsdl:input name="subscribeRequest">
2650         <wsdlsoap:body use="literal"/>
2651     </wsdl:input>
2652     <wsdl:output name="subscribeResponse">
2653         <wsdlsoap:body use="literal"/>
2654     </wsdl:output>
2655     <wsdl:fault name="NoSuchNameExceptionFault">
2656         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2657     </wsdl:fault>
2658     <wsdl:fault name="InvalidURIExceptionFault">
2659         <wsdlsoap:fault name="InvalidURIExceptionFault" use="literal"/>
2660     </wsdl:fault>
2661     <wsdl:fault name="DuplicateSubscriptionExceptionFault">
2662         <wsdlsoap:fault name="DuplicateSubscriptionExceptionFault" use="literal"/>
2663     </wsdl:fault>
2664     <wsdl:fault name="SecurityExceptionFault">
2665         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2666     </wsdl:fault>
2667     <wsdl:fault name="ImplementationExceptionFault">
2668         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2669     </wsdl:fault>
2670 </wsdl:operation>
2671
2672 <wsdl:operation name="unsubscribe">
2673     <wsdlsoap:operation soapAction=""/>
2674     <wsdl:input name="unsubscribeRequest">
2675         <wsdlsoap:body use="literal"/>
2676     </wsdl:input>
2677     <wsdl:output name="unsubscribeResponse">
2678         <wsdlsoap:body use="literal"/>
2679     </wsdl:output>
2680     <wsdl:fault name="NoSuchNameExceptionFault">
2681         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2682     </wsdl:fault>
2683     <wsdl:fault name="NoSuchSubscriberExceptionFault">
2684         <wsdlsoap:fault name="NoSuchSubscriberExceptionFault" use="literal"/>
2685     </wsdl:fault>
2686     <wsdl:fault name="InvalidURIExceptionFault">
2687         <wsdlsoap:fault name="InvalidURIExceptionFault" use="literal"/>
2688     </wsdl:fault>
2689     <wsdl:fault name="SecurityExceptionFault">
2690         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2691     </wsdl:fault>
2692     <wsdl:fault name="ImplementationExceptionFault">
2693         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2694     </wsdl:fault>
2695 </wsdl:operation>
2696
2697 <wsdl:operation name="poll">
2698     <wsdlsoap:operation soapAction=""/>
2699     <wsdl:input name="pollRequest">
2700         <wsdlsoap:body use="literal"/>
2701     </wsdl:input>
2702     <wsdl:output name="pollResponse">
2703         <wsdlsoap:body use="literal"/>
2704     </wsdl:output>
2705     <wsdl:fault name="NoSuchNameExceptionFault">
2706         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2707     </wsdl:fault>
2708     <wsdl:fault name="SecurityExceptionFault">
2709         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2710     </wsdl:fault>
2711     <wsdl:fault name="ImplementationExceptionFault">
2712         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2713     </wsdl:fault>
2714 </wsdl:operation>
2715
2716 <wsdl:operation name="immediate">
2717     <wsdlsoap:operation soapAction=""/>
2718     <wsdl:input name="immediateRequest">

```

```

2719     <wsdlsoap:body use="literal"/>
2720 </wsdl:input>
2721 <wsdl:output name="immediateResponse">
2722     <wsdlsoap:body use="literal"/>
2723 </wsdl:output>
2724 <wsdl:fault name="ECSpecValidationExceptionFault">
2725     <wsdlsoap:fault name="ECSpecValidationExceptionFault" use="literal"/>
2726 </wsdl:fault>
2727 <wsdl:fault name="SecurityExceptionFault">
2728     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2729 </wsdl:fault>
2730 <wsdl:fault name="ImplementationExceptionFault">
2731     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2732 </wsdl:fault>
2733 </wsdl:operation>
2734
2735 <wsdl:operation name="getSubscribers">
2736     <wsdlsoap:operation soapAction=""/>
2737     <wsdl:input name="getSubscribersRequest">
2738         <wsdlsoap:body use="literal"/>
2739     </wsdl:input>
2740     <wsdl:output name="getSubscribersResponse">
2741         <wsdlsoap:body use="literal"/>
2742     </wsdl:output>
2743     <wsdl:fault name="NoSuchNameExceptionFault">
2744         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2745     </wsdl:fault>
2746     <wsdl:fault name="SecurityExceptionFault">
2747         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2748     </wsdl:fault>
2749     <wsdl:fault name="ImplementationExceptionFault">
2750         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2751     </wsdl:fault>
2752 </wsdl:operation>
2753
2754 <wsdl:operation name="getStandardVersion">
2755     <wsdlsoap:operation soapAction=""/>
2756     <wsdl:input name="getStandardVersionRequest">
2757         <wsdlsoap:body use="literal"/>
2758     </wsdl:input>
2759     <wsdl:output name="getStandardVersionResponse">
2760         <wsdlsoap:body use="literal"/>
2761     </wsdl:output>
2762     <wsdl:fault name="ImplementationExceptionFault">
2763         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2764     </wsdl:fault>
2765 </wsdl:operation>
2766
2767 <wsdl:operation name="getVendorVersion">
2768     <wsdlsoap:operation soapAction=""/>
2769     <wsdl:input name="getVendorVersionRequest">
2770         <wsdlsoap:body use="literal"/>
2771     </wsdl:input>
2772     <wsdl:output name="getVendorVersionResponse">
2773         <wsdlsoap:body use="literal"/>
2774     </wsdl:output>
2775     <wsdl:fault name="ImplementationExceptionFault">
2776         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2777     </wsdl:fault>
2778 </wsdl:operation>
2779 </wsdl:binding>
2780
2781 <!-- ALESERVICE -->
2782 <wsdl:service name="ALEService">
2783     <wsdl:port binding="impl:ALEServiceBinding" name="ALEServicePort">
2784         <!-- The value of the location attribute below is an example only;
2785             Implementations are free to choose any appropriate URL. -->
2786         <wsdlsoap:address location="http://localhost:8080/services/ALEService"/>
2787     </wsdl:port>
2788 </wsdl:service>

```

2789 </wsdl:definitions>

## 2790 4.4 ALE Writing API SOAP Binding

2791 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
2792 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Writing  
2793 API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0 [WSI].

```
2794 <?xml version="1.0" encoding="UTF-8"?>
2795 <!-- ALECCSERVICE DEFINITIONS -->
2796 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2797                 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
2798                 xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
2799                 xmlns:ale="urn:epcglobal:ale:xsd:1"
2800                 xmlns:impl="urn:epcglobal:alecc:wsdl:1"
2801                 targetNamespace="urn:epcglobal:alecc:wsdl:1">
2802   <!-- ALECCSERVICE TYPES -->
2803   <wsdl:types>
2804     <xsd:schema targetNamespace="urn:epcglobal:alecc:wsdl:1">
2805       <xsd:import namespace="urn:epcglobal:ale:xsd:1"
2806                 schemaLocation="EPCglobal-ale-1_1-alecc.xsd"/>
2807       <!-- ALECCSERVICE MESSAGE WRAPPERS -->
2808
2809       <xsd:element name="Define" type="impl:Define"/>
2810       <xsd:complexType name="Define">
2811         <xsd:sequence>
2812           <xsd:element name="specName" type="xsd:string"/>
2813           <xsd:element name="spec" type="ale:CCSpec"/>
2814         </xsd:sequence>
2815       </xsd:complexType>
2816       <xsd:element name="DefineResult">
2817         <xsd:complexType/>
2818       </xsd:element>
2819
2820       <xsd:element name="Undefine" type="impl:Undefine"/>
2821       <xsd:complexType name="Undefine">
2822         <xsd:sequence>
2823           <xsd:element name="specName" type="xsd:string"/>
2824         </xsd:sequence>
2825       </xsd:complexType>
2826       <xsd:element name="UndefineResult">
2827         <xsd:complexType/>
2828       </xsd:element>
2829
2830       <xsd:element name="GetCCSpec" type="impl:GetCCSpec"/>
2831       <xsd:complexType name="GetCCSpec">
2832         <xsd:sequence>
2833           <xsd:element name="specName" type="xsd:string"/>
2834         </xsd:sequence>
2835       </xsd:complexType>
2836       <xsd:element name="GetCCSpecResult" type="ale:CCSpec"/>
2837
2838       <xsd:element name="GetCCSpecNames" type="impl:EmptyParms"/>
2839       <xsd:element name="GetCCSpecNamesResult" type="impl:ArrayOfString"/>
2840
2841       <xsd:element name="Subscribe" type="impl:Subscribe"/>
2842       <xsd:complexType name="Subscribe">
2843         <xsd:sequence>
2844           <xsd:element name="specName" type="xsd:string"/>
2845           <xsd:element name="notificationURI" type="xsd:string"/>
2846         </xsd:sequence>
2847       </xsd:complexType>
2848       <xsd:element name="SubscribeResult">
2849         <xsd:complexType/>
2850       </xsd:element>
2851
2852       <xsd:element name="Unsubscribe" type="impl:Unsubscribe"/>
2853       <xsd:complexType name="Unsubscribe">
```

```

2854     <xsd:sequence>
2855         <xsd:element name="specName" type="xsd:string"/>
2856         <xsd:element name="notificationURI" type="xsd:string"/>
2857     </xsd:sequence>
2858 </xsd:complexType>
2859 <xsd:element name="UnsubscribeResult">
2860     <xsd:complexType/>
2861 </xsd:element>
2862
2863 <xsd:element name="Poll" type="impl:Poll"/>
2864 <xsd:complexType name="Poll">
2865     <xsd:sequence>
2866         <xsd:element name="specName" type="xsd:string"/>
2867         <xsd:element name="params" type="ale:CCParameterList"/>
2868     </xsd:sequence>
2869 </xsd:complexType>
2870 <xsd:element name="PollResult" type="ale:CCReports"/>
2871
2872 <xsd:element name="Immediate" type="impl:Immediate"/>
2873 <xsd:complexType name="Immediate">
2874     <xsd:sequence>
2875         <xsd:element name="spec" type="ale:CCSpec"/>
2876     </xsd:sequence>
2877 </xsd:complexType>
2878 <xsd:element name="ImmediateResult" type="ale:CCReports"/>
2879
2880 <xsd:element name="GetSubscribers" type="impl:GetSubscribers"/>
2881 <xsd:complexType name="GetSubscribers">
2882     <xsd:sequence>
2883         <xsd:element name="specName" type="xsd:string"/>
2884     </xsd:sequence>
2885 </xsd:complexType>
2886 <xsd:element name="GetSubscribersResult" type="impl:ArrayOfString"/>
2887
2888 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
2889 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
2890
2891 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
2892 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
2893
2894 <xsd:element name="DefineEPCCache" type="impl:DefineEPCCache"/>
2895 <xsd:complexType name="DefineEPCCache">
2896     <xsd:sequence>
2897         <xsd:element name="cacheName" type="xsd:string"/>
2898         <xsd:element name="spec" type="ale:EPCCacheSpec"/>
2899         <xsd:element name="replenishment" type="ale:EPCCPatternList"/>
2900     </xsd:sequence>
2901 </xsd:complexType>
2902 <xsd:element name="DefineEPCCacheResult">
2903     <xsd:complexType/>
2904 </xsd:element>
2905
2906 <xsd:element name="UndefineEPCCache" type="impl:UndefineEPCCache"/>
2907 <xsd:complexType name="UndefineEPCCache">
2908     <xsd:sequence>
2909         <xsd:element name="cacheName" type="xsd:string"/>
2910     </xsd:sequence>
2911 </xsd:complexType>
2912 <xsd:element name="UndefineEPCCacheResult" type="ale:EPCCPatternList"/>
2913
2914 <xsd:element name="GetEPCCache" type="impl:GetEPCCache"/>
2915 <xsd:complexType name="GetEPCCache">
2916     <xsd:sequence>
2917         <xsd:element name="cacheName" type="xsd:string"/>
2918     </xsd:sequence>
2919 </xsd:complexType>
2920 <xsd:element name="GetEPCCacheResult" type="ale:EPCCacheSpec"/>
2921
2922 <xsd:element name="GetEPCCacheNames" type="impl:EmptyParms"/>
2923 <xsd:element name="GetEPCCacheNamesResult" type="impl:ArrayOfString"/>

```

```

2924
2925 <xsd:element name="ReplenishEPCCache" type="impl:ReplenishEPCCache"/>
2926 <xsd:complexType name="ReplenishEPCCache">
2927   <xsd:sequence>
2928     <xsd:element name="cacheName" type="xsd:string"/>
2929     <xsd:element name="replenishment" type="ale:EPCCPatternList"/>
2930   </xsd:sequence>
2931 </xsd:complexType>
2932 <xsd:element name="ReplenishEPCCacheResult">
2933   <xsd:complexType/>
2934 </xsd:element>
2935
2936 <xsd:element name="DepleteEPCCache" type="impl:DepleteEPCCache"/>
2937 <xsd:complexType name="DepleteEPCCache">
2938   <xsd:sequence>
2939     <xsd:element name="cacheName" type="xsd:string"/>
2940   </xsd:sequence>
2941 </xsd:complexType>
2942 <xsd:element name="DepleteEPCCacheResult" type="ale:EPCCPatternList"/>
2943
2944 <xsd:element name="GetEPCCacheContents" type="impl:GetEPCCacheContents"/>
2945 <xsd:complexType name="GetEPCCacheContents">
2946   <xsd:sequence>
2947     <xsd:element name="cacheName" type="xsd:string"/>
2948   </xsd:sequence>
2949 </xsd:complexType>
2950 <xsd:element name="GetEPCCacheContentsResult" type="ale:EPCCPatternList"/>
2951
2952 <xsd:element name="DefineAssocTable" type="impl:DefineAssocTable"/>
2953 <xsd:complexType name="DefineAssocTable">
2954   <xsd:sequence>
2955     <xsd:element name="tableName" type="xsd:string"/>
2956     <xsd:element name="spec" type="ale:AssocTableSpec"/>
2957     <xsd:element name="entries" type="ale:AssocTableEntryList"/>
2958   </xsd:sequence>
2959 </xsd:complexType>
2960 <xsd:element name="DefineAssocTableResult">
2961   <xsd:complexType/>
2962 </xsd:element>
2963
2964 <xsd:element name="UndefineAssocTable" type="impl:UndefineAssocTable"/>
2965 <xsd:complexType name="UndefineAssocTable">
2966   <xsd:sequence>
2967     <xsd:element name="tableName" type="xsd:string"/>
2968   </xsd:sequence>
2969 </xsd:complexType>
2970 <xsd:element name="UndefineAssocTableResult">
2971   <xsd:complexType/>
2972 </xsd:element>
2973
2974 <xsd:element name="GetAssocTableNames" type="impl:EmptyParms"/>
2975 <xsd:element name="GetAssocTableNamesResult" type="impl:ArrayOfString"/>
2976
2977 <xsd:element name="GetAssocTable" type="impl:GetAssocTable"/>
2978 <xsd:complexType name="GetAssocTable">
2979   <xsd:sequence>
2980     <xsd:element name="tableName" type="xsd:string"/>
2981   </xsd:sequence>
2982 </xsd:complexType>
2983 <xsd:element name="GetAssocTableResult" type="ale:AssocTableSpec"/>
2984
2985 <xsd:element name="PutAssocTableEntries" type="impl:PutAssocTableEntries"/>
2986 <xsd:complexType name="PutAssocTableEntries">
2987   <xsd:sequence>
2988     <xsd:element name="tableName" type="xsd:string"/>
2989     <xsd:element name="entries" type="ale:AssocTableEntryList"/>
2990   </xsd:sequence>
2991 </xsd:complexType>
2992 <xsd:element name="PutAssocTableEntriesResult">
2993   <xsd:complexType/>

```

```

2994     </xsd:element>
2995
2996     <xsd:element name="GetAssocTableValue" type="impl:GetAssocTableValue"/>
2997     <xsd:complexType name="GetAssocTableValue">
2998         <xsd:sequence>
2999             <xsd:element name="tableName" type="xsd:string"/>
3000             <xsd:element name="epc" type="xsd:string"/>
3001         </xsd:sequence>
3002     </xsd:complexType>
3003     <xsd:element name="GetAssocTableValueResult" type="xsd:string"/>
3004
3005     <xsd:element name="GetAssocTableEntries" type="impl:GetAssocTableEntries"/>
3006     <xsd:complexType name="GetAssocTableEntries">
3007         <xsd:sequence>
3008             <xsd:element name="tableName" type="xsd:string"/>
3009             <xsd:element name="patList" type="ale:EPCPatternList"/>
3010         </xsd:sequence>
3011     </xsd:complexType>
3012     <xsd:element name="GetAssocTableEntriesResult" type="ale:AssocTableEntryList"/>
3013
3014     <xsd:element name="RemoveAssocTableEntry" type="impl:RemoveAssocTableEntry"/>
3015     <xsd:complexType name="RemoveAssocTableEntry">
3016         <xsd:sequence>
3017             <xsd:element name="tableName" type="xsd:string"/>
3018             <xsd:element name="epc" type="xsd:string"/>
3019         </xsd:sequence>
3020     </xsd:complexType>
3021     <xsd:element name="RemoveAssocTableEntryResult">
3022         <xsd:complexType/>
3023     </xsd:element>
3024
3025     <xsd:element name="RemoveAssocTableEntries" type="impl:RemoveAssocTableEntries"/>
3026     <xsd:complexType name="RemoveAssocTableEntries">
3027         <xsd:sequence>
3028             <xsd:element name="tableName" type="xsd:string"/>
3029             <xsd:element name="patList" type="ale:EPCPatternList"/>
3030         </xsd:sequence>
3031     </xsd:complexType>
3032     <xsd:element name="RemoveAssocTableEntriesResult">
3033         <xsd:complexType/>
3034     </xsd:element>
3035
3036     <xsd:element name="DefineRNG" type="impl:DefineRNG"/>
3037     <xsd:complexType name="DefineRNG">
3038         <xsd:sequence>
3039             <xsd:element name="rngName" type="xsd:string"/>
3040             <xsd:element name="rngSpec" type="ale:RNGSpec"/>
3041         </xsd:sequence>
3042     </xsd:complexType>
3043     <xsd:element name="DefineRNGResult">
3044         <xsd:complexType/>
3045     </xsd:element>
3046
3047     <xsd:element name="UndefineRNG" type="impl:UndefineRNG"/>
3048     <xsd:complexType name="UndefineRNG">
3049         <xsd:sequence>
3050             <xsd:element name="rngName" type="xsd:string"/>
3051         </xsd:sequence>
3052     </xsd:complexType>
3053     <xsd:element name="UndefineRNGResult">
3054         <xsd:complexType/>
3055     </xsd:element>
3056
3057     <xsd:element name="GetRNGNames" type="impl:EmptyParms"/>
3058     <xsd:element name="GetRNGNamesResult" type="impl:ArrayOfString"/>
3059
3060     <xsd:element name="GetRNG" type="impl:GetRNG"/>
3061     <xsd:complexType name="GetRNG">
3062         <xsd:sequence>
3063             <xsd:element name="rngName" type="xsd:string"/>

```

```

3064         </xsd:sequence>
3065     </xsd:complexType>
3066     <xsd:element name="GetRNGResult" type="ale:RNGSpec"/>
3067
3068     <xsd:element name="ALEException" type="impl:ALEException"/>
3069     <xsd:complexType name="ALEException">
3070         <xsd:sequence>
3071             <xsd:element name="reason" type="xsd:string"/>
3072         </xsd:sequence>
3073     </xsd:complexType>
3074
3075     <xsd:element name="SecurityException" type="impl:SecurityException"/>
3076     <xsd:complexType name="SecurityException">
3077         <xsd:complexContent>
3078             <xsd:extension base="impl:ALEException"/>
3079         </xsd:complexContent>
3080     </xsd:complexType>
3081
3082     <xsd:element name="DuplicateNameException" type="impl:DuplicateNameException"/>
3083     <xsd:complexType name="DuplicateNameException">
3084         <xsd:complexContent>
3085             <xsd:extension base="impl:ALEException"/>
3086         </xsd:complexContent>
3087     </xsd:complexType>
3088
3089     <xsd:element name="CCSpecValidationException"
3090         type="impl:CCSpecValidationException"/>
3091     <xsd:complexType name="CCSpecValidationException">
3092         <xsd:complexContent>
3093             <xsd:extension base="impl:ALEException"/>
3094         </xsd:complexContent>
3095     </xsd:complexType>
3096
3097     <xsd:element name="InvalidURIException" type="impl:InvalidURIException"/>
3098     <xsd:complexType name="InvalidURIException">
3099         <xsd:complexContent>
3100             <xsd:extension base="impl:ALEException"/>
3101         </xsd:complexContent>
3102     </xsd:complexType>
3103
3104     <xsd:element name="NoSuchNameException" type="impl:NoSuchNameException"/>
3105     <xsd:complexType name="NoSuchNameException">
3106         <xsd:complexContent>
3107             <xsd:extension base="impl:ALEException"/>
3108         </xsd:complexContent>
3109     </xsd:complexType>
3110
3111     <xsd:element name="NoSuchSubscriberException"
3112         type="impl:NoSuchSubscriberException"/>
3113     <xsd:complexType name="NoSuchSubscriberException">
3114         <xsd:complexContent>
3115             <xsd:extension base="impl:ALEException"/>
3116         </xsd:complexContent>
3117     </xsd:complexType>
3118
3119     <xsd:element name="DuplicateSubscriptionException"
3120         type="impl:DuplicateSubscriptionException"/>
3121     <xsd:complexType name="DuplicateSubscriptionException">
3122         <xsd:complexContent>
3123             <xsd:extension base="impl:ALEException"/>
3124         </xsd:complexContent>
3125     </xsd:complexType>
3126
3127     <xsd:element name="ParameterException" type="impl:ParameterException"/>
3128     <xsd:complexType name="ParameterException">
3129         <xsd:complexContent>
3130             <xsd:extension base="impl:ALEException"/>
3131         </xsd:complexContent>
3132     </xsd:complexType>
3133

```

```

3134 <xsd:element name="ParameterForbiddenException"
3135         type="impl:ParameterForbiddenException" />
3136 <xsd:complexType name="ParameterForbiddenException">
3137     <xsd:complexContent>
3138         <xsd:extension base="impl:ALEException" />
3139     </xsd:complexContent>
3140 </xsd:complexType>
3141
3142 <xsd:element name="ImplementationException" type="impl:ImplementationException" />
3143 <xsd:complexType name="ImplementationException">
3144     <xsd:complexContent>
3145         <xsd:extension base="impl:ALEException">
3146             <xsd:sequence>
3147                 <xsd:element name="severity" type="impl:ImplementationExceptionSeverity" />
3148             </xsd:sequence>
3149         </xsd:extension>
3150     </xsd:complexContent>
3151 </xsd:complexType>
3152
3153 <xsd:element name="EPCCacheSpecValidationException"
3154         type="impl:EPCCacheSpecValidationException" />
3155 <xsd:complexType name="EPCCacheSpecValidationException">
3156     <xsd:complexContent>
3157         <xsd:extension base="impl:ALEException" />
3158     </xsd:complexContent>
3159 </xsd:complexType>
3160
3161 <xsd:element name="InvalidPatternException" type="impl:InvalidPatternException" />
3162 <xsd:complexType name="InvalidPatternException">
3163     <xsd:complexContent>
3164         <xsd:extension base="impl:ALEException" />
3165     </xsd:complexContent>
3166 </xsd:complexType>
3167
3168 <xsd:element name="InUseException" type="impl:InUseException" />
3169 <xsd:complexType name="InUseException">
3170     <xsd:complexContent>
3171         <xsd:extension base="impl:ALEException" />
3172     </xsd:complexContent>
3173 </xsd:complexType>
3174
3175 <xsd:element name="AssocTableValidationException"
3176         type="impl:AssocTableValidationException" />
3177 <xsd:complexType name="AssocTableValidationException">
3178     <xsd:complexContent>
3179         <xsd:extension base="impl:ALEException" />
3180     </xsd:complexContent>
3181 </xsd:complexType>
3182
3183 <xsd:element name="InvalidEPCEException" type="impl:InvalidEPCEException" />
3184 <xsd:complexType name="InvalidEPCEException">
3185     <xsd:complexContent>
3186         <xsd:extension base="impl:ALEException" />
3187     </xsd:complexContent>
3188 </xsd:complexType>
3189
3190 <xsd:element name="InvalidAssocTableEntryException"
3191         type="impl:InvalidAssocTableEntryException" />
3192 <xsd:complexType name="InvalidAssocTableEntryException">
3193     <xsd:complexContent>
3194         <xsd:extension base="impl:ALEException" />
3195     </xsd:complexContent>
3196 </xsd:complexType>
3197
3198 <xsd:element name="RNGValidationException" type="impl:RNGValidationException" />
3199 <xsd:complexType name="RNGValidationException">
3200     <xsd:complexContent>
3201         <xsd:extension base="impl:ALEException" />
3202     </xsd:complexContent>
3203 </xsd:complexType>

```

```

3204
3205     <xsd:complexType name="ArrayOfString">
3206         <xsd:sequence>
3207             <xsd:element name="string" type="xsd:string" minOccurs="0"
3208                 maxOccurs="unbounded" />
3209         </xsd:sequence>
3210     </xsd:complexType>
3211
3212     <!-- The ImplementationExceptionSeverity type is an enumerated type.
3213         The following strings are legal values for this type:
3214         ERROR
3215         SEVERE
3216         -->
3217     <xsd:simpleType name="ImplementationExceptionSeverity">
3218         <xsd:restriction base="xsd:string" />
3219     </xsd:simpleType>
3220
3221     <xsd:complexType name="EmptyParms" />
3222 </xsd:schema>
3223 </wsdl:types>
3224 <!-- ALECCSERVICE MESSAGES -->
3225
3226 <wsdl:message name="defineRequest">
3227     <wsdl:part name="parms" element="impl:Define" />
3228 </wsdl:message>
3229 <wsdl:message name="defineResponse">
3230     <wsdl:part name="defineReturn" element="impl:DefineResult" />
3231 </wsdl:message>
3232
3233 <wsdl:message name="undefineRequest">
3234     <wsdl:part name="parms" element="impl:Undefine" />
3235 </wsdl:message>
3236 <wsdl:message name="undefineResponse">
3237     <wsdl:part name="undefineReturn" element="impl:UndefineResult" />
3238 </wsdl:message>
3239
3240 <wsdl:message name="getCCSpecRequest">
3241     <wsdl:part name="parms" element="impl:GetCCSpec" />
3242 </wsdl:message>
3243 <wsdl:message name="getCCSpecResponse">
3244     <wsdl:part name="getCCSpecReturn" element="impl:GetCCSpecResult" />
3245 </wsdl:message>
3246
3247 <wsdl:message name="getCCSpecNamesRequest">
3248     <wsdl:part name="parms" element="impl:GetCCSpecNames" />
3249 </wsdl:message>
3250 <wsdl:message name="getCCSpecNamesResponse">
3251     <wsdl:part name="getCCSpecNamesReturn" element="impl:GetCCSpecNamesResult" />
3252 </wsdl:message>
3253
3254 <wsdl:message name="subscribeRequest">
3255     <wsdl:part name="parms" element="impl:Subscribe" />
3256 </wsdl:message>
3257 <wsdl:message name="subscribeResponse">
3258     <wsdl:part name="subscribeReturn" element="impl:SubscribeResult" />
3259 </wsdl:message>
3260
3261 <wsdl:message name="unsubscribeRequest">
3262     <wsdl:part name="parms" element="impl:Unsubscribe" />
3263 </wsdl:message>
3264 <wsdl:message name="unsubscribeResponse">
3265     <wsdl:part name="unsubscribeReturn" element="impl:UnsubscribeResult" />
3266 </wsdl:message>
3267
3268 <wsdl:message name="pollRequest">
3269     <wsdl:part name="parms" element="impl:Poll" />
3270 </wsdl:message>
3271 <wsdl:message name="pollResponse">
3272     <wsdl:part name="pollReturn" element="impl:PollResult" />
3273 </wsdl:message>

```

```

3274
3275 <wsdl:message name="immediateRequest">
3276   <wsdl:part name="parms" element="impl:Immediate"/>
3277 </wsdl:message>
3278 <wsdl:message name="immediateResponse">
3279   <wsdl:part name="immediateReturn" element="impl:ImmediateResult"/>
3280 </wsdl:message>
3281
3282 <wsdl:message name="getSubscribersRequest">
3283   <wsdl:part name="parms" element="impl:GetSubscribers"/>
3284 </wsdl:message>
3285 <wsdl:message name="getSubscribersResponse">
3286   <wsdl:part name="getSubscribersReturn" element="impl:GetSubscribersResult"/>
3287 </wsdl:message>
3288
3289 <wsdl:message name="getStandardVersionRequest">
3290   <wsdl:part name="parms" element="impl:GetStandardVersion"/>
3291 </wsdl:message>
3292 <wsdl:message name="getStandardVersionResponse">
3293   <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult"/>
3294 </wsdl:message>
3295
3296 <wsdl:message name="getVendorVersionRequest">
3297   <wsdl:part name="parms" element="impl:GetVendorVersion"/>
3298 </wsdl:message>
3299 <wsdl:message name="getVendorVersionResponse">
3300   <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult"/>
3301 </wsdl:message>
3302
3303 <wsdl:message name="defineEPCCacheRequest">
3304   <wsdl:part name="parms" element="impl:DefineEPCCache"/>
3305 </wsdl:message>
3306 <wsdl:message name="defineEPCCacheResponse">
3307   <wsdl:part name="defineEPCCacheReturn" element="impl:DefineEPCCacheResult"/>
3308 </wsdl:message>
3309
3310 <wsdl:message name="undefineEPCCacheRequest">
3311   <wsdl:part name="parms" element="impl:UndefineEPCCache"/>
3312 </wsdl:message>
3313 <wsdl:message name="undefineEPCCacheResponse">
3314   <wsdl:part name="undefineEPCCacheReturn" element="impl:UndefineEPCCacheResult"/>
3315 </wsdl:message>
3316
3317 <wsdl:message name="getEPCCacheRequest">
3318   <wsdl:part name="parms" element="impl:GetEPCCache"/>
3319 </wsdl:message>
3320 <wsdl:message name="getEPCCacheResponse">
3321   <wsdl:part name="getEPCCacheReturn" element="impl:GetEPCCacheResult"/>
3322 </wsdl:message>
3323
3324 <wsdl:message name="getEPCCacheNamesRequest">
3325   <wsdl:part name="parms" element="impl:GetEPCCacheNames"/>
3326 </wsdl:message>
3327 <wsdl:message name="getEPCCacheNamesResponse">
3328   <wsdl:part name="getEPCCacheNamesReturn" element="impl:GetEPCCacheNamesResult"/>
3329 </wsdl:message>
3330
3331 <wsdl:message name="replenishEPCCacheRequest">
3332   <wsdl:part name="parms" element="impl:ReplenishEPCCache"/>
3333 </wsdl:message>
3334 <wsdl:message name="replenishEPCCacheResponse">
3335   <wsdl:part name="replenishEPCCacheReturn" element="impl:ReplenishEPCCacheResult"/>
3336 </wsdl:message>
3337
3338 <wsdl:message name="depleteEPCCacheRequest">
3339   <wsdl:part name="parms" element="impl:DepleteEPCCache"/>
3340 </wsdl:message>
3341 <wsdl:message name="depleteEPCCacheResponse">
3342   <wsdl:part name="depleteEPCCacheReturn" element="impl:DepleteEPCCacheResult"/>
3343 </wsdl:message>

```

```

3344
3345 <wsdl:message name="getEPCCacheContentsRequest">
3346   <wsdl:part name="parms" element="impl:GetEPCCacheContents"/>
3347 </wsdl:message>
3348 <wsdl:message name="getEPCCacheContentsResponse">
3349   <wsdl:part name="getEPCCacheContentsReturn"
3350     element="impl:GetEPCCacheContentsResult"/>
3351 </wsdl:message>
3352
3353 <wsdl:message name="defineAssocTableRequest">
3354   <wsdl:part name="parms" element="impl:DefineAssocTable"/>
3355 </wsdl:message>
3356 <wsdl:message name="defineAssocTableResponse">
3357   <wsdl:part name="defineAssocTableReturn" element="impl:DefineAssocTableResult"/>
3358 </wsdl:message>
3359
3360 <wsdl:message name="undefineAssocTableRequest">
3361   <wsdl:part name="parms" element="impl:UndefineAssocTable"/>
3362 </wsdl:message>
3363 <wsdl:message name="undefineAssocTableResponse">
3364   <wsdl:part name="undefineAssocTableReturn" element="impl:UndefineAssocTableResult"/>
3365 </wsdl:message>
3366
3367 <wsdl:message name="getAssocTableNamesRequest">
3368   <wsdl:part name="parms" element="impl:GetAssocTableNames"/>
3369 </wsdl:message>
3370 <wsdl:message name="getAssocTableNamesResponse">
3371   <wsdl:part name="getAssocTableNamesReturn" element="impl:GetAssocTableNamesResult"/>
3372 </wsdl:message>
3373
3374 <wsdl:message name="getAssocTableRequest">
3375   <wsdl:part name="parms" element="impl:GetAssocTable"/>
3376 </wsdl:message>
3377 <wsdl:message name="getAssocTableResponse">
3378   <wsdl:part name="getAssocTableReturn" element="impl:GetAssocTableResult"/>
3379 </wsdl:message>
3380
3381 <wsdl:message name="putAssocTableEntriesRequest">
3382   <wsdl:part name="parms" element="impl:PutAssocTableEntries"/>
3383 </wsdl:message>
3384 <wsdl:message name="putAssocTableEntriesResponse">
3385   <wsdl:part name="putAssocTableEntriesReturn"
3386     element="impl:PutAssocTableEntriesResult"/>
3387 </wsdl:message>
3388
3389 <wsdl:message name="getAssocTableValueRequest">
3390   <wsdl:part name="parms" element="impl:GetAssocTableValue"/>
3391 </wsdl:message>
3392 <wsdl:message name="getAssocTableValueResponse">
3393   <wsdl:part name="getAssocTableValueReturn" element="impl:GetAssocTableValueResult"/>
3394 </wsdl:message>
3395
3396 <wsdl:message name="getAssocTableEntriesRequest">
3397   <wsdl:part name="parms" element="impl:GetAssocTableEntries"/>
3398 </wsdl:message>
3399 <wsdl:message name="getAssocTableEntriesResponse">
3400   <wsdl:part name="getAssocTableEntriesReturn"
3401     element="impl:GetAssocTableEntriesResult"/>
3402 </wsdl:message>
3403
3404 <wsdl:message name="removeAssocTableEntryRequest">
3405   <wsdl:part name="parms" element="impl:RemoveAssocTableEntry"/>
3406 </wsdl:message>
3407 <wsdl:message name="removeAssocTableEntryResponse">
3408   <wsdl:part name="removeAssocTableEntryReturn"
3409     element="impl:RemoveAssocTableEntryResult"/>
3410 </wsdl:message>
3411
3412 <wsdl:message name="removeAssocTableEntriesRequest">
3413   <wsdl:part name="parms" element="impl:RemoveAssocTableEntries"/>

```

```

3414 </wsdl:message>
3415 <wsdl:message name="removeAssocTableEntriesResponse">
3416   <wsdl:part name="removeAssocTableEntriesReturn"
3417     element="impl:RemoveAssocTableEntriesResult"/>
3418 </wsdl:message>
3419
3420 <wsdl:message name="defineRNGRequest">
3421   <wsdl:part name="parms" element="impl:DefineRNG"/>
3422 </wsdl:message>
3423 <wsdl:message name="defineRNGResponse">
3424   <wsdl:part name="defineRNGReturn" element="impl:DefineRNGResult"/>
3425 </wsdl:message>
3426
3427 <wsdl:message name="undefineRNGRequest">
3428   <wsdl:part name="parms" element="impl:UndefineRNG"/>
3429 </wsdl:message>
3430 <wsdl:message name="undefineRNGResponse">
3431   <wsdl:part name="undefineRNGReturn" element="impl:UndefineRNGResult"/>
3432 </wsdl:message>
3433
3434 <wsdl:message name="getRNGNamesRequest">
3435   <wsdl:part name="parms" element="impl:GetRNGNames"/>
3436 </wsdl:message>
3437 <wsdl:message name="getRNGNamesResponse">
3438   <wsdl:part name="getRNGNamesReturn" element="impl:GetRNGNamesResult"/>
3439 </wsdl:message>
3440
3441 <wsdl:message name="getRNGRequest">
3442   <wsdl:part name="parms" element="impl:GetRNG"/>
3443 </wsdl:message>
3444 <wsdl:message name="getRNGResponse">
3445   <wsdl:part name="getRNGReturn" element="impl:GetRNGResult"/>
3446 </wsdl:message>
3447
3448 <wsdl:message name="SecurityExceptionResponse">
3449   <wsdl:part name="fault" element="impl:SecurityException"/>
3450 </wsdl:message>
3451
3452 <wsdl:message name="DuplicateNameExceptionResponse">
3453   <wsdl:part name="fault" element="impl:DuplicateNameException"/>
3454 </wsdl:message>
3455
3456 <wsdl:message name="CCSpecValidationExceptionResponse">
3457   <wsdl:part name="fault" element="impl:CCSpecValidationException"/>
3458 </wsdl:message>
3459
3460 <wsdl:message name="InvalidURIExceptionResponse">
3461   <wsdl:part name="fault" element="impl:InvalidURIException"/>
3462 </wsdl:message>
3463
3464 <wsdl:message name="NoSuchNameExceptionResponse">
3465   <wsdl:part name="fault" element="impl:NoSuchNameException"/>
3466 </wsdl:message>
3467
3468 <wsdl:message name="NoSuchSubscriberExceptionResponse">
3469   <wsdl:part name="fault" element="impl:NoSuchSubscriberException"/>
3470 </wsdl:message>
3471
3472 <wsdl:message name="DuplicateSubscriptionExceptionResponse">
3473   <wsdl:part name="fault" element="impl:DuplicateSubscriptionException"/>
3474 </wsdl:message>
3475
3476 <wsdl:message name="ParameterExceptionResponse">
3477   <wsdl:part name="fault" element="impl:ParameterException"/>
3478 </wsdl:message>
3479
3480 <wsdl:message name="ParameterForbiddenExceptionResponse">
3481   <wsdl:part name="fault" element="impl:ParameterForbiddenException"/>
3482 </wsdl:message>
3483

```

```

3484 <wsdl:message name="ImplementationExceptionResponse">
3485 <wsdl:part name="fault" element="impl:ImplementationException"/>
3486 </wsdl:message>
3487
3488 <wsdl:message name="EPCCacheSpecValidationExceptionResponse">
3489 <wsdl:part name="fault" element="impl:EPCCacheSpecValidationException"/>
3490 </wsdl:message>
3491
3492 <wsdl:message name="InvalidPatternExceptionResponse">
3493 <wsdl:part name="fault" element="impl:InvalidPatternException"/>
3494 </wsdl:message>
3495
3496 <wsdl:message name="InUseExceptionResponse">
3497 <wsdl:part name="fault" element="impl:InUseException"/>
3498 </wsdl:message>
3499
3500 <wsdl:message name="AssocTableValidationExceptionResponse">
3501 <wsdl:part name="fault" element="impl:AssocTableValidationException"/>
3502 </wsdl:message>
3503
3504 <wsdl:message name="InvalidEPCEExceptionResponse">
3505 <wsdl:part name="fault" element="impl:InvalidEPCEException"/>
3506 </wsdl:message>
3507
3508 <wsdl:message name="InvalidAssocTableEntryExceptionResponse">
3509 <wsdl:part name="fault" element="impl:InvalidAssocTableEntryException"/>
3510 </wsdl:message>
3511
3512 <wsdl:message name="RNGValidationExceptionResponse">
3513 <wsdl:part name="fault" element="impl:RNGValidationException"/>
3514 </wsdl:message>
3515 <!-- ALECCSERVICE PORTTYPE -->
3516
3517 <wsdl:portType name="ALECCServicePortType">
3518
3519 <wsdl:operation name="define">
3520 <wsdl:input message="impl:defineRequest" name="defineRequest"/>
3521 <wsdl:output message="impl:defineResponse" name="defineResponse"/>
3522 <wsdl:fault message="impl:DuplicateNameExceptionResponse"
3523 name="DuplicateNameExceptionFault"/>
3524 <wsdl:fault message="impl:CCSpecValidationExceptionResponse"
3525 name="CCSpecValidationExceptionFault"/>
3526 <wsdl:fault message="impl:SecurityExceptionResponse"
3527 name="SecurityExceptionFault"/>
3528 <wsdl:fault message="impl:ImplementationExceptionResponse"
3529 name="ImplementationExceptionFault"/>
3530 </wsdl:operation>
3531
3532 <wsdl:operation name="undefine">
3533 <wsdl:input message="impl:undefineRequest" name="undefineRequest"/>
3534 <wsdl:output message="impl:undefineResponse" name="undefineResponse"/>
3535 <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3536 name="NoSuchNameExceptionFault"/>
3537 <wsdl:fault message="impl:SecurityExceptionResponse"
3538 name="SecurityExceptionFault"/>
3539 <wsdl:fault message="impl:ImplementationExceptionResponse"
3540 name="ImplementationExceptionFault"/>
3541 </wsdl:operation>
3542
3543 <wsdl:operation name="getCCSpec">
3544 <wsdl:input message="impl:getCCSpecRequest" name="getCCSpecRequest"/>
3545 <wsdl:output message="impl:getCCSpecResponse" name="getCCSpecResponse"/>
3546 <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3547 name="NoSuchNameExceptionFault"/>
3548 <wsdl:fault message="impl:SecurityExceptionResponse"
3549 name="SecurityExceptionFault"/>
3550 <wsdl:fault message="impl:ImplementationExceptionResponse"
3551 name="ImplementationExceptionFault"/>
3552 </wsdl:operation>
3553

```

```

3554 <wsdl:operation name="getCCSpecNames">
3555   <wsdl:input message="impl:getCCSpecNamesRequest" name="getCCSpecNamesRequest" />
3556   <wsdl:output message="impl:getCCSpecNamesResponse" name="getCCSpecNamesResponse" />
3557   <wsdl:fault message="impl:SecurityExceptionResponse"
3558     name="SecurityExceptionFault" />
3559   <wsdl:fault message="impl:ImplementationExceptionResponse"
3560     name="ImplementationExceptionFault" />
3561 </wsdl:operation>
3562
3563 <wsdl:operation name="subscribe">
3564   <wsdl:input message="impl:subscribeRequest" name="subscribeRequest" />
3565   <wsdl:output message="impl:subscribeResponse" name="subscribeResponse" />
3566   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3567     name="NoSuchNameExceptionFault" />
3568   <wsdl:fault message="impl:InvalidURIExceptionResponse"
3569     name="InvalidURIExceptionFault" />
3570   <wsdl:fault message="impl:DuplicateSubscriptionExceptionResponse"
3571     name="DuplicateSubscriptionExceptionFault" />
3572   <wsdl:fault message="impl:ParameterForbiddenExceptionResponse"
3573     name="ParameterForbiddenExceptionFault" />
3574   <wsdl:fault message="impl:SecurityExceptionResponse"
3575     name="SecurityExceptionFault" />
3576   <wsdl:fault message="impl:ImplementationExceptionResponse"
3577     name="ImplementationExceptionFault" />
3578 </wsdl:operation>
3579
3580 <wsdl:operation name="unsubscribe">
3581   <wsdl:input message="impl:unsubscribeRequest" name="unsubscribeRequest" />
3582   <wsdl:output message="impl:unsubscribeResponse" name="unsubscribeResponse" />
3583   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3584     name="NoSuchNameExceptionFault" />
3585   <wsdl:fault message="impl:NoSuchSubscriberExceptionResponse"
3586     name="NoSuchSubscriberExceptionFault" />
3587   <wsdl:fault message="impl:InvalidURIExceptionResponse"
3588     name="InvalidURIExceptionFault" />
3589   <wsdl:fault message="impl:SecurityExceptionResponse"
3590     name="SecurityExceptionFault" />
3591   <wsdl:fault message="impl:ImplementationExceptionResponse"
3592     name="ImplementationExceptionFault" />
3593 </wsdl:operation>
3594
3595 <wsdl:operation name="poll">
3596   <wsdl:input message="impl:pollRequest" name="pollRequest" />
3597   <wsdl:output message="impl:pollResponse" name="pollResponse" />
3598   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3599     name="NoSuchNameExceptionFault" />
3600   <wsdl:fault message="impl:ParameterExceptionResponse"
3601     name="ParameterExceptionFault" />
3602   <wsdl:fault message="impl:SecurityExceptionResponse"
3603     name="SecurityExceptionFault" />
3604   <wsdl:fault message="impl:ImplementationExceptionResponse"
3605     name="ImplementationExceptionFault" />
3606 </wsdl:operation>
3607
3608 <wsdl:operation name="immediate">
3609   <wsdl:input message="impl:immediateRequest" name="immediateRequest" />
3610   <wsdl:output message="impl:immediateResponse" name="immediateResponse" />
3611   <wsdl:fault message="impl:CCSpecValidationExceptionResponse"
3612     name="CCSpecValidationExceptionFault" />
3613   <wsdl:fault message="impl:ParameterForbiddenExceptionResponse"
3614     name="ParameterForbiddenExceptionFault" />
3615   <wsdl:fault message="impl:SecurityExceptionResponse"
3616     name="SecurityExceptionFault" />
3617   <wsdl:fault message="impl:ImplementationExceptionResponse"
3618     name="ImplementationExceptionFault" />
3619 </wsdl:operation>
3620
3621 <wsdl:operation name="getSubscribers">
3622   <wsdl:input message="impl:getSubscribersRequest" name="getSubscribersRequest" />
3623   <wsdl:output message="impl:getSubscribersResponse" name="getSubscribersResponse" />

```

```

3624     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3625         name="NoSuchNameExceptionFault" />
3626     <wsdl:fault message="impl:SecurityExceptionResponse"
3627         name="SecurityExceptionFault" />
3628     <wsdl:fault message="impl:ImplementationExceptionResponse"
3629         name="ImplementationExceptionFault" />
3630 </wsdl:operation>
3631
3632 <wsdl:operation name="getStandardVersion">
3633     <wsdl:input message="impl:getStandardVersionRequest"
3634         name="getStandardVersionRequest" />
3635     <wsdl:output message="impl:getStandardVersionResponse"
3636         name="getStandardVersionResponse" />
3637     <wsdl:fault message="impl:ImplementationExceptionResponse"
3638         name="ImplementationExceptionFault" />
3639 </wsdl:operation>
3640
3641 <wsdl:operation name="getVendorVersion">
3642     <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest" />
3643     <wsdl:output message="impl:getVendorVersionResponse"
3644         name="getVendorVersionResponse" />
3645     <wsdl:fault message="impl:ImplementationExceptionResponse"
3646         name="ImplementationExceptionFault" />
3647 </wsdl:operation>
3648
3649 <wsdl:operation name="defineEPCCache">
3650     <wsdl:input message="impl:defineEPCCacheRequest" name="defineEPCCacheRequest" />
3651     <wsdl:output message="impl:defineEPCCacheResponse" name="defineEPCCacheResponse" />
3652     <wsdl:fault message="impl:DuplicateNameExceptionResponse"
3653         name="DuplicateNameExceptionFault" />
3654     <wsdl:fault message="impl:EPCCacheSpecValidationExceptionResponse"
3655         name="EPCCacheSpecValidationExceptionFault" />
3656     <wsdl:fault message="impl:InvalidPatternExceptionResponse"
3657         name="InvalidPatternExceptionFault" />
3658     <wsdl:fault message="impl:SecurityExceptionResponse"
3659         name="SecurityExceptionFault" />
3660     <wsdl:fault message="impl:ImplementationExceptionResponse"
3661         name="ImplementationExceptionFault" />
3662 </wsdl:operation>
3663
3664 <wsdl:operation name="undefineEPCCache">
3665     <wsdl:input message="impl:undefineEPCCacheRequest" name="undefineEPCCacheRequest" />
3666     <wsdl:output message="impl:undefineEPCCacheResponse"
3667         name="undefineEPCCacheResponse" />
3668     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3669         name="NoSuchNameExceptionFault" />
3670     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
3671     <wsdl:fault message="impl:SecurityExceptionResponse"
3672         name="SecurityExceptionFault" />
3673     <wsdl:fault message="impl:ImplementationExceptionResponse"
3674         name="ImplementationExceptionFault" />
3675 </wsdl:operation>
3676
3677 <wsdl:operation name="getEPCCache">
3678     <wsdl:input message="impl:getEPCCacheRequest" name="getEPCCacheRequest" />
3679     <wsdl:output message="impl:getEPCCacheResponse" name="getEPCCacheResponse" />
3680     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3681         name="NoSuchNameExceptionFault" />
3682     <wsdl:fault message="impl:SecurityExceptionResponse"
3683         name="SecurityExceptionFault" />
3684     <wsdl:fault message="impl:ImplementationExceptionResponse"
3685         name="ImplementationExceptionFault" />
3686 </wsdl:operation>
3687
3688 <wsdl:operation name="getEPCCacheNames">
3689     <wsdl:input message="impl:getEPCCacheNamesRequest" name="getEPCCacheNamesRequest" />
3690     <wsdl:output message="impl:getEPCCacheNamesResponse"
3691         name="getEPCCacheNamesResponse" />
3692     <wsdl:fault message="impl:SecurityExceptionResponse"
3693         name="SecurityExceptionFault" />

```

```

3694     <wsdl:fault message="impl:ImplementationExceptionResponse"
3695             name="ImplementationExceptionFault" />
3696 </wsdl:operation>
3697
3698 <wsdl:operation name="replenishEPCCache">
3699   <wsdl:input message="impl:replenishEPCCacheRequest"
3700             name="replenishEPCCacheRequest" />
3701   <wsdl:output message="impl:replenishEPCCacheResponse"
3702              name="replenishEPCCacheResponse" />
3703   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3704              name="NoSuchNameExceptionFault" />
3705   <wsdl:fault message="impl:InvalidPatternExceptionResponse"
3706              name="InvalidPatternExceptionFault" />
3707   <wsdl:fault message="impl:SecurityExceptionResponse"
3708              name="SecurityExceptionFault" />
3709   <wsdl:fault message="impl:ImplementationExceptionResponse"
3710              name="ImplementationExceptionFault" />
3711 </wsdl:operation>
3712
3713 <wsdl:operation name="depleteEPCCache">
3714   <wsdl:input message="impl:depleteEPCCacheRequest" name="depleteEPCCacheRequest" />
3715   <wsdl:output message="impl:depleteEPCCacheResponse"
3716              name="depleteEPCCacheResponse" />
3717   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3718              name="NoSuchNameExceptionFault" />
3719   <wsdl:fault message="impl:SecurityExceptionResponse"
3720              name="SecurityExceptionFault" />
3721   <wsdl:fault message="impl:ImplementationExceptionResponse"
3722              name="ImplementationExceptionFault" />
3723 </wsdl:operation>
3724
3725 <wsdl:operation name="getEPCCacheContents">
3726   <wsdl:input message="impl:getEPCCacheContentsRequest"
3727             name="getEPCCacheContentsRequest" />
3728   <wsdl:output message="impl:getEPCCacheContentsResponse"
3729              name="getEPCCacheContentsResponse" />
3730   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3731              name="NoSuchNameExceptionFault" />
3732   <wsdl:fault message="impl:SecurityExceptionResponse"
3733              name="SecurityExceptionFault" />
3734   <wsdl:fault message="impl:ImplementationExceptionResponse"
3735              name="ImplementationExceptionFault" />
3736 </wsdl:operation>
3737
3738 <wsdl:operation name="defineAssocTable">
3739   <wsdl:input message="impl:defineAssocTableRequest" name="defineAssocTableRequest" />
3740   <wsdl:output message="impl:defineAssocTableResponse"
3741              name="defineAssocTableResponse" />
3742   <wsdl:fault message="impl:DuplicateNameExceptionResponse"
3743              name="DuplicateNameExceptionFault" />
3744   <wsdl:fault message="impl:AssocTableValidationExceptionResponse"
3745              name="AssocTableValidationExceptionFault" />
3746   <wsdl:fault message="impl:InvalidAssocTableEntryExceptionResponse"
3747              name="InvalidAssocTableEntryExceptionFault" />
3748   <wsdl:fault message="impl:SecurityExceptionResponse"
3749              name="SecurityExceptionFault" />
3750   <wsdl:fault message="impl:ImplementationExceptionResponse"
3751              name="ImplementationExceptionFault" />
3752 </wsdl:operation>
3753
3754 <wsdl:operation name="undefineAssocTable">
3755   <wsdl:input message="impl:undefineAssocTableRequest"
3756             name="undefineAssocTableRequest" />
3757   <wsdl:output message="impl:undefineAssocTableResponse"
3758              name="undefineAssocTableResponse" />
3759   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3760              name="NoSuchNameExceptionFault" />
3761   <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
3762   <wsdl:fault message="impl:SecurityExceptionResponse"
3763              name="SecurityExceptionFault" />

```

```

3764     <wsdl:fault message="impl:ImplementationExceptionResponse"
3765             name="ImplementationExceptionFault" />
3766 </wsdl:operation>
3767
3768 <wsdl:operation name="getAssocTableNames">
3769   <wsdl:input message="impl:getAssocTableNamesRequest"
3770             name="getAssocTableNamesRequest" />
3771   <wsdl:output message="impl:getAssocTableNamesResponse"
3772              name="getAssocTableNamesResponse" />
3773   <wsdl:fault message="impl:SecurityExceptionResponse"
3774             name="SecurityExceptionFault" />
3775   <wsdl:fault message="impl:ImplementationExceptionResponse"
3776             name="ImplementationExceptionFault" />
3777 </wsdl:operation>
3778
3779 <wsdl:operation name="getAssocTable">
3780   <wsdl:input message="impl:getAssocTableRequest" name="getAssocTableRequest" />
3781   <wsdl:output message="impl:getAssocTableResponse" name="getAssocTableResponse" />
3782   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3783             name="NoSuchNameExceptionFault" />
3784   <wsdl:fault message="impl:SecurityExceptionResponse"
3785             name="SecurityExceptionFault" />
3786   <wsdl:fault message="impl:ImplementationExceptionResponse"
3787             name="ImplementationExceptionFault" />
3788 </wsdl:operation>
3789
3790 <wsdl:operation name="putAssocTableEntries">
3791   <wsdl:input message="impl:putAssocTableEntriesRequest"
3792             name="putAssocTableEntriesRequest" />
3793   <wsdl:output message="impl:putAssocTableEntriesResponse"
3794             name="putAssocTableEntriesResponse" />
3795   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3796             name="NoSuchNameExceptionFault" />
3797   <wsdl:fault message="impl:InvalidAssocTableEntryExceptionResponse"
3798             name="InvalidAssocTableEntryExceptionFault" />
3799   <wsdl:fault message="impl:SecurityExceptionResponse"
3800             name="SecurityExceptionFault" />
3801   <wsdl:fault message="impl:ImplementationExceptionResponse"
3802             name="ImplementationExceptionFault" />
3803 </wsdl:operation>
3804
3805 <wsdl:operation name="getAssocTableValue">
3806   <wsdl:input message="impl:getAssocTableValueRequest"
3807             name="getAssocTableValueRequest" />
3808   <wsdl:output message="impl:getAssocTableValueResponse"
3809             name="getAssocTableValueResponse" />
3810   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3811             name="NoSuchNameExceptionFault" />
3812   <wsdl:fault message="impl:InvalidEPCEExceptionResponse"
3813             name="InvalidEPCEExceptionFault" />
3814   <wsdl:fault message="impl:SecurityExceptionResponse"
3815             name="SecurityExceptionFault" />
3816   <wsdl:fault message="impl:ImplementationExceptionResponse"
3817             name="ImplementationExceptionFault" />
3818 </wsdl:operation>
3819
3820 <wsdl:operation name="getAssocTableEntries">
3821   <wsdl:input message="impl:getAssocTableEntriesRequest"
3822             name="getAssocTableEntriesRequest" />
3823   <wsdl:output message="impl:getAssocTableEntriesResponse"
3824             name="getAssocTableEntriesResponse" />
3825   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3826             name="NoSuchNameExceptionFault" />
3827   <wsdl:fault message="impl:InvalidPatternExceptionResponse"
3828             name="InvalidPatternExceptionFault" />
3829   <wsdl:fault message="impl:SecurityExceptionResponse"
3830             name="SecurityExceptionFault" />
3831   <wsdl:fault message="impl:ImplementationExceptionResponse"
3832             name="ImplementationExceptionFault" />
3833 </wsdl:operation>

```

```

3834
3835 <wsdl:operation name="removeAssocTableEntry">
3836   <wsdl:input message="impl:removeAssocTableEntryRequest"
3837     name="removeAssocTableEntryRequest" />
3838   <wsdl:output message="impl:removeAssocTableEntryResponse"
3839     name="removeAssocTableEntryResponse" />
3840   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3841     name="NoSuchNameExceptionFault" />
3842   <wsdl:fault message="impl:InvalidEPCEExceptionResponse"
3843     name="InvalidEPCEExceptionFault" />
3844   <wsdl:fault message="impl:SecurityExceptionResponse"
3845     name="SecurityExceptionFault" />
3846   <wsdl:fault message="impl:ImplementationExceptionResponse"
3847     name="ImplementationExceptionFault" />
3848 </wsdl:operation>
3849
3850 <wsdl:operation name="removeAssocTableEntries">
3851   <wsdl:input message="impl:removeAssocTableEntriesRequest"
3852     name="removeAssocTableEntriesRequest" />
3853   <wsdl:output message="impl:removeAssocTableEntriesResponse"
3854     name="removeAssocTableEntriesResponse" />
3855   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3856     name="NoSuchNameExceptionFault" />
3857   <wsdl:fault message="impl:InvalidPatternExceptionResponse"
3858     name="InvalidPatternExceptionFault" />
3859   <wsdl:fault message="impl:SecurityExceptionResponse"
3860     name="SecurityExceptionFault" />
3861   <wsdl:fault message="impl:ImplementationExceptionResponse"
3862     name="ImplementationExceptionFault" />
3863 </wsdl:operation>
3864
3865 <wsdl:operation name="defineRNG">
3866   <wsdl:input message="impl:defineRNGRequest" name="defineRNGRequest" />
3867   <wsdl:output message="impl:defineRNGResponse" name="defineRNGResponse" />
3868   <wsdl:fault message="impl:DuplicateNameExceptionResponse"
3869     name="DuplicateNameExceptionFault" />
3870   <wsdl:fault message="impl:RNGValidationExceptionResponse"
3871     name="RNGValidationExceptionFault" />
3872   <wsdl:fault message="impl:SecurityExceptionResponse"
3873     name="SecurityExceptionFault" />
3874   <wsdl:fault message="impl:ImplementationExceptionResponse"
3875     name="ImplementationExceptionFault" />
3876 </wsdl:operation>
3877
3878 <wsdl:operation name="undefineRNG">
3879   <wsdl:input message="impl:undefineRNGRequest" name="undefineRNGRequest" />
3880   <wsdl:output message="impl:undefineRNGResponse" name="undefineRNGResponse" />
3881   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3882     name="NoSuchNameExceptionFault" />
3883   <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
3884   <wsdl:fault message="impl:SecurityExceptionResponse"
3885     name="SecurityExceptionFault" />
3886   <wsdl:fault message="impl:ImplementationExceptionResponse"
3887     name="ImplementationExceptionFault" />
3888 </wsdl:operation>
3889
3890 <wsdl:operation name="getRNGNames">
3891   <wsdl:input message="impl:getRNGNamesRequest" name="getRNGNamesRequest" />
3892   <wsdl:output message="impl:getRNGNamesResponse" name="getRNGNamesResponse" />
3893   <wsdl:fault message="impl:SecurityExceptionResponse"
3894     name="SecurityExceptionFault" />
3895   <wsdl:fault message="impl:ImplementationExceptionResponse"
3896     name="ImplementationExceptionFault" />
3897 </wsdl:operation>
3898
3899 <wsdl:operation name="getRNG">
3900   <wsdl:input message="impl:getRNGRequest" name="getRNGRequest" />
3901   <wsdl:output message="impl:getRNGResponse" name="getRNGResponse" />
3902   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3903     name="NoSuchNameExceptionFault" />

```

```

3904     <wsdl:fault message="impl:SecurityExceptionResponse"
3905         name="SecurityExceptionFault"/>
3906     <wsdl:fault message="impl:ImplementationExceptionResponse"
3907         name="ImplementationExceptionFault"/>
3908 </wsdl:operation>
3909 </wsdl:portType>
3910 <!-- ALECCSERVICE BINDING -->
3911
3912 <wsdl:binding name="ALECCServiceBinding" type="impl:ALECCServicePortType">
3913     <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
3914
3915     <wsdl:operation name="define">
3916         <wsdlsoap:operation soapAction=""/>
3917         <wsdl:input name="defineRequest">
3918             <wsdlsoap:body use="literal"/>
3919         </wsdl:input>
3920         <wsdl:output name="defineResponse">
3921             <wsdlsoap:body use="literal"/>
3922         </wsdl:output>
3923         <wsdl:fault name="DuplicateNameExceptionFault">
3924             <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
3925         </wsdl:fault>
3926         <wsdl:fault name="CCSpecValidationExceptionFault">
3927             <wsdlsoap:fault name="CCSpecValidationExceptionFault" use="literal"/>
3928         </wsdl:fault>
3929         <wsdl:fault name="SecurityExceptionFault">
3930             <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
3931         </wsdl:fault>
3932         <wsdl:fault name="ImplementationExceptionFault">
3933             <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
3934         </wsdl:fault>
3935     </wsdl:operation>
3936
3937     <wsdl:operation name="undefine">
3938         <wsdlsoap:operation soapAction=""/>
3939         <wsdl:input name="undefineRequest">
3940             <wsdlsoap:body use="literal"/>
3941         </wsdl:input>
3942         <wsdl:output name="undefineResponse">
3943             <wsdlsoap:body use="literal"/>
3944         </wsdl:output>
3945         <wsdl:fault name="NoSuchNameExceptionFault">
3946             <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
3947         </wsdl:fault>
3948         <wsdl:fault name="SecurityExceptionFault">
3949             <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
3950         </wsdl:fault>
3951         <wsdl:fault name="ImplementationExceptionFault">
3952             <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
3953         </wsdl:fault>
3954     </wsdl:operation>
3955
3956     <wsdl:operation name="getCCSpec">
3957         <wsdlsoap:operation soapAction=""/>
3958         <wsdl:input name="getCCSpecRequest">
3959             <wsdlsoap:body use="literal"/>
3960         </wsdl:input>
3961         <wsdl:output name="getCCSpecResponse">
3962             <wsdlsoap:body use="literal"/>
3963         </wsdl:output>
3964         <wsdl:fault name="NoSuchNameExceptionFault">
3965             <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
3966         </wsdl:fault>
3967         <wsdl:fault name="SecurityExceptionFault">
3968             <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
3969         </wsdl:fault>
3970         <wsdl:fault name="ImplementationExceptionFault">
3971             <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
3972         </wsdl:fault>
3973     </wsdl:operation>

```

```

3974
3975 <wsdl:operation name="getCCSpecNames">
3976   <wsdlsoap:operation soapAction="" />
3977   <wsdl:input name="getCCSpecNamesRequest">
3978     <wsdlsoap:body use="literal" />
3979   </wsdl:input>
3980   <wsdl:output name="getCCSpecNamesResponse">
3981     <wsdlsoap:body use="literal" />
3982   </wsdl:output>
3983   <wsdl:fault name="SecurityExceptionFault">
3984     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
3985   </wsdl:fault>
3986   <wsdl:fault name="ImplementationExceptionFault">
3987     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
3988   </wsdl:fault>
3989 </wsdl:operation>
3990
3991 <wsdl:operation name="subscribe">
3992   <wsdlsoap:operation soapAction="" />
3993   <wsdl:input name="subscribeRequest">
3994     <wsdlsoap:body use="literal" />
3995   </wsdl:input>
3996   <wsdl:output name="subscribeResponse">
3997     <wsdlsoap:body use="literal" />
3998   </wsdl:output>
3999   <wsdl:fault name="NoSuchNameExceptionFault">
4000     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4001   </wsdl:fault>
4002   <wsdl:fault name="InvalidURIExceptionFault">
4003     <wsdlsoap:fault name="InvalidURIExceptionFault" use="literal" />
4004   </wsdl:fault>
4005   <wsdl:fault name="DuplicateSubscriptionExceptionFault">
4006     <wsdlsoap:fault name="DuplicateSubscriptionExceptionFault" use="literal" />
4007   </wsdl:fault>
4008   <wsdl:fault name="ParameterForbiddenExceptionFault">
4009     <wsdlsoap:fault name="ParameterForbiddenExceptionFault" use="literal" />
4010   </wsdl:fault>
4011   <wsdl:fault name="SecurityExceptionFault">
4012     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4013   </wsdl:fault>
4014   <wsdl:fault name="ImplementationExceptionFault">
4015     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4016   </wsdl:fault>
4017 </wsdl:operation>
4018
4019 <wsdl:operation name="unsubscribe">
4020   <wsdlsoap:operation soapAction="" />
4021   <wsdl:input name="unsubscribeRequest">
4022     <wsdlsoap:body use="literal" />
4023   </wsdl:input>
4024   <wsdl:output name="unsubscribeResponse">
4025     <wsdlsoap:body use="literal" />
4026   </wsdl:output>
4027   <wsdl:fault name="NoSuchNameExceptionFault">
4028     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4029   </wsdl:fault>
4030   <wsdl:fault name="NoSuchSubscriberExceptionFault">
4031     <wsdlsoap:fault name="NoSuchSubscriberExceptionFault" use="literal" />
4032   </wsdl:fault>
4033   <wsdl:fault name="InvalidURIExceptionFault">
4034     <wsdlsoap:fault name="InvalidURIExceptionFault" use="literal" />
4035   </wsdl:fault>
4036   <wsdl:fault name="SecurityExceptionFault">
4037     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4038   </wsdl:fault>
4039   <wsdl:fault name="ImplementationExceptionFault">
4040     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4041   </wsdl:fault>
4042 </wsdl:operation>
4043

```

```

4044 <wsdl:operation name="poll">
4045   <wsdlsoap:operation soapAction="" />
4046   <wsdl:input name="pollRequest">
4047     <wsdlsoap:body use="literal" />
4048   </wsdl:input>
4049   <wsdl:output name="pollResponse">
4050     <wsdlsoap:body use="literal" />
4051   </wsdl:output>
4052   <wsdl:fault name="NoSuchNameExceptionFault">
4053     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4054   </wsdl:fault>
4055   <wsdl:fault name="ParameterExceptionFault">
4056     <wsdlsoap:fault name="ParameterExceptionFault" use="literal" />
4057   </wsdl:fault>
4058   <wsdl:fault name="SecurityExceptionFault">
4059     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4060   </wsdl:fault>
4061   <wsdl:fault name="ImplementationExceptionFault">
4062     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4063   </wsdl:fault>
4064 </wsdl:operation>
4065
4066 <wsdl:operation name="immediate">
4067   <wsdlsoap:operation soapAction="" />
4068   <wsdl:input name="immediateRequest">
4069     <wsdlsoap:body use="literal" />
4070   </wsdl:input>
4071   <wsdl:output name="immediateResponse">
4072     <wsdlsoap:body use="literal" />
4073   </wsdl:output>
4074   <wsdl:fault name="CCSpecValidationExceptionFault">
4075     <wsdlsoap:fault name="CCSpecValidationExceptionFault" use="literal" />
4076   </wsdl:fault>
4077   <wsdl:fault name="ParameterForbiddenExceptionFault">
4078     <wsdlsoap:fault name="ParameterForbiddenExceptionFault" use="literal" />
4079   </wsdl:fault>
4080   <wsdl:fault name="SecurityExceptionFault">
4081     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4082   </wsdl:fault>
4083   <wsdl:fault name="ImplementationExceptionFault">
4084     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4085   </wsdl:fault>
4086 </wsdl:operation>
4087
4088 <wsdl:operation name="getSubscribers">
4089   <wsdlsoap:operation soapAction="" />
4090   <wsdl:input name="getSubscribersRequest">
4091     <wsdlsoap:body use="literal" />
4092   </wsdl:input>
4093   <wsdl:output name="getSubscribersResponse">
4094     <wsdlsoap:body use="literal" />
4095   </wsdl:output>
4096   <wsdl:fault name="NoSuchNameExceptionFault">
4097     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4098   </wsdl:fault>
4099   <wsdl:fault name="SecurityExceptionFault">
4100     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4101   </wsdl:fault>
4102   <wsdl:fault name="ImplementationExceptionFault">
4103     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4104   </wsdl:fault>
4105 </wsdl:operation>
4106
4107 <wsdl:operation name="getStandardVersion">
4108   <wsdlsoap:operation soapAction="" />
4109   <wsdl:input name="getStandardVersionRequest">
4110     <wsdlsoap:body use="literal" />
4111   </wsdl:input>
4112   <wsdl:output name="getStandardVersionResponse">
4113     <wsdlsoap:body use="literal" />

```

```

4114     </wsdl:output>
4115     <wsdl:fault name="ImplementationExceptionFault">
4116         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4117     </wsdl:fault>
4118 </wsdl:operation>
4119
4120 <wsdl:operation name="getVendorVersion">
4121     <wsdlsoap:operation soapAction=""/>
4122     <wsdl:input name="getVendorVersionRequest">
4123         <wsdlsoap:body use="literal"/>
4124     </wsdl:input>
4125     <wsdl:output name="getVendorVersionResponse">
4126         <wsdlsoap:body use="literal"/>
4127     </wsdl:output>
4128     <wsdl:fault name="ImplementationExceptionFault">
4129         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4130     </wsdl:fault>
4131 </wsdl:operation>
4132
4133 <wsdl:operation name="defineEPCCache">
4134     <wsdlsoap:operation soapAction=""/>
4135     <wsdl:input name="defineEPCCacheRequest">
4136         <wsdlsoap:body use="literal"/>
4137     </wsdl:input>
4138     <wsdl:output name="defineEPCCacheResponse">
4139         <wsdlsoap:body use="literal"/>
4140     </wsdl:output>
4141     <wsdl:fault name="DuplicateNameExceptionFault">
4142         <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
4143     </wsdl:fault>
4144     <wsdl:fault name="EPCCacheSpecValidationExceptionFault">
4145         <wsdlsoap:fault name="EPCCacheSpecValidationExceptionFault" use="literal"/>
4146     </wsdl:fault>
4147     <wsdl:fault name="InvalidPatternExceptionFault">
4148         <wsdlsoap:fault name="InvalidPatternExceptionFault" use="literal"/>
4149     </wsdl:fault>
4150     <wsdl:fault name="SecurityExceptionFault">
4151         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4152     </wsdl:fault>
4153     <wsdl:fault name="ImplementationExceptionFault">
4154         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4155     </wsdl:fault>
4156 </wsdl:operation>
4157
4158 <wsdl:operation name="undefineEPCCache">
4159     <wsdlsoap:operation soapAction=""/>
4160     <wsdl:input name="undefineEPCCacheRequest">
4161         <wsdlsoap:body use="literal"/>
4162     </wsdl:input>
4163     <wsdl:output name="undefineEPCCacheResponse">
4164         <wsdlsoap:body use="literal"/>
4165     </wsdl:output>
4166     <wsdl:fault name="NoSuchNameExceptionFault">
4167         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4168     </wsdl:fault>
4169     <wsdl:fault name="InUseExceptionFault">
4170         <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
4171     </wsdl:fault>
4172     <wsdl:fault name="SecurityExceptionFault">
4173         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4174     </wsdl:fault>
4175     <wsdl:fault name="ImplementationExceptionFault">
4176         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4177     </wsdl:fault>
4178 </wsdl:operation>
4179
4180 <wsdl:operation name="getEPCCache">
4181     <wsdlsoap:operation soapAction=""/>
4182     <wsdl:input name="getEPCCacheRequest">
4183         <wsdlsoap:body use="literal"/>

```

```

4184     </wsdl:input>
4185     <wsdl:output name="getEPCCacheResponse">
4186       <wsdlsoap:body use="literal"/>
4187     </wsdl:output>
4188     <wsdl:fault name="NoSuchNameExceptionFault">
4189       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4190     </wsdl:fault>
4191     <wsdl:fault name="SecurityExceptionFault">
4192       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4193     </wsdl:fault>
4194     <wsdl:fault name="ImplementationExceptionFault">
4195       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4196     </wsdl:fault>
4197   </wsdl:operation>
4198
4199   <wsdl:operation name="getEPCCacheNames">
4200     <wsdlsoap:operation soapAction=""/>
4201     <wsdl:input name="getEPCCacheNamesRequest">
4202       <wsdlsoap:body use="literal"/>
4203     </wsdl:input>
4204     <wsdl:output name="getEPCCacheNamesResponse">
4205       <wsdlsoap:body use="literal"/>
4206     </wsdl:output>
4207     <wsdl:fault name="SecurityExceptionFault">
4208       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4209     </wsdl:fault>
4210     <wsdl:fault name="ImplementationExceptionFault">
4211       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4212     </wsdl:fault>
4213   </wsdl:operation>
4214
4215   <wsdl:operation name="replenishEPCCache">
4216     <wsdlsoap:operation soapAction=""/>
4217     <wsdl:input name="replenishEPCCacheRequest">
4218       <wsdlsoap:body use="literal"/>
4219     </wsdl:input>
4220     <wsdl:output name="replenishEPCCacheResponse">
4221       <wsdlsoap:body use="literal"/>
4222     </wsdl:output>
4223     <wsdl:fault name="NoSuchNameExceptionFault">
4224       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4225     </wsdl:fault>
4226     <wsdl:fault name="InvalidPatternExceptionFault">
4227       <wsdlsoap:fault name="InvalidPatternExceptionFault" use="literal"/>
4228     </wsdl:fault>
4229     <wsdl:fault name="SecurityExceptionFault">
4230       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4231     </wsdl:fault>
4232     <wsdl:fault name="ImplementationExceptionFault">
4233       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4234     </wsdl:fault>
4235   </wsdl:operation>
4236
4237   <wsdl:operation name="depleteEPCCache">
4238     <wsdlsoap:operation soapAction=""/>
4239     <wsdl:input name="depleteEPCCacheRequest">
4240       <wsdlsoap:body use="literal"/>
4241     </wsdl:input>
4242     <wsdl:output name="depleteEPCCacheResponse">
4243       <wsdlsoap:body use="literal"/>
4244     </wsdl:output>
4245     <wsdl:fault name="NoSuchNameExceptionFault">
4246       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4247     </wsdl:fault>
4248     <wsdl:fault name="SecurityExceptionFault">
4249       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4250     </wsdl:fault>
4251     <wsdl:fault name="ImplementationExceptionFault">
4252       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4253     </wsdl:fault>

```

```

4254 </wsdl:operation>
4255
4256 <wsdl:operation name="getEPCCacheContents">
4257 <wsdlsoap:operation soapAction="" />
4258 <wsdl:input name="getEPCCacheContentsRequest">
4259 <wsdlsoap:body use="literal" />
4260 </wsdl:input>
4261 <wsdl:output name="getEPCCacheContentsResponse">
4262 <wsdlsoap:body use="literal" />
4263 </wsdl:output>
4264 <wsdl:fault name="NoSuchNameExceptionFault">
4265 <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4266 </wsdl:fault>
4267 <wsdl:fault name="SecurityExceptionFault">
4268 <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4269 </wsdl:fault>
4270 <wsdl:fault name="ImplementationExceptionFault">
4271 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4272 </wsdl:fault>
4273 </wsdl:operation>
4274
4275 <wsdl:operation name="defineAssocTable">
4276 <wsdlsoap:operation soapAction="" />
4277 <wsdl:input name="defineAssocTableRequest">
4278 <wsdlsoap:body use="literal" />
4279 </wsdl:input>
4280 <wsdl:output name="defineAssocTableResponse">
4281 <wsdlsoap:body use="literal" />
4282 </wsdl:output>
4283 <wsdl:fault name="DuplicateNameExceptionFault">
4284 <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal" />
4285 </wsdl:fault>
4286 <wsdl:fault name="AssocTableValidationExceptionFault">
4287 <wsdlsoap:fault name="AssocTableValidationExceptionFault" use="literal" />
4288 </wsdl:fault>
4289 <wsdl:fault name="InvalidAssocTableEntryExceptionFault">
4290 <wsdlsoap:fault name="InvalidAssocTableEntryExceptionFault" use="literal" />
4291 </wsdl:fault>
4292 <wsdl:fault name="SecurityExceptionFault">
4293 <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4294 </wsdl:fault>
4295 <wsdl:fault name="ImplementationExceptionFault">
4296 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4297 </wsdl:fault>
4298 </wsdl:operation>
4299
4300 <wsdl:operation name="undefineAssocTable">
4301 <wsdlsoap:operation soapAction="" />
4302 <wsdl:input name="undefineAssocTableRequest">
4303 <wsdlsoap:body use="literal" />
4304 </wsdl:input>
4305 <wsdl:output name="undefineAssocTableResponse">
4306 <wsdlsoap:body use="literal" />
4307 </wsdl:output>
4308 <wsdl:fault name="NoSuchNameExceptionFault">
4309 <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4310 </wsdl:fault>
4311 <wsdl:fault name="InUseExceptionFault">
4312 <wsdlsoap:fault name="InUseExceptionFault" use="literal" />
4313 </wsdl:fault>
4314 <wsdl:fault name="SecurityExceptionFault">
4315 <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4316 </wsdl:fault>
4317 <wsdl:fault name="ImplementationExceptionFault">
4318 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4319 </wsdl:fault>
4320 </wsdl:operation>
4321
4322 <wsdl:operation name="getAssocTableNames">
4323 <wsdlsoap:operation soapAction="" />

```

```

4324     <wsdl:input name="getAssocTableNamesRequest">
4325         <wsdlsoap:body use="literal"/>
4326     </wsdl:input>
4327     <wsdl:output name="getAssocTableNamesResponse">
4328         <wsdlsoap:body use="literal"/>
4329     </wsdl:output>
4330     <wsdl:fault name="SecurityExceptionFault">
4331         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4332     </wsdl:fault>
4333     <wsdl:fault name="ImplementationExceptionFault">
4334         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4335     </wsdl:fault>
4336 </wsdl:operation>
4337
4338 <wsdl:operation name="getAssocTable">
4339     <wsdlsoap:operation soapAction=""/>
4340     <wsdl:input name="getAssocTableRequest">
4341         <wsdlsoap:body use="literal"/>
4342     </wsdl:input>
4343     <wsdl:output name="getAssocTableResponse">
4344         <wsdlsoap:body use="literal"/>
4345     </wsdl:output>
4346     <wsdl:fault name="NoSuchNameExceptionFault">
4347         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4348     </wsdl:fault>
4349     <wsdl:fault name="SecurityExceptionFault">
4350         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4351     </wsdl:fault>
4352     <wsdl:fault name="ImplementationExceptionFault">
4353         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4354     </wsdl:fault>
4355 </wsdl:operation>
4356
4357 <wsdl:operation name="putAssocTableEntries">
4358     <wsdlsoap:operation soapAction=""/>
4359     <wsdl:input name="putAssocTableEntriesRequest">
4360         <wsdlsoap:body use="literal"/>
4361     </wsdl:input>
4362     <wsdl:output name="putAssocTableEntriesResponse">
4363         <wsdlsoap:body use="literal"/>
4364     </wsdl:output>
4365     <wsdl:fault name="NoSuchNameExceptionFault">
4366         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4367     </wsdl:fault>
4368     <wsdl:fault name="InvalidAssocTableEntryExceptionFault">
4369         <wsdlsoap:fault name="InvalidAssocTableEntryExceptionFault" use="literal"/>
4370     </wsdl:fault>
4371     <wsdl:fault name="SecurityExceptionFault">
4372         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4373     </wsdl:fault>
4374     <wsdl:fault name="ImplementationExceptionFault">
4375         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4376     </wsdl:fault>
4377 </wsdl:operation>
4378
4379 <wsdl:operation name="getAssocTableValue">
4380     <wsdlsoap:operation soapAction=""/>
4381     <wsdl:input name="getAssocTableValueRequest">
4382         <wsdlsoap:body use="literal"/>
4383     </wsdl:input>
4384     <wsdl:output name="getAssocTableValueResponse">
4385         <wsdlsoap:body use="literal"/>
4386     </wsdl:output>
4387     <wsdl:fault name="NoSuchNameExceptionFault">
4388         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4389     </wsdl:fault>
4390     <wsdl:fault name="InvalidEPCEExceptionFault">
4391         <wsdlsoap:fault name="InvalidEPCEExceptionFault" use="literal"/>
4392     </wsdl:fault>
4393     <wsdl:fault name="SecurityExceptionFault">

```

```

4394     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4395 </wsdl:fault>
4396 <wsdl:fault name="ImplementationExceptionFault">
4397     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4398 </wsdl:fault>
4399 </wsdl:operation>
4400
4401 <wsdl:operation name="getAssocTableEntries">
4402     <wsdlsoap:operation soapAction=""/>
4403     <wsdl:input name="getAssocTableEntriesRequest">
4404         <wsdlsoap:body use="literal"/>
4405     </wsdl:input>
4406     <wsdl:output name="getAssocTableEntriesResponse">
4407         <wsdlsoap:body use="literal"/>
4408     </wsdl:output>
4409     <wsdl:fault name="NoSuchNameExceptionFault">
4410         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4411     </wsdl:fault>
4412     <wsdl:fault name="InvalidPatternExceptionFault">
4413         <wsdlsoap:fault name="InvalidPatternExceptionFault" use="literal"/>
4414     </wsdl:fault>
4415     <wsdl:fault name="SecurityExceptionFault">
4416         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4417     </wsdl:fault>
4418     <wsdl:fault name="ImplementationExceptionFault">
4419         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4420     </wsdl:fault>
4421 </wsdl:operation>
4422
4423 <wsdl:operation name="removeAssocTableEntry">
4424     <wsdlsoap:operation soapAction=""/>
4425     <wsdl:input name="removeAssocTableEntryRequest">
4426         <wsdlsoap:body use="literal"/>
4427     </wsdl:input>
4428     <wsdl:output name="removeAssocTableEntryResponse">
4429         <wsdlsoap:body use="literal"/>
4430     </wsdl:output>
4431     <wsdl:fault name="NoSuchNameExceptionFault">
4432         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4433     </wsdl:fault>
4434     <wsdl:fault name="InvalidEPCEExceptionFault">
4435         <wsdlsoap:fault name="InvalidEPCEExceptionFault" use="literal"/>
4436     </wsdl:fault>
4437     <wsdl:fault name="SecurityExceptionFault">
4438         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4439     </wsdl:fault>
4440     <wsdl:fault name="ImplementationExceptionFault">
4441         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4442     </wsdl:fault>
4443 </wsdl:operation>
4444
4445 <wsdl:operation name="removeAssocTableEntries">
4446     <wsdlsoap:operation soapAction=""/>
4447     <wsdl:input name="removeAssocTableEntriesRequest">
4448         <wsdlsoap:body use="literal"/>
4449     </wsdl:input>
4450     <wsdl:output name="removeAssocTableEntriesResponse">
4451         <wsdlsoap:body use="literal"/>
4452     </wsdl:output>
4453     <wsdl:fault name="NoSuchNameExceptionFault">
4454         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4455     </wsdl:fault>
4456     <wsdl:fault name="InvalidPatternExceptionFault">
4457         <wsdlsoap:fault name="InvalidPatternExceptionFault" use="literal"/>
4458     </wsdl:fault>
4459     <wsdl:fault name="SecurityExceptionFault">
4460         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4461     </wsdl:fault>
4462     <wsdl:fault name="ImplementationExceptionFault">
4463         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>

```

```

4464     </wsdl:fault>
4465 </wsdl:operation>
4466
4467 <wsdl:operation name="defineRNG">
4468   <wsdlsoap:operation soapAction=""/>
4469   <wsdl:input name="defineRNGRequest">
4470     <wsdlsoap:body use="literal"/>
4471   </wsdl:input>
4472   <wsdl:output name="defineRNGResponse">
4473     <wsdlsoap:body use="literal"/>
4474   </wsdl:output>
4475   <wsdl:fault name="DuplicateNameExceptionFault">
4476     <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
4477   </wsdl:fault>
4478   <wsdl:fault name="RNGValidationExceptionFault">
4479     <wsdlsoap:fault name="RNGValidationExceptionFault" use="literal"/>
4480   </wsdl:fault>
4481   <wsdl:fault name="SecurityExceptionFault">
4482     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4483   </wsdl:fault>
4484   <wsdl:fault name="ImplementationExceptionFault">
4485     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4486   </wsdl:fault>
4487 </wsdl:operation>
4488
4489 <wsdl:operation name="undefineRNG">
4490   <wsdlsoap:operation soapAction=""/>
4491   <wsdl:input name="undefineRNGRequest">
4492     <wsdlsoap:body use="literal"/>
4493   </wsdl:input>
4494   <wsdl:output name="undefineRNGResponse">
4495     <wsdlsoap:body use="literal"/>
4496   </wsdl:output>
4497   <wsdl:fault name="NoSuchNameExceptionFault">
4498     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4499   </wsdl:fault>
4500   <wsdl:fault name="InUseExceptionFault">
4501     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
4502   </wsdl:fault>
4503   <wsdl:fault name="SecurityExceptionFault">
4504     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4505   </wsdl:fault>
4506   <wsdl:fault name="ImplementationExceptionFault">
4507     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4508   </wsdl:fault>
4509 </wsdl:operation>
4510
4511 <wsdl:operation name="getRNGNames">
4512   <wsdlsoap:operation soapAction=""/>
4513   <wsdl:input name="getRNGNamesRequest">
4514     <wsdlsoap:body use="literal"/>
4515   </wsdl:input>
4516   <wsdl:output name="getRNGNamesResponse">
4517     <wsdlsoap:body use="literal"/>
4518   </wsdl:output>
4519   <wsdl:fault name="SecurityExceptionFault">
4520     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4521   </wsdl:fault>
4522   <wsdl:fault name="ImplementationExceptionFault">
4523     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4524   </wsdl:fault>
4525 </wsdl:operation>
4526
4527 <wsdl:operation name="getRNG">
4528   <wsdlsoap:operation soapAction=""/>
4529   <wsdl:input name="getRNGRequest">
4530     <wsdlsoap:body use="literal"/>
4531   </wsdl:input>
4532   <wsdl:output name="getRNGResponse">
4533     <wsdlsoap:body use="literal"/>

```

```

4534     </wsdl:output>
4535     <wsdl:fault name="NoSuchNameExceptionFault">
4536       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4537     </wsdl:fault>
4538     <wsdl:fault name="SecurityExceptionFault">
4539       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4540     </wsdl:fault>
4541     <wsdl:fault name="ImplementationExceptionFault">
4542       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4543     </wsdl:fault>
4544   </wsdl:operation>
4545 </wsdl:binding>
4546
4547 <!-- ALECCSERVICE -->
4548 <wsdl:service name="ALECCService">
4549   <wsdl:port binding="impl:ALECCServiceBinding" name="ALECCServicePort">
4550     <!-- The value of the location attribute below is an example only;
4551           Implementations are free to choose any appropriate URL. -->
4552     <wsdlsoap:address location="http://localhost:8080/services/ALECCService"/>
4553   </wsdl:port>
4554 </wsdl:service>
4555 </wsdl:definitions>

```

## 4556 4.5 ALE Tag Memory API SOAP Binding

4557 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
4558 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Tag  
4559 Memory API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0  
4560 [WSI].

```

4561 <?xml version="1.0" encoding="UTF-8"?>
4562 <!-- ALETMSERVICE DEFINITIONS -->
4563 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4564                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
4565                  xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
4566                  xmlns:ale="urn:epcglobal:ale:xsd:1"
4567                  xmlns:impl="urn:epcglobal:aletm:wsdl:1"
4568                  targetNamespace="urn:epcglobal:aletm:wsdl:1">
4569   <!-- ALETMSERVICE TYPES -->
4570   <wsdl:types>
4571     <xsd:schema targetNamespace="urn:epcglobal:aletm:wsdl:1">
4572       <xsd:import namespace="urn:epcglobal:ale:xsd:1"
4573                 schemaLocation="EPCglobal-ale-1_1-aletm.xsd"/>
4574       <!-- ALETMSERVICE MESSAGE WRAPPERS -->
4575
4576       <xsd:element name="DefineTMSpec" type="impl:DefineTMSpec"/>
4577       <xsd:complexType name="DefineTMSpec">
4578         <xsd:sequence>
4579           <xsd:element name="specName" type="xsd:string"/>
4580           <xsd:element name="spec" type="ale:TMSpec"/>
4581         </xsd:sequence>
4582       </xsd:complexType>
4583       <xsd:element name="DefineTMSpecResult">
4584         <xsd:complexType/>
4585       </xsd:element>
4586
4587       <xsd:element name="UndefineTMSpec" type="impl:UndefineTMSpec"/>
4588       <xsd:complexType name="UndefineTMSpec">
4589         <xsd:sequence>
4590           <xsd:element name="specName" type="xsd:string"/>
4591         </xsd:sequence>
4592       </xsd:complexType>
4593       <xsd:element name="UndefineTMSpecResult">
4594         <xsd:complexType/>
4595       </xsd:element>
4596
4597       <xsd:element name="GetTMSpec" type="impl:GetTMSpec"/>
4598       <xsd:complexType name="GetTMSpec">

```

```

4599     <xsd:sequence>
4600         <xsd:element name="specName" type="xsd:string"/>
4601     </xsd:sequence>
4602 </xsd:complexType>
4603 <xsd:element name="GetTMSpecResult" type="ale:TMSpec"/>
4604
4605 <xsd:element name="GetTMSpecNames" type="impl:EmptyParms"/>
4606 <xsd:element name="GetTMSpecNamesResult" type="impl:ArrayOfString"/>
4607
4608 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
4609 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
4610
4611 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
4612 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
4613
4614 <xsd:element name="ALEException" type="impl:ALEException"/>
4615 <xsd:complexType name="ALEException">
4616     <xsd:sequence>
4617         <xsd:element name="reason" type="xsd:string"/>
4618     </xsd:sequence>
4619 </xsd:complexType>
4620
4621 <xsd:element name="SecurityException" type="impl:SecurityException"/>
4622 <xsd:complexType name="SecurityException">
4623     <xsd:complexContent>
4624         <xsd:extension base="impl:ALEException"/>
4625     </xsd:complexContent>
4626 </xsd:complexType>
4627
4628 <xsd:element name="DuplicateNameException" type="impl:DuplicateNameException"/>
4629 <xsd:complexType name="DuplicateNameException">
4630     <xsd:complexContent>
4631         <xsd:extension base="impl:ALEException"/>
4632     </xsd:complexContent>
4633 </xsd:complexType>
4634
4635 <xsd:element name="TMSpecValidationException"
4636     type="impl:TMSpecValidationException"/>
4637 <xsd:complexType name="TMSpecValidationException">
4638     <xsd:complexContent>
4639         <xsd:extension base="impl:ALEException"/>
4640     </xsd:complexContent>
4641 </xsd:complexType>
4642
4643 <xsd:element name="NoSuchNameException" type="impl:NoSuchNameException"/>
4644 <xsd:complexType name="NoSuchNameException">
4645     <xsd:complexContent>
4646         <xsd:extension base="impl:ALEException"/>
4647     </xsd:complexContent>
4648 </xsd:complexType>
4649
4650 <xsd:element name="InUseException" type="impl:InUseException"/>
4651 <xsd:complexType name="InUseException">
4652     <xsd:complexContent>
4653         <xsd:extension base="impl:ALEException"/>
4654     </xsd:complexContent>
4655 </xsd:complexType>
4656
4657 <xsd:element name="ImplementationException" type="impl:ImplementationException"/>
4658 <xsd:complexType name="ImplementationException">
4659     <xsd:complexContent>
4660         <xsd:extension base="impl:ALEException">
4661             <xsd:sequence>
4662                 <xsd:element name="severity" type="impl:ImplementationExceptionSeverity"/>
4663             </xsd:sequence>
4664         </xsd:extension>
4665     </xsd:complexContent>
4666 </xsd:complexType>
4667
4668 <xsd:complexType name="ArrayOfString">

```

```

4669         <xsd:sequence>
4670             <xsd:element name="string" type="xsd:string" minOccurs="0"
4671                 maxOccurs="unbounded" />
4672         </xsd:sequence>
4673     </xsd:complexType>
4674
4675     <!-- The ImplementationExceptionSeverity type is an enumerated type.
4676         The following strings are legal values for this type:
4677         ERROR
4678         SEVERE
4679     -->
4680     <xsd:simpleType name="ImplementationExceptionSeverity">
4681         <xsd:restriction base="xsd:string" />
4682     </xsd:simpleType>
4683
4684     <xsd:complexType name="EmptyParms" />
4685 </xsd:schema>
4686 </wsdl:types>
4687 <!-- ALETMSERVICE MESSAGES -->
4688
4689 <wsdl:message name="defineTMSpecRequest">
4690     <wsdl:part name="parms" element="impl:DefineTMSpec" />
4691 </wsdl:message>
4692 <wsdl:message name="defineTMSpecResponse">
4693     <wsdl:part name="defineTMSpecReturn" element="impl:DefineTMSpecResult" />
4694 </wsdl:message>
4695
4696 <wsdl:message name="undefineTMSpecRequest">
4697     <wsdl:part name="parms" element="impl:UndefineTMSpec" />
4698 </wsdl:message>
4699 <wsdl:message name="undefineTMSpecResponse">
4700     <wsdl:part name="undefineTMSpecReturn" element="impl:UndefineTMSpecResult" />
4701 </wsdl:message>
4702
4703 <wsdl:message name="getTMSpecRequest">
4704     <wsdl:part name="parms" element="impl:GetTMSpec" />
4705 </wsdl:message>
4706 <wsdl:message name="getTMSpecResponse">
4707     <wsdl:part name="getTMSpecReturn" element="impl:GetTMSpecResult" />
4708 </wsdl:message>
4709
4710 <wsdl:message name="getTMSpecNamesRequest">
4711     <wsdl:part name="parms" element="impl:GetTMSpecNames" />
4712 </wsdl:message>
4713 <wsdl:message name="getTMSpecNamesResponse">
4714     <wsdl:part name="getTMSpecNamesReturn" element="impl:GetTMSpecNamesResult" />
4715 </wsdl:message>
4716
4717 <wsdl:message name="getStandardVersionRequest">
4718     <wsdl:part name="parms" element="impl:GetStandardVersion" />
4719 </wsdl:message>
4720 <wsdl:message name="getStandardVersionResponse">
4721     <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult" />
4722 </wsdl:message>
4723
4724 <wsdl:message name="getVendorVersionRequest">
4725     <wsdl:part name="parms" element="impl:GetVendorVersion" />
4726 </wsdl:message>
4727 <wsdl:message name="getVendorVersionResponse">
4728     <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult" />
4729 </wsdl:message>
4730
4731 <wsdl:message name="SecurityExceptionResponse">
4732     <wsdl:part name="fault" element="impl:SecurityException" />
4733 </wsdl:message>
4734
4735 <wsdl:message name="DuplicateNameExceptionResponse">
4736     <wsdl:part name="fault" element="impl:DuplicateNameException" />
4737 </wsdl:message>
4738

```

```

4739 <wsdl:message name="InUseExceptionResponse">
4740 <wsdl:part name="fault" element="impl:InUseException"/>
4741 </wsdl:message>
4742
4743 <wsdl:message name="TMSpecValidationExceptionResponse">
4744 <wsdl:part name="fault" element="impl:TMSpecValidationException"/>
4745 </wsdl:message>
4746
4747 <wsdl:message name="NoSuchNameExceptionResponse">
4748 <wsdl:part name="fault" element="impl:NoSuchNameException"/>
4749 </wsdl:message>
4750
4751 <wsdl:message name="ImplementationExceptionResponse">
4752 <wsdl:part name="fault" element="impl:ImplementationException"/>
4753 </wsdl:message>
4754 <!-- ALETMSERVICE PORTTYPE -->
4755
4756 <wsdl:portType name="ALETMSERVICEPortType">
4757
4758 <wsdl:operation name="defineTMSpec">
4759 <wsdl:input message="impl:defineTMSpecRequest" name="defineTMSpecRequest"/>
4760 <wsdl:output message="impl:defineTMSpecResponse" name="defineTMSpecResponse"/>
4761 <wsdl:fault message="impl:DuplicateNameExceptionResponse"
4762 name="DuplicateNameExceptionFault"/>
4763 <wsdl:fault message="impl:TMSpecValidationExceptionResponse"
4764 name="TMSpecValidationExceptionFault"/>
4765 <wsdl:fault message="impl:SecurityExceptionResponse"
4766 name="SecurityExceptionFault"/>
4767 <wsdl:fault message="impl:ImplementationExceptionResponse"
4768 name="ImplementationExceptionFault"/>
4769 </wsdl:operation>
4770
4771 <wsdl:operation name="undefineTMSpec">
4772 <wsdl:input message="impl:undefineTMSpecRequest" name="undefineTMSpecRequest"/>
4773 <wsdl:output message="impl:undefineTMSpecResponse" name="undefineTMSpecResponse"/>
4774 <wsdl:fault message="impl:NoSuchNameExceptionResponse"
4775 name="NoSuchNameExceptionFault"/>
4776 <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault"/>
4777 <wsdl:fault message="impl:SecurityExceptionResponse"
4778 name="SecurityExceptionFault"/>
4779 <wsdl:fault message="impl:ImplementationExceptionResponse"
4780 name="ImplementationExceptionFault"/>
4781 </wsdl:operation>
4782
4783 <wsdl:operation name="getTMSpec">
4784 <wsdl:input message="impl:getTMSpecRequest" name="getTMSpecRequest"/>
4785 <wsdl:output message="impl:getTMSpecResponse" name="getTMSpecResponse"/>
4786 <wsdl:fault message="impl:NoSuchNameExceptionResponse"
4787 name="NoSuchNameExceptionFault"/>
4788 <wsdl:fault message="impl:SecurityExceptionResponse"
4789 name="SecurityExceptionFault"/>
4790 <wsdl:fault message="impl:ImplementationExceptionResponse"
4791 name="ImplementationExceptionFault"/>
4792 </wsdl:operation>
4793
4794 <wsdl:operation name="getTMSpecNames">
4795 <wsdl:input message="impl:getTMSpecNamesRequest" name="getTMSpecNamesRequest"/>
4796 <wsdl:output message="impl:getTMSpecNamesResponse" name="getTMSpecNamesResponse"/>
4797 <wsdl:fault message="impl:SecurityExceptionResponse"
4798 name="SecurityExceptionFault"/>
4799 <wsdl:fault message="impl:ImplementationExceptionResponse"
4800 name="ImplementationExceptionFault"/>
4801 </wsdl:operation>
4802
4803 <wsdl:operation name="getStandardVersion">
4804 <wsdl:input message="impl:getStandardVersionRequest"
4805 name="getStandardVersionRequest"/>
4806 <wsdl:output message="impl:getStandardVersionResponse"
4807 name="getStandardVersionResponse"/>
4808 <wsdl:fault message="impl:ImplementationExceptionResponse"

```

```

4809         name="ImplementationExceptionFault"/>
4810     </wsdl:operation>
4811
4812     <wsdl:operation name="getVendorVersion">
4813         <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest" />
4814         <wsdl:output message="impl:getVendorVersionResponse"
4815             name="getVendorVersionResponse" />
4816         <wsdl:fault message="impl:ImplementationExceptionResponse"
4817             name="ImplementationExceptionFault" />
4818     </wsdl:operation>
4819 </wsdl:portType>
4820 <!-- ALETMSERVICE BINDING -->
4821
4822 <wsdl:binding name="ALETMServiceBinding" type="impl:ALETMServicePortType">
4823     <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
4824
4825     <wsdl:operation name="defineTMSpec">
4826         <wsdlsoap:operation soapAction="" />
4827         <wsdl:input name="defineTMSpecRequest">
4828             <wsdlsoap:body use="literal" />
4829         </wsdl:input>
4830         <wsdl:output name="defineTMSpecResponse">
4831             <wsdlsoap:body use="literal" />
4832         </wsdl:output>
4833         <wsdl:fault name="DuplicateNameExceptionFault">
4834             <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal" />
4835         </wsdl:fault>
4836         <wsdl:fault name="TMSpecValidationExceptionFault">
4837             <wsdlsoap:fault name="TMSpecValidationExceptionFault" use="literal" />
4838         </wsdl:fault>
4839         <wsdl:fault name="SecurityExceptionFault">
4840             <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4841         </wsdl:fault>
4842         <wsdl:fault name="ImplementationExceptionFault">
4843             <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4844         </wsdl:fault>
4845     </wsdl:operation>
4846
4847     <wsdl:operation name="undefineTMSpec">
4848         <wsdlsoap:operation soapAction="" />
4849         <wsdl:input name="undefineTMSpecRequest">
4850             <wsdlsoap:body use="literal" />
4851         </wsdl:input>
4852         <wsdl:output name="undefineTMSpecResponse">
4853             <wsdlsoap:body use="literal" />
4854         </wsdl:output>
4855         <wsdl:fault name="NoSuchNameExceptionFault">
4856             <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4857         </wsdl:fault>
4858         <wsdl:fault name="InUseExceptionFault">
4859             <wsdlsoap:fault name="InUseExceptionFault" use="literal" />
4860         </wsdl:fault>
4861         <wsdl:fault name="SecurityExceptionFault">
4862             <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4863         </wsdl:fault>
4864         <wsdl:fault name="ImplementationExceptionFault">
4865             <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4866         </wsdl:fault>
4867     </wsdl:operation>
4868
4869     <wsdl:operation name="getTMSpec">
4870         <wsdlsoap:operation soapAction="" />
4871         <wsdl:input name="getTMSpecRequest">
4872             <wsdlsoap:body use="literal" />
4873         </wsdl:input>
4874         <wsdl:output name="getTMSpecResponse">
4875             <wsdlsoap:body use="literal" />
4876         </wsdl:output>
4877         <wsdl:fault name="NoSuchNameExceptionFault">
4878             <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />

```

```

4879     </wsdl:fault>
4880     <wsdl:fault name="SecurityExceptionFault">
4881         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4882     </wsdl:fault>
4883     <wsdl:fault name="ImplementationExceptionFault">
4884         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4885     </wsdl:fault>
4886 </wsdl:operation>
4887
4888 <wsdl:operation name="getTMSpecNames">
4889     <wsdlsoap:operation soapAction=""/>
4890     <wsdl:input name="getTMSpecNamesRequest">
4891         <wsdlsoap:body use="literal"/>
4892     </wsdl:input>
4893     <wsdl:output name="getTMSpecNamesResponse">
4894         <wsdlsoap:body use="literal"/>
4895     </wsdl:output>
4896     <wsdl:fault name="SecurityExceptionFault">
4897         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4898     </wsdl:fault>
4899     <wsdl:fault name="ImplementationExceptionFault">
4900         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4901     </wsdl:fault>
4902 </wsdl:operation>
4903
4904 <wsdl:operation name="getStandardVersion">
4905     <wsdlsoap:operation soapAction=""/>
4906     <wsdl:input name="getStandardVersionRequest">
4907         <wsdlsoap:body use="literal"/>
4908     </wsdl:input>
4909     <wsdl:output name="getStandardVersionResponse">
4910         <wsdlsoap:body use="literal"/>
4911     </wsdl:output>
4912     <wsdl:fault name="ImplementationExceptionFault">
4913         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4914     </wsdl:fault>
4915 </wsdl:operation>
4916
4917 <wsdl:operation name="getVendorVersion">
4918     <wsdlsoap:operation soapAction=""/>
4919     <wsdl:input name="getVendorVersionRequest">
4920         <wsdlsoap:body use="literal"/>
4921     </wsdl:input>
4922     <wsdl:output name="getVendorVersionResponse">
4923         <wsdlsoap:body use="literal"/>
4924     </wsdl:output>
4925     <wsdl:fault name="ImplementationExceptionFault">
4926         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4927     </wsdl:fault>
4928 </wsdl:operation>
4929 </wsdl:binding>
4930
4931 <!-- ALETMSERVICE -->
4932 <wsdl:service name="ALETMService">
4933     <wsdl:port binding="impl:ALETMServiceBinding" name="ALETMServicePort">
4934         <!-- The value of the location attribute below is an example only;
4935             Implementations are free to choose any appropriate URL. -->
4936         <wsdlsoap:address location="http://localhost:8080/services/ALETMService"/>
4937     </wsdl:port>
4938 </wsdl:service>
4939 </wsdl:definitions>

```

## 4940 4.6 ALE Logical Reader API SOAP Binding

4941 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
4942 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Logical

4943 Reader API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0  
4944 [WSI].

```
4945 <?xml version="1.0" encoding="UTF-8"?>
4946 <!-- ALELRSERVICE DEFINITIONS -->
4947 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4948     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
4949     xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
4950     xmlns:ale="urn:epcglobal:ale:xsd:1"
4951     xmlns:impl="urn:epcglobal:alelr:wsdl:1"
4952     targetNamespace="urn:epcglobal:alelr:wsdl:1">
4953 <!-- ALELRSERVICE TYPES -->
4954 <wsdl:types>
4955   <xsd:schema targetNamespace="urn:epcglobal:alelr:wsdl:1">
4956     <xsd:import namespace="urn:epcglobal:ale:xsd:1"
4957       schemaLocation="EPCglobal-ale-1_1-alelr.xsd"/>
4958     <!-- ALELRSERVICE MESSAGE WRAPPERS -->
4959
4960     <xsd:element name="Define" type="impl:Define"/>
4961     <xsd:complexType name="Define">
4962       <xsd:sequence>
4963         <xsd:element name="name" type="xsd:string"/>
4964         <xsd:element name="spec" type="ale:LRSpec"/>
4965       </xsd:sequence>
4966     </xsd:complexType>
4967     <xsd:element name="DefineResult">
4968       <xsd:complexType/>
4969     </xsd:element>
4970
4971     <xsd:element name="Update" type="impl:Update"/>
4972     <xsd:complexType name="Update">
4973       <xsd:sequence>
4974         <xsd:element name="name" type="xsd:string"/>
4975         <xsd:element name="spec" type="ale:LRSpec"/>
4976       </xsd:sequence>
4977     </xsd:complexType>
4978     <xsd:element name="UpdateResult">
4979       <xsd:complexType/>
4980     </xsd:element>
4981
4982     <xsd:element name="Undefine" type="impl:Undefine"/>
4983     <xsd:complexType name="Undefine">
4984       <xsd:sequence>
4985         <xsd:element name="name" type="xsd:string"/>
4986       </xsd:sequence>
4987     </xsd:complexType>
4988     <xsd:element name="UndefineResult">
4989       <xsd:complexType/>
4990     </xsd:element>
4991
4992     <xsd:element name="GetLogicalReaderNames" type="impl:EmptyParms"/>
4993     <xsd:element name="GetLogicalReaderNamesResult" type="impl:ArrayOfString"/>
4994
4995     <xsd:element name="GetLRSpec" type="impl:GetLRSpec"/>
4996     <xsd:complexType name="GetLRSpec">
4997       <xsd:sequence>
4998         <xsd:element name="name" type="xsd:string"/>
4999       </xsd:sequence>
5000     </xsd:complexType>
5001     <xsd:element name="GetLRSpecResult" type="ale:LRSpec"/>
5002
5003     <xsd:element name="AddReaders" type="impl:AddReaders"/>
5004     <xsd:complexType name="AddReaders">
5005       <xsd:sequence>
5006         <xsd:element name="name" type="xsd:string"/>
5007         <xsd:element name="readers" minOccurs="0">
5008           <xsd:complexType>
5009             <xsd:sequence>
5010               <xsd:element name="reader" type="xsd:string" minOccurs="0"
5011                 maxOccurs="unbounded"/>

```

```

5012         </xsd:sequence>
5013     </xsd:complexType>
5014 </xsd:element>
5015 </xsd:sequence>
5016 </xsd:complexType>
5017 <xsd:element name="AddReadersResult">
5018     <xsd:complexType/>
5019 </xsd:element>
5020
5021 <xsd:element name="SetReaders" type="impl:SetReaders"/>
5022 <xsd:complexType name="SetReaders">
5023     <xsd:sequence>
5024         <xsd:element name="name" type="xsd:string"/>
5025         <xsd:element name="readers" minOccurs="0">
5026             <xsd:complexType>
5027                 <xsd:sequence>
5028                     <xsd:element name="reader" type="xsd:string" minOccurs="0"
5029                         maxOccurs="unbounded"/>
5030                 </xsd:sequence>
5031             </xsd:complexType>
5032         </xsd:element>
5033     </xsd:sequence>
5034 </xsd:complexType>
5035 <xsd:element name="SetReadersResult">
5036     <xsd:complexType/>
5037 </xsd:element>
5038
5039 <xsd:element name="RemoveReaders" type="impl:RemoveReaders"/>
5040 <xsd:complexType name="RemoveReaders">
5041     <xsd:sequence>
5042         <xsd:element name="name" type="xsd:string"/>
5043         <xsd:element name="readers" minOccurs="0">
5044             <xsd:complexType>
5045                 <xsd:sequence>
5046                     <xsd:element name="reader" type="xsd:string" minOccurs="0"
5047                         maxOccurs="unbounded"/>
5048                 </xsd:sequence>
5049             </xsd:complexType>
5050         </xsd:element>
5051     </xsd:sequence>
5052 </xsd:complexType>
5053 <xsd:element name="RemoveReadersResult">
5054     <xsd:complexType/>
5055 </xsd:element>
5056
5057 <xsd:element name="SetProperties" type="impl:SetProperties"/>
5058 <xsd:complexType name="SetProperties">
5059     <xsd:sequence>
5060         <xsd:element name="name" type="xsd:string"/>
5061         <xsd:element name="properties" minOccurs="0">
5062             <xsd:complexType>
5063                 <xsd:sequence>
5064                     <xsd:element name="property" type="ale:LRProperty" minOccurs="0"
5065                         maxOccurs="unbounded"/>
5066                 </xsd:sequence>
5067             </xsd:complexType>
5068         </xsd:element>
5069     </xsd:sequence>
5070 </xsd:complexType>
5071 <xsd:element name="SetPropertiesResult">
5072     <xsd:complexType/>
5073 </xsd:element>
5074
5075 <xsd:element name="GetPropertyValue" type="impl:GetPropertyValue"/>
5076 <xsd:complexType name="GetPropertyValue">
5077     <xsd:sequence>
5078         <xsd:element name="name" type="xsd:string"/>
5079         <xsd:element name="propertyName" type="xsd:string"/>
5080     </xsd:sequence>
5081 </xsd:complexType>

```

```

5082 <xsd:element name="GetPropertyValueResult" type="xsd:string"/>
5083
5084 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
5085 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
5086
5087 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
5088 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
5089
5090 <xsd:element name="ALEException" type="impl:ALEException"/>
5091 <xsd:complexType name="ALEException">
5092   <xsd:sequence>
5093     <xsd:element name="reason" type="xsd:string"/>
5094   </xsd:sequence>
5095 </xsd:complexType>
5096
5097 <xsd:element name="DuplicateNameException" type="impl:DuplicateNameException"/>
5098 <xsd:complexType name="DuplicateNameException">
5099   <xsd:complexContent>
5100     <xsd:extension base="impl:ALEException"/>
5101   </xsd:complexContent>
5102 </xsd:complexType>
5103
5104 <xsd:element name="NoSuchNameException" type="impl:NoSuchNameException"/>
5105 <xsd:complexType name="NoSuchNameException">
5106   <xsd:complexContent>
5107     <xsd:extension base="impl:ALEException"/>
5108   </xsd:complexContent>
5109 </xsd:complexType>
5110
5111 <xsd:element name="InUseException" type="impl:InUseException"/>
5112 <xsd:complexType name="InUseException">
5113   <xsd:complexContent>
5114     <xsd:extension base="impl:ALEException"/>
5115   </xsd:complexContent>
5116 </xsd:complexType>
5117
5118 <xsd:element name="ValidationException" type="impl:ValidationException"/>
5119 <xsd:complexType name="ValidationException">
5120   <xsd:complexContent>
5121     <xsd:extension base="impl:ALEException"/>
5122   </xsd:complexContent>
5123 </xsd:complexType>
5124
5125 <xsd:element name="ImmutableReaderException" type="impl:ImmutableReaderException"/>
5126 <xsd:complexType name="ImmutableReaderException">
5127   <xsd:complexContent>
5128     <xsd:extension base="impl:ALEException"/>
5129   </xsd:complexContent>
5130 </xsd:complexType>
5131
5132 <xsd:element name="NonCompositeReaderException"
5133           type="impl:NonCompositeReaderException"/>
5134 <xsd:complexType name="NonCompositeReaderException">
5135   <xsd:complexContent>
5136     <xsd:extension base="impl:ALEException"/>
5137   </xsd:complexContent>
5138 </xsd:complexType>
5139
5140 <xsd:element name="ReaderLoopException" type="impl:ReaderLoopException"/>
5141 <xsd:complexType name="ReaderLoopException">
5142   <xsd:complexContent>
5143     <xsd:extension base="impl:ALEException"/>
5144   </xsd:complexContent>
5145 </xsd:complexType>
5146
5147 <xsd:element name="SecurityException" type="impl:SecurityException"/>
5148 <xsd:complexType name="SecurityException">
5149   <xsd:complexContent>
5150     <xsd:extension base="impl:ALEException"/>
5151   </xsd:complexContent>

```

```

5152     </xsd:complexType>
5153
5154     <xsd:element name="ImplementationException" type="impl:ImplementationException"/>
5155     <xsd:complexType name="ImplementationException">
5156         <xsd:complexContent>
5157             <xsd:extension base="impl:ALEException">
5158                 <xsd:sequence>
5159                     <xsd:element name="severity" type="impl:ImplementationExceptionSeverity"/>
5160                 </xsd:sequence>
5161             </xsd:extension>
5162         </xsd:complexContent>
5163     </xsd:complexType>
5164
5165     <xsd:complexType name="ArrayOfString">
5166         <xsd:sequence>
5167             <xsd:element name="string" type="xsd:string" minOccurs="0"
5168                 maxOccurs="unbounded"/>
5169         </xsd:sequence>
5170     </xsd:complexType>
5171
5172     <!-- The ImplementationExceptionSeverity type is an enumerated type.
5173         The following strings are legal values for this type:
5174         ERROR
5175         SEVERE
5176         -->
5177     <xsd:simpleType name="ImplementationExceptionSeverity">
5178         <xsd:restriction base="xsd:string"/>
5179     </xsd:simpleType>
5180
5181     <xsd:complexType name="EmptyParms"/>
5182 </xsd:schema>
5183 </wsdl:types>
5184 <!-- ALELRSERVICE MESSAGES -->
5185
5186 <wsdl:message name="defineRequest">
5187     <wsdl:part name="parms" element="impl:Define"/>
5188 </wsdl:message>
5189 <wsdl:message name="defineResponse">
5190     <wsdl:part name="defineReturn" element="impl:DefineResult"/>
5191 </wsdl:message>
5192
5193 <wsdl:message name="updateRequest">
5194     <wsdl:part name="parms" element="impl:Update"/>
5195 </wsdl:message>
5196 <wsdl:message name="updateResponse">
5197     <wsdl:part name="updateReturn" element="impl:UpdateResult"/>
5198 </wsdl:message>
5199
5200 <wsdl:message name="undefineRequest">
5201     <wsdl:part name="parms" element="impl:Undefine"/>
5202 </wsdl:message>
5203 <wsdl:message name="undefineResponse">
5204     <wsdl:part name="undefineReturn" element="impl:UndefineResult"/>
5205 </wsdl:message>
5206
5207 <wsdl:message name="getLogicalReaderNamesRequest">
5208     <wsdl:part name="parms" element="impl:GetLogicalReaderNames"/>
5209 </wsdl:message>
5210 <wsdl:message name="getLogicalReaderNamesResponse">
5211     <wsdl:part name="getLogicalReaderNamesReturn"
5212         element="impl:GetLogicalReaderNamesResult"/>
5213 </wsdl:message>
5214
5215 <wsdl:message name="getLRSpecRequest">
5216     <wsdl:part name="parms" element="impl:GetLRSpec"/>
5217 </wsdl:message>
5218 <wsdl:message name="getLRSpecResponse">
5219     <wsdl:part name="getLRSpecReturn" element="impl:GetLRSpecResult"/>
5220 </wsdl:message>
5221

```

```

5222 <wsdl:message name="addReadersRequest">
5223   <wsdl:part name="parms" element="impl:AddReaders"/>
5224 </wsdl:message>
5225 <wsdl:message name="addReadersResponse">
5226   <wsdl:part name="addReadersReturn" element="impl:AddReadersResult"/>
5227 </wsdl:message>
5228
5229 <wsdl:message name="setReadersRequest">
5230   <wsdl:part name="parms" element="impl:SetReaders"/>
5231 </wsdl:message>
5232 <wsdl:message name="setReadersResponse">
5233   <wsdl:part name="setReadersReturn" element="impl:SetReadersResult"/>
5234 </wsdl:message>
5235
5236 <wsdl:message name="removeReadersRequest">
5237   <wsdl:part name="parms" element="impl:RemoveReaders"/>
5238 </wsdl:message>
5239 <wsdl:message name="removeReadersResponse">
5240   <wsdl:part name="removeReadersReturn" element="impl:RemoveReadersResult"/>
5241 </wsdl:message>
5242
5243 <wsdl:message name="setPropertiesRequest">
5244   <wsdl:part name="parms" element="impl:SetProperties"/>
5245 </wsdl:message>
5246 <wsdl:message name="setPropertiesResponse">
5247   <wsdl:part name="setPropertiesReturn" element="impl:SetPropertiesResult"/>
5248 </wsdl:message>
5249
5250 <wsdl:message name="getPropertyValueRequest">
5251   <wsdl:part name="parms" element="impl:GetPropertyValue"/>
5252 </wsdl:message>
5253 <wsdl:message name="getPropertyValueResponse">
5254   <wsdl:part name="getPropertyValueReturn" element="impl:GetPropertyValueResult"/>
5255 </wsdl:message>
5256
5257 <wsdl:message name="getStandardVersionRequest">
5258   <wsdl:part name="parms" element="impl:GetStandardVersion"/>
5259 </wsdl:message>
5260 <wsdl:message name="getStandardVersionResponse">
5261   <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult"/>
5262 </wsdl:message>
5263
5264 <wsdl:message name="getVendorVersionRequest">
5265   <wsdl:part name="parms" element="impl:GetVendorVersion"/>
5266 </wsdl:message>
5267 <wsdl:message name="getVendorVersionResponse">
5268   <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult"/>
5269 </wsdl:message>
5270
5271 <wsdl:message name="DuplicateNameExceptionResponse">
5272   <wsdl:part name="fault" element="impl:DuplicateNameException"/>
5273 </wsdl:message>
5274
5275 <wsdl:message name="NoSuchNameExceptionResponse">
5276   <wsdl:part name="fault" element="impl:NoSuchNameException"/>
5277 </wsdl:message>
5278
5279 <wsdl:message name="InUseExceptionResponse">
5280   <wsdl:part name="fault" element="impl:InUseException"/>
5281 </wsdl:message>
5282
5283 <wsdl:message name="ValidationExceptionResponse">
5284   <wsdl:part name="fault" element="impl:ValidationException"/>
5285 </wsdl:message>
5286
5287 <wsdl:message name="ImmutableReaderExceptionResponse">
5288   <wsdl:part name="fault" element="impl:ImmutableReaderException"/>
5289 </wsdl:message>
5290
5291 <wsdl:message name="NonCompositeReaderExceptionResponse">

```

```

5292     <wsdl:part name="fault" element="impl:NonCompositeReaderException"/>
5293 </wsdl:message>
5294
5295 <wsdl:message name="ReaderLoopExceptionResponse">
5296   <wsdl:part name="fault" element="impl:ReaderLoopException"/>
5297 </wsdl:message>
5298
5299 <wsdl:message name="SecurityExceptionResponse">
5300   <wsdl:part name="fault" element="impl:SecurityException"/>
5301 </wsdl:message>
5302
5303 <wsdl:message name="ImplementationExceptionResponse">
5304   <wsdl:part name="fault" element="impl:ImplementationException"/>
5305 </wsdl:message>
5306 <!-- ALELRSERVICE PORTTYPE -->
5307
5308 <wsdl:portType name="ALELRServicePortType">
5309
5310   <wsdl:operation name="define">
5311     <wsdl:input message="impl:defineRequest" name="defineRequest"/>
5312     <wsdl:output message="impl:defineResponse" name="defineResponse"/>
5313     <wsdl:fault message="impl:DuplicateNameExceptionResponse"
5314       name="DuplicateNameExceptionFault"/>
5315     <wsdl:fault message="impl:ValidationExceptionResponse"
5316       name="ValidationExceptionFault"/>
5317     <wsdl:fault message="impl:SecurityExceptionResponse"
5318       name="SecurityExceptionFault"/>
5319     <wsdl:fault message="impl:ImplementationExceptionResponse"
5320       name="ImplementationExceptionFault"/>
5321   </wsdl:operation>
5322
5323   <wsdl:operation name="update">
5324     <wsdl:input message="impl:updateRequest" name="updateRequest"/>
5325     <wsdl:output message="impl:updateResponse" name="updateResponse"/>
5326     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5327       name="NoSuchNameExceptionFault"/>
5328     <wsdl:fault message="impl:ValidationExceptionResponse"
5329       name="ValidationExceptionFault"/>
5330     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault"/>
5331     <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5332       name="ImmutableReaderExceptionFault"/>
5333     <wsdl:fault message="impl:ReaderLoopExceptionResponse"
5334       name="ReaderLoopExceptionFault"/>
5335     <wsdl:fault message="impl:SecurityExceptionResponse"
5336       name="SecurityExceptionFault"/>
5337     <wsdl:fault message="impl:ImplementationExceptionResponse"
5338       name="ImplementationExceptionFault"/>
5339   </wsdl:operation>
5340
5341   <wsdl:operation name="undefine">
5342     <wsdl:input message="impl:undefineRequest" name="undefineRequest"/>
5343     <wsdl:output message="impl:undefineResponse" name="undefineResponse"/>
5344     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5345       name="NoSuchNameExceptionFault"/>
5346     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault"/>
5347     <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5348       name="ImmutableReaderExceptionFault"/>
5349     <wsdl:fault message="impl:SecurityExceptionResponse"
5350       name="SecurityExceptionFault"/>
5351     <wsdl:fault message="impl:ImplementationExceptionResponse"
5352       name="ImplementationExceptionFault"/>
5353   </wsdl:operation>
5354
5355   <wsdl:operation name="getLogicalReaderNames">
5356     <wsdl:input message="impl:getLogicalReaderNamesRequest"
5357       name="getLogicalReaderNamesRequest"/>
5358     <wsdl:output message="impl:getLogicalReaderNamesResponse"
5359       name="getLogicalReaderNamesResponse"/>
5360     <wsdl:fault message="impl:SecurityExceptionResponse"
5361       name="SecurityExceptionFault"/>

```

```

5362     <wsdl:fault message="impl:ImplementationExceptionResponse"
5363             name="ImplementationExceptionFault" />
5364 </wsdl:operation>
5365
5366 <wsdl:operation name="getLRSpec">
5367     <wsdl:input message="impl:getLRSpecRequest" name="getLRSpecRequest" />
5368     <wsdl:output message="impl:getLRSpecResponse" name="getLRSpecResponse" />
5369     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5370             name="NoSuchNameExceptionFault" />
5371     <wsdl:fault message="impl:SecurityExceptionResponse"
5372             name="SecurityExceptionFault" />
5373     <wsdl:fault message="impl:ImplementationExceptionResponse"
5374             name="ImplementationExceptionFault" />
5375 </wsdl:operation>
5376
5377 <wsdl:operation name="addReaders">
5378     <wsdl:input message="impl:addReadersRequest" name="addReadersRequest" />
5379     <wsdl:output message="impl:addReadersResponse" name="addReadersResponse" />
5380     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5381             name="NoSuchNameExceptionFault" />
5382     <wsdl:fault message="impl:ValidationExceptionResponse"
5383             name="ValidationExceptionFault" />
5384     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
5385     <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5386             name="ImmutableReaderExceptionFault" />
5387     <wsdl:fault message="impl:NonCompositeReaderExceptionResponse"
5388             name="NonCompositeReaderExceptionFault" />
5389     <wsdl:fault message="impl:ReaderLoopExceptionResponse"
5390             name="ReaderLoopExceptionFault" />
5391     <wsdl:fault message="impl:SecurityExceptionResponse"
5392             name="SecurityExceptionFault" />
5393     <wsdl:fault message="impl:ImplementationExceptionResponse"
5394             name="ImplementationExceptionFault" />
5395 </wsdl:operation>
5396
5397 <wsdl:operation name="setReaders">
5398     <wsdl:input message="impl:setReadersRequest" name="setReadersRequest" />
5399     <wsdl:output message="impl:setReadersResponse" name="setReadersResponse" />
5400     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5401             name="NoSuchNameExceptionFault" />
5402     <wsdl:fault message="impl:ValidationExceptionResponse"
5403             name="ValidationExceptionFault" />
5404     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
5405     <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5406             name="ImmutableReaderExceptionFault" />
5407     <wsdl:fault message="impl:NonCompositeReaderExceptionResponse"
5408             name="NonCompositeReaderExceptionFault" />
5409     <wsdl:fault message="impl:ReaderLoopExceptionResponse"
5410             name="ReaderLoopExceptionFault" />
5411     <wsdl:fault message="impl:SecurityExceptionResponse"
5412             name="SecurityExceptionFault" />
5413     <wsdl:fault message="impl:ImplementationExceptionResponse"
5414             name="ImplementationExceptionFault" />
5415 </wsdl:operation>
5416
5417 <wsdl:operation name="removeReaders">
5418     <wsdl:input message="impl:removeReadersRequest" name="removeReadersRequest" />
5419     <wsdl:output message="impl:removeReadersResponse" name="removeReadersResponse" />
5420     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5421             name="NoSuchNameExceptionFault" />
5422     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
5423     <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5424             name="ImmutableReaderExceptionFault" />
5425     <wsdl:fault message="impl:NonCompositeReaderExceptionResponse"
5426             name="NonCompositeReaderExceptionFault" />
5427     <wsdl:fault message="impl:SecurityExceptionResponse"
5428             name="SecurityExceptionFault" />
5429     <wsdl:fault message="impl:ImplementationExceptionResponse"
5430             name="ImplementationExceptionFault" />
5431 </wsdl:operation>

```

```

5432
5433 <wsdl:operation name="setProperties">
5434   <wsdl:input message="impl:setPropertiesRequest" name="setPropertiesRequest"/>
5435   <wsdl:output message="impl:setPropertiesResponse" name="setPropertiesResponse"/>
5436   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5437     name="NoSuchNameExceptionFault"/>
5438   <wsdl:fault message="impl:ValidationExceptionResponse"
5439     name="ValidationExceptionFault"/>
5440   <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault"/>
5441   <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5442     name="ImmutableReaderExceptionFault"/>
5443   <wsdl:fault message="impl:SecurityExceptionResponse"
5444     name="SecurityExceptionFault"/>
5445   <wsdl:fault message="impl:ImplementationExceptionResponse"
5446     name="ImplementationExceptionFault"/>
5447 </wsdl:operation>
5448
5449 <wsdl:operation name="getPropertyValue">
5450   <wsdl:input message="impl:getPropertyValueRequest" name="getPropertyValueRequest"/>
5451   <wsdl:output message="impl:getPropertyValueResponse"
5452     name="getPropertyValueResponse"/>
5453   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5454     name="NoSuchNameExceptionFault"/>
5455   <wsdl:fault message="impl:SecurityExceptionResponse"
5456     name="SecurityExceptionFault"/>
5457   <wsdl:fault message="impl:ImplementationExceptionResponse"
5458     name="ImplementationExceptionFault"/>
5459 </wsdl:operation>
5460
5461 <wsdl:operation name="getStandardVersion">
5462   <wsdl:input message="impl:getStandardVersionRequest"
5463     name="getStandardVersionRequest"/>
5464   <wsdl:output message="impl:getStandardVersionResponse"
5465     name="getStandardVersionResponse"/>
5466   <wsdl:fault message="impl:ImplementationExceptionResponse"
5467     name="ImplementationExceptionFault"/>
5468 </wsdl:operation>
5469
5470 <wsdl:operation name="getVendorVersion">
5471   <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest"/>
5472   <wsdl:output message="impl:getVendorVersionResponse"
5473     name="getVendorVersionResponse"/>
5474   <wsdl:fault message="impl:ImplementationExceptionResponse"
5475     name="ImplementationExceptionFault"/>
5476 </wsdl:operation>
5477 </wsdl:portType>
5478 <!-- ALELRSERVICE BINDING -->
5479
5480 <wsdl:binding name="ALELRServiceBinding" type="impl:ALELRServicePortType">
5481   <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
5482
5483   <wsdl:operation name="define">
5484     <wsdlsoap:operation soapAction=""/>
5485     <wsdl:input name="defineRequest">
5486       <wsdlsoap:body use="literal"/>
5487     </wsdl:input>
5488     <wsdl:output name="defineResponse">
5489       <wsdlsoap:body use="literal"/>
5490     </wsdl:output>
5491     <wsdl:fault name="DuplicateNameExceptionFault">
5492       <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
5493     </wsdl:fault>
5494     <wsdl:fault name="ValidationExceptionFault">
5495       <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5496     </wsdl:fault>
5497     <wsdl:fault name="SecurityExceptionFault">
5498       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5499     </wsdl:fault>
5500     <wsdl:fault name="ImplementationExceptionFault">
5501       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>

```

```

5502     </wsdl:fault>
5503 </wsdl:operation>
5504
5505 <wsdl:operation name="update">
5506   <wsdlsoap:operation soapAction=""/>
5507   <wsdl:input name="updateRequest">
5508     <wsdlsoap:body use="literal"/>
5509   </wsdl:input>
5510   <wsdl:output name="updateResponse">
5511     <wsdlsoap:body use="literal"/>
5512   </wsdl:output>
5513   <wsdl:fault name="NoSuchNameExceptionFault">
5514     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5515   </wsdl:fault>
5516   <wsdl:fault name="ValidationExceptionFault">
5517     <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5518   </wsdl:fault>
5519   <wsdl:fault name="InUseExceptionFault">
5520     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5521   </wsdl:fault>
5522   <wsdl:fault name="ImmutableReaderExceptionFault">
5523     <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5524   </wsdl:fault>
5525   <wsdl:fault name="ReaderLoopExceptionFault">
5526     <wsdlsoap:fault name="ReaderLoopExceptionFault" use="literal"/>
5527   </wsdl:fault>
5528   <wsdl:fault name="SecurityExceptionFault">
5529     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5530   </wsdl:fault>
5531   <wsdl:fault name="ImplementationExceptionFault">
5532     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5533   </wsdl:fault>
5534 </wsdl:operation>
5535
5536 <wsdl:operation name="undefine">
5537   <wsdlsoap:operation soapAction=""/>
5538   <wsdl:input name="undefineRequest">
5539     <wsdlsoap:body use="literal"/>
5540   </wsdl:input>
5541   <wsdl:output name="undefineResponse">
5542     <wsdlsoap:body use="literal"/>
5543   </wsdl:output>
5544   <wsdl:fault name="NoSuchNameExceptionFault">
5545     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5546   </wsdl:fault>
5547   <wsdl:fault name="InUseExceptionFault">
5548     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5549   </wsdl:fault>
5550   <wsdl:fault name="ImmutableReaderExceptionFault">
5551     <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5552   </wsdl:fault>
5553   <wsdl:fault name="SecurityExceptionFault">
5554     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5555   </wsdl:fault>
5556   <wsdl:fault name="ImplementationExceptionFault">
5557     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5558   </wsdl:fault>
5559 </wsdl:operation>
5560
5561 <wsdl:operation name="getLogicalReaderNames">
5562   <wsdlsoap:operation soapAction=""/>
5563   <wsdl:input name="getLogicalReaderNamesRequest">
5564     <wsdlsoap:body use="literal"/>
5565   </wsdl:input>
5566   <wsdl:output name="getLogicalReaderNamesResponse">
5567     <wsdlsoap:body use="literal"/>
5568   </wsdl:output>
5569   <wsdl:fault name="SecurityExceptionFault">
5570     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5571   </wsdl:fault>

```

```

5572     <wsdl:fault name="ImplementationExceptionFault">
5573         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5574     </wsdl:fault>
5575 </wsdl:operation>
5576
5577 <wsdl:operation name="getLRSpec">
5578     <wsdlsoap:operation soapAction=""/>
5579     <wsdl:input name="getLRSpecRequest">
5580         <wsdlsoap:body use="literal"/>
5581     </wsdl:input>
5582     <wsdl:output name="getLRSpecResponse">
5583         <wsdlsoap:body use="literal"/>
5584     </wsdl:output>
5585     <wsdl:fault name="NoSuchNameExceptionFault">
5586         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5587     </wsdl:fault>
5588     <wsdl:fault name="SecurityExceptionFault">
5589         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5590     </wsdl:fault>
5591     <wsdl:fault name="ImplementationExceptionFault">
5592         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5593     </wsdl:fault>
5594 </wsdl:operation>
5595
5596 <wsdl:operation name="addReaders">
5597     <wsdlsoap:operation soapAction=""/>
5598     <wsdl:input name="addReadersRequest">
5599         <wsdlsoap:body use="literal"/>
5600     </wsdl:input>
5601     <wsdl:output name="addReadersResponse">
5602         <wsdlsoap:body use="literal"/>
5603     </wsdl:output>
5604     <wsdl:fault name="NoSuchNameExceptionFault">
5605         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5606     </wsdl:fault>
5607     <wsdl:fault name="ValidationExceptionFault">
5608         <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5609     </wsdl:fault>
5610     <wsdl:fault name="InUseExceptionFault">
5611         <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5612     </wsdl:fault>
5613     <wsdl:fault name="ImmutableReaderExceptionFault">
5614         <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5615     </wsdl:fault>
5616     <wsdl:fault name="NonCompositeReaderExceptionFault">
5617         <wsdlsoap:fault name="NonCompositeReaderExceptionFault" use="literal"/>
5618     </wsdl:fault>
5619     <wsdl:fault name="ReaderLoopExceptionFault">
5620         <wsdlsoap:fault name="ReaderLoopExceptionFault" use="literal"/>
5621     </wsdl:fault>
5622     <wsdl:fault name="SecurityExceptionFault">
5623         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5624     </wsdl:fault>
5625     <wsdl:fault name="ImplementationExceptionFault">
5626         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5627     </wsdl:fault>
5628 </wsdl:operation>
5629
5630 <wsdl:operation name="setReaders">
5631     <wsdlsoap:operation soapAction=""/>
5632     <wsdl:input name="setReadersRequest">
5633         <wsdlsoap:body use="literal"/>
5634     </wsdl:input>
5635     <wsdl:output name="setReadersResponse">
5636         <wsdlsoap:body use="literal"/>
5637     </wsdl:output>
5638     <wsdl:fault name="NoSuchNameExceptionFault">
5639         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5640     </wsdl:fault>
5641     <wsdl:fault name="ValidationExceptionFault">

```

```

5642     <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5643 </wsdl:fault>
5644 <wsdl:fault name="InUseExceptionFault">
5645   <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5646 </wsdl:fault>
5647 <wsdl:fault name="ImmutableReaderExceptionFault">
5648   <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5649 </wsdl:fault>
5650 <wsdl:fault name="NonCompositeReaderExceptionFault">
5651   <wsdlsoap:fault name="NonCompositeReaderExceptionFault" use="literal"/>
5652 </wsdl:fault>
5653 <wsdl:fault name="ReaderLoopExceptionFault">
5654   <wsdlsoap:fault name="ReaderLoopExceptionFault" use="literal"/>
5655 </wsdl:fault>
5656 <wsdl:fault name="SecurityExceptionFault">
5657   <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5658 </wsdl:fault>
5659 <wsdl:fault name="ImplementationExceptionFault">
5660   <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5661 </wsdl:fault>
5662 </wsdl:operation>
5663
5664 <wsdl:operation name="removeReaders">
5665   <wsdlsoap:operation soapAction="" />
5666   <wsdl:input name="removeReadersRequest">
5667     <wsdlsoap:body use="literal"/>
5668   </wsdl:input>
5669   <wsdl:output name="removeReadersResponse">
5670     <wsdlsoap:body use="literal"/>
5671   </wsdl:output>
5672   <wsdl:fault name="NoSuchNameExceptionFault">
5673     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5674   </wsdl:fault>
5675   <wsdl:fault name="InUseExceptionFault">
5676     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5677   </wsdl:fault>
5678   <wsdl:fault name="ImmutableReaderExceptionFault">
5679     <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5680   </wsdl:fault>
5681   <wsdl:fault name="NonCompositeReaderExceptionFault">
5682     <wsdlsoap:fault name="NonCompositeReaderExceptionFault" use="literal"/>
5683   </wsdl:fault>
5684   <wsdl:fault name="SecurityExceptionFault">
5685     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5686   </wsdl:fault>
5687   <wsdl:fault name="ImplementationExceptionFault">
5688     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5689   </wsdl:fault>
5690 </wsdl:operation>
5691
5692 <wsdl:operation name="setPropertyies">
5693   <wsdlsoap:operation soapAction="" />
5694   <wsdl:input name="setPropertyiesRequest">
5695     <wsdlsoap:body use="literal"/>
5696   </wsdl:input>
5697   <wsdl:output name="setPropertyiesResponse">
5698     <wsdlsoap:body use="literal"/>
5699   </wsdl:output>
5700   <wsdl:fault name="NoSuchNameExceptionFault">
5701     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5702   </wsdl:fault>
5703   <wsdl:fault name="ValidationExceptionFault">
5704     <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5705   </wsdl:fault>
5706   <wsdl:fault name="InUseExceptionFault">
5707     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5708   </wsdl:fault>
5709   <wsdl:fault name="ImmutableReaderExceptionFault">
5710     <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5711   </wsdl:fault>

```

```

5712     <wsdl:fault name="SecurityExceptionFault">
5713         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5714     </wsdl:fault>
5715     <wsdl:fault name="ImplementationExceptionFault">
5716         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5717     </wsdl:fault>
5718 </wsdl:operation>
5719
5720 <wsdl:operation name="getPropertyValue">
5721     <wsdlsoap:operation soapAction=""/>
5722     <wsdl:input name="getPropertyValueRequest">
5723         <wsdlsoap:body use="literal"/>
5724     </wsdl:input>
5725     <wsdl:output name="getPropertyValueResponse">
5726         <wsdlsoap:body use="literal"/>
5727     </wsdl:output>
5728     <wsdl:fault name="NoSuchNameExceptionFault">
5729         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5730     </wsdl:fault>
5731     <wsdl:fault name="SecurityExceptionFault">
5732         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5733     </wsdl:fault>
5734     <wsdl:fault name="ImplementationExceptionFault">
5735         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5736     </wsdl:fault>
5737 </wsdl:operation>
5738
5739 <wsdl:operation name="getStandardVersion">
5740     <wsdlsoap:operation soapAction=""/>
5741     <wsdl:input name="getStandardVersionRequest">
5742         <wsdlsoap:body use="literal"/>
5743     </wsdl:input>
5744     <wsdl:output name="getStandardVersionResponse">
5745         <wsdlsoap:body use="literal"/>
5746     </wsdl:output>
5747     <wsdl:fault name="ImplementationExceptionFault">
5748         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5749     </wsdl:fault>
5750 </wsdl:operation>
5751
5752 <wsdl:operation name="getVendorVersion">
5753     <wsdlsoap:operation soapAction=""/>
5754     <wsdl:input name="getVendorVersionRequest">
5755         <wsdlsoap:body use="literal"/>
5756     </wsdl:input>
5757     <wsdl:output name="getVendorVersionResponse">
5758         <wsdlsoap:body use="literal"/>
5759     </wsdl:output>
5760     <wsdl:fault name="ImplementationExceptionFault">
5761         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5762     </wsdl:fault>
5763 </wsdl:operation>
5764 </wsdl:binding>
5765
5766 <!-- ALELRSERVICE -->
5767 <wsdl:service name="ALELRService">
5768     <wsdl:port binding="impl:ALELRServiceBinding" name="ALELRServicePort">
5769         <!-- The value of the location attribute below is an example only;
5770             Implementations are free to choose any appropriate URL. -->
5771         <wsdlsoap:address location="http://localhost:8080/services/ALELRService"/>
5772     </wsdl:port>
5773 </wsdl:service>
5774 </wsdl:definitions>

```

## 5775 4.7 ALE Access Control API SOAP Binding

5776 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
5777 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Access

5778 Control API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0  
5779 [WSI].

```
5780 <?xml version="1.0" encoding="UTF-8"?>
5781 <!-- ALEACSERVICE DEFINITIONS -->
5782 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
5783     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
5784     xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
5785     xmlns:ale="urn:epcglobal:ale:xsd:1"
5786     xmlns:impl="urn:epcglobal:aleac:wsdl:1"
5787     targetNamespace="urn:epcglobal:aleac:wsdl:1">
5788 <!-- ALEACSERVICE TYPES -->
5789 <wsdl:types>
5790 <xsd:schema targetNamespace="urn:epcglobal:aleac:wsdl:1">
5791 <xsd:import namespace="urn:epcglobal:ale:xsd:1"
5792     schemaLocation="EPCglobal-ale-1_1-aleac.xsd"/>
5793 <!-- ALEACSERVICE MESSAGE WRAPPERS -->
5794
5795 <xsd:element name="GetPermissionNames" type="impl:EmptyParms"/>
5796 <xsd:element name="GetPermissionNamesResult" type="impl:ArrayOfString"/>
5797
5798 <xsd:element name="DefinePermission" type="impl:DefinePermission"/>
5799 <xsd:complexType name="DefinePermission">
5800 <xsd:sequence>
5801 <xsd:element name="permName" type="xsd:string"/>
5802 <xsd:element name="perm" type="ale:ACPermission"/>
5803 </xsd:sequence>
5804 </xsd:complexType>
5805 <xsd:element name="DefinePermissionResult">
5806 <xsd:complexType/>
5807 </xsd:element>
5808
5809 <xsd:element name="UpdatePermission" type="impl:UpdatePermission"/>
5810 <xsd:complexType name="UpdatePermission">
5811 <xsd:sequence>
5812 <xsd:element name="permName" type="xsd:string"/>
5813 <xsd:element name="perm" type="ale:ACPermission"/>
5814 </xsd:sequence>
5815 </xsd:complexType>
5816 <xsd:element name="UpdatePermissionResult">
5817 <xsd:complexType/>
5818 </xsd:element>
5819
5820 <xsd:element name="GetPermission" type="impl:GetPermission"/>
5821 <xsd:complexType name="GetPermission">
5822 <xsd:sequence>
5823 <xsd:element name="permName" type="xsd:string"/>
5824 </xsd:sequence>
5825 </xsd:complexType>
5826 <xsd:element name="GetPermissionResult" type="ale:ACPermission"/>
5827
5828 <xsd:element name="UndefinePermission" type="impl:UndefinePermission"/>
5829 <xsd:complexType name="UndefinePermission">
5830 <xsd:sequence>
5831 <xsd:element name="permName" type="xsd:string"/>
5832 </xsd:sequence>
5833 </xsd:complexType>
5834 <xsd:element name="UndefinePermissionResult">
5835 <xsd:complexType/>
5836 </xsd:element>
5837
5838 <xsd:element name="GetRoleNames" type="impl:EmptyParms"/>
5839 <xsd:element name="GetRoleNamesResult" type="impl:ArrayOfString"/>
5840
5841 <xsd:element name="DefineRole" type="impl:DefineRole"/>
5842 <xsd:complexType name="DefineRole">
5843 <xsd:sequence>
5844 <xsd:element name="roleName" type="xsd:string"/>
5845 <xsd:element name="role" type="ale:ACRole"/>
5846 </xsd:sequence>
```

```

5847     </xsd:complexType>
5848     <xsd:element name="DefineRoleResult">
5849       <xsd:complexType/>
5850     </xsd:element>
5851
5852     <xsd:element name="UpdateRole" type="impl:UpdateRole"/>
5853     <xsd:complexType name="UpdateRole">
5854       <xsd:sequence>
5855         <xsd:element name="roleName" type="xsd:string"/>
5856         <xsd:element name="role" type="ale:ACRole"/>
5857       </xsd:sequence>
5858     </xsd:complexType>
5859     <xsd:element name="UpdateRoleResult">
5860       <xsd:complexType/>
5861     </xsd:element>
5862
5863     <xsd:element name="GetRole" type="impl:GetRole"/>
5864     <xsd:complexType name="GetRole">
5865       <xsd:sequence>
5866         <xsd:element name="roleName" type="xsd:string"/>
5867       </xsd:sequence>
5868     </xsd:complexType>
5869     <xsd:element name="GetRoleResult" type="ale:ACRole"/>
5870
5871     <xsd:element name="UndefineRole" type="impl:UndefineRole"/>
5872     <xsd:complexType name="UndefineRole">
5873       <xsd:sequence>
5874         <xsd:element name="roleName" type="xsd:string"/>
5875       </xsd:sequence>
5876     </xsd:complexType>
5877     <xsd:element name="UndefineRoleResult">
5878       <xsd:complexType/>
5879     </xsd:element>
5880
5881     <xsd:element name="AddPermissions" type="impl:AddPermissions"/>
5882     <xsd:complexType name="AddPermissions">
5883       <xsd:sequence>
5884         <xsd:element name="roleName" type="xsd:string"/>
5885         <xsd:element name="permissionNames" minOccurs="0">
5886           <xsd:complexType>
5887             <xsd:sequence>
5888               <xsd:element name="permissionName" type="xsd:string" minOccurs="0"
5889                 maxOccurs="unbounded"/>
5890             </xsd:sequence>
5891           </xsd:complexType>
5892         </xsd:element>
5893       </xsd:sequence>
5894     </xsd:complexType>
5895     <xsd:element name="AddPermissionsResult">
5896       <xsd:complexType/>
5897     </xsd:element>
5898
5899     <xsd:element name="SetPermissions" type="impl:SetPermissions"/>
5900     <xsd:complexType name="SetPermissions">
5901       <xsd:sequence>
5902         <xsd:element name="roleName" type="xsd:string"/>
5903         <xsd:element name="permissionNames" minOccurs="0">
5904           <xsd:complexType>
5905             <xsd:sequence>
5906               <xsd:element name="permissionName" type="xsd:string" minOccurs="0"
5907                 maxOccurs="unbounded"/>
5908             </xsd:sequence>
5909           </xsd:complexType>
5910         </xsd:element>
5911       </xsd:sequence>
5912     </xsd:complexType>
5913     <xsd:element name="SetPermissionsResult">
5914       <xsd:complexType/>
5915     </xsd:element>
5916

```

```

5917 <xsd:element name="RemovePermissions" type="impl:RemovePermissions"/>
5918 <xsd:complexType name="RemovePermissions">
5919   <xsd:sequence>
5920     <xsd:element name="roleName" type="xsd:string"/>
5921     <xsd:element name="permissionNames" minOccurs="0">
5922       <xsd:complexType>
5923         <xsd:sequence>
5924           <xsd:element name="permissionName" type="xsd:string" minOccurs="0"
5925             maxOccurs="unbounded"/>
5926         </xsd:sequence>
5927       </xsd:complexType>
5928     </xsd:element>
5929   </xsd:sequence>
5930 </xsd:complexType>
5931 <xsd:element name="RemovePermissionsResult">
5932   <xsd:complexType/>
5933 </xsd:element>
5934
5935 <xsd:element name="GetClientIdentityNames" type="impl:EmptyParms"/>
5936 <xsd:element name="GetClientIdentityNamesResult" type="impl:ArrayOfString"/>
5937
5938 <xsd:element name="DefineClientIdentity" type="impl:DefineClientIdentity"/>
5939 <xsd:complexType name="DefineClientIdentity">
5940   <xsd:sequence>
5941     <xsd:element name="identityName" type="xsd:string"/>
5942     <xsd:element name="id" type="ale:ACClientIdentity"/>
5943   </xsd:sequence>
5944 </xsd:complexType>
5945 <xsd:element name="DefineClientIdentityResult">
5946   <xsd:complexType/>
5947 </xsd:element>
5948
5949 <xsd:element name="UpdateClientIdentity" type="impl:UpdateClientIdentity"/>
5950 <xsd:complexType name="UpdateClientIdentity">
5951   <xsd:sequence>
5952     <xsd:element name="identityName" type="xsd:string"/>
5953     <xsd:element name="id" type="ale:ACClientIdentity"/>
5954   </xsd:sequence>
5955 </xsd:complexType>
5956 <xsd:element name="UpdateClientIdentityResult">
5957   <xsd:complexType/>
5958 </xsd:element>
5959
5960 <xsd:element name="GetClientIdentity" type="impl:GetClientIdentity"/>
5961 <xsd:complexType name="GetClientIdentity">
5962   <xsd:sequence>
5963     <xsd:element name="identityName" type="xsd:string"/>
5964   </xsd:sequence>
5965 </xsd:complexType>
5966 <xsd:element name="GetClientIdentityResult" type="ale:ACClientIdentity"/>
5967
5968 <xsd:element name="GetClientPermissionNames" type="impl:GetClientPermissionNames"/>
5969 <xsd:complexType name="GetClientPermissionNames">
5970   <xsd:sequence>
5971     <xsd:element name="identityName" type="xsd:string"/>
5972   </xsd:sequence>
5973 </xsd:complexType>
5974 <xsd:element name="GetClientPermissionNamesResult" type="impl:ArrayOfString"/>
5975
5976 <xsd:element name="UndefineClientIdentity" type="impl:UndefineClientIdentity"/>
5977 <xsd:complexType name="UndefineClientIdentity">
5978   <xsd:sequence>
5979     <xsd:element name="identityName" type="xsd:string"/>
5980   </xsd:sequence>
5981 </xsd:complexType>
5982 <xsd:element name="UndefineClientIdentityResult">
5983   <xsd:complexType/>
5984 </xsd:element>
5985
5986 <xsd:element name="AddRoles" type="impl:AddRoles"/>

```

```

5987 <xsd:complexType name="AddRoles">
5988 <xsd:sequence>
5989 <xsd:element name="identityName" type="xsd:string"/>
5990 <xsd:element name="roleNames" minOccurs="0">
5991 <xsd:complexType>
5992 <xsd:sequence>
5993 <xsd:element name="roleName" type="xsd:string" minOccurs="0"
5994 maxOccurs="unbounded"/>
5995 </xsd:sequence>
5996 </xsd:complexType>
5997 </xsd:element>
5998 </xsd:sequence>
5999 </xsd:complexType>
6000 <xsd:element name="AddRolesResult">
6001 <xsd:complexType/>
6002 </xsd:element>
6003
6004 <xsd:element name="RemoveRoles" type="impl:RemoveRoles"/>
6005 <xsd:complexType name="RemoveRoles">
6006 <xsd:sequence>
6007 <xsd:element name="identityName" type="xsd:string"/>
6008 <xsd:element name="roleNames" minOccurs="0">
6009 <xsd:complexType>
6010 <xsd:sequence>
6011 <xsd:element name="roleName" type="xsd:string" minOccurs="0"
6012 maxOccurs="unbounded"/>
6013 </xsd:sequence>
6014 </xsd:complexType>
6015 </xsd:element>
6016 </xsd:sequence>
6017 </xsd:complexType>
6018 <xsd:element name="RemoveRolesResult">
6019 <xsd:complexType/>
6020 </xsd:element>
6021
6022 <xsd:element name="SetRoles" type="impl:SetRoles"/>
6023 <xsd:complexType name="SetRoles">
6024 <xsd:sequence>
6025 <xsd:element name="identityName" type="xsd:string"/>
6026 <xsd:element name="roleNames" minOccurs="0">
6027 <xsd:complexType>
6028 <xsd:sequence>
6029 <xsd:element name="roleName" type="xsd:string" minOccurs="0"
6030 maxOccurs="unbounded"/>
6031 </xsd:sequence>
6032 </xsd:complexType>
6033 </xsd:element>
6034 </xsd:sequence>
6035 </xsd:complexType>
6036 <xsd:element name="SetRolesResult">
6037 <xsd:complexType/>
6038 </xsd:element>
6039
6040 <xsd:element name="GetSupportedOperations" type="impl:EmptyParms"/>
6041 <xsd:element name="GetSupportedOperationsResult" type="impl:ArrayOfString"/>
6042
6043 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
6044 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
6045
6046 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
6047 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
6048
6049 <xsd:element name="ALEException" type="impl:ALEException"/>
6050 <xsd:complexType name="ALEException">
6051 <xsd:sequence>
6052 <xsd:element name="reason" type="xsd:string"/>
6053 </xsd:sequence>
6054 </xsd:complexType>
6055
6056 <xsd:element name="SecurityException" type="impl:SecurityException"/>

```

```

6057 <xsd:complexType name="SecurityException">
6058 <xsd:complexContent>
6059 <xsd:extension base="impl:ALEException"/>
6060 </xsd:complexContent>
6061 </xsd:complexType>
6062
6063 <xsd:element name="NoSuchPermissionException"
6064 type="impl:NoSuchPermissionException"/>
6065 <xsd:complexType name="NoSuchPermissionException">
6066 <xsd:complexContent>
6067 <xsd:extension base="impl:ALEException"/>
6068 </xsd:complexContent>
6069 </xsd:complexType>
6070
6071 <xsd:element name="PermissionValidationException"
6072 type="impl:PermissionValidationException"/>
6073 <xsd:complexType name="PermissionValidationException">
6074 <xsd:complexContent>
6075 <xsd:extension base="impl:ALEException"/>
6076 </xsd:complexContent>
6077 </xsd:complexType>
6078
6079 <xsd:element name="DuplicatePermissionException"
6080 type="impl:DuplicatePermissionException"/>
6081 <xsd:complexType name="DuplicatePermissionException">
6082 <xsd:complexContent>
6083 <xsd:extension base="impl:ALEException"/>
6084 </xsd:complexContent>
6085 </xsd:complexType>
6086
6087 <xsd:element name="NoSuchRoleException" type="impl:NoSuchRoleException"/>
6088 <xsd:complexType name="NoSuchRoleException">
6089 <xsd:complexContent>
6090 <xsd:extension base="impl:ALEException"/>
6091 </xsd:complexContent>
6092 </xsd:complexType>
6093
6094 <xsd:element name="RoleValidationException" type="impl:RoleValidationException"/>
6095 <xsd:complexType name="RoleValidationException">
6096 <xsd:complexContent>
6097 <xsd:extension base="impl:ALEException"/>
6098 </xsd:complexContent>
6099 </xsd:complexType>
6100
6101 <xsd:element name="DuplicateRoleException" type="impl:DuplicateRoleException"/>
6102 <xsd:complexType name="DuplicateRoleException">
6103 <xsd:complexContent>
6104 <xsd:extension base="impl:ALEException"/>
6105 </xsd:complexContent>
6106 </xsd:complexType>
6107
6108 <xsd:element name="NoSuchClientIdentityException"
6109 type="impl:NoSuchClientIdentityException"/>
6110 <xsd:complexType name="NoSuchClientIdentityException">
6111 <xsd:complexContent>
6112 <xsd:extension base="impl:ALEException"/>
6113 </xsd:complexContent>
6114 </xsd:complexType>
6115
6116 <xsd:element name="ClientIdentityValidationException"
6117 type="impl:ClientIdentityValidationException"/>
6118 <xsd:complexType name="ClientIdentityValidationException">
6119 <xsd:complexContent>
6120 <xsd:extension base="impl:ALEException"/>
6121 </xsd:complexContent>
6122 </xsd:complexType>
6123
6124 <xsd:element name="DuplicateClientIdentityException"
6125 type="impl:DuplicateClientIdentityException"/>
6126 <xsd:complexType name="DuplicateClientIdentityException">

```

```

6127     <xsd:complexContent>
6128         <xsd:extension base="impl:ALEException" />
6129     </xsd:complexContent>
6130 </xsd:complexType>
6131
6132 <xsd:element name="UnsupportedOperationException"
6133             type="impl:UnsupportedOperationException" />
6134 <xsd:complexType name="UnsupportedOperationException">
6135     <xsd:complexContent>
6136         <xsd:extension base="impl:ALEException" />
6137     </xsd:complexContent>
6138 </xsd:complexType>
6139
6140 <xsd:element name="ImplementationException" type="impl:ImplementationException" />
6141 <xsd:complexType name="ImplementationException">
6142     <xsd:complexContent>
6143         <xsd:extension base="impl:ALEException">
6144             <xsd:sequence>
6145                 <xsd:element name="severity" type="impl:ImplementationExceptionSeverity" />
6146             </xsd:sequence>
6147         </xsd:extension>
6148     </xsd:complexContent>
6149 </xsd:complexType>
6150
6151 <xsd:complexType name="ArrayOfString">
6152     <xsd:sequence>
6153         <xsd:element name="string" type="xsd:string" minOccurs="0"
6154                 maxOccurs="unbounded" />
6155     </xsd:sequence>
6156 </xsd:complexType>
6157
6158 <!-- The ImplementationExceptionSeverity type is an enumerated type.
6159      The following strings are legal values for this type:
6160          ERROR
6161          SEVERE
6162      -->
6163 <xsd:simpleType name="ImplementationExceptionSeverity">
6164     <xsd:restriction base="xsd:string" />
6165 </xsd:simpleType>
6166
6167     <xsd:complexType name="EmptyParms" />
6168 </xsd:schema>
6169 </wsdl:types>
6170 <!-- ALEACSERVICE MESSAGES -->
6171
6172 <wsdl:message name="getPermissionNamesRequest">
6173     <wsdl:part name="parms" element="impl:GetPermissionNames" />
6174 </wsdl:message>
6175 <wsdl:message name="getPermissionNamesResponse">
6176     <wsdl:part name="getPermissionNamesReturn" element="impl:GetPermissionNamesResult" />
6177 </wsdl:message>
6178
6179 <wsdl:message name="definePermissionRequest">
6180     <wsdl:part name="parms" element="impl:DefinePermission" />
6181 </wsdl:message>
6182 <wsdl:message name="definePermissionResponse">
6183     <wsdl:part name="definePermissionReturn" element="impl:DefinePermissionResult" />
6184 </wsdl:message>
6185
6186 <wsdl:message name="updatePermissionRequest">
6187     <wsdl:part name="parms" element="impl:UpdatePermission" />
6188 </wsdl:message>
6189 <wsdl:message name="updatePermissionResponse">
6190     <wsdl:part name="updatePermissionReturn" element="impl:UpdatePermissionResult" />
6191 </wsdl:message>
6192
6193 <wsdl:message name="getPermissionRequest">
6194     <wsdl:part name="parms" element="impl:GetPermission" />
6195 </wsdl:message>
6196 <wsdl:message name="getPermissionResponse">

```

```

6197     <wsdl:part name="getPermissionReturn" element="impl:GetPermissionResult"/>
6198 </wsdl:message>
6199
6200 <wsdl:message name="undefinePermissionRequest">
6201   <wsdl:part name="parms" element="impl:UndefinePermission"/>
6202 </wsdl:message>
6203 <wsdl:message name="undefinePermissionResponse">
6204   <wsdl:part name="undefinePermissionReturn" element="impl:UndefinePermissionResult"/>
6205 </wsdl:message>
6206
6207 <wsdl:message name="getRoleNamesRequest">
6208   <wsdl:part name="parms" element="impl:GetRoleNames"/>
6209 </wsdl:message>
6210 <wsdl:message name="getRoleNamesResponse">
6211   <wsdl:part name="getRoleNamesReturn" element="impl:GetRoleNamesResult"/>
6212 </wsdl:message>
6213
6214 <wsdl:message name="defineRoleRequest">
6215   <wsdl:part name="parms" element="impl:DefineRole"/>
6216 </wsdl:message>
6217 <wsdl:message name="defineRoleResponse">
6218   <wsdl:part name="defineRoleReturn" element="impl:DefineRoleResult"/>
6219 </wsdl:message>
6220
6221 <wsdl:message name="updateRoleRequest">
6222   <wsdl:part name="parms" element="impl:UpdateRole"/>
6223 </wsdl:message>
6224 <wsdl:message name="updateRoleResponse">
6225   <wsdl:part name="updateRoleReturn" element="impl:UpdateRoleResult"/>
6226 </wsdl:message>
6227
6228 <wsdl:message name="getRoleRequest">
6229   <wsdl:part name="parms" element="impl:GetRole"/>
6230 </wsdl:message>
6231 <wsdl:message name="getRoleResponse">
6232   <wsdl:part name="getRoleReturn" element="impl:GetRoleResult"/>
6233 </wsdl:message>
6234
6235 <wsdl:message name="undefineRoleRequest">
6236   <wsdl:part name="parms" element="impl:UndefineRole"/>
6237 </wsdl:message>
6238 <wsdl:message name="undefineRoleResponse">
6239   <wsdl:part name="undefineRoleReturn" element="impl:UndefineRoleResult"/>
6240 </wsdl:message>
6241
6242 <wsdl:message name="addPermissionsRequest">
6243   <wsdl:part name="parms" element="impl:AddPermissions"/>
6244 </wsdl:message>
6245 <wsdl:message name="addPermissionsResponse">
6246   <wsdl:part name="addPermissionsReturn" element="impl:AddPermissionsResult"/>
6247 </wsdl:message>
6248
6249 <wsdl:message name="setPermissionsRequest">
6250   <wsdl:part name="parms" element="impl:SetPermissions"/>
6251 </wsdl:message>
6252 <wsdl:message name="setPermissionsResponse">
6253   <wsdl:part name="setPermissionsReturn" element="impl:SetPermissionsResult"/>
6254 </wsdl:message>
6255
6256 <wsdl:message name="removePermissionsRequest">
6257   <wsdl:part name="parms" element="impl:RemovePermissions"/>
6258 </wsdl:message>
6259 <wsdl:message name="removePermissionsResponse">
6260   <wsdl:part name="removePermissionsReturn" element="impl:RemovePermissionsResult"/>
6261 </wsdl:message>
6262
6263 <wsdl:message name="getClientIdentityNamesRequest">
6264   <wsdl:part name="parms" element="impl:GetClientIdentityNames"/>
6265 </wsdl:message>
6266 <wsdl:message name="getClientIdentityNamesResponse">

```

```

6267     <wsdl:part name="getClientIdentityNamesReturn"
6268         element="impl:GetClientIdentityNamesResult"/>
6269 </wsdl:message>
6270
6271 <wsdl:message name="defineClientIdentityRequest">
6272     <wsdl:part name="parms" element="impl:DefineClientIdentity"/>
6273 </wsdl:message>
6274 <wsdl:message name="defineClientIdentityResponse">
6275     <wsdl:part name="defineClientIdentityReturn"
6276         element="impl:DefineClientIdentityResult"/>
6277 </wsdl:message>
6278
6279 <wsdl:message name="updateClientIdentityRequest">
6280     <wsdl:part name="parms" element="impl:UpdateClientIdentity"/>
6281 </wsdl:message>
6282 <wsdl:message name="updateClientIdentityResponse">
6283     <wsdl:part name="updateClientIdentityReturn"
6284         element="impl:UpdateClientIdentityResult"/>
6285 </wsdl:message>
6286
6287 <wsdl:message name="getClientIdentityRequest">
6288     <wsdl:part name="parms" element="impl:GetClientIdentity"/>
6289 </wsdl:message>
6290 <wsdl:message name="getClientIdentityResponse">
6291     <wsdl:part name="getClientIdentityReturn" element="impl:GetClientIdentityResult"/>
6292 </wsdl:message>
6293
6294 <wsdl:message name="getClientPermissionNamesRequest">
6295     <wsdl:part name="parms" element="impl:GetClientPermissionNames"/>
6296 </wsdl:message>
6297 <wsdl:message name="getClientPermissionNamesResponse">
6298     <wsdl:part name="getClientPermissionNamesReturn"
6299         element="impl:GetClientPermissionNamesResult"/>
6300 </wsdl:message>
6301
6302 <wsdl:message name="undefineClientIdentityRequest">
6303     <wsdl:part name="parms" element="impl:UndefineClientIdentity"/>
6304 </wsdl:message>
6305 <wsdl:message name="undefineClientIdentityResponse">
6306     <wsdl:part name="undefineClientIdentityReturn"
6307         element="impl:UndefineClientIdentityResult"/>
6308 </wsdl:message>
6309
6310 <wsdl:message name="addRolesRequest">
6311     <wsdl:part name="parms" element="impl:AddRoles"/>
6312 </wsdl:message>
6313 <wsdl:message name="addRolesResponse">
6314     <wsdl:part name="addRolesReturn" element="impl:AddRolesResult"/>
6315 </wsdl:message>
6316
6317 <wsdl:message name="removeRolesRequest">
6318     <wsdl:part name="parms" element="impl:RemoveRoles"/>
6319 </wsdl:message>
6320 <wsdl:message name="removeRolesResponse">
6321     <wsdl:part name="removeRolesReturn" element="impl:RemoveRolesResult"/>
6322 </wsdl:message>
6323
6324 <wsdl:message name="setRolesRequest">
6325     <wsdl:part name="parms" element="impl:SetRoles"/>
6326 </wsdl:message>
6327 <wsdl:message name="setRolesResponse">
6328     <wsdl:part name="setRolesReturn" element="impl:SetRolesResult"/>
6329 </wsdl:message>
6330
6331 <wsdl:message name="getSupportedOperationsRequest">
6332     <wsdl:part name="parms" element="impl:GetSupportedOperations"/>
6333 </wsdl:message>
6334 <wsdl:message name="getSupportedOperationsResponse">
6335     <wsdl:part name="getSupportedOperationsReturn"
6336         element="impl:GetSupportedOperationsResult"/>

```

```

6337 </wsdl:message>
6338
6339 <wsdl:message name="getStandardVersionRequest">
6340   <wsdl:part name="parms" element="impl:GetStandardVersion"/>
6341 </wsdl:message>
6342 <wsdl:message name="getStandardVersionResponse">
6343   <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult"/>
6344 </wsdl:message>
6345
6346 <wsdl:message name="getVendorVersionRequest">
6347   <wsdl:part name="parms" element="impl:GetVendorVersion"/>
6348 </wsdl:message>
6349 <wsdl:message name="getVendorVersionResponse">
6350   <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult"/>
6351 </wsdl:message>
6352
6353 <wsdl:message name="SecurityExceptionResponse">
6354   <wsdl:part name="fault" element="impl:SecurityException"/>
6355 </wsdl:message>
6356
6357 <wsdl:message name="NoSuchPermissionExceptionResponse">
6358   <wsdl:part name="fault" element="impl:NoSuchPermissionException"/>
6359 </wsdl:message>
6360
6361 <wsdl:message name="PermissionValidationExceptionResponse">
6362   <wsdl:part name="fault" element="impl:PermissionValidationException"/>
6363 </wsdl:message>
6364
6365 <wsdl:message name="DuplicatePermissionExceptionResponse">
6366   <wsdl:part name="fault" element="impl:DuplicatePermissionException"/>
6367 </wsdl:message>
6368
6369 <wsdl:message name="NoSuchRoleExceptionResponse">
6370   <wsdl:part name="fault" element="impl:NoSuchRoleException"/>
6371 </wsdl:message>
6372
6373 <wsdl:message name="RoleValidationExceptionResponse">
6374   <wsdl:part name="fault" element="impl:RoleValidationException"/>
6375 </wsdl:message>
6376
6377 <wsdl:message name="DuplicateRoleExceptionResponse">
6378   <wsdl:part name="fault" element="impl:DuplicateRoleException"/>
6379 </wsdl:message>
6380
6381 <wsdl:message name="NoSuchClientIdentityExceptionResponse">
6382   <wsdl:part name="fault" element="impl:NoSuchClientIdentityException"/>
6383 </wsdl:message>
6384
6385 <wsdl:message name="ClientIdentityValidationExceptionResponse">
6386   <wsdl:part name="fault" element="impl:ClientIdentityValidationException"/>
6387 </wsdl:message>
6388
6389 <wsdl:message name="DuplicateClientIdentityExceptionResponse">
6390   <wsdl:part name="fault" element="impl:DuplicateClientIdentityException"/>
6391 </wsdl:message>
6392
6393 <wsdl:message name="UnsupportedOperationExceptionResponse">
6394   <wsdl:part name="fault" element="impl:UnsupportedOperationException"/>
6395 </wsdl:message>
6396
6397 <wsdl:message name="ImplementationExceptionResponse">
6398   <wsdl:part name="fault" element="impl:ImplementationException"/>
6399 </wsdl:message>
6400 <!-- ALEACSERVICE PORTTYPE -->
6401
6402 <wsdl:portType name="ALEACServicePortType">
6403
6404   <wsdl:operation name="getPermissionNames">
6405     <wsdl:input message="impl:getPermissionNamesRequest"
6406       name="getPermissionNamesRequest"/>

```

```

6407     <wsdl:output message="impl:getPermissionNamesResponse"
6408         name="getPermissionNamesResponse" />
6409     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6410         name="UnsupportedOperationExceptionFault" />
6411     <wsdl:fault message="impl:SecurityExceptionResponse"
6412         name="SecurityExceptionFault" />
6413     <wsdl:fault message="impl:ImplementationExceptionResponse"
6414         name="ImplementationExceptionFault" />
6415 </wsdl:operation>
6416
6417 <wsdl:operation name="definePermission">
6418     <wsdl:input message="impl:definePermissionRequest" name="definePermissionRequest" />
6419     <wsdl:output message="impl:definePermissionResponse"
6420         name="definePermissionResponse" />
6421     <wsdl:fault message="impl:SecurityExceptionResponse"
6422         name="SecurityExceptionFault" />
6423     <wsdl:fault message="impl:DuplicatePermissionExceptionResponse"
6424         name="DuplicatePermissionExceptionFault" />
6425     <wsdl:fault message="impl:PermissionValidationExceptionResponse"
6426         name="PermissionValidationExceptionFault" />
6427     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6428         name="UnsupportedOperationExceptionFault" />
6429     <wsdl:fault message="impl:ImplementationExceptionResponse"
6430         name="ImplementationExceptionFault" />
6431 </wsdl:operation>
6432
6433 <wsdl:operation name="updatePermission">
6434     <wsdl:input message="impl:updatePermissionRequest" name="updatePermissionRequest" />
6435     <wsdl:output message="impl:updatePermissionResponse"
6436         name="updatePermissionResponse" />
6437     <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6438         name="NoSuchPermissionExceptionFault" />
6439     <wsdl:fault message="impl:PermissionValidationExceptionResponse"
6440         name="PermissionValidationExceptionFault" />
6441     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6442         name="UnsupportedOperationExceptionFault" />
6443     <wsdl:fault message="impl:SecurityExceptionResponse"
6444         name="SecurityExceptionFault" />
6445     <wsdl:fault message="impl:ImplementationExceptionResponse"
6446         name="ImplementationExceptionFault" />
6447 </wsdl:operation>
6448
6449 <wsdl:operation name="getPermission">
6450     <wsdl:input message="impl:getPermissionRequest" name="getPermissionRequest" />
6451     <wsdl:output message="impl:getPermissionResponse" name="getPermissionResponse" />
6452     <wsdl:fault message="impl:SecurityExceptionResponse"
6453         name="SecurityExceptionFault" />
6454     <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6455         name="NoSuchPermissionExceptionFault" />
6456     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6457         name="UnsupportedOperationExceptionFault" />
6458     <wsdl:fault message="impl:ImplementationExceptionResponse"
6459         name="ImplementationExceptionFault" />
6460 </wsdl:operation>
6461
6462 <wsdl:operation name="undefinePermission">
6463     <wsdl:input message="impl:undefinePermissionRequest"
6464         name="undefinePermissionRequest" />
6465     <wsdl:output message="impl:undefinePermissionResponse"
6466         name="undefinePermissionResponse" />
6467     <wsdl:fault message="impl:SecurityExceptionResponse"
6468         name="SecurityExceptionFault" />
6469     <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6470         name="NoSuchPermissionExceptionFault" />
6471     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6472         name="UnsupportedOperationExceptionFault" />
6473     <wsdl:fault message="impl:ImplementationExceptionResponse"
6474         name="ImplementationExceptionFault" />
6475 </wsdl:operation>
6476

```

```

6477 <wsdl:operation name="getRoleNames">
6478 <wsdl:input message="impl:getRoleNamesRequest" name="getRoleNamesRequest" />
6479 <wsdl:output message="impl:getRoleNamesResponse" name="getRoleNamesResponse" />
6480 <wsdl:fault message="impl:SecurityExceptionResponse"
6481 name="SecurityExceptionFault" />
6482 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6483 name="UnsupportedOperationExceptionFault" />
6484 <wsdl:fault message="impl:ImplementationExceptionResponse"
6485 name="ImplementationExceptionFault" />
6486 </wsdl:operation>
6487
6488 <wsdl:operation name="defineRole">
6489 <wsdl:input message="impl:defineRoleRequest" name="defineRoleRequest" />
6490 <wsdl:output message="impl:defineRoleResponse" name="defineRoleResponse" />
6491 <wsdl:fault message="impl:SecurityExceptionResponse"
6492 name="SecurityExceptionFault" />
6493 <wsdl:fault message="impl:DuplicateRoleExceptionResponse"
6494 name="DuplicateRoleExceptionFault" />
6495 <wsdl:fault message="impl:RoleValidationExceptionResponse"
6496 name="RoleValidationExceptionFault" />
6497 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6498 name="UnsupportedOperationExceptionFault" />
6499 <wsdl:fault message="impl:ImplementationExceptionResponse"
6500 name="ImplementationExceptionFault" />
6501 </wsdl:operation>
6502
6503 <wsdl:operation name="updateRole">
6504 <wsdl:input message="impl:updateRoleRequest" name="updateRoleRequest" />
6505 <wsdl:output message="impl:updateRoleResponse" name="updateRoleResponse" />
6506 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6507 name="NoSuchRoleExceptionFault" />
6508 <wsdl:fault message="impl:RoleValidationExceptionResponse"
6509 name="RoleValidationExceptionFault" />
6510 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6511 name="UnsupportedOperationExceptionFault" />
6512 <wsdl:fault message="impl:SecurityExceptionResponse"
6513 name="SecurityExceptionFault" />
6514 <wsdl:fault message="impl:ImplementationExceptionResponse"
6515 name="ImplementationExceptionFault" />
6516 </wsdl:operation>
6517
6518 <wsdl:operation name="getRole">
6519 <wsdl:input message="impl:getRoleRequest" name="getRoleRequest" />
6520 <wsdl:output message="impl:getRoleResponse" name="getRoleResponse" />
6521 <wsdl:fault message="impl:SecurityExceptionResponse"
6522 name="SecurityExceptionFault" />
6523 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6524 name="NoSuchRoleExceptionFault" />
6525 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6526 name="UnsupportedOperationExceptionFault" />
6527 <wsdl:fault message="impl:ImplementationExceptionResponse"
6528 name="ImplementationExceptionFault" />
6529 </wsdl:operation>
6530
6531 <wsdl:operation name="undefineRole">
6532 <wsdl:input message="impl:undefineRoleRequest" name="undefineRoleRequest" />
6533 <wsdl:output message="impl:undefineRoleResponse" name="undefineRoleResponse" />
6534 <wsdl:fault message="impl:SecurityExceptionResponse"
6535 name="SecurityExceptionFault" />
6536 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6537 name="NoSuchRoleExceptionFault" />
6538 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6539 name="UnsupportedOperationExceptionFault" />
6540 <wsdl:fault message="impl:ImplementationExceptionResponse"
6541 name="ImplementationExceptionFault" />
6542 </wsdl:operation>
6543
6544 <wsdl:operation name="addPermissions">
6545 <wsdl:input message="impl:addPermissionsRequest" name="addPermissionsRequest" />
6546 <wsdl:output message="impl:addPermissionsResponse" name="addPermissionsResponse" />

```

```

6547 <wsdl:fault message="impl:SecurityExceptionResponse"
6548 name="SecurityExceptionFault" />
6549 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6550 name="NoSuchRoleExceptionFault" />
6551 <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6552 name="NoSuchPermissionExceptionFault" />
6553 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6554 name="UnsupportedOperationExceptionFault" />
6555 <wsdl:fault message="impl:ImplementationExceptionResponse"
6556 name="ImplementationExceptionFault" />
6557 </wsdl:operation>
6558
6559 <wsdl:operation name="setPermissions">
6560 <wsdl:input message="impl:setPermissionsRequest" name="setPermissionsRequest" />
6561 <wsdl:output message="impl:setPermissionsResponse" name="setPermissionsResponse" />
6562 <wsdl:fault message="impl:SecurityExceptionResponse"
6563 name="SecurityExceptionFault" />
6564 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6565 name="NoSuchRoleExceptionFault" />
6566 <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6567 name="NoSuchPermissionExceptionFault" />
6568 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6569 name="UnsupportedOperationExceptionFault" />
6570 <wsdl:fault message="impl:ImplementationExceptionResponse"
6571 name="ImplementationExceptionFault" />
6572 </wsdl:operation>
6573
6574 <wsdl:operation name="removePermissions">
6575 <wsdl:input message="impl:removePermissionsRequest"
6576 name="removePermissionsRequest" />
6577 <wsdl:output message="impl:removePermissionsResponse"
6578 name="removePermissionsResponse" />
6579 <wsdl:fault message="impl:SecurityExceptionResponse"
6580 name="SecurityExceptionFault" />
6581 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6582 name="NoSuchRoleExceptionFault" />
6583 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6584 name="UnsupportedOperationExceptionFault" />
6585 <wsdl:fault message="impl:ImplementationExceptionResponse"
6586 name="ImplementationExceptionFault" />
6587 </wsdl:operation>
6588
6589 <wsdl:operation name="getClientIdentityNames">
6590 <wsdl:input message="impl:getClientIdentityNamesRequest"
6591 name="getClientIdentityNamesRequest" />
6592 <wsdl:output message="impl:getClientIdentityNamesResponse"
6593 name="getClientIdentityNamesResponse" />
6594 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6595 name="UnsupportedOperationExceptionFault" />
6596 <wsdl:fault message="impl:SecurityExceptionResponse"
6597 name="SecurityExceptionFault" />
6598 <wsdl:fault message="impl:ImplementationExceptionResponse"
6599 name="ImplementationExceptionFault" />
6600 </wsdl:operation>
6601
6602 <wsdl:operation name="defineClientIdentity">
6603 <wsdl:input message="impl:defineClientIdentityRequest"
6604 name="defineClientIdentityRequest" />
6605 <wsdl:output message="impl:defineClientIdentityResponse"
6606 name="defineClientIdentityResponse" />
6607 <wsdl:fault message="impl:SecurityExceptionResponse"
6608 name="SecurityExceptionFault" />
6609 <wsdl:fault message="impl:DuplicateClientIdentityExceptionResponse"
6610 name="DuplicateClientIdentityExceptionFault" />
6611 <wsdl:fault message="impl:ClientIdentityValidationExceptionResponse"
6612 name="ClientIdentityValidationExceptionFault" />
6613 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6614 name="UnsupportedOperationExceptionFault" />
6615 <wsdl:fault message="impl:ImplementationExceptionResponse"
6616 name="ImplementationExceptionFault" />

```

```

6617 </wsdl:operation>
6618
6619 <wsdl:operation name="updateClientIdentity">
6620 <wsdl:input message="impl:updateClientIdentityRequest"
6621 name="updateClientIdentityRequest" />
6622 <wsdl:output message="impl:updateClientIdentityResponse"
6623 name="updateClientIdentityResponse" />
6624 <wsdl:fault message="impl:SecurityExceptionResponse"
6625 name="SecurityExceptionFault" />
6626 <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6627 name="NoSuchClientIdentityExceptionFault" />
6628 <wsdl:fault message="impl:ClientIdentityValidationExceptionResponse"
6629 name="ClientIdentityValidationExceptionFault" />
6630 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6631 name="UnsupportedOperationExceptionFault" />
6632 <wsdl:fault message="impl:ImplementationExceptionResponse"
6633 name="ImplementationExceptionFault" />
6634 </wsdl:operation>
6635
6636 <wsdl:operation name="getClientIdentity">
6637 <wsdl:input message="impl:getClientIdentityRequest"
6638 name="getClientIdentityRequest" />
6639 <wsdl:output message="impl:getClientIdentityResponse"
6640 name="getClientIdentityResponse" />
6641 <wsdl:fault message="impl:SecurityExceptionResponse"
6642 name="SecurityExceptionFault" />
6643 <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6644 name="NoSuchClientIdentityExceptionFault" />
6645 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6646 name="UnsupportedOperationExceptionFault" />
6647 <wsdl:fault message="impl:ImplementationExceptionResponse"
6648 name="ImplementationExceptionFault" />
6649 </wsdl:operation>
6650
6651 <wsdl:operation name="getClientPermissionNames">
6652 <wsdl:input message="impl:getClientPermissionNamesRequest"
6653 name="getClientPermissionNamesRequest" />
6654 <wsdl:output message="impl:getClientPermissionNamesResponse"
6655 name="getClientPermissionNamesResponse" />
6656 <wsdl:fault message="impl:SecurityExceptionResponse"
6657 name="SecurityExceptionFault" />
6658 <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6659 name="NoSuchClientIdentityExceptionFault" />
6660 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6661 name="UnsupportedOperationExceptionFault" />
6662 <wsdl:fault message="impl:ImplementationExceptionResponse"
6663 name="ImplementationExceptionFault" />
6664 </wsdl:operation>
6665
6666 <wsdl:operation name="undefineClientIdentity">
6667 <wsdl:input message="impl:undefineClientIdentityRequest"
6668 name="undefineClientIdentityRequest" />
6669 <wsdl:output message="impl:undefineClientIdentityResponse"
6670 name="undefineClientIdentityResponse" />
6671 <wsdl:fault message="impl:SecurityExceptionResponse"
6672 name="SecurityExceptionFault" />
6673 <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6674 name="NoSuchClientIdentityExceptionFault" />
6675 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6676 name="UnsupportedOperationExceptionFault" />
6677 <wsdl:fault message="impl:ImplementationExceptionResponse"
6678 name="ImplementationExceptionFault" />
6679 </wsdl:operation>
6680
6681 <wsdl:operation name="addRoles">
6682 <wsdl:input message="impl:addRolesRequest" name="addRolesRequest" />
6683 <wsdl:output message="impl:addRolesResponse" name="addRolesResponse" />
6684 <wsdl:fault message="impl:SecurityExceptionResponse"
6685 name="SecurityExceptionFault" />
6686 <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"

```

```

6687         name="NoSuchClientIdentityExceptionFault" />
6688     <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6689         name="NoSuchRoleExceptionFault" />
6690     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6691         name="UnsupportedOperationExceptionFault" />
6692     <wsdl:fault message="impl:ImplementationExceptionResponse"
6693         name="ImplementationExceptionFault" />
6694 </wsdl:operation>
6695
6696 <wsdl:operation name="removeRoles">
6697     <wsdl:input message="impl:removeRolesRequest" name="removeRolesRequest" />
6698     <wsdl:output message="impl:removeRolesResponse" name="removeRolesResponse" />
6699     <wsdl:fault message="impl:SecurityExceptionResponse"
6700         name="SecurityExceptionFault" />
6701     <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6702         name="NoSuchClientIdentityExceptionFault" />
6703     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6704         name="UnsupportedOperationExceptionFault" />
6705     <wsdl:fault message="impl:ImplementationExceptionResponse"
6706         name="ImplementationExceptionFault" />
6707 </wsdl:operation>
6708
6709 <wsdl:operation name="setRoles">
6710     <wsdl:input message="impl:setRolesRequest" name="setRolesRequest" />
6711     <wsdl:output message="impl:setRolesResponse" name="setRolesResponse" />
6712     <wsdl:fault message="impl:SecurityExceptionResponse"
6713         name="SecurityExceptionFault" />
6714     <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6715         name="NoSuchClientIdentityExceptionFault" />
6716     <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6717         name="NoSuchRoleExceptionFault" />
6718     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6719         name="UnsupportedOperationExceptionFault" />
6720     <wsdl:fault message="impl:ImplementationExceptionResponse"
6721         name="ImplementationExceptionFault" />
6722 </wsdl:operation>
6723
6724 <wsdl:operation name="getSupportedOperations">
6725     <wsdl:input message="impl:getSupportedOperationsRequest"
6726         name="getSupportedOperationsRequest" />
6727     <wsdl:output message="impl:getSupportedOperationsResponse"
6728         name="getSupportedOperationsResponse" />
6729     <wsdl:fault message="impl:ImplementationExceptionResponse"
6730         name="ImplementationExceptionFault" />
6731 </wsdl:operation>
6732
6733 <wsdl:operation name="getStandardVersion">
6734     <wsdl:input message="impl:getStandardVersionRequest"
6735         name="getStandardVersionRequest" />
6736     <wsdl:output message="impl:getStandardVersionResponse"
6737         name="getStandardVersionResponse" />
6738     <wsdl:fault message="impl:ImplementationExceptionResponse"
6739         name="ImplementationExceptionFault" />
6740 </wsdl:operation>
6741
6742 <wsdl:operation name="getVendorVersion">
6743     <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest" />
6744     <wsdl:output message="impl:getVendorVersionResponse"
6745         name="getVendorVersionResponse" />
6746     <wsdl:fault message="impl:ImplementationExceptionResponse"
6747         name="ImplementationExceptionFault" />
6748 </wsdl:operation>
6749 </wsdl:portType>
6750 <!-- ALEACSERVICE BINDING -->
6751
6752 <wsdl:binding name="ALEACServiceBinding" type="impl:ALEACServicePortType">
6753     <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
6754
6755     <wsdl:operation name="getPermissionNames">
6756         <wsdlsoap:operation soapAction="" />

```

```

6757     <wsdl:input name="getPermissionNamesRequest">
6758         <wsdlsoap:body use="literal"/>
6759     </wsdl:input>
6760     <wsdl:output name="getPermissionNamesResponse">
6761         <wsdlsoap:body use="literal"/>
6762     </wsdl:output>
6763     <wsdl:fault name="UnsupportedOperationExceptionFault">
6764         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6765     </wsdl:fault>
6766     <wsdl:fault name="SecurityExceptionFault">
6767         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6768     </wsdl:fault>
6769     <wsdl:fault name="ImplementationExceptionFault">
6770         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6771     </wsdl:fault>
6772 </wsdl:operation>
6773
6774 <wsdl:operation name="definePermission">
6775     <wsdlsoap:operation soapAction=""/>
6776     <wsdl:input name="definePermissionRequest">
6777         <wsdlsoap:body use="literal"/>
6778     </wsdl:input>
6779     <wsdl:output name="definePermissionResponse">
6780         <wsdlsoap:body use="literal"/>
6781     </wsdl:output>
6782     <wsdl:fault name="SecurityExceptionFault">
6783         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6784     </wsdl:fault>
6785     <wsdl:fault name="DuplicatePermissionExceptionFault">
6786         <wsdlsoap:fault name="DuplicatePermissionExceptionFault" use="literal"/>
6787     </wsdl:fault>
6788     <wsdl:fault name="PermissionValidationExceptionFault">
6789         <wsdlsoap:fault name="PermissionValidationExceptionFault" use="literal"/>
6790     </wsdl:fault>
6791     <wsdl:fault name="UnsupportedOperationExceptionFault">
6792         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6793     </wsdl:fault>
6794     <wsdl:fault name="ImplementationExceptionFault">
6795         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6796     </wsdl:fault>
6797 </wsdl:operation>
6798
6799 <wsdl:operation name="updatePermission">
6800     <wsdlsoap:operation soapAction=""/>
6801     <wsdl:input name="updatePermissionRequest">
6802         <wsdlsoap:body use="literal"/>
6803     </wsdl:input>
6804     <wsdl:output name="updatePermissionResponse">
6805         <wsdlsoap:body use="literal"/>
6806     </wsdl:output>
6807     <wsdl:fault name="NoSuchPermissionExceptionFault">
6808         <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
6809     </wsdl:fault>
6810     <wsdl:fault name="PermissionValidationExceptionFault">
6811         <wsdlsoap:fault name="PermissionValidationExceptionFault" use="literal"/>
6812     </wsdl:fault>
6813     <wsdl:fault name="UnsupportedOperationExceptionFault">
6814         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6815     </wsdl:fault>
6816     <wsdl:fault name="SecurityExceptionFault">
6817         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6818     </wsdl:fault>
6819     <wsdl:fault name="ImplementationExceptionFault">
6820         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6821     </wsdl:fault>
6822 </wsdl:operation>
6823
6824 <wsdl:operation name="getPermission">
6825     <wsdlsoap:operation soapAction=""/>
6826     <wsdl:input name="getPermissionRequest">

```

```

6827     <wsdlsoap:body use="literal"/>
6828 </wsdl:input>
6829 <wsdl:output name="getPermissionResponse">
6830     <wsdlsoap:body use="literal"/>
6831 </wsdl:output>
6832 <wsdl:fault name="SecurityExceptionFault">
6833     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6834 </wsdl:fault>
6835 <wsdl:fault name="NoSuchPermissionExceptionFault">
6836     <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
6837 </wsdl:fault>
6838 <wsdl:fault name="UnsupportedOperationExceptionFault">
6839     <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6840 </wsdl:fault>
6841 <wsdl:fault name="ImplementationExceptionFault">
6842     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6843 </wsdl:fault>
6844 </wsdl:operation>
6845
6846 <wsdl:operation name="undefinePermission">
6847     <wsdlsoap:operation soapAction=""/>
6848     <wsdl:input name="undefinePermissionRequest">
6849         <wsdlsoap:body use="literal"/>
6850     </wsdl:input>
6851     <wsdl:output name="undefinePermissionResponse">
6852         <wsdlsoap:body use="literal"/>
6853     </wsdl:output>
6854     <wsdl:fault name="SecurityExceptionFault">
6855         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6856     </wsdl:fault>
6857     <wsdl:fault name="NoSuchPermissionExceptionFault">
6858         <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
6859     </wsdl:fault>
6860     <wsdl:fault name="UnsupportedOperationExceptionFault">
6861         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6862     </wsdl:fault>
6863     <wsdl:fault name="ImplementationExceptionFault">
6864         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6865     </wsdl:fault>
6866 </wsdl:operation>
6867
6868 <wsdl:operation name="getRoleNames">
6869     <wsdlsoap:operation soapAction=""/>
6870     <wsdl:input name="getRoleNamesRequest">
6871         <wsdlsoap:body use="literal"/>
6872     </wsdl:input>
6873     <wsdl:output name="getRoleNamesResponse">
6874         <wsdlsoap:body use="literal"/>
6875     </wsdl:output>
6876     <wsdl:fault name="SecurityExceptionFault">
6877         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6878     </wsdl:fault>
6879     <wsdl:fault name="UnsupportedOperationExceptionFault">
6880         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6881     </wsdl:fault>
6882     <wsdl:fault name="ImplementationExceptionFault">
6883         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6884     </wsdl:fault>
6885 </wsdl:operation>
6886
6887 <wsdl:operation name="defineRole">
6888     <wsdlsoap:operation soapAction=""/>
6889     <wsdl:input name="defineRoleRequest">
6890         <wsdlsoap:body use="literal"/>
6891     </wsdl:input>
6892     <wsdl:output name="defineRoleResponse">
6893         <wsdlsoap:body use="literal"/>
6894     </wsdl:output>
6895     <wsdl:fault name="SecurityExceptionFault">
6896         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>

```

```

6897     </wsdl:fault>
6898     <wsdl:fault name="DuplicateRoleExceptionFault">
6899         <wsdlsoap:fault name="DuplicateRoleExceptionFault" use="literal"/>
6900     </wsdl:fault>
6901     <wsdl:fault name="RoleValidationExceptionFault">
6902         <wsdlsoap:fault name="RoleValidationExceptionFault" use="literal"/>
6903     </wsdl:fault>
6904     <wsdl:fault name="UnsupportedOperationExceptionFault">
6905         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6906     </wsdl:fault>
6907     <wsdl:fault name="ImplementationExceptionFault">
6908         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6909     </wsdl:fault>
6910 </wsdl:operation>
6911
6912 <wsdl:operation name="updateRole">
6913     <wsdlsoap:operation soapAction=""/>
6914     <wsdl:input name="updateRoleRequest">
6915         <wsdlsoap:body use="literal"/>
6916     </wsdl:input>
6917     <wsdl:output name="updateRoleResponse">
6918         <wsdlsoap:body use="literal"/>
6919     </wsdl:output>
6920     <wsdl:fault name="NoSuchRoleExceptionFault">
6921         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
6922     </wsdl:fault>
6923     <wsdl:fault name="RoleValidationExceptionFault">
6924         <wsdlsoap:fault name="RoleValidationExceptionFault" use="literal"/>
6925     </wsdl:fault>
6926     <wsdl:fault name="UnsupportedOperationExceptionFault">
6927         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6928     </wsdl:fault>
6929     <wsdl:fault name="SecurityExceptionFault">
6930         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6931     </wsdl:fault>
6932     <wsdl:fault name="ImplementationExceptionFault">
6933         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6934     </wsdl:fault>
6935 </wsdl:operation>
6936
6937 <wsdl:operation name="getRole">
6938     <wsdlsoap:operation soapAction=""/>
6939     <wsdl:input name="getRoleRequest">
6940         <wsdlsoap:body use="literal"/>
6941     </wsdl:input>
6942     <wsdl:output name="getRoleResponse">
6943         <wsdlsoap:body use="literal"/>
6944     </wsdl:output>
6945     <wsdl:fault name="SecurityExceptionFault">
6946         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6947     </wsdl:fault>
6948     <wsdl:fault name="NoSuchRoleExceptionFault">
6949         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
6950     </wsdl:fault>
6951     <wsdl:fault name="UnsupportedOperationExceptionFault">
6952         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6953     </wsdl:fault>
6954     <wsdl:fault name="ImplementationExceptionFault">
6955         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6956     </wsdl:fault>
6957 </wsdl:operation>
6958
6959 <wsdl:operation name="undefineRole">
6960     <wsdlsoap:operation soapAction=""/>
6961     <wsdl:input name="undefineRoleRequest">
6962         <wsdlsoap:body use="literal"/>
6963     </wsdl:input>
6964     <wsdl:output name="undefineRoleResponse">
6965         <wsdlsoap:body use="literal"/>
6966     </wsdl:output>

```

```

6967     <wsdl:fault name="SecurityExceptionFault">
6968         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6969     </wsdl:fault>
6970     <wsdl:fault name="NoSuchRoleExceptionFault">
6971         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
6972     </wsdl:fault>
6973     <wsdl:fault name="UnsupportedOperationExceptionFault">
6974         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6975     </wsdl:fault>
6976     <wsdl:fault name="ImplementationExceptionFault">
6977         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6978     </wsdl:fault>
6979 </wsdl:operation>
6980
6981 <wsdl:operation name="addPermissions">
6982     <wsdlsoap:operation soapAction=""/>
6983     <wsdl:input name="addPermissionsRequest">
6984         <wsdlsoap:body use="literal"/>
6985     </wsdl:input>
6986     <wsdl:output name="addPermissionsResponse">
6987         <wsdlsoap:body use="literal"/>
6988     </wsdl:output>
6989     <wsdl:fault name="SecurityExceptionFault">
6990         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6991     </wsdl:fault>
6992     <wsdl:fault name="NoSuchRoleExceptionFault">
6993         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
6994     </wsdl:fault>
6995     <wsdl:fault name="NoSuchPermissionExceptionFault">
6996         <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
6997     </wsdl:fault>
6998     <wsdl:fault name="UnsupportedOperationExceptionFault">
6999         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7000     </wsdl:fault>
7001     <wsdl:fault name="ImplementationExceptionFault">
7002         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7003     </wsdl:fault>
7004 </wsdl:operation>
7005
7006 <wsdl:operation name="setPermissions">
7007     <wsdlsoap:operation soapAction=""/>
7008     <wsdl:input name="setPermissionsRequest">
7009         <wsdlsoap:body use="literal"/>
7010     </wsdl:input>
7011     <wsdl:output name="setPermissionsResponse">
7012         <wsdlsoap:body use="literal"/>
7013     </wsdl:output>
7014     <wsdl:fault name="SecurityExceptionFault">
7015         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7016     </wsdl:fault>
7017     <wsdl:fault name="NoSuchRoleExceptionFault">
7018         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7019     </wsdl:fault>
7020     <wsdl:fault name="NoSuchPermissionExceptionFault">
7021         <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
7022     </wsdl:fault>
7023     <wsdl:fault name="UnsupportedOperationExceptionFault">
7024         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7025     </wsdl:fault>
7026     <wsdl:fault name="ImplementationExceptionFault">
7027         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7028     </wsdl:fault>
7029 </wsdl:operation>
7030
7031 <wsdl:operation name="removePermissions">
7032     <wsdlsoap:operation soapAction=""/>
7033     <wsdl:input name="removePermissionsRequest">
7034         <wsdlsoap:body use="literal"/>
7035     </wsdl:input>
7036     <wsdl:output name="removePermissionsResponse">

```

```

7037     <wsdlsoap:body use="literal"/>
7038 </wsdl:output>
7039 <wsdl:fault name="SecurityExceptionFault">
7040   <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7041 </wsdl:fault>
7042 <wsdl:fault name="NoSuchRoleExceptionFault">
7043   <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7044 </wsdl:fault>
7045 <wsdl:fault name="UnsupportedOperationExceptionFault">
7046   <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7047 </wsdl:fault>
7048 <wsdl:fault name="ImplementationExceptionFault">
7049   <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7050 </wsdl:fault>
7051 </wsdl:operation>
7052
7053 <wsdl:operation name="getClientIdentityNames">
7054   <wsdlsoap:operation soapAction=""/>
7055   <wsdl:input name="getClientIdentityNamesRequest">
7056     <wsdlsoap:body use="literal"/>
7057   </wsdl:input>
7058   <wsdl:output name="getClientIdentityNamesResponse">
7059     <wsdlsoap:body use="literal"/>
7060   </wsdl:output>
7061   <wsdl:fault name="UnsupportedOperationExceptionFault">
7062     <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7063   </wsdl:fault>
7064   <wsdl:fault name="SecurityExceptionFault">
7065     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7066   </wsdl:fault>
7067   <wsdl:fault name="ImplementationExceptionFault">
7068     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7069   </wsdl:fault>
7070 </wsdl:operation>
7071
7072 <wsdl:operation name="defineClientIdentity">
7073   <wsdlsoap:operation soapAction=""/>
7074   <wsdl:input name="defineClientIdentityRequest">
7075     <wsdlsoap:body use="literal"/>
7076   </wsdl:input>
7077   <wsdl:output name="defineClientIdentityResponse">
7078     <wsdlsoap:body use="literal"/>
7079   </wsdl:output>
7080   <wsdl:fault name="SecurityExceptionFault">
7081     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7082   </wsdl:fault>
7083   <wsdl:fault name="DuplicateClientIdentityExceptionFault">
7084     <wsdlsoap:fault name="DuplicateClientIdentityExceptionFault" use="literal"/>
7085   </wsdl:fault>
7086   <wsdl:fault name="ClientIdentityValidationExceptionFault">
7087     <wsdlsoap:fault name="ClientIdentityValidationExceptionFault" use="literal"/>
7088   </wsdl:fault>
7089   <wsdl:fault name="UnsupportedOperationExceptionFault">
7090     <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7091   </wsdl:fault>
7092   <wsdl:fault name="ImplementationExceptionFault">
7093     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7094   </wsdl:fault>
7095 </wsdl:operation>
7096
7097 <wsdl:operation name="updateClientIdentity">
7098   <wsdlsoap:operation soapAction=""/>
7099   <wsdl:input name="updateClientIdentityRequest">
7100     <wsdlsoap:body use="literal"/>
7101   </wsdl:input>
7102   <wsdl:output name="updateClientIdentityResponse">
7103     <wsdlsoap:body use="literal"/>
7104   </wsdl:output>
7105   <wsdl:fault name="SecurityExceptionFault">
7106     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>

```

```

7107     </wsdl:fault>
7108     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7109         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7110     </wsdl:fault>
7111     <wsdl:fault name="ClientIdentityValidationExceptionFault">
7112         <wsdlsoap:fault name="ClientIdentityValidationExceptionFault" use="literal"/>
7113     </wsdl:fault>
7114     <wsdl:fault name="UnsupportedOperationExceptionFault">
7115         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7116     </wsdl:fault>
7117     <wsdl:fault name="ImplementationExceptionFault">
7118         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7119     </wsdl:fault>
7120 </wsdl:operation>
7121
7122 <wsdl:operation name="getClientIdentity">
7123     <wsdlsoap:operation soapAction="" />
7124     <wsdl:input name="getClientIdentityRequest">
7125         <wsdlsoap:body use="literal"/>
7126     </wsdl:input>
7127     <wsdl:output name="getClientIdentityResponse">
7128         <wsdlsoap:body use="literal"/>
7129     </wsdl:output>
7130     <wsdl:fault name="SecurityExceptionFault">
7131         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7132     </wsdl:fault>
7133     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7134         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7135     </wsdl:fault>
7136     <wsdl:fault name="UnsupportedOperationExceptionFault">
7137         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7138     </wsdl:fault>
7139     <wsdl:fault name="ImplementationExceptionFault">
7140         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7141     </wsdl:fault>
7142 </wsdl:operation>
7143
7144 <wsdl:operation name="getClientPermissionNames">
7145     <wsdlsoap:operation soapAction="" />
7146     <wsdl:input name="getClientPermissionNamesRequest">
7147         <wsdlsoap:body use="literal"/>
7148     </wsdl:input>
7149     <wsdl:output name="getClientPermissionNamesResponse">
7150         <wsdlsoap:body use="literal"/>
7151     </wsdl:output>
7152     <wsdl:fault name="SecurityExceptionFault">
7153         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7154     </wsdl:fault>
7155     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7156         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7157     </wsdl:fault>
7158     <wsdl:fault name="UnsupportedOperationExceptionFault">
7159         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7160     </wsdl:fault>
7161     <wsdl:fault name="ImplementationExceptionFault">
7162         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7163     </wsdl:fault>
7164 </wsdl:operation>
7165
7166 <wsdl:operation name="undefineClientIdentity">
7167     <wsdlsoap:operation soapAction="" />
7168     <wsdl:input name="undefineClientIdentityRequest">
7169         <wsdlsoap:body use="literal"/>
7170     </wsdl:input>
7171     <wsdl:output name="undefineClientIdentityResponse">
7172         <wsdlsoap:body use="literal"/>
7173     </wsdl:output>
7174     <wsdl:fault name="SecurityExceptionFault">
7175         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7176     </wsdl:fault>

```

```

7177     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7178         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7179     </wsdl:fault>
7180     <wsdl:fault name="UnsupportedOperationExceptionFault">
7181         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7182     </wsdl:fault>
7183     <wsdl:fault name="ImplementationExceptionFault">
7184         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7185     </wsdl:fault>
7186 </wsdl:operation>
7187
7188 <wsdl:operation name="addRoles">
7189     <wsdlsoap:operation soapAction=""/>
7190     <wsdl:input name="addRolesRequest">
7191         <wsdlsoap:body use="literal"/>
7192     </wsdl:input>
7193     <wsdl:output name="addRolesResponse">
7194         <wsdlsoap:body use="literal"/>
7195     </wsdl:output>
7196     <wsdl:fault name="SecurityExceptionFault">
7197         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7198     </wsdl:fault>
7199     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7200         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7201     </wsdl:fault>
7202     <wsdl:fault name="NoSuchRoleExceptionFault">
7203         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7204     </wsdl:fault>
7205     <wsdl:fault name="UnsupportedOperationExceptionFault">
7206         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7207     </wsdl:fault>
7208     <wsdl:fault name="ImplementationExceptionFault">
7209         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7210     </wsdl:fault>
7211 </wsdl:operation>
7212
7213 <wsdl:operation name="removeRoles">
7214     <wsdlsoap:operation soapAction=""/>
7215     <wsdl:input name="removeRolesRequest">
7216         <wsdlsoap:body use="literal"/>
7217     </wsdl:input>
7218     <wsdl:output name="removeRolesResponse">
7219         <wsdlsoap:body use="literal"/>
7220     </wsdl:output>
7221     <wsdl:fault name="SecurityExceptionFault">
7222         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7223     </wsdl:fault>
7224     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7225         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7226     </wsdl:fault>
7227     <wsdl:fault name="UnsupportedOperationExceptionFault">
7228         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7229     </wsdl:fault>
7230     <wsdl:fault name="ImplementationExceptionFault">
7231         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7232     </wsdl:fault>
7233 </wsdl:operation>
7234
7235 <wsdl:operation name="setRoles">
7236     <wsdlsoap:operation soapAction=""/>
7237     <wsdl:input name="setRolesRequest">
7238         <wsdlsoap:body use="literal"/>
7239     </wsdl:input>
7240     <wsdl:output name="setRolesResponse">
7241         <wsdlsoap:body use="literal"/>
7242     </wsdl:output>
7243     <wsdl:fault name="SecurityExceptionFault">
7244         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7245     </wsdl:fault>
7246     <wsdl:fault name="NoSuchClientIdentityExceptionFault">

```

```

7247     <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7248 </wsdl:fault>
7249 <wsdl:fault name="NoSuchRoleExceptionFault">
7250   <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7251 </wsdl:fault>
7252 <wsdl:fault name="UnsupportedOperationExceptionFault">
7253   <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7254 </wsdl:fault>
7255 <wsdl:fault name="ImplementationExceptionFault">
7256   <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7257 </wsdl:fault>
7258 </wsdl:operation>
7259
7260 <wsdl:operation name="getSupportedOperations">
7261   <wsdlsoap:operation soapAction=""/>
7262   <wsdl:input name="getSupportedOperationsRequest">
7263     <wsdlsoap:body use="literal"/>
7264   </wsdl:input>
7265   <wsdl:output name="getSupportedOperationsResponse">
7266     <wsdlsoap:body use="literal"/>
7267   </wsdl:output>
7268   <wsdl:fault name="ImplementationExceptionFault">
7269     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7270   </wsdl:fault>
7271 </wsdl:operation>
7272
7273 <wsdl:operation name="getStandardVersion">
7274   <wsdlsoap:operation soapAction=""/>
7275   <wsdl:input name="getStandardVersionRequest">
7276     <wsdlsoap:body use="literal"/>
7277   </wsdl:input>
7278   <wsdl:output name="getStandardVersionResponse">
7279     <wsdlsoap:body use="literal"/>
7280   </wsdl:output>
7281   <wsdl:fault name="ImplementationExceptionFault">
7282     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7283   </wsdl:fault>
7284 </wsdl:operation>
7285
7286 <wsdl:operation name="getVendorVersion">
7287   <wsdlsoap:operation soapAction=""/>
7288   <wsdl:input name="getVendorVersionRequest">
7289     <wsdlsoap:body use="literal"/>
7290   </wsdl:input>
7291   <wsdl:output name="getVendorVersionResponse">
7292     <wsdlsoap:body use="literal"/>
7293   </wsdl:output>
7294   <wsdl:fault name="ImplementationExceptionFault">
7295     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7296   </wsdl:fault>
7297 </wsdl:operation>
7298 </wsdl:binding>
7299
7300 <!-- ALEACSERVICE -->
7301 <wsdl:service name="ALEACService">
7302   <wsdl:port binding="impl:ALEACServiceBinding" name="ALEACServicePort">
7303     <!-- The value of the location attribute below is an example only;
7304           Implementations are free to choose any appropriate URL. -->
7305     <wsdlsoap:address location="http://localhost:8080/services/ALEACService"/>
7306   </wsdl:port>
7307 </wsdl:service>
7308 </wsdl:definitions>

```

## 7309 **5 Bindings for the Reading and Writing Callback APIs**

7310 This section specifies XML-based bindings for the ALECallback and  
7311 ALECCallback interfaces, through which the ALE Reading API and the ALE

7312 Writing API, respectively, deliver asynchronous notifications to subscribers. Each  
7313 binding of these interfaces specifies a syntax for notification URIs. A notification URI is  
7314 supplied by an ALE client as a parameter of the `subscribe` and `unsubscribe`  
7315 methods of the ALE Reading or Writing API. The notification URI both selects a  
7316 binding of the callback interface to be used for that subscriber, and provides addressing  
7317 information in a manner specified by each binding below.

7318 Each subsection below specifies the conformance requirement (MAY, SHOULD,  
7319 SHALL) for each binding. Implementations MAY provide additional, vendor-specific  
7320 bindings of the callback interfaces. If an implementation provides an additional binding  
7321 of the callback interface, it SHALL use a URI scheme that does not conflict with any of  
7322 the standardized bindings.

7323 All notification URIs recognized by bindings as legal, whether the binding is  
7324 standardized as a part of this specification or not, SHALL conform to the general syntax  
7325 for URIs as defined in [RFC2396]. Each binding may impose additional constraints upon  
7326 syntax.

7327 A given mechanism for delivery of asynchronous results may be used in a binding of the  
7328 `ALECallback` (Reading API) interface and the `ALECCallback` (Writing API)  
7329 interface. This specification defines bindings of the `ALECallback` and  
7330 `ALECCallback` for each of four delivery mechanisms: HTTP, raw TCP, File, and  
7331 HTTP over TLS (HTTPS). Because the specifications of the `ALECallback` binding  
7332 and the `ALECCallback` binding for a given delivery mechanism are nearly identical,  
7333 they are pairwise combined in the specifications below.

## 7334 **5.1 HTTP Bindings**

7335 The HTTP bindings of the `ALECallback` and `ALECCallback` interfaces provide  
7336 for delivery of `ECReports` or `CCReports`, respectively, in XML via the HTTP  
7337 protocol using the POST operation. Implementations SHOULD provide support for these  
7338 bindings.

7339 The syntax for HTTP notification URIs as used by these bindings is defined in  
7340 [RFC2616], Section 3.2.2. Informally, an HTTP URI has one of the two following  
7341 forms:

7342 `http://host:port/remainder-of-URL`

7343 `http://host/remainder-of-URL`

7344 where

- 7345 • *host* is the DNS name or IP address of the host where the callback receiver is  
7346 listening for incoming HTTP connections.
- 7347 • *port* is the TCP port on which the callback receiver is listening for incoming HTTP  
7348 connections. The port and the preceding colon character may be omitted, in which  
7349 case the port defaults to 80.
- 7350 • *remainder-of-URL* is the URL to which an HTTP POST operation will be  
7351 directed.

7352 The ALE implementation delivers event cycle or command cycle reports by sending an  
7353 HTTP POST request to the callback receiver designated in the URI, where  
7354 *remainder-of-URL* is included in the HTTP *request-line* (as defined in  
7355 [RFC2616]), and where the payload is the *ECReports* instance or *CCReports*  
7356 instance encoded in XML according to the schema specified in Section 3.5 or  
7357 Section 3.6, respectively.

7358 The interpretation by the ALE implementation of the response code returned by the  
7359 callback receiver is outside the scope of this specification; however, all implementations  
7360 SHALL interpret a response code 2xx (that is, any response code between 200 and 299,  
7361 inclusive) as a normal response, not indicative of any error.

## 7362 5.2 TCP Bindings

7363 The TCP bindings of the *ALECallback* and *ALECCallback* interfaces provide for  
7364 delivery of *ECReports* or *CCReports*, respectively, in XML via a raw TCP  
7365 connection. Implementations SHOULD provide support for these bindings.

7366 The syntax for TCP notification URIs as used by these bindings is as follows:

7367 `tcp_URL = "tcp:" "//" host ":" port`

7368 where the syntax definition for *host* and *port* is specified in [RFC2396].

7369 Informally, a TCP URI has the following form:

7370 `tcp://host:port`

7371 The ALE implementation delivers an event cycle or command cycle report by opening a  
7372 new TCP connection to the specified host and port, writing to the connection the  
7373 *ECReports* instance or *CCReports* instance encoded in XML according to the  
7374 schema specified in Section 3.5 or Section 3.6, respectively, and then closing the  
7375 connection. The ALE implementation SHALL NOT require a reply or  
7376 acknowledgement.

## 7377 5.3 FILE Bindings

7378 The FILE bindings of the *ALECallback* and *ALECCallback* interfaces provide for  
7379 writing of *ECReports* or *CCReports*, respectively, in XML to a file.  
7380 Implementations MAY provide support for these bindings.

7381 The syntax for FILE notification URIs as used by these bindings is defined in  
7382 [RFC1738], Section 3.10. Informally, a FILE URI has one of the two following forms:

7383 `file://host/path`

7384 `file:///path`

7385 where

- 7386 • *host* is the DNS name or IP address of a remote host whose filesystem is accessible  
7387 to the ALE implementation.

7388 • *path* is the pathname of a file within the remote filesystem, or the local filesystem if  
7389 *host* is omitted.

7390 The ALE implementation delivers an event cycle or command cycle report by appending  
7391 to the specified file the `ECReports` or `CCReports` instance encoded in XML  
7392 according to the schema specified in Section 3.5 or Section 3.6, respectively. Note that if  
7393 more than one event cycle completes, the file will contain a concatenation of XML  
7394 documents, rather than a single XML document.

7395 Implementations of this binding may impose additional constraints on the use of the FILE  
7396 URI. For example, some implementations of this binding may support only a local  
7397 filesystem while others may support only a remote filesystem, some implementations of  
7398 this binding may impose further restrictions on the syntax of the *path* component, and  
7399 so forth. This specification also does not define the behavior when *path* names a  
7400 directory; the behavior in that case is implementation dependent.

7401 *Rationale (non-normative): The intended use for the FILE bindings is for debugging,*  
7402 *and hence the specification is intentionally lax in order to give freedom to*  
7403 *implementations to provide the most appropriate and useful facility given the unique*  
7404 *circumstances of that implementation.*

## 7405 5.4 HTTPS Bindings

7406 The HTTPS bindings of the `ALECallback` and `ALECCallback` interfaces provide  
7407 for delivery of `ECReports` or `CCReports`, respectively, in XML via the HTTP  
7408 protocol using the POST operation, secured via TLS. The HTTPS protocol provides  
7409 link-level security, and optionally mutual authentication between an ALE implementation  
7410 and its callback receivers. Implementations MAY provide support for these bindings.

7411 The syntax for HTTPS notification URIs as used by these bindings is defined in  
7412 [RFC2818], Section 2.4, which in turn is identical to the syntax defined in [RFC2616],  
7413 Section 3.2.2, with the substitution of `https` for `http`. Informally, an HTTPS URI has  
7414 one of the two following forms:

7415 `https://host:port/remainder-of-URL`

7416 `https://host/remainder-of-URL`

7417 where

- 7418 • *host* is the DNS name or IP address of the host where the callback receiver is  
7419 listening for incoming HTTPS connections.
- 7420 • *port* is the TCP port on which the receiver is listening for incoming HTTPS  
7421 connections. The port and the preceding colon character may be omitted, in which  
7422 case the port defaults to 443.
- 7423 • *remainder-of-URL* is the URL to which an HTTP POST operation will be  
7424 directed.

7425 The ALE implementation SHALL deliver event cycle or command cycle reports by  
7426 sending an HTTP POST request to the callback receiver designated in the URI, where

7427 *remainder-of-URL* is included in the HTTP *request-line* (as defined in  
7428 [RFC2616]), and where the payload is the *ECReports* or *CCReports* instance  
7429 encoded in XML according to the schema specified in Section 0 or Section 3.6,  
7430 respectively.

7431 For these bindings, HTTP SHALL be used over TLS as defined in [RFC2818]. TLS for  
7432 this purpose SHALL be implemented as defined in [RFC2246] except that the mandatory  
7433 cipher suite is *TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA*, as defined in [RFC3268] with  
7434 *CompressionMethod.null*. Implementations MAY support additional cipher suites and  
7435 compression algorithms as desired.

7436 The interpretation by the ALE implementation of the response code returned by the  
7437 callback receiver is outside the scope of this specification; however, all implementations  
7438 SHALL interpret a response code 2xx (that is, any response code between 200 and 299,  
7439 inclusive) as a normal response, not indicative of any error.

7440 *Warning (non-normative): The HTTPS report delivery mechanism is concerned only*  
7441 *with the delivery of ECReports and CCReports from the ALE implementation to*  
7442 *subscribed receivers. Using this mechanism is more secure than using cleartext*  
7443 *protocols such as HTTP or “raw” TCP but it does not ensure that the system as a whole*  
7444 *is secure. In a typical RFID system there will be other communications paths (for*  
7445 *example reader-to-ALE, and vendor-specific ALE administrative interfaces) that would*  
7446 *also need to be secure in order to claim that the entire system was secure.*

## 7447 **6 Appendix: Schema and WSDL Differences from ALE** 7448 **1.0 (non-normative)**

7449 This section enumerates differences between the XSD/WSDL for the ALE 1.1 Reading  
7450 API and the XSD/WSDL for ALE 1.0.

### 7451 **6.1 Fully Compatible Changes**

7452 The changes below are fully compatible, in that a document conforming to the ALE 1.1  
7453 schema is valid according to the ALE 1.0 schema, and vice versa.

- 7454 • In the definition of *ECReportGroupList*, an *anyAttribute* declaration is  
7455 added. This is part of the extensibility mechanism defined in Section 3.2, but was  
7456 inadvertently omitted from the ALE 1.0 schema.
- 7457 • In the definition of *ECReportOutputSpec*, an *anyAttribute* declaration is  
7458 added. This is part of the extensibility mechanism defined in Section 3.2, but was  
7459 inadvertently omitted from the ALE 1.0 schema.
- 7460 • Several types have been extended by adding new subelements to the *<extension>*  
7461 element, following the extensibility mechanism defined in Section 3.2. A nested  
7462 *<extension>* element is also added, in order to provide for further extensions in  
7463 later versions of the specification. The types that have been extended in this way  
7464 include: *ECBoundarySpec*, *ECFilterSpec*, *ECGroupSpec*,

7465 ECReportGroupListMember, ECReportOutputSpec, ECReportSpec,  
7466 and ECSpec.

7467 • The ECReport type has been extended by the addition of a new, optional attribute,  
7468 following the extensibility mechanism defined in Section 3.2.

7469 • The following new types have been added to the schema:  
7470 ECInitiationCondition, ECReportMemberField,  
7471 ECReportOutputFieldSpec, ECStatProfileName, ECTagStat,  
7472 ECTagTimestampStat, ECFieldSpec, ECFilterListMember,  
7473 ECIncludeExclude, ECReaderStat, and ECSightingStat.

## 7474 6.2 Non-Forward Compatible Changes

7475 The changes below are backward compatible, meaning that a document conforming to  
7476 the ALE 1.0 schema is valid according to the ALE 1.1 schema, but not forward  
7477 compatible, meaning that a document conforming to the ALE 1.1 schema may not be  
7478 valid according to the ALE 1.0 schema in some cases. The lack of forward compatibility  
7479 stems from areas of non-extensibility in the ALE 1.0 schema.

- 7480 • The ECGroupSpec type has been changed to include the extensibility mechanism,  
7481 and the extensibility mechanism is used to add a new, optional subelement. An  
7482 ECGroupSpec that includes the new, optional fieldspec subelement will not be  
7483 valid according to the ALE 1.0 schema.
- 7484 • All enumeration types have been made extensible in ALE 1.1, by changing the  
7485 schema to accept any string. Consequently, a document that contains an enumeration  
7486 value not specified in ALE 1.0 will not be valid according to the ALE 1.0 schema.  
7487 The enumeration types are: ECReportSetEnum, ECTerminationCondition,  
7488 and ECTimeUnit, along with ImplementationExceptionSeverity  
7489 defined in the WSDL.

## 7490 6.3 Corrections

7491 Two errors in the ALE 1.0 schema have been corrected in the ALE 1.1 schema.

- 7492 • In the ECReportSetSpec type, the set attribute is now declared to be required.  
7493 In the ALE 1.0 schema, the use declaration was omitted which implied that the  
7494 attribute was optional.
- 7495 • In the ECTime type, the unit attribute is now declared to be required. In the ALE  
7496 1.0 schema, the use declaration was omitted which implied that the attribute was  
7497 optional.

7498 Technically, these changes are not backward compatible, in that a document that  
7499 conforms to the ALE 1.0 schema may not be valid according to the ALE 1.1 schema.  
7500 However, this only occurs if the document omitted one of the two attributes listed above.  
7501 The main body of the ALE 1.0 specification did not specify that these attributes are  
7502 optional, and no behavior was defined for the case where they are omitted. Therefore, the

7503 lack of back-compatibility only occurs for a document whose meaning is not defined  
7504 according to the ALE 1.0 specification, and so the lack of compatibility is unlikely to  
7505 arise in practice.

## 7506 **6.4 Changes in Idiom**

7507 The following changes to the schema do not affect what documents are valid or invalid  
7508 according to the schema, but simply changes the XSD description to an alternative,  
7509 equivalent form.

- 7510 • The `ECTrigger` type, which is equivalent to `xsd:string`, is now defined as a  
7511 (trivial) restriction of `xsd:string` rather than as a (trivial) extension of  
7512 `xsd:string`. This allows `ECTrigger` to be used as the type of an attribute as  
7513 well as the type of an element.
- 7514 • Several type names were defined in the ALE 1.0 schema in places where the UML  
7515 specified a field of type `List<...>`. These type names appeared only in the schema,  
7516 and not in the UML. To avoid confusion, in the ALE 1.1 schema these type names  
7517 are eliminated by embedding the type definition in the context where the reference to  
7518 the list type occurs. The type names that have been eliminated in this way are:  
7519 `ECExcludePatterns`, `ECIncludePatterns`, `ECLogicalReaders`,  
7520 `ECReportList`, and `ECReportSpecs`.

## 7521 **7 References**

- 7522 [ALE1.1Part1] EPCglobal, "The Application Level Events (ALE) Specification,  
7523 Version 1.1 Part I: Core Specification," EPCglobal Working Draft, October 2007.
- 7524 [ISODir2] ISO, "Rules for the structure and drafting of International Standards  
7525 (ISO/IEC Directives, Part 2, 2001, 4th edition)," July 2002.
- 7526 [RFC1738] T. Berners-Lee, L. Masinter, M. McCahill, "Uniform Resource Locators  
7527 (URL)," RFC 1738, December 1994, <http://www.ietf.org/rfc/rfc1738>.
- 7528 [RFC2246] T. Dierks, C. Allen, "The TLS Protocol, Version 1.0," RFC2246, January  
7529 1999, <http://www.ietf.org/rfc/rfc2246>.
- 7530 [RFC2396] T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers  
7531 (URI): Generic Syntax," RFC2396, August 1998, <http://www.ietf.org/rfc/rfc2396>.
- 7532 [RFC2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T.  
7533 Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1," RFC2616, June 1999,  
7534 <http://www.ietf.org/rfc/rfc2616>.
- 7535 [RFC2818] E. Escorla, "HTTP Over TLS," RFC2818, May 2000,  
7536 <http://www.ietf.org/rfc/rfc2818>.
- 7537 [RFC3268] P. Chown, "Advanced Encryption Standard (AES) Ciphersuites for  
7538 Transport Layer Security (TLS)," RFC3268, June 2002, <http://www.ietf.org/rfc/rfc3268>.

- 7539 [SOAP1.1] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F.  
7540 Nielsen, S. Thatte, and D. Winer, “Simple Object Access Protocol (SOAP) 1.1,” W3C  
7541 Note, May 2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>.
- 7542 [WSDL1.1] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, “Web Services  
7543 Description Language (WSDL) 1.1,” W3C Note, March 2001,  
7544 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>.
- 7545 [WSI] K. Ballinger, D. Ehnebuske, M. Gudgin, M. Nottingham, P. Yendluri, “Basic  
7546 Profile Version 1.0,” WS-i Final Material, April 2004, [http://www.ws-](http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html)  
7547 [i.org/Profiles/BasicProfile-1.0-2004-04-16.html](http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html).
- 7548 [XML1.0] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau,  
7549 “Extensible Markup Language (XML) 1.0 (Third Edition),” W3C Recommendation,  
7550 February 2004, <http://www.w3.org/TR/2004/REC-xml-20040204/>.
- 7551 [XSD1] H. Thompson, D. Beech, M. Maloney, N. Mendelsohn, “XML Schema Part 1:  
7552 Structures,” W3C Recommendation, May 2001, <http://www.w3.org/TR/xmlschema-1/>.
- 7553 [XSD2] P. Biron, A. Malhotra, “XML Schema Part 2: Datatypes,” W3C  
7554 Recommendation, May 2001, <http://www.w3.org/TR/xmlschema-2/>.
- 7555 [XMLVersioning] D. Orchard, “Versioning XML Vocabularies,” December 2003,  
7556 <http://www.xml.com/pub/a/2003/12/03/versioning.html>.

## 7557 **8 Credits**

7558 See [ALE1.1Part1], Section 18.

7559