



1

2 **The Application Level Events (ALE) Specification**  
3 **Version 1.1.1**  
4 **Part II: XML and SOAP Bindings**

5 EPCglobal Ratified Standard  
6 13 March 2009  
7 **Previous Version: 1.1**

8 **Disclaimer**

9 EPCglobal Inc™ is providing this document as a service to interested industries.  
10 This document was developed through a consensus process of interested  
11 parties.

12 Although efforts have been to assure that the document is correct, reliable, and  
13 technically accurate, EPCglobal Inc. makes NO WARRANTY, EXPRESS OR  
14 IMPLIED, THAT THIS DOCUMENT IS CORRECT, WILL NOT REQUIRE  
15 MODIFICATION AS EXPERIENCE AND TECHNOLOGICAL ADVANCES  
16 DICTATE, OR WILL BE SUITABLE FOR ANY PURPOSE OR WORKABLE IN  
17 ANY APPLICATION, OR OTHERWISE. Use of this document is with the  
18 understanding that EPCglobal Inc. has no liability for any claim to the contrary, or  
19 for any damage or loss of any kind or nature.  
20

21

**Copyright notice**

22

© 2006, 2007, 2008, 2009 EPCglobal Inc.

23

All rights reserved. Unauthorized reproduction, modification, and/or use of this document is not permitted. Requests for permission to reproduce should be addressed to [epcglobal@epcglobalinc.org](mailto:epcglobal@epcglobalinc.org).

24

25

26

27

EPCglobal Inc.™ is providing this document as a service to interested industries. This document was developed through a consensus process of interested parties. Although efforts have been to assure that the document is correct, reliable, and technically accurate, EPCglobal Inc. makes NO WARRANTY, EXPRESS OR IMPLIED, THAT THIS DOCUMENT IS CORRECT, WILL NOT REQUIRE MODIFICATION AS EXPERIENCE AND TECHNOLOGICAL ADVANCES DICTATE, OR WILL BE SUITABLE FOR ANY PURPOSE OR WORKABLE IN ANY APPLICATION, OR OTHERWISE. Use of this Document is with the understanding that EPCglobal Inc. has no liability for any claim to the contrary, or for any damage or loss of any kind or nature

28

29

30

31

32

33

34

35

## 36 Abstract

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

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

- 52 • It provides a means for clients to specify, in a high-level, declarative way, what data  
53 they are interested in or what operations they want performed, without dictating an  
54 implementation. The interface is designed to give implementations the widest  
55 possible latitude in selecting strategies for carrying out client requests; such strategies  
56 may be influenced by performance goals, the native abilities of readers or other  
57 devices which may carry out certain filtering or counting operations at the level of  
58 firmware or RF protocol, and so forth.
- 59 • It provides a standardized format for reporting accumulated, filtered data and results  
60 from carrying out operations that is largely independent of where the data originated  
61 or how it was processed.
- 62 • It abstracts the channels through which data carriers are accessed into a higher-level  
63 notion of “logical reader,” often synonymous with “location,” hiding from clients the  
64 details of exactly what physical devices were used to interact with data relevant to a  
65 particular logical location. This allows changes to occur at the physical layer (for  
66 example, replacing a 2-port multi-antenna reader at a loading dock door with three  
67 “smart antenna” readers) without affecting client applications. Similarly, it abstracts  
68 away the fine-grained details of how data is gathered (*e.g.*, how many individual tag  
69 read attempts were carried out). These features of abstraction are a consequence of  
70 the way the data specification and reporting aspects of the interface are designed.
- 71 • It abstracts the addressing of information stored on Tags and other data carriers into a  
72 higher-level notion of named, typed “fields,” hiding from clients the details of how a  
73 particular data element is encoded into a bit-level representation and stored at a  
74 particular address within a data carrier’s memory. This allows application logic to  
75 remain invariant despite differences between the memory organization of different  
76 data carriers (for example, differences between Gen 1 and Gen 2 RFID Tags), and

77 also shields application logic from having to understand complex layout or data  
78 parsing rules.

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

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

## 89 **Audience for this document**

90 The target audience for this specification includes:

- 91 • EPC Middleware vendors
- 92 • Reader vendors
- 93 • Application developers
- 94 • System integrators

## 95 **Status of this document**

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

99 The Technical Steering Committee (TSC) approved the errata fixes to ALE 1.1, Part 1,  
100 that was originally ratified on February 27, 2008. Part 2's title page was updated to carry  
101 the same version number, 1.1.1, as Part 1, on March 13, 2009.

102 Comments on this document should be sent to the EPCglobal Software Action Group  
103 Filtering and Collection Working Group mailing list  
104 [sag\\_fcl\\_1\\_wg@lists.epcglobalinc.org](mailto:sag_fcl_1_wg@lists.epcglobalinc.org).

## 105 **Errata Fixed in ALE 1.1.1**

106 All fixed errata in ALE 1.1.1 apply to Part I of the specification; there are no changes to  
107 Part II. For a comparison between ALE 1.1 and ALE 1.0, please see Section 16 of Part I.

108

109 **Table of Contents**

110 1 Introduction ..... 6

111 2 Terminology and Typographical Conventions..... 6

112 3 XML Schemas..... 6

113 3.1 Organization of the XML Schemas ..... 7

114 3.2 Extensibility Mechanism ..... 7

115 3.2.1 Extensibility Design Rules – Classes..... 8

116 3.2.2 Extensibility Design Rules – Interfaces..... 10

117 3.2.3 Extensibility Design Rules – Enumerations ..... 10

118 3.3 EPCglobal Base Schema ..... 11

119 3.4 Schema for Datatypes Common to the Reading API and Writing API..... 12

120 3.5 ALE Reading API Schema ..... 13

121 3.5.1 ECSpec – Example 1 (non-normative) ..... 21

122 3.5.2 ECSpec – Example 2 (non-normative) ..... 22

123 3.5.3 ECReports – Example 1 (non-normative)..... 23

124 3.5.4 ECReports – Example 2 (non-normative)..... 24

125 3.6 ALE Writing API Schema..... 25

126 3.7 ALE Tag Memory API Schema ..... 33

127 3.8 ALE Logical Reader API Schema..... 35

128 3.9 ALE Access Control API Schema..... 36

129 4 SOAP Bindings ..... 38

130 4.1 Organization of the SOAP Bindings ..... 38

131 4.2 Link Level Security ..... 38

132 4.3 ALE Reading API SOAP Binding..... 38

133 4.4 ALE Writing API SOAP Binding ..... 48

134 4.5 ALE Tag Memory API SOAP Binding..... 73

135 4.6 ALE Logical Reader API SOAP Binding ..... 78

136 4.7 ALE Access Control API SOAP Binding ..... 90

137 5 Bindings for the Reading and Writing Callback APIs ..... 112

138 5.1 HTTP Bindings..... 113

139 5.2 TCP Bindings ..... 114

140 5.3 FILE Bindings ..... 114

141	5.4	HTTPS Bindings.....	115
142	6	Appendix: Schema and WSDL Differences from ALE 1.0 (non-normative).....	116
143	6.1	Fully Compatible Changes .....	116
144	6.2	Non-Forward Compatible Changes.....	117
145	6.3	Corrections.....	117
146	6.4	Changes in Idiom.....	118
147	7	References .....	118
148	8	Credits .....	119
149			
150			

## 151 **1 Introduction**

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

158 Implementations may provide additional bindings of the API, including bindings to  
159 particular programming languages.

## 160 **2 Terminology and Typographical Conventions**

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

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

168 The following typographical conventions are used throughout the document:

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

## 175 **3 XML Schemas**

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

182 The schemas defined herein conform to EPCglobal standard schema design rules. The  
183 schema below imports the EPCglobal standard base schema, as mandated by the design  
184 rules.

185 **3.1 Organization of the XML Schemas**

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

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

193 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 ale-common	ECSpec ECReports
alecc	3.6	EPCglobal ale-common	CCSpec CCParameterList CCReports EPCCacheSpec EPCPatternList AssocTableSpec AssocTableEntryList RNGSpec
aletm	3.7	EPCglobal	TMSpec
alelr	3.8	EPCglobal	LRSpec LRProperty
aleac	3.9	EPCglobal	ACPermission ACRole ACClientIdentity

194

195 **3.2 Extensibility Mechanism**

196 The XML schema in this section implements the <<extension point>> given in  
 197 the UML using a methodology described in [XMLVersioning]. This methodology  
 198 provides for both vendor extension, and for extension by EPCglobal in future versions of  
 199 this specification or in supplemental specifications. Extensions introduced through this  
 200 mechanism will be *backward compatible*, in that documents conforming to older versions  
 201 of the schema will also conform to newer versions of the standard schema and to schema  
 202 containing vendor-specific extensions. Extensions will also be *forward compatible*, in  
 203 that documents that contain vendor extensions or that conform to newer versions of the  
 204 standard schema will also conform to older versions of the schema.

205 When a document contains extensions (vendor-specific or standardized in newer versions  
206 of schema), it may conform to more than one schema. For example, a document  
207 containing vendor extensions to the ALE 1.1 schema will conform both to the EPCglobal  
208 ALE 1.1 schema and to a vendor-specific schema that includes the vendor extensions. In  
209 this example, when the document is parsed using the standard schema there will be no  
210 validation of the extension elements and attributes, but when the document is parsed  
211 using the vendor-specific schema the extensions will be validated. Similarly, a document  
212 containing new features introduced in the ALE 1.1 schema will conform both to the ALE  
213 1.0 schema and to the ALE 1.1 schema, but validation of the new features will only be  
214 available using the ALE 1.1 schema.

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

### 218 3.2.1 Extensibility Design Rules – Classes

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

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

```
228     <xsd:sequence>  
229         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"  
230             namespace="##local"/>  
231     </xsd:sequence>  
232     <xsd:anyAttribute processContents="lax"/>
```

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

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

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

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

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

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



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

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

260 To create a schema that contains vendor extensions, replace the <xsd:any ...  
261 namespace="##other" /> declaration with a content group reference to a group  
262 defined in the vendor namespace; e.g., <xsd:group  
263 ref="vendor:VendorExtension">. In the schema file defining elements for  
264 the vendor namespace, define a content group using a declaration of the following  
265 form:

```
266 <xsd:group name="VendorExtension">  
267   <xsd:sequence>  
268     <!--  
269       Definitions or references to vendor elements  
270       go here. Each SHALL specify minOccurs="0".  
271     -->  
272     <xsd:any processContents="lax"  
273       minOccurs="0" maxOccurs="unbounded"  
274       namespace="##other" />  
275   </xsd:sequence>  
276 </xsd:group>
```

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

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

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

- 290 • Standard attributes may be added to any XSD type corresponding to a UML class in  
291 which <<extension point>> occurs. Standard attributes SHALL NOT be in  
292 any namespace, and SHALL NOT conflict with any existing standard attribute name.
- 293 • Standard elements may be added to any XSD type corresponding to a UML class in  
294 which <<extension point>> occurs. New elements are added using the  
295 following rules:
- 296 • Find the innermost extension element type.
- 297 • Replace the <xsd:any ... namespace="##local"/> declaration with (a)  
298 new elements (which SHALL NOT be in any namespace); followed by (b) a new  
299 extension element whose type is constructed as described before. In  
300 subsequent revisions of the standard schema, new standard elements will be added  
301 within this new extension element rather than within this one.

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

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

### 307 **3.2.2 Extensibility Design Rules – Interfaces**

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

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

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

### 315 **3.2.3 Extensibility Design Rules – Enumerations**

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

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

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

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

### 333 3.3 EPCglobal Base Schema

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

```
336 <xsd:schema targetNamespace="urn:epcglobal:xsd:1"  
337           xmlns:epcglobal="urn:epcglobal:xsd:1"  
338           xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
339           elementFormDefault="unqualified"  
340           attributeFormDefault="unqualified"  
341           version="1.0">  
342   <xsd:annotation>  
343     <xsd:documentation>  
344       <epcglobal:copyright>Copyright (C) 2004 Epcglobal Inc., All Rights  
345       Reserved.</epcglobal:copyright>  
346       <epcglobal:disclaimer>EPCglobal Inc., its members, officers, directors, employees,  
347       or agents shall not be liable for any injury, loss, damages, financial or otherwise,  
348       arising from, related to, or caused by the use of this document. The use of said  
349       document shall constitute your express consent to the foregoing  
350       exculpation.</epcglobal:disclaimer>  
351       <epcglobal:specification>EPCglobal common components Version  
352       1.0</epcglobal:specification>  
353     </xsd:documentation>  
354   </xsd:annotation>  
355   <xsd:complexType name="Document" abstract="true">  
356     <xsd:annotation>  
357       <xsd:documentation xml:lang="en">  
358         EPCglobal document properties for all messages.  
359       </xsd:documentation>  
360     </xsd:annotation>  
361     <xsd:attribute name="schemaVersion" type="xsd:decimal" use="required">  
362       <xsd:annotation>  
363         <xsd:documentation xml:lang="en">  
364           The version of the schema corresponding to which the instance conforms.  
365         </xsd:documentation>  
366       </xsd:annotation>  
367     </xsd:attribute>  
368     <xsd:attribute name="creationDate" type="xsd:dateTime" use="required">  
369       <xsd:annotation>  
370         <xsd:documentation xml:lang="en">  
371           The date the message was created. Used for auditing and logging.  
372         </xsd:documentation>  
373       </xsd:annotation>  
374     </xsd:attribute>  
375   </xsd:complexType>  
376   <xsd:complexType name="EPC">  
377     <xsd:annotation>  
378       <xsd:documentation xml:lang="en">  
379         EPC represents the Electronic Product Code.  
380       </xsd:documentation>  
381     </xsd:annotation>  
382     <xsd:simpleContent>  
383       <xsd:extension base="xsd:string"/>  
384     </xsd:simpleContent>  
385   </xsd:complexType>  
386 </xsd:schema>
```

### 387 3.4 Schema for Datatypes Common to the Reading API and 388 Writing API

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

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

```
399 <?xml version="1.0" encoding="UTF-8"?>
400 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
401           xmlns:ale="urn:epcglobal:ale:xsd:1"
402           targetNamespace="urn:epcglobal:ale:xsd:1" elementFormDefault="unqualified"
403           attributeFormDefault="unqualified" version="1.0">
404   <!-- COMMON ELEMENTS -->
405
406   <!-- COMMON TYPES -->
407
408   <xsd:complexType name="ECFieldSpec">
409     <xsd:sequence>
410       <xsd:element name="fieldname" type="xsd:string"/>
411       <xsd:element name="datatype" type="xsd:string" minOccurs="0"/>
412       <xsd:element name="format" type="xsd:string" minOccurs="0"/>
413       <xsd:element name="extension" type="ale:ECFieldSpecExtension" minOccurs="0"/>
414       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
415             namespace="##other"/>
416     </xsd:sequence>
417     <xsd:anyAttribute processContents="lax"/>
418   </xsd:complexType>
419
420   <xsd:complexType name="ECFieldSpecExtension">
421     <xsd:sequence>
422       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
423             namespace="##local"/>
424     </xsd:sequence>
425     <xsd:anyAttribute processContents="lax"/>
426   </xsd:complexType>
427
428   <xsd:complexType name="ECFilterListMember">
429     <xsd:sequence>
430       <xsd:element name="includeExclude" type="ale:ECIncludeExclude"/>
431       <xsd:element name="fieldspec" type="ale:ECFieldSpec"/>
432       <xsd:element name="patList">
433         <xsd:complexType>
434           <xsd:sequence>
435             <xsd:element name="pat" type="xsd:string" maxOccurs="unbounded"/>
436           </xsd:sequence>
437         </xsd:complexType>
438       </xsd:element>
439       <xsd:element name="extension" type="ale:ECFilterListMemberExtension"
440             minOccurs="0"/>
441       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
442             namespace="##other"/>
443     </xsd:sequence>
444     <xsd:anyAttribute processContents="lax"/>
445   </xsd:complexType>
446
447   <xsd:complexType name="ECFilterListMemberExtension">
448     <xsd:sequence>
```

```

449     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
450         namespace="##local" />
451   </xsd:sequence>
452   <xsd:anyAttribute processContents="lax" />
453 </xsd:complexType>
454
455 <!-- The ECIncludeExclude type is an enumerated type.
456       The following strings are legal values for this type:
457         INCLUDE
458         EXCLUDE
459       -->
460 <xsd:simpleType name="ECIncludeExclude">
461   <xsd:restriction base="xsd:string" />
462 </xsd:simpleType>
463
464 <xsd:complexType name="ECReaderStat">
465   <xsd:sequence>
466     <xsd:element name="readerName" type="xsd:string" />
467     <xsd:element name="sightings" minOccurs="0">
468       <xsd:complexType>
469         <xsd:sequence>
470           <xsd:element name="sighting" type="ale:ECSightingStat" minOccurs="0"
471             maxOccurs="unbounded" />
472         </xsd:sequence>
473       </xsd:complexType>
474     </xsd:element>
475   </xsd:sequence>
476 </xsd:complexType>
477
478 <xsd:complexType name="ECSightingStat" />
479
480 <xsd:complexType name="ECTime">
481   <xsd:simpleContent>
482     <xsd:extension base="xsd:long">
483       <xsd:attribute name="unit" type="ale:ECTimeUnit" use="required" />
484     </xsd:extension>
485   </xsd:simpleContent>
486 </xsd:complexType>
487
488 <!-- The ECTimeUnit type is an enumerated type.
489       The following strings are legal values for this type:
490         MS
491       An implementation may also recognize additional strings as extensions.
492       -->
493 <xsd:simpleType name="ECTimeUnit">
494   <xsd:restriction base="xsd:string" />
495 </xsd:simpleType>
496
497 <xsd:simpleType name="ECTrigger">
498   <xsd:restriction base="xsd:string" />
499 </xsd:simpleType>
500 </xsd:schema>

```

### 501 3.5 ALE Reading API Schema

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

```

504 <?xml version="1.0" encoding="UTF-8"?>
505 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
506           xmlns:ale="urn:epcglobal:ale:xsd:1"
507           targetNamespace="urn:epcglobal:ale:xsd:1"
508           xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
509           attributeFormDefault="unqualified" version="1.0">
510   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd" />
511   <xsd:include schemaLocation="EPCglobal-ale-1_1-common.xsd" />
512   <!-- ALE ELEMENTS -->
513   <xsd:element name="ECSpec" type="ale:ECSpec" />

```

```

514 <xsd:element name="ECReports" type="ale:ECReports"/>
515
516 <!-- ALE TYPES -->
517
518 <xsd:complexType name="ECBoundarySpec">
519   <xsd:sequence>
520     <xsd:element name="startTrigger" type="ale:ECTrigger" minOccurs="0"/>
521     <xsd:element name="repeatPeriod" type="ale:ECTime" minOccurs="0"/>
522     <xsd:element name="stopTrigger" type="ale:ECTrigger" minOccurs="0"/>
523     <xsd:element name="duration" type="ale:ECTime" minOccurs="0"/>
524     <xsd:element name="stableSetInterval" type="ale:ECTime" minOccurs="0"/>
525     <xsd:element name="extension" type="ale:ECBoundarySpecExtension" minOccurs="0"/>
526     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
527       namespace="##other"/>
528   </xsd:sequence>
529   <xsd:anyAttribute processContents="lax"/>
530 </xsd:complexType>
531
532 <xsd:complexType name="ECBoundarySpecExtension">
533   <xsd:sequence>
534     <xsd:element name="startTriggerList" minOccurs="0">
535       <xsd:complexType>
536         <xsd:sequence>
537           <xsd:element name="startTrigger" type="ale:ECTrigger" minOccurs="0"
538             maxOccurs="unbounded"/>
539         </xsd:sequence>
540       </xsd:complexType>
541     </xsd:element>
542     <xsd:element name="stopTriggerList" minOccurs="0">
543       <xsd:complexType>
544         <xsd:sequence>
545           <xsd:element name="stopTrigger" type="ale:ECTrigger" minOccurs="0"
546             maxOccurs="unbounded"/>
547         </xsd:sequence>
548       </xsd:complexType>
549     </xsd:element>
550     <xsd:element name="whenDataAvailable" type="xsd:boolean" minOccurs="0"/>
551     <xsd:element name="extension" type="ale:ECBoundarySpecExtension2" minOccurs="0"/>
552   </xsd:sequence>
553   <xsd:anyAttribute processContents="lax"/>
554 </xsd:complexType>
555
556 <xsd:complexType name="ECBoundarySpecExtension2">
557   <xsd:sequence>
558     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
559       namespace="##local"/>
560   </xsd:sequence>
561   <xsd:anyAttribute processContents="lax"/>
562 </xsd:complexType>
563
564 <xsd:complexType name="ECFilterSpec">
565   <xsd:sequence>
566     <xsd:element name="includePatterns" minOccurs="0">
567       <xsd:complexType>
568         <xsd:sequence>
569           <xsd:element name="includePattern" type="xsd:string" minOccurs="0"
570             maxOccurs="unbounded"/>
571         </xsd:sequence>
572       </xsd:complexType>
573     </xsd:element>
574     <xsd:element name="excludePatterns" minOccurs="0">
575       <xsd:complexType>
576         <xsd:sequence>
577           <xsd:element name="excludePattern" type="xsd:string" minOccurs="0"
578             maxOccurs="unbounded"/>
579         </xsd:sequence>
580       </xsd:complexType>
581     </xsd:element>
582     <xsd:element name="extension" type="ale:ECFilterSpecExtension" minOccurs="0"/>
583     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"

```



```

584         namespace="##other" />
585     </xsd:sequence>
586 <xsd:anyAttribute processContents="lax" />
587 </xsd:complexType>
588
589 <xsd:complexType name="ECFilterSpecExtension">
590     <xsd:sequence>
591         <xsd:element name="filterList" minOccurs="0">
592             <xsd:complexType>
593                 <xsd:sequence>
594                     <xsd:element name="filter" type="ale:ECFilterListMember" minOccurs="0"
595                         maxOccurs="unbounded" />
596                 </xsd:sequence>
597             </xsd:complexType>
598         </xsd:element>
599         <xsd:element name="extension" type="ale:ECFilterSpecExtension2" minOccurs="0" />
600     </xsd:sequence>
601     <xsd:anyAttribute processContents="lax" />
602 </xsd:complexType>
603
604 <xsd:complexType name="ECFilterSpecExtension2">
605     <xsd:sequence>
606         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
607             namespace="##local" />
608     </xsd:sequence>
609     <xsd:anyAttribute processContents="lax" />
610 </xsd:complexType>
611
612 <xsd:complexType name="ECGroupSpec">
613     <xsd:sequence>
614         <xsd:element name="pattern" type="xsd:string" minOccurs="0" maxOccurs="unbounded" />
615         <xsd:element name="extension" type="ale:ECGroupSpecExtension" minOccurs="0" />
616         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
617             namespace="##other" />
618     </xsd:sequence>
619     <xsd:anyAttribute processContents="lax" />
620 </xsd:complexType>
621
622 <xsd:complexType name="ECGroupSpecExtension">
623     <xsd:sequence>
624         <xsd:element name="fieldspec" type="ale:ECFieldSpec" minOccurs="0" />
625         <xsd:element name="extension" type="ale:ECGroupSpecExtension2" minOccurs="0" />
626     </xsd:sequence>
627     <xsd:anyAttribute processContents="lax" />
628 </xsd:complexType>
629
630 <xsd:complexType name="ECGroupSpecExtension2">
631     <xsd:sequence>
632         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
633             namespace="##local" />
634     </xsd:sequence>
635     <xsd:anyAttribute processContents="lax" />
636 </xsd:complexType>
637
638 <!-- The ECInitiationCondition type is an enumerated type.
639     The following strings are legal values for this type:
640         TRIGGER
641         REPEAT_PERIOD
642         REQUESTED
643         UNDEFINE
644     An implementation may also recognize additional strings as extensions.
645     -->
646 <xsd:simpleType name="ECInitiationCondition">
647     <xsd:restriction base="xsd:string" />
648 </xsd:simpleType>
649
650 <xsd:complexType name="ECReport">
651     <xsd:sequence>
652         <xsd:element name="group" type="ale:ECReportGroup" minOccurs="0"
653             maxOccurs="unbounded" />

```

```

654     <xsd:element name="extension" type="ale:ECReportExtension" minOccurs="0"/>
655     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
656         namespace="##other"/>
657 </xsd:sequence>
658 <xsd:attribute name="reportName" type="xsd:string" use="required"/>
659 <xsd:anyAttribute processContents="lax"/>
660 </xsd:complexType>
661
662 <xsd:complexType name="ECReportExtension">
663     <xsd:sequence>
664         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
665             namespace="##local"/>
666     </xsd:sequence>
667     <xsd:anyAttribute processContents="lax"/>
668 </xsd:complexType>
669
670 <xsd:complexType name="ECReportGroup">
671     <xsd:sequence>
672         <xsd:element name="groupList" type="ale:ECReportGroupList" minOccurs="0"/>
673         <xsd:element name="groupCount" type="ale:ECReportGroupCount" minOccurs="0"/>
674         <xsd:element name="extension" type="ale:ECReportGroupExtension" minOccurs="0"/>
675         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
676             namespace="##other"/>
677     </xsd:sequence>
678     <!-- The groupName attribute SHALL be omitted to indicate the default group. -->
679     <xsd:attribute name="groupName" type="xsd:string" use="optional"/>
680     <xsd:anyAttribute processContents="lax"/>
681 </xsd:complexType>
682
683 <xsd:complexType name="ECReportGroupExtension">
684     <xsd:sequence>
685         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
686             namespace="##local"/>
687     </xsd:sequence>
688     <xsd:anyAttribute processContents="lax"/>
689 </xsd:complexType>
690
691 <xsd:complexType name="ECReportGroupCount">
692     <xsd:sequence>
693         <xsd:element name="count" type="xsd:int"/>
694         <xsd:element name="extension" type="ale:ECReportGroupCountExtension"
695             minOccurs="0"/>
696         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
697             namespace="##other"/>
698     </xsd:sequence>
699     <xsd:anyAttribute processContents="lax"/>
700 </xsd:complexType>
701
702 <xsd:complexType name="ECReportGroupCountExtension">
703     <xsd:sequence>
704         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
705             namespace="##local"/>
706     </xsd:sequence>
707     <xsd:anyAttribute processContents="lax"/>
708 </xsd:complexType>
709
710 <xsd:complexType name="ECReportGroupList">
711     <xsd:sequence>
712         <xsd:element name="member" type="ale:ECReportGroupListMember" minOccurs="0"
713             maxOccurs="unbounded"/>
714         <xsd:element name="extension" type="ale:ECReportGroupListExtension" minOccurs="0"/>
715         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
716             namespace="##other"/>
717     </xsd:sequence>
718     <xsd:anyAttribute processContents="lax"/>
719 </xsd:complexType>
720
721 <xsd:complexType name="ECReportGroupListExtension">
722     <xsd:sequence>
723         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"

```



```

724         namespace="##local"/>
725     </xsd:sequence>
726     <xsd:anyAttribute processContents="lax"/>
727 </xsd:complexType>
728
729 <xsd:complexType name="ECReportGroupListMember">
730     <xsd:sequence>
731         <!-- Each of the following four elements SHALL be omitted if null. -->
732         <xsd:element name="epc" type="epcglobal:EPC" minOccurs="0"/>
733         <xsd:element name="tag" type="epcglobal:EPC" minOccurs="0"/>
734         <xsd:element name="rawHex" type="epcglobal:EPC" minOccurs="0"/>
735         <xsd:element name="rawDecimal" type="epcglobal:EPC" minOccurs="0"/>
736         <xsd:element name="extension" type="ale:ECReportGroupListMemberExtension"
737             minOccurs="0"/>
738         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
739             namespace="##other"/>
740     </xsd:sequence>
741     <xsd:anyAttribute processContents="lax"/>
742 </xsd:complexType>
743
744 <xsd:complexType name="ECReportGroupListMemberExtension">
745     <xsd:sequence>
746         <xsd:element name="fieldList" minOccurs="0">
747             <xsd:complexType>
748                 <xsd:sequence>
749                     <xsd:element name="field" type="ale:ECReportMemberField" minOccurs="0"
750                         maxOccurs="unbounded"/>
751                 </xsd:sequence>
752             </xsd:complexType>
753         </xsd:element>
754         <xsd:element name="stats" minOccurs="0">
755             <xsd:complexType>
756                 <xsd:sequence>
757                     <xsd:element name="stat" type="ale:ECTagStat" minOccurs="0"
758                         maxOccurs="unbounded"/>
759                 </xsd:sequence>
760             </xsd:complexType>
761         </xsd:element>
762         <xsd:element name="extension" type="ale:ECReportGroupListMemberExtension2"
763             minOccurs="0"/>
764     </xsd:sequence>
765     <xsd:anyAttribute processContents="lax"/>
766 </xsd:complexType>
767
768 <xsd:complexType name="ECReportGroupListMemberExtension2">
769     <xsd:sequence>
770         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
771             namespace="##local"/>
772     </xsd:sequence>
773     <xsd:anyAttribute processContents="lax"/>
774 </xsd:complexType>
775
776 <xsd:complexType name="ECReportMemberField">
777     <xsd:sequence>
778         <xsd:element name="name" type="xsd:string"/>
779         <xsd:element name="value" type="xsd:string" minOccurs="0"/>
780         <xsd:element name="fieldspec" type="ale:ECFieldSpec" minOccurs="0"/>
781         <xsd:element name="extension" type="ale:ECReportMemberFieldExtension"
782             minOccurs="0"/>
783         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
784             namespace="##other"/>
785     </xsd:sequence>
786     <xsd:anyAttribute processContents="lax"/>
787 </xsd:complexType>
788
789 <xsd:complexType name="ECReportMemberFieldExtension">
790     <xsd:sequence>
791         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
792             namespace="##local"/>
793     </xsd:sequence>

```

```

794     <xsd:anyAttribute processContents="lax" />
795 </xsd:complexType>
796
797 <xsd:complexType name="ECReportOutputFieldSpec">
798   <xsd:sequence>
799     <xsd:element name="fieldspec" type="ale:ECFieldSpec" />
800     <xsd:element name="name" type="xsd:string" minOccurs="0" />
801     <xsd:element name="includeFieldSpecInReport" type="xsd:boolean" minOccurs="0" />
802     <xsd:element name="extension" type="ale:ECReportOutputFieldSpecExtension"
803       minOccurs="0" />
804     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
805       namespace="##other" />
806   </xsd:sequence>
807   <xsd:anyAttribute processContents="lax" />
808 </xsd:complexType>
809
810 <xsd:complexType name="ECReportOutputFieldSpecExtension">
811   <xsd:sequence>
812     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
813       namespace="##local" />
814   </xsd:sequence>
815   <xsd:anyAttribute processContents="lax" />
816 </xsd:complexType>
817
818 <xsd:complexType name="ECReportOutputSpec">
819   <xsd:sequence>
820     <xsd:element name="extension" type="ale:ECReportOutputSpecExtension"
821       minOccurs="0" />
822     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
823       namespace="##other" />
824   </xsd:sequence>
825   <xsd:attribute name="includeEPC" type="xsd:boolean" default="false" />
826   <xsd:attribute name="includeTag" type="xsd:boolean" default="false" />
827   <xsd:attribute name="includeRawHex" type="xsd:boolean" default="false" />
828   <xsd:attribute name="includeRawDecimal" type="xsd:boolean" default="false" />
829   <xsd:attribute name="includeCount" type="xsd:boolean" default="false" />
830   <xsd:anyAttribute processContents="lax" />
831 </xsd:complexType>
832
833 <xsd:complexType name="ECReportOutputSpecExtension">
834   <xsd:sequence>
835     <xsd:element name="fieldList" minOccurs="0">
836       <xsd:complexType>
837         <xsd:sequence>
838           <xsd:element name="field" type="ale:ECReportOutputFieldSpec" minOccurs="0"
839             maxOccurs="unbounded" />
840         </xsd:sequence>
841       </xsd:complexType>
842     </xsd:element>
843     <xsd:element name="extension" type="ale:ECReportOutputSpecExtension2"
844       minOccurs="0" />
845   </xsd:sequence>
846   <xsd:anyAttribute processContents="lax" />
847 </xsd:complexType>
848
849 <xsd:complexType name="ECReportOutputSpecExtension2">
850   <xsd:sequence>
851     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
852       namespace="##local" />
853   </xsd:sequence>
854   <xsd:anyAttribute processContents="lax" />
855 </xsd:complexType>
856
857 <xsd:complexType name="ECReports">
858   <xsd:complexContent>
859     <xsd:extension base="epcglobal:Document">
860       <xsd:sequence>
861         <xsd:element name="reports">
862           <xsd:complexType>
863             <xsd:sequence>

```

```

864         <xsd:element name="report" type="ale:ECReport" minOccurs="0"
865             maxOccurs="unbounded" />
866     </xsd:sequence>
867 </xsd:complexType>
868 </xsd:element>
869 <xsd:element name="extension" type="ale:ECReportsExtension" minOccurs="0" />
870 <xsd:element name="ECSpec" type="ale:ECSpec" minOccurs="0" />
871 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
872     namespace="##other" />
873 </xsd:sequence>
874 <xsd:attribute name="specName" type="xsd:string" use="required" />
875 <xsd:attribute name="date" type="xsd:dateTime" use="required" />
876 <xsd:attribute name="ALEID" type="xsd:string" use="required" />
877 <xsd:attribute name="totalMilliseconds" type="xsd:long" use="required" />
878 <xsd:attribute name="initiationCondition" type="ale:ECInitiationCondition"
879     use="optional" />
880 <xsd:attribute name="initiationTrigger" type="ale:ECTrigger" use="optional" />
881 <xsd:attribute name="terminationCondition" type="ale:ECTerminationCondition"
882     use="required" />
883 <xsd:attribute name="terminationTrigger" type="ale:ECTrigger" use="optional" />
884 <xsd:attribute name="schemaURL" type="xsd:string" use="optional" />
885 <xsd:anyAttribute processContents="lax" />
886 </xsd:extension>
887 </xsd:complexContent>
888 </xsd:complexType>
889
890 <xsd:complexType name="ECReportsExtension">
891 <xsd:sequence>
892 <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
893     namespace="##local" />
894 </xsd:sequence>
895 <xsd:anyAttribute processContents="lax" />
896 </xsd:complexType>
897
898 <!-- The ECReportSetEnum type is an enumerated type.
899     The following strings are legal values for this type:
900     CURRENT
901     ADDITIONS
902     DELETIONS
903     An implementation may also recognize additional strings as extensions.
904     -->
905 <xsd:simpleType name="ECReportSetEnum">
906 <xsd:restriction base="xsd:string" />
907 </xsd:simpleType>
908
909 <xsd:complexType name="ECReportSetSpec">
910 <xsd:attribute name="set" type="ale:ECReportSetEnum" use="required" />
911 </xsd:complexType>
912
913 <xsd:complexType name="ECReportSpec">
914 <xsd:sequence>
915 <xsd:element name="reportSet" type="ale:ECReportSetSpec" />
916 <xsd:element name="filterSpec" type="ale:ECFilterSpec" minOccurs="0" />
917 <xsd:element name="groupSpec" type="ale:ECGroupSpec" minOccurs="0" />
918 <xsd:element name="output" type="ale:ECReportOutputSpec" />
919 <xsd:element name="extension" type="ale:ECReportSpecExtension" minOccurs="0" />
920 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
921     namespace="##other" />
922 </xsd:sequence>
923 <xsd:attribute name="reportName" type="xsd:string" use="required" />
924 <xsd:attribute name="reportIfEmpty" type="xsd:boolean" default="false" />
925 <xsd:attribute name="reportOnlyOnChange" type="xsd:boolean" default="false" />
926 <xsd:anyAttribute processContents="lax" />
927 </xsd:complexType>
928
929 <xsd:complexType name="ECReportSpecExtension">
930 <xsd:sequence>
931 <xsd:element name="statProfileNames" minOccurs="0">
932 <xsd:complexType>
933 <xsd:sequence>

```

```

934         <xsd:element name="statProfileName" type="ale:ECStatProfileName"
935             minOccurs="0" maxOccurs="unbounded" />
936     </xsd:sequence>
937 </xsd:complexType>
938 </xsd:element>
939 <xsd:element name="extension" type="ale:ECReportSpecExtension2" minOccurs="0" />
940 </xsd:sequence>
941 <xsd:anyAttribute processContents="lax" />
942 </xsd:complexType>
943
944 <xsd:complexType name="ECReportSpecExtension2">
945     <xsd:sequence>
946         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
947             namespace="##local" />
948     </xsd:sequence>
949     <xsd:anyAttribute processContents="lax" />
950 </xsd:complexType>
951
952 <xsd:complexType name="ECSpec">
953     <xsd:complexContent>
954         <xsd:extension base="epcglobal:Document">
955             <xsd:sequence>
956                 <xsd:element name="logicalReaders">
957                     <xsd:complexType>
958                         <xsd:sequence>
959                             <xsd:element name="logicalReader" type="xsd:string"
960                                 maxOccurs="unbounded" />
961                         </xsd:sequence>
962                     </xsd:complexType>
963                 </xsd:element>
964                 <xsd:element name="boundarySpec" type="ale:ECBoundarySpec" />
965                 <xsd:element name="reportSpecs">
966                     <xsd:complexType>
967                         <xsd:sequence>
968                             <xsd:element name="reportSpec" type="ale:ECReportSpec"
969                                 maxOccurs="unbounded" />
970                         </xsd:sequence>
971                     </xsd:complexType>
972                 </xsd:element>
973                 <xsd:element name="extension" type="ale:ECSpecExtension" minOccurs="0" />
974                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
975                     namespace="##other" />
976             </xsd:sequence>
977             <xsd:attribute name="includeSpecInReports" type="xsd:boolean" default="false" />
978             <xsd:anyAttribute processContents="lax" />
979         </xsd:extension>
980     </xsd:complexContent>
981 </xsd:complexType>
982
983 <xsd:complexType name="ECSpecExtension">
984     <xsd:sequence>
985         <xsd:element name="primaryKeyFields" minOccurs="0">
986             <xsd:complexType>
987                 <xsd:sequence>
988                     <xsd:element name="primaryKeyField" type="xsd:string" minOccurs="0"
989                         maxOccurs="unbounded" />
990                 </xsd:sequence>
991             </xsd:complexType>
992         </xsd:element>
993         <xsd:element name="extension" type="ale:ECSpecExtension2" minOccurs="0" />
994     </xsd:sequence>
995     <xsd:anyAttribute processContents="lax" />
996 </xsd:complexType>
997
998 <xsd:complexType name="ECSpecExtension2">
999     <xsd:sequence>
1000         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1001             namespace="##local" />
1002     </xsd:sequence>
1003     <xsd:anyAttribute processContents="lax" />

```

```

1004 </xsd:complexType>
1005
1006 <!-- The ECStatProfileName type is an enumerated type.
1007 The following strings are legal values for this type:
1008 TagTimestamps
1009 An implementation may also recognize additional strings as extensions.
1010 -->
1011 <xsd:simpleType name="ECStatProfileName">
1012 <xsd:restriction base="xsd:string"/>
1013 </xsd:simpleType>
1014
1015 <xsd:complexType name="ECTagStat">
1016 <xsd:sequence>
1017 <xsd:element name="profile" type="ale:ECStatProfileName"/>
1018 <xsd:element name="statBlocks" minOccurs="0">
1019 <xsd:complexType>
1020 <xsd:sequence>
1021 <xsd:element name="statBlock" type="ale:ECReaderStat" minOccurs="0"
1022 maxOccurs="unbounded"/>
1023 </xsd:sequence>
1024 </xsd:complexType>
1025 </xsd:element>
1026 </xsd:sequence>
1027 </xsd:complexType>
1028
1029 <xsd:complexType name="ECTagTimestampStat">
1030 <xsd:complexContent>
1031 <xsd:extension base="ale:ECTagStat">
1032 <xsd:sequence>
1033 <xsd:element name="firstSightingTime" type="xsd:dateTime"/>
1034 <xsd:element name="lastSightingTime" type="xsd:dateTime"/>
1035 </xsd:sequence>
1036 </xsd:extension>
1037 </xsd:complexContent>
1038 </xsd:complexType>
1039
1040 <!-- The ECTerminationCondition type is an enumerated type.
1041 The following strings are legal values for this type:
1042 TRIGGER
1043 DURATION
1044 STABLE_SET
1045 DATA_AVAILABLE
1046 UNREQUEST
1047 UNDEFINE
1048 An implementation may also recognize additional strings as extensions.
1049 -->
1050 <xsd:simpleType name="ECTerminationCondition">
1051 <xsd:restriction base="xsd:string"/>
1052 </xsd:simpleType>
1053 </xsd:schema>

```

### 1054 3.5.1 ECSpec – Example 1 (non-normative)

1055 Here is an example ECSpec rendered into XML [XML1.0]:

```

1056 <?xml version="1.0" encoding="UTF-8"?>
1057
1058 <ale:ECSpec xmlns:ale="urn:epcglobal:ale:xsd:1"
1059 xmlns:epcglobal="urn:epcglobal:xsd:1"
1060 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1061 schemaVersion="1.0"
1062 creationDate="2003-08-06T10:54:06.444-05:00">
1063 <logicalReaders>
1064 <logicalReader>dock_1a</logicalReader>
1065 <logicalReader>dock_1b</logicalReader>
1066 </logicalReaders>
1067 <boundarySpec>
1068 <startTrigger>http://example.com/trigger1</startTrigger>
1069 <repeatPeriod unit="MS">20000</repeatPeriod>

```

```

1070         <stopTrigger>http://example.com/trigger2</stopTrigger>
1071         <duration unit="MS">3000</duration>
1072     </boundarySpec>
1073     <reportSpecs>
1074         <reportSpec reportName="report1">
1075             <reportSet set="CURRENT"/>
1076             <output includeTag="true"/>
1077         </reportSpec>
1078         <reportSpec reportName="report2">
1079             <reportSet set="ADDITIONS"/>
1080             <output includeCount="true"/>
1081         </reportSpec>
1082         <reportSpec reportName="report3">
1083             <reportSet set="DELETIONS"/>
1084             <groupSpec>
1085                 <pattern>urn:epc:pat:sgtin-96:X.X.X.*</pattern>
1086             </groupSpec>
1087             <output includeCount="true"/>
1088         </reportSpec>
1089     </reportSpecs>
1090 </ale:ECSpec>

```

### 1091 3.5.2 ECSpec – Example 2 (non-normative)

1092 Here is a second example ECSpec rendered into XML [XML1.0]:

```

1093 <?xml version="1.0" encoding="UTF-8"?>
1094
1095 <ale:ECSpec xmlns:ale="urn:epcglobal:ale:xsd:1"
1096             xmlns:epcglobal="urn:epcglobal:xsd:1"
1097             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1098             schemaVersion="1.0"
1099             creationDate="2007-05-14T10:10:10+02:00">
1100     <logicalReaders>
1101         <logicalReader>dock_1a</logicalReader>
1102         <logicalReader>dock_1b</logicalReader>
1103     </logicalReaders>
1104     <boundarySpec>
1105         <repeatPeriod unit="MS">20000</repeatPeriod>
1106         <duration unit="MS">3000</duration>
1107         <extension>
1108             <startTriggerList>
1109                 <startTrigger>http://example.com/trigger1</startTrigger>
1110             </startTriggerList>
1111             <stopTriggerList>
1112                 <stopTrigger>http://example.com/trigger2</stopTrigger>
1113             </stopTriggerList>
1114         </extension>
1115     </boundarySpec>
1116     <reportSpecs>
1117         <reportSpec reportName="report1">
1118             <reportSet set="CURRENT"/>
1119             <filterSpec>
1120                 <extension>
1121                     <filterList>
1122                         <filter>
1123                             <includeExclude>INCLUDE</includeExclude>
1124                             <fieldspec>
1125                                 <fieldname>epc</fieldname>
1126                             </fieldspec>
1127                             <patList>
1128                                 <pat>urn:epc:pat:gid-96:*.*. *</pat>
1129                             </patList>
1130                         </filter>
1131                     </filterList>
1132                 </extension>
1133             </filterSpec>
1134             <output includeTag="true">
1135                 <extension>

```

```

1136         <fieldList>
1137             <field>
1138                 <fieldspec>
1139                     <fieldname>LotCode</fieldname>
1140                 </fieldspec>
1141             </field>
1142         </fieldList>
1143     </extension>
1144 </output>
1145 </reportSpec>
1146 <reportSpec reportName="report2">
1147     <reportSet set="ADDITIONS"/>
1148     <output includeCount="true"/>
1149 </reportSpec>
1150 <reportSpec reportName="report3">
1151     <reportSet set="DELETIONS"/>
1152     <groupSpec>
1153         <pattern>urn:epc:pat:sgtin-96:X.X.X.*</pattern>
1154     </groupSpec>
1155     <output includeCount="true"/>
1156 </reportSpec>
1157 </reportSpecs>
1158 </ale:ECSpec>

```

### 1159 3.5.3 ECReports – Example 1 (non-normative)

1160 Here is an example ECReports rendered into XML [XML1.0]: This ECReports  
1161 instance corresponds to the ECSpec example in Section 3.5.1, assuming that the ECSpec  
1162 was defined using the name “embarcadère”.

```

1163 <?xml version="1.0" encoding="UTF-8"?>
1164 <ale:ECReports xmlns:ale="urn:epcglobal:ale:xsd:1"
1165     xmlns:epcglobal="urn:epcglobal:xsd:1"
1166     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1167     schemaVersion="1.0"
1168     creationDate="2003-08-06T10:54:06.444-05:00"
1169     specName="embarcad&#232;re"
1170     date="2003-08-06T10:54:06.444-05:00"
1171     ALEID="Edge34"
1172     totalMilliseconds="3034"
1173     initiationCondition="TRIGGER"
1174     initiationTrigger="http://example.com/trigger1"
1175     terminationCondition="DURATION">
1176     <reports>
1177         <report reportName="report1">
1178             <group>
1179                 <groupList>
1180                     <member><tag>urn:epc:tag:gid-96:10.50.1000</tag></member>
1181                     <member><tag>urn:epc:tag:gid-96:10.50.1001</tag></member>
1182                 </groupList>
1183             </group>
1184         </report>
1185         <report reportName="report2">
1186             <group><groupCount><count>6847</count></groupCount></group>
1187         </report>
1188         <report reportName="report3">
1189             <group name="urn:epc:pat:sgtin-96:3.0037000.012345.*">
1190                 <groupCount><count>2</count></groupCount>
1191             </group>
1192             <group name="urn:epc:pat:sgtin-96:3.0037000.055555.*">
1193                 <groupCount><count>3</count></groupCount>
1194             </group>
1195             <group>
1196                 <groupCount><count>6842</count></groupCount>
1197             </group>
1198         </report>
1199     </reports>
1200 </ale:ECReports>

```



### 1201 3.5.4 ECREports – Example 2 (non-normative)

1202 Here is an example ECREports rendered into XML [XML1.0]: This ECREports  
1203 instance corresponds to the ECSpec example in Section 3.5.2, assuming the ECSpec was  
1204 defined using the name “Spec1”.

```
1205 <?xml version="1.0" encoding="UTF-8"?>
1206
1207 <ale:ECREports xmlns:ale="urn:epcglobal:ale:xsd:1"
1208               xmlns:epcglobal="urn:epcglobal:xsd:1"
1209               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1210               schemaVersion="1.0"
1211               creationDate="2007-05-14T11:11:11+02:00"
1212               specName="Spec1"
1213               date="2007-05-14T11:11:11+02:00"
1214               ALEID="Edge34"
1215               totalMilliseconds="3034"
1216               initiationCondition="TRIGGER"
1217               initiationTrigger="http://example.com/trigger1"
1218               terminationCondition="TRIGGER"
1219               terminationTrigger="http://example.com/trigger2">
1220   <reports>
1221     <report reportName="report1">
1222       <group>
1223         <groupList>
1224           <member>
1225             <tag>urn:epc:tag:gid-96:10.50.1000</tag>
1226             <extension>
1227               <fieldList>
1228                 <field>
1229                   <name>LotCode</name>
1230                   <value>12345</value>
1231                 </field>
1232               </fieldList>
1233             </extension>
1234           </member>
1235           <member>
1236             <tag>urn:epc:tag:gid-96:10.50.1001</tag>
1237             <extension>
1238               <fieldList>
1239                 <field>
1240                   <name>LotCode</name>
1241                   <value>12345</value>
1242                 </field>
1243               </fieldList>
1244             </extension>
1245           </member>
1246         </groupList>
1247       </group>
1248     </report>
1249     <report reportName="report2">
1250       <group><groupCount><count>6847</count></groupCount></group>
1251     </report>
1252     <report reportName="report3">
1253       <group name="urn:epc:pat:sgtin-96:3.0037000.012345.*">
1254         <groupCount><count>2</count></groupCount>
1255       </group>
1256       <group name="urn:epc:pat:sgtin-96:3.0037000.055555.*">
1257         <groupCount><count>3</count></groupCount>
1258       </group>
1259       <group>
1260         <groupCount><count>6842</count></groupCount>
1261       </group>
1262     </report>
1263   </reports>
1264 </ale:ECREports>
```



### 1265 3.6 ALE Writing API Schema

1266 The following is an XML Schema (XSD) for the ALE Writing API, defining CCSpec,  
1267 CCParameterList, CCReports, EPCCacheSpec, EPCPatternList,  
1268 AssocTableSpec, AssocTableEntryList, and RNGSpec as top level elements.

```
1269 <?xml version="1.0" encoding="UTF-8"?>
1270 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1271           xmlns:ale="urn:epcglobal:ale:xsd:1"
1272           targetNamespace="urn:epcglobal:ale:xsd:1"
1273           xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
1274           attributeFormDefault="unqualified" version="1.0">
1275   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd"/>
1276   <xsd:include schemaLocation="EPCglobal-ale-1_1-common.xsd"/>
1277   <!-- ALECC ELEMENTS -->
1278   <xsd:element name="CCSpec" type="ale:CCSpec"/>
1279   <xsd:element name="CCReports" type="ale:CCReports"/>
1280   <xsd:element name="EPCCacheSpec" type="ale:EPCCacheSpec"/>
1281   <xsd:element name="EPCPatternList" type="ale:EPCPatternList"/>
1282   <xsd:element name="AssocTableSpec" type="ale:AssocTableSpec"/>
1283   <xsd:element name="AssocTableEntryList" type="ale:AssocTableEntryList"/>
1284   <xsd:element name="RNGSpec" type="ale:RNGSpec"/>
1285
1286   <!-- ALECC TYPES -->
1287
1288   <xsd:complexType name="AssocTableEntry">
1289     <xsd:sequence>
1290       <xsd:element name="key" type="xsd:string"/>
1291       <xsd:element name="value" type="xsd:string"/>
1292     </xsd:sequence>
1293   </xsd:complexType>
1294
1295   <xsd:complexType name="AssocTableEntryList">
1296     <xsd:sequence>
1297       <xsd:element name="entries" minOccurs="0">
1298         <xsd:complexType>
1299           <xsd:sequence>
1300             <xsd:element name="entry" type="ale:AssocTableEntry" minOccurs="0"
1301                       maxOccurs="unbounded"/>
1302           </xsd:sequence>
1303         </xsd:complexType>
1304       </xsd:element>
1305     </xsd:sequence>
1306   </xsd:complexType>
1307
1308   <xsd:complexType name="AssocTableSpec">
1309     <xsd:complexContent>
1310       <xsd:extension base="epcglobal:Document">
1311         <xsd:sequence>
1312           <xsd:element name="datatype" type="xsd:string"/>
1313           <xsd:element name="format" type="xsd:string"/>
1314           <xsd:element name="extension" type="ale:AssocTableSpecExtension"
1315                       minOccurs="0"/>
1316           <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1317                   namespace="##other"/>
1318         </xsd:sequence>
1319         <xsd:anyAttribute processContents="lax"/>
1320       </xsd:extension>
1321     </xsd:complexContent>
1322   </xsd:complexType>
1323
1324   <xsd:complexType name="AssocTableSpecExtension">
1325     <xsd:sequence>
1326       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1327               namespace="##local"/>
1328     </xsd:sequence>
1329     <xsd:anyAttribute processContents="lax"/>
1330   </xsd:complexType>
1331
```

```

1332 <xsd:complexType name="CCBoundarySpec">
1333   <xsd:sequence>
1334     <xsd:element name="startTriggerList" minOccurs="0">
1335       <xsd:complexType>
1336         <xsd:sequence>
1337           <xsd:element name="startTrigger" type="ale:ECTrigger" minOccurs="0"
1338             maxOccurs="unbounded"/>
1339         </xsd:sequence>
1340       </xsd:complexType>
1341     </xsd:element>
1342     <xsd:element name="repeatPeriod" type="ale:ECTime" minOccurs="0"/>
1343     <xsd:element name="stopTriggerList" minOccurs="0">
1344       <xsd:complexType>
1345         <xsd:sequence>
1346           <xsd:element name="stopTrigger" type="ale:ECTrigger" minOccurs="0"
1347             maxOccurs="unbounded"/>
1348         </xsd:sequence>
1349       </xsd:complexType>
1350     </xsd:element>
1351     <xsd:element name="duration" type="ale:ECTime" minOccurs="0"/>
1352     <xsd:element name="noNewTagsInterval" type="ale:ECTime" minOccurs="0"/>
1353     <xsd:element name="tagsProcessedCount" type="xsd:int" minOccurs="0"/>
1354     <xsd:element name="afterError" type="xsd:boolean" minOccurs="0"/>
1355     <xsd:element name="extension" type="ale:CCBoundarySpecExtension" minOccurs="0"/>
1356     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1357       namespace="##other"/>
1358   </xsd:sequence>
1359   <xsd:anyAttribute processContents="lax"/>
1360 </xsd:complexType>
1361
1362 <xsd:complexType name="CCBoundarySpecExtension">
1363   <xsd:sequence>
1364     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1365       namespace="##local"/>
1366   </xsd:sequence>
1367   <xsd:anyAttribute processContents="lax"/>
1368 </xsd:complexType>
1369
1370 <xsd:complexType name="CCCmdReport">
1371   <xsd:sequence>
1372     <xsd:element name="tagReports" minOccurs="0">
1373       <xsd:complexType>
1374         <xsd:sequence>
1375           <xsd:element name="tagReport" type="ale:CCTagReport" minOccurs="0"
1376             maxOccurs="unbounded"/>
1377         </xsd:sequence>
1378       </xsd:complexType>
1379     </xsd:element>
1380     <xsd:element name="extension" type="ale:CCCmdReportExtension" minOccurs="0"/>
1381     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1382       namespace="##other"/>
1383   </xsd:sequence>
1384   <xsd:attribute name="cmdSpecName" type="xsd:string" use="required"/>
1385   <xsd:anyAttribute processContents="lax"/>
1386 </xsd:complexType>
1387
1388 <xsd:complexType name="CCCmdReportExtension">
1389   <xsd:sequence>
1390     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1391       namespace="##local"/>
1392   </xsd:sequence>
1393   <xsd:anyAttribute processContents="lax"/>
1394 </xsd:complexType>
1395
1396 <xsd:complexType name="CCCmdSpec">
1397   <xsd:sequence>
1398     <xsd:element name="filterSpec" type="ale:CCFilterSpec" minOccurs="0"/>
1399     <xsd:element name="opSpecs" minOccurs="0">
1400       <xsd:complexType>
1401         <xsd:sequence>

```

```

1402         <xsd:element name="opSpec" type="ale:CCOpSpec" minOccurs="0"
1403             maxOccurs="unbounded" />
1404     </xsd:sequence>
1405 </xsd:complexType>
1406 </xsd:element>
1407 <xsd:element name="statProfileNames" minOccurs="0">
1408     <xsd:complexType>
1409         <xsd:sequence>
1410             <xsd:element name="statProfileName" type="ale:CCStatProfileName"
1411                 minOccurs="0" maxOccurs="unbounded" />
1412         </xsd:sequence>
1413     </xsd:complexType>
1414 </xsd:element>
1415 <xsd:element name="extension" type="ale:CCCmdSpecExtension" minOccurs="0" />
1416 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1417     namespace="##other" />
1418 </xsd:sequence>
1419 <xsd:attribute name="name" type="xsd:string" use="required" />
1420 <xsd:attribute name="reportIfEmpty" type="xsd:boolean" default="false" />
1421 <xsd:anyAttribute processContents="lax" />
1422 </xsd:complexType>
1423
1424 <xsd:complexType name="CCCmdSpecExtension">
1425     <xsd:sequence>
1426         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1427             namespace="##local" />
1428     </xsd:sequence>
1429     <xsd:anyAttribute processContents="lax" />
1430 </xsd:complexType>
1431
1432 <xsd:complexType name="CCFilterSpec">
1433     <xsd:sequence>
1434         <xsd:element name="filterList" minOccurs="0">
1435             <xsd:complexType>
1436                 <xsd:sequence>
1437                     <xsd:element name="filter" type="ale:ECFilterListMember" minOccurs="0"
1438                         maxOccurs="unbounded" />
1439                 </xsd:sequence>
1440             </xsd:complexType>
1441         </xsd:element>
1442         <xsd:element name="extension" type="ale:CCFilterSpecExtension" minOccurs="0" />
1443         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1444             namespace="##other" />
1445     </xsd:sequence>
1446     <xsd:anyAttribute processContents="lax" />
1447 </xsd:complexType>
1448
1449 <xsd:complexType name="CCFilterSpecExtension">
1450     <xsd:sequence>
1451         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1452             namespace="##local" />
1453     </xsd:sequence>
1454     <xsd:anyAttribute processContents="lax" />
1455 </xsd:complexType>
1456
1457 <!-- The CCInitiationCondition type is an enumerated type.
1458     The following strings are legal values for this type:
1459         TRIGGER
1460         REPEAT_PERIOD
1461         REQUESTED
1462         UNDEFINE
1463     An implementation may also recognize additional strings as extensions.
1464     -->
1465 <xsd:simpleType name="CCInitiationCondition">
1466     <xsd:restriction base="xsd:string" />
1467 </xsd:simpleType>
1468
1469 <!-- The CCLockOperation type is an enumerated type.
1470     The following strings are legal values for this type:
1471         UNLOCK

```

```

1472         PERMAUNLOCK
1473         LOCK
1474         PERMALOCK
1475     An implementation may also recognize additional strings as extensions.
1476     -->
1477 <xsd:simpleType name="CCLockOperation">
1478     <xsd:restriction base="xsd:string"/>
1479 </xsd:simpleType>
1480
1481 <xsd:complexType name="CCOpDataSpec">
1482     <xsd:sequence>
1483         <xsd:element name="specType" type="ale:CCOpDataSpecType"/>
1484         <xsd:element name="data" type="xsd:string"/>
1485         <xsd:element name="extension" type="ale:CCOpDataSpecExtension" minOccurs="0"/>
1486         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1487             namespace="##other"/>
1488     </xsd:sequence>
1489     <xsd:anyAttribute processContents="lax"/>
1490 </xsd:complexType>
1491
1492 <xsd:complexType name="CCOpDataSpecExtension">
1493     <xsd:sequence>
1494         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1495             namespace="##local"/>
1496     </xsd:sequence>
1497     <xsd:anyAttribute processContents="lax"/>
1498 </xsd:complexType>
1499
1500 <!-- The CCOpDataSpecType type is an enumerated type.
1501     The following strings are legal values for this type:
1502         LITERAL
1503         PARAMETER
1504         CACHE
1505         ASSOCIATION
1506         RANDOM
1507     An implementation may also recognize additional strings as extensions.
1508     -->
1509 <xsd:simpleType name="CCOpDataSpecType">
1510     <xsd:restriction base="xsd:string"/>
1511 </xsd:simpleType>
1512
1513 <xsd:complexType name="CCOpReport">
1514     <xsd:sequence>
1515         <xsd:element name="data" type="xsd:string" minOccurs="0"/>
1516         <xsd:element name="opStatus" type="ale:CCStatus"/>
1517         <xsd:element name="opName" type="xsd:string" minOccurs="0"/>
1518         <xsd:element name="extension" type="ale:CCOpReportExtension" minOccurs="0"/>
1519         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1520             namespace="##other"/>
1521     </xsd:sequence>
1522     <xsd:anyAttribute processContents="lax"/>
1523 </xsd:complexType>
1524
1525 <xsd:complexType name="CCOpReportExtension">
1526     <xsd:sequence>
1527         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1528             namespace="##local"/>
1529     </xsd:sequence>
1530     <xsd:anyAttribute processContents="lax"/>
1531 </xsd:complexType>
1532
1533 <xsd:complexType name="CCOpSpec">
1534     <xsd:sequence>
1535         <xsd:element name="opType" type="ale:CCOpType"/>
1536         <xsd:element name="fieldspec" type="ale:ECFieldSpec" minOccurs="0"/>
1537         <xsd:element name="dataSpec" type="ale:CCOpDataSpec" minOccurs="0"/>
1538         <xsd:element name="opName" type="xsd:string" minOccurs="0"/>
1539         <xsd:element name="extension" type="ale:CCOpSpecExtension" minOccurs="0"/>
1540         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1541             namespace="##other"/>

```

```

1542     </xsd:sequence>
1543     <xsd:anyAttribute processContents="lax" />
1544 </xsd:complexType>
1545
1546 <xsd:complexType name="CCOpSpecExtension">
1547   <xsd:sequence>
1548     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1549       namespace="##local" />
1550   </xsd:sequence>
1551   <xsd:anyAttribute processContents="lax" />
1552 </xsd:complexType>
1553
1554 <!-- The CCOpType type is an enumerated type.
1555     The following strings are legal values for this type:
1556     READ
1557     CHECK
1558     INITIALIZE
1559     ADD
1560     WRITE
1561     DELETE
1562     PASSWORD
1563     KILL
1564     LOCK
1565     An implementation may also recognize additional strings as extensions.
1566     -->
1567 <xsd:simpleType name="CCOpType">
1568   <xsd:restriction base="xsd:string" />
1569 </xsd:simpleType>
1570
1571 <xsd:complexType name="CCParameterList">
1572   <xsd:sequence>
1573     <xsd:element name="entries" minOccurs="0">
1574       <xsd:complexType>
1575         <xsd:sequence>
1576           <xsd:element name="entry" type="ale:CCParameterListEntry" minOccurs="0"
1577             maxOccurs="unbounded" />
1578         </xsd:sequence>
1579       </xsd:complexType>
1580     </xsd:element>
1581   </xsd:sequence>
1582 </xsd:complexType>
1583
1584 <xsd:complexType name="CCParameterListEntry">
1585   <xsd:sequence>
1586     <xsd:element name="name" type="xsd:string" />
1587     <xsd:element name="value" type="xsd:string" />
1588   </xsd:sequence>
1589 </xsd:complexType>
1590
1591 <xsd:complexType name="CCReports">
1592   <xsd:complexContent>
1593     <xsd:extension base="epcglobal:Document">
1594       <xsd:sequence>
1595         <xsd:element name="CCSpec" type="ale:CCSpec" minOccurs="0" />
1596         <xsd:element name="cmdReports" minOccurs="0">
1597           <xsd:complexType>
1598             <xsd:sequence>
1599               <xsd:element name="cmdReport" type="ale:CCCmdReport" minOccurs="0"
1600                 maxOccurs="unbounded" />
1601             </xsd:sequence>
1602           </xsd:complexType>
1603         </xsd:element>
1604         <xsd:element name="extension" type="ale:CCReportsExtension" minOccurs="0" />
1605         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1606           namespace="##other" />
1607       </xsd:sequence>
1608       <xsd:attribute name="specName" type="xsd:string" use="required" />
1609       <xsd:attribute name="date" type="xsd:dateTime" use="required" />
1610       <xsd:attribute name="ALEID" type="xsd:string" use="required" />
1611       <xsd:attribute name="totalMilliseconds" type="xsd:long" use="required" />

```

```

1612     <xsd:attribute name="initiationCondition" type="ale:CCInitiationCondition"
1613         use="required"/>
1614     <xsd:attribute name="initiationTrigger" type="ale:ECTrigger" use="optional"/>
1615     <xsd:attribute name="terminationCondition" type="ale:CCTerminationCondition"
1616         use="required"/>
1617     <xsd:attribute name="terminationTrigger" type="ale:ECTrigger" use="optional"/>
1618     <xsd:anyAttribute processContents="lax"/>
1619 </xsd:extension>
1620 </xsd:complexContent>
1621 </xsd:complexType>
1622
1623 <xsd:complexType name="CCReportsExtension">
1624     <xsd:sequence>
1625         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1626             namespace="##local"/>
1627     </xsd:sequence>
1628     <xsd:anyAttribute processContents="lax"/>
1629 </xsd:complexType>
1630
1631 <xsd:complexType name="CCSpec">
1632     <xsd:complexContent>
1633         <xsd:extension base="epcglobal:Document">
1634             <xsd:sequence>
1635                 <xsd:element name="logicalReaders">
1636                     <xsd:complexType>
1637                         <xsd:sequence>
1638                             <xsd:element name="logicalReader" type="xsd:string"
1639                                 maxOccurs="unbounded"/>
1640                         </xsd:sequence>
1641                     </xsd:complexType>
1642                 </xsd:element>
1643                 <xsd:element name="boundarySpec" type="ale:CCBoundarySpec"/>
1644                 <xsd:element name="cmdSpecs">
1645                     <xsd:complexType>
1646                         <xsd:sequence>
1647                             <xsd:element name="cmdSpec" type="ale:CCCmdSpec" maxOccurs="unbounded"/>
1648                         </xsd:sequence>
1649                     </xsd:complexType>
1650                 </xsd:element>
1651                 <xsd:element name="extension" type="ale:CCSpecExtension" minOccurs="0"/>
1652                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1653                     namespace="##other"/>
1654             </xsd:sequence>
1655             <xsd:attribute name="includeSpecInReports" type="xsd:boolean" default="false"/>
1656             <xsd:anyAttribute processContents="lax"/>
1657         </xsd:extension>
1658     </xsd:complexContent>
1659 </xsd:complexType>
1660
1661 <xsd:complexType name="CCSpecExtension">
1662     <xsd:sequence>
1663         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1664             namespace="##local"/>
1665     </xsd:sequence>
1666     <xsd:anyAttribute processContents="lax"/>
1667 </xsd:complexType>
1668
1669 <!-- The CCStatProfileName type is an enumerated type.
1670     The following strings are legal values for this type:
1671     An implementation may also recognize additional strings as extensions.
1672     -->
1673 <xsd:simpleType name="CCStatProfileName">
1674     <xsd:restriction base="xsd:string"/>
1675 </xsd:simpleType>
1676
1677 <!-- The CCStatus type is an enumerated type.
1678     The following strings are legal values for this type:
1679     SUCCESS
1680     MISC_ERROR_TOTAL
1681     MISC_ERROR_PARTIAL

```

```

1682     PERMISSION_ERROR
1683     PASSWORD_ERROR
1684     FIELD_NOT_FOUND_ERROR
1685     OP_NOT_POSSIBLE_ERROR
1686     OUT_OF_RANGE_ERROR
1687     FIELD_EXISTS_ERROR
1688     MEMORY_OVERFLOW_ERROR
1689     MEMORY_CHECK_ERROR
1690     ASSOCIATION_TABLE_VALUE_INVALID
1691     ASSOCIATION_TABLE_VALUE_MISSING
1692     EPC_CACHE_DEPLETED
1693     An implementation may also recognize additional strings as extensions.
1694     -->
1695     <xsd:simpleType name="CCStatus">
1696       <xsd:restriction base="xsd:string"/>
1697     </xsd:simpleType>
1698
1699     <xsd:complexType name="CCTagReport">
1700       <xsd:sequence>
1701         <xsd:element name="id" type="xsd:string" minOccurs="0"/>
1702         <xsd:element name="opReports" minOccurs="0">
1703           <xsd:complexType>
1704             <xsd:sequence>
1705               <xsd:element name="opReport" type="ale:CCOpReport" minOccurs="0"
1706                 maxOccurs="unbounded"/>
1707             </xsd:sequence>
1708           </xsd:complexType>
1709         </xsd:element>
1710         <xsd:element name="stats" minOccurs="0">
1711           <xsd:complexType>
1712             <xsd:sequence>
1713               <xsd:element name="stat" type="ale:CCTagStat" minOccurs="0"
1714                 maxOccurs="unbounded"/>
1715             </xsd:sequence>
1716           </xsd:complexType>
1717         </xsd:element>
1718         <xsd:element name="extension" type="ale:CCTagReportExtension" minOccurs="0"/>
1719         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1720           namespace="##other"/>
1721       </xsd:sequence>
1722       <xsd:anyAttribute processContents="lax"/>
1723     </xsd:complexType>
1724
1725     <xsd:complexType name="CCTagReportExtension">
1726       <xsd:sequence>
1727         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1728           namespace="##local"/>
1729       </xsd:sequence>
1730       <xsd:anyAttribute processContents="lax"/>
1731     </xsd:complexType>
1732
1733     <xsd:complexType name="CCTagStat">
1734       <xsd:sequence>
1735         <xsd:element name="profile" type="ale:CCStatProfileName"/>
1736         <xsd:element name="statBlocks" minOccurs="0">
1737           <xsd:complexType>
1738             <xsd:sequence>
1739               <xsd:element name="statBlock" type="ale:ECReaderStat" minOccurs="0"
1740                 maxOccurs="unbounded"/>
1741             </xsd:sequence>
1742           </xsd:complexType>
1743         </xsd:element>
1744       </xsd:sequence>
1745     </xsd:complexType>
1746
1747     <!-- The CCTerminationCondition type is an enumerated type.
1748         The following strings are legal values for this type:
1749         TRIGGER
1750         DURATION
1751         NO_NEW_TAGS

```



```

1752         COUNT
1753         ERROR
1754         UNREQUEST
1755         UNDEFINE
1756     An implementation may also recognize additional strings as extensions.
1757     -->
1758 <xsd:simpleType name="CCTerminationCondition">
1759   <xsd:restriction base="xsd:string"/>
1760 </xsd:simpleType>
1761
1762 <xsd:complexType name="EPCCacheSpec">
1763   <xsd:complexContent>
1764     <xsd:extension base="epcglobal:Document">
1765       <xsd:sequence>
1766         <xsd:element name="extension" type="ale:EPCCacheSpecExtension" minOccurs="0"/>
1767         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1768           namespace="##other"/>
1769       </xsd:sequence>
1770       <xsd:anyAttribute processContents="lax"/>
1771     </xsd:extension>
1772   </xsd:complexContent>
1773 </xsd:complexType>
1774
1775 <xsd:complexType name="EPCCacheSpecExtension">
1776   <xsd:sequence>
1777     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1778       namespace="##local"/>
1779   </xsd:sequence>
1780   <xsd:anyAttribute processContents="lax"/>
1781 </xsd:complexType>
1782
1783 <xsd:complexType name="EPCPatternList">
1784   <xsd:sequence>
1785     <xsd:element name="patterns" minOccurs="0">
1786       <xsd:complexType>
1787         <xsd:sequence>
1788           <xsd:element name="pattern" type="xsd:string" minOccurs="0"
1789             maxOccurs="unbounded"/>
1790         </xsd:sequence>
1791       </xsd:complexType>
1792     </xsd:element>
1793   </xsd:sequence>
1794 </xsd:complexType>
1795
1796 <xsd:complexType name="RNGSpec">
1797   <xsd:complexContent>
1798     <xsd:extension base="epcglobal:Document">
1799       <xsd:sequence>
1800         <xsd:element name="length" type="xsd:int"/>
1801         <xsd:element name="extension" type="ale:RNGSpecExtension" minOccurs="0"/>
1802         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1803           namespace="##other"/>
1804       </xsd:sequence>
1805       <xsd:anyAttribute processContents="lax"/>
1806     </xsd:extension>
1807   </xsd:complexContent>
1808 </xsd:complexType>
1809
1810 <xsd:complexType name="RNGSpecExtension">
1811   <xsd:sequence>
1812     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1813       namespace="##local"/>
1814   </xsd:sequence>
1815   <xsd:anyAttribute processContents="lax"/>
1816 </xsd:complexType>
1817 </xsd:schema>

```



### 1818 3.7 ALE Tag Memory API Schema

1819 The following is an XML Schema (XSD) for the ALE Tag Memory API, defining  
1820 TMSpec as a top level element.

```
1821 <?xml version="1.0" encoding="UTF-8"?>
1822 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1823           xmlns:ale="urn:epcglobal:ale:xsd:1"
1824           targetNamespace="urn:epcglobal:ale:xsd:1"
1825           xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
1826           attributeFormDefault="unqualified" version="1.0">
1827   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd"/>
1828   <!-- ALETM ELEMENTS -->
1829   <xsd:element name="TMSpec" type="ale:TMSpec"/>
1830
1831   <!-- ALETM TYPES -->
1832
1833   <xsd:complexType name="TMFixedFieldListSpec">
1834     <xsd:complexContent>
1835       <xsd:extension base="ale:TMSpec">
1836         <xsd:sequence>
1837           <xsd:element name="fixedFields" minOccurs="0">
1838             <xsd:complexType>
1839               <xsd:sequence>
1840                 <xsd:element name="fixedField" type="ale:TMFixedFieldSpec"
1841                           minOccurs="0" maxOccurs="unbounded"/>
1842               </xsd:sequence>
1843             </xsd:complexType>
1844           </xsd:element>
1845           <xsd:element name="extension" type="ale:TMFixedFieldListSpecExtension"
1846                     minOccurs="0"/>
1847           <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1848                 namespace="##other"/>
1849         </xsd:sequence>
1850       <xsd:anyAttribute processContents="lax"/>
1851     </xsd:extension>
1852   </xsd:complexContent>
1853 </xsd:complexType>
1854
1855   <xsd:complexType name="TMFixedFieldListSpecExtension">
1856     <xsd:sequence>
1857       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1858             namespace="##local"/>
1859     </xsd:sequence>
1860     <xsd:anyAttribute processContents="lax"/>
1861 </xsd:complexType>
1862
1863   <xsd:complexType name="TMFixedFieldSpec">
1864     <xsd:sequence>
1865       <xsd:element name="fieldname" type="xsd:string"/>
1866       <xsd:element name="bank" type="xsd:int"/>
1867       <xsd:element name="length" type="xsd:int"/>
1868       <xsd:element name="offset" type="xsd:int"/>
1869       <xsd:element name="defaultDatatype" type="xsd:string"/>
1870       <xsd:element name="defaultFormat" type="xsd:string"/>
1871       <xsd:element name="extension" type="ale:TMFixedFieldSpecExtension" minOccurs="0"/>
1872       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1873             namespace="##other"/>
1874     </xsd:sequence>
1875     <xsd:anyAttribute processContents="lax"/>
1876 </xsd:complexType>
1877
1878   <xsd:complexType name="TMFixedFieldSpecExtension">
1879     <xsd:sequence>
1880       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1881             namespace="##local"/>
1882     </xsd:sequence>
1883     <xsd:anyAttribute processContents="lax"/>
1884 </xsd:complexType>
```

```

1885
1886 <xsd:complexType name="TMSpec" abstract="true">
1887   <xsd:complexContent>
1888     <xsd:extension base="epcglobal:Document">
1889       <xsd:sequence>
1890         <xsd:element name="baseExtension" type="ale:TMSpecExtension" minOccurs="0"/>
1891       </xsd:sequence>
1892       <xsd:anyAttribute processContents="lax"/>
1893     </xsd:extension>
1894   </xsd:complexContent>
1895 </xsd:complexType>
1896
1897 <xsd:complexType name="TMSpecExtension">
1898   <xsd:sequence>
1899     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1900       namespace="##local"/>
1901   </xsd:sequence>
1902   <xsd:anyAttribute processContents="lax"/>
1903 </xsd:complexType>
1904
1905 <xsd:complexType name="TMVariableFieldListSpec">
1906   <xsd:complexContent>
1907     <xsd:extension base="ale:TMSpec">
1908       <xsd:sequence>
1909         <xsd:element name="variableFields" minOccurs="0">
1910           <xsd:complexType>
1911             <xsd:sequence>
1912               <xsd:element name="variableField" type="ale:TMVariableFieldSpec"
1913                 minOccurs="0" maxOccurs="unbounded"/>
1914             </xsd:sequence>
1915           </xsd:complexType>
1916         </xsd:element>
1917         <xsd:element name="extension" type="ale:TMVariableFieldListSpecExtension"
1918           minOccurs="0"/>
1919         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1920           namespace="##other"/>
1921       </xsd:sequence>
1922       <xsd:anyAttribute processContents="lax"/>
1923     </xsd:extension>
1924   </xsd:complexContent>
1925 </xsd:complexType>
1926
1927 <xsd:complexType name="TMVariableFieldListSpecExtension">
1928   <xsd:sequence>
1929     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1930       namespace="##local"/>
1931   </xsd:sequence>
1932   <xsd:anyAttribute processContents="lax"/>
1933 </xsd:complexType>
1934
1935 <xsd:complexType name="TMVariableFieldSpec">
1936   <xsd:sequence>
1937     <xsd:element name="fieldname" type="xsd:string"/>
1938     <xsd:element name="bank" type="xsd:int"/>
1939     <xsd:element name="oid" type="xsd:string"/>
1940     <xsd:element name="extension" type="ale:TMVariableFieldSpecExtension"
1941       minOccurs="0"/>
1942     <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
1943       namespace="##other"/>
1944   </xsd:sequence>
1945   <xsd:anyAttribute processContents="lax"/>
1946 </xsd:complexType>
1947
1948 <xsd:complexType name="TMVariableFieldSpecExtension">
1949   <xsd:sequence>
1950     <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
1951       namespace="##local"/>
1952   </xsd:sequence>
1953   <xsd:anyAttribute processContents="lax"/>
1954 </xsd:complexType>

```

1955 </xsd:schema>

### 1956 3.8 ALE Logical Reader API Schema

1957 The following is an XML Schema (XSD) for the ALE Logical Reader API, defining  
1958 LRSpec, and LRProperty as top level elements.

```
1959 <?xml version="1.0" encoding="UTF-8"?>
1960 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
1961           xmlns:ale="urn:epcglobal:ale:xsd:1"
1962           targetNamespace="urn:epcglobal:ale:xsd:1"
1963           xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
1964           attributeFormDefault="unqualified" version="1.0">
1965   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd"/>
1966   <!-- ALELR ELEMENTS -->
1967   <xsd:element name="LRSpec" type="ale:LRSpec"/>
1968   <xsd:element name="LRProperty" type="ale:LRProperty"/>
1969
1970   <!-- ALELR TYPES -->
1971
1972   <xsd:complexType name="LRProperty">
1973     <xsd:sequence>
1974       <xsd:element name="name" type="xsd:string"/>
1975       <xsd:element name="value" type="xsd:string" minOccurs="0"/>
1976     </xsd:sequence>
1977   </xsd:complexType>
1978
1979   <xsd:complexType name="LRSpec">
1980     <xsd:complexContent>
1981       <xsd:extension base="epcglobal:Document">
1982         <xsd:sequence>
1983           <xsd:element name="isComposite" type="xsd:boolean" minOccurs="0"/>
1984           <xsd:element name="readers" minOccurs="0">
1985             <xsd:complexType>
1986               <xsd:sequence>
1987                 <xsd:element name="reader" type="xsd:string" minOccurs="0"
1988                           maxOccurs="unbounded"/>
1989               </xsd:sequence>
1990             </xsd:complexType>
1991           </xsd:element>
1992           <xsd:element name="properties" minOccurs="0">
1993             <xsd:complexType>
1994               <xsd:sequence>
1995                 <xsd:element name="property" type="ale:LRProperty" minOccurs="0"
1996                           maxOccurs="unbounded"/>
1997               </xsd:sequence>
1998             </xsd:complexType>
1999           </xsd:element>
2000           <xsd:element name="extension" type="ale:LRSpecExtension" minOccurs="0"/>
2001           <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
2002                 namespace="##other"/>
2003         </xsd:sequence>
2004         <xsd:anyAttribute processContents="lax"/>
2005       </xsd:extension>
2006     </xsd:complexContent>
2007   </xsd:complexType>
2008
2009   <xsd:complexType name="LRSpecExtension">
2010     <xsd:sequence>
2011       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
2012             namespace="##local"/>
2013     </xsd:sequence>
2014     <xsd:anyAttribute processContents="lax"/>
2015   </xsd:complexType>
2016 </xsd:schema>
```

### 2017 3.9 ALE Access Control API Schema

2018 The following is an XML Schema (XSD) for the ALE Access Control API, defining  
2019 ACPermission, ACRole, and ACClientIdentity as top level elements.

```
2020 <?xml version="1.0" encoding="UTF-8"?>
2021 <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2022           xmlns:ale="urn:epcglobal:ale:xsd:1"
2023           targetNamespace="urn:epcglobal:ale:xsd:1"
2024           xmlns:epcglobal="urn:epcglobal:xsd:1" elementFormDefault="unqualified"
2025           attributeFormDefault="unqualified" version="1.0">
2026   <xsd:import namespace="urn:epcglobal:xsd:1" schemaLocation="EPCglobal.xsd"/>
2027   <!-- ALEAC ELEMENTS -->
2028   <xsd:element name="ACPermission" type="ale:ACPermission"/>
2029   <xsd:element name="ACRole" type="ale:ACRole"/>
2030   <xsd:element name="ACClientIdentity" type="ale:ACClientIdentity"/>
2031
2032   <!-- ALEAC TYPES -->
2033
2034   <!-- The ACClass type is an enumerated type.
2035         The following strings are legal values for this type:
2036         Method
2037         An implementation may also recognize additional strings as extensions.
2038         -->
2039   <xsd:simpleType name="ACClass">
2040     <xsd:restriction base="xsd:string"/>
2041   </xsd:simpleType>
2042
2043   <xsd:complexType name="ACClientCredential">
2044     <xsd:sequence>
2045       <xsd:element name="extension" type="ale:ACClientCredentialExtension"
2046                 minOccurs="0"/>
2047       <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
2048             namespace="##other"/>
2049     </xsd:sequence>
2050     <xsd:anyAttribute processContents="lax"/>
2051   </xsd:complexType>
2052
2053   <xsd:complexType name="ACClientCredentialExtension">
2054     <xsd:sequence>
2055       <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
2056             namespace="##local"/>
2057     </xsd:sequence>
2058     <xsd:anyAttribute processContents="lax"/>
2059   </xsd:complexType>
2060
2061   <xsd:complexType name="ACClientIdentity">
2062     <xsd:complexContent>
2063       <xsd:extension base="epcglobal:Document">
2064         <xsd:sequence>
2065           <xsd:element name="credentials" minOccurs="0">
2066             <xsd:complexType>
2067               <xsd:sequence>
2068                 <xsd:element name="credential" type="ale:ACClientCredential"
2069                       minOccurs="0" maxOccurs="unbounded"/>
2070             </xsd:sequence>
2071             </xsd:complexType>
2072           </xsd:element>
2073           <xsd:element name="roleNames" minOccurs="0">
2074             <xsd:complexType>
2075               <xsd:sequence>
2076                 <xsd:element name="roleName" type="xsd:string" minOccurs="0"
2077                       maxOccurs="unbounded"/>
2078             </xsd:sequence>
2079             </xsd:complexType>
2080           </xsd:element>
2081           <xsd:element name="extension" type="ale:ACClientIdentityExtension"
2082                 minOccurs="0"/>
2083         <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded">
```

```

2084         namespace="##other" />
2085     </xsd:sequence>
2086     <xsd:anyAttribute processContents="lax" />
2087 </xsd:extension>
2088 </xsd:complexContent>
2089 </xsd:complexType>
2090
2091 <xsd:complexType name="ACClientIdentityExtension">
2092     <xsd:sequence>
2093         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
2094             namespace="##local" />
2095     </xsd:sequence>
2096     <xsd:anyAttribute processContents="lax" />
2097 </xsd:complexType>
2098
2099 <xsd:complexType name="ACPermission">
2100     <xsd:complexContent>
2101         <xsd:extension base="epcglobal:Document">
2102             <xsd:sequence>
2103                 <xsd:element name="permissionClass" type="ale:ACClass" />
2104                 <xsd:element name="instances" minOccurs="0">
2105                     <xsd:complexType>
2106                         <xsd:sequence>
2107                             <xsd:element name="instance" type="xsd:string" minOccurs="0"
2108                                 maxOccurs="unbounded" />
2109                         </xsd:sequence>
2110                     </xsd:complexType>
2111                 </xsd:element>
2112                 <xsd:element name="extension" type="ale:ACPermissionExtension" 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="ACPermissionExtension">
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
2129 <xsd:complexType name="ACRole">
2130     <xsd:complexContent>
2131         <xsd:extension base="epcglobal:Document">
2132             <xsd:sequence>
2133                 <xsd:element name="permissionNames" minOccurs="0">
2134                     <xsd:complexType>
2135                         <xsd:sequence>
2136                             <xsd:element name="permissionName" type="xsd:string" minOccurs="0"
2137                                 maxOccurs="unbounded" />
2138                         </xsd:sequence>
2139                     </xsd:complexType>
2140                 </xsd:element>
2141                 <xsd:element name="extension" type="ale:ACRoleExtension" minOccurs="0" />
2142                 <xsd:any processContents="lax" minOccurs="0" maxOccurs="unbounded"
2143                     namespace="##other" />
2144             </xsd:sequence>
2145             <xsd:anyAttribute processContents="lax" />
2146         </xsd:extension>
2147     </xsd:complexContent>
2148 </xsd:complexType>
2149
2150 <xsd:complexType name="ACRoleExtension">
2151     <xsd:sequence>
2152         <xsd:any processContents="lax" minOccurs="1" maxOccurs="unbounded"
2153             namespace="##local" />

```

```

2154     </xsd:sequence>
2155     <xsd:anyAttribute processContents="lax" />
2156     </xsd:complexType>
2157 </xsd:schema>

```

## 2158 **4 SOAP Bindings**

2159 This section defines SOAP 1.1 [SOAP1.1] bindings for the five ALE APIs, using WSDL.  
 2160 All SOAP bindings are WS-I Basic Profile 1.0 [WSI] compliant for greatest  
 2161 interoperability.

### 2162 **4.1 Organization of the SOAP Bindings**

2163 Because an implementation may not implement all five ALE APIs, each API has a  
 2164 separate WSDL file. The request, response, and exception message types are defined in a  
 2165 separate XML namespace for each API. Complex data types referred to in requests and  
 2166 responses are all imported from the XML schemas defined in Section 2, which are all in a  
 2167 common XML namespace..

2168 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

2169

### 2170 **4.2 Link Level Security**

2171 ALE implementations that wish to provide a measure of security between themselves and  
 2172 their clients MAY implement an HTTPS (“HTTP Over TLS”) [RFC2818] transport for  
 2173 SOAP messages.

### 2174 **4.3 ALE Reading API SOAP Binding**

2175 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
 2176 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Reading  
 2177 API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0 [WSI].

```

2178 <?xml version="1.0" encoding="UTF-8"?>
2179 <!-- ALESERVICE DEFINITIONS -->
2180 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2181                 xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
2182                 xmlns:wSDLsoap="http://schemas.xmlsoap.org/wSDL/soap/"
2183                 xmlns:ale="urn:epcglobal:ale:xsd:1"
2184                 xmlns:impl="urn:epcglobal:ale:wSDL:1"
2185                 targetNamespace="urn:epcglobal:ale:wSDL:1">
2186   <!-- ALESERVICE TYPES -->
2187   <wSDL:types>

```

```

2188 <xsd:schema targetNamespace="urn:epcglobal:ale:wSDL:1">
2189   <xsd:import namespace="urn:epcglobal:ale:xsd:1"
2190     schemaLocation="EPCglobal-ale-1_1-ale.xsd"/>
2191   <!-- ALESERVICE MESSAGE WRAPPERS -->
2192
2193   <xsd:element name="Define" type="impl:Define"/>
2194   <xsd:complexType name="Define">
2195     <xsd:sequence>
2196       <xsd:element name="specName" type="xsd:string"/>
2197       <xsd:element name="spec" type="ale:ECSpec"/>
2198     </xsd:sequence>
2199   </xsd:complexType>
2200
2201   <xsd:element name="Undefine" type="impl:Undefine"/>
2202   <xsd:complexType name="Undefine">
2203     <xsd:sequence>
2204       <xsd:element name="specName" type="xsd:string"/>
2205     </xsd:sequence>
2206   </xsd:complexType>
2207
2208   <xsd:element name="GetECSpec" type="impl:GetECSpec"/>
2209   <xsd:complexType name="GetECSpec">
2210     <xsd:sequence>
2211       <xsd:element name="specName" type="xsd:string"/>
2212     </xsd:sequence>
2213   </xsd:complexType>
2214   <xsd:element name="GetECSpecResult" type="ale:ECSpec"/>
2215
2216   <xsd:element name="GetECSpecNames" type="impl:EmptyParms"/>
2217   <xsd:element name="GetECSpecNamesResult" type="impl:ArrayOfString"/>
2218
2219   <xsd:element name="Subscribe" type="impl:Subscribe"/>
2220   <xsd:complexType name="Subscribe">
2221     <xsd:sequence>
2222       <xsd:element name="specName" type="xsd:string"/>
2223       <xsd:element name="notificationURI" type="xsd:string"/>
2224     </xsd:sequence>
2225   </xsd:complexType>
2226
2227   <xsd:element name="Unsubscribe" type="impl:Unsubscribe"/>
2228   <xsd:complexType name="Unsubscribe">
2229     <xsd:sequence>
2230       <xsd:element name="specName" type="xsd:string"/>
2231       <xsd:element name="notificationURI" type="xsd:string"/>
2232     </xsd:sequence>
2233   </xsd:complexType>
2234
2235   <xsd:element name="Poll" type="impl:Poll"/>
2236   <xsd:complexType name="Poll">
2237     <xsd:sequence>
2238       <xsd:element name="specName" type="xsd:string"/>
2239     </xsd:sequence>
2240   </xsd:complexType>
2241   <xsd:element name="PollResult" type="ale:ECReports"/>
2242
2243   <xsd:element name="Immediate" type="impl:Immediate"/>
2244   <xsd:complexType name="Immediate">
2245     <xsd:sequence>
2246       <xsd:element name="spec" type="ale:ECSpec"/>
2247     </xsd:sequence>
2248   </xsd:complexType>
2249   <xsd:element name="ImmediateResult" type="ale:ECReports"/>
2250
2251   <xsd:element name="GetSubscribers" type="impl:GetSubscribers"/>
2252   <xsd:complexType name="GetSubscribers">
2253     <xsd:sequence>
2254       <xsd:element name="specName" type="xsd:string"/>
2255     </xsd:sequence>
2256   </xsd:complexType>
2257   <xsd:element name="GetSubscribersResult" type="impl:ArrayOfString"/>

```



```

2258
2259 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
2260 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
2261
2262 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
2263 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
2264
2265 <xsd:element name="ALEException" type="impl:ALEException"/>
2266 <xsd:complexType name="ALEException">
2267   <xsd:sequence>
2268     <xsd:element name="reason" type="xsd:string"/>
2269   </xsd:sequence>
2270 </xsd:complexType>
2271
2272 <xsd:element name="SecurityException" type="impl:SecurityException"/>
2273 <xsd:complexType name="SecurityException">
2274   <xsd:complexContent>
2275     <xsd:extension base="impl:ALEException"/>
2276   </xsd:complexContent>
2277 </xsd:complexType>
2278
2279 <xsd:element name="DuplicateNameException" type="impl:DuplicateNameException"/>
2280 <xsd:complexType name="DuplicateNameException">
2281   <xsd:complexContent>
2282     <xsd:extension base="impl:ALEException"/>
2283   </xsd:complexContent>
2284 </xsd:complexType>
2285
2286 <xsd:element name="ECSpecValidationException"
2287   type="impl:ECSpecValidationException"/>
2288 <xsd:complexType name="ECSpecValidationException">
2289   <xsd:complexContent>
2290     <xsd:extension base="impl:ALEException"/>
2291   </xsd:complexContent>
2292 </xsd:complexType>
2293
2294 <xsd:element name="InvalidURIException" type="impl:InvalidURIException"/>
2295 <xsd:complexType name="InvalidURIException">
2296   <xsd:complexContent>
2297     <xsd:extension base="impl:ALEException"/>
2298   </xsd:complexContent>
2299 </xsd:complexType>
2300
2301 <xsd:element name="NoSuchNameException" type="impl:NoSuchNameException"/>
2302 <xsd:complexType name="NoSuchNameException">
2303   <xsd:complexContent>
2304     <xsd:extension base="impl:ALEException"/>
2305   </xsd:complexContent>
2306 </xsd:complexType>
2307
2308 <xsd:element name="NoSuchSubscriberException"
2309   type="impl:NoSuchSubscriberException"/>
2310 <xsd:complexType name="NoSuchSubscriberException">
2311   <xsd:complexContent>
2312     <xsd:extension base="impl:ALEException"/>
2313   </xsd:complexContent>
2314 </xsd:complexType>
2315
2316 <xsd:element name="DuplicateSubscriptionException"
2317   type="impl:DuplicateSubscriptionException"/>
2318 <xsd:complexType name="DuplicateSubscriptionException">
2319   <xsd:complexContent>
2320     <xsd:extension base="impl:ALEException"/>
2321   </xsd:complexContent>
2322 </xsd:complexType>
2323
2324 <xsd:element name="ImplementationException" type="impl:ImplementationException"/>
2325 <xsd:complexType name="ImplementationException">
2326   <xsd:complexContent>
2327     <xsd:extension base="impl:ALEException">

```



```

2328         <xsd:sequence>
2329             <xsd:element name="severity" type="impl:ImplementationExceptionSeverity"/>
2330         </xsd:sequence>
2331     </xsd:extension>
2332 </xsd:complexContent>
2333 </xsd:complexType>
2334
2335     <xsd:complexType name="ArrayOfString">
2336         <xsd:sequence>
2337             <xsd:element name="string" type="xsd:string" minOccurs="0"
2338                 maxOccurs="unbounded"/>
2339         </xsd:sequence>
2340     </xsd:complexType>
2341
2342     <!-- The ImplementationExceptionSeverity type is an enumerated type.
2343         The following strings are legal values for this type:
2344         ERROR
2345         SEVERE
2346     -->
2347     <xsd:simpleType name="ImplementationExceptionSeverity">
2348         <xsd:restriction base="xsd:string"/>
2349     </xsd:simpleType>
2350
2351     <xsd:element name="VoidHolder" type="impl:VoidHolder"/>
2352     <xsd:complexType name="VoidHolder"/>
2353
2354     <xsd:complexType name="EmptyParms"/>
2355 </xsd:schema>
2356 </wsdl:types>
2357 <!-- ALESERVICE MESSAGES -->
2358
2359 <wsdl:message name="defineRequest">
2360     <wsdl:part name="parms" element="impl:Define"/>
2361 </wsdl:message>
2362 <wsdl:message name="defineResponse">
2363     <wsdl:part name="defineReturn" element="impl:VoidHolder"/>
2364 </wsdl:message>
2365
2366 <wsdl:message name="undefineRequest">
2367     <wsdl:part name="parms" element="impl:Undefine"/>
2368 </wsdl:message>
2369 <wsdl:message name="undefineResponse">
2370     <wsdl:part name="undefineReturn" element="impl:VoidHolder"/>
2371 </wsdl:message>
2372
2373 <wsdl:message name="getECSpecRequest">
2374     <wsdl:part name="parms" element="impl:GetECSpec"/>
2375 </wsdl:message>
2376 <wsdl:message name="getECSpecResponse">
2377     <wsdl:part name="getECSpecReturn" element="impl:GetECSpecResult"/>
2378 </wsdl:message>
2379
2380 <wsdl:message name="getECSpecNamesRequest">
2381     <wsdl:part name="parms" element="impl:GetECSpecNames"/>
2382 </wsdl:message>
2383 <wsdl:message name="getECSpecNamesResponse">
2384     <wsdl:part name="getECSpecNamesReturn" element="impl:GetECSpecNamesResult"/>
2385 </wsdl:message>
2386
2387 <wsdl:message name="subscribeRequest">
2388     <wsdl:part name="parms" element="impl:Subscribe"/>
2389 </wsdl:message>
2390 <wsdl:message name="subscribeResponse">
2391     <wsdl:part name="subscribeReturn" element="impl:VoidHolder"/>
2392 </wsdl:message>
2393
2394 <wsdl:message name="unsubscribeRequest">
2395     <wsdl:part name="parms" element="impl:Unsubscribe"/>
2396 </wsdl:message>
2397 <wsdl:message name="unsubscribeResponse">

```

```

2398     <wsdl:part name="unsubscribeReturn" element="impl:VoidHolder"/>
2399 </wsdl:message>
2400
2401 <wsdl:message name="pollRequest">
2402     <wsdl:part name="parms" element="impl:Poll"/>
2403 </wsdl:message>
2404 <wsdl:message name="pollResponse">
2405     <wsdl:part name="pollReturn" element="impl:PollResult"/>
2406 </wsdl:message>
2407
2408 <wsdl:message name="immediateRequest">
2409     <wsdl:part name="parms" element="impl:Immediate"/>
2410 </wsdl:message>
2411 <wsdl:message name="immediateResponse">
2412     <wsdl:part name="immediateReturn" element="impl:ImmediateResult"/>
2413 </wsdl:message>
2414
2415 <wsdl:message name="getSubscribersRequest">
2416     <wsdl:part name="parms" element="impl:GetSubscribers"/>
2417 </wsdl:message>
2418 <wsdl:message name="getSubscribersResponse">
2419     <wsdl:part name="getSubscribersReturn" element="impl:GetSubscribersResult"/>
2420 </wsdl:message>
2421
2422 <wsdl:message name="getStandardVersionRequest">
2423     <wsdl:part name="parms" element="impl:GetStandardVersion"/>
2424 </wsdl:message>
2425 <wsdl:message name="getStandardVersionResponse">
2426     <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult"/>
2427 </wsdl:message>
2428
2429 <wsdl:message name="getVendorVersionRequest">
2430     <wsdl:part name="parms" element="impl:GetVendorVersion"/>
2431 </wsdl:message>
2432 <wsdl:message name="getVendorVersionResponse">
2433     <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult"/>
2434 </wsdl:message>
2435
2436 <wsdl:message name="SecurityExceptionResponse">
2437     <wsdl:part name="fault" element="impl:SecurityException"/>
2438 </wsdl:message>
2439
2440 <wsdl:message name="DuplicateNameExceptionResponse">
2441     <wsdl:part name="fault" element="impl:DuplicateNameException"/>
2442 </wsdl:message>
2443
2444 <wsdl:message name="ECSpecValidationExceptionResponse">
2445     <wsdl:part name="fault" element="impl:ECSpecValidationException"/>
2446 </wsdl:message>
2447
2448 <wsdl:message name="InvalidURIExceptionResponse">
2449     <wsdl:part name="fault" element="impl:InvalidURIException"/>
2450 </wsdl:message>
2451
2452 <wsdl:message name="NoSuchNameExceptionResponse">
2453     <wsdl:part name="fault" element="impl:NoSuchNameException"/>
2454 </wsdl:message>
2455
2456 <wsdl:message name="NoSuchSubscriberExceptionResponse">
2457     <wsdl:part name="fault" element="impl:NoSuchSubscriberException"/>
2458 </wsdl:message>
2459
2460 <wsdl:message name="DuplicateSubscriptionExceptionResponse">
2461     <wsdl:part name="fault" element="impl:DuplicateSubscriptionException"/>
2462 </wsdl:message>
2463
2464 <wsdl:message name="ImplementationExceptionResponse">
2465     <wsdl:part name="fault" element="impl:ImplementationException"/>
2466 </wsdl:message>
2467 <!-- ALESERVICE PORTTYPE -->

```

```

2468
2469 <wsdl:portType name="ALEServicePortType">
2470
2471   <wsdl:operation name="define">
2472     <wsdl:input message="impl:defineRequest" name="defineRequest" />
2473     <wsdl:output message="impl:defineResponse" name="defineResponse" />
2474     <wsdl:fault message="impl:DuplicateNameExceptionResponse"
2475       name="DuplicateNameExceptionFault" />
2476     <wsdl:fault message="impl:ECSpecValidationExceptionResponse"
2477       name="ECSpecValidationExceptionFault" />
2478     <wsdl:fault message="impl:SecurityExceptionResponse"
2479       name="SecurityExceptionFault" />
2480     <wsdl:fault message="impl:ImplementationExceptionResponse"
2481       name="ImplementationExceptionFault" />
2482   </wsdl:operation>
2483
2484   <wsdl:operation name="undefine">
2485     <wsdl:input message="impl:undefineRequest" name="undefineRequest" />
2486     <wsdl:output message="impl:undefineResponse" name="undefineResponse" />
2487     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2488       name="NoSuchNameExceptionFault" />
2489     <wsdl:fault message="impl:SecurityExceptionResponse"
2490       name="SecurityExceptionFault" />
2491     <wsdl:fault message="impl:ImplementationExceptionResponse"
2492       name="ImplementationExceptionFault" />
2493   </wsdl:operation>
2494
2495   <wsdl:operation name="getECSpec">
2496     <wsdl:input message="impl:getECSpecRequest" name="getECSpecRequest" />
2497     <wsdl:output message="impl:getECSpecResponse" name="getECSpecResponse" />
2498     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2499       name="NoSuchNameExceptionFault" />
2500     <wsdl:fault message="impl:SecurityExceptionResponse"
2501       name="SecurityExceptionFault" />
2502     <wsdl:fault message="impl:ImplementationExceptionResponse"
2503       name="ImplementationExceptionFault" />
2504   </wsdl:operation>
2505
2506   <wsdl:operation name="getECSpecNames">
2507     <wsdl:input message="impl:getECSpecNamesRequest" name="getECSpecNamesRequest" />
2508     <wsdl:output message="impl:getECSpecNamesResponse" name="getECSpecNamesResponse" />
2509     <wsdl:fault message="impl:SecurityExceptionResponse"
2510       name="SecurityExceptionFault" />
2511     <wsdl:fault message="impl:ImplementationExceptionResponse"
2512       name="ImplementationExceptionFault" />
2513   </wsdl:operation>
2514
2515   <wsdl:operation name="subscribe">
2516     <wsdl:input message="impl:subscribeRequest" name="subscribeRequest" />
2517     <wsdl:output message="impl:subscribeResponse" name="subscribeResponse" />
2518     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2519       name="NoSuchNameExceptionFault" />
2520     <wsdl:fault message="impl:InvalidURIExceptionResponse"
2521       name="InvalidURIExceptionFault" />
2522     <wsdl:fault message="impl:DuplicateSubscriptionExceptionResponse"
2523       name="DuplicateSubscriptionExceptionFault" />
2524     <wsdl:fault message="impl:SecurityExceptionResponse"
2525       name="SecurityExceptionFault" />
2526     <wsdl:fault message="impl:ImplementationExceptionResponse"
2527       name="ImplementationExceptionFault" />
2528   </wsdl:operation>
2529
2530   <wsdl:operation name="unsubscribe">
2531     <wsdl:input message="impl:unsubscribeRequest" name="unsubscribeRequest" />
2532     <wsdl:output message="impl:unsubscribeResponse" name="unsubscribeResponse" />
2533     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2534       name="NoSuchNameExceptionFault" />
2535     <wsdl:fault message="impl:NoSuchSubscriberExceptionResponse"
2536       name="NoSuchSubscriberExceptionFault" />
2537     <wsdl:fault message="impl:InvalidURIExceptionResponse"

```

```

2538         name="InvalidURIExceptionFault" />
2539     <wsdl:fault message="impl:SecurityExceptionResponse"
2540         name="SecurityExceptionFault" />
2541     <wsdl:fault message="impl:ImplementationExceptionResponse"
2542         name="ImplementationExceptionFault" />
2543 </wsdl:operation>
2544
2545 <wsdl:operation name="poll">
2546     <wsdl:input message="impl:pollRequest" name="pollRequest" />
2547     <wsdl:output message="impl:pollResponse" name="pollResponse" />
2548     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2549         name="NoSuchNameExceptionFault" />
2550     <wsdl:fault message="impl:SecurityExceptionResponse"
2551         name="SecurityExceptionFault" />
2552     <wsdl:fault message="impl:ImplementationExceptionResponse"
2553         name="ImplementationExceptionFault" />
2554 </wsdl:operation>
2555
2556 <wsdl:operation name="immediate">
2557     <wsdl:input message="impl:immediateRequest" name="immediateRequest" />
2558     <wsdl:output message="impl:immediateResponse" name="immediateResponse" />
2559     <wsdl:fault message="impl:ECSpecValidationExceptionResponse"
2560         name="ECSpecValidationExceptionFault" />
2561     <wsdl:fault message="impl:SecurityExceptionResponse"
2562         name="SecurityExceptionFault" />
2563     <wsdl:fault message="impl:ImplementationExceptionResponse"
2564         name="ImplementationExceptionFault" />
2565 </wsdl:operation>
2566
2567 <wsdl:operation name="getSubscribers">
2568     <wsdl:input message="impl:getSubscribersRequest" name="getSubscribersRequest" />
2569     <wsdl:output message="impl:getSubscribersResponse" name="getSubscribersResponse" />
2570     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
2571         name="NoSuchNameExceptionFault" />
2572     <wsdl:fault message="impl:SecurityExceptionResponse"
2573         name="SecurityExceptionFault" />
2574     <wsdl:fault message="impl:ImplementationExceptionResponse"
2575         name="ImplementationExceptionFault" />
2576 </wsdl:operation>
2577
2578 <wsdl:operation name="getStandardVersion">
2579     <wsdl:input message="impl:getStandardVersionRequest"
2580         name="getStandardVersionRequest" />
2581     <wsdl:output message="impl:getStandardVersionResponse"
2582         name="getStandardVersionResponse" />
2583     <wsdl:fault message="impl:ImplementationExceptionResponse"
2584         name="ImplementationExceptionFault" />
2585 </wsdl:operation>
2586
2587 <wsdl:operation name="getVendorVersion">
2588     <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest" />
2589     <wsdl:output message="impl:getVendorVersionResponse"
2590         name="getVendorVersionResponse" />
2591     <wsdl:fault message="impl:ImplementationExceptionResponse"
2592         name="ImplementationExceptionFault" />
2593 </wsdl:operation>
2594 </wsdl:portType>
2595 <!-- ALESERVICE BINDING -->
2596
2597 <wsdl:binding name="ALEServiceBinding" type="impl:ALEServicePortType">
2598     <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
2599
2600     <wsdl:operation name="define">
2601         <wsdlsoap:operation soapAction="" />
2602         <wsdl:input name="defineRequest">
2603             <wsdlsoap:body use="literal" />
2604         </wsdl:input>
2605         <wsdl:output name="defineResponse">
2606             <wsdlsoap:body use="literal" />
2607         </wsdl:output>

```

```

2608     <wsdl:fault name="DuplicateNameExceptionFault">
2609         <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
2610     </wsdl:fault>
2611     <wsdl:fault name="ECSpecValidationExceptionFault">
2612         <wsdlsoap:fault name="ECSpecValidationExceptionFault" use="literal"/>
2613     </wsdl:fault>
2614     <wsdl:fault name="SecurityExceptionFault">
2615         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2616     </wsdl:fault>
2617     <wsdl:fault name="ImplementationExceptionFault">
2618         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2619     </wsdl:fault>
2620 </wsdl:operation>
2621
2622 <wsdl:operation name="undefine">
2623     <wsdlsoap:operation soapAction=""/>
2624     <wsdl:input name="undefineRequest">
2625         <wsdlsoap:body use="literal"/>
2626     </wsdl:input>
2627     <wsdl:output name="undefineResponse">
2628         <wsdlsoap:body use="literal"/>
2629     </wsdl:output>
2630     <wsdl:fault name="NoSuchNameExceptionFault">
2631         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2632     </wsdl:fault>
2633     <wsdl:fault name="SecurityExceptionFault">
2634         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2635     </wsdl:fault>
2636     <wsdl:fault name="ImplementationExceptionFault">
2637         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2638     </wsdl:fault>
2639 </wsdl:operation>
2640
2641 <wsdl:operation name="getECSpec">
2642     <wsdlsoap:operation soapAction=""/>
2643     <wsdl:input name="getECSpecRequest">
2644         <wsdlsoap:body use="literal"/>
2645     </wsdl:input>
2646     <wsdl:output name="getECSpecResponse">
2647         <wsdlsoap:body use="literal"/>
2648     </wsdl:output>
2649     <wsdl:fault name="NoSuchNameExceptionFault">
2650         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2651     </wsdl:fault>
2652     <wsdl:fault name="SecurityExceptionFault">
2653         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2654     </wsdl:fault>
2655     <wsdl:fault name="ImplementationExceptionFault">
2656         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2657     </wsdl:fault>
2658 </wsdl:operation>
2659
2660 <wsdl:operation name="getECSpecNames">
2661     <wsdlsoap:operation soapAction=""/>
2662     <wsdl:input name="getECSpecNamesRequest">
2663         <wsdlsoap:body use="literal"/>
2664     </wsdl:input>
2665     <wsdl:output name="getECSpecNamesResponse">
2666         <wsdlsoap:body use="literal"/>
2667     </wsdl:output>
2668     <wsdl:fault name="SecurityExceptionFault">
2669         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2670     </wsdl:fault>
2671     <wsdl:fault name="ImplementationExceptionFault">
2672         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2673     </wsdl:fault>
2674 </wsdl:operation>
2675
2676 <wsdl:operation name="subscribe">
2677     <wsdlsoap:operation soapAction=""/>

```

```

2678     <wsdl:input name="subscribeRequest">
2679         <wsdlsoap:body use="literal"/>
2680     </wsdl:input>
2681     <wsdl:output name="subscribeResponse">
2682         <wsdlsoap:body use="literal"/>
2683     </wsdl:output>
2684     <wsdl:fault name="NoSuchNameExceptionFault">
2685         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2686     </wsdl:fault>
2687     <wsdl:fault name="InvalidURIExceptionFault">
2688         <wsdlsoap:fault name="InvalidURIExceptionFault" use="literal"/>
2689     </wsdl:fault>
2690     <wsdl:fault name="DuplicateSubscriptionExceptionFault">
2691         <wsdlsoap:fault name="DuplicateSubscriptionExceptionFault" use="literal"/>
2692     </wsdl:fault>
2693     <wsdl:fault name="SecurityExceptionFault">
2694         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2695     </wsdl:fault>
2696     <wsdl:fault name="ImplementationExceptionFault">
2697         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2698     </wsdl:fault>
2699 </wsdl:operation>
2700
2701 <wsdl:operation name="unsubscribe">
2702     <wsdlsoap:operation soapAction=""/>
2703     <wsdl:input name="unsubscribeRequest">
2704         <wsdlsoap:body use="literal"/>
2705     </wsdl:input>
2706     <wsdl:output name="unsubscribeResponse">
2707         <wsdlsoap:body use="literal"/>
2708     </wsdl:output>
2709     <wsdl:fault name="NoSuchNameExceptionFault">
2710         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2711     </wsdl:fault>
2712     <wsdl:fault name="NoSuchSubscriberExceptionFault">
2713         <wsdlsoap:fault name="NoSuchSubscriberExceptionFault" use="literal"/>
2714     </wsdl:fault>
2715     <wsdl:fault name="InvalidURIExceptionFault">
2716         <wsdlsoap:fault name="InvalidURIExceptionFault" use="literal"/>
2717     </wsdl:fault>
2718     <wsdl:fault name="SecurityExceptionFault">
2719         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2720     </wsdl:fault>
2721     <wsdl:fault name="ImplementationExceptionFault">
2722         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2723     </wsdl:fault>
2724 </wsdl:operation>
2725
2726 <wsdl:operation name="poll">
2727     <wsdlsoap:operation soapAction=""/>
2728     <wsdl:input name="pollRequest">
2729         <wsdlsoap:body use="literal"/>
2730     </wsdl:input>
2731     <wsdl:output name="pollResponse">
2732         <wsdlsoap:body use="literal"/>
2733     </wsdl:output>
2734     <wsdl:fault name="NoSuchNameExceptionFault">
2735         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2736     </wsdl:fault>
2737     <wsdl:fault name="SecurityExceptionFault">
2738         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2739     </wsdl:fault>
2740     <wsdl:fault name="ImplementationExceptionFault">
2741         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2742     </wsdl:fault>
2743 </wsdl:operation>
2744
2745 <wsdl:operation name="immediate">
2746     <wsdlsoap:operation soapAction=""/>
2747     <wsdl:input name="immediateRequest">

```



```

2748     <wsdlsoap:body use="literal"/>
2749 </wsdl:input>
2750 <wsdl:output name="immediateResponse">
2751   <wsdlsoap:body use="literal"/>
2752 </wsdl:output>
2753 <wsdl:fault name="ECSpecValidationExceptionFault">
2754   <wsdlsoap:fault name="ECSpecValidationExceptionFault" use="literal"/>
2755 </wsdl:fault>
2756 <wsdl:fault name="SecurityExceptionFault">
2757   <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2758 </wsdl:fault>
2759 <wsdl:fault name="ImplementationExceptionFault">
2760   <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2761 </wsdl:fault>
2762 </wsdl:operation>
2763
2764 <wsdl:operation name="getSubscribers">
2765   <wsdlsoap:operation soapAction=""/>
2766   <wsdl:input name="getSubscribersRequest">
2767     <wsdlsoap:body use="literal"/>
2768   </wsdl:input>
2769   <wsdl:output name="getSubscribersResponse">
2770     <wsdlsoap:body use="literal"/>
2771   </wsdl:output>
2772   <wsdl:fault name="NoSuchNameExceptionFault">
2773     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
2774   </wsdl:fault>
2775   <wsdl:fault name="SecurityExceptionFault">
2776     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
2777   </wsdl:fault>
2778   <wsdl:fault name="ImplementationExceptionFault">
2779     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2780   </wsdl:fault>
2781 </wsdl:operation>
2782
2783 <wsdl:operation name="getStandardVersion">
2784   <wsdlsoap:operation soapAction=""/>
2785   <wsdl:input name="getStandardVersionRequest">
2786     <wsdlsoap:body use="literal"/>
2787   </wsdl:input>
2788   <wsdl:output name="getStandardVersionResponse">
2789     <wsdlsoap:body use="literal"/>
2790   </wsdl:output>
2791   <wsdl:fault name="ImplementationExceptionFault">
2792     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2793   </wsdl:fault>
2794 </wsdl:operation>
2795
2796 <wsdl:operation name="getVendorVersion">
2797   <wsdlsoap:operation soapAction=""/>
2798   <wsdl:input name="getVendorVersionRequest">
2799     <wsdlsoap:body use="literal"/>
2800   </wsdl:input>
2801   <wsdl:output name="getVendorVersionResponse">
2802     <wsdlsoap:body use="literal"/>
2803   </wsdl:output>
2804   <wsdl:fault name="ImplementationExceptionFault">
2805     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
2806   </wsdl:fault>
2807 </wsdl:operation>
2808 </wsdl:binding>
2809
2810 <!-- ALESERVICE -->
2811 <wsdl:service name="ALEService">
2812   <wsdl:port binding="impl:ALEServiceBinding" name="ALEServicePort">
2813     <!-- The value of the location attribute below is an example only;
2814           Implementations are free to choose any appropriate URL. -->
2815     <wsdlsoap:address location="http://localhost:8080/services/ALEService"/>
2816   </wsdl:port>
2817 </wsdl:service>

```



2818 </wsdl:definitions>

## 2819 4.4 ALE Writing API SOAP Binding

2820 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
2821 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Writing  
2822 API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0 [WSI].

```
2823 <?xml version="1.0" encoding="UTF-8"?>
2824 <!-- ALECCSERVICE DEFINITIONS -->
2825 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
2826                 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
2827                 xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
2828                 xmlns:ale="urn:epcglobal:ale:xsd:1"
2829                 xmlns:impl="urn:epcglobal:alecc:wsdl:1"
2830                 targetNamespace="urn:epcglobal:alecc:wsdl:1">
2831   <!-- ALECCSERVICE TYPES -->
2832   <wsdl:types>
2833     <xsd:schema targetNamespace="urn:epcglobal:alecc:wsdl:1">
2834       <xsd:import namespace="urn:epcglobal:ale:xsd:1"
2835                 schemaLocation="EPCglobal-ale-1_1-alecc.xsd"/>
2836       <!-- ALECCSERVICE MESSAGE WRAPPERS -->
2837
2838       <xsd:element name="Define" type="impl:Define"/>
2839       <xsd:complexType name="Define">
2840         <xsd:sequence>
2841           <xsd:element name="specName" type="xsd:string"/>
2842           <xsd:element name="spec" type="ale:CCSpec"/>
2843         </xsd:sequence>
2844       </xsd:complexType>
2845       <xsd:element name="DefineResult">
2846         <xsd:complexType/>
2847       </xsd:element>
2848
2849       <xsd:element name="Undefine" type="impl:Undefine"/>
2850       <xsd:complexType name="Undefine">
2851         <xsd:sequence>
2852           <xsd:element name="specName" type="xsd:string"/>
2853         </xsd:sequence>
2854       </xsd:complexType>
2855       <xsd:element name="UndefineResult">
2856         <xsd:complexType/>
2857       </xsd:element>
2858
2859       <xsd:element name="GetCCSpec" type="impl:GetCCSpec"/>
2860       <xsd:complexType name="GetCCSpec">
2861         <xsd:sequence>
2862           <xsd:element name="specName" type="xsd:string"/>
2863         </xsd:sequence>
2864       </xsd:complexType>
2865       <xsd:element name="GetCCSpecResult" type="ale:CCSpec"/>
2866
2867       <xsd:element name="GetCCSpecNames" type="impl:EmptyParms"/>
2868       <xsd:element name="GetCCSpecNamesResult" type="impl:ArrayOfString"/>
2869
2870       <xsd:element name="Subscribe" type="impl:Subscribe"/>
2871       <xsd:complexType name="Subscribe">
2872         <xsd:sequence>
2873           <xsd:element name="specName" type="xsd:string"/>
2874           <xsd:element name="notificationURI" type="xsd:string"/>
2875         </xsd:sequence>
2876       </xsd:complexType>
2877       <xsd:element name="SubscribeResult">
2878         <xsd:complexType/>
2879       </xsd:element>
2880
2881       <xsd:element name="Unsubscribe" type="impl:Unsubscribe"/>
2882       <xsd:complexType name="Unsubscribe">
```

```

2883     <xsd:sequence>
2884       <xsd:element name="specName" type="xsd:string"/>
2885       <xsd:element name="notificationURI" type="xsd:string"/>
2886     </xsd:sequence>
2887   </xsd:complexType>
2888   <xsd:element name="UnsubscribeResult">
2889     <xsd:complexType/>
2890   </xsd:element>
2891
2892   <xsd:element name="Poll" type="impl:Poll"/>
2893   <xsd:complexType name="Poll">
2894     <xsd:sequence>
2895       <xsd:element name="specName" type="xsd:string"/>
2896       <xsd:element name="params" type="ale:CCParameterList"/>
2897     </xsd:sequence>
2898   </xsd:complexType>
2899   <xsd:element name="PollResult" type="ale:CCReports"/>
2900
2901   <xsd:element name="Immediate" type="impl:Immediate"/>
2902   <xsd:complexType name="Immediate">
2903     <xsd:sequence>
2904       <xsd:element name="spec" type="ale:CCSpec"/>
2905     </xsd:sequence>
2906   </xsd:complexType>
2907   <xsd:element name="ImmediateResult" type="ale:CCReports"/>
2908
2909   <xsd:element name="GetSubscribers" type="impl:GetSubscribers"/>
2910   <xsd:complexType name="GetSubscribers">
2911     <xsd:sequence>
2912       <xsd:element name="specName" type="xsd:string"/>
2913     </xsd:sequence>
2914   </xsd:complexType>
2915   <xsd:element name="GetSubscribersResult" type="impl:ArrayOfString"/>
2916
2917   <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
2918   <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
2919
2920   <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
2921   <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
2922
2923   <xsd:element name="DefineEPCCache" type="impl:DefineEPCCache"/>
2924   <xsd:complexType name="DefineEPCCache">
2925     <xsd:sequence>
2926       <xsd:element name="cacheName" type="xsd:string"/>
2927       <xsd:element name="spec" type="ale:EPCCacheSpec"/>
2928       <xsd:element name="replenishment" type="ale:EPCCPatternList"/>
2929     </xsd:sequence>
2930   </xsd:complexType>
2931   <xsd:element name="DefineEPCCacheResult">
2932     <xsd:complexType/>
2933   </xsd:element>
2934
2935   <xsd:element name="UndefineEPCCache" type="impl:UndefineEPCCache"/>
2936   <xsd:complexType name="UndefineEPCCache">
2937     <xsd:sequence>
2938       <xsd:element name="cacheName" type="xsd:string"/>
2939     </xsd:sequence>
2940   </xsd:complexType>
2941   <xsd:element name="UndefineEPCCacheResult" type="ale:EPCCPatternList"/>
2942
2943   <xsd:element name="GetEPCCache" type="impl:GetEPCCache"/>
2944   <xsd:complexType name="GetEPCCache">
2945     <xsd:sequence>
2946       <xsd:element name="cacheName" type="xsd:string"/>
2947     </xsd:sequence>
2948   </xsd:complexType>
2949   <xsd:element name="GetEPCCacheResult" type="ale:EPCCacheSpec"/>
2950
2951   <xsd:element name="GetEPCCacheNames" type="impl:EmptyParms"/>
2952   <xsd:element name="GetEPCCacheNamesResult" type="impl:ArrayOfString"/>

```

```

2953
2954 <xsd:element name="ReplenishEPCCache" type="impl:ReplenishEPCCache"/>
2955 <xsd:complexType name="ReplenishEPCCache">
2956   <xsd:sequence>
2957     <xsd:element name="cacheName" type="xsd:string"/>
2958     <xsd:element name="replenishment" type="ale:EPCCPatternList"/>
2959   </xsd:sequence>
2960 </xsd:complexType>
2961 <xsd:element name="ReplenishEPCCacheResult">
2962   <xsd:complexType/>
2963 </xsd:element>
2964
2965 <xsd:element name="DepleteEPCCache" type="impl:DepleteEPCCache"/>
2966 <xsd:complexType name="DepleteEPCCache">
2967   <xsd:sequence>
2968     <xsd:element name="cacheName" type="xsd:string"/>
2969   </xsd:sequence>
2970 </xsd:complexType>
2971 <xsd:element name="DepleteEPCCacheResult" type="ale:EPCCPatternList"/>
2972
2973 <xsd:element name="GetEPCCacheContents" type="impl:GetEPCCacheContents"/>
2974 <xsd:complexType name="GetEPCCacheContents">
2975   <xsd:sequence>
2976     <xsd:element name="cacheName" type="xsd:string"/>
2977   </xsd:sequence>
2978 </xsd:complexType>
2979 <xsd:element name="GetEPCCacheContentsResult" type="ale:EPCCPatternList"/>
2980
2981 <xsd:element name="DefineAssocTable" type="impl:DefineAssocTable"/>
2982 <xsd:complexType name="DefineAssocTable">
2983   <xsd:sequence>
2984     <xsd:element name="tableName" type="xsd:string"/>
2985     <xsd:element name="spec" type="ale:AssocTableSpec"/>
2986     <xsd:element name="entries" type="ale:AssocTableEntryList"/>
2987   </xsd:sequence>
2988 </xsd:complexType>
2989 <xsd:element name="DefineAssocTableResult">
2990   <xsd:complexType/>
2991 </xsd:element>
2992
2993 <xsd:element name="UndefineAssocTable" type="impl:UndefineAssocTable"/>
2994 <xsd:complexType name="UndefineAssocTable">
2995   <xsd:sequence>
2996     <xsd:element name="tableName" type="xsd:string"/>
2997   </xsd:sequence>
2998 </xsd:complexType>
2999 <xsd:element name="UndefineAssocTableResult">
3000   <xsd:complexType/>
3001 </xsd:element>
3002
3003 <xsd:element name="GetAssocTableNames" type="impl:EmptyParms"/>
3004 <xsd:element name="GetAssocTableNamesResult" type="impl:ArrayOfString"/>
3005
3006 <xsd:element name="GetAssocTable" type="impl:GetAssocTable"/>
3007 <xsd:complexType name="GetAssocTable">
3008   <xsd:sequence>
3009     <xsd:element name="tableName" type="xsd:string"/>
3010   </xsd:sequence>
3011 </xsd:complexType>
3012 <xsd:element name="GetAssocTableResult" type="ale:AssocTableSpec"/>
3013
3014 <xsd:element name="PutAssocTableEntries" type="impl:PutAssocTableEntries"/>
3015 <xsd:complexType name="PutAssocTableEntries">
3016   <xsd:sequence>
3017     <xsd:element name="tableName" type="xsd:string"/>
3018     <xsd:element name="entries" type="ale:AssocTableEntryList"/>
3019   </xsd:sequence>
3020 </xsd:complexType>
3021 <xsd:element name="PutAssocTableEntriesResult">
3022   <xsd:complexType/>

```

```

3023     </xsd:element>
3024
3025     <xsd:element name="GetAssocTableValue" type="impl:GetAssocTableValue"/>
3026     <xsd:complexType name="GetAssocTableValue">
3027         <xsd:sequence>
3028             <xsd:element name="tableName" type="xsd:string"/>
3029             <xsd:element name="epc" type="xsd:string"/>
3030         </xsd:sequence>
3031     </xsd:complexType>
3032     <xsd:element name="GetAssocTableValueResult" type="xsd:string"/>
3033
3034     <xsd:element name="GetAssocTableEntries" type="impl:GetAssocTableEntries"/>
3035     <xsd:complexType name="GetAssocTableEntries">
3036         <xsd:sequence>
3037             <xsd:element name="tableName" type="xsd:string"/>
3038             <xsd:element name="patList" type="ale:EPCPatternList"/>
3039         </xsd:sequence>
3040     </xsd:complexType>
3041     <xsd:element name="GetAssocTableEntriesResult" type="ale:AssocTableEntryList"/>
3042
3043     <xsd:element name="RemoveAssocTableEntry" type="impl:RemoveAssocTableEntry"/>
3044     <xsd:complexType name="RemoveAssocTableEntry">
3045         <xsd:sequence>
3046             <xsd:element name="tableName" type="xsd:string"/>
3047             <xsd:element name="epc" type="xsd:string"/>
3048         </xsd:sequence>
3049     </xsd:complexType>
3050     <xsd:element name="RemoveAssocTableEntryResult">
3051         <xsd:complexType/>
3052     </xsd:element>
3053
3054     <xsd:element name="RemoveAssocTableEntries" type="impl:RemoveAssocTableEntries"/>
3055     <xsd:complexType name="RemoveAssocTableEntries">
3056         <xsd:sequence>
3057             <xsd:element name="tableName" type="xsd:string"/>
3058             <xsd:element name="patList" type="ale:EPCPatternList"/>
3059         </xsd:sequence>
3060     </xsd:complexType>
3061     <xsd:element name="RemoveAssocTableEntriesResult">
3062         <xsd:complexType/>
3063     </xsd:element>
3064
3065     <xsd:element name="DefineRNG" type="impl:DefineRNG"/>
3066     <xsd:complexType name="DefineRNG">
3067         <xsd:sequence>
3068             <xsd:element name="rngName" type="xsd:string"/>
3069             <xsd:element name="rngSpec" type="ale:RNGSpec"/>
3070         </xsd:sequence>
3071     </xsd:complexType>
3072     <xsd:element name="DefineRNGResult">
3073         <xsd:complexType/>
3074     </xsd:element>
3075
3076     <xsd:element name="UndefineRNG" type="impl:UndefineRNG"/>
3077     <xsd:complexType name="UndefineRNG">
3078         <xsd:sequence>
3079             <xsd:element name="rngName" type="xsd:string"/>
3080         </xsd:sequence>
3081     </xsd:complexType>
3082     <xsd:element name="UndefineRNGResult">
3083         <xsd:complexType/>
3084     </xsd:element>
3085
3086     <xsd:element name="GetRNGNames" type="impl:EmptyParms"/>
3087     <xsd:element name="GetRNGNamesResult" type="impl:ArrayOfString"/>
3088
3089     <xsd:element name="GetRNG" type="impl:GetRNG"/>
3090     <xsd:complexType name="GetRNG">
3091         <xsd:sequence>
3092             <xsd:element name="rngName" type="xsd:string"/>

```

```

3093     </xsd:sequence>
3094 </xsd:complexType>
3095 <xsd:element name="GetRNGResult" type="ale:RNGSpec"/>
3096
3097 <xsd:element name="ALEException" type="impl:ALEException"/>
3098 <xsd:complexType name="ALEException">
3099   <xsd:sequence>
3100     <xsd:element name="reason" type="xsd:string"/>
3101   </xsd:sequence>
3102 </xsd:complexType>
3103
3104 <xsd:element name="SecurityException" type="impl:SecurityException"/>
3105 <xsd:complexType name="SecurityException">
3106   <xsd:complexContent>
3107     <xsd:extension base="impl:ALEException"/>
3108   </xsd:complexContent>
3109 </xsd:complexType>
3110
3111 <xsd:element name="DuplicateNameException" type="impl:DuplicateNameException"/>
3112 <xsd:complexType name="DuplicateNameException">
3113   <xsd:complexContent>
3114     <xsd:extension base="impl:ALEException"/>
3115   </xsd:complexContent>
3116 </xsd:complexType>
3117
3118 <xsd:element name="CCSpecValidationException"
3119           type="impl:CCSpecValidationException"/>
3120 <xsd:complexType name="CCSpecValidationException">
3121   <xsd:complexContent>
3122     <xsd:extension base="impl:ALEException"/>
3123   </xsd:complexContent>
3124 </xsd:complexType>
3125
3126 <xsd:element name="InvalidURIException" type="impl:InvalidURIException"/>
3127 <xsd:complexType name="InvalidURIException">
3128   <xsd:complexContent>
3129     <xsd:extension base="impl:ALEException"/>
3130   </xsd:complexContent>
3131 </xsd:complexType>
3132
3133 <xsd:element name="NoSuchNameException" type="impl:NoSuchNameException"/>
3134 <xsd:complexType name="NoSuchNameException">
3135   <xsd:complexContent>
3136     <xsd:extension base="impl:ALEException"/>
3137   </xsd:complexContent>
3138 </xsd:complexType>
3139
3140 <xsd:element name="NoSuchSubscriberException"
3141           type="impl:NoSuchSubscriberException"/>
3142 <xsd:complexType name="NoSuchSubscriberException">
3143   <xsd:complexContent>
3144     <xsd:extension base="impl:ALEException"/>
3145   </xsd:complexContent>
3146 </xsd:complexType>
3147
3148 <xsd:element name="DuplicateSubscriptionException"
3149           type="impl:DuplicateSubscriptionException"/>
3150 <xsd:complexType name="DuplicateSubscriptionException">
3151   <xsd:complexContent>
3152     <xsd:extension base="impl:ALEException"/>
3153   </xsd:complexContent>
3154 </xsd:complexType>
3155
3156 <xsd:element name="ParameterException" type="impl:ParameterException"/>
3157 <xsd:complexType name="ParameterException">
3158   <xsd:complexContent>
3159     <xsd:extension base="impl:ALEException"/>
3160   </xsd:complexContent>
3161 </xsd:complexType>
3162

```

```

3163 <xsd:element name="ParameterForbiddenException"
3164           type="impl:ParameterForbiddenException" />
3165 <xsd:complexType name="ParameterForbiddenException">
3166   <xsd:complexContent>
3167     <xsd:extension base="impl:ALEException" />
3168   </xsd:complexContent>
3169 </xsd:complexType>
3170
3171 <xsd:element name="ImplementationException" type="impl:ImplementationException" />
3172 <xsd:complexType name="ImplementationException">
3173   <xsd:complexContent>
3174     <xsd:extension base="impl:ALEException">
3175       <xsd:sequence>
3176         <xsd:element name="severity" type="impl:ImplementationExceptionSeverity" />
3177       </xsd:sequence>
3178     </xsd:extension>
3179   </xsd:complexContent>
3180 </xsd:complexType>
3181
3182 <xsd:element name="EPCCacheSpecValidationException"
3183           type="impl:EPCCacheSpecValidationException" />
3184 <xsd:complexType name="EPCCacheSpecValidationException">
3185   <xsd:complexContent>
3186     <xsd:extension base="impl:ALEException" />
3187   </xsd:complexContent>
3188 </xsd:complexType>
3189
3190 <xsd:element name="InvalidPatternException" type="impl:InvalidPatternException" />
3191 <xsd:complexType name="InvalidPatternException">
3192   <xsd:complexContent>
3193     <xsd:extension base="impl:ALEException" />
3194   </xsd:complexContent>
3195 </xsd:complexType>
3196
3197 <xsd:element name="InUseException" type="impl:InUseException" />
3198 <xsd:complexType name="InUseException">
3199   <xsd:complexContent>
3200     <xsd:extension base="impl:ALEException" />
3201   </xsd:complexContent>
3202 </xsd:complexType>
3203
3204 <xsd:element name="AssocTableValidationException"
3205           type="impl:AssocTableValidationException" />
3206 <xsd:complexType name="AssocTableValidationException">
3207   <xsd:complexContent>
3208     <xsd:extension base="impl:ALEException" />
3209   </xsd:complexContent>
3210 </xsd:complexType>
3211
3212 <xsd:element name="InvalidEPCEException" type="impl:InvalidEPCEException" />
3213 <xsd:complexType name="InvalidEPCEException">
3214   <xsd:complexContent>
3215     <xsd:extension base="impl:ALEException" />
3216   </xsd:complexContent>
3217 </xsd:complexType>
3218
3219 <xsd:element name="InvalidAssocTableEntryException"
3220           type="impl:InvalidAssocTableEntryException" />
3221 <xsd:complexType name="InvalidAssocTableEntryException">
3222   <xsd:complexContent>
3223     <xsd:extension base="impl:ALEException" />
3224   </xsd:complexContent>
3225 </xsd:complexType>
3226
3227 <xsd:element name="RNGValidationException" type="impl:RNGValidationException" />
3228 <xsd:complexType name="RNGValidationException">
3229   <xsd:complexContent>
3230     <xsd:extension base="impl:ALEException" />
3231   </xsd:complexContent>
3232 </xsd:complexType>

```

```

3233
3234     <xsd:complexType name="ArrayOfString">
3235         <xsd:sequence>
3236             <xsd:element name="string" type="xsd:string" minOccurs="0"
3237                 maxOccurs="unbounded" />
3238         </xsd:sequence>
3239     </xsd:complexType>
3240
3241     <!-- The ImplementationExceptionSeverity type is an enumerated type.
3242         The following strings are legal values for this type:
3243         ERROR
3244         SEVERE
3245         -->
3246     <xsd:simpleType name="ImplementationExceptionSeverity">
3247         <xsd:restriction base="xsd:string" />
3248     </xsd:simpleType>
3249
3250     <xsd:complexType name="EmptyParms" />
3251 </xsd:schema>
3252 </wsdl:types>
3253 <!-- ALECCSERVICE MESSAGES -->
3254
3255 <wsdl:message name="defineRequest">
3256     <wsdl:part name="parms" element="impl:Define" />
3257 </wsdl:message>
3258 <wsdl:message name="defineResponse">
3259     <wsdl:part name="defineReturn" element="impl:DefineResult" />
3260 </wsdl:message>
3261
3262 <wsdl:message name="undefineRequest">
3263     <wsdl:part name="parms" element="impl:Undefine" />
3264 </wsdl:message>
3265 <wsdl:message name="undefineResponse">
3266     <wsdl:part name="undefineReturn" element="impl:UndefineResult" />
3267 </wsdl:message>
3268
3269 <wsdl:message name="getCCSpecRequest">
3270     <wsdl:part name="parms" element="impl:GetCCSpec" />
3271 </wsdl:message>
3272 <wsdl:message name="getCCSpecResponse">
3273     <wsdl:part name="getCCSpecReturn" element="impl:GetCCSpecResult" />
3274 </wsdl:message>
3275
3276 <wsdl:message name="getCCSpecNamesRequest">
3277     <wsdl:part name="parms" element="impl:GetCCSpecNames" />
3278 </wsdl:message>
3279 <wsdl:message name="getCCSpecNamesResponse">
3280     <wsdl:part name="getCCSpecNamesReturn" element="impl:GetCCSpecNamesResult" />
3281 </wsdl:message>
3282
3283 <wsdl:message name="subscribeRequest">
3284     <wsdl:part name="parms" element="impl:Subscribe" />
3285 </wsdl:message>
3286 <wsdl:message name="subscribeResponse">
3287     <wsdl:part name="subscribeReturn" element="impl:SubscribeResult" />
3288 </wsdl:message>
3289
3290 <wsdl:message name="unsubscribeRequest">
3291     <wsdl:part name="parms" element="impl:Unsubscribe" />
3292 </wsdl:message>
3293 <wsdl:message name="unsubscribeResponse">
3294     <wsdl:part name="unsubscribeReturn" element="impl:UnsubscribeResult" />
3295 </wsdl:message>
3296
3297 <wsdl:message name="pollRequest">
3298     <wsdl:part name="parms" element="impl:Poll" />
3299 </wsdl:message>
3300 <wsdl:message name="pollResponse">
3301     <wsdl:part name="pollReturn" element="impl:PollResult" />
3302 </wsdl:message>

```



```

3303
3304 <wsdl:message name="immediateRequest">
3305   <wsdl:part name="parms" element="impl:Immediate"/>
3306 </wsdl:message>
3307 <wsdl:message name="immediateResponse">
3308   <wsdl:part name="immediateReturn" element="impl:ImmediateResult"/>
3309 </wsdl:message>
3310
3311 <wsdl:message name="getSubscribersRequest">
3312   <wsdl:part name="parms" element="impl:GetSubscribers"/>
3313 </wsdl:message>
3314 <wsdl:message name="getSubscribersResponse">
3315   <wsdl:part name="getSubscribersReturn" element="impl:GetSubscribersResult"/>
3316 </wsdl:message>
3317
3318 <wsdl:message name="getStandardVersionRequest">
3319   <wsdl:part name="parms" element="impl:GetStandardVersion"/>
3320 </wsdl:message>
3321 <wsdl:message name="getStandardVersionResponse">
3322   <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult"/>
3323 </wsdl:message>
3324
3325 <wsdl:message name="getVendorVersionRequest">
3326   <wsdl:part name="parms" element="impl:GetVendorVersion"/>
3327 </wsdl:message>
3328 <wsdl:message name="getVendorVersionResponse">
3329   <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult"/>
3330 </wsdl:message>
3331
3332 <wsdl:message name="defineEPCCacheRequest">
3333   <wsdl:part name="parms" element="impl:DefineEPCCache"/>
3334 </wsdl:message>
3335 <wsdl:message name="defineEPCCacheResponse">
3336   <wsdl:part name="defineEPCCacheReturn" element="impl:DefineEPCCacheResult"/>
3337 </wsdl:message>
3338
3339 <wsdl:message name="undefineEPCCacheRequest">
3340   <wsdl:part name="parms" element="impl:UndefineEPCCache"/>
3341 </wsdl:message>
3342 <wsdl:message name="undefineEPCCacheResponse">
3343   <wsdl:part name="undefineEPCCacheReturn" element="impl:UndefineEPCCacheResult"/>
3344 </wsdl:message>
3345
3346 <wsdl:message name="getEPCCacheRequest">
3347   <wsdl:part name="parms" element="impl:GetEPCCache"/>
3348 </wsdl:message>
3349 <wsdl:message name="getEPCCacheResponse">
3350   <wsdl:part name="getEPCCacheReturn" element="impl:GetEPCCacheResult"/>
3351 </wsdl:message>
3352
3353 <wsdl:message name="getEPCCacheNamesRequest">
3354   <wsdl:part name="parms" element="impl:GetEPCCacheNames"/>
3355 </wsdl:message>
3356 <wsdl:message name="getEPCCacheNamesResponse">
3357   <wsdl:part name="getEPCCacheNamesReturn" element="impl:GetEPCCacheNamesResult"/>
3358 </wsdl:message>
3359
3360 <wsdl:message name="replenishEPCCacheRequest">
3361   <wsdl:part name="parms" element="impl:ReplenishEPCCache"/>
3362 </wsdl:message>
3363 <wsdl:message name="replenishEPCCacheResponse">
3364   <wsdl:part name="replenishEPCCacheReturn" element="impl:ReplenishEPCCacheResult"/>
3365 </wsdl:message>
3366
3367 <wsdl:message name="depleteEPCCacheRequest">
3368   <wsdl:part name="parms" element="impl:DepleteEPCCache"/>
3369 </wsdl:message>
3370 <wsdl:message name="depleteEPCCacheResponse">
3371   <wsdl:part name="depleteEPCCacheReturn" element="impl:DepleteEPCCacheResult"/>
3372 </wsdl:message>

```

```

3373
3374 <wsdl:message name="getEPCCacheContentsRequest">
3375   <wsdl:part name="parms" element="impl:GetEPCCacheContents"/>
3376 </wsdl:message>
3377 <wsdl:message name="getEPCCacheContentsResponse">
3378   <wsdl:part name="getEPCCacheContentsReturn"
3379     element="impl:GetEPCCacheContentsResult"/>
3380 </wsdl:message>
3381
3382 <wsdl:message name="defineAssocTableRequest">
3383   <wsdl:part name="parms" element="impl:DefineAssocTable"/>
3384 </wsdl:message>
3385 <wsdl:message name="defineAssocTableResponse">
3386   <wsdl:part name="defineAssocTableReturn" element="impl:DefineAssocTableResult"/>
3387 </wsdl:message>
3388
3389 <wsdl:message name="undefineAssocTableRequest">
3390   <wsdl:part name="parms" element="impl:UndefineAssocTable"/>
3391 </wsdl:message>
3392 <wsdl:message name="undefineAssocTableResponse">
3393   <wsdl:part name="undefineAssocTableReturn" element="impl:UndefineAssocTableResult"/>
3394 </wsdl:message>
3395
3396 <wsdl:message name="getAssocTableNamesRequest">
3397   <wsdl:part name="parms" element="impl:GetAssocTableNames"/>
3398 </wsdl:message>
3399 <wsdl:message name="getAssocTableNamesResponse">
3400   <wsdl:part name="getAssocTableNamesReturn" element="impl:GetAssocTableNamesResult"/>
3401 </wsdl:message>
3402
3403 <wsdl:message name="getAssocTableRequest">
3404   <wsdl:part name="parms" element="impl:GetAssocTable"/>
3405 </wsdl:message>
3406 <wsdl:message name="getAssocTableResponse">
3407   <wsdl:part name="getAssocTableReturn" element="impl:GetAssocTableResult"/>
3408 </wsdl:message>
3409
3410 <wsdl:message name="putAssocTableEntriesRequest">
3411   <wsdl:part name="parms" element="impl:PutAssocTableEntries"/>
3412 </wsdl:message>
3413 <wsdl:message name="putAssocTableEntriesResponse">
3414   <wsdl:part name="putAssocTableEntriesReturn"
3415     element="impl:PutAssocTableEntriesResult"/>
3416 </wsdl:message>
3417
3418 <wsdl:message name="getAssocTableValueRequest">
3419   <wsdl:part name="parms" element="impl:GetAssocTableValue"/>
3420 </wsdl:message>
3421 <wsdl:message name="getAssocTableValueResponse">
3422   <wsdl:part name="getAssocTableValueReturn" element="impl:GetAssocTableValueResult"/>
3423 </wsdl:message>
3424
3425 <wsdl:message name="getAssocTableEntriesRequest">
3426   <wsdl:part name="parms" element="impl:GetAssocTableEntries"/>
3427 </wsdl:message>
3428 <wsdl:message name="getAssocTableEntriesResponse">
3429   <wsdl:part name="getAssocTableEntriesReturn"
3430     element="impl:GetAssocTableEntriesResult"/>
3431 </wsdl:message>
3432
3433 <wsdl:message name="removeAssocTableEntryRequest">
3434   <wsdl:part name="parms" element="impl:RemoveAssocTableEntry"/>
3435 </wsdl:message>
3436 <wsdl:message name="removeAssocTableEntryResponse">
3437   <wsdl:part name="removeAssocTableEntryReturn"
3438     element="impl:RemoveAssocTableEntryResult"/>
3439 </wsdl:message>
3440
3441 <wsdl:message name="removeAssocTableEntriesRequest">
3442   <wsdl:part name="parms" element="impl:RemoveAssocTableEntries"/>

```

```

3443 </wsdl:message>
3444 <wsdl:message name="removeAssocTableEntriesResponse">
3445   <wsdl:part name="removeAssocTableEntriesReturn"
3446     element="impl:RemoveAssocTableEntriesResult"/>
3447 </wsdl:message>
3448
3449 <wsdl:message name="defineRNGRequest">
3450   <wsdl:part name="parms" element="impl:DefineRNG"/>
3451 </wsdl:message>
3452 <wsdl:message name="defineRNGResponse">
3453   <wsdl:part name="defineRNGReturn" element="impl:DefineRNGResult"/>
3454 </wsdl:message>
3455
3456 <wsdl:message name="undefineRNGRequest">
3457   <wsdl:part name="parms" element="impl:UndefineRNG"/>
3458 </wsdl:message>
3459 <wsdl:message name="undefineRNGResponse">
3460   <wsdl:part name="undefineRNGReturn" element="impl:UndefineRNGResult"/>
3461 </wsdl:message>
3462
3463 <wsdl:message name="getRNGNamesRequest">
3464   <wsdl:part name="parms" element="impl:GetRNGNames"/>
3465 </wsdl:message>
3466 <wsdl:message name="getRNGNamesResponse">
3467   <wsdl:part name="getRNGNamesReturn" element="impl:GetRNGNamesResult"/>
3468 </wsdl:message>
3469
3470 <wsdl:message name="getRNGRequest">
3471   <wsdl:part name="parms" element="impl:GetRNG"/>
3472 </wsdl:message>
3473 <wsdl:message name="getRNGResponse">
3474   <wsdl:part name="getRNGReturn" element="impl:GetRNGResult"/>
3475 </wsdl:message>
3476
3477 <wsdl:message name="SecurityExceptionResponse">
3478   <wsdl:part name="fault" element="impl:SecurityException"/>
3479 </wsdl:message>
3480
3481 <wsdl:message name="DuplicateNameExceptionResponse">
3482   <wsdl:part name="fault" element="impl:DuplicateNameException"/>
3483 </wsdl:message>
3484
3485 <wsdl:message name="CCSpecValidationExceptionResponse">
3486   <wsdl:part name="fault" element="impl:CCSpecValidationException"/>
3487 </wsdl:message>
3488
3489 <wsdl:message name="InvalidURIExceptionResponse">
3490   <wsdl:part name="fault" element="impl:InvalidURIException"/>
3491 </wsdl:message>
3492
3493 <wsdl:message name="NoSuchNameExceptionResponse">
3494   <wsdl:part name="fault" element="impl:NoSuchNameException"/>
3495 </wsdl:message>
3496
3497 <wsdl:message name="NoSuchSubscriberExceptionResponse">
3498   <wsdl:part name="fault" element="impl:NoSuchSubscriberException"/>
3499 </wsdl:message>
3500
3501 <wsdl:message name="DuplicateSubscriptionExceptionResponse">
3502   <wsdl:part name="fault" element="impl:DuplicateSubscriptionException"/>
3503 </wsdl:message>
3504
3505 <wsdl:message name="ParameterExceptionResponse">
3506   <wsdl:part name="fault" element="impl:ParameterException"/>
3507 </wsdl:message>
3508
3509 <wsdl:message name="ParameterForbiddenExceptionResponse">
3510   <wsdl:part name="fault" element="impl:ParameterForbiddenException"/>
3511 </wsdl:message>
3512

```

```

3513 <wsdl:message name="ImplementationExceptionResponse">
3514 <wsdl:part name="fault" element="impl:ImplementationException"/>
3515 </wsdl:message>
3516
3517 <wsdl:message name="EPCCacheSpecValidationExceptionResponse">
3518 <wsdl:part name="fault" element="impl:EPCCacheSpecValidationException"/>
3519 </wsdl:message>
3520
3521 <wsdl:message name="InvalidPatternExceptionResponse">
3522 <wsdl:part name="fault" element="impl:InvalidPatternException"/>
3523 </wsdl:message>
3524
3525 <wsdl:message name="InUseExceptionResponse">
3526 <wsdl:part name="fault" element="impl:InUseException"/>
3527 </wsdl:message>
3528
3529 <wsdl:message name="AssocTableValidationExceptionResponse">
3530 <wsdl:part name="fault" element="impl:AssocTableValidationException"/>
3531 </wsdl:message>
3532
3533 <wsdl:message name="InvalidEPCEExceptionResponse">
3534 <wsdl:part name="fault" element="impl:InvalidEPCEException"/>
3535 </wsdl:message>
3536
3537 <wsdl:message name="InvalidAssocTableEntryExceptionResponse">
3538 <wsdl:part name="fault" element="impl:InvalidAssocTableEntryException"/>
3539 </wsdl:message>
3540
3541 <wsdl:message name="RNGValidationExceptionResponse">
3542 <wsdl:part name="fault" element="impl:RNGValidationException"/>
3543 </wsdl:message>
3544 <!-- ALECCSERVICE PORTTYPE -->
3545
3546 <wsdl:portType name="ALECCServicePortType">
3547
3548 <wsdl:operation name="define">
3549 <wsdl:input message="impl:defineRequest" name="defineRequest"/>
3550 <wsdl:output message="impl:defineResponse" name="defineResponse"/>
3551 <wsdl:fault message="impl:DuplicateNameExceptionResponse"
3552 name="DuplicateNameExceptionFault"/>
3553 <wsdl:fault message="impl:CCSpecValidationExceptionResponse"
3554 name="CCSpecValidationExceptionFault"/>
3555 <wsdl:fault message="impl:SecurityExceptionResponse"
3556 name="SecurityExceptionFault"/>
3557 <wsdl:fault message="impl:ImplementationExceptionResponse"
3558 name="ImplementationExceptionFault"/>
3559 </wsdl:operation>
3560
3561 <wsdl:operation name="undefine">
3562 <wsdl:input message="impl:undefineRequest" name="undefineRequest"/>
3563 <wsdl:output message="impl:undefineResponse" name="undefineResponse"/>
3564 <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3565 name="NoSuchNameExceptionFault"/>
3566 <wsdl:fault message="impl:SecurityExceptionResponse"
3567 name="SecurityExceptionFault"/>
3568 <wsdl:fault message="impl:ImplementationExceptionResponse"
3569 name="ImplementationExceptionFault"/>
3570 </wsdl:operation>
3571
3572 <wsdl:operation name="getCCSpec">
3573 <wsdl:input message="impl:getCCSpecRequest" name="getCCSpecRequest"/>
3574 <wsdl:output message="impl:getCCSpecResponse" name="getCCSpecResponse"/>
3575 <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3576 name="NoSuchNameExceptionFault"/>
3577 <wsdl:fault message="impl:SecurityExceptionResponse"
3578 name="SecurityExceptionFault"/>
3579 <wsdl:fault message="impl:ImplementationExceptionResponse"
3580 name="ImplementationExceptionFault"/>
3581 </wsdl:operation>
3582

```

```

3583 <wsdl:operation name="getCCSpecNames">
3584   <wsdl:input message="impl:getCCSpecNamesRequest" name="getCCSpecNamesRequest" />
3585   <wsdl:output message="impl:getCCSpecNamesResponse" name="getCCSpecNamesResponse" />
3586   <wsdl:fault message="impl:SecurityExceptionResponse"
3587     name="SecurityExceptionFault" />
3588   <wsdl:fault message="impl:ImplementationExceptionResponse"
3589     name="ImplementationExceptionFault" />
3590 </wsdl:operation>
3591
3592 <wsdl:operation name="subscribe">
3593   <wsdl:input message="impl:subscribeRequest" name="subscribeRequest" />
3594   <wsdl:output message="impl:subscribeResponse" name="subscribeResponse" />
3595   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3596     name="NoSuchNameExceptionFault" />
3597   <wsdl:fault message="impl:InvalidURIExceptionResponse"
3598     name="InvalidURIExceptionFault" />
3599   <wsdl:fault message="impl:DuplicateSubscriptionExceptionResponse"
3600     name="DuplicateSubscriptionExceptionFault" />
3601   <wsdl:fault message="impl:ParameterForbiddenExceptionResponse"
3602     name="ParameterForbiddenExceptionFault" />
3603   <wsdl:fault message="impl:SecurityExceptionResponse"
3604     name="SecurityExceptionFault" />
3605   <wsdl:fault message="impl:ImplementationExceptionResponse"
3606     name="ImplementationExceptionFault" />
3607 </wsdl:operation>
3608
3609 <wsdl:operation name="unsubscribe">
3610   <wsdl:input message="impl:unsubscribeRequest" name="unsubscribeRequest" />
3611   <wsdl:output message="impl:unsubscribeResponse" name="unsubscribeResponse" />
3612   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3613     name="NoSuchNameExceptionFault" />
3614   <wsdl:fault message="impl:NoSuchSubscriberExceptionResponse"
3615     name="NoSuchSubscriberExceptionFault" />
3616   <wsdl:fault message="impl:InvalidURIExceptionResponse"
3617     name="InvalidURIExceptionFault" />
3618   <wsdl:fault message="impl:SecurityExceptionResponse"
3619     name="SecurityExceptionFault" />
3620   <wsdl:fault message="impl:ImplementationExceptionResponse"
3621     name="ImplementationExceptionFault" />
3622 </wsdl:operation>
3623
3624 <wsdl:operation name="poll">
3625   <wsdl:input message="impl:pollRequest" name="pollRequest" />
3626   <wsdl:output message="impl:pollResponse" name="pollResponse" />
3627   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3628     name="NoSuchNameExceptionFault" />
3629   <wsdl:fault message="impl:ParameterExceptionResponse"
3630     name="ParameterExceptionFault" />
3631   <wsdl:fault message="impl:SecurityExceptionResponse"
3632     name="SecurityExceptionFault" />
3633   <wsdl:fault message="impl:ImplementationExceptionResponse"
3634     name="ImplementationExceptionFault" />
3635 </wsdl:operation>
3636
3637 <wsdl:operation name="immediate">
3638   <wsdl:input message="impl:immediateRequest" name="immediateRequest" />
3639   <wsdl:output message="impl:immediateResponse" name="immediateResponse" />
3640   <wsdl:fault message="impl:CCSpecValidationExceptionResponse"
3641     name="CCSpecValidationExceptionFault" />
3642   <wsdl:fault message="impl:ParameterForbiddenExceptionResponse"
3643     name="ParameterForbiddenExceptionFault" />
3644   <wsdl:fault message="impl:SecurityExceptionResponse"
3645     name="SecurityExceptionFault" />
3646   <wsdl:fault message="impl:ImplementationExceptionResponse"
3647     name="ImplementationExceptionFault" />
3648 </wsdl:operation>
3649
3650 <wsdl:operation name="getSubscribers">
3651   <wsdl:input message="impl:getSubscribersRequest" name="getSubscribersRequest" />
3652   <wsdl:output message="impl:getSubscribersResponse" name="getSubscribersResponse" />

```

```

3653     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3654         name="NoSuchNameExceptionFault" />
3655     <wsdl:fault message="impl:SecurityExceptionResponse"
3656         name="SecurityExceptionFault" />
3657     <wsdl:fault message="impl:ImplementationExceptionResponse"
3658         name="ImplementationExceptionFault" />
3659 </wsdl:operation>
3660
3661 <wsdl:operation name="getStandardVersion">
3662     <wsdl:input message="impl:getStandardVersionRequest"
3663         name="getStandardVersionRequest" />
3664     <wsdl:output message="impl:getStandardVersionResponse"
3665         name="getStandardVersionResponse" />
3666     <wsdl:fault message="impl:ImplementationExceptionResponse"
3667         name="ImplementationExceptionFault" />
3668 </wsdl:operation>
3669
3670 <wsdl:operation name="getVendorVersion">
3671     <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest" />
3672     <wsdl:output message="impl:getVendorVersionResponse"
3673         name="getVendorVersionResponse" />
3674     <wsdl:fault message="impl:ImplementationExceptionResponse"
3675         name="ImplementationExceptionFault" />
3676 </wsdl:operation>
3677
3678 <wsdl:operation name="defineEPCCache">
3679     <wsdl:input message="impl:defineEPCCacheRequest" name="defineEPCCacheRequest" />
3680     <wsdl:output message="impl:defineEPCCacheResponse" name="defineEPCCacheResponse" />
3681     <wsdl:fault message="impl:DuplicateNameExceptionResponse"
3682         name="DuplicateNameExceptionFault" />
3683     <wsdl:fault message="impl:EPCCacheSpecValidationExceptionResponse"
3684         name="EPCCacheSpecValidationExceptionFault" />
3685     <wsdl:fault message="impl:InvalidPatternExceptionResponse"
3686         name="InvalidPatternExceptionFault" />
3687     <wsdl:fault message="impl:SecurityExceptionResponse"
3688         name="SecurityExceptionFault" />
3689     <wsdl:fault message="impl:ImplementationExceptionResponse"
3690         name="ImplementationExceptionFault" />
3691 </wsdl:operation>
3692
3693 <wsdl:operation name="undefineEPCCache">
3694     <wsdl:input message="impl:undefineEPCCacheRequest" name="undefineEPCCacheRequest" />
3695     <wsdl:output message="impl:undefineEPCCacheResponse"
3696         name="undefineEPCCacheResponse" />
3697     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3698         name="NoSuchNameExceptionFault" />
3699     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
3700     <wsdl:fault message="impl:SecurityExceptionResponse"
3701         name="SecurityExceptionFault" />
3702     <wsdl:fault message="impl:ImplementationExceptionResponse"
3703         name="ImplementationExceptionFault" />
3704 </wsdl:operation>
3705
3706 <wsdl:operation name="getEPCCache">
3707     <wsdl:input message="impl:getEPCCacheRequest" name="getEPCCacheRequest" />
3708     <wsdl:output message="impl:getEPCCacheResponse" name="getEPCCacheResponse" />
3709     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3710         name="NoSuchNameExceptionFault" />
3711     <wsdl:fault message="impl:SecurityExceptionResponse"
3712         name="SecurityExceptionFault" />
3713     <wsdl:fault message="impl:ImplementationExceptionResponse"
3714         name="ImplementationExceptionFault" />
3715 </wsdl:operation>
3716
3717 <wsdl:operation name="getEPCCacheNames">
3718     <wsdl:input message="impl:getEPCCacheNamesRequest" name="getEPCCacheNamesRequest" />
3719     <wsdl:output message="impl:getEPCCacheNamesResponse"
3720         name="getEPCCacheNamesResponse" />
3721     <wsdl:fault message="impl:SecurityExceptionResponse"
3722         name="SecurityExceptionFault" />

```



```

3723     <wsdl:fault message="impl:ImplementationExceptionResponse"
3724         name="ImplementationExceptionFault" />
3725 </wsdl:operation>
3726
3727 <wsdl:operation name="replenishEPCCache">
3728     <wsdl:input message="impl:replenishEPCCacheRequest"
3729         name="replenishEPCCacheRequest" />
3730     <wsdl:output message="impl:replenishEPCCacheResponse"
3731         name="replenishEPCCacheResponse" />
3732     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3733         name="NoSuchNameExceptionFault" />
3734     <wsdl:fault message="impl:InvalidPatternExceptionResponse"
3735         name="InvalidPatternExceptionFault" />
3736     <wsdl:fault message="impl:SecurityExceptionResponse"
3737         name="SecurityExceptionFault" />
3738     <wsdl:fault message="impl:ImplementationExceptionResponse"
3739         name="ImplementationExceptionFault" />
3740 </wsdl:operation>
3741
3742 <wsdl:operation name="depleteEPCCache">
3743     <wsdl:input message="impl:depleteEPCCacheRequest" name="depleteEPCCacheRequest" />
3744     <wsdl:output message="impl:depleteEPCCacheResponse"
3745         name="depleteEPCCacheResponse" />
3746     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3747         name="NoSuchNameExceptionFault" />
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="getEPCCacheContents">
3755     <wsdl:input message="impl:getEPCCacheContentsRequest"
3756         name="getEPCCacheContentsRequest" />
3757     <wsdl:output message="impl:getEPCCacheContentsResponse"
3758         name="getEPCCacheContentsResponse" />
3759     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3760         name="NoSuchNameExceptionFault" />
3761     <wsdl:fault message="impl:SecurityExceptionResponse"
3762         name="SecurityExceptionFault" />
3763     <wsdl:fault message="impl:ImplementationExceptionResponse"
3764         name="ImplementationExceptionFault" />
3765 </wsdl:operation>
3766
3767 <wsdl:operation name="defineAssocTable">
3768     <wsdl:input message="impl:defineAssocTableRequest" name="defineAssocTableRequest" />
3769     <wsdl:output message="impl:defineAssocTableResponse"
3770         name="defineAssocTableResponse" />
3771     <wsdl:fault message="impl:DuplicateNameExceptionResponse"
3772         name="DuplicateNameExceptionFault" />
3773     <wsdl:fault message="impl:AssocTableValidationExceptionResponse"
3774         name="AssocTableValidationExceptionFault" />
3775     <wsdl:fault message="impl:InvalidAssocTableEntryExceptionResponse"
3776         name="InvalidAssocTableEntryExceptionFault" />
3777     <wsdl:fault message="impl:SecurityExceptionResponse"
3778         name="SecurityExceptionFault" />
3779     <wsdl:fault message="impl:ImplementationExceptionResponse"
3780         name="ImplementationExceptionFault" />
3781 </wsdl:operation>
3782
3783 <wsdl:operation name="undefineAssocTable">
3784     <wsdl:input message="impl:undefineAssocTableRequest"
3785         name="undefineAssocTableRequest" />
3786     <wsdl:output message="impl:undefineAssocTableResponse"
3787         name="undefineAssocTableResponse" />
3788     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3789         name="NoSuchNameExceptionFault" />
3790     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
3791     <wsdl:fault message="impl:SecurityExceptionResponse"
3792         name="SecurityExceptionFault" />

```



```

3793     <wsdl:fault message="impl:ImplementationExceptionResponse"
3794             name="ImplementationExceptionFault" />
3795 </wsdl:operation>
3796
3797 <wsdl:operation name="getAssocTableNames">
3798   <wsdl:input message="impl:getAssocTableNamesRequest"
3799             name="getAssocTableNamesRequest" />
3800   <wsdl:output message="impl:getAssocTableNamesResponse"
3801              name="getAssocTableNamesResponse" />
3802   <wsdl:fault message="impl:SecurityExceptionResponse"
3803             name="SecurityExceptionFault" />
3804   <wsdl:fault message="impl:ImplementationExceptionResponse"
3805             name="ImplementationExceptionFault" />
3806 </wsdl:operation>
3807
3808 <wsdl:operation name="getAssocTable">
3809   <wsdl:input message="impl:getAssocTableRequest" name="getAssocTableRequest" />
3810   <wsdl:output message="impl:getAssocTableResponse" name="getAssocTableResponse" />
3811   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3812             name="NoSuchNameExceptionFault" />
3813   <wsdl:fault message="impl:SecurityExceptionResponse"
3814             name="SecurityExceptionFault" />
3815   <wsdl:fault message="impl:ImplementationExceptionResponse"
3816             name="ImplementationExceptionFault" />
3817 </wsdl:operation>
3818
3819 <wsdl:operation name="putAssocTableEntries">
3820   <wsdl:input message="impl:putAssocTableEntriesRequest"
3821             name="putAssocTableEntriesRequest" />
3822   <wsdl:output message="impl:putAssocTableEntriesResponse"
3823             name="putAssocTableEntriesResponse" />
3824   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3825             name="NoSuchNameExceptionFault" />
3826   <wsdl:fault message="impl:InvalidAssocTableEntryExceptionResponse"
3827             name="InvalidAssocTableEntryExceptionFault" />
3828   <wsdl:fault message="impl:SecurityExceptionResponse"
3829             name="SecurityExceptionFault" />
3830   <wsdl:fault message="impl:ImplementationExceptionResponse"
3831             name="ImplementationExceptionFault" />
3832 </wsdl:operation>
3833
3834 <wsdl:operation name="getAssocTableValue">
3835   <wsdl:input message="impl:getAssocTableValueRequest"
3836             name="getAssocTableValueRequest" />
3837   <wsdl:output message="impl:getAssocTableValueResponse"
3838             name="getAssocTableValueResponse" />
3839   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3840             name="NoSuchNameExceptionFault" />
3841   <wsdl:fault message="impl:InvalidEPCEExceptionResponse"
3842             name="InvalidEPCEExceptionFault" />
3843   <wsdl:fault message="impl:SecurityExceptionResponse"
3844             name="SecurityExceptionFault" />
3845   <wsdl:fault message="impl:ImplementationExceptionResponse"
3846             name="ImplementationExceptionFault" />
3847 </wsdl:operation>
3848
3849 <wsdl:operation name="getAssocTableEntries">
3850   <wsdl:input message="impl:getAssocTableEntriesRequest"
3851             name="getAssocTableEntriesRequest" />
3852   <wsdl:output message="impl:getAssocTableEntriesResponse"
3853             name="getAssocTableEntriesResponse" />
3854   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3855             name="NoSuchNameExceptionFault" />
3856   <wsdl:fault message="impl:InvalidPatternExceptionResponse"
3857             name="InvalidPatternExceptionFault" />
3858   <wsdl:fault message="impl:SecurityExceptionResponse"
3859             name="SecurityExceptionFault" />
3860   <wsdl:fault message="impl:ImplementationExceptionResponse"
3861             name="ImplementationExceptionFault" />
3862 </wsdl:operation>

```

```

3863
3864 <wsdl:operation name="removeAssocTableEntry">
3865   <wsdl:input message="impl:removeAssocTableEntryRequest"
3866     name="removeAssocTableEntryRequest" />
3867   <wsdl:output message="impl:removeAssocTableEntryResponse"
3868     name="removeAssocTableEntryResponse" />
3869   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3870     name="NoSuchNameExceptionFault" />
3871   <wsdl:fault message="impl:InvalidEPCEExceptionResponse"
3872     name="InvalidEPCEExceptionFault" />
3873   <wsdl:fault message="impl:SecurityExceptionResponse"
3874     name="SecurityExceptionFault" />
3875   <wsdl:fault message="impl:ImplementationExceptionResponse"
3876     name="ImplementationExceptionFault" />
3877 </wsdl:operation>
3878
3879 <wsdl:operation name="removeAssocTableEntries">
3880   <wsdl:input message="impl:removeAssocTableEntriesRequest"
3881     name="removeAssocTableEntriesRequest" />
3882   <wsdl:output message="impl:removeAssocTableEntriesResponse"
3883     name="removeAssocTableEntriesResponse" />
3884   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3885     name="NoSuchNameExceptionFault" />
3886   <wsdl:fault message="impl:InvalidPatternExceptionResponse"
3887     name="InvalidPatternExceptionFault" />
3888   <wsdl:fault message="impl:SecurityExceptionResponse"
3889     name="SecurityExceptionFault" />
3890   <wsdl:fault message="impl:ImplementationExceptionResponse"
3891     name="ImplementationExceptionFault" />
3892 </wsdl:operation>
3893
3894 <wsdl:operation name="defineRNG">
3895   <wsdl:input message="impl:defineRNGRequest" name="defineRNGRequest" />
3896   <wsdl:output message="impl:defineRNGResponse" name="defineRNGResponse" />
3897   <wsdl:fault message="impl:DuplicateNameExceptionResponse"
3898     name="DuplicateNameExceptionFault" />
3899   <wsdl:fault message="impl:RNGValidationExceptionResponse"
3900     name="RNGValidationExceptionFault" />
3901   <wsdl:fault message="impl:SecurityExceptionResponse"
3902     name="SecurityExceptionFault" />
3903   <wsdl:fault message="impl:ImplementationExceptionResponse"
3904     name="ImplementationExceptionFault" />
3905 </wsdl:operation>
3906
3907 <wsdl:operation name="undefineRNG">
3908   <wsdl:input message="impl:undefineRNGRequest" name="undefineRNGRequest" />
3909   <wsdl:output message="impl:undefineRNGResponse" name="undefineRNGResponse" />
3910   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3911     name="NoSuchNameExceptionFault" />
3912   <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
3913   <wsdl:fault message="impl:SecurityExceptionResponse"
3914     name="SecurityExceptionFault" />
3915   <wsdl:fault message="impl:ImplementationExceptionResponse"
3916     name="ImplementationExceptionFault" />
3917 </wsdl:operation>
3918
3919 <wsdl:operation name="getRNGNames">
3920   <wsdl:input message="impl:getRNGNamesRequest" name="getRNGNamesRequest" />
3921   <wsdl:output message="impl:getRNGNamesResponse" name="getRNGNamesResponse" />
3922   <wsdl:fault message="impl:SecurityExceptionResponse"
3923     name="SecurityExceptionFault" />
3924   <wsdl:fault message="impl:ImplementationExceptionResponse"
3925     name="ImplementationExceptionFault" />
3926 </wsdl:operation>
3927
3928 <wsdl:operation name="getRNG">
3929   <wsdl:input message="impl:getRNGRequest" name="getRNGRequest" />
3930   <wsdl:output message="impl:getRNGResponse" name="getRNGResponse" />
3931   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
3932     name="NoSuchNameExceptionFault" />

```

```

3933     <wsdl:fault message="impl:SecurityExceptionResponse"
3934             name="SecurityExceptionFault"/>
3935     <wsdl:fault message="impl:ImplementationExceptionResponse"
3936             name="ImplementationExceptionFault"/>
3937   </wsdl:operation>
3938 </wsdl:portType>
3939 <!-- ALECCSERVICE BINDING -->
3940
3941 <wsdl:binding name="ALECCServiceBinding" type="impl:ALECCServicePortType">
3942   <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
3943
3944   <wsdl:operation name="define">
3945     <wsdlsoap:operation soapAction=""/>
3946     <wsdl:input name="defineRequest">
3947       <wsdlsoap:body use="literal"/>
3948     </wsdl:input>
3949     <wsdl:output name="defineResponse">
3950       <wsdlsoap:body use="literal"/>
3951     </wsdl:output>
3952     <wsdl:fault name="DuplicateNameExceptionFault">
3953       <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
3954     </wsdl:fault>
3955     <wsdl:fault name="CCSpecValidationExceptionFault">
3956       <wsdlsoap:fault name="CCSpecValidationExceptionFault" use="literal"/>
3957     </wsdl:fault>
3958     <wsdl:fault name="SecurityExceptionFault">
3959       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
3960     </wsdl:fault>
3961     <wsdl:fault name="ImplementationExceptionFault">
3962       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
3963     </wsdl:fault>
3964   </wsdl:operation>
3965
3966   <wsdl:operation name="undefine">
3967     <wsdlsoap:operation soapAction=""/>
3968     <wsdl:input name="undefineRequest">
3969       <wsdlsoap:body use="literal"/>
3970     </wsdl:input>
3971     <wsdl:output name="undefineResponse">
3972       <wsdlsoap:body use="literal"/>
3973     </wsdl:output>
3974     <wsdl:fault name="NoSuchNameExceptionFault">
3975       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
3976     </wsdl:fault>
3977     <wsdl:fault name="SecurityExceptionFault">
3978       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
3979     </wsdl:fault>
3980     <wsdl:fault name="ImplementationExceptionFault">
3981       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
3982     </wsdl:fault>
3983   </wsdl:operation>
3984
3985   <wsdl:operation name="getCCSpec">
3986     <wsdlsoap:operation soapAction=""/>
3987     <wsdl:input name="getCCSpecRequest">
3988       <wsdlsoap:body use="literal"/>
3989     </wsdl:input>
3990     <wsdl:output name="getCCSpecResponse">
3991       <wsdlsoap:body use="literal"/>
3992     </wsdl:output>
3993     <wsdl:fault name="NoSuchNameExceptionFault">
3994       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
3995     </wsdl:fault>
3996     <wsdl:fault name="SecurityExceptionFault">
3997       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
3998     </wsdl:fault>
3999     <wsdl:fault name="ImplementationExceptionFault">
4000       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4001     </wsdl:fault>
4002   </wsdl:operation>

```

```

4003
4004 <wsdl:operation name="getCCSpecNames">
4005   <wsdlsoap:operation soapAction="" />
4006   <wsdl:input name="getCCSpecNamesRequest">
4007     <wsdlsoap:body use="literal" />
4008   </wsdl:input>
4009   <wsdl:output name="getCCSpecNamesResponse">
4010     <wsdlsoap:body use="literal" />
4011   </wsdl:output>
4012   <wsdl:fault name="SecurityExceptionFault">
4013     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4014   </wsdl:fault>
4015   <wsdl:fault name="ImplementationExceptionFault">
4016     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4017   </wsdl:fault>
4018 </wsdl:operation>
4019
4020 <wsdl:operation name="subscribe">
4021   <wsdlsoap:operation soapAction="" />
4022   <wsdl:input name="subscribeRequest">
4023     <wsdlsoap:body use="literal" />
4024   </wsdl:input>
4025   <wsdl:output name="subscribeResponse">
4026     <wsdlsoap:body use="literal" />
4027   </wsdl:output>
4028   <wsdl:fault name="NoSuchNameExceptionFault">
4029     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4030   </wsdl:fault>
4031   <wsdl:fault name="InvalidURIExceptionFault">
4032     <wsdlsoap:fault name="InvalidURIExceptionFault" use="literal" />
4033   </wsdl:fault>
4034   <wsdl:fault name="DuplicateSubscriptionExceptionFault">
4035     <wsdlsoap:fault name="DuplicateSubscriptionExceptionFault" use="literal" />
4036   </wsdl:fault>
4037   <wsdl:fault name="ParameterForbiddenExceptionFault">
4038     <wsdlsoap:fault name="ParameterForbiddenExceptionFault" use="literal" />
4039   </wsdl:fault>
4040   <wsdl:fault name="SecurityExceptionFault">
4041     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4042   </wsdl:fault>
4043   <wsdl:fault name="ImplementationExceptionFault">
4044     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4045   </wsdl:fault>
4046 </wsdl:operation>
4047
4048 <wsdl:operation name="unsubscribe">
4049   <wsdlsoap:operation soapAction="" />
4050   <wsdl:input name="unsubscribeRequest">
4051     <wsdlsoap:body use="literal" />
4052   </wsdl:input>
4053   <wsdl:output name="unsubscribeResponse">
4054     <wsdlsoap:body use="literal" />
4055   </wsdl:output>
4056   <wsdl:fault name="NoSuchNameExceptionFault">
4057     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4058   </wsdl:fault>
4059   <wsdl:fault name="NoSuchSubscriberExceptionFault">
4060     <wsdlsoap:fault name="NoSuchSubscriberExceptionFault" use="literal" />
4061   </wsdl:fault>
4062   <wsdl:fault name="InvalidURIExceptionFault">
4063     <wsdlsoap:fault name="InvalidURIExceptionFault" use="literal" />
4064   </wsdl:fault>
4065   <wsdl:fault name="SecurityExceptionFault">
4066     <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4067   </wsdl:fault>
4068   <wsdl:fault name="ImplementationExceptionFault">
4069     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4070   </wsdl:fault>
4071 </wsdl:operation>
4072

```

```

4073 <wsdl:operation name="poll">
4074 <wsdlsoap:operation soapAction=""/>
4075 <wsdl:input name="pollRequest">
4076 <wsdlsoap:body use="literal"/>
4077 </wsdl:input>
4078 <wsdl:output name="pollResponse">
4079 <wsdlsoap:body use="literal"/>
4080 </wsdl:output>
4081 <wsdl:fault name="NoSuchNameExceptionFault">
4082 <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4083 </wsdl:fault>
4084 <wsdl:fault name="ParameterExceptionFault">
4085 <wsdlsoap:fault name="ParameterExceptionFault" use="literal"/>
4086 </wsdl:fault>
4087 <wsdl:fault name="SecurityExceptionFault">
4088 <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4089 </wsdl:fault>
4090 <wsdl:fault name="ImplementationExceptionFault">
4091 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4092 </wsdl:fault>
4093 </wsdl:operation>
4094
4095 <wsdl:operation name="immediate">
4096 <wsdlsoap:operation soapAction=""/>
4097 <wsdl:input name="immediateRequest">
4098 <wsdlsoap:body use="literal"/>
4099 </wsdl:input>
4100 <wsdl:output name="immediateResponse">
4101 <wsdlsoap:body use="literal"/>
4102 </wsdl:output>
4103 <wsdl:fault name="CCSpecValidationExceptionFault">
4104 <wsdlsoap:fault name="CCSpecValidationExceptionFault" use="literal"/>
4105 </wsdl:fault>
4106 <wsdl:fault name="ParameterForbiddenExceptionFault">
4107 <wsdlsoap:fault name="ParameterForbiddenExceptionFault" use="literal"/>
4108 </wsdl:fault>
4109 <wsdl:fault name="SecurityExceptionFault">
4110 <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4111 </wsdl:fault>
4112 <wsdl:fault name="ImplementationExceptionFault">
4113 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4114 </wsdl:fault>
4115 </wsdl:operation>
4116
4117 <wsdl:operation name="getSubscribers">
4118 <wsdlsoap:operation soapAction=""/>
4119 <wsdl:input name="getSubscribersRequest">
4120 <wsdlsoap:body use="literal"/>
4121 </wsdl:input>
4122 <wsdl:output name="getSubscribersResponse">
4123 <wsdlsoap:body use="literal"/>
4124 </wsdl:output>
4125 <wsdl:fault name="NoSuchNameExceptionFault">
4126 <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4127 </wsdl:fault>
4128 <wsdl:fault name="SecurityExceptionFault">
4129 <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4130 </wsdl:fault>
4131 <wsdl:fault name="ImplementationExceptionFault">
4132 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4133 </wsdl:fault>
4134 </wsdl:operation>
4135
4136 <wsdl:operation name="getStandardVersion">
4137 <wsdlsoap:operation soapAction=""/>
4138 <wsdl:input name="getStandardVersionRequest">
4139 <wsdlsoap:body use="literal"/>
4140 </wsdl:input>
4141 <wsdl:output name="getStandardVersionResponse">
4142 <wsdlsoap:body use="literal"/>

```

```

4143     </wsdl:output>
4144     <wsdl:fault name="ImplementationExceptionFault">
4145         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4146     </wsdl:fault>
4147 </wsdl:operation>
4148
4149 <wsdl:operation name="getVendorVersion">
4150     <wsdlsoap:operation soapAction=""/>
4151     <wsdl:input name="getVendorVersionRequest">
4152         <wsdlsoap:body use="literal"/>
4153     </wsdl:input>
4154     <wsdl:output name="getVendorVersionResponse">
4155         <wsdlsoap:body use="literal"/>
4156     </wsdl:output>
4157     <wsdl:fault name="ImplementationExceptionFault">
4158         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4159     </wsdl:fault>
4160 </wsdl:operation>
4161
4162 <wsdl:operation name="defineEPCCache">
4163     <wsdlsoap:operation soapAction=""/>
4164     <wsdl:input name="defineEPCCacheRequest">
4165         <wsdlsoap:body use="literal"/>
4166     </wsdl:input>
4167     <wsdl:output name="defineEPCCacheResponse">
4168         <wsdlsoap:body use="literal"/>
4169     </wsdl:output>
4170     <wsdl:fault name="DuplicateNameExceptionFault">
4171         <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
4172     </wsdl:fault>
4173     <wsdl:fault name="EPCCacheSpecValidationExceptionFault">
4174         <wsdlsoap:fault name="EPCCacheSpecValidationExceptionFault" use="literal"/>
4175     </wsdl:fault>
4176     <wsdl:fault name="InvalidPatternExceptionFault">
4177         <wsdlsoap:fault name="InvalidPatternExceptionFault" use="literal"/>
4178     </wsdl:fault>
4179     <wsdl:fault name="SecurityExceptionFault">
4180         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4181     </wsdl:fault>
4182     <wsdl:fault name="ImplementationExceptionFault">
4183         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4184     </wsdl:fault>
4185 </wsdl:operation>
4186
4187 <wsdl:operation name="undefineEPCCache">
4188     <wsdlsoap:operation soapAction=""/>
4189     <wsdl:input name="undefineEPCCacheRequest">
4190         <wsdlsoap:body use="literal"/>
4191     </wsdl:input>
4192     <wsdl:output name="undefineEPCCacheResponse">
4193         <wsdlsoap:body use="literal"/>
4194     </wsdl:output>
4195     <wsdl:fault name="NoSuchNameExceptionFault">
4196         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4197     </wsdl:fault>
4198     <wsdl:fault name="InUseExceptionFault">
4199         <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
4200     </wsdl:fault>
4201     <wsdl:fault name="SecurityExceptionFault">
4202         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4203     </wsdl:fault>
4204     <wsdl:fault name="ImplementationExceptionFault">
4205         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4206     </wsdl:fault>
4207 </wsdl:operation>
4208
4209 <wsdl:operation name="getEPCCache">
4210     <wsdlsoap:operation soapAction=""/>
4211     <wsdl:input name="getEPCCacheRequest">
4212         <wsdlsoap:body use="literal"/>

```



```

4213     </wsdl:input>
4214     <wsdl:output name="getEPCCacheResponse">
4215         <wsdlsoap:body use="literal"/>
4216     </wsdl:output>
4217     <wsdl:fault name="NoSuchNameExceptionFault">
4218         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4219     </wsdl:fault>
4220     <wsdl:fault name="SecurityExceptionFault">
4221         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4222     </wsdl:fault>
4223     <wsdl:fault name="ImplementationExceptionFault">
4224         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4225     </wsdl:fault>
4226 </wsdl:operation>
4227
4228 <wsdl:operation name="getEPCCacheNames">
4229     <wsdlsoap:operation soapAction=""/>
4230     <wsdl:input name="getEPCCacheNamesRequest">
4231         <wsdlsoap:body use="literal"/>
4232     </wsdl:input>
4233     <wsdl:output name="getEPCCacheNamesResponse">
4234         <wsdlsoap:body use="literal"/>
4235     </wsdl:output>
4236     <wsdl:fault name="SecurityExceptionFault">
4237         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4238     </wsdl:fault>
4239     <wsdl:fault name="ImplementationExceptionFault">
4240         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4241     </wsdl:fault>
4242 </wsdl:operation>
4243
4244 <wsdl:operation name="replenishEPCCache">
4245     <wsdlsoap:operation soapAction=""/>
4246     <wsdl:input name="replenishEPCCacheRequest">
4247         <wsdlsoap:body use="literal"/>
4248     </wsdl:input>
4249     <wsdl:output name="replenishEPCCacheResponse">
4250         <wsdlsoap:body use="literal"/>
4251     </wsdl:output>
4252     <wsdl:fault name="NoSuchNameExceptionFault">
4253         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4254     </wsdl:fault>
4255     <wsdl:fault name="InvalidPatternExceptionFault">
4256         <wsdlsoap:fault name="InvalidPatternExceptionFault" use="literal"/>
4257     </wsdl:fault>
4258     <wsdl:fault name="SecurityExceptionFault">
4259         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4260     </wsdl:fault>
4261     <wsdl:fault name="ImplementationExceptionFault">
4262         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4263     </wsdl:fault>
4264 </wsdl:operation>
4265
4266 <wsdl:operation name="depleteEPCCache">
4267     <wsdlsoap:operation soapAction=""/>
4268     <wsdl:input name="depleteEPCCacheRequest">
4269         <wsdlsoap:body use="literal"/>
4270     </wsdl:input>
4271     <wsdl:output name="depleteEPCCacheResponse">
4272         <wsdlsoap:body use="literal"/>
4273     </wsdl:output>
4274     <wsdl:fault name="NoSuchNameExceptionFault">
4275         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4276     </wsdl:fault>
4277     <wsdl:fault name="SecurityExceptionFault">
4278         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4279     </wsdl:fault>
4280     <wsdl:fault name="ImplementationExceptionFault">
4281         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4282     </wsdl:fault>

```



```

4283 </wsdl:operation>
4284
4285 <wsdl:operation name="getEPCCacheContents">
4286 <wsdlsoap:operation soapAction=""/>
4287 <wsdl:input name="getEPCCacheContentsRequest">
4288 <wsdlsoap:body use="literal"/>
4289 </wsdl:input>
4290 <wsdl:output name="getEPCCacheContentsResponse">
4291 <wsdlsoap:body use="literal"/>
4292 </wsdl:output>
4293 <wsdl:fault name="NoSuchNameExceptionFault">
4294 <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4295 </wsdl:fault>
4296 <wsdl:fault name="SecurityExceptionFault">
4297 <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4298 </wsdl:fault>
4299 <wsdl:fault name="ImplementationExceptionFault">
4300 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4301 </wsdl:fault>
4302 </wsdl:operation>
4303
4304 <wsdl:operation name="defineAssocTable">
4305 <wsdlsoap:operation soapAction=""/>
4306 <wsdl:input name="defineAssocTableRequest">
4307 <wsdlsoap:body use="literal"/>
4308 </wsdl:input>
4309 <wsdl:output name="defineAssocTableResponse">
4310 <wsdlsoap:body use="literal"/>
4311 </wsdl:output>
4312 <wsdl:fault name="DuplicateNameExceptionFault">
4313 <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
4314 </wsdl:fault>
4315 <wsdl:fault name="AssocTableValidationExceptionFault">
4316 <wsdlsoap:fault name="AssocTableValidationExceptionFault" use="literal"/>
4317 </wsdl:fault>
4318 <wsdl:fault name="InvalidAssocTableEntryExceptionFault">
4319 <wsdlsoap:fault name="InvalidAssocTableEntryExceptionFault" use="literal"/>
4320 </wsdl:fault>
4321 <wsdl:fault name="SecurityExceptionFault">
4322 <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4323 </wsdl:fault>
4324 <wsdl:fault name="ImplementationExceptionFault">
4325 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4326 </wsdl:fault>
4327 </wsdl:operation>
4328
4329 <wsdl:operation name="undefineAssocTable">
4330 <wsdlsoap:operation soapAction=""/>
4331 <wsdl:input name="undefineAssocTableRequest">
4332 <wsdlsoap:body use="literal"/>
4333 </wsdl:input>
4334 <wsdl:output name="undefineAssocTableResponse">
4335 <wsdlsoap:body use="literal"/>
4336 </wsdl:output>
4337 <wsdl:fault name="NoSuchNameExceptionFault">
4338 <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4339 </wsdl:fault>
4340 <wsdl:fault name="InUseExceptionFault">
4341 <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
4342 </wsdl:fault>
4343 <wsdl:fault name="SecurityExceptionFault">
4344 <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4345 </wsdl:fault>
4346 <wsdl:fault name="ImplementationExceptionFault">
4347 <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4348 </wsdl:fault>
4349 </wsdl:operation>
4350
4351 <wsdl:operation name="getAssocTableNames">
4352 <wsdlsoap:operation soapAction=""/>

```

```

4353     <wsdl:input name="getAssocTableNamesRequest">
4354         <wsdlsoap:body use="literal"/>
4355     </wsdl:input>
4356     <wsdl:output name="getAssocTableNamesResponse">
4357         <wsdlsoap:body use="literal"/>
4358     </wsdl:output>
4359     <wsdl:fault name="SecurityExceptionFault">
4360         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4361     </wsdl:fault>
4362     <wsdl:fault name="ImplementationExceptionFault">
4363         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4364     </wsdl:fault>
4365 </wsdl:operation>
4366
4367 <wsdl:operation name="getAssocTable">
4368     <wsdlsoap:operation soapAction=""/>
4369     <wsdl:input name="getAssocTableRequest">
4370         <wsdlsoap:body use="literal"/>
4371     </wsdl:input>
4372     <wsdl:output name="getAssocTableResponse">
4373         <wsdlsoap:body use="literal"/>
4374     </wsdl:output>
4375     <wsdl:fault name="NoSuchNameExceptionFault">
4376         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4377     </wsdl:fault>
4378     <wsdl:fault name="SecurityExceptionFault">
4379         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4380     </wsdl:fault>
4381     <wsdl:fault name="ImplementationExceptionFault">
4382         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4383     </wsdl:fault>
4384 </wsdl:operation>
4385
4386 <wsdl:operation name="putAssocTableEntries">
4387     <wsdlsoap:operation soapAction=""/>
4388     <wsdl:input name="putAssocTableEntriesRequest">
4389         <wsdlsoap:body use="literal"/>
4390     </wsdl:input>
4391     <wsdl:output name="putAssocTableEntriesResponse">
4392         <wsdlsoap:body use="literal"/>
4393     </wsdl:output>
4394     <wsdl:fault name="NoSuchNameExceptionFault">
4395         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4396     </wsdl:fault>
4397     <wsdl:fault name="InvalidAssocTableEntryExceptionFault">
4398         <wsdlsoap:fault name="InvalidAssocTableEntryExceptionFault" use="literal"/>
4399     </wsdl:fault>
4400     <wsdl:fault name="SecurityExceptionFault">
4401         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4402     </wsdl:fault>
4403     <wsdl:fault name="ImplementationExceptionFault">
4404         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4405     </wsdl:fault>
4406 </wsdl:operation>
4407
4408 <wsdl:operation name="getAssocTableValue">
4409     <wsdlsoap:operation soapAction=""/>
4410     <wsdl:input name="getAssocTableValueRequest">
4411         <wsdlsoap:body use="literal"/>
4412     </wsdl:input>
4413     <wsdl:output name="getAssocTableValueResponse">
4414         <wsdlsoap:body use="literal"/>
4415     </wsdl:output>
4416     <wsdl:fault name="NoSuchNameExceptionFault">
4417         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4418     </wsdl:fault>
4419     <wsdl:fault name="InvalidEPCEExceptionFault">
4420         <wsdlsoap:fault name="InvalidEPCEExceptionFault" use="literal"/>
4421     </wsdl:fault>
4422     <wsdl:fault name="SecurityExceptionFault">

```

```

4423     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4424 </wsdl:fault>
4425 <wsdl:fault name="ImplementationExceptionFault">
4426     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4427 </wsdl:fault>
4428 </wsdl:operation>
4429
4430 <wsdl:operation name="getAssocTableEntries">
4431     <wsdlsoap:operation soapAction=""/>
4432     <wsdl:input name="getAssocTableEntriesRequest">
4433         <wsdlsoap:body use="literal"/>
4434     </wsdl:input>
4435     <wsdl:output name="getAssocTableEntriesResponse">
4436         <wsdlsoap:body use="literal"/>
4437     </wsdl:output>
4438     <wsdl:fault name="NoSuchNameExceptionFault">
4439         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4440     </wsdl:fault>
4441     <wsdl:fault name="InvalidPatternExceptionFault">
4442         <wsdlsoap:fault name="InvalidPatternExceptionFault" use="literal"/>
4443     </wsdl:fault>
4444     <wsdl:fault name="SecurityExceptionFault">
4445         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4446     </wsdl:fault>
4447     <wsdl:fault name="ImplementationExceptionFault">
4448         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4449     </wsdl:fault>
4450 </wsdl:operation>
4451
4452 <wsdl:operation name="removeAssocTableEntry">
4453     <wsdlsoap:operation soapAction=""/>
4454     <wsdl:input name="removeAssocTableEntryRequest">
4455         <wsdlsoap:body use="literal"/>
4456     </wsdl:input>
4457     <wsdl:output name="removeAssocTableEntryResponse">
4458         <wsdlsoap:body use="literal"/>
4459     </wsdl:output>
4460     <wsdl:fault name="NoSuchNameExceptionFault">
4461         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4462     </wsdl:fault>
4463     <wsdl:fault name="InvalidEPCEExceptionFault">
4464         <wsdlsoap:fault name="InvalidEPCEExceptionFault" use="literal"/>
4465     </wsdl:fault>
4466     <wsdl:fault name="SecurityExceptionFault">
4467         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4468     </wsdl:fault>
4469     <wsdl:fault name="ImplementationExceptionFault">
4470         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4471     </wsdl:fault>
4472 </wsdl:operation>
4473
4474 <wsdl:operation name="removeAssocTableEntries">
4475     <wsdlsoap:operation soapAction=""/>
4476     <wsdl:input name="removeAssocTableEntriesRequest">
4477         <wsdlsoap:body use="literal"/>
4478     </wsdl:input>
4479     <wsdl:output name="removeAssocTableEntriesResponse">
4480         <wsdlsoap:body use="literal"/>
4481     </wsdl:output>
4482     <wsdl:fault name="NoSuchNameExceptionFault">
4483         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4484     </wsdl:fault>
4485     <wsdl:fault name="InvalidPatternExceptionFault">
4486         <wsdlsoap:fault name="InvalidPatternExceptionFault" use="literal"/>
4487     </wsdl:fault>
4488     <wsdl:fault name="SecurityExceptionFault">
4489         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4490     </wsdl:fault>
4491     <wsdl:fault name="ImplementationExceptionFault">
4492         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>

```

```

4493     </wsdl:fault>
4494 </wsdl:operation>
4495
4496 <wsdl:operation name="defineRNG">
4497   <wsdlsoap:operation soapAction=""/>
4498   <wsdl:input name="defineRNGRequest">
4499     <wsdlsoap:body use="literal"/>
4500   </wsdl:input>
4501   <wsdl:output name="defineRNGResponse">
4502     <wsdlsoap:body use="literal"/>
4503   </wsdl:output>
4504   <wsdl:fault name="DuplicateNameExceptionFault">
4505     <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
4506   </wsdl:fault>
4507   <wsdl:fault name="RNGValidationExceptionFault">
4508     <wsdlsoap:fault name="RNGValidationExceptionFault" use="literal"/>
4509   </wsdl:fault>
4510   <wsdl:fault name="SecurityExceptionFault">
4511     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4512   </wsdl:fault>
4513   <wsdl:fault name="ImplementationExceptionFault">
4514     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4515   </wsdl:fault>
4516 </wsdl:operation>
4517
4518 <wsdl:operation name="undefineRNG">
4519   <wsdlsoap:operation soapAction=""/>
4520   <wsdl:input name="undefineRNGRequest">
4521     <wsdlsoap:body use="literal"/>
4522   </wsdl:input>
4523   <wsdl:output name="undefineRNGResponse">
4524     <wsdlsoap:body use="literal"/>
4525   </wsdl:output>
4526   <wsdl:fault name="NoSuchNameExceptionFault">
4527     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4528   </wsdl:fault>
4529   <wsdl:fault name="InUseExceptionFault">
4530     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
4531   </wsdl:fault>
4532   <wsdl:fault name="SecurityExceptionFault">
4533     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4534   </wsdl:fault>
4535   <wsdl:fault name="ImplementationExceptionFault">
4536     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4537   </wsdl:fault>
4538 </wsdl:operation>
4539
4540 <wsdl:operation name="getRNGNames">
4541   <wsdlsoap:operation soapAction=""/>
4542   <wsdl:input name="getRNGNamesRequest">
4543     <wsdlsoap:body use="literal"/>
4544   </wsdl:input>
4545   <wsdl:output name="getRNGNamesResponse">
4546     <wsdlsoap:body use="literal"/>
4547   </wsdl:output>
4548   <wsdl:fault name="SecurityExceptionFault">
4549     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4550   </wsdl:fault>
4551   <wsdl:fault name="ImplementationExceptionFault">
4552     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4553   </wsdl:fault>
4554 </wsdl:operation>
4555
4556 <wsdl:operation name="getRNG">
4557   <wsdlsoap:operation soapAction=""/>
4558   <wsdl:input name="getRNGRequest">
4559     <wsdlsoap:body use="literal"/>
4560   </wsdl:input>
4561   <wsdl:output name="getRNGResponse">
4562     <wsdlsoap:body use="literal"/>

```

```

4563     </wsdl:output>
4564     <wsdl:fault name="NoSuchNameExceptionFault">
4565         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
4566     </wsdl:fault>
4567     <wsdl:fault name="SecurityExceptionFault">
4568         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4569     </wsdl:fault>
4570     <wsdl:fault name="ImplementationExceptionFault">
4571         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4572     </wsdl:fault>
4573 </wsdl:operation>
4574 </wsdl:binding>
4575
4576 <!-- ALECCSERVICE -->
4577 <wsdl:service name="ALECCService">
4578     <wsdl:port binding="impl:ALECCServiceBinding" name="ALECCServicePort">
4579         <!-- The value of the location attribute below is an example only;
4580             Implementations are free to choose any appropriate URL. -->
4581         <wsdlsoap:address location="http://localhost:8080/services/ALECCService"/>
4582     </wsdl:port>
4583 </wsdl:service>
4584 </wsdl:definitions>

```

## 4585 4.5 ALE Tag Memory API SOAP Binding

4586 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
4587 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Tag  
4588 Memory API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0  
4589 [WSI].

```

4590 <?xml version="1.0" encoding="UTF-8"?>
4591 <!-- ALETMSERVICE DEFINITIONS -->
4592 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4593     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
4594     xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
4595     xmlns:ale="urn:epcglobal:ale:xsd:1"
4596     xmlns:impl="urn:epcglobal:aletm:wsdl:1"
4597     targetNamespace="urn:epcglobal:aletm:wsdl:1">
4598 <!-- ALETMSERVICE TYPES -->
4599 <wsdl:types>
4600     <xsd:schema targetNamespace="urn:epcglobal:aletm:wsdl:1">
4601         <xsd:import namespace="urn:epcglobal:ale:xsd:1"
4602             schemaLocation="EPCglobal-ale-1_1-aletm.xsd"/>
4603         <!-- ALETMSERVICE MESSAGE WRAPPERS -->
4604
4605         <xsd:element name="DefineTMSpec" type="impl:DefineTMSpec"/>
4606         <xsd:complexType name="DefineTMSpec">
4607             <xsd:sequence>
4608                 <xsd:element name="specName" type="xsd:string"/>
4609                 <xsd:element name="spec" type="ale:TMSpec"/>
4610             </xsd:sequence>
4611         </xsd:complexType>
4612         <xsd:element name="DefineTMSpecResult">
4613             <xsd:complexType/>
4614         </xsd:element>
4615
4616         <xsd:element name="UndefineTMSpec" type="impl:UndefineTMSpec"/>
4617         <xsd:complexType name="UndefineTMSpec">
4618             <xsd:sequence>
4619                 <xsd:element name="specName" type="xsd:string"/>
4620             </xsd:sequence>
4621         </xsd:complexType>
4622         <xsd:element name="UndefineTMSpecResult">
4623             <xsd:complexType/>
4624         </xsd:element>
4625
4626         <xsd:element name="GetTMSpec" type="impl:GetTMSpec"/>
4627         <xsd:complexType name="GetTMSpec">

```

```

4628     <xsd:sequence>
4629         <xsd:element name="specName" type="xsd:string"/>
4630     </xsd:sequence>
4631 </xsd:complexType>
4632 <xsd:element name="GetTMSpecResult" type="ale:TMSpec"/>
4633
4634 <xsd:element name="GetTMSpecNames" type="impl:EmptyParms"/>
4635 <xsd:element name="GetTMSpecNamesResult" type="impl:ArrayOfString"/>
4636
4637 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
4638 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
4639
4640 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
4641 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
4642
4643 <xsd:element name="ALEException" type="impl:ALEException"/>
4644 <xsd:complexType name="ALEException">
4645     <xsd:sequence>
4646         <xsd:element name="reason" type="xsd:string"/>
4647     </xsd:sequence>
4648 </xsd:complexType>
4649
4650 <xsd:element name="SecurityException" type="impl:SecurityException"/>
4651 <xsd:complexType name="SecurityException">
4652     <xsd:complexContent>
4653         <xsd:extension base="impl:ALEException"/>
4654     </xsd:complexContent>
4655 </xsd:complexType>
4656
4657 <xsd:element name="DuplicateNameException" type="impl:DuplicateNameException"/>
4658 <xsd:complexType name="DuplicateNameException">
4659     <xsd:complexContent>
4660         <xsd:extension base="impl:ALEException"/>
4661     </xsd:complexContent>
4662 </xsd:complexType>
4663
4664 <xsd:element name="TMSpecValidationException"
4665     type="impl:TMSpecValidationException"/>
4666 <xsd:complexType name="TMSpecValidationException">
4667     <xsd:complexContent>
4668         <xsd:extension base="impl:ALEException"/>
4669     </xsd:complexContent>
4670 </xsd:complexType>
4671
4672 <xsd:element name="NoSuchNameException" type="impl:NoSuchNameException"/>
4673 <xsd:complexType name="NoSuchNameException">
4674     <xsd:complexContent>
4675         <xsd:extension base="impl:ALEException"/>
4676     </xsd:complexContent>
4677 </xsd:complexType>
4678
4679 <xsd:element name="InUseException" type="impl:InUseException"/>
4680 <xsd:complexType name="InUseException">
4681     <xsd:complexContent>
4682         <xsd:extension base="impl:ALEException"/>
4683     </xsd:complexContent>
4684 </xsd:complexType>
4685
4686 <xsd:element name="ImplementationException" type="impl:ImplementationException"/>
4687 <xsd:complexType name="ImplementationException">
4688     <xsd:complexContent>
4689         <xsd:extension base="impl:ALEException">
4690             <xsd:sequence>
4691                 <xsd:element name="severity" type="impl:ImplementationExceptionSeverity"/>
4692             </xsd:sequence>
4693         </xsd:extension>
4694     </xsd:complexContent>
4695 </xsd:complexType>
4696
4697 <xsd:complexType name="ArrayOfString">

```



```

4698     <xsd:sequence>
4699         <xsd:element name="string" type="xsd:string" minOccurs="0"
4700             maxOccurs="unbounded" />
4701     </xsd:sequence>
4702 </xsd:complexType>
4703
4704 <!-- The ImplementationExceptionSeverity type is an enumerated type.
4705     The following strings are legal values for this type:
4706         ERROR
4707         SEVERE
4708     -->
4709 <xsd:simpleType name="ImplementationExceptionSeverity">
4710     <xsd:restriction base="xsd:string" />
4711 </xsd:simpleType>
4712
4713     <xsd:complexType name="EmptyParms" />
4714 </xsd:schema>
4715 </wsdl:types>
4716 <!-- ALETMSERVICE MESSAGES -->
4717
4718 <wsdl:message name="defineTMSpecRequest">
4719     <wsdl:part name="parms" element="impl:DefineTMSpec" />
4720 </wsdl:message>
4721 <wsdl:message name="defineTMSpecResponse">
4722     <wsdl:part name="defineTMSpecReturn" element="impl:DefineTMSpecResult" />
4723 </wsdl:message>
4724
4725 <wsdl:message name="undefineTMSpecRequest">
4726     <wsdl:part name="parms" element="impl:UndefineTMSpec" />
4727 </wsdl:message>
4728 <wsdl:message name="undefineTMSpecResponse">
4729     <wsdl:part name="undefineTMSpecReturn" element="impl:UndefineTMSpecResult" />
4730 </wsdl:message>
4731
4732 <wsdl:message name="getTMSpecRequest">
4733     <wsdl:part name="parms" element="impl:GetTMSpec" />
4734 </wsdl:message>
4735 <wsdl:message name="getTMSpecResponse">
4736     <wsdl:part name="getTMSpecReturn" element="impl:GetTMSpecResult" />
4737 </wsdl:message>
4738
4739 <wsdl:message name="getTMSpecNamesRequest">
4740     <wsdl:part name="parms" element="impl:GetTMSpecNames" />
4741 </wsdl:message>
4742 <wsdl:message name="getTMSpecNamesResponse">
4743     <wsdl:part name="getTMSpecNamesReturn" element="impl:GetTMSpecNamesResult" />
4744 </wsdl:message>
4745
4746 <wsdl:message name="getStandardVersionRequest">
4747     <wsdl:part name="parms" element="impl:GetStandardVersion" />
4748 </wsdl:message>
4749 <wsdl:message name="getStandardVersionResponse">
4750     <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult" />
4751 </wsdl:message>
4752
4753 <wsdl:message name="getVendorVersionRequest">
4754     <wsdl:part name="parms" element="impl:GetVendorVersion" />
4755 </wsdl:message>
4756 <wsdl:message name="getVendorVersionResponse">
4757     <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult" />
4758 </wsdl:message>
4759
4760 <wsdl:message name="SecurityExceptionResponse">
4761     <wsdl:part name="fault" element="impl:SecurityException" />
4762 </wsdl:message>
4763
4764 <wsdl:message name="DuplicateNameExceptionResponse">
4765     <wsdl:part name="fault" element="impl:DuplicateNameException" />
4766 </wsdl:message>
4767

```



```

4768 <wsdl:message name="InUseExceptionResponse">
4769   <wsdl:part name="fault" element="impl:InUseException"/>
4770 </wsdl:message>
4771
4772 <wsdl:message name="TMSpecValidationExceptionResponse">
4773   <wsdl:part name="fault" element="impl:TMSpecValidationException"/>
4774 </wsdl:message>
4775
4776 <wsdl:message name="NoSuchNameExceptionResponse">
4777   <wsdl:part name="fault" element="impl:NoSuchNameException"/>
4778 </wsdl:message>
4779
4780 <wsdl:message name="ImplementationExceptionResponse">
4781   <wsdl:part name="fault" element="impl:ImplementationException"/>
4782 </wsdl:message>
4783 <!-- ALETMSERVICE PORTTYPE -->
4784
4785 <wsdl:portType name="ALETMSERVICEPortType">
4786
4787   <wsdl:operation name="defineTMSpec">
4788     <wsdl:input message="impl:defineTMSpecRequest" name="defineTMSpecRequest"/>
4789     <wsdl:output message="impl:defineTMSpecResponse" name="defineTMSpecResponse"/>
4790     <wsdl:fault message="impl:DuplicateNameExceptionResponse"
4791       name="DuplicateNameExceptionFault"/>
4792     <wsdl:fault message="impl:TMSpecValidationExceptionResponse"
4793       name="TMSpecValidationExceptionFault"/>
4794     <wsdl:fault message="impl:SecurityExceptionResponse"
4795       name="SecurityExceptionFault"/>
4796     <wsdl:fault message="impl:ImplementationExceptionResponse"
4797       name="ImplementationExceptionFault"/>
4798   </wsdl:operation>
4799
4800   <wsdl:operation name="undefineTMSpec">
4801     <wsdl:input message="impl:undefineTMSpecRequest" name="undefineTMSpecRequest"/>
4802     <wsdl:output message="impl:undefineTMSpecResponse" name="undefineTMSpecResponse"/>
4803     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
4804       name="NoSuchNameExceptionFault"/>
4805     <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault"/>
4806     <wsdl:fault message="impl:SecurityExceptionResponse"
4807       name="SecurityExceptionFault"/>
4808     <wsdl:fault message="impl:ImplementationExceptionResponse"
4809       name="ImplementationExceptionFault"/>
4810   </wsdl:operation>
4811
4812   <wsdl:operation name="getTMSpec">
4813     <wsdl:input message="impl:getTMSpecRequest" name="getTMSpecRequest"/>
4814     <wsdl:output message="impl:getTMSpecResponse" name="getTMSpecResponse"/>
4815     <wsdl:fault message="impl:NoSuchNameExceptionResponse"
4816       name="NoSuchNameExceptionFault"/>
4817     <wsdl:fault message="impl:SecurityExceptionResponse"
4818       name="SecurityExceptionFault"/>
4819     <wsdl:fault message="impl:ImplementationExceptionResponse"
4820       name="ImplementationExceptionFault"/>
4821   </wsdl:operation>
4822
4823   <wsdl:operation name="getTMSpecNames">
4824     <wsdl:input message="impl:getTMSpecNamesRequest" name="getTMSpecNamesRequest"/>
4825     <wsdl:output message="impl:getTMSpecNamesResponse" name="getTMSpecNamesResponse"/>
4826     <wsdl:fault message="impl:SecurityExceptionResponse"
4827       name="SecurityExceptionFault"/>
4828     <wsdl:fault message="impl:ImplementationExceptionResponse"
4829       name="ImplementationExceptionFault"/>
4830   </wsdl:operation>
4831
4832   <wsdl:operation name="getStandardVersion">
4833     <wsdl:input message="impl:getStandardVersionRequest"
4834       name="getStandardVersionRequest"/>
4835     <wsdl:output message="impl:getStandardVersionResponse"
4836       name="getStandardVersionResponse"/>
4837     <wsdl:fault message="impl:ImplementationExceptionResponse"

```

```

4838         name="ImplementationExceptionFault"/>
4839     </wsdl:operation>
4840
4841     <wsdl:operation name="getVendorVersion">
4842         <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest" />
4843         <wsdl:output message="impl:getVendorVersionResponse"
4844             name="getVendorVersionResponse" />
4845         <wsdl:fault message="impl:ImplementationExceptionResponse"
4846             name="ImplementationExceptionFault" />
4847     </wsdl:operation>
4848 </wsdl:portType>
4849 <!-- ALETMSERVICE BINDING -->
4850
4851 <wsdl:binding name="ALETMSERVICEBinding" type="impl:ALETMSERVICEPortType">
4852     <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
4853
4854     <wsdl:operation name="defineTMSpec">
4855         <wsdlsoap:operation soapAction="" />
4856         <wsdl:input name="defineTMSpecRequest">
4857             <wsdlsoap:body use="literal" />
4858         </wsdl:input>
4859         <wsdl:output name="defineTMSpecResponse">
4860             <wsdlsoap:body use="literal" />
4861         </wsdl:output>
4862         <wsdl:fault name="DuplicateNameExceptionFault">
4863             <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal" />
4864         </wsdl:fault>
4865         <wsdl:fault name="TMSpecValidationExceptionFault">
4866             <wsdlsoap:fault name="TMSpecValidationExceptionFault" use="literal" />
4867         </wsdl:fault>
4868         <wsdl:fault name="SecurityExceptionFault">
4869             <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4870         </wsdl:fault>
4871         <wsdl:fault name="ImplementationExceptionFault">
4872             <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4873         </wsdl:fault>
4874     </wsdl:operation>
4875
4876     <wsdl:operation name="undefineTMSpec">
4877         <wsdlsoap:operation soapAction="" />
4878         <wsdl:input name="undefineTMSpecRequest">
4879             <wsdlsoap:body use="literal" />
4880         </wsdl:input>
4881         <wsdl:output name="undefineTMSpecResponse">
4882             <wsdlsoap:body use="literal" />
4883         </wsdl:output>
4884         <wsdl:fault name="NoSuchNameExceptionFault">
4885             <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />
4886         </wsdl:fault>
4887         <wsdl:fault name="InUseExceptionFault">
4888             <wsdlsoap:fault name="InUseExceptionFault" use="literal" />
4889         </wsdl:fault>
4890         <wsdl:fault name="SecurityExceptionFault">
4891             <wsdlsoap:fault name="SecurityExceptionFault" use="literal" />
4892         </wsdl:fault>
4893         <wsdl:fault name="ImplementationExceptionFault">
4894             <wsdlsoap:fault name="ImplementationExceptionFault" use="literal" />
4895         </wsdl:fault>
4896     </wsdl:operation>
4897
4898     <wsdl:operation name="getTMSpec">
4899         <wsdlsoap:operation soapAction="" />
4900         <wsdl:input name="getTMSpecRequest">
4901             <wsdlsoap:body use="literal" />
4902         </wsdl:input>
4903         <wsdl:output name="getTMSpecResponse">
4904             <wsdlsoap:body use="literal" />
4905         </wsdl:output>
4906         <wsdl:fault name="NoSuchNameExceptionFault">
4907             <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal" />

```

```

4908     </wsdl:fault>
4909     <wsdl:fault name="SecurityExceptionFault">
4910         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4911     </wsdl:fault>
4912     <wsdl:fault name="ImplementationExceptionFault">
4913         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4914     </wsdl:fault>
4915 </wsdl:operation>
4916
4917 <wsdl:operation name="getTMSpecNames">
4918     <wsdlsoap:operation soapAction=""/>
4919     <wsdl:input name="getTMSpecNamesRequest">
4920         <wsdlsoap:body use="literal"/>
4921     </wsdl:input>
4922     <wsdl:output name="getTMSpecNamesResponse">
4923         <wsdlsoap:body use="literal"/>
4924     </wsdl:output>
4925     <wsdl:fault name="SecurityExceptionFault">
4926         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
4927     </wsdl:fault>
4928     <wsdl:fault name="ImplementationExceptionFault">
4929         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4930     </wsdl:fault>
4931 </wsdl:operation>
4932
4933 <wsdl:operation name="getStandardVersion">
4934     <wsdlsoap:operation soapAction=""/>
4935     <wsdl:input name="getStandardVersionRequest">
4936         <wsdlsoap:body use="literal"/>
4937     </wsdl:input>
4938     <wsdl:output name="getStandardVersionResponse">
4939         <wsdlsoap:body use="literal"/>
4940     </wsdl:output>
4941     <wsdl:fault name="ImplementationExceptionFault">
4942         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4943     </wsdl:fault>
4944 </wsdl:operation>
4945
4946 <wsdl:operation name="getVendorVersion">
4947     <wsdlsoap:operation soapAction=""/>
4948     <wsdl:input name="getVendorVersionRequest">
4949         <wsdlsoap:body use="literal"/>
4950     </wsdl:input>
4951     <wsdl:output name="getVendorVersionResponse">
4952         <wsdlsoap:body use="literal"/>
4953     </wsdl:output>
4954     <wsdl:fault name="ImplementationExceptionFault">
4955         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
4956     </wsdl:fault>
4957 </wsdl:operation>
4958 </wsdl:binding>
4959
4960 <!-- ALETMSERVICE -->
4961 <wsdl:service name="ALETMService">
4962     <wsdl:port binding="impl:ALETMServiceBinding" name="ALETMServicePort">
4963         <!-- The value of the location attribute below is an example only;
4964             Implementations are free to choose any appropriate URL. -->
4965         <wsdlsoap:address location="http://localhost:8080/services/ALETMService"/>
4966     </wsdl:port>
4967 </wsdl:service>
4968 </wsdl:definitions>

```

## 4969 4.6 ALE Logical Reader API SOAP Binding

4970 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
4971 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Logical

4972 Reader API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0  
4973 [WSI].

```
4974 <?xml version="1.0" encoding="UTF-8"?>
4975 <!-- ALELRSERVICE DEFINITIONS -->
4976 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
4977     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
4978     xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
4979     xmlns:ale="urn:epcglobal:ale:xsd:1"
4980     xmlns:impl="urn:epcglobal:alelr:wsdl:1"
4981     targetNamespace="urn:epcglobal:alelr:wsdl:1">
4982 <!-- ALELRSERVICE TYPES -->
4983 <wsdl:types>
4984   <xsd:schema targetNamespace="urn:epcglobal:alelr:wsdl:1">
4985     <xsd:import namespace="urn:epcglobal:ale:xsd:1"
4986       schemaLocation="EPCglobal-ale-1_1-alelr.xsd"/>
4987     <!-- ALELRSERVICE MESSAGE WRAPPERS -->
4988
4989     <xsd:element name="Define" type="impl:Define"/>
4990     <xsd:complexType name="Define">
4991       <xsd:sequence>
4992         <xsd:element name="name" type="xsd:string"/>
4993         <xsd:element name="spec" type="ale:LRSpec"/>
4994       </xsd:sequence>
4995     </xsd:complexType>
4996     <xsd:element name="DefineResult">
4997       <xsd:complexType/>
4998     </xsd:element>
4999
5000     <xsd:element name="Update" type="impl:Update"/>
5001     <xsd:complexType name="Update">
5002       <xsd:sequence>
5003         <xsd:element name="name" type="xsd:string"/>
5004         <xsd:element name="spec" type="ale:LRSpec"/>
5005       </xsd:sequence>
5006     </xsd:complexType>
5007     <xsd:element name="UpdateResult">
5008       <xsd:complexType/>
5009     </xsd:element>
5010
5011     <xsd:element name="Undefine" type="impl:Undefine"/>
5012     <xsd:complexType name="Undefine">
5013       <xsd:sequence>
5014         <xsd:element name="name" type="xsd:string"/>
5015       </xsd:sequence>
5016     </xsd:complexType>
5017     <xsd:element name="UndefineResult">
5018       <xsd:complexType/>
5019     </xsd:element>
5020
5021     <xsd:element name="GetLogicalReaderNames" type="impl:EmptyParms"/>
5022     <xsd:element name="GetLogicalReaderNamesResult" type="impl:ArrayOfString"/>
5023
5024     <xsd:element name="GetLRSpec" type="impl:GetLRSpec"/>
5025     <xsd:complexType name="GetLRSpec">
5026       <xsd:sequence>
5027         <xsd:element name="name" type="xsd:string"/>
5028       </xsd:sequence>
5029     </xsd:complexType>
5030     <xsd:element name="GetLRSpecResult" type="ale:LRSpec"/>
5031
5032     <xsd:element name="AddReaders" type="impl:AddReaders"/>
5033     <xsd:complexType name="AddReaders">
5034       <xsd:sequence>
5035         <xsd:element name="name" type="xsd:string"/>
5036         <xsd:element name="readers" minOccurs="0">
5037           <xsd:complexType>
5038             <xsd:sequence>
5039               <xsd:element name="reader" type="xsd:string" minOccurs="0"
5040                 maxOccurs="unbounded"/>

```

```

5041         </xsd:sequence>
5042     </xsd:complexType>
5043 </xsd:element>
5044 </xsd:sequence>
5045 </xsd:complexType>
5046 <xsd:element name="AddReadersResult">
5047     <xsd:complexType/>
5048 </xsd:element>
5049
5050 <xsd:element name="SetReaders" type="impl:SetReaders"/>
5051 <xsd:complexType name="SetReaders">
5052     <xsd:sequence>
5053         <xsd:element name="name" type="xsd:string"/>
5054         <xsd:element name="readers" minOccurs="0">
5055             <xsd:complexType>
5056                 <xsd:sequence>
5057                     <xsd:element name="reader" type="xsd:string" minOccurs="0"
5058                         maxOccurs="unbounded"/>
5059                 </xsd:sequence>
5060             </xsd:complexType>
5061         </xsd:element>
5062     </xsd:sequence>
5063 </xsd:complexType>
5064 <xsd:element name="SetReadersResult">
5065     <xsd:complexType/>
5066 </xsd:element>
5067
5068 <xsd:element name="RemoveReaders" type="impl:RemoveReaders"/>
5069 <xsd:complexType name="RemoveReaders">
5070     <xsd:sequence>
5071         <xsd:element name="name" type="xsd:string"/>
5072         <xsd:element name="readers" minOccurs="0">
5073             <xsd:complexType>
5074                 <xsd:sequence>
5075                     <xsd:element name="reader" type="xsd:string" minOccurs="0"
5076                         maxOccurs="unbounded"/>
5077                 </xsd:sequence>
5078             </xsd:complexType>
5079         </xsd:element>
5080     </xsd:sequence>
5081 </xsd:complexType>
5082 <xsd:element name="RemoveReadersResult">
5083     <xsd:complexType/>
5084 </xsd:element>
5085
5086 <xsd:element name="SetProperties" type="impl:SetProperties"/>
5087 <xsd:complexType name="SetProperties">
5088     <xsd:sequence>
5089         <xsd:element name="name" type="xsd:string"/>
5090         <xsd:element name="properties" minOccurs="0">
5091             <xsd:complexType>
5092                 <xsd:sequence>
5093                     <xsd:element name="property" type="ale:LRProperty" minOccurs="0"
5094                         maxOccurs="unbounded"/>
5095                 </xsd:sequence>
5096             </xsd:complexType>
5097         </xsd:element>
5098     </xsd:sequence>
5099 </xsd:complexType>
5100 <xsd:element name="SetPropertiesResult">
5101     <xsd:complexType/>
5102 </xsd:element>
5103
5104 <xsd:element name="GetPropertyValue" type="impl:GetPropertyValue"/>
5105 <xsd:complexType name="GetPropertyValue">
5106     <xsd:sequence>
5107         <xsd:element name="name" type="xsd:string"/>
5108         <xsd:element name="propertyName" type="xsd:string"/>
5109     </xsd:sequence>
5110 </xsd:complexType>

```

```

5111 <xsd:element name="GetPropertyValueResult" type="xsd:string"/>
5112
5113 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
5114 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
5115
5116 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
5117 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
5118
5119 <xsd:element name="ALEException" type="impl:ALEException"/>
5120 <xsd:complexType name="ALEException">
5121 <xsd:sequence>
5122 <xsd:element name="reason" type="xsd:string"/>
5123 </xsd:sequence>
5124 </xsd:complexType>
5125
5126 <xsd:element name="DuplicateNameException" type="impl:DuplicateNameException"/>
5127 <xsd:complexType name="DuplicateNameException">
5128 <xsd:complexContent>
5129 <xsd:extension base="impl:ALEException"/>
5130 </xsd:complexContent>
5131 </xsd:complexType>
5132
5133 <xsd:element name="NoSuchNameException" type="impl:NoSuchNameException"/>
5134 <xsd:complexType name="NoSuchNameException">
5135 <xsd:complexContent>
5136 <xsd:extension base="impl:ALEException"/>
5137 </xsd:complexContent>
5138 </xsd:complexType>
5139
5140 <xsd:element name="InUseException" type="impl:InUseException"/>
5141 <xsd:complexType name="InUseException">
5142 <xsd:complexContent>
5143 <xsd:extension base="impl:ALEException"/>
5144 </xsd:complexContent>
5145 </xsd:complexType>
5146
5147 <xsd:element name="ValidationException" type="impl:ValidationException"/>
5148 <xsd:complexType name="ValidationException">
5149 <xsd:complexContent>
5150 <xsd:extension base="impl:ALEException"/>
5151 </xsd:complexContent>
5152 </xsd:complexType>
5153
5154 <xsd:element name="ImmutableReaderException" type="impl:ImmutableReaderException"/>
5155 <xsd:complexType name="ImmutableReaderException">
5156 <xsd:complexContent>
5157 <xsd:extension base="impl:ALEException"/>
5158 </xsd:complexContent>
5159 </xsd:complexType>
5160
5161 <xsd:element name="NonCompositeReaderException"
5162 <xsd:complexType name="NonCompositeReaderException">
5163 <xsd:complexContent>
5164 <xsd:extension base="impl:ALEException"/>
5165 </xsd:complexContent>
5166 </xsd:complexType>
5167
5168 <xsd:element name="ReaderLoopException" type="impl:ReaderLoopException"/>
5169 <xsd:complexType name="ReaderLoopException">
5170 <xsd:complexContent>
5171 <xsd:extension base="impl:ALEException"/>
5172 </xsd:complexContent>
5173 </xsd:complexType>
5174
5175 <xsd:element name="SecurityException" type="impl:SecurityException"/>
5176 <xsd:complexType name="SecurityException">
5177 <xsd:complexContent>
5178 <xsd:extension base="impl:ALEException"/>
5179 </xsd:complexContent>
5180

```

```

5181     </xsd:complexType>
5182
5183     <xsd:element name="ImplementationException" type="impl:ImplementationException"/>
5184     <xsd:complexType name="ImplementationException">
5185         <xsd:complexContent>
5186             <xsd:extension base="impl:ALEException">
5187                 <xsd:sequence>
5188                     <xsd:element name="severity" type="impl:ImplementationExceptionSeverity"/>
5189                 </xsd:sequence>
5190             </xsd:extension>
5191         </xsd:complexContent>
5192     </xsd:complexType>
5193
5194     <xsd:complexType name="ArrayOfString">
5195         <xsd:sequence>
5196             <xsd:element name="string" type="xsd:string" minOccurs="0"
5197                 maxOccurs="unbounded"/>
5198         </xsd:sequence>
5199     </xsd:complexType>
5200
5201     <!-- The ImplementationExceptionSeverity type is an enumerated type.
5202         The following strings are legal values for this type:
5203         ERROR
5204         SEVERE
5205         -->
5206     <xsd:simpleType name="ImplementationExceptionSeverity">
5207         <xsd:restriction base="xsd:string"/>
5208     </xsd:simpleType>
5209
5210     <xsd:complexType name="EmptyParms"/>
5211 </xsd:schema>
5212 </wsdl:types>
5213 <!-- ALELRSERVICE MESSAGES -->
5214
5215 <wsdl:message name="defineRequest">
5216     <wsdl:part name="parms" element="impl:Define"/>
5217 </wsdl:message>
5218 <wsdl:message name="defineResponse">
5219     <wsdl:part name="defineReturn" element="impl:DefineResult"/>
5220 </wsdl:message>
5221
5222 <wsdl:message name="updateRequest">
5223     <wsdl:part name="parms" element="impl:Update"/>
5224 </wsdl:message>
5225 <wsdl:message name="updateResponse">
5226     <wsdl:part name="updateReturn" element="impl:UpdateResult"/>
5227 </wsdl:message>
5228
5229 <wsdl:message name="undefineRequest">
5230     <wsdl:part name="parms" element="impl:Undefine"/>
5231 </wsdl:message>
5232 <wsdl:message name="undefineResponse">
5233     <wsdl:part name="undefineReturn" element="impl:UndefineResult"/>
5234 </wsdl:message>
5235
5236 <wsdl:message name="getLogicalReaderNamesRequest">
5237     <wsdl:part name="parms" element="impl:GetLogicalReaderNames"/>
5238 </wsdl:message>
5239 <wsdl:message name="getLogicalReaderNamesResponse">
5240     <wsdl:part name="getLogicalReaderNamesReturn"
5241         element="impl:GetLogicalReaderNamesResult"/>
5242 </wsdl:message>
5243
5244 <wsdl:message name="getLRSpecRequest">
5245     <wsdl:part name="parms" element="impl:GetLRSpec"/>
5246 </wsdl:message>
5247 <wsdl:message name="getLRSpecResponse">
5248     <wsdl:part name="getLRSpecReturn" element="impl:GetLRSpecResult"/>
5249 </wsdl:message>
5250

```



```

5251 <wsdl:message name="addReadersRequest">
5252   <wsdl:part name="parms" element="impl:AddReaders"/>
5253 </wsdl:message>
5254 <wsdl:message name="addReadersResponse">
5255   <wsdl:part name="addReadersReturn" element="impl:AddReadersResult"/>
5256 </wsdl:message>
5257
5258 <wsdl:message name="setReadersRequest">
5259   <wsdl:part name="parms" element="impl:SetReaders"/>
5260 </wsdl:message>
5261 <wsdl:message name="setReadersResponse">
5262   <wsdl:part name="setReadersReturn" element="impl:SetReadersResult"/>
5263 </wsdl:message>
5264
5265 <wsdl:message name="removeReadersRequest">
5266   <wsdl:part name="parms" element="impl:RemoveReaders"/>
5267 </wsdl:message>
5268 <wsdl:message name="removeReadersResponse">
5269   <wsdl:part name="removeReadersReturn" element="impl:RemoveReadersResult"/>
5270 </wsdl:message>
5271
5272 <wsdl:message name="setPropertyRequest">
5273   <wsdl:part name="parms" element="impl:SetProperties"/>
5274 </wsdl:message>
5275 <wsdl:message name="setPropertyResponse">
5276   <wsdl:part name="setPropertyReturn" element="impl:SetPropertiesResult"/>
5277 </wsdl:message>
5278
5279 <wsdl:message name="getPropertyValueRequest">
5280   <wsdl:part name="parms" element="impl:GetPropertyValue"/>
5281 </wsdl:message>
5282 <wsdl:message name="getPropertyValueResponse">
5283   <wsdl:part name="getPropertyValueReturn" element="impl:GetPropertyValueResult"/>
5284 </wsdl:message>
5285
5286 <wsdl:message name="getStandardVersionRequest">
5287   <wsdl:part name="parms" element="impl:GetStandardVersion"/>
5288 </wsdl:message>
5289 <wsdl:message name="getStandardVersionResponse">
5290   <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult"/>
5291 </wsdl:message>
5292
5293 <wsdl:message name="getVendorVersionRequest">
5294   <wsdl:part name="parms" element="impl:GetVendorVersion"/>
5295 </wsdl:message>
5296 <wsdl:message name="getVendorVersionResponse">
5297   <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult"/>
5298 </wsdl:message>
5299
5300 <wsdl:message name="DuplicateNameExceptionResponse">
5301   <wsdl:part name="fault" element="impl:DuplicateNameException"/>
5302 </wsdl:message>
5303
5304 <wsdl:message name="NoSuchNameExceptionResponse">
5305   <wsdl:part name="fault" element="impl:NoSuchNameException"/>
5306 </wsdl:message>
5307
5308 <wsdl:message name="InUseExceptionResponse">
5309   <wsdl:part name="fault" element="impl:InUseException"/>
5310 </wsdl:message>
5311
5312 <wsdl:message name="ValidationExceptionResponse">
5313   <wsdl:part name="fault" element="impl:ValidationException"/>
5314 </wsdl:message>
5315
5316 <wsdl:message name="ImmutableReaderExceptionResponse">
5317   <wsdl:part name="fault" element="impl:ImmutableReaderException"/>
5318 </wsdl:message>
5319
5320 <wsdl:message name="NonCompositeReaderExceptionResponse">

```

```

5321     <wsdl:part name="fault" element="impl:NonCompositeReaderException"/>
5322 </wsdl:message>
5323
5324 <wsdl:message name="ReaderLoopExceptionResponse">
5325     <wsdl:part name="fault" element="impl:ReaderLoopException"/>
5326 </wsdl:message>
5327
5328 <wsdl:message name="SecurityExceptionResponse">
5329     <wsdl:part name="fault" element="impl:SecurityException"/>
5330 </wsdl:message>
5331
5332 <wsdl:message name="ImplementationExceptionResponse">
5333     <wsdl:part name="fault" element="impl:ImplementationException"/>
5334 </wsdl:message>
5335 <!-- ALELRSERVICE PORTTYPE -->
5336
5337 <wsdl:portType name="ALELRServicePortType">
5338
5339     <wsdl:operation name="define">
5340         <wsdl:input message="impl:defineRequest" name="defineRequest"/>
5341         <wsdl:output message="impl:defineResponse" name="defineResponse"/>
5342         <wsdl:fault message="impl:DuplicateNameExceptionResponse"
5343             name="DuplicateNameExceptionFault"/>
5344         <wsdl:fault message="impl:ValidationExceptionResponse"
5345             name="ValidationExceptionFault"/>
5346         <wsdl:fault message="impl:SecurityExceptionResponse"
5347             name="SecurityExceptionFault"/>
5348         <wsdl:fault message="impl:ImplementationExceptionResponse"
5349             name="ImplementationExceptionFault"/>
5350     </wsdl:operation>
5351
5352     <wsdl:operation name="update">
5353         <wsdl:input message="impl:updateRequest" name="updateRequest"/>
5354         <wsdl:output message="impl:updateResponse" name="updateResponse"/>
5355         <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5356             name="NoSuchNameExceptionFault"/>
5357         <wsdl:fault message="impl:ValidationExceptionResponse"
5358             name="ValidationExceptionFault"/>
5359         <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault"/>
5360         <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5361             name="ImmutableReaderExceptionFault"/>
5362         <wsdl:fault message="impl:ReaderLoopExceptionResponse"
5363             name="ReaderLoopExceptionFault"/>
5364         <wsdl:fault message="impl:SecurityExceptionResponse"
5365             name="SecurityExceptionFault"/>
5366         <wsdl:fault message="impl:ImplementationExceptionResponse"
5367             name="ImplementationExceptionFault"/>
5368     </wsdl:operation>
5369
5370     <wsdl:operation name="undefine">
5371         <wsdl:input message="impl:undefineRequest" name="undefineRequest"/>
5372         <wsdl:output message="impl:undefineResponse" name="undefineResponse"/>
5373         <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5374             name="NoSuchNameExceptionFault"/>
5375         <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault"/>
5376         <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5377             name="ImmutableReaderExceptionFault"/>
5378         <wsdl:fault message="impl:SecurityExceptionResponse"
5379             name="SecurityExceptionFault"/>
5380         <wsdl:fault message="impl:ImplementationExceptionResponse"
5381             name="ImplementationExceptionFault"/>
5382     </wsdl:operation>
5383
5384     <wsdl:operation name="getLogicalReaderNames">
5385         <wsdl:input message="impl:getLogicalReaderNamesRequest"
5386             name="getLogicalReaderNamesRequest"/>
5387         <wsdl:output message="impl:getLogicalReaderNamesResponse"
5388             name="getLogicalReaderNamesResponse"/>
5389         <wsdl:fault message="impl:SecurityExceptionResponse"
5390             name="SecurityExceptionFault"/>

```

```

5391     <wsdl:fault message="impl:ImplementationExceptionResponse"
5392             name="ImplementationExceptionFault" />
5393 </wsdl:operation>
5394
5395 <wsdl:operation name="getLRSpec">
5396   <wsdl:input message="impl:getLRSpecRequest" name="getLRSpecRequest" />
5397   <wsdl:output message="impl:getLRSpecResponse" name="getLRSpecResponse" />
5398   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5399             name="NoSuchNameExceptionFault" />
5400   <wsdl:fault message="impl:SecurityExceptionResponse"
5401             name="SecurityExceptionFault" />
5402   <wsdl:fault message="impl:ImplementationExceptionResponse"
5403             name="ImplementationExceptionFault" />
5404 </wsdl:operation>
5405
5406 <wsdl:operation name="addReaders">
5407   <wsdl:input message="impl:addReadersRequest" name="addReadersRequest" />
5408   <wsdl:output message="impl:addReadersResponse" name="addReadersResponse" />
5409   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5410             name="NoSuchNameExceptionFault" />
5411   <wsdl:fault message="impl:ValidationExceptionResponse"
5412             name="ValidationExceptionFault" />
5413   <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
5414   <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5415             name="ImmutableReaderExceptionFault" />
5416   <wsdl:fault message="impl:NonCompositeReaderExceptionResponse"
5417             name="NonCompositeReaderExceptionFault" />
5418   <wsdl:fault message="impl:ReaderLoopExceptionResponse"
5419             name="ReaderLoopExceptionFault" />
5420   <wsdl:fault message="impl:SecurityExceptionResponse"
5421             name="SecurityExceptionFault" />
5422   <wsdl:fault message="impl:ImplementationExceptionResponse"
5423             name="ImplementationExceptionFault" />
5424 </wsdl:operation>
5425
5426 <wsdl:operation name="setReaders">
5427   <wsdl:input message="impl:setReadersRequest" name="setReadersRequest" />
5428   <wsdl:output message="impl:setReadersResponse" name="setReadersResponse" />
5429   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5430             name="NoSuchNameExceptionFault" />
5431   <wsdl:fault message="impl:ValidationExceptionResponse"
5432             name="ValidationExceptionFault" />
5433   <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
5434   <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5435             name="ImmutableReaderExceptionFault" />
5436   <wsdl:fault message="impl:NonCompositeReaderExceptionResponse"
5437             name="NonCompositeReaderExceptionFault" />
5438   <wsdl:fault message="impl:ReaderLoopExceptionResponse"
5439             name="ReaderLoopExceptionFault" />
5440   <wsdl:fault message="impl:SecurityExceptionResponse"
5441             name="SecurityExceptionFault" />
5442   <wsdl:fault message="impl:ImplementationExceptionResponse"
5443             name="ImplementationExceptionFault" />
5444 </wsdl:operation>
5445
5446 <wsdl:operation name="removeReaders">
5447   <wsdl:input message="impl:removeReadersRequest" name="removeReadersRequest" />
5448   <wsdl:output message="impl:removeReadersResponse" name="removeReadersResponse" />
5449   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5450             name="NoSuchNameExceptionFault" />
5451   <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault" />
5452   <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5453             name="ImmutableReaderExceptionFault" />
5454   <wsdl:fault message="impl:NonCompositeReaderExceptionResponse"
5455             name="NonCompositeReaderExceptionFault" />
5456   <wsdl:fault message="impl:SecurityExceptionResponse"
5457             name="SecurityExceptionFault" />
5458   <wsdl:fault message="impl:ImplementationExceptionResponse"
5459             name="ImplementationExceptionFault" />
5460 </wsdl:operation>

```

```

5461
5462 <wsdl:operation name="setProperties">
5463   <wsdl:input message="impl:setPropertiesRequest" name="setPropertiesRequest"/>
5464   <wsdl:output message="impl:setPropertiesResponse" name="setPropertiesResponse"/>
5465   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5466     name="NoSuchNameExceptionFault"/>
5467   <wsdl:fault message="impl:ValidationExceptionResponse"
5468     name="ValidationExceptionFault"/>
5469   <wsdl:fault message="impl:InUseExceptionResponse" name="InUseExceptionFault"/>
5470   <wsdl:fault message="impl:ImmutableReaderExceptionResponse"
5471     name="ImmutableReaderExceptionFault"/>
5472   <wsdl:fault message="impl:SecurityExceptionResponse"
5473     name="SecurityExceptionFault"/>
5474   <wsdl:fault message="impl:ImplementationExceptionResponse"
5475     name="ImplementationExceptionFault"/>
5476 </wsdl:operation>
5477
5478 <wsdl:operation name="getPropertyValue">
5479   <wsdl:input message="impl:getPropertyValueRequest" name="getPropertyValueRequest"/>
5480   <wsdl:output message="impl:getPropertyValueResponse"
5481     name="getPropertyValueResponse"/>
5482   <wsdl:fault message="impl:NoSuchNameExceptionResponse"
5483     name="NoSuchNameExceptionFault"/>
5484   <wsdl:fault message="impl:SecurityExceptionResponse"
5485     name="SecurityExceptionFault"/>
5486   <wsdl:fault message="impl:ImplementationExceptionResponse"
5487     name="ImplementationExceptionFault"/>
5488 </wsdl:operation>
5489
5490 <wsdl:operation name="getStandardVersion">
5491   <wsdl:input message="impl:getStandardVersionRequest"
5492     name="getStandardVersionRequest"/>
5493   <wsdl:output message="impl:getStandardVersionResponse"
5494     name="getStandardVersionResponse"/>
5495   <wsdl:fault message="impl:ImplementationExceptionResponse"
5496     name="ImplementationExceptionFault"/>
5497 </wsdl:operation>
5498
5499 <wsdl:operation name="getVendorVersion">
5500   <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest"/>
5501   <wsdl:output message="impl:getVendorVersionResponse"
5502     name="getVendorVersionResponse"/>
5503   <wsdl:fault message="impl:ImplementationExceptionResponse"
5504     name="ImplementationExceptionFault"/>
5505 </wsdl:operation>
5506 </wsdl:portType>
5507 <!-- ALELRService BINDING -->
5508
5509 <wsdl:binding name="ALELRServiceBinding" type="impl:ALELRServicePortType">
5510   <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
5511
5512   <wsdl:operation name="define">
5513     <wsdlsoap:operation soapAction=""/>
5514     <wsdl:input name="defineRequest">
5515       <wsdlsoap:body use="literal"/>
5516     </wsdl:input>
5517     <wsdl:output name="defineResponse">
5518       <wsdlsoap:body use="literal"/>
5519     </wsdl:output>
5520     <wsdl:fault name="DuplicateNameExceptionFault">
5521       <wsdlsoap:fault name="DuplicateNameExceptionFault" use="literal"/>
5522     </wsdl:fault>
5523     <wsdl:fault name="ValidationExceptionFault">
5524       <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5525     </wsdl:fault>
5526     <wsdl:fault name="SecurityExceptionFault">
5527       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5528     </wsdl:fault>
5529     <wsdl:fault name="ImplementationExceptionFault">
5530       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>

```

```

5531     </wsdl:fault>
5532 </wsdl:operation>
5533
5534 <wsdl:operation name="update">
5535   <wsdlsoap:operation soapAction=""/>
5536   <wsdl:input name="updateRequest">
5537     <wsdlsoap:body use="literal"/>
5538   </wsdl:input>
5539   <wsdl:output name="updateResponse">
5540     <wsdlsoap:body use="literal"/>
5541   </wsdl:output>
5542   <wsdl:fault name="NoSuchNameExceptionFault">
5543     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5544   </wsdl:fault>
5545   <wsdl:fault name="ValidationExceptionFault">
5546     <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5547   </wsdl:fault>
5548   <wsdl:fault name="InUseExceptionFault">
5549     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5550   </wsdl:fault>
5551   <wsdl:fault name="ImmutableReaderExceptionFault">
5552     <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5553   </wsdl:fault>
5554   <wsdl:fault name="ReaderLoopExceptionFault">
5555     <wsdlsoap:fault name="ReaderLoopExceptionFault" use="literal"/>
5556   </wsdl:fault>
5557   <wsdl:fault name="SecurityExceptionFault">
5558     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5559   </wsdl:fault>
5560   <wsdl:fault name="ImplementationExceptionFault">
5561     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5562   </wsdl:fault>
5563 </wsdl:operation>
5564
5565 <wsdl:operation name="undefine">
5566   <wsdlsoap:operation soapAction=""/>
5567   <wsdl:input name="undefineRequest">
5568     <wsdlsoap:body use="literal"/>
5569   </wsdl:input>
5570   <wsdl:output name="undefineResponse">
5571     <wsdlsoap:body use="literal"/>
5572   </wsdl:output>
5573   <wsdl:fault name="NoSuchNameExceptionFault">
5574     <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5575   </wsdl:fault>
5576   <wsdl:fault name="InUseExceptionFault">
5577     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5578   </wsdl:fault>
5579   <wsdl:fault name="ImmutableReaderExceptionFault">
5580     <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5581   </wsdl:fault>
5582   <wsdl:fault name="SecurityExceptionFault">
5583     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5584   </wsdl:fault>
5585   <wsdl:fault name="ImplementationExceptionFault">
5586     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5587   </wsdl:fault>
5588 </wsdl:operation>
5589
5590 <wsdl:operation name="getLogicalReaderNames">
5591   <wsdlsoap:operation soapAction=""/>
5592   <wsdl:input name="getLogicalReaderNamesRequest">
5593     <wsdlsoap:body use="literal"/>
5594   </wsdl:input>
5595   <wsdl:output name="getLogicalReaderNamesResponse">
5596     <wsdlsoap:body use="literal"/>
5597   </wsdl:output>
5598   <wsdl:fault name="SecurityExceptionFault">
5599     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5600   </wsdl:fault>

```

```

5601     <wsdl:fault name="ImplementationExceptionFault">
5602         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5603     </wsdl:fault>
5604 </wsdl:operation>
5605
5606 <wsdl:operation name="getLRSpec">
5607     <wsdlsoap:operation soapAction=""/>
5608     <wsdl:input name="getLRSpecRequest">
5609         <wsdlsoap:body use="literal"/>
5610     </wsdl:input>
5611     <wsdl:output name="getLRSpecResponse">
5612         <wsdlsoap:body use="literal"/>
5613     </wsdl:output>
5614     <wsdl:fault name="NoSuchNameExceptionFault">
5615         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5616     </wsdl:fault>
5617     <wsdl:fault name="SecurityExceptionFault">
5618         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5619     </wsdl:fault>
5620     <wsdl:fault name="ImplementationExceptionFault">
5621         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5622     </wsdl:fault>
5623 </wsdl:operation>
5624
5625 <wsdl:operation name="addReaders">
5626     <wsdlsoap:operation soapAction=""/>
5627     <wsdl:input name="addReadersRequest">
5628         <wsdlsoap:body use="literal"/>
5629     </wsdl:input>
5630     <wsdl:output name="addReadersResponse">
5631         <wsdlsoap:body use="literal"/>
5632     </wsdl:output>
5633     <wsdl:fault name="NoSuchNameExceptionFault">
5634         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5635     </wsdl:fault>
5636     <wsdl:fault name="ValidationExceptionFault">
5637         <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5638     </wsdl:fault>
5639     <wsdl:fault name="InUseExceptionFault">
5640         <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5641     </wsdl:fault>
5642     <wsdl:fault name="ImmutableReaderExceptionFault">
5643         <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5644     </wsdl:fault>
5645     <wsdl:fault name="NonCompositeReaderExceptionFault">
5646         <wsdlsoap:fault name="NonCompositeReaderExceptionFault" use="literal"/>
5647     </wsdl:fault>
5648     <wsdl:fault name="ReaderLoopExceptionFault">
5649         <wsdlsoap:fault name="ReaderLoopExceptionFault" use="literal"/>
5650     </wsdl:fault>
5651     <wsdl:fault name="SecurityExceptionFault">
5652         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5653     </wsdl:fault>
5654     <wsdl:fault name="ImplementationExceptionFault">
5655         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5656     </wsdl:fault>
5657 </wsdl:operation>
5658
5659 <wsdl:operation name="setReaders">
5660     <wsdlsoap:operation soapAction=""/>
5661     <wsdl:input name="setReadersRequest">
5662         <wsdlsoap:body use="literal"/>
5663     </wsdl:input>
5664     <wsdl:output name="setReadersResponse">
5665         <wsdlsoap:body use="literal"/>
5666     </wsdl:output>
5667     <wsdl:fault name="NoSuchNameExceptionFault">
5668         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5669     </wsdl:fault>
5670     <wsdl:fault name="ValidationExceptionFault">

```



```

5671     <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5672 </wsdl:fault>
5673 <wsdl:fault name="InUseExceptionFault">
5674     <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5675 </wsdl:fault>
5676 <wsdl:fault name="ImmutableReaderExceptionFault">
5677     <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5678 </wsdl:fault>
5679 <wsdl:fault name="NonCompositeReaderExceptionFault">
5680     <wsdlsoap:fault name="NonCompositeReaderExceptionFault" use="literal"/>
5681 </wsdl:fault>
5682 <wsdl:fault name="ReaderLoopExceptionFault">
5683     <wsdlsoap:fault name="ReaderLoopExceptionFault" use="literal"/>
5684 </wsdl:fault>
5685 <wsdl:fault name="SecurityExceptionFault">
5686     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5687 </wsdl:fault>
5688 <wsdl:fault name="ImplementationExceptionFault">
5689     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5690 </wsdl:fault>
5691 </wsdl:operation>
5692
5693 <wsdl:operation name="removeReaders">
5694     <wsdlsoap:operation soapAction="" />
5695     <wsdl:input name="removeReadersRequest">
5696         <wsdlsoap:body use="literal"/>
5697     </wsdl:input>
5698     <wsdl:output name="removeReadersResponse">
5699         <wsdlsoap:body use="literal"/>
5700     </wsdl:output>
5701     <wsdl:fault name="NoSuchNameExceptionFault">
5702         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5703     </wsdl:fault>
5704     <wsdl:fault name="InUseExceptionFault">
5705         <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5706     </wsdl:fault>
5707     <wsdl:fault name="ImmutableReaderExceptionFault">
5708         <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5709     </wsdl:fault>
5710     <wsdl:fault name="NonCompositeReaderExceptionFault">
5711         <wsdlsoap:fault name="NonCompositeReaderExceptionFault" use="literal"/>
5712     </wsdl:fault>
5713     <wsdl:fault name="SecurityExceptionFault">
5714         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5715     </wsdl:fault>
5716     <wsdl:fault name="ImplementationExceptionFault">
5717         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5718     </wsdl:fault>
5719 </wsdl:operation>
5720
5721 <wsdl:operation name="setPropertyies">
5722     <wsdlsoap:operation soapAction="" />
5723     <wsdl:input name="setPropertyiesRequest">
5724         <wsdlsoap:body use="literal"/>
5725     </wsdl:input>
5726     <wsdl:output name="setPropertyiesResponse">
5727         <wsdlsoap:body use="literal"/>
5728     </wsdl:output>
5729     <wsdl:fault name="NoSuchNameExceptionFault">
5730         <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5731     </wsdl:fault>
5732     <wsdl:fault name="ValidationExceptionFault">
5733         <wsdlsoap:fault name="ValidationExceptionFault" use="literal"/>
5734     </wsdl:fault>
5735     <wsdl:fault name="InUseExceptionFault">
5736         <wsdlsoap:fault name="InUseExceptionFault" use="literal"/>
5737     </wsdl:fault>
5738     <wsdl:fault name="ImmutableReaderExceptionFault">
5739         <wsdlsoap:fault name="ImmutableReaderExceptionFault" use="literal"/>
5740     </wsdl:fault>

```



```

5741     <wsdl:fault name="SecurityExceptionFault">
5742       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5743     </wsdl:fault>
5744     <wsdl:fault name="ImplementationExceptionFault">
5745       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5746     </wsdl:fault>
5747   </wsdl:operation>
5748
5749   <wsdl:operation name="getPropertyValue">
5750     <wsdlsoap:operation soapAction=""/>
5751     <wsdl:input name="getPropertyValueRequest">
5752       <wsdlsoap:body use="literal"/>
5753     </wsdl:input>
5754     <wsdl:output name="getPropertyValueResponse">
5755       <wsdlsoap:body use="literal"/>
5756     </wsdl:output>
5757     <wsdl:fault name="NoSuchNameExceptionFault">
5758       <wsdlsoap:fault name="NoSuchNameExceptionFault" use="literal"/>
5759     </wsdl:fault>
5760     <wsdl:fault name="SecurityExceptionFault">
5761       <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
5762     </wsdl:fault>
5763     <wsdl:fault name="ImplementationExceptionFault">
5764       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5765     </wsdl:fault>
5766   </wsdl:operation>
5767
5768   <wsdl:operation name="getStandardVersion">
5769     <wsdlsoap:operation soapAction=""/>
5770     <wsdl:input name="getStandardVersionRequest">
5771       <wsdlsoap:body use="literal"/>
5772     </wsdl:input>
5773     <wsdl:output name="getStandardVersionResponse">
5774       <wsdlsoap:body use="literal"/>
5775     </wsdl:output>
5776     <wsdl:fault name="ImplementationExceptionFault">
5777       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5778     </wsdl:fault>
5779   </wsdl:operation>
5780
5781   <wsdl:operation name="getVendorVersion">
5782     <wsdlsoap:operation soapAction=""/>
5783     <wsdl:input name="getVendorVersionRequest">
5784       <wsdlsoap:body use="literal"/>
5785     </wsdl:input>
5786     <wsdl:output name="getVendorVersionResponse">
5787       <wsdlsoap:body use="literal"/>
5788     </wsdl:output>
5789     <wsdl:fault name="ImplementationExceptionFault">
5790       <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
5791     </wsdl:fault>
5792   </wsdl:operation>
5793 </wsdl:binding>
5794
5795 <!-- ALELRSERVICE -->
5796 <wsdl:service name="ALELRService">
5797   <wsdl:port binding="impl:ALELRServiceBinding" name="ALELRServicePort">
5798     <!-- The value of the location attribute below is an example only;
5799       Implementations are free to choose any appropriate URL. -->
5800     <wsdlsoap:address location="http://localhost:8080/services/ALELRService"/>
5801   </wsdl:port>
5802 </wsdl:service>
5803 </wsdl:definitions>

```

## 5804 4.7 ALE Access Control API SOAP Binding

5805 The following is a Web Services Description Language (WSDL) 1.1 [WSDL1.1]  
5806 specification defining the standard SOAP 1.1 [SOAP1.1] binding of the ALE Access

5807 Control API. This SOAP binding is compliant with the WS-I Basic Profile Version 1.0  
5808 [WSI].

```
5809 <?xml version="1.0" encoding="UTF-8"?>
5810 <!-- ALEACSERVICE DEFINITIONS -->
5811 <wsdl:definitions xmlns:xsd="http://www.w3.org/2001/XMLSchema"
5812     xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
5813     xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
5814     xmlns:ale="urn:epcglobal:ale:xsd:1"
5815     xmlns:impl="urn:epcglobal:aleac:wsdl:1"
5816     targetNamespace="urn:epcglobal:aleac:wsdl:1">
5817   <!-- ALEACSERVICE TYPES -->
5818   <wsdl:types>
5819     <xsd:schema targetNamespace="urn:epcglobal:aleac:wsdl:1">
5820       <xsd:import namespace="urn:epcglobal:ale:xsd:1"
5821         schemaLocation="EPCglobal-ale-1_1-aleac.xsd"/>
5822       <!-- ALEACSERVICE MESSAGE WRAPPERS -->
5823
5824       <xsd:element name="GetPermissionNames" type="impl:EmptyParms"/>
5825       <xsd:element name="GetPermissionNamesResult" type="impl:ArrayOfString"/>
5826
5827       <xsd:element name="DefinePermission" type="impl:DefinePermission"/>
5828       <xsd:complexType name="DefinePermission">
5829         <xsd:sequence>
5830           <xsd:element name="permName" type="xsd:string"/>
5831           <xsd:element name="perm" type="ale:ACPermission"/>
5832         </xsd:sequence>
5833       </xsd:complexType>
5834       <xsd:element name="DefinePermissionResult">
5835         <xsd:complexType/>
5836       </xsd:element>
5837
5838       <xsd:element name="UpdatePermission" type="impl:UpdatePermission"/>
5839       <xsd:complexType name="UpdatePermission">
5840         <xsd:sequence>
5841           <xsd:element name="permName" type="xsd:string"/>
5842           <xsd:element name="perm" type="ale:ACPermission"/>
5843         </xsd:sequence>
5844       </xsd:complexType>
5845       <xsd:element name="UpdatePermissionResult">
5846         <xsd:complexType/>
5847       </xsd:element>
5848
5849       <xsd:element name="GetPermission" type="impl:GetPermission"/>
5850       <xsd:complexType name="GetPermission">
5851         <xsd:sequence>
5852           <xsd:element name="permName" type="xsd:string"/>
5853         </xsd:sequence>
5854       </xsd:complexType>
5855       <xsd:element name="GetPermissionResult" type="ale:ACPermission"/>
5856
5857       <xsd:element name="UndefinePermission" type="impl:UndefinePermission"/>
5858       <xsd:complexType name="UndefinePermission">
5859         <xsd:sequence>
5860           <xsd:element name="permName" type="xsd:string"/>
5861         </xsd:sequence>
5862       </xsd:complexType>
5863       <xsd:element name="UndefinePermissionResult">
5864         <xsd:complexType/>
5865       </xsd:element>
5866
5867       <xsd:element name="GetRoleNames" type="impl:EmptyParms"/>
5868       <xsd:element name="GetRoleNamesResult" type="impl:ArrayOfString"/>
5869
5870       <xsd:element name="DefineRole" type="impl:DefineRole"/>
5871       <xsd:complexType name="DefineRole">
5872         <xsd:sequence>
5873           <xsd:element name="roleName" type="xsd:string"/>
5874           <xsd:element name="role" type="ale:ACRole"/>
5875         </xsd:sequence>
```

```

5876     </xsd:complexType>
5877     <xsd:element name="DefineRoleResult">
5878       <xsd:complexType/>
5879     </xsd:element>
5880
5881     <xsd:element name="UpdateRole" type="impl:UpdateRole"/>
5882     <xsd:complexType name="UpdateRole">
5883       <xsd:sequence>
5884         <xsd:element name="roleName" type="xsd:string"/>
5885         <xsd:element name="role" type="ale:ACRole"/>
5886       </xsd:sequence>
5887     </xsd:complexType>
5888     <xsd:element name="UpdateRoleResult">
5889       <xsd:complexType/>
5890     </xsd:element>
5891
5892     <xsd:element name="GetRole" type="impl:GetRole"/>
5893     <xsd:complexType name="GetRole">
5894       <xsd:sequence>
5895         <xsd:element name="roleName" type="xsd:string"/>
5896       </xsd:sequence>
5897     </xsd:complexType>
5898     <xsd:element name="GetRoleResult" type="ale:ACRole"/>
5899
5900     <xsd:element name="UndefineRole" type="impl:UndefineRole"/>
5901     <xsd:complexType name="UndefineRole">
5902       <xsd:sequence>
5903         <xsd:element name="roleName" type="xsd:string"/>
5904       </xsd:sequence>
5905     </xsd:complexType>
5906     <xsd:element name="UndefineRoleResult">
5907       <xsd:complexType/>
5908     </xsd:element>
5909
5910     <xsd:element name="AddPermissions" type="impl:AddPermissions"/>
5911     <xsd:complexType name="AddPermissions">
5912       <xsd:sequence>
5913         <xsd:element name="roleName" type="xsd:string"/>
5914         <xsd:element name="permissionNames" minOccurs="0">
5915           <xsd:complexType>
5916             <xsd:sequence>
5917               <xsd:element name="permissionName" type="xsd:string" minOccurs="0"
5918                 maxOccurs="unbounded"/>
5919             </xsd:sequence>
5920           </xsd:complexType>
5921         </xsd:element>
5922       </xsd:sequence>
5923     </xsd:complexType>
5924     <xsd:element name="AddPermissionsResult">
5925       <xsd:complexType/>
5926     </xsd:element>
5927
5928     <xsd:element name="SetPermissions" type="impl:SetPermissions"/>
5929     <xsd:complexType name="SetPermissions">
5930       <xsd:sequence>
5931         <xsd:element name="roleName" type="xsd:string"/>
5932         <xsd:element name="permissionNames" minOccurs="0">
5933           <xsd:complexType>
5934             <xsd:sequence>
5935               <xsd:element name="permissionName" type="xsd:string" minOccurs="0"
5936                 maxOccurs="unbounded"/>
5937             </xsd:sequence>
5938           </xsd:complexType>
5939         </xsd:element>
5940       </xsd:sequence>
5941     </xsd:complexType>
5942     <xsd:element name="SetPermissionsResult">
5943       <xsd:complexType/>
5944     </xsd:element>
5945

```

```

5946 <xsd:element name="RemovePermissions" type="impl:RemovePermissions"/>
5947 <xsd:complexType name="RemovePermissions">
5948 <xsd:sequence>
5949 <xsd:element name="roleName" type="xsd:string"/>
5950 <xsd:element name="permissionNames" minOccurs="0">
5951 <xsd:complexType>
5952 <xsd:sequence>
5953 <xsd:element name="permissionName" type="xsd:string" minOccurs="0"
5954 maxOccurs="unbounded"/>
5955 </xsd:sequence>
5956 </xsd:complexType>
5957 </xsd:element>
5958 </xsd:sequence>
5959 </xsd:complexType>
5960 <xsd:element name="RemovePermissionsResult">
5961 <xsd:complexType/>
5962 </xsd:element>
5963
5964 <xsd:element name="GetClientIdentityNames" type="impl:EmptyParms"/>
5965 <xsd:element name="GetClientIdentityNamesResult" type="impl:ArrayOfString"/>
5966
5967 <xsd:element name="DefineClientIdentity" type="impl:DefineClientIdentity"/>
5968 <xsd:complexType name="DefineClientIdentity">
5969 <xsd:sequence>
5970 <xsd:element name="identityName" type="xsd:string"/>
5971 <xsd:element name="id" type="ale:ACClientIdentity"/>
5972 </xsd:sequence>
5973 </xsd:complexType>
5974 <xsd:element name="DefineClientIdentityResult">
5975 <xsd:complexType/>
5976 </xsd:element>
5977
5978 <xsd:element name="UpdateClientIdentity" type="impl:UpdateClientIdentity"/>
5979 <xsd:complexType name="UpdateClientIdentity">
5980 <xsd:sequence>
5981 <xsd:element name="identityName" type="xsd:string"/>
5982 <xsd:element name="id" type="ale:ACClientIdentity"/>
5983 </xsd:sequence>
5984 </xsd:complexType>
5985 <xsd:element name="UpdateClientIdentityResult">
5986 <xsd:complexType/>
5987 </xsd:element>
5988
5989 <xsd:element name="GetClientIdentity" type="impl:GetClientIdentity"/>
5990 <xsd:complexType name="GetClientIdentity">
5991 <xsd:sequence>
5992 <xsd:element name="identityName" type="xsd:string"/>
5993 </xsd:sequence>
5994 </xsd:complexType>
5995 <xsd:element name="GetClientIdentityResult" type="ale:ACClientIdentity"/>
5996
5997 <xsd:element name="GetClientPermissionNames" type="impl:GetClientPermissionNames"/>
5998 <xsd:complexType name="GetClientPermissionNames">
5999 <xsd:sequence>
6000 <xsd:element name="identityName" type="xsd:string"/>
6001 </xsd:sequence>
6002 </xsd:complexType>
6003 <xsd:element name="GetClientPermissionNamesResult" type="impl:ArrayOfString"/>
6004
6005 <xsd:element name="UndefineClientIdentity" type="impl:UndefineClientIdentity"/>
6006 <xsd:complexType name="UndefineClientIdentity">
6007 <xsd:sequence>
6008 <xsd:element name="identityName" type="xsd:string"/>
6009 </xsd:sequence>
6010 </xsd:complexType>
6011 <xsd:element name="UndefineClientIdentityResult">
6012 <xsd:complexType/>
6013 </xsd:element>
6014
6015 <xsd:element name="AddRoles" type="impl:AddRoles"/>

```

```

6016 <xsd:complexType name="AddRoles">
6017 <xsd:sequence>
6018 <xsd:element name="identityName" type="xsd:string"/>
6019 <xsd:element name="roleNames" minOccurs="0">
6020 <xsd:complexType>
6021 <xsd:sequence>
6022 <xsd:element name="roleName" type="xsd:string" minOccurs="0"
6023 <maxOccurs="unbounded"/>
6024 </xsd:sequence>
6025 </xsd:complexType>
6026 </xsd:element>
6027 </xsd:sequence>
6028 </xsd:complexType>
6029 <xsd:element name="AddRolesResult">
6030 <xsd:complexType/>
6031 </xsd:element>
6032
6033 <xsd:element name="RemoveRoles" type="impl:RemoveRoles"/>
6034 <xsd:complexType name="RemoveRoles">
6035 <xsd:sequence>
6036 <xsd:element name="identityName" type="xsd:string"/>
6037 <xsd:element name="roleNames" minOccurs="0">
6038 <xsd:complexType>
6039 <xsd:sequence>
6040 <xsd:element name="roleName" type="xsd:string" minOccurs="0"
6041 <maxOccurs="unbounded"/>
6042 </xsd:sequence>
6043 </xsd:complexType>
6044 </xsd:element>
6045 </xsd:sequence>
6046 </xsd:complexType>
6047 <xsd:element name="RemoveRolesResult">
6048 <xsd:complexType/>
6049 </xsd:element>
6050
6051 <xsd:element name="SetRoles" type="impl:SetRoles"/>
6052 <xsd:complexType name="SetRoles">
6053 <xsd:sequence>
6054 <xsd:element name="identityName" type="xsd:string"/>
6055 <xsd:element name="roleNames" minOccurs="0">
6056 <xsd:complexType>
6057 <xsd:sequence>
6058 <xsd:element name="roleName" type="xsd:string" minOccurs="0"
6059 <maxOccurs="unbounded"/>
6060 </xsd:sequence>
6061 </xsd:complexType>
6062 </xsd:element>
6063 </xsd:sequence>
6064 </xsd:complexType>
6065 <xsd:element name="SetRolesResult">
6066 <xsd:complexType/>
6067 </xsd:element>
6068
6069 <xsd:element name="GetSupportedOperations" type="impl:EmptyParms"/>
6070 <xsd:element name="GetSupportedOperationsResult" type="impl:ArrayOfString"/>
6071
6072 <xsd:element name="GetStandardVersion" type="impl:EmptyParms"/>
6073 <xsd:element name="GetStandardVersionResult" type="xsd:string"/>
6074
6075 <xsd:element name="GetVendorVersion" type="impl:EmptyParms"/>
6076 <xsd:element name="GetVendorVersionResult" type="xsd:string"/>
6077
6078 <xsd:element name="ALEException" type="impl:ALEException"/>
6079 <xsd:complexType name="ALEException">
6080 <xsd:sequence>
6081 <xsd:element name="reason" type="xsd:string"/>
6082 </xsd:sequence>
6083 </xsd:complexType>
6084
6085 <xsd:element name="SecurityException" type="impl:SecurityException"/>

```

```

6086 <xsd:complexType name="SecurityException">
6087 <xsd:complexContent>
6088 <xsd:extension base="impl:ALEException"/>
6089 </xsd:complexContent>
6090 </xsd:complexType>
6091
6092 <xsd:element name="NoSuchPermissionException"
6093 <type="impl:NoSuchPermissionException"/>
6094 <xsd:complexType name="NoSuchPermissionException">
6095 <xsd:complexContent>
6096 <xsd:extension base="impl:ALEException"/>
6097 </xsd:complexContent>
6098 </xsd:complexType>
6099
6100 <xsd:element name="PermissionValidationException"
6101 <type="impl:PermissionValidationException"/>
6102 <xsd:complexType name="PermissionValidationException">
6103 <xsd:complexContent>
6104 <xsd:extension base="impl:ALEException"/>
6105 </xsd:complexContent>
6106 </xsd:complexType>
6107
6108 <xsd:element name="DuplicatePermissionException"
6109 <type="impl:DuplicatePermissionException"/>
6110 <xsd:complexType name="DuplicatePermissionException">
6111 <xsd:complexContent>
6112 <xsd:extension base="impl:ALEException"/>
6113 </xsd:complexContent>
6114 </xsd:complexType>
6115
6116 <xsd:element name="NoSuchRoleException" type="impl:NoSuchRoleException"/>
6117 <xsd:complexType name="NoSuchRoleException">
6118 <xsd:complexContent>
6119 <xsd:extension base="impl:ALEException"/>
6120 </xsd:complexContent>
6121 </xsd:complexType>
6122
6123 <xsd:element name="RoleValidationException" type="impl:RoleValidationException"/>
6124 <xsd:complexType name="RoleValidationException">
6125 <xsd:complexContent>
6126 <xsd:extension base="impl:ALEException"/>
6127 </xsd:complexContent>
6128 </xsd:complexType>
6129
6130 <xsd:element name="DuplicateRoleException" type="impl:DuplicateRoleException"/>
6131 <xsd:complexType name="DuplicateRoleException">
6132 <xsd:complexContent>
6133 <xsd:extension base="impl:ALEException"/>
6134 </xsd:complexContent>
6135 </xsd:complexType>
6136
6137 <xsd:element name="NoSuchClientIdentityException"
6138 <type="impl:NoSuchClientIdentityException"/>
6139 <xsd:complexType name="NoSuchClientIdentityException">
6140 <xsd:complexContent>
6141 <xsd:extension base="impl:ALEException"/>
6142 </xsd:complexContent>
6143 </xsd:complexType>
6144
6145 <xsd:element name="ClientIdentityValidationException"
6146 <type="impl:ClientIdentityValidationException"/>
6147 <xsd:complexType name="ClientIdentityValidationException">
6148 <xsd:complexContent>
6149 <xsd:extension base="impl:ALEException"/>
6150 </xsd:complexContent>
6151 </xsd:complexType>
6152
6153 <xsd:element name="DuplicateClientIdentityException"
6154 <type="impl:DuplicateClientIdentityException"/>
6155 <xsd:complexType name="DuplicateClientIdentityException">

```

```

6156     <xsd:complexContent>
6157         <xsd:extension base="impl:ALEException"/>
6158     </xsd:complexContent>
6159 </xsd:complexType>
6160
6161 <xsd:element name="UnsupportedOperationException"
6162             type="impl:UnsupportedOperationException"/>
6163 <xsd:complexType name="UnsupportedOperationException">
6164     <xsd:complexContent>
6165         <xsd:extension base="impl:ALEException"/>
6166     </xsd:complexContent>
6167 </xsd:complexType>
6168
6169 <xsd:element name="ImplementationException" type="impl:ImplementationException"/>
6170 <xsd:complexType name="ImplementationException">
6171     <xsd:complexContent>
6172         <xsd:extension base="impl:ALEException">
6173             <xsd:sequence>
6174                 <xsd:element name="severity" type="impl:ImplementationExceptionSeverity"/>
6175             </xsd:sequence>
6176         </xsd:extension>
6177     </xsd:complexContent>
6178 </xsd:complexType>
6179
6180 <xsd:complexType name="ArrayOfString">
6181     <xsd:sequence>
6182         <xsd:element name="string" type="xsd:string" minOccurs="0"
6183                 maxOccurs="unbounded"/>
6184     </xsd:sequence>
6185 </xsd:complexType>
6186
6187 <!-- The ImplementationExceptionSeverity type is an enumerated type.
6188      The following strings are legal values for this type:
6189          ERROR
6190          SEVERE
6191      -->
6192 <xsd:simpleType name="ImplementationExceptionSeverity">
6193     <xsd:restriction base="xsd:string"/>
6194 </xsd:simpleType>
6195
6196     <xsd:complexType name="EmptyParms"/>
6197 </xsd:schema>
6198 </wsdl:types>
6199 <!-- ALEACSERVICE MESSAGES -->
6200
6201 <wsdl:message name="getPermissionNamesRequest">
6202     <wsdl:part name="parms" element="impl:GetPermissionNames"/>
6203 </wsdl:message>
6204 <wsdl:message name="getPermissionNamesResponse">
6205     <wsdl:part name="getPermissionNamesReturn" element="impl:GetPermissionNamesResult"/>
6206 </wsdl:message>
6207
6208 <wsdl:message name="definePermissionRequest">
6209     <wsdl:part name="parms" element="impl:DefinePermission"/>
6210 </wsdl:message>
6211 <wsdl:message name="definePermissionResponse">
6212     <wsdl:part name="definePermissionReturn" element="impl:DefinePermissionResult"/>
6213 </wsdl:message>
6214
6215 <wsdl:message name="updatePermissionRequest">
6216     <wsdl:part name="parms" element="impl:UpdatePermission"/>
6217 </wsdl:message>
6218 <wsdl:message name="updatePermissionResponse">
6219     <wsdl:part name="updatePermissionReturn" element="impl:UpdatePermissionResult"/>
6220 </wsdl:message>
6221
6222 <wsdl:message name="getPermissionRequest">
6223     <wsdl:part name="parms" element="impl:GetPermission"/>
6224 </wsdl:message>
6225 <wsdl:message name="getPermissionResponse">

```



```

6226     <wsdl:part name="getPermissionReturn" element="impl:GetPermissionResult"/>
6227 </wsdl:message>
6228
6229 <wsdl:message name="undefinePermissionRequest">
6230   <wsdl:part name="parms" element="impl:UndefinePermission"/>
6231 </wsdl:message>
6232 <wsdl:message name="undefinePermissionResponse">
6233   <wsdl:part name="undefinePermissionReturn" element="impl:UndefinePermissionResult"/>
6234 </wsdl:message>
6235
6236 <wsdl:message name="getRoleNamesRequest">
6237   <wsdl:part name="parms" element="impl:GetRoleNames"/>
6238 </wsdl:message>
6239 <wsdl:message name="getRoleNamesResponse">
6240   <wsdl:part name="getRoleNamesReturn" element="impl:GetRoleNamesResult"/>
6241 </wsdl:message>
6242
6243 <wsdl:message name="defineRoleRequest">
6244   <wsdl:part name="parms" element="impl:DefineRole"/>
6245 </wsdl:message>
6246 <wsdl:message name="defineRoleResponse">
6247   <wsdl:part name="defineRoleReturn" element="impl:DefineRoleResult"/>
6248 </wsdl:message>
6249
6250 <wsdl:message name="updateRoleRequest">
6251   <wsdl:part name="parms" element="impl:UpdateRole"/>
6252 </wsdl:message>
6253 <wsdl:message name="updateRoleResponse">
6254   <wsdl:part name="updateRoleReturn" element="impl:UpdateRoleResult"/>
6255 </wsdl:message>
6256
6257 <wsdl:message name="getRoleRequest">
6258   <wsdl:part name="parms" element="impl:GetRole"/>
6259 </wsdl:message>
6260 <wsdl:message name="getRoleResponse">
6261   <wsdl:part name="getRoleReturn" element="impl:GetRoleResult"/>
6262 </wsdl:message>
6263
6264 <wsdl:message name="undefineRoleRequest">
6265   <wsdl:part name="parms" element="impl:UndefineRole"/>
6266 </wsdl:message>
6267 <wsdl:message name="undefineRoleResponse">
6268   <wsdl:part name="undefineRoleReturn" element="impl:UndefineRoleResult"/>
6269 </wsdl:message>
6270
6271 <wsdl:message name="addPermissionsRequest">
6272   <wsdl:part name="parms" element="impl:AddPermissions"/>
6273 </wsdl:message>
6274 <wsdl:message name="addPermissionsResponse">
6275   <wsdl:part name="addPermissionsReturn" element="impl:AddPermissionsResult"/>
6276 </wsdl:message>
6277
6278 <wsdl:message name="setPermissionsRequest">
6279   <wsdl:part name="parms" element="impl:SetPermissions"/>
6280 </wsdl:message>
6281 <wsdl:message name="setPermissionsResponse">
6282   <wsdl:part name="setPermissionsReturn" element="impl:SetPermissionsResult"/>
6283 </wsdl:message>
6284
6285 <wsdl:message name="removePermissionsRequest">
6286   <wsdl:part name="parms" element="impl:RemovePermissions"/>
6287 </wsdl:message>
6288 <wsdl:message name="removePermissionsResponse">
6289   <wsdl:part name="removePermissionsReturn" element="impl:RemovePermissionsResult"/>
6290 </wsdl:message>
6291
6292 <wsdl:message name="getClientIdentityNamesRequest">
6293   <wsdl:part name="parms" element="impl:GetClientIdentityNames"/>
6294 </wsdl:message>
6295 <wsdl:message name="getClientIdentityNamesResponse">

```

```

6296     <wsdl:part name="getClientIdentityNamesReturn"
6297         element="impl:GetClientIdentityNamesResult"/>
6298 </wsdl:message>
6299
6300 <wsdl:message name="defineClientIdentityRequest">
6301     <wsdl:part name="parms" element="impl:DefineClientIdentity"/>
6302 </wsdl:message>
6303 <wsdl:message name="defineClientIdentityResponse">
6304     <wsdl:part name="defineClientIdentityReturn"
6305         element="impl:DefineClientIdentityResult"/>
6306 </wsdl:message>
6307
6308 <wsdl:message name="updateClientIdentityRequest">
6309     <wsdl:part name="parms" element="impl:UpdateClientIdentity"/>
6310 </wsdl:message>
6311 <wsdl:message name="updateClientIdentityResponse">
6312     <wsdl:part name="updateClientIdentityReturn"
6313         element="impl:UpdateClientIdentityResult"/>
6314 </wsdl:message>
6315
6316 <wsdl:message name="getClientIdentityRequest">
6317     <wsdl:part name="parms" element="impl:GetClientIdentity"/>
6318 </wsdl:message>
6319 <wsdl:message name="getClientIdentityResponse">
6320     <wsdl:part name="getClientIdentityReturn" element="impl:GetClientIdentityResult"/>
6321 </wsdl:message>
6322
6323 <wsdl:message name="getClientPermissionNamesRequest">
6324     <wsdl:part name="parms" element="impl:GetClientPermissionNames"/>
6325 </wsdl:message>
6326 <wsdl:message name="getClientPermissionNamesResponse">
6327     <wsdl:part name="getClientPermissionNamesReturn"
6328         element="impl:GetClientPermissionNamesResult"/>
6329 </wsdl:message>
6330
6331 <wsdl:message name="undefineClientIdentityRequest">
6332     <wsdl:part name="parms" element="impl:UndefineClientIdentity"/>
6333 </wsdl:message>
6334 <wsdl:message name="undefineClientIdentityResponse">
6335     <wsdl:part name="undefineClientIdentityReturn"
6336         element="impl:UndefineClientIdentityResult"/>
6337 </wsdl:message>
6338
6339 <wsdl:message name="addRolesRequest">
6340     <wsdl:part name="parms" element="impl:AddRoles"/>
6341 </wsdl:message>
6342 <wsdl:message name="addRolesResponse">
6343     <wsdl:part name="addRolesReturn" element="impl:AddRolesResult"/>
6344 </wsdl:message>
6345
6346 <wsdl:message name="removeRolesRequest">
6347     <wsdl:part name="parms" element="impl:RemoveRoles"/>
6348 </wsdl:message>
6349 <wsdl:message name="removeRolesResponse">
6350     <wsdl:part name="removeRolesReturn" element="impl:RemoveRolesResult"/>
6351 </wsdl:message>
6352
6353 <wsdl:message name="setRolesRequest">
6354     <wsdl:part name="parms" element="impl:SetRoles"/>
6355 </wsdl:message>
6356 <wsdl:message name="setRolesResponse">
6357     <wsdl:part name="setRolesReturn" element="impl:SetRolesResult"/>
6358 </wsdl:message>
6359
6360 <wsdl:message name="getSupportedOperationsRequest">
6361     <wsdl:part name="parms" element="impl:GetSupportedOperations"/>
6362 </wsdl:message>
6363 <wsdl:message name="getSupportedOperationsResponse">
6364     <wsdl:part name="getSupportedOperationsReturn"
6365         element="impl:GetSupportedOperationsResult"/>

```

```

6366 </wsdl:message>
6367
6368 <wsdl:message name="getStandardVersionRequest">
6369   <wsdl:part name="parms" element="impl:GetStandardVersion"/>
6370 </wsdl:message>
6371 <wsdl:message name="getStandardVersionResponse">
6372   <wsdl:part name="getStandardVersionReturn" element="impl:GetStandardVersionResult"/>
6373 </wsdl:message>
6374
6375 <wsdl:message name="getVendorVersionRequest">
6376   <wsdl:part name="parms" element="impl:GetVendorVersion"/>
6377 </wsdl:message>
6378 <wsdl:message name="getVendorVersionResponse">
6379   <wsdl:part name="getVendorVersionReturn" element="impl:GetVendorVersionResult"/>
6380 </wsdl:message>
6381
6382 <wsdl:message name="SecurityExceptionResponse">
6383   <wsdl:part name="fault" element="impl:SecurityException"/>
6384 </wsdl:message>
6385
6386 <wsdl:message name="NoSuchPermissionExceptionResponse">
6387   <wsdl:part name="fault" element="impl:NoSuchPermissionException"/>
6388 </wsdl:message>
6389
6390 <wsdl:message name="PermissionValidationExceptionResponse">
6391   <wsdl:part name="fault" element="impl:PermissionValidationException"/>
6392 </wsdl:message>
6393
6394 <wsdl:message name="DuplicatePermissionExceptionResponse">
6395   <wsdl:part name="fault" element="impl:DuplicatePermissionException"/>
6396 </wsdl:message>
6397
6398 <wsdl:message name="NoSuchRoleExceptionResponse">
6399   <wsdl:part name="fault" element="impl:NoSuchRoleException"/>
6400 </wsdl:message>
6401
6402 <wsdl:message name="RoleValidationExceptionResponse">
6403   <wsdl:part name="fault" element="impl:RoleValidationException"/>
6404 </wsdl:message>
6405
6406 <wsdl:message name="DuplicateRoleExceptionResponse">
6407   <wsdl:part name="fault" element="impl:DuplicateRoleException"/>
6408 </wsdl:message>
6409
6410 <wsdl:message name="NoSuchClientIdentityExceptionResponse">
6411   <wsdl:part name="fault" element="impl:NoSuchClientIdentityException"/>
6412 </wsdl:message>
6413
6414 <wsdl:message name="ClientIdentityValidationExceptionResponse">
6415   <wsdl:part name="fault" element="impl:ClientIdentityValidationException"/>
6416 </wsdl:message>
6417
6418 <wsdl:message name="DuplicateClientIdentityExceptionResponse">
6419   <wsdl:part name="fault" element="impl:DuplicateClientIdentityException"/>
6420 </wsdl:message>
6421
6422 <wsdl:message name="UnsupportedOperationExceptionResponse">
6423   <wsdl:part name="fault" element="impl:UnsupportedOperationException"/>
6424 </wsdl:message>
6425
6426 <wsdl:message name="ImplementationExceptionResponse">
6427   <wsdl:part name="fault" element="impl:ImplementationException"/>
6428 </wsdl:message>
6429 <!-- ALEACSERVICE PORTTYPE -->
6430
6431 <wsdl:portType name="ALEACServicePortType">
6432
6433   <wsdl:operation name="getPermissionNames">
6434     <wsdl:input message="impl:getPermissionNamesRequest"
6435       name="getPermissionNamesRequest"/>

```

```

6436     <wsdl:output message="impl:getPermissionNamesResponse"
6437         name="getPermissionNamesResponse" />
6438     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6439         name="UnsupportedOperationExceptionFault" />
6440     <wsdl:fault message="impl:SecurityExceptionResponse"
6441         name="SecurityExceptionFault" />
6442     <wsdl:fault message="impl:ImplementationExceptionResponse"
6443         name="ImplementationExceptionFault" />
6444 </wsdl:operation>
6445
6446 <wsdl:operation name="definePermission">
6447     <wsdl:input message="impl:definePermissionRequest" name="definePermissionRequest" />
6448     <wsdl:output message="impl:definePermissionResponse"
6449         name="definePermissionResponse" />
6450     <wsdl:fault message="impl:SecurityExceptionResponse"
6451         name="SecurityExceptionFault" />
6452     <wsdl:fault message="impl:DuplicatePermissionExceptionResponse"
6453         name="DuplicatePermissionExceptionFault" />
6454     <wsdl:fault message="impl:PermissionValidationExceptionResponse"
6455         name="PermissionValidationExceptionFault" />
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="updatePermission">
6463     <wsdl:input message="impl:updatePermissionRequest" name="updatePermissionRequest" />
6464     <wsdl:output message="impl:updatePermissionResponse"
6465         name="updatePermissionResponse" />
6466     <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6467         name="NoSuchPermissionExceptionFault" />
6468     <wsdl:fault message="impl:PermissionValidationExceptionResponse"
6469         name="PermissionValidationExceptionFault" />
6470     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6471         name="UnsupportedOperationExceptionFault" />
6472     <wsdl:fault message="impl:SecurityExceptionResponse"
6473         name="SecurityExceptionFault" />
6474     <wsdl:fault message="impl:ImplementationExceptionResponse"
6475         name="ImplementationExceptionFault" />
6476 </wsdl:operation>
6477
6478 <wsdl:operation name="getPermission">
6479     <wsdl:input message="impl:getPermissionRequest" name="getPermissionRequest" />
6480     <wsdl:output message="impl:getPermissionResponse" name="getPermissionResponse" />
6481     <wsdl:fault message="impl:SecurityExceptionResponse"
6482         name="SecurityExceptionFault" />
6483     <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6484         name="NoSuchPermissionExceptionFault" />
6485     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6486         name="UnsupportedOperationExceptionFault" />
6487     <wsdl:fault message="impl:ImplementationExceptionResponse"
6488         name="ImplementationExceptionFault" />
6489 </wsdl:operation>
6490
6491 <wsdl:operation name="undefinePermission">
6492     <wsdl:input message="impl:undefinePermissionRequest"
6493         name="undefinePermissionRequest" />
6494     <wsdl:output message="impl:undefinePermissionResponse"
6495         name="undefinePermissionResponse" />
6496     <wsdl:fault message="impl:SecurityExceptionResponse"
6497         name="SecurityExceptionFault" />
6498     <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6499         name="NoSuchPermissionExceptionFault" />
6500     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6501         name="UnsupportedOperationExceptionFault" />
6502     <wsdl:fault message="impl:ImplementationExceptionResponse"
6503         name="ImplementationExceptionFault" />
6504 </wsdl:operation>
6505

```

```

6506 <wsdl:operation name="getRoleNames">
6507 <wsdl:input message="impl:getRoleNamesRequest" name="getRoleNamesRequest" />
6508 <wsdl:output message="impl:getRoleNamesResponse" name="getRoleNamesResponse" />
6509 <wsdl:fault message="impl:SecurityExceptionResponse"
6510 name="SecurityExceptionFault" />
6511 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6512 name="UnsupportedOperationExceptionFault" />
6513 <wsdl:fault message="impl:ImplementationExceptionResponse"
6514 name="ImplementationExceptionFault" />
6515 </wsdl:operation>
6516
6517 <wsdl:operation name="defineRole">
6518 <wsdl:input message="impl:defineRoleRequest" name="defineRoleRequest" />
6519 <wsdl:output message="impl:defineRoleResponse" name="defineRoleResponse" />
6520 <wsdl:fault message="impl:SecurityExceptionResponse"
6521 name="SecurityExceptionFault" />
6522 <wsdl:fault message="impl:DuplicateRoleExceptionResponse"
6523 name="DuplicateRoleExceptionFault" />
6524 <wsdl:fault message="impl:RoleValidationExceptionResponse"
6525 name="RoleValidationExceptionFault" />
6526 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6527 name="UnsupportedOperationExceptionFault" />
6528 <wsdl:fault message="impl:ImplementationExceptionResponse"
6529 name="ImplementationExceptionFault" />
6530 </wsdl:operation>
6531
6532 <wsdl:operation name="updateRole">
6533 <wsdl:input message="impl:updateRoleRequest" name="updateRoleRequest" />
6534 <wsdl:output message="impl:updateRoleResponse" name="updateRoleResponse" />
6535 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6536 name="NoSuchRoleExceptionFault" />
6537 <wsdl:fault message="impl:RoleValidationExceptionResponse"
6538 name="RoleValidationExceptionFault" />
6539 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6540 name="UnsupportedOperationExceptionFault" />
6541 <wsdl:fault message="impl:SecurityExceptionResponse"
6542 name="SecurityExceptionFault" />
6543 <wsdl:fault message="impl:ImplementationExceptionResponse"
6544 name="ImplementationExceptionFault" />
6545 </wsdl:operation>
6546
6547 <wsdl:operation name="getRole">
6548 <wsdl:input message="impl:getRoleRequest" name="getRoleRequest" />
6549 <wsdl:output message="impl:getRoleResponse" name="getRoleResponse" />
6550 <wsdl:fault message="impl:SecurityExceptionResponse"
6551 name="SecurityExceptionFault" />
6552 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6553 name="NoSuchRoleExceptionFault" />
6554 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6555 name="UnsupportedOperationExceptionFault" />
6556 <wsdl:fault message="impl:ImplementationExceptionResponse"
6557 name="ImplementationExceptionFault" />
6558 </wsdl:operation>
6559
6560 <wsdl:operation name="undefineRole">
6561 <wsdl:input message="impl:undefineRoleRequest" name="undefineRoleRequest" />
6562 <wsdl:output message="impl:undefineRoleResponse" name="undefineRoleResponse" />
6563 <wsdl:fault message="impl:SecurityExceptionResponse"
6564 name="SecurityExceptionFault" />
6565 <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6566 name="NoSuchRoleExceptionFault" />
6567 <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6568 name="UnsupportedOperationExceptionFault" />
6569 <wsdl:fault message="impl:ImplementationExceptionResponse"
6570 name="ImplementationExceptionFault" />
6571 </wsdl:operation>
6572
6573 <wsdl:operation name="addPermissions">
6574 <wsdl:input message="impl:addPermissionsRequest" name="addPermissionsRequest" />
6575 <wsdl:output message="impl:addPermissionsResponse" name="addPermissionsResponse" />

```

```

6576     <wsdl:fault message="impl:SecurityExceptionResponse"
6577             name="SecurityExceptionFault" />
6578     <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6579             name="NoSuchRoleExceptionFault" />
6580     <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6581             name="NoSuchPermissionExceptionFault" />
6582     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6583             name="UnsupportedOperationExceptionFault" />
6584     <wsdl:fault message="impl:ImplementationExceptionResponse"
6585             name="ImplementationExceptionFault" />
6586 </wsdl:operation>
6587
6588 <wsdl:operation name="setPermissions">
6589     <wsdl:input message="impl:setPermissionsRequest" name="setPermissionsRequest" />
6590     <wsdl:output message="impl:setPermissionsResponse" name="setPermissionsResponse" />
6591     <wsdl:fault message="impl:SecurityExceptionResponse"
6592             name="SecurityExceptionFault" />
6593     <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6594             name="NoSuchRoleExceptionFault" />
6595     <wsdl:fault message="impl:NoSuchPermissionExceptionResponse"
6596             name="NoSuchPermissionExceptionFault" />
6597     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6598             name="UnsupportedOperationExceptionFault" />
6599     <wsdl:fault message="impl:ImplementationExceptionResponse"
6600             name="ImplementationExceptionFault" />
6601 </wsdl:operation>
6602
6603 <wsdl:operation name="removePermissions">
6604     <wsdl:input message="impl:removePermissionsRequest"
6605             name="removePermissionsRequest" />
6606     <wsdl:output message="impl:removePermissionsResponse"
6607             name="removePermissionsResponse" />
6608     <wsdl:fault message="impl:SecurityExceptionResponse"
6609             name="SecurityExceptionFault" />
6610     <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6611             name="NoSuchRoleExceptionFault" />
6612     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6613             name="UnsupportedOperationExceptionFault" />
6614     <wsdl:fault message="impl:ImplementationExceptionResponse"
6615             name="ImplementationExceptionFault" />
6616 </wsdl:operation>
6617
6618 <wsdl:operation name="getClientIdentityNames">
6619     <wsdl:input message="impl:getClientIdentityNamesRequest"
6620             name="getClientIdentityNamesRequest" />
6621     <wsdl:output message="impl:getClientIdentityNamesResponse"
6622             name="getClientIdentityNamesResponse" />
6623     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6624             name="UnsupportedOperationExceptionFault" />
6625     <wsdl:fault message="impl:SecurityExceptionResponse"
6626             name="SecurityExceptionFault" />
6627     <wsdl:fault message="impl:ImplementationExceptionResponse"
6628             name="ImplementationExceptionFault" />
6629 </wsdl:operation>
6630
6631 <wsdl:operation name="defineClientIdentity">
6632     <wsdl:input message="impl:defineClientIdentityRequest"
6633             name="defineClientIdentityRequest" />
6634     <wsdl:output message="impl:defineClientIdentityResponse"
6635             name="defineClientIdentityResponse" />
6636     <wsdl:fault message="impl:SecurityExceptionResponse"
6637             name="SecurityExceptionFault" />
6638     <wsdl:fault message="impl:DuplicateClientIdentityExceptionResponse"
6639             name="DuplicateClientIdentityExceptionFault" />
6640     <wsdl:fault message="impl:ClientIdentityValidationExceptionResponse"
6641             name="ClientIdentityValidationExceptionFault" />
6642     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6643             name="UnsupportedOperationExceptionFault" />
6644     <wsdl:fault message="impl:ImplementationExceptionResponse"
6645             name="ImplementationExceptionFault" />

```



```

6646 </wsdl:operation>
6647
6648 <wsdl:operation name="updateClientIdentity">
6649   <wsdl:input message="impl:updateClientIdentityRequest"
6650     name="updateClientIdentityRequest" />
6651   <wsdl:output message="impl:updateClientIdentityResponse"
6652     name="updateClientIdentityResponse" />
6653   <wsdl:fault message="impl:SecurityExceptionResponse"
6654     name="SecurityExceptionFault" />
6655   <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6656     name="NoSuchClientIdentityExceptionFault" />
6657   <wsdl:fault message="impl:ClientIdentityValidationExceptionResponse"
6658     name="ClientIdentityValidationExceptionFault" />
6659   <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6660     name="UnsupportedOperationExceptionFault" />
6661   <wsdl:fault message="impl:ImplementationExceptionResponse"
6662     name="ImplementationExceptionFault" />
6663 </wsdl:operation>
6664
6665 <wsdl:operation name="getClientIdentity">
6666   <wsdl:input message="impl:getClientIdentityRequest"
6667     name="getClientIdentityRequest" />
6668   <wsdl:output message="impl:getClientIdentityResponse"
6669     name="getClientIdentityResponse" />
6670   <wsdl:fault message="impl:SecurityExceptionResponse"
6671     name="SecurityExceptionFault" />
6672   <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6673     name="NoSuchClientIdentityExceptionFault" />
6674   <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6675     name="UnsupportedOperationExceptionFault" />
6676   <wsdl:fault message="impl:ImplementationExceptionResponse"
6677     name="ImplementationExceptionFault" />
6678 </wsdl:operation>
6679
6680 <wsdl:operation name="getClientPermissionNames">
6681   <wsdl:input message="impl:getClientPermissionNamesRequest"
6682     name="getClientPermissionNamesRequest" />
6683   <wsdl:output message="impl:getClientPermissionNamesResponse"
6684     name="getClientPermissionNamesResponse" />
6685   <wsdl:fault message="impl:SecurityExceptionResponse"
6686     name="SecurityExceptionFault" />
6687   <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6688     name="NoSuchClientIdentityExceptionFault" />
6689   <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6690     name="UnsupportedOperationExceptionFault" />
6691   <wsdl:fault message="impl:ImplementationExceptionResponse"
6692     name="ImplementationExceptionFault" />
6693 </wsdl:operation>
6694
6695 <wsdl:operation name="undefineClientIdentity">
6696   <wsdl:input message="impl:undefineClientIdentityRequest"
6697     name="undefineClientIdentityRequest" />
6698   <wsdl:output message="impl:undefineClientIdentityResponse"
6699     name="undefineClientIdentityResponse" />
6700   <wsdl:fault message="impl:SecurityExceptionResponse"
6701     name="SecurityExceptionFault" />
6702   <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6703     name="NoSuchClientIdentityExceptionFault" />
6704   <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6705     name="UnsupportedOperationExceptionFault" />
6706   <wsdl:fault message="impl:ImplementationExceptionResponse"
6707     name="ImplementationExceptionFault" />
6708 </wsdl:operation>
6709
6710 <wsdl:operation name="addRoles">
6711   <wsdl:input message="impl:addRolesRequest" name="addRolesRequest" />
6712   <wsdl:output message="impl:addRolesResponse" name="addRolesResponse" />
6713   <wsdl:fault message="impl:SecurityExceptionResponse"
6714     name="SecurityExceptionFault" />
6715   <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"

```



```

6716         name="NoSuchClientIdentityExceptionFault" />
6717     <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6718         name="NoSuchRoleExceptionFault" />
6719     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6720         name="UnsupportedOperationExceptionFault" />
6721     <wsdl:fault message="impl:ImplementationExceptionResponse"
6722         name="ImplementationExceptionFault" />
6723 </wsdl:operation>
6724
6725 <wsdl:operation name="removeRoles">
6726     <wsdl:input message="impl:removeRolesRequest" name="removeRolesRequest" />
6727     <wsdl:output message="impl:removeRolesResponse" name="removeRolesResponse" />
6728     <wsdl:fault message="impl:SecurityExceptionResponse"
6729         name="SecurityExceptionFault" />
6730     <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6731         name="NoSuchClientIdentityExceptionFault" />
6732     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6733         name="UnsupportedOperationExceptionFault" />
6734     <wsdl:fault message="impl:ImplementationExceptionResponse"
6735         name="ImplementationExceptionFault" />
6736 </wsdl:operation>
6737
6738 <wsdl:operation name="setRoles">
6739     <wsdl:input message="impl:setRolesRequest" name="setRolesRequest" />
6740     <wsdl:output message="impl:setRolesResponse" name="setRolesResponse" />
6741     <wsdl:fault message="impl:SecurityExceptionResponse"
6742         name="SecurityExceptionFault" />
6743     <wsdl:fault message="impl:NoSuchClientIdentityExceptionResponse"
6744         name="NoSuchClientIdentityExceptionFault" />
6745     <wsdl:fault message="impl:NoSuchRoleExceptionResponse"
6746         name="NoSuchRoleExceptionFault" />
6747     <wsdl:fault message="impl:UnsupportedOperationExceptionResponse"
6748         name="UnsupportedOperationExceptionFault" />
6749     <wsdl:fault message="impl:ImplementationExceptionResponse"
6750         name="ImplementationExceptionFault" />
6751 </wsdl:operation>
6752
6753 <wsdl:operation name="getSupportedOperations">
6754     <wsdl:input message="impl:getSupportedOperationsRequest"
6755         name="getSupportedOperationsRequest" />
6756     <wsdl:output message="impl:getSupportedOperationsResponse"
6757         name="getSupportedOperationsResponse" />
6758     <wsdl:fault message="impl:ImplementationExceptionResponse"
6759         name="ImplementationExceptionFault" />
6760 </wsdl:operation>
6761
6762 <wsdl:operation name="getStandardVersion">
6763     <wsdl:input message="impl:getStandardVersionRequest"
6764         name="getStandardVersionRequest" />
6765     <wsdl:output message="impl:getStandardVersionResponse"
6766         name="getStandardVersionResponse" />
6767     <wsdl:fault message="impl:ImplementationExceptionResponse"
6768         name="ImplementationExceptionFault" />
6769 </wsdl:operation>
6770
6771 <wsdl:operation name="getVendorVersion">
6772     <wsdl:input message="impl:getVendorVersionRequest" name="getVendorVersionRequest" />
6773     <wsdl:output message="impl:getVendorVersionResponse"
6774         name="getVendorVersionResponse" />
6775     <wsdl:fault message="impl:ImplementationExceptionResponse"
6776         name="ImplementationExceptionFault" />
6777 </wsdl:operation>
6778 </wsdl:portType>
6779 <!-- ALEACSERVICE BINDING -->
6780
6781 <wsdl:binding name="ALEACServiceBinding" type="impl:ALEACServicePortType">
6782     <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
6783
6784     <wsdl:operation name="getPermissionNames">
6785         <wsdlsoap:operation soapAction="" />

```

```

6786     <wsdl:input name="getPermissionNamesRequest">
6787         <wsdlsoap:body use="literal"/>
6788     </wsdl:input>
6789     <wsdl:output name="getPermissionNamesResponse">
6790         <wsdlsoap:body use="literal"/>
6791     </wsdl:output>
6792     <wsdl:fault name="UnsupportedOperationExceptionFault">
6793         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6794     </wsdl:fault>
6795     <wsdl:fault name="SecurityExceptionFault">
6796         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6797     </wsdl:fault>
6798     <wsdl:fault name="ImplementationExceptionFault">
6799         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6800     </wsdl:fault>
6801 </wsdl:operation>
6802
6803 <wsdl:operation name="definePermission">
6804     <wsdlsoap:operation soapAction=""/>
6805     <wsdl:input name="definePermissionRequest">
6806         <wsdlsoap:body use="literal"/>
6807     </wsdl:input>
6808     <wsdl:output name="definePermissionResponse">
6809         <wsdlsoap:body use="literal"/>
6810     </wsdl:output>
6811     <wsdl:fault name="SecurityExceptionFault">
6812         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6813     </wsdl:fault>
6814     <wsdl:fault name="DuplicatePermissionExceptionFault">
6815         <wsdlsoap:fault name="DuplicatePermissionExceptionFault" use="literal"/>
6816     </wsdl:fault>
6817     <wsdl:fault name="PermissionValidationExceptionFault">
6818         <wsdlsoap:fault name="PermissionValidationExceptionFault" use="literal"/>
6819     </wsdl:fault>
6820     <wsdl:fault name="UnsupportedOperationExceptionFault">
6821         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6822     </wsdl:fault>
6823     <wsdl:fault name="ImplementationExceptionFault">
6824         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6825     </wsdl:fault>
6826 </wsdl:operation>
6827
6828 <wsdl:operation name="updatePermission">
6829     <wsdlsoap:operation soapAction=""/>
6830     <wsdl:input name="updatePermissionRequest">
6831         <wsdlsoap:body use="literal"/>
6832     </wsdl:input>
6833     <wsdl:output name="updatePermissionResponse">
6834         <wsdlsoap:body use="literal"/>
6835     </wsdl:output>
6836     <wsdl:fault name="NoSuchPermissionExceptionFault">
6837         <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
6838     </wsdl:fault>
6839     <wsdl:fault name="PermissionValidationExceptionFault">
6840         <wsdlsoap:fault name="PermissionValidationExceptionFault" use="literal"/>
6841     </wsdl:fault>
6842     <wsdl:fault name="UnsupportedOperationExceptionFault">
6843         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6844     </wsdl:fault>
6845     <wsdl:fault name="SecurityExceptionFault">
6846         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6847     </wsdl:fault>
6848     <wsdl:fault name="ImplementationExceptionFault">
6849         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6850     </wsdl:fault>
6851 </wsdl:operation>
6852
6853 <wsdl:operation name="getPermission">
6854     <wsdlsoap:operation soapAction=""/>
6855     <wsdl:input name="getPermissionRequest">

```

```

6856     <wsdlsoap:body use="literal"/>
6857 </wsdl:input>
6858 <wsdl:output name="getPermissionResponse">
6859   <wsdlsoap:body use="literal"/>
6860 </wsdl:output>
6861 <wsdl:fault name="SecurityExceptionFault">
6862   <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6863 </wsdl:fault>
6864 <wsdl:fault name="NoSuchPermissionExceptionFault">
6865   <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
6866 </wsdl:fault>
6867 <wsdl:fault name="UnsupportedOperationExceptionFault">
6868   <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6869 </wsdl:fault>
6870 <wsdl:fault name="ImplementationExceptionFault">
6871   <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6872 </wsdl:fault>
6873 </wsdl:operation>
6874
6875 <wsdl:operation name="undefinePermission">
6876   <wsdlsoap:operation soapAction=""/>
6877   <wsdl:input name="undefinePermissionRequest">
6878     <wsdlsoap:body use="literal"/>
6879   </wsdl:input>
6880   <wsdl:output name="undefinePermissionResponse">
6881     <wsdlsoap:body use="literal"/>
6882   </wsdl:output>
6883   <wsdl:fault name="SecurityExceptionFault">
6884     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6885   </wsdl:fault>
6886   <wsdl:fault name="NoSuchPermissionExceptionFault">
6887     <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
6888   </wsdl:fault>
6889   <wsdl:fault name="UnsupportedOperationExceptionFault">
6890     <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6891   </wsdl:fault>
6892   <wsdl:fault name="ImplementationExceptionFault">
6893     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6894   </wsdl:fault>
6895 </wsdl:operation>
6896
6897 <wsdl:operation name="getRoleNames">
6898   <wsdlsoap:operation soapAction=""/>
6899   <wsdl:input name="getRoleNamesRequest">
6900     <wsdlsoap:body use="literal"/>
6901   </wsdl:input>
6902   <wsdl:output name="getRoleNamesResponse">
6903     <wsdlsoap:body use="literal"/>
6904   </wsdl:output>
6905   <wsdl:fault name="SecurityExceptionFault">
6906     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6907   </wsdl:fault>
6908   <wsdl:fault name="UnsupportedOperationExceptionFault">
6909     <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6910   </wsdl:fault>
6911   <wsdl:fault name="ImplementationExceptionFault">
6912     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6913   </wsdl:fault>
6914 </wsdl:operation>
6915
6916 <wsdl:operation name="defineRole">
6917   <wsdlsoap:operation soapAction=""/>
6918   <wsdl:input name="defineRoleRequest">
6919     <wsdlsoap:body use="literal"/>
6920   </wsdl:input>
6921   <wsdl:output name="defineRoleResponse">
6922     <wsdlsoap:body use="literal"/>
6923   </wsdl:output>
6924   <wsdl:fault name="SecurityExceptionFault">
6925     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>

```

```

6926     </wsdl:fault>
6927     <wsdl:fault name="DuplicateRoleExceptionFault">
6928         <wsdlsoap:fault name="DuplicateRoleExceptionFault" use="literal"/>
6929     </wsdl:fault>
6930     <wsdl:fault name="RoleValidationExceptionFault">
6931         <wsdlsoap:fault name="RoleValidationExceptionFault" use="literal"/>
6932     </wsdl:fault>
6933     <wsdl:fault name="UnsupportedOperationExceptionFault">
6934         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6935     </wsdl:fault>
6936     <wsdl:fault name="ImplementationExceptionFault">
6937         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6938     </wsdl:fault>
6939 </wsdl:operation>
6940
6941 <wsdl:operation name="updateRole">
6942     <wsdlsoap:operation soapAction=""/>
6943     <wsdl:input name="updateRoleRequest">
6944         <wsdlsoap:body use="literal"/>
6945     </wsdl:input>
6946     <wsdl:output name="updateRoleResponse">
6947         <wsdlsoap:body use="literal"/>
6948     </wsdl:output>
6949     <wsdl:fault name="NoSuchRoleExceptionFault">
6950         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
6951     </wsdl:fault>
6952     <wsdl:fault name="RoleValidationExceptionFault">
6953         <wsdlsoap:fault name="RoleValidationExceptionFault" use="literal"/>
6954     </wsdl:fault>
6955     <wsdl:fault name="UnsupportedOperationExceptionFault">
6956         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6957     </wsdl:fault>
6958     <wsdl:fault name="SecurityExceptionFault">
6959         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6960     </wsdl:fault>
6961     <wsdl:fault name="ImplementationExceptionFault">
6962         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6963     </wsdl:fault>
6964 </wsdl:operation>
6965
6966 <wsdl:operation name="getRole">
6967     <wsdlsoap:operation soapAction=""/>
6968     <wsdl:input name="getRoleRequest">
6969         <wsdlsoap:body use="literal"/>
6970     </wsdl:input>
6971     <wsdl:output name="getRoleResponse">
6972         <wsdlsoap:body use="literal"/>
6973     </wsdl:output>
6974     <wsdl:fault name="SecurityExceptionFault">
6975         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6976     </wsdl:fault>
6977     <wsdl:fault name="NoSuchRoleExceptionFault">
6978         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
6979     </wsdl:fault>
6980     <wsdl:fault name="UnsupportedOperationExceptionFault">
6981         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
6982     </wsdl:fault>
6983     <wsdl:fault name="ImplementationExceptionFault">
6984         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
6985     </wsdl:fault>
6986 </wsdl:operation>
6987
6988 <wsdl:operation name="undefineRole">
6989     <wsdlsoap:operation soapAction=""/>
6990     <wsdl:input name="undefineRoleRequest">
6991         <wsdlsoap:body use="literal"/>
6992     </wsdl:input>
6993     <wsdl:output name="undefineRoleResponse">
6994         <wsdlsoap:body use="literal"/>
6995     </wsdl:output>

```

```

6996     <wsdl:fault name="SecurityExceptionFault">
6997         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
6998     </wsdl:fault>
6999     <wsdl:fault name="NoSuchRoleExceptionFault">
7000         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7001     </wsdl:fault>
7002     <wsdl:fault name="UnsupportedOperationExceptionFault">
7003         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7004     </wsdl:fault>
7005     <wsdl:fault name="ImplementationExceptionFault">
7006         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7007     </wsdl:fault>
7008 </wsdl:operation>
7009
7010 <wsdl:operation name="addPermissions">
7011     <wsdlsoap:operation soapAction=""/>
7012     <wsdl:input name="addPermissionsRequest">
7013         <wsdlsoap:body use="literal"/>
7014     </wsdl:input>
7015     <wsdl:output name="addPermissionsResponse">
7016         <wsdlsoap:body use="literal"/>
7017     </wsdl:output>
7018     <wsdl:fault name="SecurityExceptionFault">
7019         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7020     </wsdl:fault>
7021     <wsdl:fault name="NoSuchRoleExceptionFault">
7022         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7023     </wsdl:fault>
7024     <wsdl:fault name="NoSuchPermissionExceptionFault">
7025         <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
7026     </wsdl:fault>
7027     <wsdl:fault name="UnsupportedOperationExceptionFault">
7028         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7029     </wsdl:fault>
7030     <wsdl:fault name="ImplementationExceptionFault">
7031         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7032     </wsdl:fault>
7033 </wsdl:operation>
7034
7035 <wsdl:operation name="setPermissions">
7036     <wsdlsoap:operation soapAction=""/>
7037     <wsdl:input name="setPermissionsRequest">
7038         <wsdlsoap:body use="literal"/>
7039     </wsdl:input>
7040     <wsdl:output name="setPermissionsResponse">
7041         <wsdlsoap:body use="literal"/>
7042     </wsdl:output>
7043     <wsdl:fault name="SecurityExceptionFault">
7044         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7045     </wsdl:fault>
7046     <wsdl:fault name="NoSuchRoleExceptionFault">
7047         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7048     </wsdl:fault>
7049     <wsdl:fault name="NoSuchPermissionExceptionFault">
7050         <wsdlsoap:fault name="NoSuchPermissionExceptionFault" use="literal"/>
7051     </wsdl:fault>
7052     <wsdl:fault name="UnsupportedOperationExceptionFault">
7053         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7054     </wsdl:fault>
7055     <wsdl:fault name="ImplementationExceptionFault">
7056         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7057     </wsdl:fault>
7058 </wsdl:operation>
7059
7060 <wsdl:operation name="removePermissions">
7061     <wsdlsoap:operation soapAction=""/>
7062     <wsdl:input name="removePermissionsRequest">
7063         <wsdlsoap:body use="literal"/>
7064     </wsdl:input>
7065     <wsdl:output name="removePermissionsResponse">

```

```

7066     <wsdlsoap:body use="literal"/>
7067 </wsdl:output>
7068 <wsdl:fault name="SecurityExceptionFault">
7069   <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7070 </wsdl:fault>
7071 <wsdl:fault name="NoSuchRoleExceptionFault">
7072   <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7073 </wsdl:fault>
7074 <wsdl:fault name="UnsupportedOperationExceptionFault">
7075   <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7076 </wsdl:fault>
7077 <wsdl:fault name="ImplementationExceptionFault">
7078   <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7079 </wsdl:fault>
7080 </wsdl:operation>
7081
7082 <wsdl:operation name="getClientIdentityNames">
7083   <wsdlsoap:operation soapAction=""/>
7084   <wsdl:input name="getClientIdentityNamesRequest">
7085     <wsdlsoap:body use="literal"/>
7086   </wsdl:input>
7087   <wsdl:output name="getClientIdentityNamesResponse">
7088     <wsdlsoap:body use="literal"/>
7089   </wsdl:output>
7090   <wsdl:fault name="UnsupportedOperationExceptionFault">
7091     <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7092   </wsdl:fault>
7093   <wsdl:fault name="SecurityExceptionFault">
7094     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7095   </wsdl:fault>
7096   <wsdl:fault name="ImplementationExceptionFault">
7097     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7098   </wsdl:fault>
7099 </wsdl:operation>
7100
7101 <wsdl:operation name="defineClientIdentity">
7102   <wsdlsoap:operation soapAction=""/>
7103   <wsdl:input name="defineClientIdentityRequest">
7104     <wsdlsoap:body use="literal"/>
7105   </wsdl:input>
7106   <wsdl:output name="defineClientIdentityResponse">
7107     <wsdlsoap:body use="literal"/>
7108   </wsdl:output>
7109   <wsdl:fault name="SecurityExceptionFault">
7110     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7111   </wsdl:fault>
7112   <wsdl:fault name="DuplicateClientIdentityExceptionFault">
7113     <wsdlsoap:fault name="DuplicateClientIdentityExceptionFault" use="literal"/>
7114   </wsdl:fault>
7115   <wsdl:fault name="ClientIdentityValidationExceptionFault">
7116     <wsdlsoap:fault name="ClientIdentityValidationExceptionFault" use="literal"/>
7117   </wsdl:fault>
7118   <wsdl:fault name="UnsupportedOperationExceptionFault">
7119     <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7120   </wsdl:fault>
7121   <wsdl:fault name="ImplementationExceptionFault">
7122     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7123   </wsdl:fault>
7124 </wsdl:operation>
7125
7126 <wsdl:operation name="updateClientIdentity">
7127   <wsdlsoap:operation soapAction=""/>
7128   <wsdl:input name="updateClientIdentityRequest">
7129     <wsdlsoap:body use="literal"/>
7130   </wsdl:input>
7131   <wsdl:output name="updateClientIdentityResponse">
7132     <wsdlsoap:body use="literal"/>
7133   </wsdl:output>
7134   <wsdl:fault name="SecurityExceptionFault">
7135     <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>

```



```

7136     </wsdl:fault>
7137     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7138         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7139     </wsdl:fault>
7140     <wsdl:fault name="ClientIdentityValidationExceptionFault">
7141         <wsdlsoap:fault name="ClientIdentityValidationExceptionFault" use="literal"/>
7142     </wsdl:fault>
7143     <wsdl:fault name="UnsupportedOperationExceptionFault">
7144         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7145     </wsdl:fault>
7146     <wsdl:fault name="ImplementationExceptionFault">
7147         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7148     </wsdl:fault>
7149 </wsdl:operation>
7150
7151 <wsdl:operation name="getClientIdentity">
7152     <wsdlsoap:operation soapAction="" />
7153     <wsdl:input name="getClientIdentityRequest">
7154         <wsdlsoap:body use="literal"/>
7155     </wsdl:input>
7156     <wsdl:output name="getClientIdentityResponse">
7157         <wsdlsoap:body use="literal"/>
7158     </wsdl:output>
7159     <wsdl:fault name="SecurityExceptionFault">
7160         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7161     </wsdl:fault>
7162     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7163         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7164     </wsdl:fault>
7165     <wsdl:fault name="UnsupportedOperationExceptionFault">
7166         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7167     </wsdl:fault>
7168     <wsdl:fault name="ImplementationExceptionFault">
7169         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7170     </wsdl:fault>
7171 </wsdl:operation>
7172
7173 <wsdl:operation name="getClientPermissionNames">
7174     <wsdlsoap:operation soapAction="" />
7175     <wsdl:input name="getClientPermissionNamesRequest">
7176         <wsdlsoap:body use="literal"/>
7177     </wsdl:input>
7178     <wsdl:output name="getClientPermissionNamesResponse">
7179         <wsdlsoap:body use="literal"/>
7180     </wsdl:output>
7181     <wsdl:fault name="SecurityExceptionFault">
7182         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7183     </wsdl:fault>
7184     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7185         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7186     </wsdl:fault>
7187     <wsdl:fault name="UnsupportedOperationExceptionFault">
7188         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7189     </wsdl:fault>
7190     <wsdl:fault name="ImplementationExceptionFault">
7191         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7192     </wsdl:fault>
7193 </wsdl:operation>
7194
7195 <wsdl:operation name="undefineClientIdentity">
7196     <wsdlsoap:operation soapAction="" />
7197     <wsdl:input name="undefineClientIdentityRequest">
7198         <wsdlsoap:body use="literal"/>
7199     </wsdl:input>
7200     <wsdl:output name="undefineClientIdentityResponse">
7201         <wsdlsoap:body use="literal"/>
7202     </wsdl:output>
7203     <wsdl:fault name="SecurityExceptionFault">
7204         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7205     </wsdl:fault>

```

```

7206     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7207         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7208     </wsdl:fault>
7209     <wsdl:fault name="UnsupportedOperationExceptionFault">
7210         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7211     </wsdl:fault>
7212     <wsdl:fault name="ImplementationExceptionFault">
7213         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7214     </wsdl:fault>
7215 </wsdl:operation>
7216
7217 <wsdl:operation name="addRoles">
7218     <wsdlsoap:operation soapAction=""/>
7219     <wsdl:input name="addRolesRequest">
7220         <wsdlsoap:body use="literal"/>
7221     </wsdl:input>
7222     <wsdl:output name="addRolesResponse">
7223         <wsdlsoap:body use="literal"/>
7224     </wsdl:output>
7225     <wsdl:fault name="SecurityExceptionFault">
7226         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7227     </wsdl:fault>
7228     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7229         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7230     </wsdl:fault>
7231     <wsdl:fault name="NoSuchRoleExceptionFault">
7232         <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7233     </wsdl:fault>
7234     <wsdl:fault name="UnsupportedOperationExceptionFault">
7235         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7236     </wsdl:fault>
7237     <wsdl:fault name="ImplementationExceptionFault">
7238         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7239     </wsdl:fault>
7240 </wsdl:operation>
7241
7242 <wsdl:operation name="removeRoles">
7243     <wsdlsoap:operation soapAction=""/>
7244     <wsdl:input name="removeRolesRequest">
7245         <wsdlsoap:body use="literal"/>
7246     </wsdl:input>
7247     <wsdl:output name="removeRolesResponse">
7248         <wsdlsoap:body use="literal"/>
7249     </wsdl:output>
7250     <wsdl:fault name="SecurityExceptionFault">
7251         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7252     </wsdl:fault>
7253     <wsdl:fault name="NoSuchClientIdentityExceptionFault">
7254         <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7255     </wsdl:fault>
7256     <wsdl:fault name="UnsupportedOperationExceptionFault">
7257         <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7258     </wsdl:fault>
7259     <wsdl:fault name="ImplementationExceptionFault">
7260         <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7261     </wsdl:fault>
7262 </wsdl:operation>
7263
7264 <wsdl:operation name="setRoles">
7265     <wsdlsoap:operation soapAction=""/>
7266     <wsdl:input name="setRolesRequest">
7267         <wsdlsoap:body use="literal"/>
7268     </wsdl:input>
7269     <wsdl:output name="setRolesResponse">
7270         <wsdlsoap:body use="literal"/>
7271     </wsdl:output>
7272     <wsdl:fault name="SecurityExceptionFault">
7273         <wsdlsoap:fault name="SecurityExceptionFault" use="literal"/>
7274     </wsdl:fault>
7275     <wsdl:fault name="NoSuchClientIdentityExceptionFault">

```

```

7276     <wsdlsoap:fault name="NoSuchClientIdentityExceptionFault" use="literal"/>
7277 </wsdl:fault>
7278 <wsdl:fault name="NoSuchRoleExceptionFault">
7279   <wsdlsoap:fault name="NoSuchRoleExceptionFault" use="literal"/>
7280 </wsdl:fault>
7281 <wsdl:fault name="UnsupportedOperationExceptionFault">
7282   <wsdlsoap:fault name="UnsupportedOperationExceptionFault" use="literal"/>
7283 </wsdl:fault>
7284 <wsdl:fault name="ImplementationExceptionFault">
7285   <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7286 </wsdl:fault>
7287 </wsdl:operation>
7288
7289 <wsdl:operation name="getSupportedOperations">
7290   <wsdlsoap:operation soapAction=""/>
7291   <wsdl:input name="getSupportedOperationsRequest">
7292     <wsdlsoap:body use="literal"/>
7293   </wsdl:input>
7294   <wsdl:output name="getSupportedOperationsResponse">
7295     <wsdlsoap:body use="literal"/>
7296   </wsdl:output>
7297   <wsdl:fault name="ImplementationExceptionFault">
7298     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7299   </wsdl:fault>
7300 </wsdl:operation>
7301
7302 <wsdl:operation name="getStandardVersion">
7303   <wsdlsoap:operation soapAction=""/>
7304   <wsdl:input name="getStandardVersionRequest">
7305     <wsdlsoap:body use="literal"/>
7306   </wsdl:input>
7307   <wsdl:output name="getStandardVersionResponse">
7308     <wsdlsoap:body use="literal"/>
7309   </wsdl:output>
7310   <wsdl:fault name="ImplementationExceptionFault">
7311     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7312   </wsdl:fault>
7313 </wsdl:operation>
7314
7315 <wsdl:operation name="getVendorVersion">
7316   <wsdlsoap:operation soapAction=""/>
7317   <wsdl:input name="getVendorVersionRequest">
7318     <wsdlsoap:body use="literal"/>
7319   </wsdl:input>
7320   <wsdl:output name="getVendorVersionResponse">
7321     <wsdlsoap:body use="literal"/>
7322   </wsdl:output>
7323   <wsdl:fault name="ImplementationExceptionFault">
7324     <wsdlsoap:fault name="ImplementationExceptionFault" use="literal"/>
7325   </wsdl:fault>
7326 </wsdl:operation>
7327 </wsdl:binding>
7328
7329 <!-- ALEACSERVICE -->
7330 <wsdl:service name="ALEACService">
7331   <wsdl:port binding="impl:ALEACServiceBinding" name="ALEACServicePort">
7332     <!-- The value of the location attribute below is an example only;
7333       Implementations are free to choose any appropriate URL. -->
7334     <wsdlsoap:address location="http://localhost:8080/services/ALEACService"/>
7335   </wsdl:port>
7336 </wsdl:service>
7337 </wsdl:definitions>

```

## 7338 5 Bindings for the Reading and Writing Callback APIs

7339 This section specifies XML-based bindings for the ALECallback and  
7340 ALECCallback interfaces, through which the ALE Reading API and the ALE

7341 Writing API, respectively, deliver asynchronous notifications to subscribers. Each  
7342 binding of these interfaces specifies a syntax for notification URIs. A notification URI is  
7343 supplied by an ALE client as a parameter of the `subscribe` and `unsubscribe`  
7344 methods of the ALE Reading or Writing API. The notification URI both selects a  
7345 binding of the callback interface to be used for that subscriber, and provides addressing  
7346 information in a manner specified by each binding below.

7347 Each subsection below specifies the conformance requirement (MAY, SHOULD,  
7348 SHALL) for each binding. Implementations MAY provide additional, vendor-specific  
7349 bindings of the callback interfaces. If an implementation provides an additional binding  
7350 of the callback interface, it SHALL use a URI scheme that does not conflict with any of  
7351 the standardized bindings.

7352 All notification URIs recognized by bindings as legal, whether the binding is  
7353 standardized as a part of this specification or not, SHALL conform to the general syntax  
7354 for URIs as defined in [RFC2396]. Each binding may impose additional constraints upon  
7355 syntax.

7356 A given mechanism for delivery of asynchronous results may be used in a binding of the  
7357 `ALECallback` (Reading API) interface and the `ALECCallback` (Writing API)  
7358 interface. This specification defines bindings of the `ALECallback` and  
7359 `ALECCallback` for each of four delivery mechanisms: HTTP, raw TCP, File, and  
7360 HTTP over TLS (HTTPS). Because the specifications of the `ALECallback` binding  
7361 and the `ALECCallback` binding for a given delivery mechanism are nearly identical,  
7362 they are pairwise combined in the specifications below.

## 7363 **5.1 HTTP Bindings**

7364 The HTTP bindings of the `ALECallback` and `ALECCallback` interfaces provide  
7365 for delivery of `ECReports` or `CCReports`, respectively, in XML via the HTTP  
7366 protocol using the POST operation. Implementations SHOULD provide support for these  
7367 bindings.

7368 The syntax for HTTP notification URIs as used by these bindings is defined in  
7369 [RFC2616], Section 3.2.2. Informally, an HTTP URI has one of the two following  
7370 forms:

7371 `http://host:port/remainder-of-URL`

7372 `http://host/remainder-of-URL`

7373 where

- 7374 • *host* is the DNS name or IP address of the host where the callback receiver is  
7375 listening for incoming HTTP connections.
- 7376 • *port* is the TCP port on which the callback receiver is listening for incoming HTTP  
7377 connections. The port and the preceding colon character may be omitted, in which  
7378 case the port defaults to 80.
- 7379 • *remainder-of-URL* is the URL to which an HTTP POST operation will be  
7380 directed.

7381 The ALE implementation delivers event cycle or command cycle reports by sending an  
7382 HTTP POST request to the callback receiver designated in the URI, where  
7383 *remainder-of-URL* is included in the HTTP *request-line* (as defined in  
7384 [RFC2616]), and where the payload is the *ECReports* instance or *CCReports*  
7385 instance encoded in XML according to the schema specified in Section 3.5 or  
7386 Section 3.6, respectively.

7387 The interpretation by the ALE implementation of the response code returned by the  
7388 callback receiver is outside the scope of this specification; however, all implementations  
7389 SHALL interpret a response code 2xx (that is, any response code between 200 and 299,  
7390 inclusive) as a normal response, not indicative of any error.

## 7391 5.2 TCP Bindings

7392 The TCP bindings of the *ALECallback* and *ALECCallback* interfaces provide for  
7393 delivery of *ECReports* or *CCReports*, respectively, in XML via a raw TCP  
7394 connection. Implementations SHOULD provide support for these bindings.

7395 The syntax for TCP notification URIs as used by these bindings is as follows:

7396 `tcp_URL = "tcp:" "//" host ":" port`

7397 where the syntax definition for *host* and *port* is specified in [RFC2396].

7398 Informally, a TCP URI has the following form:

7399 `tcp://host:port`

7400 The ALE implementation delivers an event cycle or command cycle report by opening a  
7401 new TCP connection to the specified host and port, writing to the connection the  
7402 *ECReports* instance or *CCReports* instance encoded in XML according to the  
7403 schema specified in Section 3.5 or Section 3.6, respectively, and then closing the  
7404 connection. The ALE implementation SHALL NOT require a reply or  
7405 acknowledgement.

## 7406 5.3 FILE Bindings

7407 The FILE bindings of the *ALECallback* and *ALECCallback* interfaces provide for  
7408 writing of *ECReports* or *CCReports*, respectively, in XML to a file.  
7409 Implementations MAY provide support for these bindings.

7410 The syntax for FILE notification URIs as used by these bindings is defined in  
7411 [RFC1738], Section 3.10. Informally, a FILE URI has one of the two following forms:

7412 `file://host/path`

7413 `file:///path`

7414 where

- 7415 • *host* is the DNS name or IP address of a remote host whose filesystem is accessible  
7416 to the ALE implementation.

7417 • *path* is the pathname of a file within the remote filesystem, or the local filesystem if  
7418 *host* is omitted.

7419 The ALE implementation delivers an event cycle or command cycle report by appending  
7420 to the specified file the `ECReports` or `CCReports` instance encoded in XML  
7421 according to the schema specified in Section 3.5 or Section 3.6, respectively. Note that if  
7422 more than one event cycle completes, the file will contain a concatenation of XML  
7423 documents, rather than a single XML document.

7424 Implementations of this binding may impose additional constraints on the use of the FILE  
7425 URI. For example, some implementations of this binding may support only a local  
7426 filesystem while others may support only a remote filesystem, some implementations of  
7427 this binding may impose further restrictions on the syntax of the *path* component, and  
7428 so forth. This specification also does not define the behavior when *path* names a  
7429 directory; the behavior in that case is implementation dependent.

7430 *Rationale (non-normative): The intended use for the FILE bindings is for debugging,*  
7431 *and hence the specification is intentionally lax in order to give freedom to*  
7432 *implementations to provide the most appropriate and useful facility given the unique*  
7433 *circumstances of that implementation.*

## 7434 5.4 HTTPS Bindings

7435 The HTTPS bindings of the `ALECallback` and `ALECCallback` interfaces provide  
7436 for delivery of `ECReports` or `CCReports`, respectively, in XML via the HTTP  
7437 protocol using the POST operation, secured via TLS. The HTTPS protocol provides  
7438 link-level security, and optionally mutual authentication between an ALE implementation  
7439 and its callback receivers. Implementations MAY provide support for these bindings.

7440 The syntax for HTTPS notification URIs as used by these bindings is defined in  
7441 [RFC2818], Section 2.4, which in turn is identical to the syntax defined in [RFC2616],  
7442 Section 3.2.2, with the substitution of `https` for `http`. Informally, an HTTPS URI has  
7443 one of the two following forms:

7444 `https://host:port/remainder-of-URL`

7445 `https://host/remainder-of-URL`

7446 where

- 7447 • *host* is the DNS name or IP address of the host where the callback receiver is  
7448 listening for incoming HTTPS connections.
- 7449 • *port* is the TCP port on which the receiver is listening for incoming HTTPS  
7450 connections. The port and the preceding colon character may be omitted, in which  
7451 case the port defaults to 443.
- 7452 • *remainder-of-URL* is the URL to which an HTTP POST operation will be  
7453 directed.

7454 The ALE implementation SHALL deliver event cycle or command cycle reports by  
7455 sending an HTTP POST request to the callback receiver designated in the URI, where



7456 *remainder-of-URL* is included in the HTTP *request-line* (as defined in  
7457 [RFC2616]), and where the payload is the *ECReports* or *CCReports* instance  
7458 encoded in XML according to the schema specified in Section 0 or Section 3.6,  
7459 respectively.

7460 For these bindings, HTTP SHALL be used over TLS as defined in [RFC2818]. TLS for  
7461 this purpose SHALL be implemented as defined in [RFC2246] except that the mandatory  
7462 cipher suite is *TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA*, as defined in [RFC3268] with  
7463 *CompressionMethod.null*. Implementations MAY support additional cipher suites and  
7464 compression algorithms as desired.

7465 The interpretation by the ALE implementation of the response code returned by the  
7466 callback receiver is outside the scope of this specification; however, all implementations  
7467 SHALL interpret a response code 2xx (that is, any response code between 200 and 299,  
7468 inclusive) as a normal response, not indicative of any error.

7469 *Warning (non-normative): The HTTPS report delivery mechanism is concerned only*  
7470 *with the delivery of ECReports and CCReports from the ALE implementation to*  
7471 *subscribed receivers. Using this mechanism is more secure than using cleartext*  
7472 *protocols such as HTTP or “raw” TCP but it does not ensure that the system as a whole*  
7473 *is secure. In a typical RFID system there will be other communications paths (for*  
7474 *example reader-to-ALE, and vendor-specific ALE administrative interfaces) that would*  
7475 *also need to be secure in order to claim that the entire system was secure.*

## 7476 **6 Appendix: Schema and WSDL Differences from ALE** 7477 **1.0 (non-normative)**

7478 This section enumerates differences between the XSD/WSDL for the ALE 1.1 Reading  
7479 API and the XSD/WSDL for ALE 1.0.

### 7480 **6.1 Fully Compatible Changes**

7481 The changes below are fully compatible, in that a document conforming to the ALE 1.1  
7482 schema is valid according to the ALE 1.0 schema, and vice versa.

- 7483 • In the definition of *ECReportGroupList*, an *anyAttribute* declaration is  
7484 added. This is part of the extensibility mechanism defined in Section 3.2, but was  
7485 inadvertently omitted from the ALE 1.0 schema.
- 7486 • In the definition of *ECReportOutputSpec*, an *anyAttribute* declaration is  
7487 added. This is part of the extensibility mechanism defined in Section 3.2, but was  
7488 inadvertently omitted from the ALE 1.0 schema.
- 7489 • Several types have been extended by adding new subelements to the *<extension>*  
7490 element, following the extensibility mechanism defined in Section 3.2. A nested  
7491 *<extension>* element is also added, in order to provide for further extensions in  
7492 later versions of the specification. The types that have been extended in this way  
7493 include: *ECBoundarySpec*, *ECFilterSpec*, *ECGroupSpec*,

- 7494 ECReportGroupListMember, ECReportOutputSpec, ECReportSpec,  
7495 and ECSpec.
- 7496 • The ECReport type has been extended by the addition of a new, optional attribute,  
7497 following the extensibility mechanism defined in Section 3.2.
- 7498 • The following new types have been added to the schema:  
7499 ECInitiationCondition, ECReportMemberField,  
7500 ECReportOutputFieldSpec, ECStatProfileName, ECTagStat,  
7501 ECTagTimestampStat, ECFieldSpec, ECFilterListMember,  
7502 ECIncludeExclude, ECReaderStat, and ECSightingStat.

## 7503 6.2 Non-Forward Compatible Changes

7504 The changes below are backward compatible, meaning that a document conforming to  
7505 the ALE 1.0 schema is valid according to the ALE 1.1 schema, but not forward  
7506 compatible, meaning that a document conforming to the ALE 1.1 schema may not be  
7507 valid according to the ALE 1.0 schema in some cases. The lack of forward compatibility  
7508 stems from areas of non-extensibility in the ALE 1.0 schema.

- 7509 • The ECGroupSpec type has been changed to include the extensibility mechanism,  
7510 and the extensibility mechanism is used to add a new, optional subelement. An  
7511 ECGroupSpec that includes the new, optional fieldspec subelement will not be  
7512 valid according to the ALE 1.0 schema.
- 7513 • All enumeration types have been made extensible in ALE 1.1, by changing the  
7514 schema to accept any string. Consequently, a document that contains an enumeration  
7515 value not specified in ALE 1.0 will not be valid according to the ALE 1.0 schema.  
7516 The enumeration types are: ECReportSetEnum, ECTerminationCondition,  
7517 and ECTimeUnit, along with ImplementationExceptionSeverity  
7518 defined in the WSDL.

## 7519 6.3 Corrections

7520 Two errors in the ALE 1.0 schema have been corrected in the ALE 1.1 schema.

- 7521 • In the ECReportSetSpec type, the set attribute is now declared to be required.  
7522 In the ALE 1.0 schema, the use declaration was omitted which implied that the  
7523 attribute was optional.
- 7524 • In the ECTime type, the unit attribute is now declared to be required. In the ALE  
7525 1.0 schema, the use declaration was omitted which implied that the attribute was  
7526 optional.

7527 Technically, these changes are not backward compatible, in that a document that  
7528 conforms to the ALE 1.0 schema may not be valid according to the ALE 1.1 schema.  
7529 However, this only occurs if the document omitted one of the two attributes listed above.  
7530 The main body of the ALE 1.0 specification did not specify that these attributes are  
7531 optional, and no behavior was defined for the case where they are omitted. Therefore, the

7532 lack of back-compatibility only occurs for a document whose meaning is not defined  
7533 according to the ALE 1.0 specification, and so the lack of compatibility is unlikely to  
7534 arise in practice.

## 7535 **6.4 Changes in Idiom**

7536 The following changes to the schema do not affect what documents are valid or invalid  
7537 according to the schema, but simply changes the XSD description to an alternative,  
7538 equivalent form.

- 7539 • The `ECTrigger` type, which is equivalent to `xsd:string`, is now defined as a  
7540 (trivial) restriction of `xsd:string` rather than as a (trivial) extension of  
7541 `xsd:string`. This allows `ECTrigger` to be used as the type of an attribute as  
7542 well as the type of an element.
- 7543 • Several type names were defined in the ALE 1.0 schema in places where the UML  
7544 specified a field of type `List<...>`. These type names appeared only in the schema,  
7545 and not in the UML. To avoid confusion, in the ALE 1.1 schema these type names  
7546 are eliminated by embedding the type definition in the context where the reference to  
7547 the list type occurs. The type names that have been eliminated in this way are:  
7548 `ECExcludePatterns`, `ECIncludePatterns`, `ECLogicalReaders`,  
7549 `ECReportList`, and `ECReportSpecs`.

## 7550 **7 References**

- 7551 [ALE1.1Part1] EPCglobal, "The Application Level Events (ALE) Specification,  
7552 Version 1.1.1 Part I: Core Specification," EPCglobal Ratified Standard, March 2009,  
7553 [http://www.epcglobalinc.org/standards/ale/ale\\_1\\_1\\_1-standard-core-20090313.pdf](http://www.epcglobalinc.org/standards/ale/ale_1_1_1-standard-core-20090313.pdf).
- 7554 [ISODir2] ISO, "Rules for the structure and drafting of International Standards  
7555 (ISO/IEC Directives, Part 2, 2001, 4th edition)," July 2002.
- 7556 [RFC1738] T. Berners-Lee, L. Masinter, M. McCahill, "Uniform Resource Locators  
7557 (URL)," RFC 1738, December 1994, <http://www.ietf.org/rfc/rfc1738>.
- 7558 [RFC2246] T. Dierks, C. Allen, "The TLS Protocol, Version 1.0," RFC2246, January  
7559 1999, <http://www.ietf.org/rfc/rfc2246>.
- 7560 [RFC2396] T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers  
7561 (URI): Generic Syntax," RFC2396, August 1998, <http://www.ietf.org/rfc/rfc2396>.
- 7562 [RFC2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T.  
7563 Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1," RFC2616, June 1999,  
7564 <http://www.ietf.org/rfc/rfc2616>.
- 7565 [RFC2818] E. Escorla, "HTTP Over TLS," RFC2818, May 2000,  
7566 <http://www.ietf.org/rfc/rfc2818>.
- 7567 [RFC3268] P. Chown, "Advanced Encryption Standard (AES) Ciphersuites for  
7568 Transport Layer Security (TLS)," RFC3268, June 2002, <http://www.ietf.org/rfc/rfc3268>.

- 7569 [SOAP1.1] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F.  
7570 Nielsen, S. Thatte, and D. Winer, “Simple Object Access Protocol (SOAP) 1.1,” W3C  
7571 Note, May 2000, <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>.
- 7572 [WSDL1.1] E. Christensen, F. Curbera, G. Meredith, S. Weerawarana, “Web Services  
7573 Description Language (WSDL) 1.1,” W3C Note, March 2001,  
7574 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>.
- 7575 [WSI] K. Ballinger, D. Ehnebuske, M. Gudgin, M. Nottingham, P. Yendluri, “Basic  
7576 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)  
7577 [i.org/Profiles/BasicProfile-1.0-2004-04-16.html](http://www.ws-i.org/Profiles/BasicProfile-1.0-2004-04-16.html).
- 7578 [XML1.0] T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau,  
7579 “Extensible Markup Language (XML) 1.0 (Third Edition),” W3C Recommendation,  
7580 February 2004, <http://www.w3.org/TR/2004/REC-xml-20040204/>.
- 7581 [XSD1] H. Thompson, D. Beech, M. Maloney, N. Mendelsohn, “XML Schema Part 1:  
7582 Structures,” W3C Recommendation, May 2001, <http://www.w3.org/TR/xmlschema-1/>.
- 7583 [XSD2] P. Biron, A. Malhotra, “XML Schema Part 2: Datatypes,” W3C  
7584 Recommendation, May 2001, <http://www.w3.org/TR/xmlschema-2/>.
- 7585 [XMLVersioning] D. Orchard, “Versioning XML Vocabularies,” December 2003,  
7586 <http://www.xml.com/pub/a/2003/12/03/versioning.html>.

## 7587 **8 Credits**

7588 See [ALE1.1Part1], Section 18.

7589