



GS1 Object Name Service (ONS) Version 2.0.1

Ratified Standard
Issue 2 January 31, 2013



Document Summary

Document Item	Current Value
Document Title	Draft Standard Specification GS1 Object Name Service (ONS)
Date Last Modified	January 31, 2013
Document Issue	Issue 2 of Ratified Standard-updated with fixed errata
Document Status	Ratified Standard
Document Description	

Contributors

Name (& Notable Role)	Organization
Sandoche Balakrichenan-- Conformance Requirements Editor and Prototype Test Participant	AFNIC
Mark Harrison	Auto-ID Labs
Philippe Gautier	Business2Any
Marc-Antoine Mouilleron	France Telecom Orange
Philippe Rodier	France Telecom Orange
Jacques Madelaine	GREYC
Adrien Laurence	GREYC
Jerome Le Moulec	GREYC
Steven Pereira	GS1 Australia
Manfred Piller	GS1 Austria
Kevin Dean Co—Chair, Editor of ONS Solution; GS1 AG Liaison	GS1 Canada
Han Du	GS1 China
Zhang Xu	GS1 China
Giovanni Biffi	GS1 Colombia
Juan Ochoa	GS1 Colombia
Felipe Serrano	GS1 Columbia
Douglas Hill	GS1 Denmark
Per Kiilsholm	GS1 Denmark
Pertti Hakala	GS1 Finland
Nicolas Pauvre co-Chair, Prototype Test Participant	GS1 France
Andreas Fuessler	GS1 Germany
Ralph Troeger	GS1 Germany
Dipan Anarkat	GS1 Global Office

Name (& Notable Role)	Organization
Henri Barthel	GS1 Global Office
Mark Frey-- Facilitator & Process Manager	GS1 Global Office
Janice Kite	GS1 Global Office
Sean Lockhead	GS1 Global Office
Craig Alan Repec	GS1 Global Office
KK Suen	GS1 HK
Tany Hui --Prototype Test Participant	GS1 Hong Kong
Albert Tsang	GS1 Hong Kong
Noriyuki Mama	GS1 Japan
Reiko Moritani	GS1 Japan
Martin Beno	GS1 Slovakia
Miroslava Staffenova	GS1 Slovakia
Jeremy Morton--Prototype Test Participant	GS1 Sweden
Steffen Olsson	GS1 Sweden
Heinz Graf	GS1 Switzerland
Francis Kienlen-- Prototype Test Participant	GS1 Switzerland
Daniel Mueller	GS1 Switzerland
Christian Schneider	GS1 Switzerland
Andrew Osborne	GS1 UK
Roberto Quilez	INRIA
Ken Traub	Ken Traub Consulting LLC

Log of Changes in Ratified Standard

Issue No.	Date of Change	Changed By	Summary of Change

Disclaimer

WHILST EVERY EFFORT HAS BEEN MADE TO ENSURE THAT THE GUIDELINES TO USE THE GS1 STANDARDS CONTAINED IN THE DOCUMENT ARE CORRECT, GS1 AND ANY OTHER PARTY INVOLVED IN THE CREATION OF THE DOCUMENT HEREBY STATE THAT THE DOCUMENT IS PROVIDED WITHOUT WARRANTY, EITHER EXPRESSED OR IMPLIED, REGARDING ANY MATTER, INCLUDING BUT NOT LIMITED TO THE OF ACCURACY, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE, AND HEREBY DISCLAIM ANY AND ALL LIABILITY, DIRECT OR INDIRECT, FOR ANY DAMAGES OR LOSS RELATING TO OR RESULTING FROM THE USE OF THE DOCUMENT. THE DOCUMENT MAY BE MODIFIED, SUBJECT TO DEVELOPMENTS IN TECHNOLOGY, CHANGES TO THE STANDARDS, OR NEW LEGAL REQUIREMENTS. SEVERAL PRODUCTS AND COMPANY NAMES MENTIONED HEREIN MAY BE TRADEMARKS AND/OR REGISTERED TRADEMARKS OF THEIR RESPECTIVE COMPANIES. GS1 IS A REGISTERED TRADEMARK OF GS1 AISBL.

Table of Contents

1. Introduction	7
1.1. Status of this document	7
2. Terminology and Typographical Conventions	7
3. ONS Introduction (non-normative)	7
3.1. Background Information	8
3.2. The Domain Name System (DNS)	8
3.2.1. Client's View	8
3.2.2. Publisher's View	9
3.3. ONS's Usage of DNS	10
3.3.1. Serial Number Level Queries to the ONS	12
3.4. Why the Federated Model?	12
3.5. Actors	13
4. ONS Delegation Architecture Organization	13
4.1. Delegation	14
4.1.1. A Note about Alliance Numbers	15
4.1.2. Modification of the zone in the ONS Peer Root	15
5. ONS Formal Specification	17
6. ONS DDDS Application Specification	17
6.1. Application Unique String (AUS)	17
6.1.1. Converting an EPC to an AUS	18
6.1.2. Examples (non-normative)	18
6.2. First Well-Known Rule	19
6.2.1. Examples (non-normative)	20
6.2.2. A note about non-numeric characters	21
7. ONS DDDS database	21
7.1. DNS query format	21
7.2. NAPTR RR	22
7.2.1. Flags	23
7.2.2. Services	23
7.2.3. Regular expressions (non-normative)	25
7.2.4. Service type hierarchy (non-normative)	25
8. Processing ONS Query Responses	26
9. Examples (non-normative)	27
9.1. Finding the length of the GS1 Company Prefix (GCP)	29

9.2.	Finding an authoritative EPCIS server for a product	30
9.3.	Finding a Mobile Commerce service for product.....	30
9.4.	Dynamic interaction with extended services	30
9.5.	Sample ServiceType XML	31
10.	References	32
11.	Appendix A – Glossary (non-normative).....	32

1. Introduction

This document gives an overview of Object Name Service (ONS) and specifies how ONS uses the Domain Name System (DNS) infrastructure and implementation. This document is a formal specification of an implementation of the Dynamic Delegation Discovery System (DDDS) algorithm to locate authoritative metadata and services associated with a given GS1 Identification Key. Its target audience is all interested parties who seek to have a technical overview of ONS and particularly to developers that will be implementing ONS resolution systems for applications.

1.1. Status of this document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest version of this document series is maintained by GS1. This document is the ratified version of the GS1 Object Name Service (ONS) and has been updated to fix some errata

Comments on this document should be sent to GSMP@gs1.org.

2. Terminology and Typographical Conventions

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

All sections of this document are normative, except where explicitly noted as non-normative.

The following typographical conventions are used throughout the document:

- ALL CAPS type is used for the special terms from [ISODir2] enumerated above.
- `Monospace` type is used to denote programming language, UML, and XML identifiers, as well as for the text of XML documents.

3. ONS Introduction (non-normative)

In keeping with the assumption that GS1 standards should leverage existing standards and infrastructure where appropriate, ONS uses the Internet's existing DNS for looking up (resolving) information about a GS1 Identification Key. This means that the query and response formats must adhere to the DNS standards, meaning that the GS1 Identification Key will be converted to a domain name and the results must be a valid DNS Resource Record (RR).

Important terminology note: the usage of the terms "ONS" and "DNS" may seem arbitrary but they are not. The term DNS is used when the discussion is generally applicable to the DNS system. ONS is used when the discussion is specifically about querying the DNS for a GS1 Identification Key. For example, a query for a mail exchange (MX) resource record (RR) is a DNS query. A query to locate an EPCIS server through a GS1 Identification Key would be called an ONS query, even though the query is carried out using DNS.

Additionally, it is important to outline the difference between a "service" and a "server" in the context of this document. A service is a set of functions that accomplish some task. That set of functions may actually be implemented using one or more networked computer systems (which we refer to as servers) acting in concert. A good example is the concept of a "Local ONS", or what is commonly

41 referred to as a local name service, implementing ONS functionality. In most situations this service is
42 provided by two physically separate servers that act as backups for one another.

43 3.1. Background Information

44 This document draws from the initial work at the Auto-ID Center, and we recognize the contribution of
45 the following individuals: Joe Foley (MIT), Erik Nygren (MIT), Sanjay Sarma (MIT), David Brock (MIT),
46 Sunny Siu (MIT), Laxmiprasad Putta (OATSystems), Sridhar Ramachandran (OATSystems). The
47 following papers capture the contributions of these individuals:

- 48 ■ Engels, D., Foley, J., Waldrop, J., Sarma, S. and Brock, D., "The Networked Physical World:
49 An Automated Identification Architecture" Proceedings of the 2nd IEEE Workshop on Internet
50 Applications (WIAPP '01), 76-77, 2001.
- 51 ■ The Object Name Service Technical Manual, Version 0.5 (Beta)
52 <http://www.autoidlabs.org/whitepapers/MIT-AUTOID-TM-004.pdf>

53 3.2. The Domain Name System (DNS)

54 This section is for readers who may not be familiar with DNS. Explaining the DNS infrastructure and
55 implementation is not within the scope of this document. Hence, this is a basic introduction only and
56 for those interested, references to appropriate technical documents on DNS will be provided.

57 In order to have a basic understanding of DNS it is necessary to understand the system from two
58 major viewpoints. The first is from the viewpoint of the client that is querying DNS. The second is from
59 the viewpoint of an entity publishing data into DNS to be queried by a client.

60 3.2.1. Client's View

61 A classical DNS usage pattern is to resolve the IP address of a Fully Qualified Domain name (FQDN)
62 such as "www.gs1.org". A user who wishes to view the web site "www.gs1.org" uses an application
63 such as a web browser. The application sends a request to a stub resolver in the local operating
64 system. The stub resolver will invariably have a cache containing the results of recent DNS requests. If
65 the cache can provide the answer to the request, the resolver will return the value in the cache to the
66 application that made the request. If the cache does not contain the answer, the resolver will send the
67 request to one or more designated DNS servers until it obtains the response for the request. Each of
68 these DNS servers may in turn delegate to other DNS servers further up the chain.

69 From the application (e.g. web browser) standpoint, the DNS is a black box. Questions go in and
70 answers come out. Where the answer comes from, how long it can be cached, and what sequence of
71 delegations it took to find the answer are all hidden from the application that issues the query.

72 A DNS resolver used to perform a query can resolve the query with a minimum of two pieces
73 information: the domain name in question and the type of resource record (RR) that is being
74 requested.

75 For example, on a computer that has a properly configured network, the following snippet of Java code
76 is one of the many possible ways to issue a DNS query and retrieve the results:

```
import javax.naming.directory.*;  
DirContext ictx = new InitialDirContext();  
Attributes attrs = ictx.getAttributes("dns://corp.example.com", "MX");
```

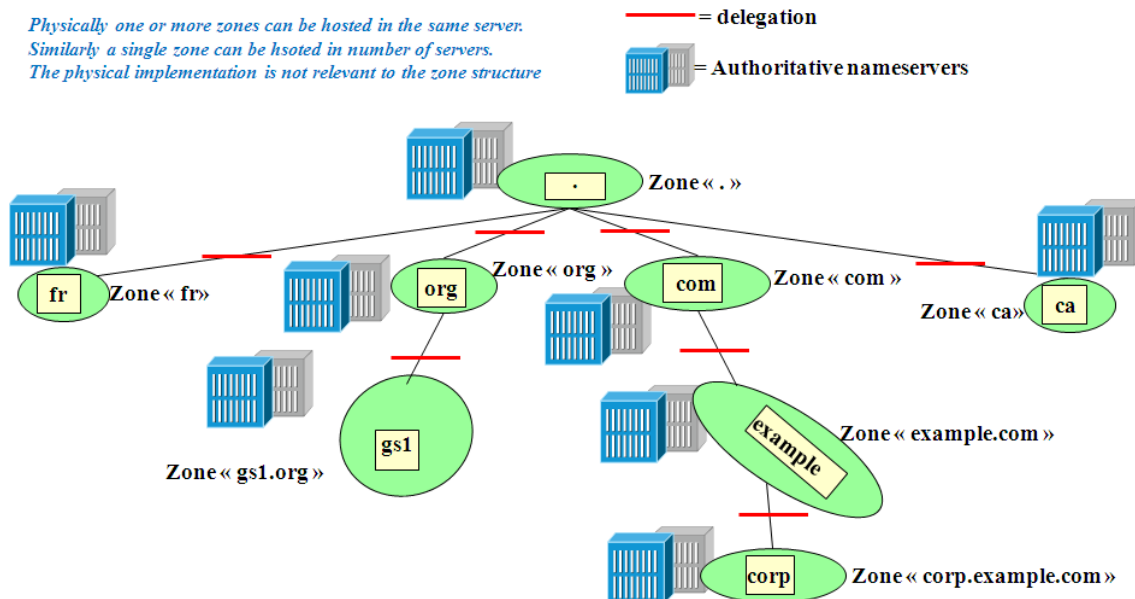
77
78 The "corp.example.com" is the domain name and the "MX" is the record type that is being requested.
79 "MX" is the "mail exchanger" that specifies a server responsible for accepting email for a domain.
80 When the DNS resolution process is successful, the "attrs" variable will contain the hostname for the
81 MX server for the "corp.example.com" domain. All other DNS issues such as finding the root servers

82 and the authoritative name servers, how long the DNS record is stored in the cache etc., are hidden
 83 from the application.
 84 For a more detailed explanation of how the DNS protocol works please refer to [DNS].

85 3.2.2. Publisher's View

86 While applications are happily able to ignore how the DNS infrastructure actually works, the data the
 87 client is consuming must be provisioned in a way that is understandable by the DNS protocol. A basic
 88 introduction to the key concepts of DNS infrastructure is provided here.

89 DNS is a distributed database that is indexed by domain names. Each domain name is essentially just
 90 a path in a large inverted tree called the domain namespace. The tree has a single root at the top. It is
 91 referred simply as the "root" and denoted by a "." (dot). Entries in the root "zone" are called Top Level
 92 Domains (TLDs) such as ".com", ".net", ".org", ".fr", ".us", etc. The process of separating a descendant
 93 of a zone into a separate zone is called "delegation". The delegation is accomplished with Nameserver
 94 (NS) records. The delegation system enables hosts to control given chunks of the database and the
 95 whole database is reachable by traversing the path from the root to each lower level; each point at
 96 which there is a "." in a domain name, there is a delegation to domains lower in the hierarchy.
 97 Generally speaking, for each delegation there is a corresponding network server that contains data for
 98 that subsection of the hierarchy. In the previous example of "corp.example.com" the delegation of
 99 servers would look as in Figure 1:



100
 101 **Figure 1 – Server delegation**

102 The server that stores information about a domain namespace is called a nameserver. For example
 103 the root domain has thirteen nameservers, each of which contains the same data. The data that a
 104 nameserver publishes is called a "zone". In many situations a single nameserver can contain multiple
 105 zones.

106 Nameservers generally have complete information about some part of the domain namespace. There
 107 are two types of nameservers, Authoritative and Caching.

108 The Authoritative nameserver contains an entire copy of the zone that is derived from the local
 109 configuration data, possibly with the help of another authoritative nameserver for the zone. A server
 110 can be authoritative for one zone but not authoritative for another. The Master (primary) server is an
 111 Authoritative nameserver that gets its zone data from the local configuration, not from an outside

112 source. This is where all changes to a zone's contents are made. The DNS protocol provides an
113 automatic mechanism for propagating the contents of a zone to slave (secondary) servers, which will
114 be queried in case of the failure of primary servers.

115 A caching nameserver is usually a local nameserver. It initially looks up a name within the current
116 zone and ends up asking one of the zone's nameservers for the answer and keeps the result of all the
117 name resolutions in its cache for a Time To Live (TTL). The TTL on a resource record is the length of
118 the time for which any nameserver can cache that record.

119 The data associated with domain names are contained in Resource Records (RRs). These define
120 some attribute for a domain name such as its IP address, a string of text, or a mail route. A
121 nameserver RR declares that a given zone is served by a given nameserver. Every nameserver
122 record is either a delegation record or an authority record. If the name of the nameserver record is the
123 name of the zone it appears in, it is an authority record. If the name of the nameserver record is that of
124 a descendant zone, then it is a delegation record.

125 For further details please refer to [DNS].

126 3.3. ONS's Usage of DNS

127 In order to use DNS to find information about an item, the item's GS1 Identification Key must be
128 converted into a format that DNS can understand, which is the typical, "dot" delimited, left-to-right form
129 of all domain names.

130 As the purpose of ONS is to discover data and services associated with a GS1 Identification Key and
131 multiple sets of data and services may exist for that key, the appropriate DNS record type is the
132 Naming Authority PoinTeR (NAPTR) [RFC 3403]. This record type contains several fields for denoting
133 the protocol, services, and features that a given service endpoint exposes. It also allows the service
134 endpoint to be expressed as a URI, thus allowing complex services to be encoded in a standard way.

135 Figure 2 describes a typical ONS query from start to finish from the viewpoint of an application. In this
136 example, the starting point is a bar code or RFID tag. However, the source of the GS1 Identification
137 Key is not restricted to data carriers; it could be part of a transaction document (e.g. a purchase order),
138 an event record, a master data record, or any other source.

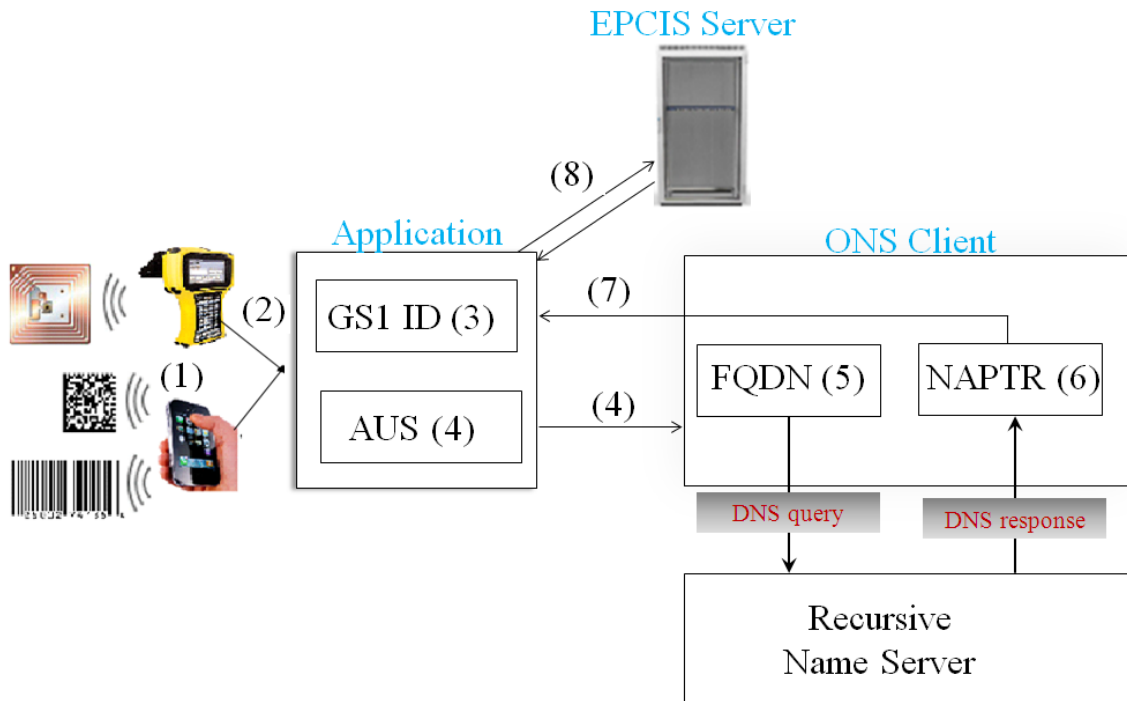


Figure 2 – A typical ONS query

1. A sequence of data denoting a GS1 Identification Key and optional supplemental data (e.g. ship-to location, batch/lot number, etc.) is read from a bar code or an RFID tag using an appropriate reader. When presented to the application layer, this data will be in text form.
2. The reader sends that sequence of data to an ONS application.
3. The ONS application extracts the desired GS1 Identification Key type and the GS1 Identification Key itself from the sequence of data. It should be noted that there is no requirement that the desired GS1 Identification Key be the primary GS1 Identification Key in the data stream. For example, while a shipping container will carry a Serial Shipping Container Code (SSCC) as a primary identifier, the application may be interested in discovering services associated with the Global Trade Item Number (GTIN) in the container. The GTIN is indicated using AI 02 in a supplementary bar code accompanying the SSCC. For example, if the data extracted from the bar codes, converted to a GS1 Element String, is as follows:

```
(00)306141417782246356(02)50614141322607(37)20
```

then the contained GTIN is 50614141322607.

4. The ONS application presents the GS1 Identification Key type, the GS1 Identification Key itself, the client language code (optional), and the client country code (optional) to the ONS client. For example, assuming an application, configured to operate in English in Canada:

```
en|ca|gtin|50614141322607
```

5. The ONS client converts the GS1 Identification Key type and the GS1 Identification Key into an appropriate FQDN as per Section 6.2 and issues a DNS query for NAPTR records for that domain. Example:

```
5.0.6.2.2.3.1.4.1.4.1.6.0.gtin.gs1.id.onsepc.com
```

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

- 163
164
165
6. The DNS infrastructure returns a series of answers that contain service types and associated data (likely Uniform Resource Locators (URLs) that point to one or more services such as EPCIS or Mobile Commerce). See Section 9 for examples.
- 166
167
7. The ONS client extracts the service type and the service data from the DNS NAPTR record, interprets it according to the rules in Section 8 and presents it back to the ONS application.
- 168
169
8. The application interprets the data appropriate to the service type. Example:
The application contacts the EPCIS server at the resolved URL.

170 3.3.1. Serial Number Level Queries to the ONS

171
172
173
174
175
176
177

It is important to note that this version of ONS does not support queries at the serial number or “instance” level. Instead, the lowest level of granularity supported is the “Object Class” level. Subsequent queries for information about a given serial number must be resolved by querying the application layer server designated by the ONS results. This is true for all GS1 Identification Keys. The ability to specify an ONS query at the serial number level as well as the architectural and economic impacts of that capability is an open issue that may be addressed in subsequent versions of this document. Its lack of mention here should not be construed as making that behaviour legal or illegal.

178 3.4. Why the Federated Model?

179
180

The ONS lookup function, being based upon DNS, is implemented by a distributed set of servers with a single (replicated) root, currently at onsepc.com.

181
182
183

The governance model and the architecture of the ONS have therefore raised concerns within some industrial and political communities. These concerns were notably expressed in the Internet of Things action plan for Europe¹.

184
185
186
187
188
189
190

The ONS design is consequently considered as a political issue that has to be negotiated on a governance perspective before it can be implemented. In particular, logistic applications that could imply sensitive product localization (e.g. drugs, weapons, nuclear waste...) or sensitive applications (product recall, anti-counterfeiting applications, food or drugs safety control...) are considered as critical applications as well as the architectures that support them. Many countries have indeed laws and regulations forbidding hosting of critical citizen and industry data outside their borders. Initial country routing (via onsepc.com) violates that requirement.

191
192

Concisely, in the global implementation of ONS, it has become thus evident that having a single resolution root for the entire system is not acceptable, due to the following concerns:

- 193
194
- Political concerns (risk of loss of sovereignty, control over information may be lost when subject to laws/control of other countries, etc.)

195
196
197

 - Capability concerns (risk of loss of domestic and strategic capability (risk of loss of infrastructure in case of breakdowns, risk of service level reduction compared with local suppliers, etc.)

198

 - Security concerns (risk of business intelligence gathering, etc.)

199
200
201

 - Commercial concerns (risk of high price and maintenance costs, risk of financial viability of the supplier, risk of contract lock-in, risks arising from repatriation and/or transfer to another supplier if required, etc.)

202

 - Innovation concern (no competitive challenge for technical innovation & marketing, etc.)

¹ http://ec.europa.eu/information_society/policy/rfid/documents/commiot2009.pdf

203 The answer to these issues is that ONS shall support a federated model in which multiple ONS Peer
 204 Roots may co-exist. Each ONS Peer Root shall be equivalent in functionality to every other; queries at
 205 any peer shall resolve to the same results regardless of the peer initially chosen. The ONS Peer Roots
 206 shall work collaboratively at the same hierarchical level to resolve client queries while respecting each
 207 peer's scope of authority.

208 This federated ONS model will be thus better adapted to local ownership and control requirements and
 209 will mitigate risk of single-country control.

210 From this point forward, the term "peer" shall be taken to refer to any one of the equivalent ONS Peer
 211 Root nodes within the federation.

212 3.5. Actors

213 A number of actors are referred to throughout this document. These may be legal entities, persons, or
 214 automated systems.

215 GS1 Global Office – This is the office located in Brussels, Belgium. It is not a traditional head office in
 216 that its work is driven by the country-level GS1 Member Organizations but it has a coordinating
 217 responsibility among all the GS1 country offices.

218 GS1 Member Organization – This is an office with responsibility for a country (e.g. GS1 Canada) or set
 219 of countries (e.g. GS1 UK).

220 GS1 Company Prefix Subscriber – This is an organization that has licensed a GS1 Company Prefix
 221 from a GS1 Member Organization. The traditional term "brand owner" is often used, but as a brand
 222 refers to a product and ONS supports all GS1 Identification Keys, the term GS1 Company Prefix
 223 Subscriber is preferred.

224 User – This is a human using an application to interact, directly or indirectly, with ONS.

225 Application – This is a traditional computer software application.

226 Client – This is a set of software libraries that may be part of an application or may be remotely
 227 accessible to an application. The application provides its parameters (the Application Unique String
 228 (AUS) and optionally the service(s) it is looking for) to the client. The client converts the AUS to a Fully
 229 Qualified Domain Name (FQDN) and is capable of performing a recursive search of the ONS
 230 infrastructure to locate Resource Records (RRs) required by the application.

231 ONS Peer Root – This is a logical server (optionally composed of multiple physical distributed
 232 servers), partitioned by the range of GS1 Company Prefixes that it supports. For queries within the
 233 range of GS1 Company Prefixes that it supports, it either answers them directly or delegates to
 234 another name server in the sub-domain; for queries outside the range of GS1 Company Prefixes it
 235 supports, it delegates to another ONS Peer Root.

236 4. ONS Delegation Architecture Organization

237 The previous section covered DNS and how ONS uses DNS from a client point of view. What it did not
 238 cover was the processes and procedures for making ONS data available in the proper form for the
 239 client. The main issue discussed in this section will be delegation for ONS at different levels.

240 Throughout this section, a few sample domains will be used. These are samples only and do not
 241 authoritatively identify the domains that may be chosen by the Member Organizations (MOs) that
 242 implement distributed ONS nodes. The domains are:

Domain	Usage
onsepc.com	ONS maintained by GS1 Global
ons.epcglobalcanada.org	Canadian ONS

Domain	Usage
onsepc.fr	French ONS
ons.epcglobal.cn	Chinese ONS

243 For purposes of example, the following GTIN and serial number in GS1 Element String form will be
 244 used:

245 (01)50614141322607(21)400

246 wherein “01” is the GTIN Application Identifier (AI), 50614141322607 represents the GTIN, “21” is
 247 the serial number AI, and 400 is the serial number.

248 Other GS1 identification keys have similar structure. Since the GTIN is the one most familiar to
 249 readers, it is used throughout.

250 Note again that the serial number in all implementations is actually irrelevant to ONS as ONS works
 251 only at the class level.

252 4.1. Delegation

253 An ONS Peer Root is one that may be used as a starting point for any ONS query. All ONS Peer
 254 Roots SHALL resolve to the same set of results.

255 At the ONS Peer Root level there are two types of delegations:

- 256 1. Nameserver (NS) and
- 257 2. Delegation name (DNAME).

258 NS delegation preserves the domain name and simply partitions a portion of the space to another
 259 nameserver. DNAME delegation, however, replaces one domain name with another and continues the
 260 query on the other domain name.

261 All GS1 Company Prefix Subscribers that wish to manage their own ONS entries under an ONS Peer
 262 Root namespace SHALL use NS delegation. An example of a NS delegation for a GS1 Company
 263 Prefix Subscriber at ONS Peer Root “onsepc.com” is:

264 1.4.1.4.1.6.0.gt.in.gs1.id.onsepc.com. IN NS ns1.corp.example.com.

265 1.4.1.4.1.6.0.gt.in.gs1.id.onsepc.com. IN NS ns2.corp.example.com.

266 Assuming that GS1 US uses “onsepc.com” as its root, a user under the GS1 US namespace, would
 267 query using the FQDN:

268 5.0.6.2.2.3.1.4.1.4.1.6.0.gt.in.gs1.id.onsepc.com

269 As onsepc.com is the authoritative domain for GS1 US, this query would be resolved locally (with the
 270 caveat that there might be further delegation to a local ONS run by the GS1 Company Prefix
 271 Subscriber) under the “onsepc.com” namespace.

272 Let's suppose a user having the Canadian ONS Peer Root as the starting point for resolution would
 273 query for the same product using the following FQDN:

274 5.0.6.2.2.3.1.4.1.4.1.6.0.gt.in.gs1.id.ons.epcglobalcanada.org

275 For the above query, the DNS resolver initially interrogates the Canadian ONS Peer Root. To redirect
 276 the query to “onsepc.com” (since the query is for a product under GS1 US namespace), the Canadian
 277 peer “epcglobalcanada.org” needs to know that the query is destined for the GS1 US ONS Peer Root.
 278 This is where DNAME delegation is required.

279 With the help of the GS1 prefix (last three digits “1.6.0” of the queried FQDN) “epcglobalcanada.org”
 280 can identify that the query is destined to the GS1 US ONS Peer Root.

281 To map a GS1 prefix to the corresponding ONS Peer Root and to append the appropriate peer
 282 domain name string to the query, there needs to be a mechanism within the existing DNS database.

283 The redirection SHALL be done based on the GS1 prefix using DNAME [RFC 2672] as follows:

```
284 1.6.0.gtin.gs1.id.ons.epcglobalcanada.org.          IN          DNAME
285 1.6.0.gtin.gs1.id.onsepc.com.
```

286 The facility with DNAME is that there is no necessity for each ONS Peer Root to have the knowledge
 287 about the zone information of its peers. The only necessity is that there should be DNAME redirection
 288 RRs for all GS1 prefixes for all MOs that participate in the ONS network. Each peer will have the
 289 DNAME resource records for all the GS1 MOs other than its own.

290 4.1.1. A Note about Alliance Numbers

291 Many MOs in their early years sublicensed GS1 GCPs from larger neighbours rather than managing
 292 their own. Typical of such MOs is GS1 Canada, formerly the Electronic Commerce Council of Canada,
 293 which sublicensed numerous GCPs from GS1 US, then the Uniform Code Council.

294 Reasons for such sublicensing are varied. In Canada's case, because the US is a significant trading
 295 partner and its retailers supported primarily the U.P.C. (now the GTIN-12) at the time, it was necessary
 296 to use US-assigned GCPs. Other countries may not have had MOs of their own at the time the GCPs
 297 were assigned by a neighbouring country but the management of those GCPs was delegated to the
 298 new country MO when the MO was chartered.

299 These numbers are referred to in GS1 as Alliance Numbers. They have the effect of fragmenting the
 300 GS1 namespace beyond the GS1 Prefix level. However, as there is no pattern to the Alliance
 301 Numbers, the delegation of queries for Alliance Numbers has to be the result of a bilateral agreement
 302 between the licensor MO and the licensee MO. When such delegation occurs, it SHALL be done using
 303 DNAME resource records.

304 4.1.2. Modification of the zone in the ONS Peer Root

305 There are multiple scenarios where a ONS Peer Root has to be updated with information pertaining to
 306 its peers. For example, suppose a GS1 MO decides not to manage its own ONS Peer Root but to be
 307 part of another existing peer. Similarly there may be a case where a small GS1 MO which has been
 308 part of a community ONS Peer Root decides to manage its own peer. There also cases wherein a new
 309 ONS Peer Root is added to the ONS network. In all these cases there needs to be a modification in
 310 the ONS Peer Root zone to correctly resolve to the modified peer.

311 The modification for the NS delegations at the concerned ONS Peer Root(s) has to be managed by
 312 the MO. For updating the DNAME redirections in the ONS Peer Roots a Peer Delegation file SHALL
 313 be used. This Peer Delegation file SHALL contain a list of valid ONS Peer Roots and the prefix ranges
 314 supported by each MO. This file SHALL be managed by GS1 Global Office and SHALL be accessible
 315 only to peers in the ONS network.

316 The Peer Delegation file is not used for resolution and in no way should the Peer Delegation file have
 317 a privileged position. The contents of the Peer Delegation file SHALL be used by each ONS Peer Root
 318 to update itself with the modifications that have occurred in any of its peers.

319 The Peer Delegation file SHALL be constructed according to the following XSD:

```
320 <?xml version="1.0" encoding="UTF-8"?>
321 <xs:schema targetNamespace="urn:epcglobal:ons:peerDelegation:xsd:1"
322 xmlns:xs="http://www.w3.org/2001/XMLSchema"
323 xmlns:tns="urn:epcglobal:ons:peerDelegation:xsd:1">
324
325   <xs:complexType name="PrefixRange">
326     <xs:annotation>
327       <xs:documentation>Defines a range of prefixes supported by the enclosing peer
328       node.</xs:documentation>
329     </xs:annotation>
```

```

330         <xs:sequence>
331             <xs:element name="Low" minOccurs="1" maxOccurs="1">
332                 <xs:annotation>
333                     <xs:documentation>The low value (inclusive) of the prefix
334 range.</xs:documentation>
335                 </xs:annotation>
336                 <xs:simpleType>
337                     <xs:restriction base="xs:string">
338                         <xs:length value="3"/>
339                     </xs:restriction>
340                 </xs:simpleType>
341             </xs:element>
342             <xs:element name="High" minOccurs="1" maxOccurs="1">
343                 <xs:annotation>
344                     <xs:documentation>The high value (inclusive) of the prefix
345 range.</xs:documentation>
346                 </xs:annotation>
347                 <xs:simpleType>
348                     <xs:restriction base="xs:string">
349                         <xs:length value="3"/>
350                     </xs:restriction>
351                 </xs:simpleType>
352             </xs:element>
353         </xs:sequence>
354     </xs:complexType>
355
356     <xs:complexType name="PeerNode">
357         <xs:annotation>
358             <xs:documentation>Defines the domain name and range of prefixes for a single node
359 in the ONS federation.</xs:documentation>
360         </xs:annotation>
361         <xs:sequence>
362             <xs:element name="DomainName" type="xs:string" minOccurs="1" maxOccurs="1">
363                 <xs:annotation>
364                     <xs:documentation>The domain name of the peer node.</xs:documentation>
365                 </xs:annotation>
366             </xs:element>
367             <xs:element name="PrefixRange" type="tns:PrefixRange" minOccurs="1"
368 maxOccurs="unbounded">
369                 <xs:annotation>
370                     <xs:documentation>The prefix ranges supported by the peer
371 node.</xs:documentation>
372                 </xs:annotation>
373             </xs:element>
374         </xs:sequence>
375     </xs:complexType>
376
377     <xs:complexType name="PeerDelegation">
378         <xs:annotation>
379             <xs:documentation>Defines the peer nodes involved in the ONS
380 delegation.</xs:documentation>
381         </xs:annotation>
382         <xs:sequence>
383             <xs:element name="LastUpdated" type="xs:dateTime" minOccurs="1" maxOccurs="1">
384                 <xs:annotation>
385                     <xs:documentation>The date and time the file was last
386 updated.</xs:documentation>
387                 </xs:annotation>
388             </xs:element>
389             <xs:element name="PeerNode" type="tns:PeerNode" minOccurs="1"
390 maxOccurs="unbounded">
391                 <xs:annotation>
392                     <xs:documentation>Peer nodes involved in the ONS
393 delegation.</xs:documentation>
394                 </xs:annotation>
395             </xs:element>
396         </xs:sequence>
397     </xs:complexType>
398
399     <xs:element name="PeerDelegation" type="tns:PeerDelegation"/>
400 </xs:schema>

```


401 In case there is a modification for a domain name for any GS1 Prefix, the responsible MO SHALL
402 notify GS1 Global Office so that the Peer Delegation file may be updated. The Peer Delegation file is
403 then updated appropriately and the "LastUpdated" field set to the current date and time.

404 Each peer accesses the Peer Delegation file at a scheduled time and compares the "LastUpdated"
405 field with the date and time it was last used. In the event that the "LastUpdated" field is later than the
406 last date and time the Peer Delegation file was used, the new Peer Delegation file is used to update
407 the DNAME resource records for the ONS Peer Nodes. Each peer SHALL check for updates to the
408 Peer Delegation file at most every 24 hours and SHALL update its local zone file if the file has been
409 updated.

410 5. ONS Formal Specification

411 The formal specification of ONS is a set of technical rules and procedures to be followed by
412 Publishers, Clients, and Peer Node Operators (MOs).

413 An ONS Client is an application that wishes to use ONS to identify a service that may provide
414 information related to a specific GS1 Identification Key. An ONS Publisher is an entity responsible for
415 making services available to ONS Clients by creating service pointer entries in an ONS Peer Root. An
416 ONS Peer Root is an implementation of a DNS Server; because ONS differs from DNS only in what
417 data is provided by the server, not in the operation of the server itself, there is no separate
418 specification for an ONS Peer Root. Any DNS server compliant with [DNS] and [RFC 3403] may be
419 used as an ONS Peer Root.

420 The ONS specification consists of four ingredients:

- 421 • The DDDS Application Specification (Section 6) for ONS.
- 422 • A procedure (Section 7.1) that an ONS Client SHALL follow in order to present a query to
423 ONS. This procedure specifies how a GS1 Identification Key is converted to a DNS NAPTR
424 query.
- 425 • A set of rules (Section 7.2) that ONS Publishers SHALL follow to represent ONS information
426 (namely, pointers to services for GS1 Identification Keys) as DNS NAPTR records within an
427 ONS Peer Root.
- 428 • A procedure (Section 8) that an ONS Client SHALL follow in order to interpret the results of an
429 ONS query. This procedure specifies how an ONS Client can locate a service using the
430 information provided by the ONS Peer Root.

431 6. ONS DDDS Application Specification

432 This section defines how the ONS client uses the methods explained in RFC 3402 and RFC 3403 to
433 resolve a GS1 Identification Key. The DDDS database used by this application is the DNS database
434 as explained in RFC 3403.

435 6.1. Application Unique String (AUS)

436 As per RFC 3402, the AUS is the initial input to a DDDS application. The DDDS application here is the
437 ONS client.

438 The AUS is composed of:

- 439 ■ the lower case ISO 639-1 language code (optional) of the client querying the database;
- 440 ■ the lower case ISO 3166-1 alpha-2 country code (optional) of the client querying the
441 database;

- 442
- 443
- the GS1 Identification Key type abbreviation in lower case; and
 - the full-length GS1 Identification Key itself, less any serial number.

444 Each field in the AUS is separated by the vertical bar ('|', ASCII 124 decimal or 7C hexadecimal). This
 445 character has been chosen as it is not legitimately part of any of the fields within the AUS. Optional
 446 fields that have no value are encoded using empty strings; all field separators are required.

447 Valid GS1 Identification Key type abbreviations in lower case are:

Abbreviation (lower case)	GS1 Identification Key type
gtin	Global Trade Item Number
sscc	Serial Shipping Container Code
gln	Global Location Number
grai	Global Returnable Asset Identifier
giai	Global Individual Asset Identifier
gsrn	Global Service Relation Number
gdti	Global Document Type Identifier
gsin	Global Shipment Identification Number
ginc	Global Identification Number for Consignment

448 6.1.1. Converting an EPC to an AUS

449 In order to query the DNS for an EPC, the EPC URI form must be converted to a GS1 Identification
 450 Key. Note that this applies only to EPCs that are derived from GS1 Identification Keys; EPCs
 451 representing the General Identifier (GID), US Department of Defense Identifier (DOD), the Aerospace
 452 and Defense Identifier (ADI), and any others cannot be transformed into GS1 Identification Keys and
 453 are therefore not supported.

454 The procedure for the conversion is documented in Section 7 of the EPCglobal Tag Data Standards
 455 [EPC].

456 6.1.2. Examples (non-normative)

- 457
- 458
1. The GTIN-13 0614141322602 for a client that is independent of language and country is represented as an AUS as:

459 |gtin|00614141322602

460 Note the addition of the digit '0' to the beginning to pad the GTIN to a full 14 digits.

- 461
- 462
2. The GIAI 0614141997uc-171+UX for a client that is independent of language but is looking for a Brazilian service is represented as an AUS as:

463 |br|giai|0614141997uc-171+UX

- 464
- 465
3. The GDTI 0614141665815 with serial number 999888777 for a client that is looking for an English service in Canada is represented as an AUS as:

466 en|ca|gdti|0614141665815

- 467
- 468
4. The SSCC 306141417782246356 for a client that is looking for a French service in any country is represented as an AUS as:

469 fr||sscc|306141417782246356

470 6.2. First Well-Known Rule

471 As per RFC 3402, the first well-known rule is a rewrite rule that is defined by the application (i.e. ONS
472 client). This rule is used to produce the first valid key.

473 This rule applies only to GS1 Identification Keys; while ONS is not required to explicitly support non-
474 GS1 identification systems, it must not explicitly impede such systems either. The first well-known rule
475 is applied to the GS1 Identification Key and GS1 Identification Key type. In general, the format of the
476 first valid key for ONS is:

```
477     <transformation of the identification key>.<identification key  
478     type>.<organisation namespace>.id.<valid ONS Peer Root domain name>
```

479 For GS1:

- 480 ■ transformation of the identification key is explained below;
- 481 ■ identification key type is the lower-case GS1 Identification Key type abbreviation defined in
482 Section 6.1;
- 483 ■ organisation namespace is “gs1”; and
- 484 ■ valid ONS Peer Root domain name is one that is formally approved as a ONS Peer Root by
485 GS1.

486 Note that non-GS1 identification systems which may implement ONS may use ONS domains wholly
487 separate from those used by GS1.

488 The first valid key for a GS1 AUS as follows:

- 489 ■ Start with an empty string.
- 490 ■ Append a transformation of the GS1 Identification Key within the AUS.
 - 491 ○ Strip the optional serial number if applicable (GDTI and GRAI).
 - 492 ○ Strip the checksum digit if applicable (all except GIAI and GINC which, as alphanumeric
493 GS1 Identification Keys, do not have a checksum digit).
 - 494 ○ Hold the leading digit (GTIN indicator digit, SSCC extension digit, GRAI zero) in its position
495 if applicable.
 - 496 ○ Reverse all the remaining characters.
 - 497 ○ For each character:
 - 498 ■ If the character is a number or a lower-case letter, leave unmodified.
 - 499 ■ If the character is an upper-case letter, convert it to lower case and prefix it with the
500 letter ‘u’.
 - 501 ■ Otherwise, convert the character to its two-digit hexadecimal equivalent and prefix it
502 with the letter ‘x’.
 - 503 ○ For each character or its conversion as per the previous step, append a ‘.’ (period).
- 504 ■ Append the identification key type.
- 505 ■ Append “.gs1.id.”.
- 506 ■ Append a valid ONS Peer Root domain name.

507 Although a single AUS may be encoded as the first valid key in exactly the number of ways matching
508 the number of ONS Peer Root domains (see the last step above), each instance shall be equivalent
509 and shall resolve to the same results.

510 6.2.1. Examples (non-normative)

511 1. Given the GTIN-13 0614141322602 as the primary identifier for a product, its representation as
512 an AUS for a client that is independent of language and country is:

513 |gtin|00614141322602

514 Its representation as the first valid key with onsepc.com as the starting point for the query is:

515 0.0.6.2.2.3.1.4.1.4.1.6.0.gtin.gs1.id.onsepc.com

516 Its representation as the first valid key with ons.epcglobalcanada.org as the starting point for the
517 query is:

518 0.0.6.2.2.3.1.4.1.4.1.6.0.gtin.gs1.id.ons.epcglobalcanada.org

519 Even though the string appended is different for the above two examples, they will finally resolve
520 to the same service/information that is appropriate to the GS1 Identification Key. The two domains
521 are equivalent.

522 2. Given the GTIN 50614141322607 as the primary identifier for a product, its representation as an
523 AUS for a client that is independent of language but is looking for a Brazilian service is:

524 |br|gtin|50614141322607

525 Its representation as the first valid key with onsepc.com as the starting point for the query is:

526 5.0.6.2.2.3.1.4.1.4.1.6.0.gtin.gs1.id.onsepc.com

527 Its representation as the first valid key with ons.epcglobalcanada.org as the starting point for the
528 query is:

529 5.0.6.2.2.3.1.4.1.4.1.6.0.gtin.gs1.id.ons.epcglobalcanada.org

530 The two domains are equivalent.

531 3. Given the GIAI 0614141997uc-171+UX as the primary identifier for an asset, its representation
532 as an AUS for a client that is looking for an English service in Canada is:

533 en|ca|giai|0614141997uc-171+UX

534 This GS1 Identification Key has a mix of lower- and upper-case characters, digits, and punctuation
535 characters. The lower-case characters and digits are translated as is, the upper-case characters
536 are escaped with "u", and the punctuation characters are escaped with "x" and their hexadecimal
537 equivalent.

538 Its representation as the first valid key with onsepc.com as the starting point for the query is:

539 ux.uu.x2b.1.7.1.x2d.c.u.7.9.9.1.4.1.4.1.6.0.giai.gs1.id.onsepc.com

540 Its representation as the first valid key with ons.epcglobalcanada.org as the starting point for the
541 query is:

542 ux.uu.x2b.1.7.1.x2d.c.u.7.9.9.1.4.1.4.1.6.0.giai.gs1.id.ons.epcglobalcan
543 ada.org

544 The two domains are equivalent.

545 4. Given the GDTI 0614141665815 with serial number 999888777 as the primary identifier for a
546 document, its representation as an AUS for a client that is looking for an English service in any
547 country is:

548 en|gdti|0614141665815

549 Its representation as the first valid key with onsepc.com as the starting point for the query is:

550 1.8.5.6.6.1.4.1.4.1.6.0.gdti.gs1.id.onsepc.com

551 Its representation as the first valid key with `ons.epcglobalcanada.org` as the starting point for the
552 query is:

553 `1.8.5.6.6.1.4.1.4.1.6.0.gdti.gs1.id.ons.epcglobalcanada.org`

554 The two domains are equivalent.

555 **5.** Given the SSCC `306141417782246356` as the primary identifier for a shipping container, its
556 representation as an AUS for a client that is looking for a French service in any country is:

557 `fr||sscc|306141417782246356`

558 Its representation as the first valid key with `onsepc.com` as the starting point for the query is:

559 `3.5.3.6.4.2.2.8.7.7.1.4.1.4.1.6.0.sccc.gs1.id.onsepc.com`

560 Its representation as the first valid key with `ons.epcglobalcanada.org` as the starting point for the
561 query is:

562 `3.5.3.6.4.2.2.8.7.7.1.4.1.4.1.6.0.sccc.gs1.id.ons.epcglobalcanada.org`

563 The two domains are equivalent.

564 **6.2.2. A note about non-numeric characters**

565 The following statements are true:

566 **1.** The GCP is numeric only.

567 **2.** The first non-numeric character encountered in a GS1 Identification Key must therefore be beyond
568 the GCP.

569 **3.** The GS1 Identification Keys that support non-numeric characters (GLN with extension, GRAI,
570 GIAI, GDTI, and GINC) do so only at the instance or serial level, not at the class level.

571 **4.** "Section 4.3 – Granularity of identification" in the Federated ONS Requirements Document states
572 that: "The granularity of identification (the lowest level at which queries can be made) shall be at
573 minimum one level above the serial number in the identifier. ONS is about discovering class-level
574 services only."

575 It is therefore possible to stop encoding using the first well known rule the moment the first non-
576 numeric character is encountered. However, there is no guarantee that the above restriction will be in
577 place in the next iteration of ONS so to maximize the future benefits of ONS the domain name SHALL
578 be encoded in its entirety as above.

579 **7. ONS DDDS database**

580 The DDDS database used for ONS resolution is the DNS. RFC 3403 specifies a DDDS Database that
581 uses the NAPTR DNS resource record to contain the rewrite rules. The Keys for this database are
582 encoded as domain-names. Section 6 explains how the keys are generated.

583 This domain-name is used to request NAPTR records which may contain the end result. A brief
584 description of the contents of the database is explained in the following subsections.

585 **7.1. DNS query format**

586 The following specifies the procedure an ONS client SHALL follow to present a query to ONS for a
587 GS1 Identification Key:

588 **1.** Begin with an AUS as defined in Section 6.1.

- 589
590
2. Follow the procedure in Section 6.2 to convert the AUS into a domain name, ending with the host domain name of a valid ONS Peer Root, for example:
- 591
592
- ```
5.0.6.2.2.3.1.4.1.4.1.6.0.gtin.id.gs1.onsepc.com.
```
3. Use a DNS resolver to query for DNS Type Code 35 (NAPTR) records [RFC 3403] for the domain name from Step 2. The method for obtaining and using the DNS resolver is outside the scope of this specification. An ONS Client MAY use any DNS resolver conforming to [DNS], using whatever API is available.
- 593  
594  
595  
596

## 597 7.2. NAPTR RR

598 ONS contains pointers to authoritative information for GS1 Identification Keys in the DNS database.  
599 Each such pointer takes the form of a DNS Type Code 35 (NAPTR) record [RFC 3403]. As explained  
600 in section 8.1, the ONS client resolves for the NAPTR records with the domain names as the input.  
601 This section specifies how ONS Publishers must encode information into NAPTR records.

602 The contents of a DNS NAPTR record are logically formatted as follows:

| Order | Pref | Flags | Service       | Regexp                                   | Replacement  |
|-------|------|-------|---------------|------------------------------------------|--------------|
| 0     | 0    | u     | (Service URL) | !^.*\$!http://example.com/cgi-bin/epcis! | . (a period) |

603 ONS Publishers SHALL obey the following rules:

- 604
- The **Order** field SHALL be a non-negative integer.
- 605 *Explanation (non-normative): The Order field is used in DNS applications where a series of*  
606 *regular expressions from distinct NAPTR records are applied consecutively to an input. Where*  
607 *multiple service implementations are available (e.g. segmented by language and/or country),*  
608 *the Order field specifies the order in which the rules are applied to the AUS.*
- The **Pref** field SHALL be a non-negative integer. The value of the **Pref** field is an ordinal that specifies that the service in one record is preferred to the service in another record having the same **Service** field. An ONS Client SHOULD attempt to use a service having a lower **Pref** number before using an equivalent service having a higher **Pref** number. The **Pref** field may be used, for example, to select a primary, high-capacity server over a secondary backup server to be used only when the primary server fails.
  - The **Flags** field SHALL be set to 'u' to indicate that the **Regexp** field contains a URI OR 't' to indicate that the **Regexp** field contains plain text.
  - The **Service** field contains an indicator of the type of service that can be found at the URI in question. This feature allows for the ONS service to indicate different services for different classes of service. The value of the Service field SHALL take the form of a URL and be interpreted as defined in Section 7.2.2.
  - The **Regexp** field specifies an expression to be applied to the AUS together with a replacement for the AUS should there be a match to the expression. The value of this field SHALL be of the form *!expression!substitution!* where the exclamation mark is the field delimiter, *expression* is pattern to match to the AUS, and *substitution* is the replacement string, either a URI or text as appropriate (depending on the flag).
  - The **Replacement** field is not used by ONS but as it is a special DNS field its value SHALL be set to a single period ('.') instead of simply a blank.

626 A detailed explanation of certain fields in the NAPTR resource records are given here:  
627  
628



## 629 7.2.1. Flags

630 The following flags are defined:

- 631 ■ “u” means that the output of the rule is a URI (typically a URL to a web service or a web  
632 page). Interpretation of the URI and the interaction model with the target (if any) of the URI is  
633 dependent on the service type identifier.
- 634 ■ “t” means that the output of the rule is plain text. Interpretation of the text is dependent on the  
635 service type identifier.

636 The remaining alphabetic flags are reserved for future versions of this specification. The numeric flags  
637 may be used for local experimentation. The flags are all mutually exclusive and resolution libraries  
638 MAY signal an error if more than one is given. (Experimental code and code for assisting in the  
639 creation of Rewrite Rules would be more likely to signal such an error than a client such as a browser.)  
640 Multiple flags may be allowed in the future, so implementers MUST NOT assume that the flags field  
641 can contain only 0 or 1 characters. Finally, if a client encounters a record with an unknown flag, it  
642 MUST ignore it and move to the next Rule. This test takes precedence over any ordering since flags  
643 can control the interpretation placed on fields. A novel flag might change the interpretation of the regex  
644 and/or replacement fields such that it is impossible to determine if a record matches a given target.

645 The above flags are terminal flags and they halt the looping of the DDDS algorithm. If none of the  
646 above flags is present, clients may assume that another Rule exists at the Key produced by the  
647 current Rewrite Rule.

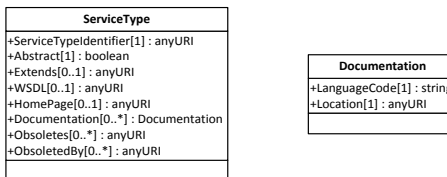
## 648 7.2.2. Services

649 ONS is required to support a dynamic service definition model wherein any service may be defined by  
650 GS1 Global, a GS1 Member Organization, an industry group, a member, a service provider, or in fact  
651 any entity that wishes to publish services based on GS1 identification keys in ONS. The service  
652 definitions need to support scope definitions at the global and local levels.

653 Service type identifiers for this Application SHALL take the form of a URL.

654 The service type URL on its own SHALL point to an XML document describing the service. Note that  
655 this does not imply that the XML document is publicly accessible; if the service type has restricted  
656 circulation (e.g. within an industry group), the organization defining the service type MAY require  
657 credentials to access the document (e.g. basic HTTP authentication, FTP username and password).

658 The XML document shall have the following structure:



659  
660 The structure is represented by the following XSD:

```

661 <?xml version="1.0" encoding="UTF-8"?>
662 <xs:schema targetNamespace="urn:epcglobal:ons:serviceType:xsd:1"
663 xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:tns="urn:epcglobal:ons:serviceType:xsd:1">
664
665 <xs:complexType name="ServiceType">
666 <xs:sequence>
667 <xs:element name="ServiceTypeIdentifier" type="xs:anyURI" minOccurs="1"
668 maxOccurs="1" />
669 <xs:element name="Abstract" type="xs:boolean" minOccurs="1" maxOccurs="1" />
670 <xs:element name="Extends" type="xs:anyURI" minOccurs="0" maxOccurs="1" />
671 <xs:element name="TimeToLive" type="xs:int" minOccurs="1" maxOccurs="1" />
672 <xs:element name="WSDL" type="xs:anyURI" minOccurs="0" maxOccurs="1" />
673 <xs:element name="HomePage" type="xs:anyURI" minOccurs="0" maxOccurs="1" />

```

```

674 <xs:element name="Documentation" type="tns:Documentation" minOccurs="0"
675 maxOccurs="unbounded"/>
676 <xs:element name="Obsoletes" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded"/>
677 <xs:element name="ObsoletedBy" type="xs:anyURI" minOccurs="0"
678 maxOccurs="unbounded"/>
679 </xs:sequence>
680 </xs:complexType>
681
682 <xs:complexType name="Documentation">
683 <xs:sequence>
684 <xs:element name="LanguageCode" type="xs:string" minOccurs="1" maxOccurs="1"/>
685 <xs:element name="Location" type="xs:anyURI" minOccurs="1" maxOccurs="1"/>
686 </xs:sequence>
687 </xs:complexType>
688
689 <xs:element name="ServiceType" type="tns:ServiceType"/>
690
691 </xs:schema>

```

### 692 7.2.2.1. Service Type Class

693 The ServiceType class is the definition of the service type itself.

| Field                 | Description                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ServiceTypeIdentifier | The service type identifier, also the URL at which the XML representation of this structure may be found.                                                                                                                                                                                                                                                                                    |
| Abstract              | True if this is an abstract service type. An abstract service type is one that is used as the base for other service types rather than having a concrete implementation of its own. An example of an abstract service type is a generic data retrieval service that returns structured data whose contents are specified more concretely by the service types that extend this service type. |
| Extends               | The service type identifier that this service type extends if any.                                                                                                                                                                                                                                                                                                                           |
| TimeToLive            | The time to live (TTL) in minutes. The document may be cached by a client only for as long as specified in this field. This allows a publisher to be certain that updates to fields subsequent to this one will be picked up in a timely fashion. The recommended value is 10080 minutes (one week).                                                                                         |
| WSDL                  | The location of the Web Service Description Language file if any. Not all service types will be based on Web Services but for those that do, this field is required.                                                                                                                                                                                                                         |
| HomePage              | The location of a human-readable home page if any. Typically, the home page for a service type is a page that provides a basic outline for the service type and may contain links to other, related service types. The presence, structure, and content of a home page are entirely at the discretion of the service type author.                                                            |
| Documentation         | Documentation, possibly in multiple languages, for this service type. Supplements the documentation of the "Extends" service type if any.                                                                                                                                                                                                                                                    |



| Field       | Description                                                                                                                                                                                                                                                                                                                |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Obsoletes   | A list of service type identifiers that this service type obsoletes. As services evolve, new versions will occasionally replace old versions; this provides the necessary continuity between versions. The target service types should have the service type identifier for this service type in their "ObsoletedBy" list. |
| ObsoletedBy | A list of service type identifiers that obsolete this service type. As services evolve, new versions will occasionally replace old versions; this provides the necessary continuity between versions. The target service types should have the service type identifier for this service type in their "Obsoletes" list.    |

#### 694 7.2.2.2. Documentation class

695 The service type documentation in a given language. This is a contained class that exists only within  
 696 the context of the service type. Documentation may or may not exist for a service type at the discretion  
 697 of the service type definer.

| Field        | Description                                                                                                                                             |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| LanguageCode | The ISO language code representing the language in which the human-readable documentation is written, unique within the service type.                   |
| Location     | The URL for human-readable documentation in the language specified. The presence of the URL field does not mean that the target is publicly accessible. |

#### 698 7.2.3. Regular expressions (non-normative)

699 The reason the **Regexp** field is in the form of a regular expression is that the NAPTR record is used by  
 700 other applications that have the need to conditionally rewrite the URI or text to include other  
 701 information. While none of the examples here make use of this feature, it has not been determined if  
 702 this will always be the case. In the future it may become necessary to allow full regular expression and  
 703 replacement functions within the Regexp field. Therefore, implementers would be wise to not assume  
 704 that the URI or text can simply be extracted without any regular expression processing.

705 The general form of the Regexp field is as a Posix Extended Regular Expression. This form states that  
 706 the first character encountered is the field delimiter between the regular expression and the  
 707 replacement portion of the entire rewrite expression. In the example at the beginning of this  
 708 Section 7.2 the delimiter is the exclamation point (!) character. The regular expression portion in most  
 709 cases is `^.*$` which equates to 'match anything'. The substitution portion is text (for text services) or a  
 710 URI (e.g. `http://example.com/cgi-bin/epcis`). The choice of '!' as the delimiter instead of a  
 711 more traditional '/' makes the entire line much easier to read and less error prone.

#### 712 7.2.4. Service type hierarchy (non-normative)

713 An application may be interested in multiple services, either directly by supporting multiple interaction  
 714 models with multiple service types, indirectly by supporting a service type and all service types derived  
 715 from it (see Section 7.2.2), or a combination of the two. Because of this, the necessary steps after  
 716 issuing the query are to filter the results by service type and then group by service type.

- 717 The concept of inheritance in services is no different from that as applied to traditional object-oriented  
718 programming languages. Take, for example, the concept of a Product Recall in a consumer-based  
719 Mobile Commerce application.
- 720 In general, a Product Recall describes what is being recalled (e.g. product and set of lot numbers),  
721 how to identify what is being recalled (e.g. locating the lot number on the package), and what to do to  
722 process the recall (e.g. returning to the store from which it was purchased).
- 723 Local regulations may require additional data, such as a consumer hotline number to call for more  
724 information. As a result, there may be not only a global Product Recall service, but one defined for  
725 Canada, another for Germany, and so on. The country-specific Product Recall service types each  
726 have the URL of the global Product Recall service type in their “Extends” field.
- 727 A Canadian consumer, Alice, has a Mobile Commerce application, customized for Canada, on her  
728 device. In addition to displaying product data, images, and consumer reviews, the application also has  
729 the ability to discover and interact with a Product Recall service. She is in Canada and scans a  
730 product; through the ONS lookup process, she finds the Canadian Product Recall service, calls it, and  
731 is given either a null response (product is not subject to a recall) or full details of the recall, including  
732 any Canadian extensions.
- 733 Another Canadian consumer, Bob, has a similar application, but one that does not understand the  
734 Canadian extensions. He scans the same product, looking for the global Product Recall service type.  
735 The application doesn’t find it, but it does find the Canadian one, and inspecting the service type  
736 reveals that the Canadian service is just an extension of the global service. Bob’s application can  
737 interact with the Canadian, but only with the methods and data structures supported by the global  
738 service. Nevertheless, Bob gets the data he is looking for.
- 739 Alice now travels the Germany. A product scanned in Germany is likely to return the German Product  
740 Recall service type, not the global or the Canadian one. In order for Alice’s application to work, it has  
741 to understand not only the specific Canadian Product Recall service type but also the global Product  
742 Recall service type so that it can interact with the German Product Recall services in the same manner  
743 as Bob’s application.
- 744 For further details and a flow diagram for discovering derived service types, see Section 9.4.

## 745 8. Processing ONS Query Responses

- 746 ONS Clients SHALL use the following procedure to interpret the results returned by an ONS query as  
747 formulated in Section 7.1.
- 748 1. The result from the ONS query is a set of NAPTR records as described in Section 7.2. If no entry  
749 exists for the query for whatever reason, the result SHALL be considered to be an empty set.
  - 750 2. Remove all records from Step 1 where the Service field is not supported by the application.
    - 751 a. A Service is supported by the application if it is explicitly declared to be supported or one of its  
752 base classes is declared to be supported and extensions of that base class are permitted by the  
753 application.
  - 754 3. Group the results from Step 2 by the Service field.
  - 755 4. For each group in the results from Step 3:
    - 756 a. Order the results according to the Order field.
    - 757 b. For each result, apply the Regexp substitution expression to the AUS. If a match is found,  
758 continue only with those results where the Order field is the same as the one in which there is  
759 the first match.
    - 760 c. From among the results in Step b, select those with the lowest Pref field value.
    - 761 d. From among the results in Step c, select a record at random.

- 762 e. Interpret the result in Step d as determined by the Flags.
- 763 i. For a text field, apply the text as per the rules of the Service.
- 764 ii. For a URI field, use the URI as per the rules of the Service.
- 765 f. If Step e is not successful, repeat Step d, excluding this and all previously failed selections. If all
- 766 records from Step c have been tried, from among the results in Step b, select those having the
- 767 next lowest value in the Pref field and return to Step d. If all records from Step b have been
- 768 tried, stop: no service is available.

## 769 9. Examples (non-normative)

770 In the following examples the GS1 Identification Key in question is the GTIN 50614141322607 which

771 represents a case of Example Corporation Model 100 Widgets.

772 The ONS client attempts to learn about this product by first following the procedure in Section 4, which

773 converts the GS1 Identification Key into this domain name:

774 `5.0.6.2.2.3.1.4.1.4.1.6.0.gtin.gs1.id.onsepc.com`

775 The application then queries the DNS for NAPTR records for that domain name and receives the

776 following records (The record number column is for reference only and not part of the result set):

777

| # | Order | Pref | Flags | Service                           | Regexp                                         | Replacement |
|---|-------|------|-------|-----------------------------------|------------------------------------------------|-------------|
| 1 | 0     | 0    | t     | http://www.gs1.org/ons/prefix     | !^.*\$!06,7!                                   | .           |
| 2 | 0     | 0    | u     | http://www.gs1.org/ons/epcis      | !^.*\$!http://epcis.example.com/!              | .           |
| 3 | 0     | 0    | u     | http://www.gs1.org/ons/mobilecomm | !^fr\ ca\ .*\$!http://fr-ca.example.com/!      | .           |
| 4 | 1     | 0    | u     | http://www.gs1.org/ons/mobilecomm | !^[a-z]*\ fr\ .*\$!http://all-fr.example.com/! | .           |
| 5 | 2     | 0    | u     | http://www.gs1.org/ons/mobilecomm | !^fr\ .*\$!http://fr-all.example.com/!         | .           |
| 6 | 3     | 0    | u     | http://www.gs1.org/ons/mobilecomm | !^es\ us\ .*\$!http://es-us1.example.com/!     | .           |
| 7 | 3     | 1    | u     | http://www.gs1.org/ons/mobilecomm | !^es\ us\ .*\$!http://es-us2.example.com/!     | .           |
| 8 | 4     | 0    | u     | http://www.gs1.org/ons/mobilecomm | !^.*\$!http://en-all.example.com/!             | .           |

778 Note: All of the URLs in this table are fictitious and do not represent any standard current or proposed. Final URLs for services defined by GS1  
 779 may not match those shown here.

- 780 Each of these records conforms to the rules specified in Section 7. The interpretation is as follows (the  
781 numbers in the list below match the record number on the previous page):
- 782 1. In this example, the service type represents a parsing of the GS1 Company Prefix. The flags  
783 specify text as the record type, which in this case represents the leading digits of the GS1  
784 Company Prefix and its length. The string "06,7" means that all GS1 Company Prefixes starting  
785 with "06" are seven digits long; the GS1 Company Prefix is therefore "0614141".
  - 786 2. The service type represents EPCIS. The flags specify a URI as the record type: the regular  
787 expression substitution results in the URL of the EPCIS server for every AUS ("^.\*\$" matches all  
788 strings).
  - 789 3. The service type represents a Mobile Commerce data provider. The flags specify a URI as the  
790 record type: the regular expression substitution results in the URL of the Mobile Commerce server  
791 for French Canadian data. The order of "0" (the lowest in all equal service types) means that this is  
792 the first record whose regular expression is applied to the client AUS.
  - 793 4. Same service type as before, but is applied to the client AUS if the previous regular expression  
794 produced no match. The regular expression matches all clients operating in France (note the  
795 character set match for all languages).
  - 796 5. Same service type as before, but is applied to the client AUS if the previous regular expression  
797 produced no match. The regular expression matches all clients operating in French in all  
798 countries.
  - 799 6. Same service type as before, but is applied to the client AUS if the previous regular expression  
800 produced no match. The regular expression matches all clients operating in Spanish in the United  
801 States.
  - 802 7. This is the same as the previous (the order is the same) and so will match the string as well, but  
803 will return a different URL as a result of the substitution and has a different preference. The URL in  
804 this record will be used if the first URL does not respond.
  - 805 8. Same service type as before, but is applied to the client AUS if the previous regular expression  
806 produced no match. The regular expression matches all clients operating in all countries.
- 807 Finally, depending on the service that the ONS Client desires, it uses one or more of the records  
808 returned to locate an appropriate service. The following sections describe specific examples of  
809 services an ONS Client might locate. In each case, the ONS Client uses the procedure specified in  
810 Section 8 to locate a service, using the records returned above.

## 811 9.1. Finding the length of the GS1 Company Prefix (GCP)

812 One of the simplest but most powerful examples is association some application-specific text with a  
813 GS1 Identification Key. In the above example, the Service URL `http://www.gs1.org/ons/prefix` is  
814 associated with the length of the GCP. The text returned is interpreted as "for any GS1 Identification  
815 Key starting with '06', the length of the GCP is 7". This allows applications that need to encode RFID  
816 tags for a product where the length of the GCP is not previously known (e.g. when a distributor has to  
817 add RFID tags to a manufacturer's product) to encode the tags properly.

818 The application issues the query and receives the NAPTR records above. It iterates through the list  
819 looking for the required Service URL which returns text that can be parsed by an appropriate  
820 application to determine the length of the GCP. It locates that service in the first record and returns the  
821 text found in the Regexp field. The application then parses the text according to the rules defined by  
822 the service.

## 823 9.2. Finding an authoritative EPCIS server for a product

824 This example shows how a GS1 Identification Key can be used to retrieve a pointer to an EPCIS  
825 server associated with the owner of the GS1 Identification Key. Again, using the same results from  
826 above, the client uses the second record and extracts the URI from the Regexp. It then uses that URI  
827 as the end point to which to send the EPCIS query.

## 828 9.3. Finding a Mobile Commerce service for product

829 This example shows how to interpret multiple instances of Mobile Commerce services according to the  
830 language and country requirements of a client.

831 A client looking for information in French in Canada would choose record 3. A client looking for  
832 information in any language in France would choose record 4. A client looking for information in  
833 French in any country would choose record 5.

834 A client looking for information in Spanish in the United States would choose records 6 and 7; the  
835 client would first try to access the URL in record 6 and only if that fails would it fall over to the URL in  
836 record 7.

837 All other clients would choose record 8 as the default as it matches all languages and countries.

## 838 9.4. Dynamic interaction with extended services

839 The ability to define one service as an extension of another is particularly useful when dealing with  
840 web services that have to interact in certain ways, typically to retrieve data.

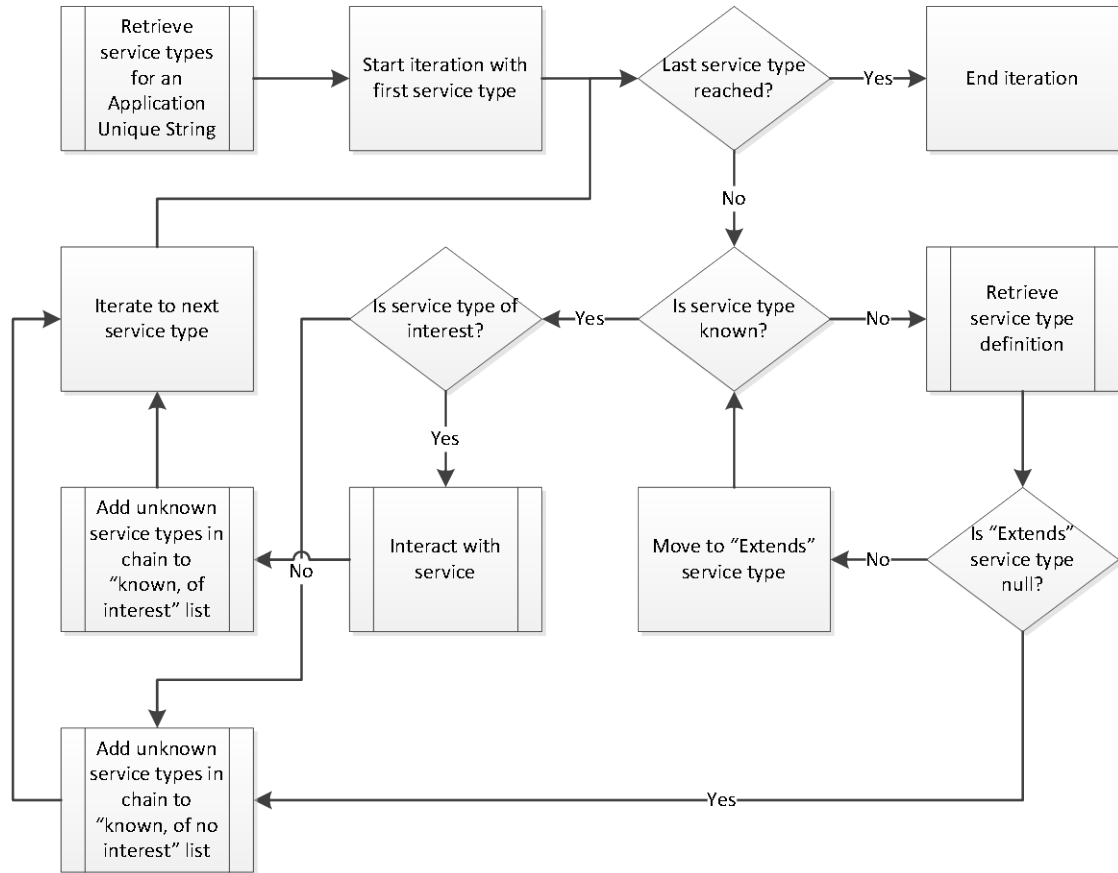
841 Assume, for example, that GS1 has defined an abstract service type with a single “getData()” method  
842 that takes a GS1 Identification Key as a parameter. The return type for this method is a basic object  
843 type with no attributes of interest.

844 Now assume that the Mobile Commerce working group has defined a number of data objects for  
845 products: nutrition data, allergen data, health certification data, and so on. Each of these data objects  
846 has a corresponding service type to define their particular implementation of the “getData()” method.

847 A Mobile Commerce application that is aware of these service types can easily discover them when  
848 querying ONS and can interact with services that are identified by these service types. When dealing  
849 with nutrition data, for example, the Mobile Commerce application can trigger warnings when the  
850 nutrition content of the product exceeds the user’s dietary requirements.

851 Suppose, however, that the Mobile Commerce developer wants to design the application to retrieve  
852 any data defined for a product, not just those that are hard-coded into the application. In theory, the  
853 application can interact with any service type that extends the generic data retrieval service type; it  
854 can’t interpret the data as it has no logic to understand its meaning, but it can at least display it to the  
855 user.

856 The discovery and interaction model would look like something like this:



857  
858  
859  
860

Using this model, the Mobile Commerce application can interact with any service type that implements the generic data retrieval framework. For performance purposes, it can build up its list of known service types so that it can limit the number of queries to retrieve the service type definition.

861

## 9.5. Sample ServiceType XML

862

The following is a sample ServiceType XML file:

863

```
<?xml version="1.0" encoding="UTF-8"?>
<tns:ServiceType xmlns:tns="urn:epcglobal:ons:serviceType:xsd:1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.gs1.org/ons/xsd/1/ServiceType.xsd">
```

864

865

866

867

```
<ServiceTypeIdentifier>http://www.example.com/ons/SampleServiceType.xml</ServiceTypeIdentifier>
```

868

869

```
>
 <Abstract>>false</Abstract>
 <Extends>http://www.example.com/ons/SampleBaseServiceType.xml</Extends>
```

870

871

872

873

874

875

```
<TimeToLive>10080</TimeToLive>
<WSDL>http://service.example.com/services/SampleService?wsdl</WSDL>
<HomePage>http://www.example.com/projects/SampleProject.xhtml</HomePage>
<Documentation>
```

876

877

878

879

```
 <LanguageCode>en</LanguageCode>
 <Location>http://www.example.com/projects/en/SampleServiceType.xhtml</Location>
</Documentation>
<Documentation>
 <LanguageCode>fr</LanguageCode>
 <Location>http://www.example.com/projects/fr/SampleServiceType.xhtml</Location>
</Documentation>
```

880

881

882

883

```
</tns:ServiceType>
```

884

The service type XML is self-referential; its location on the Internet is

885

<http://www.example.com/ons/SampleServiceType.xml>. It is a concrete (not abstract) service type. It



886 extends the base service type <http://service.example.com/services/SampleService?wsdl>, has a  
887 WSDL at <http://service.example.com/services/SampleService?wsdl>, and has a human-readable  
888 home page at <http://www.example.com/projects/SampleProject.xhtml>. Detailed documentation is  
889 available in English and in French.

890 It should be pointed out that the suggested nomenclature here does not preclude that the same home  
891 page being shared by multiple service types. The example here shows that the home page is based  
892 on a project, which may define multiple service types. The same applies to documentation; there is no  
893 reason for a documentation page to apply to a single service type, though the names of the  
894 documentation URLs suggest that they apply to this service type only.

## 895 10. References

### 896 EPC

897 EPCglobal, "EPCglobal Tag Data Standards Version 1.6," EPCglobal Ratified Standard, September  
898 2011. (See <http://www.epcglobalinc.org/standards/tds>)

### 899 DNS

900 (Internet Engineering Task Force). STD0013, *RFC 1034, RFC 1035*, ed Mockapetris, P. 2000. (See  
901 <http://www.ietf.org/rfc/std/std13.txt>)

### 902 RFC 2396

903 T. Berners-Lee, R. Fielding, L. Masinter. *Uniform Resource Identifiers (URI): Generic Syntax*, 1998.  
904 (See <http://www.ietf.org/rfc/rfc2396.txt>)

### 905 RFC 2672

906 Crawford, P. *Non-Terminal DNS Name Redirection*, 1999. (See <http://www.ietf.org/rfc/rfc2672.txt>)

### 907 RFC 3403

908 Mealling, Michael. *Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name*  
909 *System (DNS) Database*, 2002. (See <http://www.ietf.org/rfc/rfc3403.txt>)

## 910 11. Appendix A – Glossary (non-normative)

### 911 Auto-ID

912 "Automatic Identification" – a set of technologies used to identify anything, anywhere, automatically.

### 913 Berkeley Internet Name Domain

914 The Berkeley Internet Name Domain is the most widely used DNS implementation on the Internet.

### 915 BIND

916 See Berkeley Internet Name Domain.

### 917 Domain name

918 A hierarchical, 'dot' (.) separated namespace used to identify hosts on the Internet

### 919 DNS

920 See Domain Name System

### 921 Domain Name System



- 922 An infrastructure-level Internet service used to discover information about a domain name. It was  
923 originally developed to map a host name to an IP address, but has since been extended to other uses  
924 (such as ENUM, which maps a phone number to one or more communication services).
- 925 **EPCIS**
- 926 EPC Information Services – A GS1 EPCglobal specification that defines capture and query interfaces  
927 for event and other data.
- 928 **FQDN**
- 929 A fully qualified domain name (FQDN), sometimes also referred as an absolute domain name, is a  
930 domain name that specifies its exact location in the tree hierarchy of the DNS. It specifies all domain  
931 levels, including the top-level domain and the root domain. A fully qualified domain name is  
932 distinguished by its unambiguity; it can only be interpreted one way.
- 933 **Member Organization**
- 934 A Member Organization is a GS1 representative office with local authority over the management of the  
935 GS1 identification space and the promotion of its standards within a specific geographic area, typically  
936 a single country.
- 937 **MO**
- 938 See Member Organization.
- 939 **NAPTR**
- 940 "Naming Authority PoinTeR" -- A DNS record type (35) that contains information about a specific  
941 delegation point within some other namespace using regular expressions.
- 942 **Object Name Service**
- 943 A resolution system, based on DNS, for discovering authoritative data and services related to a GS1  
944 Identification Key.
- 945 **ONS**
- 946 See Object Name Service.
- 947 **Object Class code**
- 948 A code that identifies a particular type of object that is created by a particular manufacturer
- 949 **ONS Peer Root**
- 950 A node within the ONS hierarchy that may be used as the starting point for an ONS query. Every ONS  
951 Peer Root is equivalent to every other in that a query that starts at ONS Peer Root A resolves to the  
952 same results as the same query that starts at ONS Peer Root B.
- 953 **Reader**
- 954 A radio enabled device that communicates with a tag.
- 955 **RFID**
- 956 "Radio Frequency Identification" -- A method of identifying unique items using radio waves. The big  
957 advantage over bar code technology is lasers must see a bar code to read it. Radio waves do not  
958 require line of sight and can pass through materials such as cardboard and plastic.
- 959 **Regular Expression**
- 960 A standard language for pattern matching within a string of characters and for composing new strings  
961 based on matched subcomponents of the original string (i.e. a search and replace function)
- 962 **Serial Number**
- 963 A number that identifies a particular instance of an object class.

- 964            **tag**
- 965            A microchip and antenna combo that is attached to a product. When activated by a tag Reader the tag  
966            emits its EPC plus other data it may have
- 967            **URI**
- 968            "Uniform Resource Identifier" -- the superclass of all identifiers that follow the 'scheme:scheme-  
969            specific-string' convention as specified in RFC 2396 [RFC2396] (e.g., "urn:isbn:2-9700369-0-8" or  
970            "http://example.com/news.html")
- 971            **URL**
- 972            "Uniform Resource Locator" -- a URI that identifies a resource via a representation of its primary  
973            access mechanism (e.g., its network "location"), rather than identifying the resource by name or by  
974            some other attribute(s) of that resource.