Location-to-Service Translation (LoST) Protocol Extensions

Abstract

   An important class of location-based services answers the question,
   "What instances of this service are closest to me?" Examples include
   finding restaurants, gas stations, stores, automated teller machines,
   wireless access points (hot spots), or parking spaces. Currently,
   the Location-to-Service Translation (LoST) protocol only supports
   mapping locations to a single service based on service regions. This
   document describes an extension that allows queries of the type "N
   nearest", "within distance X", and "served by".

Status of This Memo

   This document is not an Internet Standards Track specification; it is
   published for examination, experimental implementation, and
   evaluation.

   This document defines an Experimental Protocol for the Internet
   community. This document is a product of the Internet Engineering
   Task Force (IETF). It represents the consensus of the IETF
   community. It has received public review and has been approved for
   publication by the Internet Engineering Steering Group (IESG). Not
   all documents approved by the IESG are a candidate for any level of
   Internet Standard; see Section 2 of RFC 5741.

   Information about the current status of this document, any errata,
   and how to provide feedback on it may be obtained at
   http://www.rfc-editor.org/info/rfc6451.

Copyright Notice

Table of Contents

1. Introduction

   The Location-to-Service Translation (LoST) protocol [RFC5222] maps
   service identifiers (URNs) and civic or geospatial information to
   service URIs, based on service regions. While motivated by mapping
   locations to the public safety answering point (PSAP) serving that
   location, the protocol has been designed to generalize to other
   location-mapping services.

   However, the current LoST query model assumes that each service URI
   has a service region and that service regions do not overlap. This
   fits the emergency services model, where the service region of a PSAP
   is given by jurisdictional boundaries, but does not work as well for
   other services that do not have clearly defined boundaries. For
   example, any given location is likely served by a number of different
   restaurants, depending on how far the prospective customer is willing
   to travel.

We extend LoST with three additional <findService> query types,
giving the protocol the ability to find the N nearest instances of a
particular service, all services within a given distance, and all
services whose service region includes the user's current location.

2. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in [RFC2119].

3. Service Regions

Generally speaking, service regions apply only to a subset of
services.

In Section 1 of [RFC5222], a service region is defined as follows:

"To minimize round trips and to provide robustness against network
failures, LoST supports caching of individual mappings and indicates
the region for which the same answer would be returned ("service
region")."

Section 5.5 of [RFC5222] further defines a service region:

"A response MAY indicate the region for which the service URL
returned would be the same as in the actual query, the so-called
service region."

For emergency services, service region and service area, as defined
in [RFC5222], represent the same geographical area. This is due to
the fact that each PSAP serves its own area without overlapping with
the service area of any other PSAP. For as long as the client is
located in the service area of a PSAP, the same PSAP is returned by
the LoST server, that is, the service region does not change. A
service region is the service area of a PSAP.

For non-emergency services, different points of service may have
different overlapping service areas. This means that one service
region will probably include a large number of service areas. Since
we can get a large number of service URIs for each query, a service
region per the definition above would be the region within which a
user would get the same set of service URIs. If one or more of the
URIs in the set changes, the set of URIs changes, i.e., the service
region changes. Therefore, for non-emergency services, the service
region defined in [RFC5222] would change frequently, thus greatly
reducing the benefit of caching responses by service region.

Generally speaking, we can divide location-based services into two
main categories based on:

o how far they are from the user (e.g., automatic teller machine,
   food takeout);

o whether or not their service area includes the user's current
   location (e.g., pizza delivery, PSAP).

For services included in the first category, service areas and
therefore service regions are not relevant while they are important
for services included in the second category. This distinction
becomes obvious if we consider, for example, the difference between
takeout (first category) and delivery (second category). In the case
of takeout, the user wants to go to a particular restaurant and buy
dinner, regardless of whether his location falls into the delivery
service area of the restaurant or not. For delivery, the user cares
about the restaurant service area as the restaurant will deliver food
to him only if his location falls within the restaurant service area.

There is a clear distinction between services that require service
areas and services that do not. The LoST extensions defined in this
document take this into account by using the service classification
mentioned above.

4. New <findService> Query Types: "N nearest", "within distance X", and
    "served by"

    We introduce three new types of <findService> queries: "N nearest",
    "within distance X", and "served by". The first query returns the N
    points of interest (POIs) closest to the client's physical location;
    the second query discovers all the points of interest located within
    a given distance from the client's physical location; and the third
    query returns all the points of interest whose service area includes
    the client's current location.

5. LoST Extensions

    For "within distance X" queries, the LoST client needs to specify to
    the server the range within which instances of a particular service
    should be searched. In order to do this, we make use of various
    shapes [RFC5491] in LoST queries.

    For "served by" queries, the LoST client needs to let the server know
    that it MUST return only those services whose service area includes
    the user's current location. In order to do this, we introduce the

<region> element in <findService> queries. Service region boundaries
MAY be returned in a LoST <findServiceResponse> as described in
[RFC5222].

For "N nearest" queries, the LoST client needs to let the server know
N, i.e., the maximum number of service URIs to be returned in a
response. In order to specify this, we introduce the <limit> element
in <findService> queries.

Also, we introduce a new element in LoST responses, namely
<serviceLocation>. This new element is used by the server to
indicate to the client the physical location of points of interest.
In doing so, the client can compute the distance and other metrics
between its current location and the points of interest.

The new elements <region>, <limit>, and <serviceLocation> are defined
in the "lost-ext" namespace. This new namespace is defined in
Section 7.

5.1. New Use of Shapes in Queries

In [RFC5491], different shapes are defined in order to represent a
point and an area of uncertainty within which the user might be
situated. While this remains true for "served by" queries, for
"within distance X" queries, such shapes can be interpreted as the
area within which we want to find a service. In particular, we want
to search for points of service within that area because our location
is within that area with a certain probability. We can think of the
area of uncertainty in a shape as the probability that a user might
be within that area, so we want to look for services within that
area. Thus, the "within distance X" query manually sets the
uncertainty in user location to an uncertainty shape with
parameter X.

For example, Figure 1 shows a "within distance X" <findService>
geodetic query using the circular shape. With the query shown in
Figure 1, we are asking the LoST server to send us a list of service
URIs for pizza places within 200 meters from our approximate position
specified in <gml:pos>.

```
<?xml version="1.0" encoding="UTF-8"?>
<findService
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:ext="urn:ietf:params:xml:ns:lost-ext"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:gs="http://www.opengis.net/pidflo/1.0"
  serviceBoundary="value"
  recursive="true">
  <ext:region>false</ext:region>
  <location id="6020688f1ce1896d" profile="geodetic-2d">
    <gs:Circle srsName="urn:ogc:def:crs:EPSG::4326">
      <gml:pos>37.775 -122.422</gml:pos>
      <gs:radius uom="urn:ogc:def:uom:EPSG::9001">
        200
      </gs:radius>
    </gs:Circle>
  </location>
  <service>urn:service:food.pizza</service>
</findService>
```

Figure 1: A "within distance X" <findService> geodetic query using
the circular shape (a hypothetical service URN of
"urn:service:food.pizza" is used)

Aside from the circular shape, other shapes are also useful. In
particular, there are situations in which it is useful to query for
services in a certain direction of movement rather than in an exact
physical location. For example, if a user is driving north from New
York City to Boston, it would be useful for this user to be able to
query for services north of where he currently is, that is, not at
his current physical location nor at his final destination.

In order to implement such direction-of-travel searches, this
document supports the use of shapes such as an ellipse. The ellipse
has a major and a minor dimension, thus allowing for defining a
"privileged" direction by having the major dimension in the direction
of movement. In the present context, the circular shape allows a
device to search for services in any direction surrounding its
physical location, while shapes such as the ellipse allow the device
to search for services in a more specific direction. Figure 2 shows
a "within distance X" <findService> geodetic query using the
elliptical shape. The ellipse shape is defined in Section 5.2.4 of
[RFC5491].

```
<?xml version="1.0" encoding="UTF-8"?>
<findService
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:ext="urn:ietf:params:xml:ns:lost-ext"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:gs="http://www.opengis.net/pidflo/1.0"
  serviceBoundary="value"
  recursive="true">
  <ext:region>false</ext:region>
  <location id="6020688f1ce1896d" profile="geodetic-2d">
    <gs:Ellipse srsName="urn:ogc:def:crs:EPSG::4326">
      <gml:pos>42.5463 -73.2512</gml:pos>
      <gs:semiMajorAxis uom="urn:ogc:def:uom:EPSG::9001">
        1235
      </gs:semiMajorAxis>
      <gs:semiMinorAxis uom="urn:ogc:def:uom:EPSG::9001">
        660
      </gs:semiMinorAxis>
      <gs:orientation uom="urn:ogc:def:uom:EPSG::9102">
        41.2
      </gs:orientation>
    </gs:Ellipse>
  </location>
  <service>urn:service:food.pizza</service>
</findService>
```

Figure 2: A "within distance X" <findService> geodetic query using
the elliptical shape (a hypothetical service URN of
"urn:service:food.pizza" is used)

5.2. Queries Based on Service Regions

   As mentioned in Section 3, we can divide location-based services into
   two main categories based on:

   o how far they are from the user;

   o whether or not their service area includes the user's current
      location.

   A "within distance X" query addresses services included in the first
   category, while a "served by" query addresses services included in
   the second category.

   When querying LoST regarding a specific service, we need to specify
   if such service belongs to either the first or the second category.
   This is necessary since depending on the category to which the

service belongs, the LoST server has to follow a different metric in
selecting the results to include in the response.

For example, Figure 3 shows three points of interest with their
service areas. The user location (i.e., the LoST client location) is
represented by a circular shape (e.g., GPS). If POI 1, POI 2, and
POI 3 belong to the first category of service ("within distance X"
query), their service area is irrelevant; what matters is how far
they are from the user. For such services, the shape representing
the user location represents the distance within which the user wants
to search for services (see Section 5.1). In the example shown in
Figure 3, the LoST server returns only POI 3, as POI 3 is the only
point of interest falling within the user location represented by the
circle, i.e., the area within which the user wants to search for
services. On the other hand, if the three points of service belong
to the second category ("served by" query), then what matters is
their service area. In this second scenario, since the circle
representing the user location overlaps with all three service areas,
all three POIs can serve the location of the user, and the LoST
server has to return all three POIs, that is, POI 1, POI 2, and
POI 3.
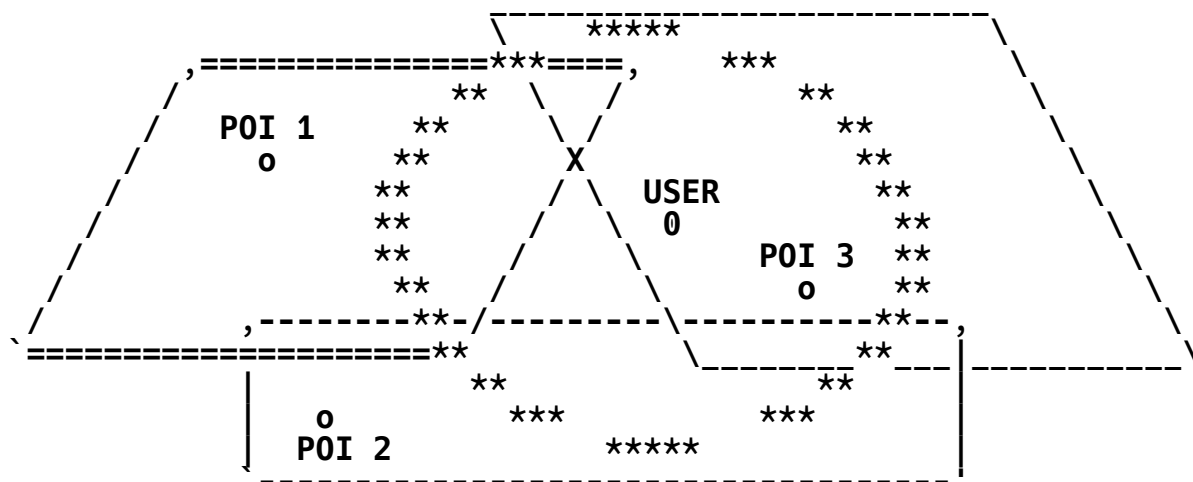
Figure 3: LoST client location (circle) overlapping three service
areas of three different points of interest (POI 1, POI 2, POI 3)

In order for the client to specify which of the two categories the
service belongs to, we introduce the <region> element. This new
element is of type boolean. When its value is false, the LoST server
MUST perform a search based on the distance between the user and the
points of service ("within distance X" query). When its value is
either true or the <region> element is missing (see Section 5.3), the

requested service belongs to the second category, and a search based
on service areas MUST be performed by the LoST server ("served by"
query). When present, the <region> element MUST be conveyed inside
the <findService> element defined in [RFC5222].

For a search based on service regions, the LoST server MUST return
only those services whose service area includes the user's current
location. Service region boundaries MAY be returned in a LoST
<findServiceResponse> as described in [RFC5222].

```
<?xml version="1.0" encoding="UTF-8"?>
 <findService
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:ext="urn:ietf:params:xml:ns:lost-ext"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:gs="http://www.opengis.net/pidflo/1.0"
  serviceBoundary="value" recursive="true">
  <ext:region>true</ext:region>
  <location id="6020688f1ce1896d" profile="geodetic-2d">
    <gs:Circle srsName="urn:ogc:def:crs:EPSG::4326">
      <gml:pos>37.775 -122.422</gml:pos>
      <gs:radius uom="urn:ogc:def:uom:EPSG::9001">
        200
      </gs:radius>
    </gs:Circle>
  </location>
 <service>urn:service:food.pizza</service>
</findService>
```

Figure 4: A "served by" <findService> geodetic query with the new
<region> element (a hypothetical service URN of
"urn:service:food.pizza" is used)

5.3. Difference between "within distance X" and "served by" Queries

Figures 1 and 4 show examples of a "within distance X" query and a
"served by" query, respectively. Although very similar, these two
types of queries have three important differences:

o A "served by" query can support all the shapes a "within distance
   X" query can support plus the point shape. The point shape does
   not make sense for a "within distance X" query and SHOULD NOT be
   used for this query as it would be equivalent to a within-zero-
   meters search.

o In a "within distance X" query, we manually set the uncertainty
   level in user location to X, and we search for services within the
   area represented by such uncertain location. In all other types

of queries, including a "served by" query, the level of
uncertainty in user location cannot be changed by the user, and a
search based on service areas is performed.

o In a "within distance X" query, the value of the <region> element
   MUST be set to false. A "served by" query SHALL have the value of
   the <region> element set to true. If the <region> element is not
   present, its value MUST be assumed to be equal to true, and the
   query will be a "served by" query. This behavior is consistent
   with [RFC5222].

5.4. Limiting the Number of Returned Service URIs

Limiting the number of results is helpful, particularly for mobile
devices with limited bandwidth. For "N nearest" queries, the client
needs to be able to tell the server to return no more than N service
URIs. In order to specify such a limit, we introduce a new element,
namely <limit>. This new element is OPTIONAL, but when present, it
MUST be conveyed inside the <findService> element defined in
[RFC5222].

Figures 5, 6, and 7 show a <findService> geodetic query where the
client asks the server to return no more than 20 service URIs. In
particular, Figure 5 shows an "N nearest" query; Figure 6 shows a
query that is a combination of "N nearest" and "within distance X";
and Figure 7 shows a query that is a combination of "N nearest" and
"served by". When receiving such queries, the LoST server will
return a list of no more than 20 points of interest.

If the available points of interest are more than N, the server has
to identify, among those, the N points of interest closest to the
client's physical location and MUST return those in the response.

When the <limit> element is not present in a <findService> query,
then all available points of interest for the requested type of
service SHOULD be returned by the LoST server. This behavior is
consistent with [RFC5222].

```
<?xml version="1.0" encoding="UTF-8"?>
<findService
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:ext="urn:ietf:params:xml:ns:lost-ext"
  xmlns:gml="http://www.opengis.net/gml"
  serviceBoundary="value" recursive="true">
  <ext:limit>20</ext:limit>
  <location id="6020688f1ce1896d" profile="geodetic-2d">
    <gml:Point id="point1" srsName="urn:ogc:def:crs:EPSG::4326">
      <gml:pos>40.7128 -74.0092</gml:pos>
    </gml:Point>
  </location>
<service>urn:service:food.pizza</service>
</findService>
```

Figure 5: An "N nearest" <findService> geodetic query with the new
<limit> element (a hypothetical service URN of
"urn:service:food.pizza" is used)

```
<?xml version="1.0" encoding="UTF-8"?>
<findService
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:ext="urn:ietf:params:xml:ns:lost-ext"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:gs="http://www.opengis.net/pidflo/1.0"
  serviceBoundary="value"
  recursive="true">
  <ext:region>false</ext:region>
  <ext:limit>20</ext:limit>
  <location id="6020688f1ce1896d" profile="geodetic-2d">
    <gs:Circle srsName="urn:ogc:def:crs:EPSG::4326">
      <gml:pos>37.775 -122.422</gml:pos>
      <gs:radius uom="urn:ogc:def:uom:EPSG::9001">
         200
      </gs:radius>
    </gs:Circle>
  </location>
  <service>urn:service:food.pizza</service>
</findService>
```

Figure 6: A <findService> geodetic query with the new <limit> and
<region> elements. This query is a combination of the "N nearest"
and "within distance X" queries (a hypothetical service URN of
"urn:service:food.pizza" is used)

```
<?xml version="1.0" encoding="UTF-8"?>
<findService
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:ext="urn:ietf:params:xml:ns:lost-ext"
  xmlns:gml="http://www.opengis.net/gml"
  xmlns:gs="http://www.opengis.net/pidflo/1.0"
  serviceBoundary="value"
  recursive="true">
  <ext:region>true</ext:region>
  <ext:limit>20</ext:limit>
  <location id="6020688f1ce1896d" profile="geodetic-2d">
    <gs:Circle srsName="urn:ogc:def:crs:EPSG::4326">
      <gml:pos>37.775 -122.422</gml:pos>
      <gs:radius uom="urn:ogc:def:uom:EPSG::9001">
        100
      </gs:radius>
    </gs:Circle>
  </location>
  <service>urn:service:food.pizza</service>
</findService>
```

Figure 7: A <findService> geodetic query with the new <limit> and
<region> elements. This query is a combination of the "N nearest"
and "served by" queries (a hypothetical service URN of
"urn:service:food.pizza" is used)

5.5. The <serviceLocation> Element in Responses

It is important for the LoST client to know the location of a point
of interest so that distance, route, and other metrics can be
computed. We introduce a new element, namely <serviceLocation>. The
<serviceLocation> element contains the location of a point of
service. When it is used, it MUST be contained in a <mapping>
element. In responses such as <findServiceResponse> [RFC5222], a
list of service URIs, each with its own <serviceLocation> element,
SHOULD be returned. The order of service URIs in the list is not
significant.

The <serviceLocation> element has a single attribute, "profile", that
specifies the profile used. Both civic and geodetic profiles can be
used. The geodetic profiles SHOULD be used in order to compute
distance, route, and other metrics as, at some point, computing such
metrics would require geocoding of the civic address in geodetic
coordinates. Because of this, the position specified in
<serviceLocation> with a geodetic profile SHOULD be represented by
the <Point> element. The <Point> element is described in Section

12.2 of [RFC5222] and in Section 5.2.1 of [RFC5491]. Figure 8 shows
a <findServiceResponse> answer containing two location-to-service-URI
mappings.

[NOTE: The <locationUsed> element cannot be extended for this
purpose, as it is defined outside of the <mapping> element. In
particular, in a response, the <locationUsed> element is always one,
while the number of service URIs is typically more than one.]

There are situations, however, in which it is helpful to include a
civic address together with the geodetic coordinates of a point of
service. Usually, databases already contain the civic address of
points of interest, and for devices with limited capabilities, it is
not always possible to perform decoding of geocoordinates in order to
determine the civic address. Because of this, including the civic
address in a response can be useful. In order to do this, we use a
civic profile for the <serviceLocation> element and specify the POI
civic address in a <civicAddress> element contained in the
<serviceLocation> element. The basic civic location profile is
defined in Section 12.3 of [RFC5222].

Per [RFC5139], it is RECOMMENDED to use multiple <serviceLocation>
elements when multiple forms of service location are available, and
it is RECOMMENDED to provide a geodetic form whenever possible. When
multiple <serviceLocation> elements are present for one POI, all of
them MUST be contained in the same <mapping> element, that is, the
<mapping> element for that POI. Figure 8 shows a
<findServiceResponse> answer with both geodetic and civic locations.

```
<?xml version="1.0" encoding="UTF-8"?>
<findServiceResponse
  xmlns="urn:ietf:params:xml:ns:lost1"
  xmlns:ext="urn:ietf:params:xml:ns:lost-ext"
  xmlns:gml="http://www.opengis.net/gml">
  <mapping
    expires="2007-01-01T01:44:33Z"
    lastUpdated="2006-11-01T01:00:00Z"
    source="authoritative.example"
    sourceId="7e3f40b098c711dbb6060800200c9a66">
    <displayName xml:lang="it">
      Che bella pizza e all' anima da' pizza da Toto'
    </displayName>
    <service>urn:service:food.pizza</service>
    <uri>sip:chebella@example.com</uri>
    <uri>xmpp:chebella@example.com</uri>
    <serviceNumber>2129397040</serviceNumber>
    <ext:serviceLocation profile="geodetic-2d">
      <gml:Point id="point1" srsName="urn:ogc:def:crs:EPSG:4326">
```

```
              <gml:pos>33.665 -112.432</gml:pos>
            </gml:Point>
          </ext:serviceLocation>
          <ext:serviceLocation profile="civic">
            <civicAddress
                xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
              <country>US</country>
              <A1>New York</A1>
              <A3>New York</A3>
              <A6>Broadway</A6>
              <HNO>321</HNO>
              <PC>10027</PC>
            </civicAddress>
          </ext:serviceLocation>
        </mapping>
        <mapping
          expires="2007-01-01T01:44:33Z"
          lastUpdated="2006-11-01T01:00:00Z"
          source="authoritative.example"
          sourceId="7e3f40b098c711dbb6060800200c9b356">
          <displayName xml:lang="en">
            King Mario's Pizza
          </displayName>
          <service>urn:service:food.pizza</service>
          <uri>sip:marios@example.com</uri>
          <uri>xmpp:marios@example.com</uri>
          <serviceNumber>2129397157</serviceNumber>
          <ext:serviceLocation profile="geodetic-2d">
            <gml:Point id="point1" srsName="urn:ogc:def:crs:EPSG:4326">
              <gml:pos>33.683 -112.412</gml:pos>
            </gml:Point>
          </ext:serviceLocation>
          <ext:serviceLocation profile="civic">
            <civicAddress
                xmlns="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr">
              <country>US</country>
              <A1>New York</A1>
              <A3>New York</A3>
              <A6>Amsterdam Avenue</A6>
              <HNO>123</HNO>
              <PC>10027</PC>
            </civicAddress>
          </ext:serviceLocation>
        </mapping>
        <path>
          <via source="resolver.example"/>
          <via source="authoritative.example"/>
        </path>
```

```
        <locationUsed id="6020688f1ce1896d"/>
      </findServiceResponse>
```

   Figure 8: A <findServiceResponse> answer

6. Emergency Services

   The LoST extensions defined in this document SHOULD NOT be used when
   routing emergency sessions, as there may be LoST servers that do not
   support these extensions.

   Figure 9 shows a <findService> query for emergency services as
   defined in [RFC5222]. In such a query, both the <region> element and
   the <limit> element are missing. According to the LoST extensions
   defined in this document, when the <region> element is missing, its
   value defaults to true, and the query is a "served by" query (see
   Section 5.3). When the <limit> element is missing, no limit is
   specified, that is, the LoST server can return any number of results
   (see Section 5.4). This behavior is consistent with [RFC5222] so
   that PSAPs are selected according to their service area, and when a
   user's location overlaps multiple service areas, the LoST server MAY
   return multiple PSAPs.

   The LoST extensions defined in this document are consistent with the
   behavior defined in [RFC5222], and, as such, they do not modify LoST
   behavior for emergency services.

```
   <?xml version="1.0" encoding="UTF-8"?>
      <findService
        xmlns="urn:ietf:params:xml:ns:lost1"
        xmlns:p2="http://www.opengis.net/gml"
        serviceBoundary="value"
        recursive="true">

        <location id="6020688f1ce1896d" profile="geodetic-2d">
          <p2:Point id="point1" srsName="urn:ogc:def:crs:EPSG::4326">
            <p2:pos>37.775 -122.422</p2:pos>
          </p2:Point>
        </location>
        <service>urn:service:sos.police</service>

      </findService>
```

   Figure 9: A <findService> geodetic query for emergency services

   Unlike emergency services, where information such as service
   boundaries of PSAPs and locations of fire stations does not change
   very often, if at all, non-emergency services have information that

   may become inaccurate quickly. Implementers should take this into
   account when designing applications for non-emergency location-based
   services.

7. RELAX NG Schema

   This section provides the RELAX NG schema of LoST extensions in the
   compact form. The verbose form is included in Section 9.

   namespace a = "http://relaxng.org/ns/compatibility/annotations/1.0"
   default namespace ns1 = "urn:ietf:params:xml:ns:lost-ext"

   ##
   ##   Extensions to the Location-to-Service Translation (LoST)
   ##   Protocol

   ##
   ##   LoST Extensions define three new elements: limit, region, and
   ##   serviceLocation.
   ##
   start =
     limit
     | region
     | serviceLocation

   ##
   ##   A limit to the number of returned results.
   ##
   div {
     limit=
       element limit {
         xsd:positiveInteger
       }
   }

   ##
   ##   A boolean variable to request a search
   ##   based on either service areas or distance.
   ##
   ##   NOTE: The W3C XML Schema has two different
   ##   lexical representations for boolean:
   ##   '1' or 'true' vs. '0' or 'false'.
   ##
   div {
     region=
       element region {
         xsd:boolean
       }

```
      }

   ##
   ##      Location Information
   ##
   div {
     locationInformation =
       extensionPoint+,
       attribute profile { xsd:NMTOKEN }?
   }

   ##
   ##      Location Information about the returned point
   ##      of service.
   ##
   div {
     serviceLocation=
       element serviceLocation { locationInformation }+
   }

   ##
   ##      Patterns for inclusion of elements from schemas in
   ##      other namespaces.
   ##
   div {

     ##
     ##      Any element not in the LoST Extensions
     ##      namespace.
     ##
     notLostExt = element * - (ns1:* | ns1:*) { anyElement }

     ##
     ##      A wildcard pattern for including any element
     ##      from any other namespace.
     ##
     anyElement =
       (element * { anyElement }
        | attribute * { text }
        | text)*

     ##
     ##      A point where future extensions
     ##      (elements from other namespaces)
     ##      can be added.
     ##
     extensionPoint = notLostExt*
   }
```

8. Security Considerations

   The overall LoST architecture and framework are defined in [RFC5582].
   All LoST queries for both emergency and non-emergency services, if
   not cached, are sent from the LoST client to a first-hop LoST server.
   In [RFC5582] terminology, a LoST client is called Seeker, and the
   first-hop LoST server is called Resolver (for more rigorous
   definitions, please refer to [RFC5582]). The Resolver will contact
   other LoST servers, and eventually an authoritative LoST server will
   be found. A response will then be sent back to the Seeker.

   When considering both emergency and non-emergency services, there is
   the possibility of the Resolver getting overloaded by non-emergency-
   service queries, thus being unable to process emergency-service
   queries. Such a situation can be addressed in several ways. For
   example, the service provider could dimension the LoST server to
   accommodate anticipated combined traffic loads and then give priority
   to emergency service requests during overload situations, possibly
   with the help of HTTP load balancers.

   The security considerations in [RFC5222] apply. In particular, in
   order to maintain integrity and confidentiality of requests and
   responses, Transport Layer Security (TLS) MUST be implemented and
   SHOULD be used as described in Sections 1, 14, and 18 of [RFC5222].

9. IANA Considerations

9.1. LoST Extensions RELAX NG Schema Registration

   URI: urn:ietf:params:xml:schema:lost-ext

   Registrant Contact: Andrea G. Forte, forte@att.com;
   Henning Schulzrinne, hgs@cs.columbia.edu

   RELAX NG Schema: The RELAX NG schema to be registered is contained in
   Section 7. Its first line is

   default namespace ns1 = "urn:ietf:params:xml:ns:lost-ext"

   and its last line is

   }

9.2. LoST Extensions Namespace Registration

URI: urn:ietf:params:xml:ns:lost-ext

Registrant Contact: Andrea G. Forte, forte@att.com;
Henning Schulzrinne, hgs@cs.columbia.edu

XML:

```
BEGIN
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
   "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="content-type"
        content="text/html;charset=iso-8859-1"/>
  <title>LoST Extensions Namespace</title>
</head>
<body>
  <h1>Namespace for LoST Extensions</h1>
  <h2>urn:ietf:params:xml:ns:lost-ext</h2>
<p>See <a href="http://www.rfc-editor.org/rfc/rfc6451.txt">
  RFC 6451</a>.</p>
</body>
</html>
END
```

10. Non-Normative RELAX NG Schema in XML Syntax

```
<?xml version="1.0" encoding="UTF-8"?>
   <grammar ns="urn:ietf:params:xml:ns:lost-ext"
         xmlns="http://relaxng.org/ns/structure/1.0"
         xmlns:a="http://relaxng.org/ns/compatibility/annotations/1.0"
         datatypeLibrary="http://www.w3.org/2001/XMLSchema-datatypes">

         <start>
     <a:documentation>
       Extensions to the Location-to-Service Translation (LoST)
       Protocol.
       LoST Extensions define three new elements: limit, region and
       serviceLocation.
     </a:documentation>
     <choice>
       <ref name="limit"/>
       <ref name="region"/>
       <ref name="serviceLocation"/>
     </choice>
```

```
               </start>

        <div>
          <a:documentation>
            A limit to the number of returned results.
          </a:documentation>

          <define name="limit">
            <element name="limit">
              <data type="positiveInteger"/>
            </element>
          </define>
        </div>

        <div>
          <a:documentation>
            A boolean variable to request a search
            based on either service areas or distance.
          </a:documentation>

          <define name="region">
            <element name="region">
              <data type="boolean"/>
            </element>
          </define>
        </div>

        <div>
          <a:documentation>
            Location Information
          </a:documentation>

          <define name="locationInformation">
            <oneOrMore>
              <ref name="extensionPoint"/>
            </oneOrMore>
            <optional>
              <attribute name="profile">
                <data type="NMTOKEN"/>
              </attribute>
            </optional>
          </define>
        </div>

        <div>
          <a:documentation>
            Location Information about the returned point of service.
          </a:documentation>
```

```
      <define name="serviceLocation">
        <element name="serviceLocation">
          <ref name="locationInformation"/>
        </element>
      </define>
    </div>

    <div>
      <a:documentation>
        Patterns for inclusion of elements from schemas in
        other namespaces.
      </a:documentation>

      <define name="notLostExt">
        <a:documentation>
          Any element not in the LoST Extensions namespace.
        </a:documentation>
        <element>
          <anyName>
            <except>
              <nsName ns="urn:ietf:params:xml:ns:lost-ext"/>
              <nsName/>
            </except>
          </anyName>
          <ref name="anyElement"/>
        </element>
      </define>

      <define name="anyElement">
        <a:documentation>
          A wildcard pattern for including any element
          from any other namespace.
        </a:documentation>
        <zeroOrMore>
          <choice>
            <element>
              <anyName/>
              <ref name="anyElement"/>
            </element>
            <attribute>
              <anyName/>
            </attribute>
            <text/>
          </choice>
        </zeroOrMore>
      </define>

      <define name="extensionPoint">
```

```
            <a:documentation>
              A point where future extensions
              (elements from other namespaces)
              can be added.
            </a:documentation>
            <zeroOrMore>
              <ref name="notLostExt"/>
            </zeroOrMore>
          </define>
        </div>

    </grammar>
```

11. Acknowledgments

We would like to thank Shida Schubert for reviewing an early version
of this document. We also appreciate the suggestions from members of
the ECRIT working group. In particular, we are grateful to Richard
L. Barnes, Robert Sparks, and Martin Thomson for their overall
feedback and for their comments on how non-emergency services may
affect the provisioning of emergency services lookups.

12. Normative References

[RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
            Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5222]   Hardie, T., Newton, A., Schulzrinne, H., and H.
            Tschofenig, "LoST: A Location-to-Service Translation
            Protocol", RFC 5222, August 2008.

[RFC5139]   Thomson, M. and J. Winterbottom, "Revised Civic Location
            Format for Presence Information Data Format Location
            Object (PIDF-LO)", RFC 5139, February 2008.

[RFC5491]   Winterbottom, J., Thomson, M., and H. Tschofenig, "GEOPRIV
            Presence Information Data Format Location Object (PIDF-LO)
            Usage Clarification, Considerations, and Recommendations",
            RFC 5491, March 2009.

[RFC5582]   Schulzrinne, H., "Location-to-URL Mapping Architecture and
            Framework", RFC 5582, September 2009.

Authors' Addresses

   Andrea G. Forte
   AT&T
   Security Research Center
   33 Thomas Street
   New York, NY 10007
   USA

   EMail: forte@att.com

   Henning Schulzrinne
   Columbia University
   Department of Computer Science
   1214 Amsterdam Avenue, MC 0401
   New York, NY 10027
   USA

   EMail: hgs@cs.columbia.edu