

# 511 Data Exchange including an Open511 Protocol

---

September 24, 2014

Version 1.0



METROPOLITAN  
TRANSPORTATION  
COMMISSION

## Table of Contents

1.0 511 Data Exchange including an Open511 Protocol.....	2
1.1 Why 511 Data Exchange?.....	2
1.2 Contributions.....	2
1.3 What is 511 Data Exchange.....	2
1.4 Locating 511 Data Resources.....	3
2.0 511 Data Exchange Communication Protocol.....	4
2.1 Data Exchange Standards, Delivery Mechanism, and Encoding.....	4
2.2 Open511.....	6
2.3 511 Data Types/Resources.....	6
2.4 API: Discovery.....	7
2.5 API: Jurisdiction.....	10
2.6 API: Jurisdiction Geography.....	12
3.0 Technical Guidelines.....	14
3.1 Data Encoding.....	14
3.2 Authentication and Encryption Mechanisms.....	14
3.3 Response Language.....	14
3.4 API Rate and Throttling.....	15
3.5 Cross domain requests.....	15
3.6 Next steps?.....	15
Appendix A: API Response Messages- XML.....	16
Section I – General XML.....	16
Appendix B: API Response Messages- JSON.....	18
Section I: General JSON.....	18

## List of Tables

A.1.1 Example Discovery Response (XML).....	16
A.1.2 Example Jurisdiction Response (XML).....	17
A.1.3 Example JurisdictionGeography Response (XML) .....	18
B.1.1 Example Discovery Response (JSON).....	18
B.1.2 Example Jurisdiction Response (JSON).....	19
B.1.3 Example Jurisdiction GeographyResponse (JSON) .....	21

## Document History

Description	Version	Date
Working Draft - addressed reorganization comments	0.9	08/28/13
First published version with transit, traffic, tolling, and parking APIs	1.0	09/13/13
Update Traffic APIs' structure information, parameters and filters, and their examples to sync with specification provided on Open511.org.	1.0	5/2/2014
Add GTFS-realtime Trip Updates and Vehicle Positions, and their examples.	1.0	5/7/2014
Minor updates and corrections	1.0	5/28/2014
Add sample request endpoint and parameters and filters tables for Section 3.14 and 3.15. Update references for resource endpoints with their exact URL.	1.0	6/12/2014
Minor updates to Section 3.14 and 3.15	1.0	7/17/2014
Split the API specification document to have a separate General document covering common sections	1.0	09/24/2014

## 1.0 511 Data Exchange including an Open511 Protocol

### 1.1 Why 511 Data Exchange?

511 systems across North America collect and disseminate traveller information to public for free. Government entities – state and/or regional, provide traffic, transit, bicycling, and ridesharing information, depending on the agency’s mandate and jurisdiction, for their respective population through abbreviated phone number, 511, as well as web and other dissemination channels such as mobile web, smartphone apps, SMS, and large-screen display devices and kiosks. 511 systems host a wealth of traveller information that can be a valuable resource for innovative application development by external parties if the data can be exposed through a data exchange standard. In addition to sharing data with developers, adoption of standard based data exchange would also help share data between a 511 system and other data sources, a transit agency for example, as well as neighbouring 511 systems, facilitating traveller information across neighbouring 511 jurisdictions.

Each 511 system has developed its own mechanism to collect data and disseminate information. An open standard for disseminating data would help 511 data consumers easily access data and develop their products based on a set of known interfaces. Open511 is a newly designed open standard that defines a set of data interfaces in order to facilitate access to 511 data. These interfaces are intended to benefit both internal and external traveller application development.

### 1.2 Contributions

MTC and OpenNorth developed the initial draft. A group of stakeholders from government and private sectors have contributed by providing valuable feedback through the Open511 community (<http://open511.org>).

### 1.3 What is 511 Data Exchange

Each 511 system collects data and disseminates information relevant for travellers within a given geographical jurisdiction. The 511 data exchange adopts a set of existing standards for data collection and dissemination. In addition, it also includes a new and open data dissemination interface standard, named Open511, and targeted for data consumers delivering end-user applications. The following principles guided the development of 511 data exchange and the Open511:

- At one end 511 systems collect data from various public (DOTs, cities/counties, transit agencies, etc.) and private (traffic data vendors) sector data generators. At the other end data is consumed by internal and external applications. 511 data exchange specifications should facilitate both data collection and dissemination.
- A number of traffic and transit data exchange standards exist that are well known in the industry. SIRI, NeTEx, GTFS, and GTFS-Realtime for transit data and TMDD for traffic data to name a few. 511 data exchange specifications should adopt a suitable existing standard where it serves the data exchange use case.
- Because some of the existing standards do not lend themselves to wider use due to inherent technical complexity and additional data processing, a simple and open data interface should be supported to facilitate direct consumption.
- 511 data exchange specification should describe:
  - Communication protocol that establishes the relationship between two interacting systems including security, authentication, data access privileges, and system status awareness.

- Protocol for locating 511 data resources; metadata services for geographical coverage (cities, counties), travel modes (traffic, transit, bicycling, ridesharing, walking, parking, air), and transportation infrastructure/service types (highway/arterial/local for traffic, bus/ferry/train for transit).
- Data seeking and delivery mechanisms.
- Data structure and encoding formats.

#### 1.4 Locating 511 Data Resources

511 data disseminators may decide the best way to disseminate data. An online registry for 511 data resources similar to GTFS Data Exchange (<http://www.gtfs-data-exchange.com>) would make 511 data more widely available. The online registry may provide a service that can be queried to get a list of 511 data resources and saved locally in order to support resource discovery. 511 data disseminators should also provide online information about their own open 511 data resources.

## 2.0 511 Data Exchange Communication Protocol

Communication for 511 data exchange incorporates a simple, easy to implement protocol.

- Each data disseminator would decide whether or not a license agreement and/or Terms of Use are required by an external consumer to access data. To make data easily accessible such requirements should be as simple and streamlined as possible to encourage wider use of the data.
- If a registration is required to access data, that process may be used to identify data needs by the registrant. Successful registration should result in an authentication key/ID provided to the consumer.
- Data requests should be accompanied with a valid key/ID and would be limited to the granted privileges.
- Communication protocol for 511 data collection would be set up as per the mutual agreement between the two organizations.

## 2.1 Data Exchange Standards, Delivery Mechanism, and Encoding

511 data exchange involves both bulk and ad-hoc query-based data sharing. Bulk data exchange would provide large volume of data, useful for C2C (center to center) communication whereas app developers are mostly interested in ad-hoc, on demand data in small quantities. Data users who prefer to store data in their own system would choose bulk data transfer. On the other hand, a mobile application would need small data packaged in readily consumable format that the query based APIs would fulfil.

511 data exchange specifications take advantage of existing standards, when feasible. Existing standards that are adopted as 511 standards are travel mode specific. Table 1 and 2 below provide adopted 511 data exchange standards, delivery mechanism, and encoding for transit and traffic data. Travel modes other than traffic and transit will be added to this standard later.

In discussions with agencies and developer community it became obvious that one particular data and communication standard cannot fulfil everyone's needs. For example, agencies producing traffic data prefer standards such as TMDD for data sharing with other agencies. On the other hand, a developer working on a mobile application for traffic information is more comfortable using widespread REST based APIs and JSON/XML data format. Though standards like TMDD are very comprehensive and accepted in the ITS industry they are often seen as heavy and cumbersome by the developer community who are primarily interested in building a simple web/mobile based application. Therefore it was necessary to address this diversity in use cases by adopting TMDD for low level (bulk) data communication between agencies and local data storage and a REST based Open511 API providing data in both XML and JSON formats. 511 data to be collected and disseminated can be broadly classified into two categories:

**Configuration data** – This category includes data that are mostly static and defines the configuration of a transportation system. For example, routes and stops for a transit service or the roadway network for traffic do not change frequently. Configuration data is very important because of the fact that real-time information is based on the underlying configuration data. Most users would like to download, save, and perform additional processing on configuration data before it is consumed in smaller pieces by their system and applications. Because configuration dataset for a transportation system/service would be large in size, it is efficient to provide this bulk data through transport mechanism such as FTP or data downloadable over HTTP. When it is necessary to provide configuration data in small quantities it will be done through Open511 APIs over Request/Response based communication.

**Real-time data**—Data in this category generally has a short life time and changes frequently. Data such as transit departure predictions, and incidents and travel times on roadway fall in this category. Whether used in bulk or through Open511 APIs real-time data should be provided through a Request/Response based communication.

Based on the amount of data exchanged and data encoding used, 511 data exchanges have been organized into two groups, bulk and query-based. Bulk data exchanges provide data using data encoding adopted by the given standard. Encoding for query-based Open511 APIs can be JSON, XML, or both. Depending on the use case some users may choose bulk exchanges using standards such as TMDD for both configuration and real-time, some may use only Open511 APIs for configuration and real-time data, others may first download configuration data to provide the foundation for their application and make ad-hoc queries through the Open511 API calls for real-time data. Table 1 and 2 below provide a summary of bulk and query-based data exchanges, respectively.

**Table 1 511 Adopted Standards, Delivery Mechanisms and Encodings for Bulk Exchanges**

Travel Mode	Data Type	Data Definition <sup>1</sup>	Pattern of Interaction <sup>2</sup>	Data Transport	Data Encoding
Transit	Service configuration and scheduled timetable	GTFS, NeTex <sup>3</sup>	Downloaded	HTTP	GTFS: CSV NeTex: XML
	Predicted and unplanned changes data	SIRI, GTFS-RT	Request/Response	HTTP	SIRI: XML GTFS-RT: Proto Buffer
Traffic	Network configuration data	TMDD V3.0	Request/Response	HTTP	XML
	Real-time speed and incidents data	TMDD V3.0	Request/Response	HTTP	XML

**Table 2 Open 511 Data Standards, Delivery Mechanism and Encoding for Query-Based Exchanges**

Travel Mode	Data Type	Data Definition <sup>1</sup>	Pattern of Interaction	Data Transport	Data Encoding
Transit	Service configuration and scheduled timetable	NeTex	Request/Response	HTTP	JSON/XML
	Predicted and unplanned changes	SIRI	Request/Response	HTTP	JSON/XML
Traffic	Road network configuration	Open511	Request/Response	HTTP	JSON/XML

	Real-time speed and incidents	Open511	Request/Response	HTTP	JSON/XML
Parking	Parking configuration and pricing information	Custom <sup>4</sup>	Request/Response	HTTP	JSON/XML
Tolling	Configuration, and static and dynamic pricing	Custom <sup>4</sup>	Request/Response	HTTP	JSON/XML

1. GTFS – General Transit Feed Specification, GTFS-RT – GTFS for real-time transit information, SIRI – Service Interface for Real-Time Information, TMDD – Traffic Management Data Dictionary.
2. All bulk data exchange interactions will be preceded by new data availability notification to registered data subscribers.
3. Each 511 system may adopt a suitable custom data exchange standard that facilitates transit data collection from transit agencies within its jurisdiction. For example, San Francisco Bay Area 511 has developed an XML schema that is currently being used by Bay Area transit agencies to provide data to the 511 system. The custom XML schema may get replaced in future by an enhanced GTFS.
4. May get adopted in the Open511 in future.

## 2.2 Open511

Open511 aims to create simple, uniform and resource driven APIs that can be easily used by consumers to retrieve data from 511 systems. In order for these APIs to be easily discovered Open511 API providers must create a single entry discovery point that provides links and details for all other resources available from a disseminator/Jurisdiction. Open511 defines a discovery resource that will help consumers locate APIs and other resources. Also to provide a context of the data and other details, Open511 recommends defining a Jurisdiction resource which will provide details on the jurisdiction and its coverage.

## 2.3 511 Data Types/Resources

As shown in the tables above, 511 data exchange adopts existing data standards such as GTFS and TMDD, where those standards are most suitable. Information on adopted standards can be obtained from online resources as listed below.

### **Transit (service configuration and schedules)**

GTFS: <https://developers.google.com/transit/gtfs/>

NeTEx: <http://www.kizoom.com/standards/netex/>

### **Transit (real-time information)**

GTFS-RT: <https://developers.google.com/transit/gtfs-realtime/>

SIRI: <http://www.kizoom.com/standards/siri/>

### **Traffic**

TMDD: <http://www.ite.org/standards/tmdd/>

Open511: <http://open511.org/>

Since Open511 specification is fairly new and evolving, please refer to the above website for latest updates.

In addition, data types for Open511 APIs are defined below.

## 2.4 API: Discovery

Open511 recommends providing an API discovery mechanism/endpoint similar to open311. Jurisdictions/disseminators will setup a service directory that will list all available resources that are currently supported by a jurisdiction and their endpoints. The service directory will also provide information on endpoints for multiple versions.

*The structure of an API discovery document consists of*

Field	Type	Mandatory / Optional	Description
<b>jurisdictions</b>	Collection of <b>jurisdiction</b>	<i>Mandatory</i>	List all the jurisdictions that are supported by the endpoint. Most of the time, there will be only one occurrence, except for multiple jurisdictions endpoints and aggregators.  <i>At least one jurisdiction element is mandatory.</i>
— <b>id</b>	String	<i>Mandatory</i>	ID of the jurisdiction.
— <b>name</b>	Free text	<i>Mandatory</i>	Full text name of the jurisdiction.
— <b>self/url</b>	Link	<i>Mandatory</i>	Link to the jurisdiction resource. In XML, the <i>rel</i> attribute needs the value <i>self</i> . An aggregator should always point the original jurisdiction resource and not copy it locally.
<b>services</b>	Collection of <b>service</b>	<i>Mandatory</i>	List all the services supported by the endpoint.
— <b>service_type</b>	Link	<i>Mandatory</i>	A URL representing an Open511 service like Events or Areas. The URL follows the same logic as an XML namespace: it's a unique identifier for the service type, though clients will never need to visit that URL. It's recommended that the URL point to human-readable documentation for the service.  All services that are part of this specification will have URLs starting with <code>http://open511.org/services/</code> . Other organizations may want to build services that are outside the current scope of this specification but otherwise fit in with the other Open511 services; such services will have URLs not on open511.org.

			Such services must follow the guidelines of this specification.  Values for current Open511 services are:  http://open511.org/services/events/ http://open511.org/services/areas/ http://open511.org/services/cameras/ http://open511.org/services/roads/ http://open511.org/services/traffic_segments/
— self/url	Link	<i>Mandatory</i>	Link pointing to the resource. In XML, the <i>rel</i> attribute needs the value <i>self</i> . Even if the current endpoint supports multiple jurisdictions or is an aggregator, there is only one service resource that aggregates the data for all the jurisdictions.
— supported_versions	Collection of supported_versions	<i>Optional</i>	List all the versions supported by the current server.
— supported_version	Enum	<i>Optional</i>	Version identifier. As the specification evolves, version identifiers will be added. For the moment, the only version supported is v0 and is not an official value.

Sample request endpoint for discovery

<b>Request Type</b>	GET
<b>Request Endpoint Example</b>	http://api.511.org

Parameters and Filters

The discovery resource does not support any URL parameter or filter besides the format and the version negotiation. The discovery response for XML is shown in Appendix A Section A.I.I. The discovery response for JSON is shown in Appendix B Section B.I.I.

Parameter	Mandatory/Optional	Description
<b>format</b>	<i>Optional</i>	The response format (json/xml) desired. If none specified, then default response would be JSON.  e.g.  ?format=json (returns json response for v1, if v1 is the latest version or specified version via parameter)  ?format=xml

		(returns XML response for v1)
<b>version</b>	<i>Optional</i>	The version of Open511 desired. e.g  ?version=v1 (returns response for v1 in conjunction with format requested.)
<b>api_key</b>	<i>mandatory</i>	Unique key assigned to a user after they signup for Open511.

Possible Errors

The numbers represent the HTTP status code returned for each error type:

- 401 – Unauthorized (Invalid API key)
- 500 - Internal Server Error (System has issues processing your request)

## 2.5 API: Jurisdiction

The jurisdiction represents any government entity (city, state/province, agency, etc.) that publishes Open511 data. It is a functional concept used to provide metadata such as contact information, etc.

*The structure of a Jurisdiction resource consists of*

Field	Type	Mandatory / Optional	Description
self	Link	Mandatory	Self link to the current resource.
id	String	<i>Mandatory</i>	Unique identifier for the jurisdiction. It's expected to be unique across all Open511 implementations. It must take the form of a hostname within a domain owned by the government entity. The hostname doesn't necessarily need to resolve in DNS; for example, if Roadsville owns <i>roadsville.gov</i> , it's fine to create <i>agency1.roadsville.gov</i> and <i>agency2.roadsville.gov</i> jurisdictions, even if those don't resolve to IP addresses.
name	Free text	<i>Mandatory</i>	Name of the jurisdiction.
email	String	<i>Mandatory</i>	Valid email that can be used to contact the department or the person in charge of the data provided by the Open511 API.
description	Link	<i>Optional</i>	Link pointing to a human readable resource (web page, pdf file, etc) providing details about the Open511 service and the jurisdiction covered.
phone	String	<i>Optional</i>	Phone number to contact the department or person in charge of the data provided by the Open511 API.
description	Free text	<i>Optional</i>	Free text that can be used by a client application to provide some information about the Open511 service and the jurisdiction covered.
geography	Link	<i>Mandatory</i>	The geography field links an open511 jurisdiction geography resource providing the jurisdiction boundaries.
timezone	timezone	<i>Optional</i>	The timezone of the jurisdiction; will be used as the default for events belonging to this jurisdiction. Should always be provided, except in cases where a jurisdiction spans multiple timezones.
distance_unit	Enum	<i>Optional</i>	The unit this jurisdiction uses to measure distances. Can be KILOMETRES or MILES, but kilometres are the default and so jurisdictions using kilometres need not specify a <i>distance_unit</i> .
license	Link	<i>Mandatory</i>	Link to a resource containing the license covering the data provided in the API.
languages	Collection of language	<i>Optional</i>	List of languages supported by this endpoint. A language element should be provided for each language supported.
— language	Enum	<i>Mandatory</i>	Multiple occurrences supported. The values supported are the same as the language negotiation feature.

*Sample request endpoint for jurisdiction*

<b>Request Type</b>	GET
<b>Request Endpoint Example</b>	For e.g. http://api.511.org/Jurisdictions/511.org

Parameters and Filters

The jurisdiction resource does not support any URL parameter or filter besides the format selection and the language negotiation. The jurisdiction response for XML is shown in Appendix A Section A.1.2. The discovery response for JSON is shown in Appendix B Section B.1.2.

Parameter	Mandatory/ Optional	Description
<b>format</b>	<i>Optional</i>	The response format (json/xml) desired. If none specified, then default response would be JSON.  e.g.  ?format=json (returns json response for v1, if v1 is the latest version or specified via version parameter)  ?format=xml (returns XML response for v1)
<b>version</b>	<i>Optional</i>	The version of Open511 desired. e.g.  ?version=v1 (returns response for v1 in conjunction with format requested.)
<b>api_key</b>	<i>mandatory</i>	Unique key assigned to a user after they signup for Open511.

Possible Errors

The numbers represent the HTTP status code returned for each error type:

- 401 – Unauthorized (Invalid API key)
- 500 - Internal Server Error (System has issues processing your request)

## 2.6 API: Jurisdiction Geography

The jurisdiction geography represents the boundaries of the jurisdiction.

*The structure of a Jurisdiction resource consists of*

Field	Type	Mandatory / Optional	Description
geography	GeoSpatial	Mandatory	Boundaries of the jurisdiction; a Polygon or MultiPolygon.

*Sample request endpoint for jurisdiction*

Request Type	GET
Request Endpoint Example	For e.g. <a href="http://api.511.org/Jurisdictions/511.org/geography">http://api.511.org/Jurisdictions/511.org/geography</a>

Parameters and Filters

The jurisdiction geography resource does not support any URL parameter or filter besides the format selection. The jurisdiction response for XML is shown in Appendix A Section A.1.2. The discovery response for JSON is shown in Appendix B Section B.1.2.

Parameter	Mandatory/ Optional	Description
<b>format</b>	<i>Optional</i>	The response format (json/xml) desired. If none specified, then default response would be JSON.  e.g.  ?format=json (returns json response for v1, if v1 is the latest version or specified via version parameter)  ?format=xml (returns XML response for v1)
<b>version</b>	<i>Optional</i>	The version of Open511 desired. e.g  ?version=v1 (returns response for v1 in conjunction with format requested.)
<b>api_key</b>	<i>mandatory</i>	Unique key assigned to a user after they signup for Open511.

Possible Errors

The numbers represent the HTTP status code returned for each error type:

- 401 – Unauthorized (Invalid API key)
- 500 - Internal Server Error (System has issues processing your request)

## 3.0 Technical Guidelines

### 3.1 Data Encoding

Character encoding is in UTF-8 for all the APIs developed using SIRI, Natex and Open511.

APIs developed under Open511 follow additional guidelines proposed by Open511 specifications. Please refer to <http://open511.org/guidelines.html> for details.

#### 3.1.1 Response Serialization: XML and JSON

Both JSON and XML serialization are supported, with JSON being default serialization format.

XML data is serialized following XSD provided by the specification.

Please refer to <http://www.kizoom.com/standards/siri/> for details regarding data serialization for SIRI based APIs.

Please refer to <http://open511.org/guidelines.html> for details regarding data serialization for Open511 based APIs.

#### 3.1.2 Format and version negotiation

Format and version negotiation is done via HTTP headers and/or a queystring parameter (See <http://open511.org/guidelines.html> )

Users can indicate their format preference via one of two mechanisms:

- By using the Accept HTTP header and specifying application/json and application/xml as types.
- By specifying `?format=json` and `?format=xml` in URL parameter. If provided, it takes precedence over the Accept header.

Users may request responses conforming to a specific Open511 version via one of two mechanisms (Only applies to Open511 based APIs):

- By specifying an Open511-Version header, with the desired version string as a value, e.g. Open511-Version: v1
- By specifying `?version=v1` in URL parametre. If provided, it takes precedence over the Open511-Version header.

## 3.2 Authentication and Encryption Mechanisms

For accessing these APIs, users must use API keys (tokens). That token must be sent via an `api_key` queystring parameter. Users can signup for new token at `http://{token_URL}`

### 3.3 Response Language

Even though Open511 is capable to support multiple languages; the current implementation for Open511 based APIs is unilingual. Each API response includes a lang attribute set to "en".

### 3.4 API Rate and Throttling

In order to avoid excessive load and usage on system, access to APIs has been restricted based on usage. Each API will have throttle limit defined and when throttle limits are exceeded by a consumer, appropriate messages and response code will be provided for subsequent requests. For e.g. if the Traffic Segment API has a request limit of 60 requests/hour per api\_key and if a consumer requests the API more than 60 times in a hour, the API response would comeback with 429 - Too Many Requests.

### 3.5 Cross domain requests

All the above listed APIs support cross-domain requests, via both:

- JSONP, using a parameter named callback. Example: ...?callback=func\_name
- CORS, by including an Access-Control-Allow-Origin: \* header in all responses to GET queries

### 3.6 Next steps?

Open511 specifications may add additional travel modes such as parking, tolling, ridesharing, bicycling, etc. and additional data services for modes already in the specifications. Modification and enhancement to this specification will follow a systematic versioning scheme.

## Appendix A: API Response Messages- XML

### Section 1 – General XML

#### A.1.1 Example Discovery Response (XML)

```

<open511 xml:lang="en" xml:base="http://api.511.org/info/" version="v1">
  <jurisdictions>
    <jurisdiction>
      <id>511.org</id>
      <name>Bay Area 511</name>
      <link rel="self" href="http://api.511.org/jurisdictions/511.org/">
    </jurisdiction>
  </jurisdictions>
  <services>
    <service>
      <link rel="self" href="/traffic/areas"/>
      <link rel="service_type" href="http://api.511.org/services/areas"/>
      <supported_versions>
        <supported_version>v1</supported_version>
      </supported_versions>
    </service>
    <service>
      <link rel="self" href="/traffic/events"/>
      <link rel="service_type" href="http://api.511.org/services/events"/>
      <supported_versions>
        <supported_version>v1</supported_version>
      </supported_versions>
    </service>
    <service>
      <link rel="self" href="/traffic/roads"/>
      <link rel="service_type" href="http://api.511.org/services/roads"/>
      <supported_versions>
        <supported_version>v1</supported_version>
      </supported_versions>
    </service>
    <service>
      <link rel="self" href="/traffic/traffic_segments"/>
      <link rel="service_type" href="http://api.511.org/services/traffic_segments"/>
      <supported_versions>
        <supported_version>v1</supported_version>
      </supported_versions>
    </service>
    <service>
      <link rel="self" href="/transit/stopmonitoring"/>
      <link rel="service_type"
href="http://user47094.vs.easily.co.uk/siri/schema/1.4/siri.xsd"/>
      <supported_versions>
        <supported_version>v1</supported_version>
      </supported_versions>
    </service>
    <service>
      <link rel="self" href="/transit/vehiclemonitoring"/>
      <link rel="service_type"
href="http://user47094.vs.easily.co.uk/siri/schema/1.4/siri.xsd"/>
      <supported_versions>
        <supported_version>v1</supported_version>
      </supported_versions>
    </service>
  </services>
</open511>

```

```

<service>
  <link rel="self" href="/transit/tripupdates"/>
  <link rel="service_type" href="https://developers.google.com/transit/gtfs-
realtime/trip-updates"/>
  <supported_versions>
    <supported_version>v1</supported_version>
  </supported_versions>
</service>
<service>
  <link rel="self" href="/transit/vehiclepositions"/>
  <link rel="service_type" href="https://developers.google.com/transit/gtfs-
realtime/vehicle-positions"/>
  <supported_versions>
    <supported_version>v1</supported_version>
  </supported_versions>
</service>
</services>
<link rel="self" href="/?api_key={api_key}"/>
</open511>

```

### A.1.2 Example Jurisdiction Response (XML)

```

<open511 xml:lang="en" xml:base="http://api.511.org/info/" version="v1">
  <jurisdiction>
    <link rel="self" href="http://api.511.org/jurisdictions/511.org/">
    <id>511.org</id>
    <name>Bay Area 511</name>
    <email>someone@mtc.ca.gov</email>
    <phone>+1 555-555-5555</phone>
    <description>Official traffic data (construction) from SF Bay Area</description>
    <link rel="description" href="/presentation.html"/>
    <timezone>America/Los_Angeles</timezone>
    <link rel="geography" href="http://api.511.org/jurisdictions/511.org/geography"/>
    <link rel="license" href="http://511.org/opendatalicense/TBD"/>
    <languages>
      <language>en</language>
    </languages>
  </jurisdiction>
  <link rel="up" href="/">
  <link rel="self" href="/jurisdictions/511.org/?api_key={api_key}"/>
</open511>

```

### A.1.3 Example JurisdictionGeography Response (XML)

```
<open511
  xmlns:gml="http://www.opengis.net/gml"
  xml:lang="en"
  xml:base="http://api.511.org"
  version="v1"
>
  <geographies>
    <geography>
      <gml:Polygon srsName="EPSG:4326">
        <gml:outerBoundaryIs>
          <gml:LinearRing>
            <gml:coordinates>-73.515580304999958,45.743558682000106 -
73.516801182999984,45.744177048000076 -73.516754262999996,45.745199120000088 -
73.516741493999973,45.745477517000076 -73.51918957800001,45.746719115000111 -
73.519343021999987,45.748003618000077 -73.520130775999974,45.748595939000076 -
73.520120826999971,45.749015964000108 -73.52193461600001,45.750027519000106 -
73.529479503999994,45.754234632000113 -73.527583232999973,45.777799543000107 -
73.515580304999958,45.743558682000106</gml:coordinates>
          </gml:LinearRing>
        </gml:outerBoundaryIs>
      </gml:Polygon>
    </geography>
  </geographies>
  <link rel="self" href="/jurisdictions/SFBayArea/geography/?apikey={api_key}" />
  <link rel="up" href="/jurisdictions/SFBayArea/" />
</open511>
```

## Appendix B: API Response Messages- JSON

### Section 1: General JSON

#### B.1.1 Example Discovery Response (JSON)

```
{
  "jurisdictions": [
    {
      "id": "511.org",
      "name": "Bay Area 511",
      "url": "http://api.511.org/jurisdictions/511.org/"
    }
  ],
  "services": [
    {
      "service_type_url": "http://api.511.org/services/areas",
      "supported_versions": [
        "v1"
      ],
      "url": "/traffic/areas"
    },
    {
      "service_type_url": "http://api.511.org/services/events",
      "supported_versions": [
        "v1"
      ],
      "url": "/traffic/events"
    }
  ],
}
```

```

{
  "service_type_url": "http://api.511.org/services/roads",
  "supported_versions": [
    "v1"
  ],
  "url": "/traffic/roads"
},
{
  "service_type_url": "http://api.511.org/services/traffic_segments",
  "supported_versions": [
    "v1"
  ],
  "url": "/traffic/traffic_segments"
},
{
  "service_type_url": "http://user47094.vs.easily.co.uk/siri/schema/1.4/siri.xsd",
  "supported_versions": [
    "v1"
  ],
  "url": "/transit/stopmonitoring"
},
{
  "service_type_url": "http://user47094.vs.easily.co.uk/siri/schema/1.4/siri.xsd",
  "supported_versions": [
    "v1"
  ],
  "url": "/transit/vehiclemonitoring"
},
{
  "service_type_url": "https://developers.google.com/transit/gtfs-realtime/trip-
updates",
  "supported_versions": [
    "v1"
  ],
  "url": "/transit/tripupdates"
},
{
  "service_type_url": "https://developers.google.com/transit/gtfs-realtime/vehicle-
positions",
  "supported_versions": [
    "v1"
  ],
  "url": "/transit/vehiclepositions"
}
],
"meta": {
  "version": "v1",
  "url": "/?api_key={api_key}"
}
}

```

### B.1.2 Example Jurisdiction Response (JSON)

```

{
  "id": "511.org",
  "url": "http://api.511.org/jurisdictions/511.org/",
  "name": "Bay Area 511",
  "email": "someone@mtc.ca.gov",
  "phone": "+1 555-555-5555",

```

```
"description": "Official traffic data (construction) from SF Bay Area",  
"description_url": "/presentation.html",  
"timezone": "America/Los_Angeles",  
"geography_url": "http://api.511.org/jurisdictions/511.org/geography",  
"license_url": "http://511.org/opendatalicense/TBD",  
"languages": [  
  "en"  
],  
"meta": {  
  "version": "v1",  
  "url": "/jurisdictions/511.org/?api_key={api_key}",  
  "up_url": "/"  
}  
}
```

### B.1.3 Example Jurisdiction GeographyResponse (JSON)

```
{
  "geographies": [
    {
      "type": "Polygon",
      "coordinates": [
        [ [71.17,47.33], [-71.15,47.36], [-71.10,47.35],
          [-71.20,47.40], [-71.17,47.33] ]
      ]
    }
  ],
  "meta": {
    "version": "v1",
    "url": "/jurisdictions/SFBayArea/geography/?apikey={api_key}",
    "up_url": "/jurisdictions/SFBayArea/"
  }
}
```