511 Regional Real-Time Transit System

Extensible Markup Language (XML) Document Type Definitions (DTDs) for Java Message Service (JMS) Implementation

v. 9.7

January 27, 2014

Introduction

This document contains the Extensible Markup Format (XML) document type definitions (DTDs) for the messages exchanged between the transit agency Java Message Service (JMS) client and the 511 system's JMS server. These definitions were developed upon research conducted by the Metropolitan Transportation Commission (MTC) and the Telvent Farradyne Corporation.

All transit agencies will receive a JMS client that will respond to requests from the 511 JMS server for transit arrived-status data. The transit agency JMS client will also accept a subscription from the 511 JMS server and will publish arrival/departure predictions based on those subscriptions. The 511 real-time system will access all real-time data from the central 511 JMS server.

Transit agencies will publish prediction data updates to the central JMS server via the JMS client for consumption by all prediction data subscribers. If transit agencies are also interested in receiving the real-time information of partner agencies, they could also subscribe to the prediction data sent from the JMS server. In addition, 511 will request configuration data and arrived status data from each agency once per day. When an agency receives either of these requests, it must respond with the requested configuration or arrived status messages.

All messages will be sent using ZIP compression. The first four (4) bytes of the compressed data should contain the message size of the uncompressed data in network (i.e. "big-endian") byte order.

In addition to the XML Messages sent via JMS Agencies will also need to decide if they will send configuration and prediction data by route and direction **or** by route and pattern. A pattern is typically a subset of stops on a route and direction. If an agency decides to send route and pattern; they must supply a Pattern-to-Direction Table. The definition of the table is also included in this document

Transit Agency to Regional Real-Time Transit Interface Summary

This section contains a summary of the activities to be performed by the Transit Agencies as well as those being performed by MTC and its contractors
For this project, transit agencies are required to connect to MTC's JMS Server, publish real-time prediction data, and accept and respond to requests for transit inventories and Arrival Status
In order to connect to the regional real-time transit system, four things are needed:

a) A Transit Interface application (supplied by MTC) that publishes prediction data and responds to requests for transit route and stop inventory and Arrival Status data

b) The Transit Interface application needs access to the transit agency's real-time prediction data, so it can collect and publish it

c) The Transit Interface application requires access to transit route and stop inventories and arrival status data, so it can respond to MTC requests for this data

d) Access to the internet so the Transit Interface application can connect to the MTC JMS Data Server.

MTC is supplying the JMS Server so transit agencies do not need to configure or maintain a JMS server. MTC has supplied a template version of the Transit Interface application that performs the following functions (The template includes a file with sample prediction data as well as hard-coded data as examples.):

- Logs in to the JMS Server
- Publishes prediction data
- Responds to requests for Transit Route and Stop Inventory
- Responds to requests for Arrival Status

The transit agency's responsibility is to modify the Transit Interface application by:

- Replacing the prediction data from the provided file with the template with live data from the transit agency's prediction system
- Changing out the hard-coded data in the template section that responds to transit route and stop inventory with current data from the transit agency's internal systems
- Replacing the hard-coded data in the template section that responds to arrival status requests with current arrival status data from the transit agency's internal systems

Once the application is tested and installed, no additional changes are expected.

XML Data Fields

All date/time fields in the request and response messages below are formatted as follows:

YYYY-MM-DDTHH:MM:SSZ
where Z = (+/-)HH:MM

The year month and day are specified, followed by a “T” (for Time) and then the hours, minutues, and seconds, followed by a time zone indicator. Note that the time in hours is specified on a 24 hour clock – am and pm cannot be used. For the timezone “Z”, indicate the difference from UTC in the format: (+/-)HH:MM.

Example 1) Jan 3, 2014 at 4:15pm Pacific Standard Time would be encoded as follows:
2014-01-03T16:15:00-08:00

Example 2) Jan 6, 2014 at 11:05am in Paris, France which is in Central European Time:
2014-01-06T11:05:00+01:00

Configuration Request Message

To obtain updated configuration information from the agencies, 511 issues a configuration request message to the transit agency's JMS client via the JMS server. There is no necessary data within the request message. The format of the message that the transit agency will receive is documented in the XML DTD shown below:

The format of the message is documented in the XML DTD shown below:

Configuration Request:

```
<!-- Request sent for on-demand updates for agency
     configuration information. -->
<!DOCTYPE SystemInformationRequest [
<!ELEMENT SystemInformationRequest EMPTY >
<!ATTLIST SystemInformationRequest
      agency        CDATA #REQUIRED  #Format: Comma separated list>
]>
```

Configuration Response Message

Upon issuance of a configuration request message, a configuration response message will be sent. The configuration response should always include the full and entire schedule, not just a subset of the data. A response that includes only the current day schedule or some other subset of the data would not be sufficient.

The response message will contain either a list of errors or the requested configuration data and optionally, a record containing a mapping between trip pattern IDs and direction codes if necessary.

Errors

If present, an error record will contain the following information (data element names are shown in parenthesis):

- The integer error code if applicable (**`errorCode`**)
- The text of the error (**`errorText`**)

Trip Patterns (optional)

Route directions can be specified by either explicitly indicating a direction codes or by specifying a trip pattern which implies a direction. When using the former, trip patterns are not needed and should not be included in the response message. The chosen method used for specifying directions must be used exclusively throughout the configuration; the methods cannot be mixed.

Each trip pattern mapping data record must contain the following:

- The name of the agency (**`agency`**)
- The version number of the mapping data (used to determine when changes have occurred since the last update) (**`version`**)
- The date/time of the current snapshot of data (**`TimeStamp`**)
- The list of trip pattern ID mappings for the agency (**`TripMapping`**)

Each trip pattern mapping record contains the following (some fields are optional):

- The integer identifier that distinguishes the trip pattern along the route (**`tripID`**)
- An identifier or alias that uniquely distinguishes each route from all others (**`routeKey`**)
- An identifier or alias that uniquely distinguishes each direction from all others – for example, inbound, outbound, eastbound, southbound, clockwise, etc. (**`dirKey`**)
- The publicly understandable general destination name of the direction – for example ‘downtown’ or ‘Hillsdale shopping center’ (**`dirTitle`**). This field should be as short as possible.
- An optional flag (true/false) indicating if route/direction combination represented by the trip pattern ID contains all stops (**`defaultTripForRouteDir`**)

Routes, Directions and Stops

Each configuration data record contains the following (some fields are optional):

- The name of the agency (**`agency`**)

- The version number of the configuration data (used to determine when changes have occurred since the last configuration update) (**`version`**)
- The date/time of the current snapshot of data (**`TimeStamp`**)
- The total number of stops (**`numStops`**)
- An optional description of the data – e.g. description of data quality (**`msg`**)
- The list of bus routes for the agency (**`Route`**)

Each agency may have one or more routes. Each route record may contain the following information (data element names are shown in parenthesis):

- An identifier or alias that uniquely distinguishes each route from all others (**`key`**). This should be the publicly known route number/code.
- The name of the route (**`title`**). This should be a short name that might be in use as public information along with route number/code. If such name is not applicable same value used in the **key** field should be used in this field.
- The list of directions of the route used in the **dirKey** field in trip mapping record above or the **key** field in the direction/destination record below (**`Direction`**)

Each route may have one or more directions/destinations. Each direction/destination record contains the following information (data element names are shown in parenthesis - some fields are optional):

- An identifier or alias that uniquely distinguishes each direction from all others – for example, inbound, outbound, eastbound, southbound, clockwise, etc. (**`key`**)
- The publicly understandable general destination name of the direction – for example 'downtown' or 'Hillsdale shopping center' (**`title`**). This field should be as short as possible.
- The type of the identifier – either DIRECTION_CODE or TRIP_ID (**`dirType`**)
- The list of stops in each direction/destination (**`Stop`**)
- An optional flag (true/false) indicating if direction contains all stops for the route (**`containsAllStopsInRoute`**)

Each direction may have one or more stops. Each stop record contains the following information (data element names are shown in parenthesis - some fields are optional):

- An identifier or alias that uniquely distinguishes each stop from all others (**`key`**)
- The name of the stop – usually the name of an intersection or station name (**`title`**)
- The position of the stop along the route (**`stopOrder`**)

The hierarchical relationship between all of the aforementioned data elements is documented in the XML DTD shown below.

Configuration Response:

```
<!-- Configuration information response received. -->
<!DOCTYPE ConfigurationDataMessage [
	<!ELEMENT ConfigurationDataMessage (Error* | (TripMappingData*,
ConfigurationData+)) >
	<!ELEMENT Error EMPTY >
	<!ATTLIST Error
		errorCode	CDATA #IMPLIED
		errorText	CDATA #REQUIRED >
```

```
	<!ELEMENT TripMappingData (TripMapping+) >
	<!ATTLIST TripMappingData
		agency		CDATA #REQUIRED
		version		CDATA #REQUIRED
		TimeStamp	CDATA #REQUIRED #Format: YYYY-MM-DDTHH:MM:SSZ where Z
indicates difference from UTC in (+/-)HH:MM format >
	<!ELEMENT TripMapping EMPTY >
	<!ATTLIST TripMapping
		tripID		CDATA #REQUIRED
		routeKey	CDATA #REQUIRED
		dirKey		CDATA #REQUIRED
		dirTitle	CDATA #IMPLIED
		tripType	CDATA #IMPLIED
		defaultTripForRouteDir (TRUE | FALSE) #IMPLIED >
	<!ELEMENT ConfigurationData (Route+ ) >
	<!ATTLIST ConfigurationData
		agency		CDATA #REQUIRED
		version		CDATA #REQUIRED
		numStops	CDATA #REQUIRED
		TimeStamp	CDATA #REQUIRED #Format: YYYY-MM-DDTHH:MM:SSZ where Z
indicates difference from UTC in (+/-)HH:MM format
		msg		CDATA >
	<!ELEMENT Route (Direction+) >
	<!ATTLIST Route
		key		CDATA #REQUIRED
		title		CDATA #REQUIRED >
	<!ELEMENT Direction (Stop+ ) >
	<!ATTLIST Direction
		key		CDATA #REQUIRED
		title		CDATA #IMPLIED
		dirType		(DIRECTION_CODE | TRIP_ID) #REQUIRED
		containsAllStopsInRoute (TRUE | FALSE) #IMPLIED >
	<!ELEMENT Stop EMPTY >
	<!ATTLIST Stop
		key		CDATA #REQUIRED
		title		CDATA #REQUIRED
		stopOrder	CDATA #IMPLIED
		shortCode	CDATA #IMPLIED >
]>
```

Agencies providing configuration data will follow the guidelines below in order to ensure correct processing of the configuration response message:

- Only one (1) Route element may exist per route code/key within a ConfigurationData element (i.e. routes must be unique within a ConfigurationData element).

- Only one (1) Direction element may exist per direction key within a Route element (i.e. direction codes/keys must be unique within a Route element).

- Only one (1) Stop element may exist per stop code/key within a Direction element (i.e. stop codes/keys must be unique within a Direction element).

- If a TripMappingData element is provided, only one (1) TripMapping element may exist per trip pattern ID within a TripMappingData element (i.e. trip pattern IDs must be unique within a TripMappingData element).

- If direction code based direction mapping is used (dirType=”DIRECTION_CODE”) configuration data shall not contain trip pattern mapping data section.

- If trip ID based direction mapping is used (dirType=”TRIP_ID”) it must use trip IDs used in trip pattern mapping section as direction key (“key”) value in the ‘Direction’ element.

- If trip ID based direction mapping is used in configuration data, prediction data shall use trip ID values for direction attributes (“dir”) in stop predictions elements (“StopPredictions”).

- Entire configuration dataset shall use consistent trip pattern mapping (either trip ID or direction key) for all directions across configuration dataset.

- Optional tripType attribute allows to include trips that are not revenue/passenger related. When included this attribute allows for one of two possible values - 'R' for revenue trip, or 'O' for out of service/deadhead/non-revenue trips. When this attribute is omitted the trip will be considered revenue.
- Optional stop attribute 'shortCode' in the Stop element allows to include a short code for stops/stations/terminal/berths using three or four alpha-numeric characters similar to airports (SFO, OAK, SJC).

Prediction Data Message

To obtain departure/arrival time predictions, the 511 system (as well as any other applications) may subscribe to updates from the JMS server. The JMS server will then send a prediction data message to all subscribers whenever departure/arrival time predictions change. The response message will contain either a list of errors or the requested prediction data.

Errors

An error record will contain the following information (data element names are shown in parenthesis):

- The integer error code if applicable (**`errorCode`**)
- The text of the error (**`errorText`**)

Predictions

Each prediction data record must contain the following information (data element names are shown in parenthesis):

- The name of the agency (**`agency`**)
- The version number of the configuration data (used to determine when changes have been since the last configuration update) (**`version`**)
- The date/time of the current snapshot of data (**`TimeStamp`**)
- The total number of stops (**`numStops`**)
- The list of departure/arrival time predictions – one for each route (**`StopPredictions`**)

Each agency may have one or more bus routes. Each bus route may possess multiple directions/destinations and may have several stops. Thus, each route must contain a list of the departure/arrival time predictions for each stop in <u>all</u> directions/destinations. Each prediction record must contain the following information (data element names are shown in parenthesis):

- The identifier or alias that uniquely distinguishes each route (**`route`**)
- The identifier of the direction/trip (**`dir`**) – Trip ID if TripMapping is used in configuration, direction key if TripMapping is not used.
- The identifier or alias that uniquely distinguishes each stop (**`stop`**)
- An optional message containing information concerning the route (**`msg`**)
- A list of predicted arrival and departure times for each stop (**`Ptimes`**)
- The type of prediction; either arrival or departure (**`PredictionType`**)
- Each stop may have one or more departure/arrival time predictions. Each departure/arrival time record must contain the time at which a vehicle is predicted to arrive at or depart from the stop (**`PredictionTime`**).
- Each departure/arrival time prediction is associated with a trip (tripID)

Each agency may have one or more bus routes. Each bus route may possess multiple directions/destinations and may operate several trips. Each trip record for a particular route and direction must contain the following information (data element names are shown in parenthesis):

- Each trip record will contain trip id (tripID)
- Each trip record will contain optional vehicle location information for that trip (vehicleID, vehicleLat, vehicleLong)

The hierarchical relationship between all of the aforementioned data elements is documented in the XML DTD shown below.

Prediction Response:

```
<!-- Prediction information received. -->
        <!DOCTYPE PredictionDataMessage [
             <!ELEMENT PredictionDataMessage (Error* | PredictionData*) >
             <!ELEMENT Error EMPTY >
             <!ATTLIST Error
                    errorCode    CDATA #IMPLIED
                    errorText    CDATA #REQUIRED >
             <!ELEMENT PredictionData (StopPredictions+, VehicleLocationData*) >
             <!ATTLIST PredictionData
                    agency       CDATA #REQUIRED
                    version      CDATA #REQUIRED
                    numStops     CDATA #REQUIRED
                    TimeStamp    CDATA #REQUIRED #Format: YYYY-MM-DDTHH:MM:SSZ where Z
indicates difference from UTC in (+/-)HH:MM format >
             <!ELEMENT StopPredictions (Ptimes*) >
             <!ATTLIST StopPredictions
                    route        CDATA #REQUIRED
                    dir          CDATA #REQUIRED
                    stop         CDATA #REQUIRED
                    msg          CDATA #IMPLIED >
             <!ELEMENT Ptimes EMPTY >
             <!ATTLIST Ptimes
                    PredictionType      (D | A ) #REQUIRED
                                 #Format: D – Departure Time
                                 #        A – Arrival Time
                    PredictionTime      CDATA #REQUIRED #Format: YYYY-MM-DDTHH:MM:SSZ
where Z indicates difference from UTC in (+/-)HH:MM format
                    tripID              CDATA #IMPLIED
                    vehicleID           CDATA #IMPLIED
        >
             <!ELEMENT VehicleLocationData (VehicleLocation*) >
             <!ATTLIST VehicleLocationData
                    route        CDATA #IMPLIED
                    dir          CDATA #IMPLIED >
             <!ELEMENT VehicleLocation EMPTY >
             <!ATTLIST VehicleLocation
                    tripID              CDATA #IMPLIED
                    vehicleID           CDATA #IMPLIED
                    vehicleLat          CDATA #IMPLIED
                    vehicleLong         CDATA #IMPLIED>
]>
```

Arrived Status Request Message

To obtain updated actual arrival times of the vehicles from the agencies, 511 issues an arrived status request message to the JMS server, which will get forwarded to the specific agencies. Each record optionally may contain the following information (data element names are shown in parenthesis):

- The name of the agency (**`agency`**)
- The identifier or alias that uniquely distinguishes each route (**`route`**)
- The identifier of the direction (**`dir`**) – Trip ID if TripMapping is used in configuration, direction key if TripMapping is not used.
- The identifier of the stop (**`stop)`**
- The start time for which data is requested (**`StartTime)`**
- The stop time for which data is requested (**`StopTime)`**
- The identifier of the trip (**`TripId)`**

All of the attributes are optional. If an attribute is left blank or not included, all data for that attribute should be retrieved (e.g. if attribute 'route' is null or blank, data for all routes should be retrieved).
The StartTime and StopTime fields are the time range for which status is being requested. Typically, this message should be requested once per day, and should be for the full previous day. For example, the Arrived Status Request Message could be configured to be sent once per day at 2AM. If it was sent on February 2, 2006, the start date would be 2/1/2006 12:00:00 AM, and end date would be 2/1/2006 11:59:59 PM

The format of the message is documented in the XML DTD shown below:

`Arrival Status Request:`

```
<!-- Request sent to the JMS server for updates for actual vehicle arrival and
departure times. -->
<!DOCTYPE ArrivalStatusRequest [
<!ELEMENT ArrivalStatusRequest>
<!ATTLIST ArrivalStatusRequest
       agency       CDATA #IMPLIED #Format: Comma separated list
       route        CDATA #IMPLIED #Format: Comma separated list
       dir          CDATA #IMPLIED #Format: Comma separated list
       stop         CDATA #IMPLIED #Format: Comma separated list
       startTime    CDATA #IMPLIED #Format: YYYY-MM-DDTHH:MM:SSZ where Z indicates
difference from UTC in (+/-)HH:MM format
       stopTime     CDATA #IMPLIED #Format: YYYY-MM-DDTHH:MM:SSZ where Z indicates
difference from UTC in (+/-)HH:MM format >
]>
```

Arrived Status Response Message

In response to arrived status request messages, an arrived status response message will be sent. The response message will contain either a list of errors or the requested arrival status data.

Errors

If present, an error record will contain the following information (data element names are shown in parenthesis):

- The integer error code if applicable (**`errorCode`**)
- The text of the error (**`errorText`**)

Arrivals

Each arrived status response record must contain the following information (data element names are shown in parenthesis):

- The name of the agency (**`agency`**)
- The date/time of the current snapshot of data (**`TimeStamp`**)
- The identifier or alias that uniquely distinguishes each route (**`route`**)
- The name of the direction/destination (**`dir`**) – Trip ID if TripMapping is used in configuration, direction key if TripMapping is not used.
- The identifier or alias that uniquely distinguishes each stop (**`stop`**)
- The unique identifier of the vehicle (**`VehicleId`**)
- The time at which the bus arrived at the stop (**`ArrivalTime`**)

The relationship between all of the aforementioned data elements is documented in the XML DTD shown below.

`Arrival Status Response:`

```
<!-- Arrival Status information received. -->
<!DOCTYPE ArrivalStatusDataMessage [
	<!ELEMENT ArrivalStatusDataMessage (Error* | ArrivalStatusData*) >
	<!ELEMENT Error EMPTY >
	<!ATTLIST Error
		errorCode	CDATA #IMPLIED
		errorText	CDATA #REQUIRED >
	<!ELEMENT ArrivalStatusData EMPTY >
	<!ATTLIST ArrivalStatusData
		agency		CDATA #REQUIRED
		TimeStamp	CDATA #REQUIRED #Format: YYYY-MM-DDTHH:MM:SSZ where Z
indicates difference from UTC in (+/-)HH:MM format
		route		CDATA #REQUIRED
		dir		CDATA #REQUIRED
		stop		CDATA #REQUIRED
		VehicleId	CDATA #REQUIRED
		ArrivalTime	CDATA #REQUIRED #Format: YYYY-MM-DDTHH:MM:SSZ where Z
indicates difference from UTC in (+/-)HH:MM format>
]>
```

Pattern to Direction Mapping Table

If an agency wishes to supply configuration and prediction data with routes and patterns, they must supply a table consisting of one row for each pattern in their system. Each row must contain the following fields. All fields are character strings

Field Name	**Description**
Route Code	The agency internal route code
Trip Pattern ID	The agency internal pattern id
Direction Code	The internal agency direction code
Direction	The direction of the route: e.g. inbound or outbound
Description	Agency internal description of the pattern

Frequently Asked Questions

Q. Can you give us examples of the expected xml for:
Configuration Response Message
Prediction Data Message
Arrived Status Response Message

This would give us a clearer idea of what is specifically expected for the text attribute fields. For example, would this be a good direction Element since we have multiple direction/destinations within the same route?
<direction key="4356" title="Inbound to Transbay Terminal from 42nd Ave & Clement" >
A. Examples of each message have been included with the sample JMS application

Q. The TimeStamp is defined with an offset format in 2 cases. What is the meaning of the offset in the Configuration Response Message and the Prediction Time of the Prediction Data Message. Offset from what?
A> Offset from Greenwich Mean Time

Q. Since one stop can service multiple route/directions combinations, I am assuming that the numStops attribute is the number of unique stops defined in the whole response.
A. Yes the numStops is the number of stops in the entire response

Q. Is the Configuration Response Message Direction key attribute the same as the Prediction Data Message Predictions "dir" value?
A. Yes –
MTC will allow two types of direction keys. Each agency must decide whether they will be sending direction names or patterns. For this project each route can have multiple patterns per direction. If the agency will be sending patterns, they must also provide a mapping table, which will allow MTC to map from the pattern to a direction name that will be reported to callers

If an agency decides to provide patterns, they must supply a pattern to direction mapping table. The description of the table is in this document on page 11.

See the first question and response for more details

Q. What is the expected number of predictions per stop? Nextbus proposes returning a minimum of the next 3 per route/direction/stop combination if available. Since stops are shared between directions some stops will end-up having more than 3 predictions.
A. A maximum of 4 predictions with a prediction time of no more than 90 minutes in the future will be accepted by 511. All other predictions will be discarded.