



ADVANCED WIRELESS VEHICLE DETECTION TECHNOLOGY



# **Sensys™ Wireless Vehicle Detection System**

## **Sensys Travel Time Server Protocol Preliminary Specification**

P/N 152-240-001-018

May 2009

Sensys Networks  
2560 Ninth Street, Suite 219  
Berkeley, CA 94710  
[www.sensysnetworks.com](http://www.sensysnetworks.com)

---

# Contents

<b>Sensys Travel Time Server Protocol</b> .....	<b>5</b>
<b>Up-Link Messages</b> .....	<b>5</b>
<b>Down-Link Messages</b> .....	<b>6</b>
<b>Policy File</b> .....	<b>7</b>
<b>Configuration File</b> .....	<b>7</b>
Example: C Code to Compute the Distance Between Two Lat/Long Points:.....	7
Example: Configuration File.....	8
<b>Message Attributes</b> .....	<b>8</b>
Example: Messages.....	10
<b>Filters</b> .....	<b>10</b>

## Index of Tables

<b>Table 1: Up-link (Client to Server) Messages</b> .....	<b>5</b>
<b>Table 2: Down-link (Server to Client) Messages</b> .....	<b>7</b>
<b>Table 3: Message Attributes</b> .....	<b>10</b>

---

## Document Properties

This document is reference material for the Sensys™ Wireless Vehicle Detection System from Sensys Networks, Inc.

P/N 152-240-001-018 Revision: C

Sensys Networks, Inc. makes no representation or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Furthermore, Sensys Networks, Inc. reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Sensys Networks, Inc. to notify any person or organization of such revisions or changes.

© 2008, 2009 – All rights reserved.

Sensys and the Sensys logo are trademarks of Sensys Networks, Inc. All other products, names and services are trademarks or registered trademarks of their respective owners.

## Regulatory Statements

### FCC Compliance Statement

This device complies with part 15 of the FCC rules. Operation is subject to the following two conditions:

- (1) This device may not cause harmful interference.
- (2) This device must accept any interference received, including interference that may cause undesired operation.

Any changes or modifications to this product not authorized by Sensys Networks, Inc., could void the EMC compliance and negate the authority to operate the product.

### RF Exposure Statement

This device has been tested and meets the FCC RF exposure guidelines. It should be installed and operated with a minimum distance of 20 cm between the radiator of RF energy and the body of users, operators or others.

Improper use or tampering with the device is prohibited and may not ensure compliance with FCC exposure guidelines.

## Warnings

### No Safety Switching

Sensys Networks, Inc. does not allow its equipment to be used for safety applications such as controlling a mechanical gate or switching a train to avoid a collision.

### Lithium Thionyl Chloride Batteries

Sensys Networks uses Lithium Thionyl Chloride batteries in the following products:

- Sensors (VSN240-F, VSN240-T)
- Repeaters (RP240-B, and RP240-B-LL)

---

Lithium batteries are widely used in electronic products because they contain more energy per unit - weight than conventional batteries. However, the same properties that deliver high energy density also contribute to potential hazards if the batteries are damaged. Improper use or handling of the batteries may result in leakage or release of battery contents, explosion or fire.

Following are the recommendations of the battery manufacturer for proper use and handling of batteries in the Sensys devices mentioned above:

- **DO NOT** charge or attempt to recharge the batteries (batteries are NOT rechargeable)
- **DO NOT** crush or puncture batteries
- **DO NOT** short-circuit the batteries
- **DO NOT** force over-discharge of the batteries
- **DO NOT** incinerate or expose batteries to excessive heating
- **DO NOT** expose battery contents to water
- **DO** dispose of batteries and devices containing batteries in accordance with local regulations

Sensys Networks sensors contain no serviceable parts and should never be disassembled. Installation and removal of sensors from pavement should only be done by trained personnel and care should be taken to insure that the sensor casing is not punctured or crushed.

Additional safety information is available from the battery's manufacturer:

- Sensor battery cell: [http://www.able-battery.com/msds/ABLE\\_MSDS\\_ER14505.pdf](http://www.able-battery.com/msds/ABLE_MSDS_ER14505.pdf)
- Repeater battery cell: [http://www.able-battery.com/msds/ABLE\\_MSDS\\_ER34615.pdf](http://www.able-battery.com/msds/ABLE_MSDS_ER34615.pdf)

## Nature of Specifications

Sensys Networks continually reviews and revises its products, applications, and services. Information contained herein is subject to change without notice and should not be interpreted as a commitment by Sensys Networks to provide any particular feature or method.

## Document Control

Please address questions, suggestions or corrections to [support@sensysnetworks.com](mailto:support@sensysnetworks.com).

## Contact Information

Sensys Networks, Inc.  
2560 Ninth Street, Suite 219  
Berkeley, CA 94710 USA  
+1 (510) 548-4620  
[www.sensysnetworks.com](http://www.sensysnetworks.com)

# Sensys Travel Time Server Protocol

The Sensys Travel Time Server (STTS) protocol was developed to export information from the Sensys Travel Time Engine to IP based applications. The protocol was designed to be compatible with XML based Adobe Flash 9 (AS3) client programs, but other clients can be used if they implement this protocol. The STTS operates as a TCP server.

The data model for the STTS consists of a list of segments. Each segment corresponds to vehicles traveling from a start-point (upstream sensor array) to an end-point (downstream sensor array). Each segment must have a unique id.

The STTS protocol is implemented using a TCP connection on a predetermined port. One port can handle a large number of travel time segments. Generally, each port provides travel time information for a geographical region, for example, San Diego or San Francisco. The STTS can operate on any port but ports below 1024 should be avoided.

To be compatible with Flash 9, each message in either direction consists of a single XML element followed by the ASCII NUL character (i.e. zero byte). Messages are streamed onto the TCP connection.

## Up-Link Messages

The table below lists the up-link (client to server) messages:

XML Element tag	XML Attributes	Usage
policy-file-request	none	Security feature of Flash 9 – optional otherwise
configuration-file-request	none	The server will send the current configuration in the down-link.
aggregate-request	none	Send the most recent aggregate message for each segment.
add-filter	msgName, id, include	Allows a client to restrict the number of down-link messages by msgName or by id.

Table 1: Up-link (Client to Server) Messages

## Down-Link Messages

Down-link messages are organized in four groups: *management*, *aggregate*, *matching*, and *real-time*. Every vehicle detected on either the upstream or downstream sensor array will be counted in one or two of the match/unmatched-up/unmatched-down messages and one or two of the vehicle-up/vehicle-down messages. The aggregate messages use a running interval and therefore a vehicle may influence one or more aggregate messages.

XML Element tag	XML Attributes	Usage
policy-file	none	Security feature of Flash 9
configuration	none	The current configuration of all segments.
confirm-filter	none	Reply message for the add-filter message
aggregate	id, time, travelTimeDist, color, upstream, downstream, timeWindow, carsInSegment90, matches, averageScore, los, upstreamOccupancy, downstreamOccupancy	Sent with each vehicle match providing the current aggregate travel time. Aggregates are taken over an interval of the past 100 vehicles or the past 30 minutes, whichever is less.
match	id, time, travelTime, score, and carsInSegment	Sent with each vehicle match providing per vehicle travel time. <i>carsInSegment</i> is the number of vehicles in the segment when this match was at the downstream array. These messages are used for aggregation. Some validation is performed on matches to insure that they are in range and scores are reasonable.
unmatched-up	id, time, carsInSegment	Sent with each unmatched vehicle at the upstream sensor array. For total volume at the upstream, sum the number of these messages and match messages.
unmatched-down	id, time, carsInSegment	Sent with each unmatched vehicle at the downstream sensor array. For total volume at the downstream, sum the number of these messages and match messages.
vehicle-up	id, time, carsInSegment, diag, duration, and bias	Sent in real time when a vehicle is detected at the upstream sensor. The <i>carsInSegment</i> attribute is updated in real time and therefore not as accurate as those in the match or unmatched messages but will suffer no algorithmic delay.

XML Element tag	XML Attributes	Usage
vehicle-down	id, time, carsInSegment, diag, duration, and bias	Sent in real time when a vehicle is detected at the downstream sensor. The <i>carsInSegment</i> attribute is updated in real time and therefore not as accurate as those in the match or unmatched messages but will suffer no algorithmic delay.

Table 2: Down-link (Server to Client) Messages

## Policy File

The policy file is set in accordance with the requirements of Adobe Flash Player 9. See [Policy file changes in Flash Player 9 and Flash Player 10](#) on the Adobe web site for a description of this file. Non flash-based clients need not use the policy file.

## Configuration File

The configuration file consists of regional information and a description for each segment. The configuration file is sent as a single configuration XML element. The configuration is sent with a status of “complete” - meaning the configuration file is complete - or “incomplete” - meaning the configuration is missing information for some segments or that some of the segment information is not complete. Configuration elements will be sent at any time when requested with the *configuration-file-request* message and will be sent automatically and spontaneously as a result of configuration changes. Changes include the addition of segments, removal of segments, changes to existing segments, as well as changes to the regional information.

The regional information consists of the center latitude/longitude (lat/long) and the initial zoom level for use in a Google Maps application. Other information will be included in the future.

The description for each segment includes a unique id, description, classification (I, II, or III), pixel offsets for a map marker, zoom levels for map display, coordinates and width of a line showing the path of the segment on a map. The first point in the segment path is lat/long of the starting sensor array; the last point is the ending sensor array. The length of a segment can be computed by summing the distance between the lat/long of the points along the path. An optional attribute “miles” or “km” may be provided for a segment in which case the value given in that attribute will override the computed segment length.

### Example: C Code to Compute the Distance Between Two Lat/Long Points:

```
static double // return distance in miles
LatLongDistance(double lat1, double long1, double lat2, double long2)
{
    double t;

    lat1 *= M_PI/180.0; // convert degrees to radians
```

```

lat2 *= M_PI/180.0;
long1 *= M_PI/180.0;
long2 *= M_PI/180.0;

t = sin(lat1)*sin(lat2) + cos(lat1)*cos(lat2)*cos(long1 - long2);
return (atan(-t/sqrt(1.0 - t*t)) + M_PI/2.0) * 3956.15898292;
}

```

## Example: Configuration File

```

<configuration status="complete" >
<center lat="32.631496" long="-117.01375" zoomLevel="14"></center>
<segments>
  <segment id="008006" description="Telegraph Canyon Rd/La Media Rd-Heritage Dr"
  classification="I">
    <markerOffset x="-5" y="-5"></markerOffset>
    <zoomLevels min="12" max="12"></zoomLevels>
    <points width="12">
      <point lat="32.625500" long="-117.008400"></point>
      <point lat="32.626353" long="-117.006369"></point>
      <point lat="32.629669" long="-117.002077"></point>
      <point lat="32.632190" long="-116.998783"></point>
      <point lat="32.633527" long="-116.996326"></point>
      <point lat="32.638153" long="-116.991391"></point>
      <point lat="32.638650" long="-116.990758"></point>
    </points>
  </segment>
  <segment id="005003" description="Telegraph Canyon Rd/Paseo Ladera-Paseo del Rey"
  classification="I">
    <markerOffset y="-10"></markerOffset>
    <zoomLevels min="12" max="12"></zoomLevels>
    <points width="12">
      <point lat="32.627600" long="-117.034600"></point>
      <point lat="32.626515" long="-117.032247"></point>
      <point lat="32.625521" long="-117.027129"></point>
      <point lat="32.625000" long="-117.025900"></point>
      <point lat="32.623642" long="-117.022634"></point>
      <point lat="32.623400" long="-117.021000"></point>
    </points>
  </segment>
</segments>
</configuration>

```

## Message Attributes

Data for each segment is sent using a separate set of messages. Messages use an element tag followed by one or more of the following attributes:

XML Attribute	Format	Units	Meaning
averageScore	Floating Point		Average "score" value of matches over the aggregate interval. This may be a reasonable long term measure of the effectiveness of the sensor array.
bias	Floating Point	Seconds	The moment of a vehicle as it drives over the sensor array. Indicates the lateral position of a vehicle in the lane.
carsInSegment	Integer	Vehicles	Number of vehicles in the segment.
carsInSegment90	Integer	Vehicles	90 <sup>th</sup> percentile number of vehicles in the segment over the interval. Used in aggregate messages.
color	String	0xRRGGBB	Segment color indicating congestion.
diag	String		This attribute is only present if an error condition (such as malfunctioning or missing sensor) is detected.
downstream	Integer	Vehicles	Number of vehicles counted over the downstream array during the aggregate interval.
downstreamOccupancy	Floating Point	Percent	Occupancy of downstream array over the aggregate interval. Number is between 0.0 and 100.0.
duration	Floating Point	Seconds	The amount of time the vehicle was over the sensor array.
id	String		The unique segment id. Segment Id's are current assigned, by convention, by concatenating two sensor array tags. This convention may change.
los	String	A-F	Level of Service is computed over the aggregate interval. Level of Service uses a segment classification I, II, or III in the configuration message and the tables in the Highway Capacity Manual and the median travel time to compute a grade level of service. If classification is not provided a freeFlowSpeed value can be specified and the Level of Service is computed using percentages of free-flow speed.
matches	Integer	Vehicles	The actual number of samples matches used to generate the aggregate. Will range from 0 to 100.
score	Floating Point		Error measure indicating the quality of the vehicle match. Values <=0.25 indicate a good match, >= .50 a bad match. Scores > 1000 indicate incomplete signatures.
time	Floating Point	Epoch time	Time vehicle leaves a sensor array
timeWindow	Floating Point	Seconds	The actual time interval for the aggregate.
travelTime	Floating Point	Seconds	Travel time for a single vehicle from one sensor array to the next
travelTimeDist	Floating Point	Seconds, Seconds, ...	Distribution of travel times. First number is the minimum, 2 <sup>nd</sup> value is 10 <sup>th</sup> percentile travel time, 3 <sup>rd</sup> is 20 <sup>th</sup> percentile ... up to 11 <sup>th</sup> value is the maximum.

XML Attribute	Format	Units	Meaning
upstream	Integer	Vehicles	Number of vehicles counted over the upstream array during the aggregate interval. The interval used for upstream is based on estimated downstream times, not actual upstream times.
upstreamOccupancy	Floating Point	Percent	Occupancy of upstream array over the aggregate interval. Number is between 0.0 and 100.0.

Table 3: Message Attributes

## Example: Messages

```
<aggregate id="001002" time="1231888442"
travelTimeDist="12,15,16,17,18,19,20,34,45,60,80" color="0x00ff00" up="161"
down="180" timeWindow="836" carsInSegment="11" matches="100" averageScore="0.20"
los="A" upstreamOccupancy="1.0" downstreamOccupancy="2.0" />

<match id="004005" time="1229156012" travelTime="22" score="0.19" carsInSegment="2"/>

<vehicle-up id="004005" time="1229156089" carsInSegment="3" duration="0.400"
bias="0.2"/>
```

## Filters

To reduce the bandwidth required between the server and the client, the server implements a set of filters to drop data messages that the client does not want. By default filtering is disabled and all messages are sent to a client. The *add-filter* message has three possible attributes: *msgName*, *id*, and *include*.

- *Include* can be set to either “0” (do not include) or “1” (default, include).
- *Id* can be set to either a valid id or by default all ids are included.
- *MsgName* can be set to either a specific message element tag or by default all tags are included.

Filters are processed in LIFO order.

Below is an example of the filter commands required to configure the server to send only segment-aggregate messages from all segments:

```
<add-filter include="0" />
<add-filter msgName="aggregate" include="1" />
```

A *filter-confirm* message is sent to the client upon activation of the filter. A *filter-reject* message is sent to the client if the filter cannot be activated. Either message contains a copy of the original *add-filter* message:

```
<confirm-filter> <add-filter include="0" /> </confirm-filter>  
<confirm-filter> <add-filter msgName="aggregate" include="1" /> </confirm-filter>
```

*Note:* the implementation of filters is currently pretty crude so do not rely on filtering to implement client side logic.

