

Basic GTS Enterprise Device Communication Server

CalAmp

Copyright © 2007-2012 GeoTelematic Solutions, Inc.
All rights reserved

projects@geotelematic.com
<http://www.geotelematic.com>

<i>Manual Revision History</i>			
Rev	Date	Changed	Author
0.1.0	2010/12/31	Initial Release	MDF
0.1.1	2011/04/01	Updated CalAmp accumulator properties	MDF

Device Communication Server - CalAmp

Contents:

- 1 Introduction**
- 2 Configuring the Server**
 - 2.1 Changing the Server "Listen" Ports**
 - 2.2 Setting the "Unique-ID" Prefix Characters**
 - 2.3 Setting the CalAmp Properties**
 - 2.4 Changing the Default EventCode to StatusCode Mapping**
 - 2.5 Setting the Available Server-To-Device Commands**
- 3 Running the Server**
 - 3.1 Manually Starting the Server**
 - 3.2 Automatically Starting the Server on System Reboot**
 - 3.3 Monitoring the Log File**
- 4 Adding Devices to an Account**
 - 4.1 Creating a New Device Record**
 - 4.2 The "Server ID" Field.**
- 5 Testing a New Configured Device**
 - 5.1 Monitoring the Log Files**
 - 5.2 Viewing the Unassigned Devices Report**

Appendix:

- A) Troubleshooting CalAmp Device Connection Issues**

1) Introduction

This manual describes how to configure and run the GTS Enterprise device communication server (DCS) for the **CalAmp** hardware GPS tracking/telematic devices. The following features are supported for the **CalAmp** DCS:

- Receive incoming data packets via UDP.
- Support for sending commands to the device through the login web-interface.

2) Configuring the Server

The following section refers to the runtime configuration file for the **CalAmp** device communication server, which can be found in the GTS Installation directory at "dcservers/dcserver_calamp.xml".

2.1) Changing the Server "Listen" Ports.

The ports on which the **CalAmp** DCS listens for incoming data packets is specified on the "ListenPorts" tag:

```
<ListenPorts
  tcpPort=""
  udpPort="20500"
/>
```

If required, the "listen" port can be changed to fit the requirements of your runtime environment, however the specified ports above are those recommended by **CalAmp** and should remain as specified unless otherwise necessary. The script programmed into the **CalAmp** device will also need to be configured to transmit data to the same port as the server used to listen for incoming data packets. (NOTE: Some versions of the **CalAmp** device firmware require that the specified port be less than 32767).

The "listen" ports must be open through the firewall in order for the remote device to send data to the **CalAmp** server.

If packet acknowledgment is required, any acknowledgments sent by the server back to the remote device must be sent from the same IP address to which the remote device sent its data packet. If your server responds to more than one IP address, then the **CalAmp** server listener must be bound to the same IP address/interface used by the remote tracking devices. This is set in the top-level "dcservers.xml" file, on the "DCServerConfig" tag, "bindAddress" attribute.

2.2) Setting the "Unique-ID" Prefix Characters.

The Unique-ID prefix characters can be set in the "UniqueIDPrefix" tag section:

```
<UniqueIDPrefix><![CDATA[
  ca_
  calamp_
  imei_
  *
]]></UniqueIDPrefix>
```

These prefix characters are used to 'prepend' to the IMEI or ESN number reported by the device to look up the owning Account/Device record for this device. For instance, if the **CalAmp** device reports the ESN as the mobile-ID and the ESN number is "4321098765" (for example), then the system will search for the owning Device using the following Unique-ID keys, in the order specified:

```
ca_4321098765
calamp_4321098765
imei_4321098765
4321098765
```

Note that the '*' character by itself indicates that the system should look up the ESN number without any prefixing characters.

The **CalAmp** devices can be scripted to report the device ESN, IMEI, or other possible unique mobile id, however, the default 10-digit ESN (which typically begins with the digit "4") is the recommended value to use as the unique mobile id.

To bind a **CalAmp** device to a specified Account/Device record, set the "Unique ID:" field on the Device/Vehicle Admin page to the appropriate prefixed unique-id value. For example:

```
Unique ID: ca_4321098765
```

2.3) Setting the CalAmp Properties

Properties which effect the behavior of the **CalAmp** server are set in the "Properties" tag section. The following properties may be set (check the "dcservers/dcserver_calamp.xml" file for the latest available property settings):

```
<Property key="minimumSpeedKPH">3.0</Property>
```

This is the minimum acceptable speed value, below which the device will considered not moving, and the speed will be explicitly set to "0.0".

```
<Property key="statusLocationInMotion">true</Property>
```

If "true", the DCS will replace an event which otherwise is defined to be a general **STATUS_LOCATION** status code instead with a **STATUS_MOTION_IN_MOTION** status code, if the indicated speed of the vehicle is greater than zero.

```
<Property key="simulateDigitalInputs">0x00</Property>
```

If specified, this mask value indicates which of the **CalAmp** digital inputs should be check for state changes, and if a state change is detected, an event with the appropriate digital input state change status code will be generated. The special value "false" is the same as entering a mask value of "0", which indicates that no digital input state changes should be detected.

```
<Property key="estimateOdometer">false</Property>
```

If "true", the DCS will calculate the current event odometer based on the distance traveled since the last valid GPS location.

```
<Property key="simulateGeozones">true</Property>
```

If "true", the DCS will check for geozone arrivals/departures and insert the appropriate arrive/depart events.

```
<Property key="startStopSupported">false</Property>
```

This property should be set "true" if the **CalAmp** script supports generated Start/Stop status code events from within the **CalAmp** tracking device itself. This property should be set "false" if the **CalAmp** script does not support generating Start/Stop status code events.

```
<Property key="accum.odometer">-1</Property>
```

```
<Property key="accum.battery">-1</Property>
```

```
<Property key="accum.idleTime">-1</Property>
```

```
<Property key="accum.runTime">-1</Property>
```

```
<Property key="accum.workTime">-1</Property>
```

```
<Property key="accum.brakeForce">-1</Property>
```

```
<Property key="accum.driverID">-1</Property>
```

```
<Property key="accum.fuelLevel">-1</Property>
```

```
<Property key="accum.temp0">-1</Property>
```

```
<Property key="accum.temp1">-1</Property>
```

```
<Property key="accum.temp2">-1</Property>
```

```
<Property key="accum.temp3">-1</Property>
```

The **CalAmp** server supports various accumulator values, including *Odometer* (based on internally calculated GPS distances), *Idle-Time* (internally calculated number of seconds that the engine is running, but the vehicle is stopped), *Run-Time* (internally calculated number of seconds that the engine is running), *Driver ID* (usually obtained from an iButton key-fob), and *Temperatures* (usually obtained from a Dallas Semiconductor 1-wire temperature sensor). The value for these properties should match the accumulator used on the **CalAmp** device which provides this information. A value of "-1" indicates that the **CalAmp** script does not provide information for the specified accumulator type.

See the "accum.*xxxxxxxx*" property definitions in the "dcservers/dcserver_calamp.xml" file for the latest available accumulator settings.

2.4) Changing the Default EventCode to StatusCode Mapping.

The EventCode to StatusCode mapping is specified in the "EventCodeMap" and "Code" tag sections. The following is the default mapping used to map **CalAmp** event codes to GTS status codes:

```
<EventCodeMap enabled="true">
  <!-- These are the default EventCode to StatusCode translations: -->
  <Code key="0x00">0xF020</Code> <!-- 0: StatusCodes.STATUS_LOCATION -->
  <Code key="0x01">0xF010</Code> <!-- 1: StatusCodes.STATUS_INITIALIZED -->
  <Code key="0x02">0xFD41</Code> <!-- 2: StatusCodes.STATUS_CONFIG_RESET -->
  <Code key="0x0A">0xF040</Code> <!-- 10: StatusCodes.STATUS_QUERY -->
  <Code key="0x11">0xF111</Code> <!-- 17: StatusCodes.STATUS_MOTION_START -->
  <Code key="0x12">0xF112</Code> <!-- 18: StatusCodes.STATUS_MOTION_IN_MOTION -->
  <Code key="0x13">0xF113</Code> <!-- 19: StatusCodes.STATUS_MOTION_STOP -->
  <Code key="0x14">0xF114</Code> <!-- 20: StatusCodes.STATUS_MOTION_DORMANT -->
  <Code key="0x15">0xF116</Code> <!-- 21: StatusCodes.STATUS_MOTION_IDLE -->
  <Code key="0x16">0xF11A</Code> <!-- 22: StatusCodes.STATUS_MOTION_EXCESS_SPEED -->
  <Code key="0x17">0xF118</Code> <!-- 23: StatusCodes.STATUS_MOTION_EXCESS_IDLE -->
  <Code key="0x18">0xF930</Code> <!-- 24: StatusCodes.STATUS_EXCESS BRAKING -->
  <Code key="0x19">0xF937</Code> <!-- 25: StatusCodes.STATUS_EXCESS CORNERING -->
  <Code key="0x1B">0xF123</Code> <!-- 27: StatusCodes.STATUS_MOTION_ACCELERATION -->
  <Code key="0x1C">0xF11F</Code> <!-- 28: StatusCodes.STATUS_MOTION_HEADING -->
  <Code key="0x21">0xF210</Code> <!-- 33: StatusCodes.STATUS_GEOFENCE_ARRIVE -->
  <Code key="0x22">0xF230</Code> <!-- 34: StatusCodes.STATUS_GEOFENCE_DEPART -->
  <Code key="0x30">0xF420</Code> <!-- 48: StatusCodes.STATUS_INPUT_ON_00 -->
  <Code key="0x31">0xF421</Code> <!-- 49: StatusCodes.STATUS_INPUT_ON_01 -->
  <Code key="0x32">0xF422</Code> <!-- 50: StatusCodes.STATUS_INPUT_ON_02 -->
  <Code key="0x33">0xF423</Code> <!-- 51: StatusCodes.STATUS_INPUT_ON_03 -->
  <Code key="0x34">0xF424</Code> <!-- 52: StatusCodes.STATUS_INPUT_ON_04 -->
  <Code key="0x35">0xF425</Code> <!-- 53: StatusCodes.STATUS_INPUT_ON_05 -->
  <Code key="0x36">0xF426</Code> <!-- 54: StatusCodes.STATUS_INPUT_ON_06 -->
  <Code key="0x37">0xF427</Code> <!-- 55: StatusCodes.STATUS_INPUT_ON_07 -->
  <Code key="0x38">0xF428</Code> <!-- 56: StatusCodes.STATUS_INPUT_ON_08 -->
  <Code key="0x3F">0xF401</Code> <!-- 63: StatusCodes.STATUS_IGNITION_ON -->
  <Code key="0x40">0xF440</Code> <!-- 64: StatusCodes.STATUS_INPUT_OFF_00 -->
  <Code key="0x41">0xF441</Code> <!-- 65: StatusCodes.STATUS_INPUT_OFF_01 -->
  <Code key="0x42">0xF442</Code> <!-- 66: StatusCodes.STATUS_INPUT_OFF_02 -->
  <Code key="0x43">0xF443</Code> <!-- 67: StatusCodes.STATUS_INPUT_OFF_03 -->
  <Code key="0x44">0xF444</Code> <!-- 68: StatusCodes.STATUS_INPUT_OFF_04 -->
  <Code key="0x45">0xF445</Code> <!-- 69: StatusCodes.STATUS_INPUT_OFF_05 -->
  <Code key="0x46">0xF446</Code> <!-- 70: StatusCodes.STATUS_INPUT_OFF_06 -->
  <Code key="0x47">0xF447</Code> <!-- 71: StatusCodes.STATUS_INPUT_OFF_07 -->
  <Code key="0x48">0xF448</Code> <!-- 72: StatusCodes.STATUS_INPUT_OFF_08 -->
  <Code key="0x4F">0xF403</Code> <!-- 79: StatusCodes.STATUS_IGNITION_OFF -->
```

```

<Code key="0x50">0xF4B0</Code> <!-- 80: StatusCodes.STATUS_ELAPSED_LIMIT_00 -->
<Code key="0x51">0xF4B1</Code> <!-- 81: StatusCodes.STATUS_ELAPSED_LIMIT_01 -->
<Code key="0x52">0xF4B2</Code> <!-- 82: StatusCodes.STATUS_ELAPSED_LIMIT_02 -->
<Code key="0x53">0xF4B3</Code> <!-- 83: StatusCodes.STATUS_ELAPSED_LIMIT_03 -->
<Code key="0x54">0xF4B4</Code> <!-- 84: StatusCodes.STATUS_ELAPSED_LIMIT_04 -->
<Code key="0x55">0xF4B5</Code> <!-- 85: StatusCodes.STATUS_ELAPSED_LIMIT_05 -->
<Code key="0x56">0xF4B6</Code> <!-- 86: StatusCodes.STATUS_ELAPSED_LIMIT_06 -->
<Code key="0x57">0xF4B7</Code> <!-- 87: StatusCodes.STATUS_ELAPSED_LIMIT_07 -->
<Code key="0x60">0xFD48</Code> <!-- 96: StatusCodes.STATUS_SUSPEND -->
<Code key="0x61">0xFD4A</Code> <!-- 97: StatusCodes.STATUS_RESUME -->
<Code key="0x66">0xFD11</Code> <!-- 102: StatusCodes.STATUS_BATTERY_LEVEL -->
<Code key="0x70">0xF960</Code> <!-- 112: StatusCodes.STATUS_EXCESS_ACCEL -->
<Code key="0x71">0xF961</Code> <!-- 113: StatusCodes.STATUS_EXCESS_ACCEL_2 -->
<Code key="0x72">0xF962</Code> <!-- 114: StatusCodes.STATUS_EXCESS_ACCEL_3 -->
<Code key="0x76">0xF930</Code> <!-- 118: StatusCodes.STATUS_EXCESS BRAKING -->
<Code key="0x77">0xF931</Code> <!-- 119: StatusCodes.STATUS_EXCESS BRAKING_2 -->
<Code key="0x78">0xF932</Code> <!-- 120: StatusCodes.STATUS_EXCESS BRAKING_3 -->
<Code key="0x7C">0xF937</Code> <!-- 124: StatusCodes.STATUS_EXCESS CORNERING -->
<Code key="0x7D">0xF938</Code> <!-- 125: StatusCodes.STATUS_EXCESS CORNERING_2 -->
<Code key="0x7E">0xF939</Code> <!-- 126: StatusCodes.STATUS_EXCESS CORNERING_3 -->
<Code key="0x90">0xF941</Code> <!-- 144: StatusCodes.STATUS_IMPACT -->
<Code key="0xFF">0xF020</Code> <!-- 255: StatusCodes.STATUS_LOCATION -->
</EventCodeMap>

```

This section is used to map a defined **CalAmp** event-code to a GTS Enterprise status-code. It can also be used to ignore an event for a specific event code value.

The value that should be specified on the Code "key" attribute is the defined event code which is sent by the **CalAmp** device on the data packet. This value may be specified in either decimal, or hexadecimal notation.

The text value of the "Code" tag should be the status-code to which the event code should be mapped. The special value "ignore" can be used to cause events which specify this event code to be ignored. The special value "default" indicates that the status code on the generated event should be **STATUS_LOCATION** if the vehicle is not moving, and **STATUS MOTION IN MOTION** if the vehicle is moving. Values specified as either decimal or hexadecimal will be used as the status code on the generated event.

Values received from the device which are not mapped in the above "EventCodeMap" section will be automatically mapped to the code "0xE0xx", where "xx" is the hex representation of the code received from the device.

Refer to the "**Status Codes and Description**" documentation for a list of currently defined status codes.

2.5) Setting the Available Server-To-Device Commands.

Commands which are to be sent to the **CalAmp** device based on user requests are defined in the "Commands" tag section:

```
<Commands dispatchPort="31050">

  <AclName>acl.dcs.calamp</AclName>

  <Command name="LocateNow" enabled="true">
    <Type>map, admin</Type>
    <Description>Locate Now</Description>
    <String protocol="udp"><![CDATA[ack:false action:10 data8:1]]></String>
    <StatusCode></StatusCode>
  </Command>

  ...

  <Command name="LocateSMS" enabled="true">
    <Type>map, admin</Type>
    <Description>SMS Locate</Description>
    <String protocol="sms"><![CDATA[!R3,1,10]]></String>
    <!--
      === !R3,1,10
      ===      1 = Send report in ACK format
      ===      10 = The status/event code placed in the reported event
    -->
    <StatusCode></StatusCode>
  </Command>

  ...

</Commands>
```

The "Commands" tag attribute "dispatchPort" is used to define the local port that the web-interface will use to connect to the **CalAmp** device communication server to indicate that a given command should be sent. The web-interface sends a command request to the **CalAmp** DCS, which then forwards the command to the **CalAmp** device based on the specified transport media (ie. UDP or SMS(email)).

The "Commands" sub-tag "AclName" is used to specify the ACL name used to allow/deny access to this device command feature for specific users. This ACL name should also be specified in the "private.xml" or "private_common.xml" file. To control specific commands, the ACL key defined in the "private.xml" or "private_common.xml" file, should be this ACL name, followed by a colon (":"), followed by the specific command name.

The "Command" sub-tags define the specific available commands. The "name" attribute specifies the command name. The "enabled" attribute specifies whether this command is enabled or disabled. Disabled commands will not be displayed.

The "Command" tag, "Type" sub-tag, specifies where this command should be made available for user selection. Valid values may be separated by commas, and may be "map" to specify that the command should be available on the Device Track Map page in a command pull-down selection menu, or "admin" to indicate that the command should be available on the "Properties" page which can be displayed from the Device Admin list page.

The "Command" tag, "Description" sub-tag, specifies the Description/Title that will be used to describe the command in the pull-down selection menu, or when displayed on the Device Admin Properties page.

The "Command" tag, "String" sub-tag, specifies the command which is to be sent to the **CalAmp** device. The "protocol" attribute specified the transport media used to send the command to the device. Acceptable values are wither "udp" or "sms" (sms commands will be sent to the SMS email gateway defined in the Device record). In the case of "sms" protocol/transport, the value of this tag is the actual command string which will be sent to the **CalAmp** device as a text message (assuming that an appropriate SMS gateway is in use).

The "Command" tag, "StatusCode" sub-tag, specifies the status-code of an audit event that should be generated when this command is sent to the device at the request of a user. If no status-code is specified, then no event will be generated.

3) Running the Server

To begin listening for incoming events the server must be started. This section describes the process for manually starting the **CalAmp** server, and how to set up the system to have it automatically start the **CalAmp** server on system reboot.

3.1) Manually Starting the Server

The command for manually starting the **CalAmp** server is as follows:

```
> cd $GTS_HOME
> bin/runserver.pl -s calamp
```

To start the **CalAmp** server with debug logging (useful when testing or debugging), the option "-debug" may be added to the command line.

The server will start, and logging information will be sent to the file "\$GTS_HOME/logs/calamp.log".

For debug purposes, it is sometimes useful to have the logging output sent directly to the console, instead of the log file. In this case the option "-i" can also be added to the command-line. When logging to the console, hit control-C to stop the server.

To view the running server, you can use the "bin/psjava" command:

```
> $GTS_HOME/bin/psjava
```

PID	Parent	L	User	Java class/jar
54639	(1)	1	opengts	org.apache.catalina.startup.Bootstrap
68936	(1)	1	opengts	/usr/local/GTS_2.4.4-B23/build/lib/calamp.jar

To stop the running **CalAmp** server, enter the following command:

```
> cd $GTS_HOME
> bin/runserver.pl -s calamp -kill
```

This will stop the running **CalAmp** server.

3.2) Automatically Starting the Server on System Reboot

The auto-start script for Fedora/CentOS is located at "\$GTS_HOME/bin/onboot/fedora/opengts", and should have been installed into the system directory "/etc/init.d" when the GTS was installed (other distributions of Linux will require other auto-startup scripts).

This startup script uses the configuration specified in the file "\$GTS_HOME/bin/serverList" to determine which device communication servers to start up when the system is rebooted. The line in this file that refers to the **CalAmp** DCS should appear similar to the following:

```
execServer "CalAmp" "calamp" "${option}" ""
```

The first quoted string contains the DCS description. The second quoted string contains the ID of the device communication server and must match a library jar file with the same name at "\$GTS_HOME/build/lib/calamp.jar". The third quoted string must contain the exact value "\${option}", which is used within the startup script to pass command-line arguments to the DCS startup code. The fourth quoted string is used to pass other optional default or constant arguments to the DCS startup code.

3.3) Monitoring the Log Files

When started, the **CalAmp** DCS will create the following output log files:

```
$GTS_HOME/logs/calamp.pid
```

This file contains the process-id (PID) of the **CalAmp** DCS execution process. This PID is used by the "-kill" option to terminate the running **CalAmp** DCS.

```
$GTS_HOME/logs/calamp.log
```

This log file is where all other logging information is placed regarding the receipt and parsing of data from the remote **CalAmp** tracking devices. As this file grows, it will be "rotated" into other log files timestamped as follows:

```
calamp.log.YYYYMMDDHHMMSS.log
```

Where "YYYYMMDDHHMMSS" represents the Year/Month/Day/Hour/Minutes/Seconds time that the file was trimmed and rotated.

The "calamp.log" file will reflect any current connection attempts from remote **CalAmp** tracking devices. As devices send their data to the server, the receipt of the incoming data packets, along with parsing results, will be displayed in this log file.

5) Testing a New Configured Device

This section describes the process for monitoring newly configured **CalAmp** devices that have been assigned to an Account/Device record.

5.1) Monitoring for Incoming Connections

The Account report "Last Known Device Location" can be used to display the last known location of a given device, which can also be used to determine whether any events have been received from a specific **CalAmp** device.

The "Server ID" field on the Device record will also indicate if a data packet has arrived from a specific **CalAmp** device and successfully assigned to the Device record.

If no indication on the Device reports, or "Server ID" field is evident, then the log file itself can be consulted for indications of incoming data packets from the **CalAmp** device. The information in the log file can indicate whether an IMEI number may not have been properly assigned, so the **CalAmp** DCS is unable to determine which Account/Device the incoming data packet belongs to.

5.2) Viewing the Unassigned Device Report

In the case where an **CalAmp** device is put into service without having been assigned to an Account/Device record, or where the ESN number was incorrectly entered in to the Device record, the **CalAmp** DCS may not know to which Account/Device the incoming data packet belongs.

When the **CalAmp** DCS cannot determine the ownership of an incoming data packet, it will place the ESN/IMEI and currently GPS location into the "UnassignedDevices" table. The "Unassigned Devices" report can be selected from the System Administrator login panel ("System Admin" tab, "SysAdmin Reports" menu option, "Unassigned Devices" report). This report will show the "Server ID" (**CalAmp**), "Unique ID" (ESN or IMEI number), and the last time data was received from this device. This information can be used to determine whether an ESN/IMEI number was ever assigned to an Account/Device record, or if an ESN/IMEI number was incorrectly assigned to an Account/Device (ie. transposed digits, etc).

Appendix)

A) Troubleshooting CalAmp Device Connection Issues

The following are frequently-asked-questions regarding commonly occurring connection issues.

Q: I've configured an **CalAmp** device to send data to the server, but have not received any data.

A: Monitor the "calamp.log" file for possible incoming connections from the device. If there is no indication that the server is receiving any communication from the remote device, the most common reasons to check include:

- Make sure device has a valid/active SIM card.
- Make sure the device has been programmed with the proper APN ("Access Point Name") configuration as specified by your wireless service provider.
- Make sure the device has been programmed with the proper host and port of your server.
- Make sure the server firewall allows incoming UDP/TCP connections on the specified port. If the server itself provides its own firewall, then check the firewall settings. On Linux, this is usually controlled by "iptables". The command to display the current iptables configuration is "iptables-save" (must be run as "root"). See "<http://www.faqs.org/docs/iptables/iptables-save.html>" for more information.

Q: I see data arriving for my device in the "calamp.log" file, but it is always the same event that is being sent over and over.

A: If this occurs for all configured/connected **CalAmp** devices, the problem is likely that returned UDP acknowledgments are not being returned properly to the device. The most likely reason for this is that your computer responds to more than one IP address, and the returned UDP packets are being sent from a different IP address than the one that the device is configured to send data to. This can be fixed by setting the "bindAddress" attribute in the "dcservers.xml" file in the GTS installation directory (then restart the **CalAmp** DCS). In some cases, the SIM card wireless service provider does not allow returned UDP packets to be sent from the server back to a device. In this case, it may be necessary to program the **CalAmp** devices to not require a return acknowledgment.

A: If this occurs for only one device (ie. other devices are reporting as expected), this is likely due to the GPS receiver's inability to obtain a new GPS fix, and the previous GPS fix is being resent. This usually means that the device is simply in an area where the GPS satellites cannot be seen (ie. Indoors, etc). On rare occasions, this can mean that the GPS antenna has become unplugged, or has been damaged.

Q: The received events have a valid latitude/longitude, but do not have an odometer value.

A: The **CalAmp** DCS property "estimateOdometer" allows enabling a calculated odometer value, based on the distance traveled between successive GPS points. To enable a calculated estimated odometer value, make sure this property is set to "true". Alternatively, the **CalAmp** can be programmed to include an odometer value in the data packet, calculated internally based on successive GPS locations.