GpsGate TrackerOne

Reference Design

Table of Contents

GpsGate TrackerOne....3
Introduction....3
GpsGate Protocol....3
TrackerOne use cases....4
Setup (mandatory feature)....4
Reset (mandatory feature)....5
Start tracking (mandatory feature)....6
Stop tracking (mandatory feature)....6
Poll position (mandatory feature)....7
TrackerOne activated event (optional feature)....7
Odometer (optional feature)....7
Testing....7
Help....7
Appendix 1 – NMEA Checksum....8
Appendix 2 - Command structure and flow....9
Error handling....9
Interrupted commands....10
Failed to execute command....10
Command not supported....10
Command flow....10
Authentication....11
Appendix 3 – Commands....12
_GprsSettings....12
_SendMessage....13
_StartTracking....14
_StopTracking....15
_PollPosition....15
_Ping....16
_DeviceReset....17
Appendix 4 – Transport....18
TCP/IP....18
SMS....18
UDP....18
HTTP GET....18
Appendix 5 – Status signals from device....19
Appendix 6 – GpsGate TrackerOne logo program....20

GpsGate TrackerOne

Introduction

GpsGate TrackerOne is a reference design for how a tracking unit interacts with GpsGate Server. TrackerOne can be implemented as a software client on a mobile phone, or similar mobile device. Or as a black box tracking device. The TrackerOne reference design is focused on tracker / server interaction and leaves the field open for how the actual software and hardware is designed and implemented.

TrackerOne defines the whole life span for a tracker. This includes configuration, how tracking is managed, how status signals are transferred. Continuous tracking, motion based and request based tracking is supported in the design.

TrackerOne defines the core “tracking engine” necessary in all personal, vehicle, and asset trackers. The protocol can be extended, and there is a lot of freedom to build on top of the core to extend and customize functionality. If GpsGate Server is used as backend, there is an open plugin architecture to support custom commands and features from a TrackerOne based device.

A TrackerOne device supports SMS and TCP/IP over for example GPRS. The tracker can either be optimised for mainly GPRS use, only SMS use or mixed GPRS/SMS use. A TrackerOne device has an IMEI or ESN number. TrackerOne uses a subset of GpsGate Protocol. UDP and HTTP can also optionally be supported, see appendix.

TrackerOne can be used for vehicle tracking, personal tracking, asset tracking or sport event tracking. TrackerOne defines the essential features necessary to work in those fields. The same hardware design will not function in all those segments, you will need to develop specific hardware designs for each segment. Some segments will require the optional features in TrackerOne for best performance.

A tracker implementing GpsGate TrackerOne can be used by the "QuickStart" feature on GpsGate.com and in GpsGate Server. This will enable a user to get started with a tracking device in less than a minute

GpsGate Protocol and GpsGate TrackerOne are free to use without any license fees. GpsGate is a trademark of GpsGate AB, Sweden.

GpsGate Protocol

GpsGate TrackerOne uses a subset of GpsGate Protocol (which is described in a separate document). This document will include all necessary parts of GpsGate Protocol necessary to implement a TrackerOne device.

When communicating over TCP/IP or SMS, GpsGate Protocol uses a “NMEA-like” style. And each sentence has a checksum (algorithm in Appendix 1) for HTTP and SMS checksum is optional. Command structure and flow between client and server is described in Appendix 2. Each individual command used in the GpsGate TrackerOne implementation is covered in Appendix 3.

In the “TrackerOne use cases” below mandatory and optional scenarios that a TrackerOne compliant tracker should support are covered.

TrackerOne use cases

In this section mandatory and optional scenarios that a TrackerOne compliant tracker should support are covered. In the samples below the data flowing between server and client is only outlined. Sometimes checksums are missing, sometimes specific arguments are missing. For strict specifications and samples with real data check the Appendix.

Setup (mandatory feature)

See Appendix for strict command syntax and flow.

GpsGate Server SMS to TrackerOne -->

```
$FRCMD,IMEI,_GprsSettings,,apn,,,online.gpsgate.com,30175
```

The sender phone number of this SMS is also the phone number the tracker should send SMS to. The tracker should store the sender phone number, and use it when sending SMS to the server. Only this command _GprsSettings will change the phone number the device sends SMS to.

The tracker can be reconfigured at any time. Knowing the phone number and IMEI of the tracker is considered to be enough authentication.

<-- TrackerOne to GpsGate Server GPRS or SMS

```
$FRRET,IMEI,_GprsSettings,,GpsGate TrackerOne,Super Tracker,1.31
```

Tracker tells which kind of tracker it is, in this case “GpsGate TrackerOne”, and it also tells its brand name and firmware version.

Tracker first tries to send reply over TCP/IP (e.g. GPRS/3G/CDMA).
If TCP/IP fails the answer is sent over SMS instead.

In this way the server can tell if the tracker can use TCP/IP or not and inform the user what to do to try to resolve this. Possible causes are typically SIM card subscription issues.

If the tracker finds some kind of error, and can send a message over TCP/IP or SMS it should send back a $FRERR. If the tracker cannot communicate with the server it should preferably inform the user by some LED or display on the tracker itself.

Optional - GpsGate Server 2.0.6 – The tracker can optionally specify of commands sent from the server should use SMS or TCP/IP only. By default the server will use TCP/IP if the tracker is online, and SMS if it isn't online. Add a “sms” at the response end to indicate that the server should only use SMS to send commands to the tracker. Add a “tcp” to only use TCP/IP.

```
$FRRET,IMEI,_GprsSettings,,GpsGate TrackerOne,Super Tracker,1.31,sms
```

<-- TrackerOne to GpsGate Server GPRS or SMS

```
$FRCMD,IMEI,_SendMessage,,latitude,hemi,longitude,hemi,alt,speed,heading,date,time
,valid
```

Tracker now enables GPS, and tries to get a fix for a maximum of 10 minutes. If successful the valid

position is sent to the server over TCP/IP or SMS depending on if TCP/IP is working or not.

If no fix is acquired an invalid position is sent.

After a successful _GprsSettings command the tracker should be considered to be reset. This means that both the server and tracker should consider all status variables to be zero or false. If the tracker wants to update the values, it can include those values in the above _SendMessage command.

--> GpsGate Server to TrackerOne

If TCP/IP is used the server answers

```
$FRRET,IMEI,_SendMessage
```

If SMS is used, the server will not respond. The server will not ACK SMS from a "GpsGate TrackerOne" device.

This completes the signup process.

The tracker will not start tracking and send information to the server. It will rather wait for a new command from the server to know what to do. Or some kind of user interaction on the device that triggers a report to the server.

If the tracker is online using TCP/IP any commands will be sent over TCP/IP. If not online, commands will be sent over SMS. This means that a device may try to keep an idle GPRS connection just to save SMS costs. The device may at any time send a $FRCMD,IMEI,_Ping command to keep the GPRS connection alive. If there is no response, the tracker could try to reconnect. In most GPRS networks you need to send some data at least every 3 minute to keep the connection alive. The response over TCP/IP from the server will be $FRRET,IMEI,_Ping

_Ping can also be sent over SMS, but should be avoided. The server will answer to _Ping both over SMS and TCP/IP.

Reset (mandatory feature)

See Appendix for strict command syntax and flow.

When tracker restarts – cycles its power or is reset by some reset button on the hardware – it should send a _Reset command to the server.

All status signals should be considered to be zero or false after a _Reset. All analog values are considered to be zero, all switches to be false, and all buttons unpressed. To change this status the device needs to resend any status values.

--> TrackerOne to GpsGate Server (TCP/IP or SMS)

```
$FRCMD,IMEI,_DeviceReset
```

<-- GpsGate Server to TrackerOne (TCP/IP or SMS)

```
$FRRET,IMEI,_DeviceReset
```

Start tracking (mandatory feature)

See Appendix for strict command syntax and flow.

--> GpsGate Server to TrackerOne (TCP/IP or SMS)

```
$FRCMD,IMEI,_StartTracking,TimeFilter=60,SmsTimeFilter=0,Motion=1
```

If “Motion” is set to 1 the tracker will only send reports to the server if moved. The tracker should be moved at least 50 meters. The motion feature can typically be improved by implementing a motion sensor in the tracker, but don't send a message to the server each time the motion sensor is activated, since it will generate unnecessary traffic.

If TCP/IP isn't available, SMS is used for reports using the interval SmsTimeFilter. If SmsTimeFilter is omitted (left out) or set to 0, then SMS should not be used for continuous tracking.

If the device is online this command is sent over TCP/IP. The device adjusts to the new tracking rules.

<-- TrackerOne to GpsGate Server (over SMS or TCP/IP)

```
$FRRET,IMEI,_StartTracking
$FRCMD,IMEI,_SendMessage,...,Motion=1
$FRCMD,IMEI,_SendMessage,...
$FRCMD,IMEI,_SendMessage,...,SosButton=1
```

If _StartTracking isn't sent the tracker should only send a report on direct user interaction with the device, such as a button pressed or an electrical input signal activated etc.

--> GpsGate Server to TrackerOne.

```
$FRRET,IMEI,_SendMessage
```

For each _SendMessage sent to server it will respond if TCP/IP is used. For SMS no response is sent from server.

Stop tracking (mandatory feature)

See Appendix for strict command syntax and flow.

--> GpsGate Server to TrackerOne (TCP/IP or SMS)

```
$FRCMD,IMEI,_StopTracking
```

If tracker is online using TCP/IP this command is sent over TCP/IP. Otherwise the command is sent over SMS.

<-- TrackerOne to GpsGate Server (TCP/IP or SMS)

```
$FRRET,IMEI,_StopTracking
```

Poll position (mandatory feature)

_PollPosition can be executed while _StartTracking is active.

--> GpsGate Server to TrackerOne (TCP/IP or SMS)

```
$FRCMD,IMEI,_PollPosition
```

<-- TrackerOne to GpsGate Server (TCP/IP or SMS)

Tracker will try to send a position update back to server. If it has no fix, it will try to acquire a fix for 5 minutes. If not acquired an invalid position update is sent to server.

```
$FRRET,IMEI,_PollPosition,lon,lat,alt,...
```

TrackerOne activated event (optional feature)

The TrackerOne device may have buttons or other sensors on the device itself that trigger reports to be sent to the server. This can for example be a SOS button, Motion sensor, Parking mode button, Digital/Analog inputs or Temperature sensors. All of those are optional.

SosButton is mapped to the SOS channel.

See Appendix 4 for a list of custom signals the tracker can send to the server.

Odometer (optional feature)

_SendMessage has an optional Odometer variable. If used this value can be mapped to the odometer feature on GpsGate Server to get exact distance reports.

Testing

Download and install GpsGate Server here: http://gpsgate.com/download

Monitor device to server communication using the Terminal:
http://forum.gpsgate.com/topic.asp?TOPIC_ID=10740

Help

GpsGate Server Developer's Guide
http://forum.gpsgate.com/topic.asp?TOPIC_ID=8097

Help forum: http://forum.gpsgate.com

Email:support@gpsgate.com

Appendix 1 – NMEA Checksum

C++

```
#define NIBBLE2HEX(c) ((c) > 9 ? (c) + 'A' - 10 : (c) + '0')

// buf is a char[] array which contains the NMEA sentence without trailing
checksum.
// E.g. "$FRLIN,,user1,8IVHF" and "*7A" will be added

// buf_inx is an index to the last free position in the buffer

int checksum = 0;
int inx;

for(inx = 1; inx < buf_inx; inx++)
{
      checksum ^= buf[inx];
}

buf[buf_inx++] = '*';
buf[buf_inx++] = NIBBLE2HEX((checksum >> 4) & 0xf);
buf[buf_inx++] = NIBBLE2HEX(checksum & 0xf);
```

Appendix 2 - Command structure and flow

This structure is used when sending messages to execute commands. Messages can be sent from server to client and client to server. Several commands can be executed between client and server in one session.

Command Sentences

Syntax:

```
$FRCMD,IMEI,command,Inline,param1,param2,...,paramN*XX
```

IMEI	The trackers 15 or 16 digit IMEI number. Used for authentication. (GpsGate Protocol also includes other means for authentication which are not covered in this document.)
command	the command to be executed. E.g. "_StartTracking".
Inline	Argument type. Can be left empty to save space. For custom commands “Nmea” style arguments can be used as well. See main “GpsGate Protocol” specifications.
param1 - N	Parameters for the command.
XX	NMEA checksum. Algorithm to calculate shown in Appendix 1

Example:

```
$FRCMD,353857014816785,_StartTracking,,TimeFilter=30,SmsTimeFilter=60,Motion=1*2B
```

Return values

A return value is sent as a reply to an executed command.

Syntax:

```
$FRCMD,IMEI,command,Inline,param1,param2,...,paramN*XX
```

IMEI	The trackers 15 or 16 digit IMEI number. Used for authentication.
command	the command to be executed. E.g. "_StartTracking".
Inline	indicates that the rest of the fields in the sentence are arguments to the command. Can be left empty to save space.
param1 - N	Parameters to the command
XX	NMEA checksum. Algorithm to calculate shown in Appendix 1

```
$FRRET,353857014816785,_StopTracking*20
```

Error handling

If a command for some reason cannot be executed FRERR is returned and not FRRET. The reason can be that the there is some kind of execution error, the command is not supported, or there is a protocol

error. When FRERR is returned the client should assume no data has been affected on the server. The protocol is transaction based.

In all cases except when there is a protocol or connection error the exchange of commands can continue between the server and the client.

Interrupted commands

If several FRCMD is sent without the previous one being finished, the peer executing the command should roll back the current command and start executing the new command. Example: If two FRCMD is received in a row, the last FRCMD should be executed.

Failed to execute command

If a FRCMD fails to execute a FRERR sentence with code “CannotExecute” is returned. The client is free to execute a new command after receiving FRERR.

Sample communication:

If the client tries to save data to a track recorder which doesn't belong to the logged in user.

Client	Server
`$FRCMD,,_SaveTrackData,Nmea,2,2*54`	
	`$FRERR,CannotExecute,TrackRecorder not found for user*2B`

Command not supported

A client or server that implements v1.1 of the GpsGate protocol does not need to implement any of the commands (see blow for a list of supported commands) described in this document. For commands that aren't supported a FRERR with code “NotSupported”. The client is free to execute a new command after receiving this error message.

Sample communication:

The command “_dummy” is not supported by the peer (in this case the server).

Client	Server
`$FRCMD,,_dummy,Inline*6C`	
	`$FRERR,NotSupported,_dummy not supported*66`

Command flow

A command can be executed in either direction. From server to client and from client to server. Only one command can be active in each direction at the same time.

NOTE! Consider the scenario when the client and server sends a new command at the same time. This scenario must be handled.

Commands can be executed over any transport layer. In the case of TrackerOne, SMS and optionally GPRS should be supported. Use GPRS when available since it is cheaper under most circumstances.

NOTE! A very important scenario to consider is that a command request ($FRCMD) might be sent over one transport layer (e.g. SMS) and the response might come over another transport layer (e.g.

GPRS). The typical case is a _PollPosition or _StartTracking command sent from server where the tracker responds over GPRS. Another example is _GprsSettings which the server sends over SMS, and which the tracker should answer to over GPRS if available.

Since SMS can be expensive, the tracker can keep a connection to the server open over GPRS. In this case the tracker should send _Ping commands over the connection with some minutes interval. The server will respond to each _Ping. If there is an open TCP/IP connection the server will use this connection rather than SMS when sending commands to the tracker.

$FRCMD is always responded to by $FRRET or $FRERR.

NOTE! This is with one exception. _SendMessage sent over SMS from the tracker will not be acknowledged by the server over SMS. However over GPRS it will be acknowledged.

Authentication

A device is authenticated by its phone number in combination with IMEI number. The server is authenticated by knowing the device's phone number and IMEI.

Over SMS $FRERR is authenticated by phone number only and IMEI is not included.

Appendix 3 – Commands

_GprsSettings

Used by server to send out new communication settings to the tracker. The tracker should always be ready to receive this command.

This command is always sent over SMS. The tracker should save the phone number this SMS was sent from. When the tracker sends commands over SMS to the server it should use this phone number.

All status signals are considered to be reset after a _GprsSettings command.

Syntax command from server to tracker:

$FRCMD,IMEI,_GprsSettings,,APN,APN_user,APN_pw,hostname,port*XX

`IMEI`	The trackers 15 or 16 digit IMEI number. Used for authentication. (GpsGate Protocol also includes other means for authentication which are not covered in this document.)
`_GprsSettings`	Name of command. In this case "_GprsSettings".
`Inline`	Argument type. Can be left empty to save space. For custom commands "Nmea" style arguments can be used as well. See main "GpsGate Protocol" specifications.
`APN`	GPRS APN. E.g. "online.telia.com"
`APN_user`	APN username. Often kept empty.
`APN_pw`	APN password. Often kept empty.
`Hostname`	GpsGate Server address. The tracker must support DNS lookups.
`Port`	TCP/IP port to connect to to send commands to server.
`XX`	NMEA checksum. Algorithm to calculate shown in Appendix 1

Syntax respons from tracker to server:

$FRRET,IMEI,_GprsSettings,,GpsGate TrackerOne,Name,version,transport*7A

`IMEI`	The trackers 15 or 16 digit IMEI number. Used for authentication. (GpsGate Protocol also includes other means for authentication which are not covered in this document.)
`_GprsSettings`	Name of command. In this case "_GprsSettings".
`Inline`	Argument type. Can be left empty to save space. For custom commands "Nmea" style arguments can be used as well. See main "GpsGate Protocol" specifications.
`Definition`	Should always be "GpsGate TrackerOne"

Name	Device name.
Version	Tracker firmware version in the format major.minor.revision.build e.g. 1.2.2.3 or 1.21
Transport	Optional. Can be “sms” or “tcp”. If specified the server will only use specified transport to send outgoing commands from server to tracker. By default (if this field is left out or left balnk, the server will use TCP/IP if the tracker is online, else SMS is used.)
XX	NMEA checksum. Algorithm to calculate shown in Appendix 1 Optional for SMS and HTTP.

Sample
From server to tracker:

$FRCMD,353857014816785,_GprsSettings,,online.telia.com,guest,pw,online.gpsgate.com,30175*6C

From tracker to server:
$FRRET,353857014816785,_GprsSettings,,GpsGate TrackerOne,TestDevice,0.01*7A

_SendMessage

_SendMessage is sent by the tracker when it should send a position according to the update rules sent to it by _StartTracking, or if the is any event on the tracker, like a button is being pressed, or a motion sensor is activated. Any number of status signals can be added to the end of _SendMessage.

Syntax

$FRCMD,IMEI,_SendMessage,,DDMM.mmmm,N,DDMM.mmmm,E,AA.a,SSS.ss,HHH.h,DDMMYY,hhmmss.dd,valid,var1=value,var2=value...*XX

DDMM.mmmm	Latitude (NMEA format)
<N|S>	Hemisphere N or S
DDMM.mmmm	Longitude DDMM.mmmm (NMEA format)
<E|W>	Hemisphere E or W
AA.a	Altitude in meters above sea level
SSS.ss	Speed over ground in knots
HHH.h	Heading over ground in degrees
DDMMYY	date
hhmmss.dd	Time (UTC)
valid	1 if a vlid fix. 0 if not a valid fix.

`var1=value`	Status signals. See Appendix 4 for a list of signal names and possible values.
`XX`	NMEA checksum. Algorithm to calculate shown in Appendix 1 Optional for SMS and HTTP.

Sample

Send from tracker
$FRCMD,353857014816785,_SendMessage,,4748.00000,N,3430.00000,W,34.6,234,90,041108,155059.411,1,SosButton=1*27

Reply from server if SMS is used as transport.
In case SMS is used as transport, there is no reply from server.

Reply from server if GPRS is used as transport.
$FRRET,353857014816785,_SendMessage*40

Example of position update without status signal:
$FRCMD,353857014816785,_SendMessage,,4748.00000,N,3530.00000,E,34.6,234,90,170109,120523.657,1*65

_StartTracking

This command is sent from server to tracker when the server wants to set up new tracking rules for the tracker.

Syntax

$FRCMD,IMEI,_StartTracking,,Rule1=value1,Rule2=value2,...*XX

`IMEI`	The trackers 15 or 16 digit IMEI number. Used for authentication. (GpsGate Protocol also includes other means for authentication which are not covered in this document.)
`_StartTracking`	Name of command.

`Rule1=value1`	A set of rules the tracker is recommended to follow when sending _SendMessage commands sent to the server. The following three are included in the TrackerOne definition.
	`TimeFilter`
	Number of seconds between each _SendMessage command sent to server. E.g. “TimeFilter=60” means a _SendMessage each minute.
	`SmsTimeFilter`
	Number of seconds between each _SendMessage command sent to server if GPRS is unavailable and SMS is used. **If SmsTimeFilter is omitted or set to 0, no _SendMessage commands should be sent over SMS.** E.g. “SmsTimeFilter=3600” sends one _SendMessage command over SMS each hour if GPRS is unavailable.

	Motion If set to 1 the tracker should only send _SendMessage commands if the tracker is moving. A tracker can typically be in sleep mode and then activated by a motion sensor. The tracker should NOT send any command to the server until it has moved at least 50 meters to avoid false motion events. E.g. “Motion=1”
XX	NMEA checksum. Algorithm to calculate shown in Appendix 1

Sample

Server to tracker:

$FRCMD,353857014816785,_StartTracking,,TimeFilter=30,SmsTimeFilter=60,Motion=1*2B

Tracker to server:

$FRRET,353857014816785,_StartTracking*58

The tracker will now continue to send _SendMessage commands to the server according to the rules set by _StartTracker. _StartTracker can be sent again at any time, and NOT necessarily with a _StopTracking in between.

_StopTracking

This command is sent from the server to the tracker to stop the tracker from sending _SendMessage commands to the server.

Syntax

$FRCMD,IMEI,_StopTracking*XX

IMEI	The trackers 15 or 16 digit IMEI number. Used for authentication. (GpsGate Protocol also includes other means for authentication which are not covered in this document.)
_StopTracking	Name of command.
XX	NMEA checksum. Algorithm to calculate shown in Appendix 1

Sample
Server to tracker:
$FRCMD,353857014816785,_StopTracking*29

Tracker to server
$FRRET,353857014816785,_StopTracking*20

_PollPosition

This command is sent to the server to the tracker when the server wants one single position update, with optional included status signals.

Syntax from server

$FRCMD,IMEI,_PollPosition*38

Syntax reply from tracker

$FRRET,IMEI,_PollPosition,,DDMM.mmmm,N,DDMM.mmmm,E,AA.a,SSS.ss,HHH.h,DDMMYY,hhmmss.dd,valid,var1=value1,var2=value2...*XX

DDMM.mmmm	Latitude (NMEA format)
<N|S>	Hemisphere N or S
DDMM.mmmm	Longitude DDMM.mmmm (NMEA format)
<E|W>	Hemisphere E or W
AA.a	Altitude in meters above sea level
SSS.ss	Speed over ground in knots
HHH.h	Heading over ground in degrees
DDMMYY	date
hhmmss.dd	Time (UTC)
Valid	0 if not a valid fix. 1 if a vlid fix.
var1=value1	Status signals. See Appendix 4 for a list of signal names and possible values.
XX	NMEA checksum. Algorithm to calculate shown in Appendix 1

Sample
From server to tracker:
$FRCMD,353857014816785,_PollPosition*38

From tracker to server:
$FRRET,353857014816785,_PollPosition,,4748.00000,N,3430.00000,W,34.6,234,90,041108,163006.161,1*09

_Ping

This command is sent from tracker to the server. The server will not send _Ping commands to the tracker.

This command has two main purposes. The first is for simple testing. A _Ping command to the server will test if a connection works. The server will always respond. The _Ping command can be executed over SMS as well as GPRS (TCP/IP).

Syntax

$FRCMD,IMEI,_Ping*XX

IMEI	The trackers 15 or 16 digit IMEI number. Used for authentication. (GpsGate Protocol also includes other means for authentication which are not covered in this document.)

`_Ping`	Name of command.
`XX`	NMEA checksum. Algorithm to calculate shown in Appendix 1 Optional for SMS and HTTP.

Sample
From tracker to server.
$FRCMD,353857014816785,_Ping*0E

Response from server
$FRRET,353857014816785,_Ping*07

_DeviceReset

Should always be sent from tracker to server after power has been cycled (turn off/on) or if the device has been reset for some reason.

After a reset all status signals are considered to be false or zero.

Syntax

$FRCMD,IMEI,_DeviceReset*XX

`IMEI`	The trackers 15 or 16 digit IMEI number. Used for authentication. (GpsGate Protocol also includes other means for authentication which are not covered in this document.)
`_DeviceReset`	Name of command.
`XX`	NMEA checksum. Algorithm to calculate shown in Appendix 1

Sample
From tracker to server.
$FRCMD,353857014816785,_DeviceReset*6B

Response from server
$FRRET,353857014816785,_DeviceReset*62

Appendix 4 – Transport

Data can be sent from the tracker to the server using TCP/IP, SMS, UDP or HTTP.

TCP/IP

Uses port 30175 by default. Outgoing commands from the server are sent over TCP/IP if the tracker is online. GpsGate Server can always receive commands from the tracker over TCP/IP.

Checksum is required for commands sent over TCP/IP.

SMS

SMS must be supported by a TrackerOne device. Outgoing commands uses SMS if the tracker isn't online. A command cannot span several SMS.

GpsGate Server 2.0.6 - SMS commands from the tracker to the server *do not need any checksum*. Both commands with and without checksum is accepted by the server.

UDP

GpsGate Server 2.0.6 - Uses port 30175 by default. Outgoing commands from server to tracker are not supported over UDP. GpsGate Server can always receive commands from the tracker over UDP.

Checksum is required for commands sent over UDP.

HTTP GET

Supported in GpsGate Server 2.0.6 and later. By default port 8008 is used. Only commands from the tracker to server can use HTTP. The respons body contans the server reply ($FRRET).

Commands sent over HTTP *do not need any checksum*. Both commands with and without checksum is accepted by the server.

Syntax:

http://hostname:8008/GpsGate/?cmd=$FRCMD...

Example:

http://online.gpsgate.com:8008/GpsGate/?cmd=$FRCMD,353857014816785,_SendMessage,,4748.00000,N,3530.00000,E,34.6,234,90,170109,120523.657,1

Appendix 5 – Status signals from device

TrackerOne has a set of custom signals. Those can be mapped on the server to suitable functionality. Add status signals (variables) to the end of _SendMessage commands sent to the server.

SosButton (boolean button, optional) – this signal is mapped to the SOS event on the server. E.g. SosButton=1

Motion (boolean button, optional) - this signal should be included if the tracker starts tracking based on motion. On the server this signal can optionally be connected to alarms of various kinds. E.g. Motion=1

Odometer (double, optional) – if the tracker supports odometer (distance calculation), use this signal to send it to server. The value can either be absolute or a delta value from last odometer report. The Odometer signal is scaled on the server. E.g. “Odometer=1023445.3”

BatteryLow (boolean button, mandatory) – sent to server when battery is low. E.g. “BatteryLow=1”

Button1 (boolean button, optional) – A custom signal to be mapped on the server.

Button2 (boolean button, optional) – A custom signal to be mapped on the server.

Button3 (boolean button, optional) – A custom signal to be mapped on the server.

Button4 (boolean button, optional) – A custom signal to be mapped on the server.

Switch1 (boolean switch, optional) – A custom signal to be mapped on the server. E.g. “Switch1=1”

Switch2 (boolean switch, optional) – A custom signal to be mapped on the server.

Switch3 (boolean switch, optional) – A custom signal to be mapped on the server.

Switch4 (boolean switch, optional) – A custom signal to be mapped on the server.

Analog1 (double, optional) – A custom signal to be mapped on the server. E.g. “Analog1=45.7”

Analog2 (double, optional) – A custom signal to be mapped on the server.

Analog3 (double, optional) – A custom signal to be mapped on the server.

Analog4 (double, optional) – A custom signal to be mapped on the server.

Appendix 6 – GpsGate TrackerOne logo program

Please contact support@gpsgate.com for more information.