VComm Signal Quality Service Specification

SimPhonics, Inc.
3226 North Falkenburg Road
Tampa, Florida 33619
Voice (877) 205-4901 X102
FAX (813) 623-5119
CAGE: 0L4C8
Email: info@simphonics.com

Table of Contents

List of Tables

1 Preface

This document defines the VComm Signal Quality Service protocol.

1.1 Trademarks and Copyrights

Any trademarks shown throughout this document are the property of their respective owners. V+ is a trademark of SimPhonics, Incorporated.

1.2 Revision History

Version	Revision	Date
1.0	Initial Release	August 24, 2009
2.0	Build 100 messages made obsolete	May 13, 2010
2.1	Corrected byte order to be little endian	October 27, 2011
2.2	Added additional status value.	March 25, 2014

This document was authored using Microsoft Word 2007 and may be copied freely for any purpose.

For more information on this product, go to the following:
http://www.simphonics.com

1.3 Before Reading This Document

The reader should be familiar with the Department of Defense DIS (Distributed Interactive Simulation) and HLA (High Level Architecture) specifications. A basic understanding of networking principles is important, as well as a good understanding of the V+ Visual Programming System and VComm. It is assumed that the reader also has Administrator privileges and knows basic Windows system administration.

2 Signal Quality Service

The concept of a Signal Quality Service evolved from the need to incorporate external factors into the signal quality calculations of VComm.

By default, VComm determines the signal quality of received radio communications using built in line of sight and signal quality calculations. Line of sight calculations are based on a round earth model while signal quality calculations are based on standard RF propagation equations. In particular, these calculations do not take into account terrain, weather effects, and other possible sources of RF interference. To incorporate these aspects into the VComm signal quality calculations, VComm uses a Signal Quality Service.

The Signal Quality Service is a separate process that communicates with VComm according to the remote message protocol defined within this document. Any process that is compliant with the protocol may be used as a signal quality service. The process must act as the server while VComm acts as a client. The VComm Terrain Server (VTS) is an example of a server that is a Signal Quality Service. It provides VComm with signal attenuation data based on terrain.

This Signal Quality Service specification defines all the messages that make up the remote message protocol of the Signal Quality Service. Servers and clients that are compliant with the remote message protocol do not need to support all messages. For example, VComm and VTS only use a subset of the available messages. VTS, for example, only supports those messages that are relevant to a terrain server. A server or client that claims compliance with the remote message protocol must publish which messages it supports. For VComm and VTS this support is documented in sections 4 and 5 respectively.

3 Remote Message Protocol Definition

The Remote Message Protocol is TCP/IP. Messages are defined in terms of "Gets" and "Sets". The client side of the TCP/IP connection sends "Get" messages while the server side of the TCP/IP connection sends "Set" messages.

The messages are categorized as either Build 100 or Build 200 messages. Build 100 messages are legacy messages and are obsolete. Build 100 messages have an ID of less than 200.

Table 1 identifies the obsolete Build 100 messages. These should not be used for new applications.

Table 2 identifies the Build 200 messages.

In general, there are three forms of each message with each one using one of the following three coordinate systems:

1. Geocentric (X, Y, Z)
2. Geodetic (Latitude, Longitude, Altitude)
3. Universal Transverse Mercator (UTM – Easting, Northing, Altitude, Zone, Hemisphere)

Each message definition in Table 1 and Table 2 identifies which of these coordinate systems is used.

Table 1. Remote Control Message Identification (Build 100 - Obsolete)

ID	Name	Description
1	Get Antenna Pair Loss (Geocentric)	Sent by a client to the server as a request for the amount of attenuation and separation between two antennas using **Geocentric** coordinates.
2	Get Antenna Pair Loss (Geodetic)	Sent by a client to the server as a request for the amount of attenuation and separation between two antennas using **Geodetic** coordinates.
3	Get Antenna Pair Loss (UTM)	Sent by a client to the server as a request for the amount of attenuation and separation between two antennas using **UTM** coordinates.
4	Get Height Above Terrain (Geocentric)	Sent by a client to the server as a request for the height above terrain for a given earth point using **Geocentric** coordinates.
5	Get Height Above Terrain (Geodetic)	Sent by a client to the server as a request for the height above terrain for a given earth point using **Geodetic** coordinates.
6	Get Height Above Terrain (UTM)	Sent by a client to the server as a request for the height above terrain for a given earth point using **UTM** coordinates.
101	Set Antenna Pair Loss (Geocentric)	Sent by the server to a client in response to a Get Antenna Pair request using **Geocentric** coordinates.
102	Set Antenna Pair Loss (Geodetic)	Sent by the server to a client in response to a Get Antenna Pair request using **Geodetic** coordinates.
103	Set Antenna Pair Loss (UTM)	Sent by the server to a client in response to a Get Antenna Pair request using **UTM** coordinates.
104	Set Height Above Terrain (Geocentric)	Sent by the server to a client in response to a Get Height Above Terrain request using **Geocentric** coordinates.
105	Set Height Above Terrain (Geodetic)	Sent by the server to a client in response to a Get Height Above Terrain request using **Geodetic** coordinates.
106	Set Height Above Terrain (UTM)	Sent by the server to a client in response to a Get Height Above Terrain request using **UTM** coordinates.

Table 2. Remote Control Message Identification (Build 200)

ID	Name	Description
201	Get Communications Radio Pair Signal Strength (Geocentric)	Sent by a client to the server as a request for the signal and noise strength between two communication radio antennas using **Geocentric** coordinates.
202	Get Communications Radio Pair Signal Strength (Geodetic)	Sent by a client to the server as a request for the signal and noise strength between two communication radio antennas using **Geodetic** coordinates.
203	Get Communications Radio Pair Signal Strength (UTM)	Sent by a client to the server as a request for the signal and noise strength between two communication radio antennas using **UTM** coordinates.
211	Get Antenna Pair Loss (Geocentric) with Exercise ID	Sent by a client to the server as a request for the amount of attenuation and separation between two antennas using **Geocentric** coordinates.
212	Get Antenna Pair Loss (Geodetic) with Exercise ID	Sent by a client to the server as a request for the amount of attenuation and separation between two antennas using **Geodetic** coordinates.
213	Get Antenna Pair Loss (UTM) with Exercise ID	Sent by a client to the server as a request for the amount of attenuation and separation between two antennas using **UTM** coordinates.
214	Get Height Above Terrain (Geocentric) with Exercise ID	Sent by a client to the server as a request for the height above terrain for a given earth point using **Geocentric** coordinates.
215	Get Height Above Terrain (Geodetic) with Exercise ID	Sent by a client to the server as a request for the height above terrain for a given earth point using **Geodetic** coordinates.
216	Get Height Above Terrain (UTM) with Exercise ID	Sent by a client to the server as a request for the height above terrain for a given earth point using **UTM** coordinates.
301	Set Communications Radio Pair Signal Strength (Geocentric)	Sent by the server to a client in response to a Get Communications Radio Pair Signal Strength request using **Geocentric** coordinates.
302	Set Communications Radio Pair Signal Strength (Geodetic)	Sent by the server to a client in response to a Get Communications Radio Pair Signal Strength request using **Geodetic** coordinates.
303	Set Communications Radio Pair Signal Strength (UTM)	Sent by the server to a client in response to a Get Communications Radio Pair Signal Strength request using **UTM** coordinates.
311	Set Antenna Pair Loss (Geocentric) with Exercise ID	Sent by the server to a client in response to a Get Antenna Pair request using **Geocentric** coordinates.
312	Set Antenna Pair Loss (Geodetic) with Exercise ID	Sent by the server to a client in response to a Get Antenna Pair request using **Geodetic** coordinates.
313	Set Antenna Pair Loss (UTM) with Exercise ID	Sent by the server to a client in response to a Get Antenna Pair request using **UTM** coordinates.
314	Set Height Above Terrain (Geocentric) with Exercise ID	Sent by the server to a client in response to a Get Height Above Terrain request using **Geocentric** coordinates.

Table 2. Remote Control Message Identification (Build 200)

ID	Name	Description
315	Set Height Above Terrain (Geodetic) with Exercise ID	Sent by the server to a client in response to a Get Height Above Terrain request using **Geodetic** coordinates.
316	Set Height Above Terrain (UTM) with Exercise ID	Sent by the server to a client in response to a Get Height Above Terrain request using **UTM** coordinates.

SimPhonics uses the following enumeration for defining the message identifiers. These enumerated values are used in the message descriptions that follow.

```
typedef enum
{
   VTS_NO_MSG                          =   0,

   //
   // Build 100 messages
   //
   VTS_GET_ANT_PAIR_LOSS_GEOCENTRIC    =   1,
   VTS_GET_ANT_PAIR_LOSS_GEODETIC      =   2,
   VTS_GET_ANT_PAIR_LOSS_UTM           =   3,
   VTS_GET_HEIGHT_GEOCENTRIC           =   4,
   VTS_GET_HEIGHT_GEODETIC             =   5,
   VTS_GET_HEIGHT_UTM                  =   6,

   VTS_SET_ANT_PAIR_LOSS_GEOCENTRIC    = 101,
   VTS_SET_ANT_PAIR_LOSS_GEODETIC      = 102,
   VTS_SET_ANT_PAIR_LOSS_UTM           = 103,
   VTS_SET_HEIGHT_GEOCENTRIC           = 104,
   VTS_SET_HEIGHT_GEODETIC             = 105,
   VTS_SET_HEIGHT_UTM                  = 106,

   //
   // Build 200 messages
   //
   VTS_GET_COMM_RADIO_SIG_STRENGTH_GEOCENTRIC = 201,
   VTS_GET_COMM_RADIO_SIG_STRENGTH_GEODETIC   = 202,
   VTS_GET_COMM_RADIO_SIG_STRENGTH_UTM        = 203,

   VTS_WITHEX_GET_ANT_PAIR_LOSS_GEOCENTRIC    = 211,
   VTS_WITHEX_GET_ANT_PAIR_LOSS_GEODETIC      = 212,
   VTS_WITHEX_GET_ANT_PAIR_LOSS_UTM           = 213,
   VTS_WITHEX_GET_HEIGHT_GEOCENTRIC           = 214,
   VTS_WITHEX_GET_HEIGHT_GEODETIC             = 215,
   VTS_WITHEX_GET_HEIGHT_UTM                  = 216,

   VTS_SET_COMM_RADIO_SIG_STRENGTH_GEOCENTRIC = 301,
   VTS_SET_COMM_RADIO_SIG_STRENGTH_GEODETIC   = 302,
   VTS_SET_COMM_RADIO_SIG_STRENGTH_UTM        = 303,

   VTS_WITHEX_SET_ANT_PAIR_LOSS_GEOCENTRIC    = 311,
   VTS_WITHEX_SET_ANT_PAIR_LOSS_GEODETIC      = 312,
   VTS_WITHEX_SET_ANT_PAIR_LOSS_UTM           = 313,
   VTS_WITHEX_SET_HEIGHT_GEOCENTRIC           = 314,
   VTS_WITHEX_SET_HEIGHT_GEODETIC             = 315,
   VTS_WITHEX_SET_HEIGHT_UTM                  = 316

} VTSRemoteMessageIDs;
```

3.1 Message Descriptions and Use

All messages are constructed with data with types as described in Table 3. The Integer and Float data transferred over the network is in Little Endian byte order.

Table 3. Message Element Data Types

Name	Description
Byte	8-bit signed integer
UByte	8-bit unsigned integer
Word	16-bit signed integer
UWord	16-bit unsigned integer
Int	32-bit signed integer
UInt	32-bit unsigned integer
DInt	64-bit signed integer
DUInt	64-bit unsigned integer
Float	32-bit floating point number
DFloat	64-bit floating point number
Char(xxx)	Character string sized at xxx

The following enumeration is used within the Hemisphere fields of messages using UTM coordinates. Use these values when sending a UTM based message.

```
typedef enum
{
   VTS_HEMI_NORTH  = -2,
   VTS_HEMI_WEST   = -1,
   VTS_HEMI_CENTER =  0,
   VTS_HEMI_EAST   =  1,
   VTS_HEMI_SOUTH  =  2

} VTSHemisphere;
```

The precise format for each message is described in the subsections that follow. The **Offset** columns for each of the messages is exactly as shown. This means that the 64-bit types are not forcibly aligned to 8-byte boundaries.

3.1.1 Get Antenna Pair Loss (Geocentric)

This message is generated by a client as a request for the amount of attenuation and separation between two antennas in Geocentric coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 4.

Table 4. Get Antenna Pair Loss (Geocentric) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **1** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Frequency	Frequency of the radio transmission in units Hz.
20	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
28	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
36	DFloat	Antenna 1 X	X in meters.
44	DFloat	Antenna 1 Y	Y in meters.
52	DFloat	Antenna 1 Z	Z in meters.
60	DFloat	Antenna 2 X	X in meters.
68	DFloat	Antenna 2 Y	Y in meters.
76	DFloat	Antenna 2 Z	Z in meters.
Packet Length: 84 bytes			

Note: *The Antenna 1 ID and Antenna 2 ID elements are 64-bit unsigned integers to accommodate a DIS Entity ID which is a structure of 4 16-bit unsigned integers. The example below shows how one might move DIS Entity IDs into this message structure.*

The server is required to respond to this message with a **Set Antenna Pair Loss (Geocentric)** message.

Here's an example of what this message might look like if it were written in C/C++.

```
typedef struct _EntityID_
{
  USHORT   wSite;
  USHORT   wHost;
  USHORT   wEntity;
  USHORT   wRadio;
} EntityID;


typedef struct _GetAntPairLossGeocentric_
{
  UINT      msgId;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  double    ant1_X;
  double    ant1_Y;
  double    ant1_Z;
  double    ant2_X;
  double    ant2_Y;
  double    ant2_Z;
} GetAntPairLossGeocentric;

EntityID                 ant1;
EntityID                 ant2;
GetAntPairLossGeocentric getAntPair;

ant1.wSite   = 1;
ant1.wHost   = 2;
ant1.wEntity = 3;
ant1.wRadio  = 4;
ant2.wSite   = 1;
ant2.wHost   = 2;
ant2.wEntity = 3;
ant2.wRadio  = 5;

memcpy (&getAntPair.ant1_Id, &ant1, sizeof(EntityID));
memcpy (&getAntPair.ant2_Id, &ant2, sizeof(EntityID));
getAntPair.msgId     = VTS_GET_ANT_PAIR_LOSS_GEOCENTRIC;
getAntPair.timestamp = 1;
getAntPair.freq      = 108125000;
getAntPair.ant1_X    = -1368635.125;
getAntPair.ant1_Y    =  4772999.000;
getAntPair.ant1_Z    =  3993953.250;
getAntPair.ant2_X    = -1376963.375;
getAntPair.ant2_Y    =  4770603.000;
getAntPair.ant2_Z    =  3993953.250;
```

3.1.2 Get Antenna Pair Loss (Geodetic)

This message is generated by a client as a request for the amount of attenuation and separation between two antennas in Geodetic coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 5.

Table 5. Get Antenna Pair Loss (Geodetic) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **2** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Frequency	Frequency of the radio transmission in units Hz.
20	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
28	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
36	DFloat	Antenna 1 Latitude	Latitude in degrees.
44	DFloat	Antenna 1 Longitude	Longitude in degrees.
52	DFloat	Antenna 1 Altitude	Altitude in meters.
60	DFloat	Antenna 2 Latitude	Latitude in degrees.
68	DFloat	Antenna 2 Longitude	Longitude in degrees.
76	DFloat	Antenna 2 Altitude	Altitude in meters.
Packet Length: 84 bytes			

Note: *The Antenna 1 ID and Antenna 2 ID elements are 64-bit unsigned integers to accommodate a DIS Entity ID which is a structure of 4 16-bit unsigned integers. The example below shows how one might move DIS Entity IDs into this message structure.*

The server is required to respond to this message with a **Set Antenna Pair Loss (Geodetic)** message.

Here's an example of what this message might look like if it were written in C/C++.

```
typedef struct _EntityID_
{
  USHORT   wSite;
  USHORT   wHost;
  USHORT   wEntity;
  USHORT   wRadio;
} EntityID;


typedef struct _GetAntPairLossGeodetic_
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  double    ant1_Lat;
  double    ant1_Long;
  double    ant1_Alt;
  double    ant2_Lat;
  double    ant2_Long;
  double    ant2_Alt;
} GetAntPairLossGeodetic;

EntityID               ant1;
EntityID               ant2;
GetAntPairLossGeodetic getAntPair;

ant1.wSite   = 1;
ant1.wHost   = 2;
ant1.wEntity = 3;
ant1.wRadio  = 4;
ant2.wSite   = 1;
ant2.wHost   = 2;
ant2.wEntity = 3;
ant2.wRadio  = 5;

memcpy (&getAntPair.ant1_Id, &ant1, sizeof(EntityID));
memcpy (&getAntPair.ant2_Id, &ant2, sizeof(EntityID));
getAntPair.msgId     = VTS_GET_ANT_PAIR_LOSS_GEODETIC;
getAntPair.timestamp = 1;
getAntPair.freq      = 108125000;
getAntPair.ant1_X    =   39.000;
getAntPair.ant1_Y    =  106.000;
getAntPair.ant1_Z    = 2600.000;
getAntPair.ant2_X    =   39.000;
getAntPair.ant2_Y    =  106.100;
getAntPair.ant2_Z    = 2600.000;
```

3.1.3 Get Antenna Pair Loss (UTM)

This message is generated by a client as a request for the amount of attenuation and separation between two antennas in UTM coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 6.

Table 6. Get Antenna Pair Loss (UTM) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **3** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Frequency	Frequency of the radio transmission in units Hz.
20	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
28	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
32	UInt	Antenna 1 UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
36	UInt	Antenna 2 UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
40	Int	Antenna 1 Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
44	Int	Antenna 2 Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
48	DFloat	Antenna 1 Easting	Easting in meters.
56	DFloat	Antenna 1 Northing	Northing in meters.
64	DFloat	Antenna 1 Altitude	Altitude in meters.
72	DFloat	Antenna 2 Easting	Easting in meters.
80	DFloat	Antenna 2 Northing	Northing in meters.
88	DFloat	Antenna 2 Altitude	Altitude in meters.
Packet Length: 96 bytes			

Note: *The Antenna 1 ID and Antenna 2 ID elements are 64-bit unsigned integers to accommodate a DIS Entity ID which is a structure of 4 16-bit unsigned integers. The example below shows how one might move DIS Entity IDs into this message structure.*

The server is required to respond to this message with a **Set Antenna Pair Loss (UTM)** message.

Here's an example of what this message might look like if it were written in C/C++.

```
typedef struct _EntityID_
{
  USHORT   wSite;
  USHORT   wHost;
  USHORT   wEntity;
  USHORT   wRadio;
} EntityID;


typedef struct _GetAntPairLossUTM
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  UINT      ant1_Zone;
  UINT      ant2_Zone;
  int       ant1_Hemi;
  int       ant2_Hemi;
  double    ant1_East;
  double    ant1_North;
  double    ant1_Alt;
  double    ant2_East;
  double    ant2_North;
  double    ant2_Alt;
} GetAntPairUTM;

EntityID          ant1;
EntityID          ant2;
GetAntPairLossUTM getAntPair;

ant1.wSite   = 1;
ant1.wHost   = 2;
ant1.wEntity = 3;
ant1.wRadio  = 4;
ant2.wSite   = 1;
ant2.wHost   = 2;
ant2.wEntity = 3;
ant2.wRadio  = 5;

memcpy (&getAntPair.ant1_Id, &ant1, sizeof(EntityID));
memcpy (&getAntPair.ant2_Id, &ant2, sizeof(EntityID));
getAntPair.msgId      = VTS_GET_ANT_PAIR_LOSS_UTM;
getAntPair.freq       = 108125000;
getAntPair.ant1_Zone  = 48;
getAntPair.ant2_Zone  = 48;
getAntPair.ant1_Hemi  = VTS_HEMI_NORTH;
getAntPair.ant2_Hemi  = VTS_HEMI_NORTH;
getAntPair.ant1_East  =  586592.678;
getAntPair.ant1_North = 4317252.165;
getAntPair.ant1_Alt   =    2600.000;
getAntPair.ant2_East  =  595252.159;
getAntPair.ant2_North = 4317352.045;
getAntPair.ant2_Alt   =    2600.000;
```

3.1.4 Get Height Above Terrain (Geocentric)

This message is generated by a client as a request for the height above terrain at a given position in Geocentric coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 7.

Table 7. Get Height Above Terrain (Geocentric) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **4** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DFloat	Position X	X in meters.
20	DFloat	Position Y	Y in meters.
28	DFloat	Position Z	Z in meters.
Packet Length: 36 bytes			

The server is required to respond to this message with a **Set Height Above Terrain (Geocentric)** message.

3.1.5 Get Height Above Terrain (Geodetic)

This message is generated by a client as a request for the height above terrain at a given position in Geodetic coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 8.

Table 8. Get Height Above Terrain (Geodetic) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **5** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DFloat	Position Latitude	Latitude in degrees.
20	DFloat	Position Longitude	Longitude in degrees.
28	DFloat	Position Altitude	Altitude in meters.
Packet Length: 36 bytes			

The server is required to respond to this message with a **Set Height Above Terrain (Geodetic)** message.

3.1.6 Get Height Above Terrain (UTM)

This message is generated by a client as a request for the height above terrain at a given position in UTM coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 9.

Table 9. Get Height Above Terrain (UTM) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **6** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	UInt	Position Zone	Identifies the UTM zone in which the position is located; Range 1 to 60.
16	Int	Position Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
20	DFloat	Position Easting	Easting in meters.
28	DFloat	Position Northing	Northing in meters.
36	DFloat	Position Altitude	Altitude in meters.
Packet Length: 44 bytes			

The server is required to respond to this message with a **Set Height Above Terrain (UTM)** message.

3.1.7 Set Antenna Pair Loss (Geocentric)

This message is generated by the server in response to a **Get Antenna Pair Loss (Geocentric)** request for the amount of attenuation and separation between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 10.

Table 10. Set Antenna Pair Loss (Geocentric) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **101** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Frequency	Frequency of the radio transmission in units Hz.
20	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
28	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
36	DFloat	Antenna 1 X	X in meters.
44	DFloat	Antenna 1 Y	Y in meters.
52	DFloat	Antenna 1 Z	Z in meters.
60	DFloat	Antenna 2 X	X in meters.
68	DFloat	Antenna 2 Y	Y in meters.
76	DFloat	Antenna 2 Z	Z in meters.
84	DFloat	Separation	Separation between the two radios in meters.
92	Float	Attenuation	Radio transmission attenuation between the two radios expressed in dBm.
96	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 100 bytes			

3.1.8 Set Antenna Pair Loss (Geodetic)

This message is generated by the server in response to a **Get Antenna Pair Loss (Geodetic)** request for the amount of attenuation and separation between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 11.

Table 11. Set Antenna Pair Loss (Geodetic) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **102** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Frequency	Frequency of the radio transmission in units Hz.
20	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
28	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
36	DFloat	Antenna 1 Latitude	Latitude in degrees.
44	DFloat	Antenna 1 Longitude	Longitude in degrees.
52	DFloat	Antenna 1 Altitude	Altitude in meters.
60	DFloat	Antenna 2 Latitude	Latitude in degrees.
68	DFloat	Antenna 2 Longitude	Longitude in degrees.
76	DFloat	Antenna 2 Altitude	Altitude in meters.
84	DFloat	Separation	Separation between the two radios in meters.
92	Float	Attenuation	Radio transmission attenuation between the two radios expressed in dBm.
96	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 100 bytes			

3.1.9 Set Antenna Pair Loss (UTM)

This message is generated by the server in response to a **Get Antenna Pair Loss (UTM)** request for the amount of attenuation and separation between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 12.

Table 12. Set Antenna Pair Loss (UTM) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **103** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Frequency	Frequency of the radio transmission in units Hz.
20	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
28	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
36	UInt	Antenna 1 UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
40	UInt	Antenna 2 UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
44	Int	Antenna 1 Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
48	Int	Antenna 2 Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
52	DFloat	Antenna 1 Easting	Easting in meters.
60	DFloat	Antenna 1 Northing	Northing in meters.
68	DFloat	Antenna 1 Altitude	Altitude in meters.
76	DFloat	Antenna 2 Easting	Easting in meters.
84	DFloat	Antenna 2 Northing	Northing in meters.
92	DFloat	Antenna 2 Altitude	Altitude in meters.
100	DFloat	Separation	Separation between the two radios in meters.
108	Float	Attenuation	Radio transmission attenuation between the two radios expressed in dBm.
112	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 116 bytes			

3.1.10 Set Height Above Terrain (Geocentric)

This message is generated by the server in response to a **Get Height Above Terrain (Geocentric)** request for the height above terrain at a given earth point. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 13.

Table 13. Set Height Above Terrain (Geocentric) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **104** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DFloat	Position X	X in meters.
20	DFloat	Position Y	Y in meters.
28	DFloat	Position Z	Z in meters.
36	DFloat	Height	Height in meters
44	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 48 bytes			

3.1.11 Set Height Above Terrain (Geodetic)

This message is generated by the server in response to a **Get Height Above Terrain (Geodetic)** request for the height above terrain at a given earth point. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 14.

Table 14. Set Height Above Terrain (Geodetic) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **105** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DFloat	Position Latitude	Latitude in degrees.
20	DFloat	Position Longitude	Longitude in degrees.
28	DFloat	Position Altitude	Altitude in meters.
36	DFloat	Height	Height in meters
44	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 48 bytes			

3.1.12 Set Height Above Terrain (UTM)

This message is generated by the server in response to a **Get Height Above Terrain (UTM)** request for the height above terrain at a given earth point. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 15.

Table 15. Set Height Above Terrain (UTM) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **106** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	UInt	Position Zone	Identifies the UTM zone in which the position is located; Range 1 to 60.
16	Int	Position Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
20	DFloat	Position Easting	Easting in meters.
28	DFloat	Position Northing	Northing in meters.
36	DFloat	Position Altitude	Altitude in meters.
44	DFloat	Height	Height in meters
52	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 56 bytes			

3.1.13 Get Communications Radio Pair Signal Strength (Geocentric)

This message is generated by a client as a request for the signal strength between two antennas in Geocentric coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 16.

Table 16. Get Communications Radio Pair Signal Strength (Geocentric) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **201** (as per Table 2).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this field exactly how it received it.
12	UByte	Exercise ID	The DIS Exercise ID for this antenna pair.
13	DUInt	Frequency	Frequency of the radio transmission in units Hz.
21	DUInt	Tx Antenna ID	Unique identifier for the transmitter.
29	DFloat	Tx Antenna X	X in units meters.
27	DFloat	Tx Antenna Y	Y in units meters.
45	DFloat	Tx Antenna Z	Z in units meters.
53	Float	Tx Radio Power	Transmitting radio's power level expressed in units dBm.
57	Float	Tx Antenna Bandwidth	Frequency bandwidth of the transmitting radio in units Hz.
61	UWord	Tx Major Modulation	Transmitting radio's Major Modulation enumeration as per SISO DIS enumerations.
63	UWord	Tx Detail Modulation	Transmitting radio's Detail Modulation enumeration as per SISO DIS enumerations.
65	DUInt	Rx Antenna ID	Unique identifier for the receiver.
73	DFloat	Rx Antenna X	X in units meters.
81	DFloat	Rx Antenna Y	Y in units meters.
89	DFloat	Rx Antenna Z	Z in units meters.
97	Float	Rx Antenna Bandwidth	Frequency bandwidth of the receiving radio in units Hz.
101	UWord	Rx Major Modulation	Receiving radio's Major Modulation value as per SISO DIS enumerations.
103	UWord	Rx Detail Modulation	Receiving radio's Detail Modulation value as per SISO DIS enumerations.
Packet Length: 105 bytes			

The server is required to respond to this message with a **Set Communications Radio Pair Signal Strength (Geocentric)** message.

3.1.14 Get Communications Radio Pair Signal Strength (Geodetic)

This message is generated by a client as a request for the signal strength between two antennas in Geodetic coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 17.

Table 17. Get Communications Radio Pair Signal Strength (Geodetic) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **202** (as per Table 2).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this field exactly how it received it.
12	UByte	Exercise ID	The DIS Exercise ID for this antenna pair.
13	DUInt	Frequency	Frequency of the radio transmission in units Hz.
21	DUInt	Tx Antenna ID	Unique identifier for the transmitter.
29	DFloat	Tx Antenna Latitude	Latitude in degrees.
27	DFloat	Tx Antenna Longitude	Longitude in degrees.
45	DFloat	Tx Antenna Altitude	Altitude in meters.
53	Float	Tx Radio Power	Transmitting radio's power level expressed in units dBm.
57	Float	Tx Antenna Bandwidth	Frequency bandwidth of the transmitting radio in units Hz.
61	UWord	Tx Major Modulation	Transmitting radio's Major Modulation enumeration as per SISO DIS enumerations.
63	UWord	Tx Detail Modulation	Transmitting radio's Detail Modulation enumeration as per SISO DIS enumerations.
65	DUInt	Rx Antenna ID	Unique identifier for the receiver.
73	DFloat	Rx Antenna Latitude	Latitude in degrees.
81	DFloat	Rx Antenna Longitude	Longitude in degrees.
89	DFloat	Rx Antenna Altitude	Altitude in meters.
97	Float	Rx Antenna Bandwidth	Frequency bandwidth of the receiving radio in units Hz.
101	UWord	Rx Major Modulation	Receiving radio's Major Modulation value as per SISO DIS enumerations.
103	UWord	Rx Detail Modulation	Receiving radio's Detail Modulation value as per SISO DIS enumerations.
Packet Length: 105 bytes			

The server is required to respond to this message with a **Set Communications Radio Pair Signal Strength (Geodetic)** message.

3.1.15 Get Communications Radio Pair Signal Strength (UTM)

This message is generated by a client as a request for the signal strength between two antennas in UTM coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 18.

Table 18. Get Communications Radio Pair Signal Strength (UTM) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **203** (as per Table 2).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this field exactly how it received it.
12	UByte	Exercise ID	The DIS Exercise ID for this antenna pair.
13	DUInt	Frequency	Frequency of the radio transmission in units Hz.
21	DUInt	Tx Antenna ID	Unique identifier for the transmitter.
29	UInt	Tx Antenna UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
33	UInt	Tx Antenna Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
37	DFloat	Tx Antenna Easting	Easting in meters.
45	DFloat	Tx Antenna Northing	Northing in meters.
53	DFloat	Tx Antenna Altitude	Altitude in meters.
61	Float	Tx Radio Power	Transmitting radio's power level expressed in units dBm.
65	Float	Tx Antenna Bandwidth	Frequency bandwidth of the transmitting radio in units Hz.
69	UWord	Tx Major Modulation	Transmitting radio's Major Modulation enumeration as per SISO DIS enumerations.
71	UWord	Tx Detail Modulation	Transmitting radio's Detail Modulation enumeration as per SISO DIS enumerations.
73	DUInt	Rx Antenna ID	Unique identifier for the receiver.
81	UInt	Rx Antenna UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
85	UInt	Rx Antenna Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
89	DFloat	Rx Antenna Easting	Easting in meters.
97	DFloat	Rx Antenna Northing	Northing in meters.
105	DFloat	Rx Antenna Altitude	Altitude in meters.
113	Float	Rx Antenna Bandwidth	Frequency bandwidth of the receiving radio in units Hz.
117	UWord	Rx Major Modulation	Receiving radio's Major Modulation value as per SISO DIS enumerations.
119	UWord	Rx Detail Modulation	Receiving radio's Detail Modulation value as per SISO DIS enumerations.
Packet Length: 121 bytes			

The server is required to respond to this message with a **Set Communications Radio Pair Signal Strength (UTM)** message.

3.1.16 Set Communications Radio Pair Signal Strength (Geocentric)

This message is generated by the server in response to a **Get Communications Radio Pair Signal Strength (Geocentric)** request for the signal strength between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 19.

Table 19. Set Communications Radio Pair Signal Strength (Geocentric) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **301** (as per Table 2).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this field exactly how it received it.
12	UByte	Exercise ID	The DIS Exercise ID for this antenna pair.
13	DUInt	Frequency	Frequency of the radio transmission in units Hz.
21	DUInt	Tx Antenna ID	Unique identifier for the transmitter .
29	DFloat	Tx Antenna X	X in units meters.
37	DFloat	Tx Antenna Y	Y in units meters.
45	DFloat	Tx Antenna Z	Z in units meters.
53	DUInt	Rx Antenna ID	Unique identifier for the receiver.
61	DFloat	Rx Antenna X	X in units meters.
69	DFloat	Rx Antenna Y	Y in units meters.
77	DFloat	Rx Antenna Z	Z in units meters.
85	DFloat	Separation	Separation between the two radios in meters.
93	Float	Signal Strength	Radio transmission signal strength between the two radios expressed in units dBm.
97	Float	Noise Strength	Total noise strength at receiver from all noise sources expressed in units dBm.
Packet Length: 101 bytes			

3.1.17 Set Communications Radio Pair Signal Strength (Geodetic)

This message is generated by the server in response to a **Get Communications Radio Pair Signal Strength (Geodetic)** request for the signal strength between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 20.

Table 20. Set Communications Radio Pair Signal Strength (Geodetic) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **302** (as per Table 2).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this field exactly how it received it.
12	UByte	Exercise ID	The DIS Exercise ID for this antenna pair.
13	DUInt	Frequency	Frequency of the radio transmission in units Hz.
21	DUInt	Tx Antenna ID	Unique identifier for the transmitter .
29	DFloat	Tx Antenna Latitude	Latitude in degrees;
37	DFloat	Tx Antenna Longitude	Longitude in degrees.
45	DFloat	Tx Antenna Altitude	Altitude in meters.
53	DUInt	Rx Antenna ID	Unique identifier for the receiver.
61	DFloat	Rx Antenna Latitude	Latitude in degrees;
69	DFloat	Rx Antenna Longitude	Longitude in degrees.
77	DFloat	Rx Antenna Altitude	Altitude in meters.
85	DFloat	Separation	Separation between the two radios in meters.
93	Float	Signal Strength	Radio transmission signal strength between the two radios expressed in units dBm.
97	Float	Noise Strength	Total noise strength at receiver from all noise sources expressed in units dBm.
Packet Length: 101 bytes			

3.1.18 Set Communications Radio Pair Signal Strength (UTM)

This message is generated by the server in response to a **Get Communications Radio Pair Signal Strength (UTM)** request for the signal strength between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 21.

Table 21. Set Communications Radio Pair Signal Strength (UTM) Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **303** (as per Table 2).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this field exactly how it received it.
12	UByte	Exercise ID	The DIS Exercise ID for this antenna pair.
13	DUInt	Frequency	Frequency of the radio transmission in units Hz.
21	DUInt	Tx Antenna ID	Unique identifier for the transmitter .
29	UInt	Tx Antenna UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
33	UInt	Tx Antenna Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
37	DFloat	Tx Antenna Easting	Easting in meters.
45	DFloat	Tx Antenna Northing	Northing in meters.
53	DFloat	Tx Antenna Altitude	Altitude in meters.
61	DUInt	Rx Antenna ID	Unique identifier for the receiver.
69	UInt	Rx Antenna UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
73	UInt	Rx Antenna Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
77	DFloat	Rx Antenna Easting	Easting in meters.
85	DFloat	Rx Antenna Northing	Northing in meters.
93	DFloat	Rx Antenna Altitude	Altitude in meters.
101	DFloat	Separation	Separation between the two radios in meters.
109	Float	Signal Strength	Radio transmission signal strength between the two radios expressed in units dBm.
113	Float	Noise Strength	Total noise strength at receiver from all noise sources expressed in units dBm.
Packet Length: 117 bytes			

3.1.19 Get Antenna Pair Loss (Geocentric) with Exercise ID

This message is generated by a client as a request for the amount of attenuation and separation between two antennas in Geocentric coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 22.

Table 22. Get Antenna Pair Loss (Geocentric) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **211** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DUInt	Frequency	Frequency of the radio transmission in units Hz.
28	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
36	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
44	DFloat	Antenna 1 X	X in meters.
52	DFloat	Antenna 1 Y	Y in meters.
60	DFloat	Antenna 1 Z	Z in meters.
68	DFloat	Antenna 2 X	X in meters.
76	DFloat	Antenna 2 Y	Y in meters.
84	DFloat	Antenna 2 Z	Z in meters.
Packet Length: 92 bytes			

Note: *The Antenna 1 ID and Antenna 2 ID elements are 64-bit unsigned integers to accommodate a DIS Entity ID which is a structure of 4 16-bit unsigned integers.*

The server is required to respond to this message with a **Set Antenna Pair Loss (Geocentric) with Exercise ID** message.

3.1.20 Get Antenna Pair Loss (Geodetic) with Exercise ID

This message is generated by a client as a request for the amount of attenuation and separation between two antennas in Geodetic coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 23.

Table 23. Get Antenna Pair Loss (Geodetic) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **212** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DUInt	Frequency	Frequency of the radio transmission in units Hz.
28	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
36	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
44	DFloat	Antenna 1 Latitude	Latitude in degrees.
52	DFloat	Antenna 1 Longitude	Longitude in degrees.
60	DFloat	Antenna 1 Altitude	Altitude in meters.
68	DFloat	Antenna 2 Latitude	Latitude in degrees.
76	DFloat	Antenna 2 Longitude	Longitude in degrees.
84	DFloat	Antenna 2 Altitude	Altitude in meters.
Packet Length: 92 bytes			

Note: *The Antenna 1 ID and Antenna 2 ID elements are 64-bit unsigned integers to accommodate a DIS Entity ID which is a structure of 4 16-bit unsigned integers.*

The server is required to respond to this message with a **Set Antenna Pair Loss (Geodetic) with Exercise ID** message.

3.1.21 Get Antenna Pair Loss (UTM) with Exercise ID

This message is generated by a client as a request for the amount of attenuation and separation between two antennas in UTM coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 24.

Table 24. Get Antenna Pair Loss (UTM) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **213** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DUInt	Frequency	Frequency of the radio transmission in units Hz.
28	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
36	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
40	UInt	Antenna 1 UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
44	UInt	Antenna 2 UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
48	Int	Antenna 1 Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
52	Int	Antenna 2 Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
56	DFloat	Antenna 1 Easting	Easting in meters.
64	DFloat	Antenna 1 Northing	Northing in meters.
72	DFloat	Antenna 1 Altitude	Altitude in meters.
80	DFloat	Antenna 2 Easting	Easting in meters.
88	DFloat	Antenna 2 Northing	Northing in meters.
96	DFloat	Antenna 2 Altitude	Altitude in meters.
Packet Length: 104 bytes			

Note: *The Antenna 1 ID and Antenna 2 ID elements are 64-bit unsigned integers to accommodate a DIS Entity ID which is a structure of 4 16-bit unsigned integers.*

The server is required to respond to this message with a **Set Antenna Pair Loss (UTM) with Exercise ID** message.

3.1.22 Get Height Above Terrain (Geocentric) with Exercise ID

This message is generated by a client as a request for the height above terrain at a given position in Geocentric coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 25.

Table 25. Get Height Above Terrain (Geocentric) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **214** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DFloat	Position X	X in meters.
28	DFloat	Position Y	Y in meters.
36	DFloat	Position Z	Z in meters.
Packet Length: 44 bytes			

The server is required to respond to this message with a **Set Height Above Terrain (Geocentric) with Exercise ID** message.

3.1.23 Get Height Above Terrain (Geodetic) with Exercise ID

This message is generated by a client as a request for the height above terrain at a given position in Geodetic coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 26.

Table 26. Get Height Above Terrain (Geodetic) with Ex ID Message Structure			
Offset	**Type**	**Name**	**Description**
0	UInt	Message ID	Unique message identifier which is set to a value of **215** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DFloat	Position Latitude	Latitude in degrees.
28	DFloat	Position Longitude	Longitude in degrees.
36	DFloat	Position Altitude	Altitude in meters.
Packet Length: 44 bytes			

The server is required to respond to this message with a **Set Height Above Terrain (Geodetic) with Exercise ID** message.

3.1.24 Get Height Above Terrain (UTM) with Exercise ID

This message is generated by a client as a request for the height above terrain at a given position in UTM coordinates. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 27.

Table 27. Get Height Above Terrain (UTM) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **216** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server shall return this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	UInt	Position Zone	Identifies the UTM zone in which the position is located; Range 1 to 60.
24	Int	Position Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
28	DFloat	Position Easting	Easting in meters.
36	DFloat	Position Northing	Northing in meters.
44	DFloat	Position Altitude	Altitude in meters.
Packet Length: 52 bytes			

The server is required to respond to this message with a **Set Height Above Terrain (UTM) with Exercise ID** message.

3.1.25 Set Antenna Pair Loss (Geocentric) with Exercise ID

This message is generated by the server in response to a **Get Antenna Pair Loss (Geocentric) with Exercise ID** request for the amount of attenuation and separation between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 28.

Table 28. Set Antenna Pair Loss (Geocentric) with Ex ID Message Structure			
Offset	**Type**	**Name**	**Description**
0	UInt	Message ID	Unique message identifier which is set to a value of **311** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DUInt	Frequency	Frequency of the radio transmission in units Hz.
28	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
36	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
44	DFloat	Antenna 1 X	X in meters.
52	DFloat	Antenna 1 Y	Y in meters.
60	DFloat	Antenna 1 Z	Z in meters.
68	DFloat	Antenna 2 X	X in meters.
76	DFloat	Antenna 2 Y	Y in meters.
84	DFloat	Antenna 2 Z	Z in meters.
92	DFloat	Separation	Separation between the two radios in meters.
100	Float	Attenuation	Radio transmission attenuation between the two radios expressed in dBm.
104	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 108 bytes			

3.1.26 Set Antenna Pair Loss (Geodetic) with Exercise ID

This message is generated by the server in response to a **Get Antenna Pair Loss (Geodetic) with Exercise ID** request for the amount of attenuation and separation between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 29.

Table 29. Set Antenna Pair Loss (Geodetic) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **312** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DUInt	Frequency	Frequency of the radio transmission in units Hz.
28	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
36	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
44	DFloat	Antenna 1 Latitude	Latitude in degrees.
52	DFloat	Antenna 1 Longitude	Longitude in degrees.
60	DFloat	Antenna 1 Altitude	Altitude in meters.
68	DFloat	Antenna 2 Latitude	Latitude in degrees.
76	DFloat	Antenna 2 Longitude	Longitude in degrees.
84	DFloat	Antenna 2 Altitude	Altitude in meters.
92	DFloat	Separation	Separation between the two radios in meters.
100	Float	Attenuation	Radio transmission attenuation between the two radios expressed in dBm.
104	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 108 bytes			

3.1.27 Set Antenna Pair Loss (UTM) with Exercise ID

This message is generated by the server in response to a **Get Antenna Pair Loss (UTM) with Exercise ID** request for the amount of attenuation and separation between two antennas. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 30.

Table 30. Set Antenna Pair Loss (UTM) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **313** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DUInt	Frequency	Frequency of the radio transmission in units Hz.
28	DUInt	Antenna 1 ID	Unique identifier for Antenna 1.
36	DUInt	Antenna 2 ID	Unique identifier for Antenna 2.
44	UInt	Antenna 1 UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
48	UInt	Antenna 2 UTM Zone	Identifies the UTM zone in which the antenna is located; Range 1 to 60.
52	Int	Antenna 1 Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
56	Int	Antenna 2 Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
60	DFloat	Antenna 1 Easting	Easting in meters.
68	DFloat	Antenna 1 Northing	Northing in meters.
76	DFloat	Antenna 1 Altitude	Altitude in meters.
84	DFloat	Antenna 2 Easting	Easting in meters.
92	DFloat	Antenna 2 Northing	Northing in meters.
100	DFloat	Antenna 2 Altitude	Altitude in meters.
108	DFloat	Separation	Separation between the two radios in meters.
116	Float	Attenuation	Radio transmission attenuation between the two radios expressed in dBm.
120	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 124 bytes			

3.1.28 Set Height Above Terrain (Geocentric) with Exercise ID

This message is generated by the server in response to a **Get Height Above Terrain (Geocentric) with Exercise ID** request for the height above terrain at a given earth point. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 31.

Table 31. Set Height Above Terrain (Geocentric) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **314** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DFloat	Position X	X in meters.
28	DFloat	Position Y	Y in meters.
36	DFloat	Position Z	Z in meters.
44	DFloat	Height	Height in meters
52	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 56 bytes			

3.1.29 Set Height Above Terrain (Geodetic) with Exercise ID

This message is generated by the server in response to a **Get Height Above Terrain (Geodetic) with Exercise ID** request for the height above terrain at a given earth point. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 32.

Table 32. Set Height Above Terrain (Geodetic) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **315** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	DFloat	Position Latitude	Latitude in degrees.
28	DFloat	Position Longitude	Longitude in degrees.
36	DFloat	Position Altitude	Altitude in meters.
44	DFloat	Height	Height in meters
52	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 56 bytes			

3.1.30 Set Height Above Terrain (UTM) with Exercise ID

This message is generated by the server in response to a **Get Height Above Terrain (UTM) with Exercise ID** request for the height above terrain at a given earth point. The offset of each element in the message is calculated in 8-bit bytes. Type descriptions are provided in Table 3. The structure for this message is provided in Table 33.

Table 33. Set Height Above Terrain (UTM) with Ex ID Message Structure

Offset	Type	Name	Description
0	UInt	Message ID	Unique message identifier which is set to a value of **316** (as per Table 1).
4	DUInt	Timestamp	This field is provided for the client to use. The server returns this datum exactly how it received it.
12	DUInt	Exercise ID	Exercise Identifier
20	UInt	Position Zone	Identifies the UTM zone in which the position is located; Range 1 to 60.
24	Int	Position Hemisphere	Defines the hemisphere of the UTM zone. See section 3.1 for acceptable values.
28	DFloat	Position Easting	Easting in meters.
36	DFloat	Position Northing	Northing in meters.
44	DFloat	Position Altitude	Altitude in meters.
52	DFloat	Height	Height in meters
60	UInt	VTS Message Status	Status code as defined in Table 34.
Packet Length: 64 bytes			

3.2 Status Codes

Table 34 identifies and describes the possible status codes returned from the server.

Table 34. VTS Remote Message Status Codes

Code	Value	Description
VTS_STAT_COOL	0	All systems go - Everything be cool.
VTS_STAT_MSG_TOO_LONG	1	The message received by the server is too long with respect to the message id.
VTS_STAT_MSG_TOO_SHORT	2	The message received by the server is too short with respect to the message id.
VTS_STAT_MSG_NO_TERRAIN	3	No terrain data was available for the specified location.

3.3 Header File

The following code segment contains the enumerations and structure definitions contained in the VTSRemoteMessage.h header file which defines the remote message protocol.

```
///////////////////////////////////////////////////////////////////////////////
//
// VTS Remote Message Data Types
//
///////////////////////////////////////////////////////////////////////////////
#pragma once       // Include only once

#define TERRAIN_SERVER_BUILD_100    100
#define TERRAIN_SERVER_BUILD_200    200

typedef enum
{
  VTS_STAT_COOL           = 0,
  VTS_STAT_MSG_TOO_LONG   = 1,
  VTS_STAT_MSG_TOO_SHORT  = 2,
  VTS_STAT_MSG_NO_TERRAIN = 3
} VTSMessageStatus;

typedef enum
{
  VTS_NO_MSG                         =   0,

  //
  // Build 100 messages
  //
  VTS_GET_ANT_PAIR_LOSS_GEOCENTRIC   =   1,
  VTS_GET_ANT_PAIR_LOSS_GEODETIC     =   2,
  VTS_GET_ANT_PAIR_LOSS_UTM          =   3,
  VTS_GET_HEIGHT_GEOCENTRIC          =   4,
  VTS_GET_HEIGHT_GEODETIC            =   5,
  VTS_GET_HEIGHT_UTM                 =   6,

  VTS_SET_ANT_PAIR_LOSS_GEOCENTRIC   = 101,
  VTS_SET_ANT_PAIR_LOSS_GEODETIC     = 102,
  VTS_SET_ANT_PAIR_LOSS_UTM          = 103,
  VTS_SET_HEIGHT_GEOCENTRIC          = 104,
  VTS_SET_HEIGHT_GEODETIC            = 105,
  VTS_SET_HEIGHT_UTM                 = 106,

  //
  // Build 200 messages
  //
  VTS_GET_COMM_RADIO_SIG_STRENGTH_GEOCENTRIC = 201,
  VTS_GET_COMM_RADIO_SIG_STRENGTH_GEODETIC   = 202,
  VTS_GET_COMM_RADIO_SIG_STRENGTH_UTM        = 203,

  VTS_WITHEX_GET_ANT_PAIR_LOSS_GEOCENTRIC    = 211,
  VTS_WITHEX_GET_ANT_PAIR_LOSS_GEODETIC      = 212,
  VTS_WITHEX_GET_ANT_PAIR_LOSS_UTM           = 213,
  VTS_WITHEX_GET_HEIGHT_GEOCENTRIC           = 214,
```

```
  VTS_WITHEX_GET_HEIGHT_GEODETIC                 = 215,
  VTS_WITHEX_GET_HEIGHT_UTM                      = 216,


  VTS_SET_COMM_RADIO_SIG_STRENGTH_GEOCENTRIC = 301,
  VTS_SET_COMM_RADIO_SIG_STRENGTH_GEODETIC   = 302,
  VTS_SET_COMM_RADIO_SIG_STRENGTH_UTM        = 303,

  VTS_WITHEX_SET_ANT_PAIR_LOSS_GEOCENTRIC    = 311,
  VTS_WITHEX_SET_ANT_PAIR_LOSS_GEODETIC      = 312,
  VTS_WITHEX_SET_ANT_PAIR_LOSS_UTM           = 313,
  VTS_WITHEX_SET_HEIGHT_GEOCENTRIC           = 314,
  VTS_WITHEX_SET_HEIGHT_GEODETIC             = 315,
  VTS_WITHEX_SET_HEIGHT_UTM                  = 316

} VTSRemoteMessageIDs;


typedef struct _GetAntPairLossGeocentric_        // Message ID = 1
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  double    ant1_X;
  double    ant1_Y;
  double    ant1_Z;
  double    ant2_X;
  double    ant2_Y;
  double    ant2_Z;
} GetAntPairLossGeocentric;

typedef struct _GetAntPairLossGeodetic_          // Message ID = 2
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  double    ant1_Lat;
  double    ant1_Long;
  double    ant1_Alt;
  double    ant2_Lat;
  double    ant2_Long;
  double    ant2_Alt;
} GetAntPairLossGeodetic;

typedef struct _GetAntPairLossUTM_               // Message ID = 3
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  UINT      ant1_Zone;
  UINT      ant2_Zone;
```

```
   int        ant1_Hemi;
   int        ant2_Hemi;
   double     ant1_East;
   double     ant1_North;
   double     ant1_Alt;
   double     ant2_East;
   double     ant2_North;
   double     ant2_Alt;
} GetAntPairLossUTM;

typedef struct _GetHeightAboveTerrainGeocentric_  // Message ID = 4
{
   UINT       msgId;
   ULONG64    timestamp;
   double     posX;
   double     posY;
   double     posZ;
} GetHeightAboveTerrainGeocentric;

typedef struct _GetHeightAboveTerrainGeodetic_    // Message ID = 5
{
   UINT       msgId;
   ULONG64    timestamp;
   double     posLat;
   double     posLong;
   double     posAlt;
} GetHeightAboveTerrainGeodetic;

typedef struct _GetHeightAboveTerrainUTM_         // Message ID = 6
{
   UINT       msgId;
   ULONG64    timestamp;
   UINT       posZone;
   int        posHemi;
   double     posEast;
   double     posNorth;
   double     posAlt;
} GetHeightAboveTerrainUTM;

typedef struct _SetAntPairLossGeocentric_         // Message ID = 101
{
   UINT              msgId;
   ULONG64           timestamp;
   ULONG64           freq;
   ULONG64           ant1_Id;
   ULONG64           ant2_Id;
   double            ant1_X;
   double            ant1_Y;
   double            ant1_Z;
   double            ant2_X;
   double            ant2_Y;
   double            ant2_Z;
   double            separation;
   float             atten;
   VTSMessageStatus status;
} SetAntPairLossGeocentric;
```

```
typedef struct _SetAntPairLossGeodetic_          // Message ID = 102
{
  UINT             msgId;
  ULONG64          timestamp;
  UINT             freq;
  ULONG64          ant1_Id;
  ULONG64          ant2_Id;
  double           ant1_Lat;
  double           ant1_Long;
  double           ant1_Alt;
  double           ant2_Lat;
  double           ant2_Long;
  double           ant2_Alt;
  double           separation;
  float            atten;
  VTSMessageStatus status;
} SetAntPairLossGeodetic;

typedef struct _SetAntPairLossUTM_               // Message ID = 103
{
  UINT             msgId;
  ULONG64          timestamp;
  ULONG64          freq;
  ULONG64          ant1_Id;
  ULONG64          ant2_Id;
  UINT             ant1_Zone;
  UINT             ant2_Zone;
  int              ant1_Hemi;
  int              ant2_Hemi;
  double           ant1_East;
  double           ant1_North;
  double           ant1_Alt;
  double           ant2_East;
  double           ant2_North;
  double           ant2_Alt;
  double           separation;
  float            atten;
  VTSMessageStatus status;
} SetAntPairLossUTM;

typedef struct _SetHeightAboveTerrainGeocentric_  // Message ID = 104
{
  UINT             msgId;
  ULONG64          timestamp;
  double           posX;
  double           posY;
  double           posZ;
  double           height;
  VTSMessageStatus status;
} SetHeightAboveTerrainGeocentric;

typedef struct _SetHeightAboveTerrainGeodetic_    // Message ID = 105
{
  UINT             msgId;
  ULONG64          timestamp;
  double           posLat;
  double           posLong;
```

```
  double           posAlt;
  double           height;
  VTSMessageStatus status;
} SetHeightAboveTerrainGeodetic;

typedef struct _SetHeightAboveTerrainUTM_         // Message ID = 106
{
  UINT             msgId;
  ULONG64          timestamp;
  UINT             posZone;
  int              posHemi;
  double           posEast;
  double           posNorth;
  double           posAlt;
  double           height;
  VTSMessageStatus status;
} SetHeightAboveTerrainUTM;

typedef struct _GetCommRadioSigStrengthGeocentric_       // Message ID = 201
{
  UINT      msgId;
  ULONG64   timestamp;
  UCHAR    exerciseId;
  ULONG64   freq;
  ULONG64   txAntennaId;
  double    txAntennaX;
  double    txAntennaY;
  double    txAntennaZ;
  float     txRadioPower;
  float     txAntennaBandwidth;
  USHORT    txMajorModulation;
  USHORT    txDetailModulation;
  ULONG64   rxAntennaId;
  double    rxAntennaX;
  double    rxAntennaY;
  double    rxAntennaZ;
  float     rxAntennaBandwidth;
  USHORT    rxMajorModulation;
  USHORT    rxDetailModulation;
} GetCommRadioSigStrengthGeocentric;

typedef struct _GetAntPairLossGeocentricEx_         // Message ID = 211
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64  exercise_Id;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  double    ant1_X;
  double    ant1_Y;
  double    ant1_Z;
  double    ant2_X;
  double    ant2_Y;
  double    ant2_Z;
} GetAntPairLossGeocentricEx;
```

```
typedef struct _GetAntPairLossGeodeticEx_          // Message ID = 212
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   exercise_Id;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  double    ant1_Lat;
  double    ant1_Long;
  double    ant1_Alt;
  double    ant2_Lat;
  double    ant2_Long;
  double    ant2_Alt;
} GetAntPairLossGeodeticEx;

typedef struct _GetAntPairLossUTMEx_               // Message ID = 213
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   exercise_Id;
  ULONG64   freq;
  ULONG64   ant1_Id;
  ULONG64   ant2_Id;
  UINT      ant1_Zone;
  UINT      ant2_Zone;
  int       ant1_Hemi;
  int       ant2_Hemi;
  double    ant1_East;
  double    ant1_North;
  double    ant1_Alt;
  double    ant2_East;
  double    ant2_North;
  double    ant2_Alt;
} GetAntPairLossUTMEx;

typedef struct _GetHeightAboveTerrainGeocentricEx_  // Message ID = 214
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   exercise_Id;
  double    posX;
  double    posY;
  double    posZ;
} GetHeightAboveTerrainGeocentricEx;

typedef struct _GetHeightAboveTerrainGeodeticEx_    // Message ID = 215
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64   exercise_Id;
  double    posLat;
  double    posLong;
  double    posAlt;
} GetHeightAboveTerrainGeodeticEx;

typedef struct _GetHeightAboveTerrainUTMEx_         // Message ID = 216
```

```
{
  UINT      msgId;
  ULONG64   timestamp;
  ULONG64  exercise_Id;
  UINT      posZone;
  int       posHemi;
  double    posEast;
  double    posNorth;
  double    posAlt;
} GetHeightAboveTerrainUTMEx;

typedef struct _SetCommRadioSigStrengthGeocentric_  // Message ID = 301
{
  UINT      msgId;
  ULONG64   timestamp;
  UCHAR     exerciseId;
  ULONG64   freq;
  ULONG64   txAntennaId;
  double    txAntennaX;
  double    txAntennaY;
  double    txAntennaZ;
  ULONG64   rxAntennaId;
  double    rxAntennaX;
  double    rxAntennaY;
  double    rxAntennaZ;
  double    separation;
  float     signalStrength;
  float     noiseStrength;
} SetCommRadioSigStrengthGeocentric;

typedef struct _SetAntPairLossGeocentricEx_         // Message ID = 311
{
  UINT             msgId;
  ULONG64          timestamp;
  ULONG64          exercise_Id;
  ULONG64          freq;
  ULONG64          ant1_Id;
  ULONG64          ant2_Id;
  double           ant1_X;
  double           ant1_Y;
  double           ant1_Z;
  double           ant2_X;
  double           ant2_Y;
  double           ant2_Z;
  double           separation;
  float            atten;
  VTSMessageStatus status;
} SetAntPairLossGeocentricEx;

typedef struct _SetAntPairLossGeodeticEx_           // Message ID = 312
{
  UINT             msgId;
  ULONG64          timestamp;
  ULONG64          exercise_Id;
  UINT             freq;
  ULONG64          ant1_Id;
  ULONG64          ant2_Id;
```

```
   double           ant1_Lat;
   double           ant1_Long;
   double           ant1_Alt;
   double           ant2_Lat;
   double           ant2_Long;
   double           ant2_Alt;
   double           separation;
   float            atten;
   VTSMessageStatus status;
} SetAntPairLossGeodeticEx;

typedef struct _SetAntPairLossUTMEx_                // Message ID = 313
{
   UINT             msgId;
   ULONG64          timestamp;
   ULONG64          exercise_Id;
   ULONG64          freq;
   ULONG64          ant1_Id;
   ULONG64          ant2_Id;
   UINT             ant1_Zone;
   UINT             ant2_Zone;
   int              ant1_Hemi;
   int              ant2_Hemi;
   double           ant1_East;
   double           ant1_North;
   double           ant1_Alt;
   double           ant2_East;
   double           ant2_North;
   double           ant2_Alt;
   double           separation;
   float            atten;
   VTSMessageStatus status;
} SetAntPairLossUTMEx;

typedef struct _SetHeightAboveTerrainGeocentricEx_  // Message ID = 314
{
   UINT             msgId;
   ULONG64          timestamp;
   ULONG64          exercise_Id;
   double           posX;
   double           posY;
   double           posZ;
   double           height;
   VTSMessageStatus status;
} SetHeightAboveTerrainGeocentricEx;

typedef struct _SetHeightAboveTerrainGeodeticEx_    // Message ID = 315
{
   UINT             msgId;
   ULONG64          timestamp;
   ULONG64          exercise_Id;
   double           posLat;
   double           posLong;
   double           posAlt;
   double           height;
   VTSMessageStatus status;
} SetHeightAboveTerrainGeodeticEx;
```

```
typedef struct _SetHeightAboveTerrainUTMEx_         // Message ID = 316
{
  UINT             msgId;
  ULONG64          timestamp;
  ULONG64          exercise_Id;
  UINT             posZone;
  int              posHemi;
  double           posEast;
  double           posNorth;
  double           posAlt;
  double           height;
  VTSMessageStatus status;
} SetHeightAboveTerrainUTMEx;
```

4 VComm Use of Remote Message Protocol

VComm uses the Remote Message Protocol as a client to retrieve either terrain attenuation or signal quality from a signal quality service. See the VComm User Manual for information on how to configure VComm to query for terrain attenuation or signal quality. Depending on its configuration, VComm uses different messages to query the signal quality service.

4.1 VComm Configured to Query for Terrain Attenuation

4.1.1 Builds 331 through 365

When VComm is configured to query for terrain attenuation, it can be further configured to either use message id 1 or message id 211 to query the signal quality service. VComm expects either message id 101 or message id 311 to be returned by the server. The terrain attenuation values returned by the server are used by VComm in determining signal quality.

4.1.2 Builds 366 and Later

When VComm is configured to query for terrain attenuation, it uses message id 211 to query the signal quality service. VComm expects message id 311 to be returned by the server. The terrain attenuation values returned by the server are used by VComm in determining signal quality.

4.2 VComm Configured to Query for Signal Quality

When VComm is configured to query for signal quality, it queries the server using message id 201. VComm expects message id 301 to be returned by the server. The signal strength and noise strength values returned by the server are used by VComm to directly determine signal quality, thereby bypassing VComm's internal modeling.

5 VTS Use of Remote Message Protocol

The VComm Terrain Server uses the Remote Message Protocol as a server to provide terrain attenuation and height above terrain to a client. See the VComm Terrain Server User Manual for information on how to install, configure, and use the VTS.

Version 1.0 of the VTS supported Build 100 messages. These messages are now considered obsolete.

Versions 2.0 and later of the VTS support the following Build 200 messages: 211, 212, 213, 214, 215, 216, 311, 312, 313, 314, 315, and 316.