Total Pages 65

Joint Tactical Networking Center Standard
Global Positioning System
Application Program Interface

Version: 2.1.4
24 July 2014

Statement A - Approved for public release; distribution is unlimited (29 July 2014)

REVISION HISTORY

Version	Description	Last Modified Date
1.0	Initial release. **ICWG Approved**	28-March-06
1.1	Updates from submitted SAR to add the MGRS position format	11-October-06
1.2	Update to Latitude/Longitude extension introduction.	31-October-06
1.3	Updates based on Increment 6 comments.	06-December-06
2.0	Updated to camelback naming convention. **ICWG Approved**	13-December-06
2.1 <Draft>	Removed *gpsInitialTime* interface from GPS Device	27-September-07
2.1	Updated based on ICWG Review	01-October-07
2.1.1	Updated based on ICWG: - Updated port diagrams to remove references to deleted *gpsInitialTime* interface **ICWG Approved**	11-October-07
2.1.2	Updated document stamp for public release	30-May-08
2.1.3<Draft>	Update latitude/longitude valid ranges in section B.5.3.2 Gps::GpsLatLong Structure	10-July-12
2.1.3<Final Draft>	No further changes	7-August-12
2.1.3	No further changes **ICWG Approved**	14-August-12
2.1.4	Document updated to reflect program and organizational changes. Updated document stamp for public release	24-July-14

Table of Contents

A. GPS DEVICE 8

B. LATITUDE/LONGITUDE EXTENSION 30

C. MGRS EXTENSION 48

Table of Contents

A. GPS DEVICE 8
A.1 Introduction 8
A.1.1 Overview 8
A.1.2 Service Layer Description 8
A.1.2.1 GPS Device Port Connections 8
A.1.3 Modes of Service 9
A.1.4 Service States 10
A.1.4.1 GPS Device State Diagram 10
A.1.5 Referenced Documents 11
A.1.5.1 Government Documents 11
A.1.5.2 Commercial Standards 11
A.2 Services 12
A.2.1 Provide Services 12
A.2.2 Use Services 12
A.2.3 Interface Modules 12
A.2.4 Sequence Diagrams 12
A.3 Service Primitives and Attributes 13
A.4 IDL 14
A.4.1 GpsDevice IDL 14
A.5 UML 17
A.5.1 Enumerations 17
A.5.2 Exceptions 17
A.5.3 Structures 17
A.5.3.1 Gps::GpsTime Structure 17
A.5.3.2 Gps::GpsVel Structure 18
A.5.3.3 Gps::GpsFomData Structure 19

B. LATITUDE/LONGITUDE EXTENSION 30
B.1 Introduction 30
B.1.1 Overview 30
B.1.2 Service Layer Description 31
B.1.2.1 GPS Device Latitude/Longitude Extension Port Connections 31
B.1.3 Modes of Service 32
B.1.4 Service States 32
B.1.5 Referenced Documents 32
B.2 Services 33
B.2.1 Provide Services 33
B.2.2 Use Services 33
B.2.3 Interface Modules 34
B.2.3.1 Latitude/Longitude Extension 34
B.2.4 Sequence Diagrams 35
B.2.4.1 Latitude/Longitude 1pps Event Sequence 35
B.2.4.2 Latitude/Longitude Non-Blocking Read Sequence 36
B.2.4.3 Latitude/Longitude Blocking Read Sequence 37
B.3 Service Primitives and Attributes 38

B.3.1 Gps::LatLongDataProducer 38
B.3.1.1 *readLatLong* Operation 38
B.3.2 Gps::LatLong1PpsConsumer 39
B.3.2.1 *pushLatLong1Pps* Operation 39
B.4 IDL 40
B.4.1 GpsLatLongExt IDL 40
B.5 UML 42
B.5.1 Enumerations 43
B.5.2 Exceptions 43
B.5.2.1 Gps::DeviceError Exception 43
B.5.3 Structures 44
B.5.3.1 Gps::GpsLatLongMessage Structure 44
B.5.3.2 Gps::GpsLatLong Structure 45

C. MGRS EXTENSION 48
C.1 Introduction 48
C.1.1 Overview 48
C.1.2 Service Layer Description 49
C.1.2.1 GPS Device MGRS Extension Port Connections 49
C.1.3 Modes of Service 50
C.1.4 Service States 50
C.1.5 Referenced Documents 50
C.2 Services 51
C.2.1 Provide Services 51
C.2.2 Use Services 51
C.2.3 Interface Modules 52
C.2.3.1 MGRS Extension 52
C.2.4 Sequence Diagrams 53
C.2.4.1 MGRS 1pps Event Sequence 53
C.2.4.2 MGRS Non-Blocking Read Sequence 54
C.2.4.3 MGRS Blocking Read Sequence 55
C.3 Service Primitives and Attributes 56
C.3.1 Gps::MgrsDataProducer 56
C.3.1.1 *readMgrs* Operation 56
C.3.2 Gps::Mgrs1PpsConsumer 57
C.3.2.1 *pushMgrs1Pps* Operation 57
C.4 IDL 58
C.4.1 GpsMgrsExt IDL 58
C.5 UML 60
C.5.1 Enumerations 61
C.5.2 Exceptions 61
C.5.2.1 Gps::DeviceError Exception 61
C.5.3 Structures 62
C.5.3.1 Gps::GpsMgrsMessage Structure 62
C.5.3.2 Gps::GpsMgrs Structure 63

Lists of Figures

FIGURE 1 – GPS DEVICE STATE DIAGRAM 10
FIGURE 2 – GPS INTERFACE MESSAGE FORMAT 24
FIGURE 3 – LATITUDE/LONGITUDE EXTENSION PORT DIAGRAM 31
FIGURE 4 – LATITUDE/LONGITUDE EXTENSION CLASS DIAGRAM 34
FIGURE 5 – LATLONG1PPSCONSUMER INTERFACE CLASS DIAGRAM 34
FIGURE 6 – LATLONGDATAPRODUCER INTERFACE CLASS DIAGRAM 34
FIGURE 7 – LATITUDE/LONGITUDE 1PPS EVENT SEQUENCE DIAGRAM 35
FIGURE 8 – LATITUDE/LONGITUDE NON-BLOCKING READ SEQUENCE DIAGRAM 36
FIGURE 9 – LATITUDE/LONGITUDE BLOCKING READ SEQUENCE DIAGRAM 37
FIGURE 10 – LATITUDE/LONGITUDE EXTENSION COMPONENT DIAGRAM 42
FIGURE 11 – MGRS EXTENSION PORT DIAGRAM 49
FIGURE 12 – MGRS EXTENSION CLASS DIAGRAM 52
FIGURE 13 – MGRS1PPSCONSUMER INTERFACE CLASS DIAGRAM 52
FIGURE 14 – MGRSDATAPRODUCER INTERFACE CLASS DIAGRAM 52
FIGURE 15 – MGRS 1PPS EVENT SEQUENCE DIAGRAM 53
FIGURE 16 – MGRS NON-BLOCKING READ SEQUENCE DIAGRAM 54
FIGURE 17 – MGRS BLOCKING READ SEQUENCE DIAGRAM 55
FIGURE 18 – MGRS EXTENSION COMPONENT DIAGRAM 60

List of Tables

TABLE 1 – FIGURE OF MERIT 25
TABLE 2 – TIME FIGURE OF MERIT 26
TABLE 3 – DATUM NUMBER / IDENTIFIER LIST 27
TABLE 4 – DATUM DESCRIPTION LIST 27
TABLE 5 – LATITUDE/LONGITUDE EXTENSION PROVIDE SERVICE INTERFACE 33
TABLE 6 – LATITUDE/LONGITUDE EXTENSION USE SERVICE INTERFACE 33
TABLE 7 – LATITUDE/LONGITUDE EXTENSION PERFORMANCE SPECIFICATION 47
TABLE 8 – MGRS EXTENSION PROVIDE SERVICE INTERFACE 51
TABLE 9 – MGRS EXTENSION USE SERVICE INTERFACE 51
TABLE 10 – MGRS EXTENSION PERFORMANCE SPECIFICATION 65

A. GPS DEVICE

A.1 INTRODUCTION

The *GPS Device* supports methods and attributes that are specific to the Global Positioning System (GPS) Receiver Hardware (HW) device it represents. It provides a set of common structures are used by the *GPS Device* extensions.

It should be noted that the base GPS device requires the use of at least one of the extensions to provide position, velocity, and time (PVT) data to the Device User (see Latitude/Longitude Extension and MGRS Extension).

This document defines a common set of GPS provided services and interfaces required by most Joint Tactical Radio (JTR) Sets.

The *GPS Device* acts as "device adapter". It is used by Common Object Request Broker Architecture (CORBA) components (e.g., waveform application components) to access Joint Tactical Radio (JTR) Set GPS Receiver hardware.

A.1.1Overview

This base contains as follows:

a. Section A.1, *Introduction*, of this document contains the introductory material regarding the overview, Service Layer Description, Modes, States and Referenced Documents of this document.

b. Section A.2, *Services*, provides summary of service interface uses, interface for each device component, port connections, and sequence diagrams.

c. Section A.3, *Service Primitives and Attributes*, specifies the operations that are provided by the *GPS Device*.

d. Section A.4, *IDL*.

e. Section A.5, *UML*.

f. Appendix A.A, *Abbreviations and Acronyms.*

g. Appendix A.B, *Performance Specification.*

h. Appendix A.C, *GPS Message Formats*.

i. Appendix A.D, *Figure of Merit (FOM).*

j. Appendix A.E, *Time Figure of Merit (TFOM)*.

k. Appendix A.F, *Datum Identifier*.

A.1.2Service Layer Description

A.1.2.1 GPS Device Port Connections

None. This device must be used in combination with one or more *GPS Device Extensions*.

A.1.3Modes of Service

Not applicable.

A.1.4 Service States

A.1.4.1 GPS Device State Diagram

The *GPS Device* states are illustrated in the following diagram. The *GPS Device* states ensure that received operations are only executed when the *GPS Device* is in the proper state. The five states of the *GPS Device* are as follow:

- CONSTRUCTED - The state transitioned to upon successful creation.
- INITIALIZED - The state transitioned to upon successful initialization.
- ENABLED - The state transitioned to upon successful start.
- DISABLED - The state transitioned to upon successful stop.
- RELEASED - The state transitioned to upon successful release.

The *GPS Device* transitions between states in response to the *initialize*, *start*, *stop*, and *releaseObject* operations [3].

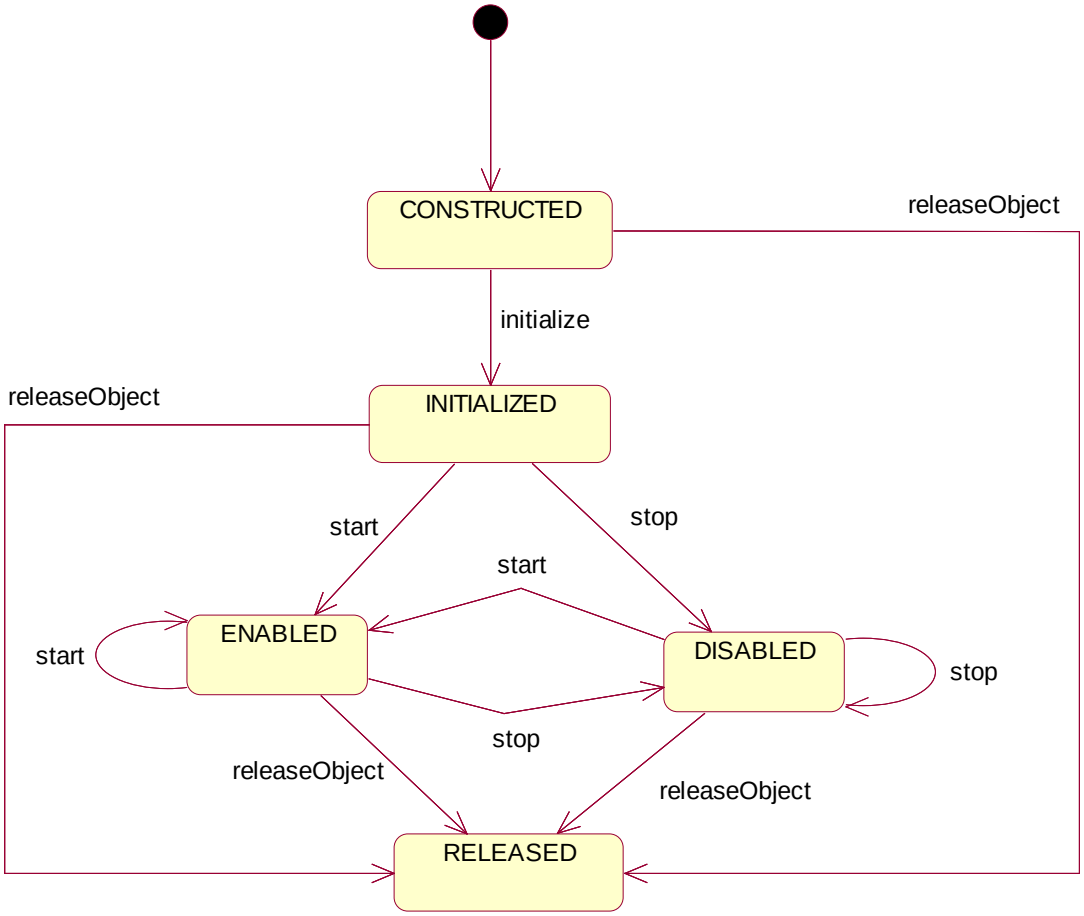

Figure 1 – GPS Device State Diagram

A.1.5 Referenced Documents

The following documents of the exact issue shown form a part of this specification to the extent specified herein.

A.1.5.1 Government Documents

A.1.5.1.1 Specifications

A.1.5.1.1.1 Federal Specifications

None

A.1.5.1.1.2 Military Specifications

None

A.1.5.1.2 Other Government Agency Documents

[1] JTRS Standard, “JTRS CORBA Types,” JPEO, Version 1.0.2.
[2] “GPS User Equipment Interface Control Document for the RS-232/RS-422 Interface of DOD Standard GPS UE Radio Receivers,” IAW-ICD-GPS-153, Version C, December 12, 2002.
[3] JTRS Standard, “Software Communications Architecture,” JPEO, Version 2.2.2.

A.1.5.2 Commercial Standards

None

A.2 SERVICES

A.2.1Provide Services

None

A.2.2Use Services

None

A.2.3Interface Modules

None

A.2.4Sequence Diagrams

None

A.3 SERVICE PRIMITIVES AND ATTRIBUTES

None

A.4 IDL

A.4.1GpsDevice IDL

```
/*
**  GpsDevice.idl
*/

#ifndef __GPSDEVICE_DEFINED
#define __GPSDEVICE_DEFINED

#ifndef __JTRSCORBATYPES_DEFINED
#include "JtrsCorbaTypes.idl"
#endif

module Gps {

        struct GpsTime {
                /* The nanosecond attribute is used to indicate current nanosecond from GPS time. */
                unsigned short nanosecond;
                /* The microsecond attribute is used to indicate current microsecond from GPS time. */
                unsigned short microsecond;
                /* The millisecond attribute is used to indicate current millisecond from GPS time. */
                unsigned short millisecond;
                /* The second attribute is used to indicate current second from GPS time. */
                unsigned short second;
                /* The minute attribute is used to indicate current minute from GPS time. */
                unsigned short minute;
                /* The hour attribute is used to indicate current hour from GPS time. */
                unsigned short hour;
                /* The day attribute is used to indicate current date of week from GPS time. */
                unsigned short day;
                /* The month attribute is used to indicate current month from GPS time. */
                unsigned short month;
                /* The year attribute is used to indicate current year from GPS time. */
                unsigned short year;
        };

        struct GpsVel {
```

```
        /* Velocity in East direction */
        float velocityEast;
        /* Velocity in North direction */
        float velocityNorth;
        /* Velocity in Up direction */
        float velocityUp;
        /* Attitude pitch position */
        float attitudePitch;
        /* Attitude Roll position */
        float attitudeRoll;
        /* A GPS system time in second. */
        double gpsTimeTag;
        /* UTC time tag in second */
        double utcTimeTag;
        /* True heading direction */
        float trueHeading;
        /* Altitude data. */
        float altitude;
};


struct GpsFomData {
        /* A GPS system time in second. */
        double gpsTimeTag;
        /* A figure of merit is output as integer value between 1 and 9. */
        unsigned short fom;
        /* A time figure of merit is output as integer value between 1 and 9. */
        unsigned short tfom;
        /* indicates a valid velocity. */
        boolean validVelocity;
        /* Present GPS Receiver track output as degrees reference to true. */
        float track;
        /* Present GPS Receiver track output as degrees reference to magnetic. */
        unsigned short trackUnit;
        /* Present GPS Receiver track output as degrees reference to grid north. */
        unsigned short northReference;
        /* Estimate of horizontal error. */
        float ehe;
        /* A unit EHE is determined by values of EHE units
          (meters, feet, yards, kilometers, miles, nautical miles) */
        unsigned short eheUnit;
```

```
                    /* Estimate position error. */
                    float epe;
                    /* A unit EPE is determined by values of EPE units
                      (meters, feet, yards, kilometers, miles, nautical miles) */
                    unsigned short epeUnit;
                    /* GPS receiver modes (continuous, fix, averaging, time only, standby,
                      2D training, 3D training, rehearsal training) */
                    unsigned short gpsMode;
                    /* Estimate velocity Error. */
                    float eve;
                    /* A unit EVE is determined by values of EVE units
                      (meters, feet, yards, kilometers, miles, nautical miles) */
                    unsigned short eveUnit;
                    /* Datum Identification. */
                    unsigned short datumId;
            };

};

#endif // __GPSDEVICE_DEFINED
```

A.5 UML

None

A.5.1Enumerations

None

A.5.2Exceptions

None

A.5.3Structures

A.5.3.1 Gps::GpsTime Structure

The *GpsTime* structure defines the GPS Time.

```
struct GpsTime {
        /* The nanosecond attribute is used to indicate current nanosecond from GPS time. */
        unsigned short nanosecond;
        /* The microsecond attribute is used to indicate current microsecond from GPS time. */
        unsigned short microsecond;
        /* The millisecond attribute is used to indicate current millisecond from GPS time. */
        unsigned short millisecond;
        /* The second attribute is used to indicate current second from GPS time. */
        unsigned short second;
        /* The minute attribute is used to indicate current minute from GPS time. */
        unsigned short minute;
        /* The hour attribute is used to indicate current hour from GPS time. */
        unsigned short hour;
        /* The day attribute is used to indicate current date of week from GPS time. */
        unsigned short day;
        /* The month attribute is used to indicate current month from GPS time. */
        unsigned short month;
        /* The year attribute is used to indicate current year from GPS time. */
        unsigned short year;
};
```

Struct	Attributes	Description	Type	Valid Range
GpsTime	nanosecond	The nanosecond attribute is used to indicate current nanosecond from GPS time.	unsigned short	0-999
	microsecond	The microsecond attribute is used to indicate current microsecond from GPS time.	unsigned short	0-999
	millisecond	The millisecond attribute is used to indicate current millisecond from GPS time.	unsigned short	0-999
	second	The second attribute is used to indicate current second from GPS time.	unsigned short	0-59
	minute	The minute attribute is used to indicate current minute from GPS time.	unsigned short	0-59
	hour	The hour attribute is used to indicate current hour from GPS time.	unsigned short	0-23

Struct	Attributes	Description	Type	Valid Range
	day	The day attribute is used to indicate current date of month from GPS time.	unsigned short	1-31
	month	The month attribute is used to indicate current month from GPS time.	unsigned short	1-12
	year	The year attribute is used to indicate current year from GPS time.	unsigned short	00-99

A.5.3.2 Gps::GpsVel Structure

The *GpsVel* structure defines the GPS altitude, time tag, and velocity.

```
struct GpsVel {
        /* Velocity in East direction */
        float velocityEast;
        /* Velocity in North direction */
        float velocityNorth;
        /* Velocity in Up direction */
        float velocityUp;
        /* Attitude pitch position */
        float attitudePitch;
        /* Attitude Roll position */
        float attitudeRoll;
        /* A GPS system time in second. */
        double gpsTimeTag;
        /* UTC time tag in second */
        double utcTimeTag;
        /* True heading direction */
        float trueHeading;
        /* Altitude data. */
        float altitude;
};
```

Struct	Attributes	Type	Unit	Valid Range	Description
GpsVel	velocityEast	float	meter/second	Not Applicable	Velocity in the East direction
	velocityNorth	float	meter/second	Not Applicable	Velocity in the North direction
	velocityUp	float	meter/second	Not Applicable	Velocity in the Up direction
	attitudePitch	float	radian	0 - 2π	Attitude pitch position
	attitudeRoll	float	radian	0 - 2π	Attitude roll position
	gpsTimeTag	double	second	Not Applicable	GPS system time in seconds
	utcTimeTag	double	second	Not Applicable	Universal Time Coordinate (UTC) time tag in seconds
	trueHeading	float	radian	0 - 2π	True heading direction
	altitude	float	meter	Not Applicable	Altitude data in Ellipsoid.

A.5.3.3 Gps::GpsFomData Structure

The *GpsFomData* structure defines the GPS position and time errors, the GPS mode, and datum identification. All data is with respect to the user selected datum.

```
struct GpsFomData {
        /* A GPS system time in second. */
        double gpsTimeTag;
        /* A figure of merit is output as integer value between 1 and 9. */
        unsigned short fom;
        /* A time figure of merit is output as integer value between 1 and 9. */
        unsigned short tfom;
        /* indicates a valid velocity. */
        boolean validVelocity;
        /* Present GPS Receiver track output as degrees reference to true. */
        float track;
        /* Present GPS Receiver track output as degrees reference to magnetic. */
        unsigned short trackUnit;
        /* Present GPS Receiver track output as degrees reference to grid north. */
        unsigned short northReference;
        /* Estimate of horizontal error. */
        float ehe;
        /* A unit EHE is determined by values of EHE units
          (meters, feet, yards, kilometers, miles, nautical miles) */
        unsigned short eheUnit;
        /* Estimate position error. */
        float epe;
        /* A unit EPE is determined by values of EPE units
          (meters, feet, yards, kilometers, miles, nautical miles) */
        unsigned short epeUnit;
        /* GPS receiver modes (continuous, fix, averaging, time only, standby,
          2D training, 3D training, rehearsal training) */
        unsigned short gpsMode;
        /* Estimate velocity Error. */
        float eve;
        /* A unit EVE is determined by values of EVE units
          (meters, feet, yards, kilometers, miles, nautical miles) */
        unsigned short eveUnit;
        /* Datum Identification. */
        unsigned short datumId;
};
```

Struct	Attributes	Type	Unit	Valid Range	Description
GpsFomData	gpsTimeTag	double	second	Not Applicable	GPS system time
	fom	unsigned short	Not Applicable	See Appendix A.D	Figure of Merit
	tfom	unsigned short	Not Applicable	See Appendix A.E	Time Figure of Merit
	validVelocity	boolean	Not Applicable	True = valid; False = invalid	Indicates whether the velocity is valid

Struct	Attributes	Type	Unit	Valid Range	Description
	track	float	Applicable unit is set by the trackUnit attribute	0-6400 mils or 0-360 degree	Present GPS receiver track in the units indicated by track_unit and the reference indicated by north_reference.
	trackUnit	unsigned short	mils or degrees	0=mils, 1=degree	This attribute indicates the unit (either mils or deg) for the track attribute.
	northReference	unsigned short	Not Applicable	0 = true, 1 = magnetic, 2 = grid	This attribute indicates the north reference (either true, magnetic, or grid) for the track attribute.
	ehe	float	Applicable unit set by the eheUnit attribute.	Not Applicable	Estimate Horizontal Error
	eheUnit	unsigned short	meters, feet, yards, kilometers, miles, or nautical miles	0=m, 1=ft, 2=yd, 3=km, 4=mi, 5=nm	This attribute sets the unit (either meter, feet, yards, kilometers, miles, or nautical miles) for the ehe attribute.
	epe	float	Applicable unit set by the epeUnit attribute.	Not Applicable	Estimate Position Error
	epeUnit	unsigned short	meters, feet, yards, kilometers, miles, or nautical miles	0=m, 1=ft, 2=yd, 3=km, 4=mi, 5=nm	This attribute sets the unit (either meters, feet, yards, kilometers, miles, or nautical miles) for the epe attribute.

Struct	Attributes	Type	Unit	Valid Range	Description
	gpsMode	unsigned short	Not Applicable	0=continuous, 1=fix, 2=averaging, 3=time only, 4=standby, 5=2D training, 6=3D training, 7=rehearsal training	This attribute sets the GPS receiver mode (either continuous, fix, averaging, time only, standby, 2 dimensional training, 3 dimensional training, or rehearsal training).
	eve	float	Applicable unit set by the eveUnit attribute.	Not Applicable	Estimated Velocity Error
	eveUnit	unsigned short	meters/second, feet/second, yards/second, kilometers/second, miles/second, nautical miles/second	0=m/s, 1=ft/s, 2=yd/s, 3=km/s, 4=mi/s, 5=nm/s	This attribute sets the unit (either meters/second, feet/second, yards/second, kilometers/second, miles/second, or nautical miles/second) for the eve attribute.
	datumId	unsigned short	Not Applicable	See Appendix A.F	Datum Identification

APPENDIX A.A ABBREVIATIONS AND ACRONYMS

API	Application Program Interface
CF	Core Framework
D	Dimensional
DEL	Data Escape Link
DOD	Department of Defense
EHE	Estimated Horizontal Error
EPE	Estimated Position Error
EVE	Estimated Velocity Error
FOM	Figure of Merit
GPS	Global Positioning System
GSSIP	GPS Standard Serial Interface Protocol
HW	Hardware
ID	Identification
IDL	Interface Definition Language
JPEO	JTRS Program Executive Office
JTR	Joint Tactical Radio
JTRS	Joint Tactical Radio System
NIMA	National Imagery and Mapping Agency
PVT	Position, Velocity, Time
SOH	Start of Header
TFOM	Time Figure of Merit
UML	Unified Modeling Language
UTC	Universal Time Coordinate

APPENDIX A.B PERFORMANCE SPECIFICATION

Not applicable.

APPENDIX A.C GPS MESSAGE FORMATS

The following GPS message format is for reference only.

The GPS data and commands are transmitted using GPS Standard Serial Interface Protocol (GSSIP) formatted into messages. Messages contain a header portion and may or may not contain a data portion. The maximum number of words a message may contain is 106 (five header portion words and 101 data portion words). The header portion contains four header words plus one header checksum #1 word (words 1 through 5). The data portion may contain a maximum of 100 data words plus one data checksum #2 word (words 6 through 5+N, where $1 \leq N \leq 100$). AFigure 2 describes an example Message format.

A header portion contains five words:

- Word 1: Data Escape Link (DEL) or Start of Header (SOH).
- Word 2: Message ID. A message ID of Zero indicates the Universal Reset message.
- Word 3: Word Count. It indicates the number of message words contained in the data portion of the message excluding the checksum #2 word.
- Word 4: Flag. It provides flag bits necessary for protocol operation.
- Word 5, Checksum #1. It is used to valid the header portion of the message.

Data Portion contains up to a maximum of 100 data message words.

- Word 6 to 5+N: N Data Words. It indicates number of message data words. This number is the same as Message word count.
- Word 6+N, Checksum #2. This checksum is used to validate the data portion of the message.

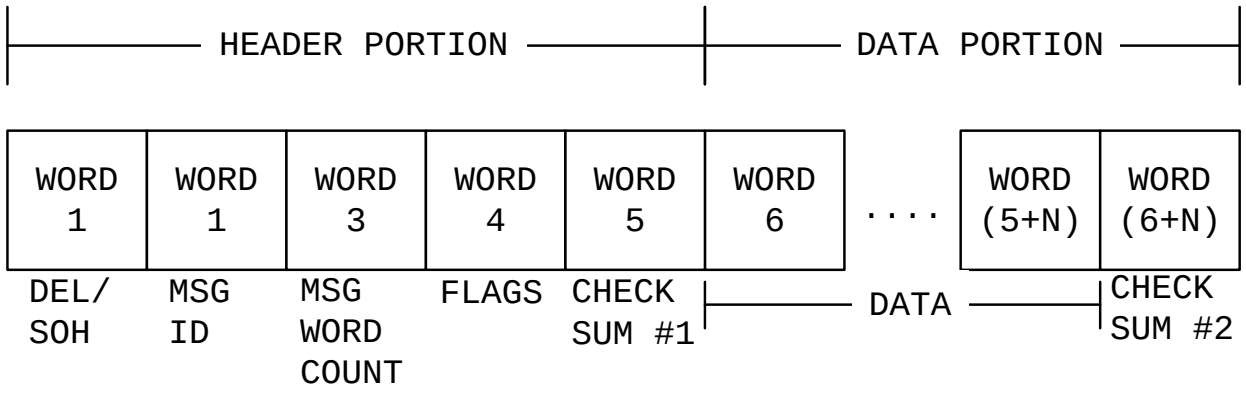

Figure 2 – GPS Interface Message Format

APPENDIX A.D FIGURE OF MERIT (FOM)

Table 1 – Figure of Merit

FOM Integer	EPE in meters, 1 sigma
0	Not used
1	EPE ≤ 25
2	25 < EPE ≤ 50
3	50 < EPE ≤ 75
4	75 < EPE ≤ 100
5	100 < EPE ≤ 200
6	200 < EPE ≤ 500
7	500 < EPE ≤ 1000
8	1000 ≤ EPE ≤ 5000
9 Note 1	5000 < EPE
10-15	Not used

Note 1: A FOM=9 with Nav Converged=False or Nav Data Not Valid set to 1 indicates that there is no position solution, otherwise, a FOM=9 indicates an EPE > 5000 meters.

APPENDIX A.E TIME FIGURE OF MERIT (TFOM)

Table 2 – Time Figure of Merit

TFOM Value	Estimated Time Error (ETE)
0	Not Used
1	≤ 1 nanosecond (nsec)
2	> 1 nsec ≤ 10 nsec
3	> 10 nsec ≤ 100 nsec
4	> 100 nsec ≤ 1 μsec
5	> 1 μsec ≤ 10 μsec
6	> 10 μsec ≤ 100 μsec
7	> 100 μsec ≤ 1msec
8	> 1 msec ≤ 10 msec
9	> 10 msec

Note 1: A TFOM=9 with Nav Converged=False or Nav Data Not Valid set to 1 indicates that there is no GPS time solution, otherwise, a TFOM=9 indicates an ETE > 10 ms.

APPENDIX A.F DATUM IDENTIFIER

The following tables are taken from the "*GPS User Equipment Interface Control Document for the RS-232/RS-422 Interface of DOD Standard GPS UE Radio Receiver*s" [2].

Table 3 – Datum Number / Identifier List

No.	NIMA DATUM Code	Datum	No.	NIMA DATUM Code	Datum
0		Note 1*	27	LUZ-A	LUZON
1	ARF-M	ARC 1950	28	MAS	MASSAWA
2	ARS	ARC 1960	29	MER	MERCHICH
3	AUA	AUSTRALIAN GEODETIC 1966	30	MIN-B	MINNA
4	AUG	AUSTRALIAN GEODETIC 1984	31	NAH-C	NARHWAN
5	BOO	BOGOTA OBSERVATORY	32	NAR	NORTH AMERICAN 1983
6	CAI	CAMPO INCHAUSPE	33	NAS-C	NORTH AMERICAN 1927 (CONUS)
7	CAP	CAPE	34	NAS-D	ALASKA
8	CGE	CARTHAGE	35	NAS-E	CANADA
9	CHI	CHATHAM 1971	36	NAS-N	CENTRAL AMERICA
10	CHU	CHUA ASTRO	37	OEG	OLD EGYPTIAN
11	COA	CORREGO ALEGRE	38	OGB-M	ORDINANCE SURVEY OF GREAT BRITAIN 1936
12	EUR-A	EUROPEAN 1950	39	OHA-M	OLD HAWAIIAN
13	EUR-E	CYPRUS	40	PIT	PITCAIRN ASTRO 1967
14	EUR-F	EGYPT	41	QAT	QATAR NATIONAL
15	EUR-H	IRAN	42	QUO	QORNOQ
16	EUR-J	SICILY	43	SAN-M	SOUTH AMERICA 1969
17	EUS	EUROPEAN 1979	44	SCK	SCHWARZECK
18	FAH	OMAN	45	TIL	TIMBALAI 1948
19	GAA	GANDAJIKA BASE	46	TOY-M	TOKYO
20	GEO	GEODETIC DATUM 1949	47	WGD	WGS 1984
21	HJO	HJORSEY 1955	48	WGS	WGS 1972
22	INF-A	INDIAN (THAILAND AND VIETNAM)	49	ZAN	ZANDERIJ
23	IND-I	INDIAN (BANGLADESH, INDIA, AND NEPAL)	50	Note 2*	USER DEFINED 1
24	IRL	IRELAND 1965	51	Note 2*	USER DEFINED 2
25	KEA	KERTAU 1948	61	BBOHM	Note 3*
26	LIB	LIBERIA 1964			

Note 1*: A "0" in the Datum Number field of an output message indicates that the Datum entered using the NIMA Datum Code field does not have a number assigned. Refer to the NIMA Datum Code field to identify the Datum.

Note 2*: The Power On default value for the NIMA style datum codes associated with UDDs 1 and 2 is "USER1" and "USER2", respectively, and will be changed by entering a new ASCII string via words 5-7 of Message 5067 or keypad/display, if applicable.

Note 3*: Refer to the receiver specification to determine if this datum transformation is implemented.

Table 4 – Datum Description List

NIMA Datum Code	Datum Description	NIMA Datum Code	Datum Description	NIMA Datum Code	Datum Description
ADI-A	ADIND-Ethi/Clk80	CHI	Chatham 71/Int24	IND-I	IND-In-Npl/Evr56
ADI-B	ADIND-Suda/Clk80	CHU	Chua Astro/Int24	IND-P	IND-Pakis/Evrst
ADI-C	ADIND-Mali/Clk80	COA	Corrego Al/Int24	INF-A	IND54-Thai/Evr30
ADI-D	ADIND-Sene/Clk80	DAL	DabolaGuin/Clk80	ING-A	IND60-Viet/Evr30
ADI-E	ADIND-Burk/Clk80	DID	DecepIslan/Clk80	ING-B	IND60-CSI/Evr30
ADI-F	ADIND-Camr/Clk80	DOB	GuadalCana/Int24	INH-A	IND75-Thai/Evr30
ADI-M	ADIND-Mean/Clk80	EAS	EasterIs67/Int24	INH-A1	IND75-Thai/Evr30
AFG	Somalia/Krs40	ENW	Wake-Eni60/Hou60	IRL	Ireland 65/MAiry
AIA	AntiguaIsl/Clk80	EST	Estonia37/Bsl41	ISG	SGeorgiaIs/Int24
AIN-A	Ain70-Bahr/Int24	EUR-A	Eur50-West/Int24	IST	DiegoGarci/Int24
AIN-B	AinElABD70/Int24	EUR-B	Eur50-Grce/Int24	JOH	JohnsIs61/Int24
AMA	AmerSamo62/Clk66	EUR-C	Eur50-Norw/Int24	KAN	Kandawala/Evr30
ANO	AnnaAstr65/AusNa	EUR-D	Eur50-Spai/Int24	KEA	Kertau 48/Evr48
ARF-A	ARC50-Bots/Clk80	EUR-E	Eur50-Cypr/Int24	KEG	KergueIs49/Int24
ARF-B	ARC50-Leso/Clk80	EUR-F	Eur50-Egyp/Int24	KGS	SouthKorea/WGS84
ARF-C	ARC50-Mala/Clk80	EUR-G	Eur50-Engl/Int24	KUS	KusaiAst51/Int24
ARF-D	ARC50-Swaz/Clk80	EUR-H	Eur50-Iran/Int24	LCF	LC5Astro61/Clk66
ARF-E	ARC50-Zair/Clk80	EUR-I	Eur50-Sard/Int24	LEH	Leigon/Clk80
ARF-F	ARC50-Zamb/Clk80	EUR-J	Eur50-Scly/Int24	LIB	Liberia 64/Clk80
ARF-G	ARC50-Zimb/Clk80	EUR-K	Eur50-UK/Int24	LUZ-A	Luzon-Phil/Clk66
ARF-H	ARC50-Buru/Clk80	EUR-L	Eur50-Malt/Int24	LUZ-B	Luzon-Mind/Clk66
ARF-M	ARC 1950-M/Clk80	EUR-M	Eur50-Mean/Int24	MAS	Massawa/Bsl41
ARS-A	ARC60-Keny/Clk80	EUR-S	Eur50-MidE/Int24	MER	Merchich/Clk80
ARS-B	ARC60-Tanz/Clk80	EUR-T	Eur50-Tuni/Int24	MID	MdwayAst61/Int24
ARS-M	ARC 1960/Clk80	EUS	European79/Int24	MIK	MaheIs71/Clk80
ASC	AscensionI/Int24	FAH	Oman/Clk80	MIN-A	Minna-Came/Clk80
ASM	MontseIs58/Clk80	FLO	Azores39/Int24	MIN-B	Minna/Clk80
ASQ	MarcusIs52/Int24	FOT	Ft.Tom55/Clk80	MOD	Rome40-Sar/Int24
ATF	Astro45Iwo/Int24	GAA	Gan 70/Int24	MPO	Mpor-Gabon/Clk80
AUA	Austral.66/AusNa	GEO	Geo Dtm 49/Int24	MVS	VitiLev16/Clk80
AUG	Austral.84/AusNa	GIZ	GizoIs68/Int24	NAH-A	NahrwanMIs/Clk80
BAT	Djakarta/Bsl41	GRA	Azores48/Int24	NAH-B	NahrwanUAE/Clk80
BER	Bermuda57/Clk66	GSE	Kalimanta/Bsl41	NAH-C	Nahrwan/Clk80
BID	BissauGuin/Int24	GUA	Guam63/Clk66	NAP	Naparima/Int24
BOO	Bogota Obs/Int24	HEN	Afghanist/Int24	NAR-A	NA83Alaska/GRS80
BUR	BukitRimpa/Bsl41	HER	Yugoslav/Bsl41	NAR-B	NA83Canada/GRS80
CAC	CapeCanav/Clk66	HIT	SoChile63/Int24	NAR-C	NA83CONUS/GRS80
CAI	Campo Inch/Int24	HJO	Hjorsey 55/Int24	NAR-D	NA83Mexico/GRS80
CAO	CantAstr66/Int24	HKD	HongKong63/Int24	NAR-E	NA83Aleuti/GRS80
CAP	Cape/Clk80	HTN	HuTzuShan/Int24	NAR-H	NA83Hawaii/GRS80
CAZ	CampMcMurd/Int24	IBE	Bellevue/Int24	NAS-A	NA27EastUS/Clk66
CCD	S-JTSKCzec/Bsl41	IDN	IND74-Indo/Ind74	NAS-B	NA27WestUS/Clk66
CGE	Carthage/Clk80	IND-B	IND-Bangla/Evr30	NAS-C	NA27CONUS/Clk66

NIMA Datum Code	Datum Description	NIMA Datum Code	Datum Description	NIMA Datum Code	Datum Description
NAS-D	NA27Alaska/Clk66	PLN	CanaryIs/Int24	SGM	SelGran38/Int24
NAS-E	NA27Canada/Clk66	POS	PortoSan36/Int24	SHB	AstroDos71/Int24
NAS-F	NA27AlbBrC/Clk66	PRP-A	PSAm56-Bol/Int24	SIR	SIRGASSAm/GRS80
NAS-G	NA27ECanad/Clk66	PRP-B	PSAm56-NCh/Int24	SOA	Sasia-Sing/MFi60
NAS-H	NA27ManOnt/Clk66	PRP-C	PSAm56-SCh/Int24	SPK-A	Pulk42-Hun/Krs40
NAS-I	NA27NWTSas/Clk66	PRP-D	PSAm56-Col/Int24	SPK-B	Pulk42-Pol/Krs40
NAS-J	NA27Yukon/Clk66	PRP-E	PSAm56-Ecu/Int24	SPK-C	Pulk42-Cze/Krs40
NAS-L	NA27Mexico/Clk66	PRP-F	PSAm56-Guy/Int24	SPK-D	Pulk42-Lat/Krs40
NAS-N	NA27CentAm/Clk66	PRP-G	PSAm56-Per/Int24	SPK-E	Pulk42-Kaz/Krs40
NAS-O	NA27CanalZ/Clk66	PRP-H	PSAm56-Ven/Int24	SPK-F	Pulk42-Alb/Krs40
NAS-P	NA27Caribb/Clk66	PRP-M	ProSoAm56/Int24	SPK-G	Pulk42-Rom/Krs40
NAS-Q	NA27Bahama/Clk66	PTB	Pt58-Burk/Clk80	SRL	SierrLeo60/Clk80
NAS-R	NA27SanSal/Clk66	PTN	PtN48-Cong/Clk80	TAN	TananObs25/Int24
NAS-T	NA27Cuba/Clk66	PUK	Pulkovo42/Krs40	TDC	TristAst68/Int24
NAS-U	NA27Greenl/Clk66	PUR	PuertoRico/Clk66	TIL	Timbalai48/Evrst
NAS-V	NA27EaluIs/Clk66	QAT	Qatar Nat/Int24	TOY-A	Tokyo-Japa/Bsl41
NAS-W	NA27WaluIs/Clk66	QUO	Qornoq/Int24	TOY-B	Tokyo-Kore/Bsl41
NSD	No Sah 59/Clk80	REU	Reunion/Int24	TOY-B1	Tokyo-Kore/Bs141
OEG	OldEgypt07/Hlm06	SAE	SantoIs65/Int24	TOY-C	Tokyo-Okin/Bsl41
OGB-A	OSGB36-Engl/Airy	SAN-A	SAm69-Arge/SAm69	TOY-M	Tokyo-Mean/Bsl41
OGB-B	OSGB36-EnWa/Airy	SAN-B	SAm69-Boli/SAm69	TRN	AstTernI61/Int24
OGB-C	OSGB36-Scot/Airy	SAN-C	SAm69-Braz/SAm69	Note 1*	USER1
OGB-D	OSGB36-Wale/Airy	SAN-D	SAm69-Chil/SAm69	Note 1*	USER2
OGB-M	OrdSrvGB36/Airy	SAN-E	SAm69-Colo/SAm69	Note 1*	USER3
OHA-A	OldHaw-Haw/Clk66	SAN-F	SAm69-Ecua/SAm69	Note 1*	USER4
OHA-B	OldHaw-Kau/Clk66	SAN-G	SAm69-Guya/SAm69	Note 1*	USER5
OHA-C	OldHaw-Mau/Clk66	SAN-H	SAm69-Para/SAm69	Note 1*	USER6
OHA-D	OldHaw-Oah/Clk66	SAN-I	SAm69-Peru/SAm69	VOI	Voirol1874/Clk80
OHA-M	OldHaw-Mea/Clk66	SAN-J	SAm69-Balt/SAm69	VOR	Voirol1960/Clk80
OHI-A	OldHaw-Haw/Int24	SAN-K	SAm69-Trin/SAm69	WAK	WakeIAst52/Int24
OHI-B	OldHaw-Kau/Int24	SAN-L	SAm69-Vene/SAm69	WGD	WGS-84
OHI-C	OldHaw-Mau/Int24	SAN-M	So Amer 69/SAm69	WGS	WGS-72
OHI-D	OldHaw-Oah/Int24	SAO	SaoBraz-Az/Int24	YAC	Yacare-Uru/Int24
OHI-M	OldHaw-Mea/Int24	SAP	SapperHill/Int24	ZAN	Zanderij/Int24
PIT	Pitcairn67/Int24	SCK	Schwarzeck/Bsl41		

Note 1*: The Power On default value for the NIMA Datum Code associated with UDDs 1 through 6 is "USER1" through "USER6", respectively, and will be changed by entering a new ASCII string via words 5-7 of Message 5067 or keypad/display, if applicable.

B. LATITUDE/LONGITUDE EXTENSION

B.1 INTRODUCTION

The *Latitude/Longitude (LatLong) Extension* is based upon the *GPS Device API*. It extends the functionality of the *GPS Device* to report the GPS Position, Velocity, and Time (PVT) data using radian latitude and longitude coordinates.

B.1.1 Overview

This extension contains as follows:

a. Section B.1, *Introduction*, of this document contains the introductory material regarding the overview, Service Layer description, Modes, States and Referenced Documents of this document.

b. Section B.2, *Services*, provides summary of service interface uses, interface for each device component, port connections, and sequence diagrams.

c. Section B.3, *Service Primitives and Attributes*, specifies the operations that are provided by the *Latitude/Longitude Extension*.

d. Section B.4, *IDL*.

e. Section B.5, *UML*.

f. Appendix B.A, *Abbreviations and Acronyms*.

g. Appendix B.B, *Performance Specification.*

B.1.2 Service Layer Description

B.1.2.1 GPS Device Latitude/Longitude Extension Port Connections

The following figure shows the port connections for the *GPS Device* with latitude/longitude reporting capabilities. All port names are for reference only.

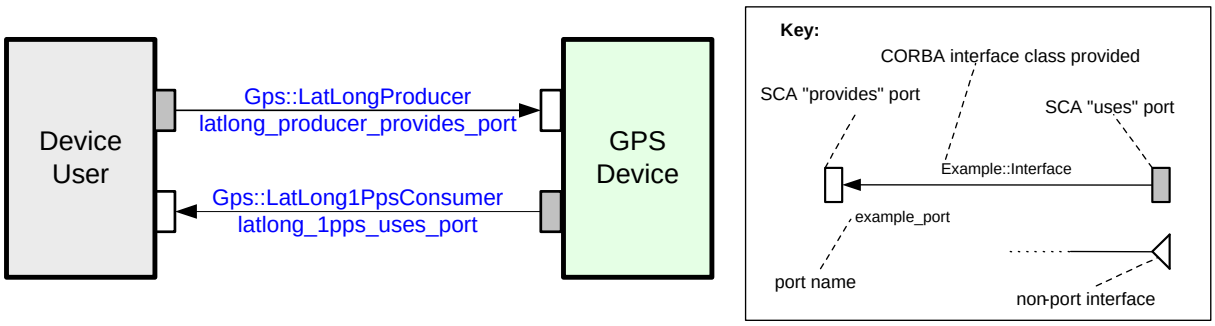

Figure 3 – Latitude/Longitude Extension Port Diagram

***GPS Device* Latitude/Longitude Extension Provides Ports Definitions**

latlong_producer_provides_port is provided by *GPS Device* to produce PVT data.

***GPS Device* Latitude/Longitude Extension Uses Ports Definitions**

latlong_1pps_uses_port is used by the *GPS Device* to push PVT data to the *Device User* when a 1 pulse per second (pps) event occurs.

B.1.3 Modes of Service

Not applicable.

B.1.4 Service States

There are no changes from A. GPS Device.

B.1.5 Referenced Documents

There are no changes from A. GPS Device.

B.2 SERVICES

B.2.1 Provide Services

The *Latitude/Longitude Extension* Provide Service consists of the following service ports, interfaces, and primitives, which can be called by other client components:

Table 5 – Latitude/Longitude Extension Provide Service Interface

Service Group (Port Name)	**Service (Interface Provided)**	**Primitives (Provided)**
latlong_producer_provides_port	GpsLatLongDataProducer	readLatLong()

B.2.2 Use Services

The *Latitude/Longitude Extension* Use Service set consists of the following service ports, interfaces, and primitives. Since the *Latitude/Longitude Extension* acts as a client with respect to these services from other components, it is required to connect these ports with corresponding service ports applied by the server component.

Table 6 – Latitude/Longitude Extension Use Service Interface

Service Group (Port Name)	**Service (Interface Used)**	**Primitives (Used)**
latlong_1pps_uses_port	GpsLatLong1PpsConsumer	pushLatLong1Pps()

B.2.3 Interface Modules

B.2.3.1 Latitude/Longitude Extension

The *Latitude/Longitude Extension* class diagram is shows below.

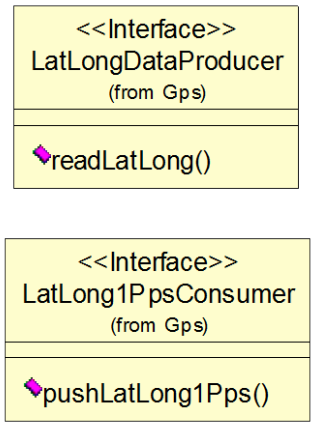

Figure 4 – Latitude/Longitude Extension Class Diagram

B.2.3.1.1 LatLong1PpsConsumer Interface Description

The interface design of the *LatLong1PpsConsumer* is shown in the interface class diagram below. It provides the ability to transfer GPS data to the Device User when a 1pps event occurs.

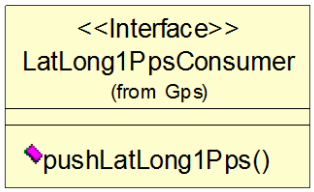

Figure 5 – LatLong1PpsConsumer Interface Class Diagram

B.2.3.1.2 LatLongDataProducer Interface Description

The interface design of the *LatLongDataProducer* is shown in the interface class diagram. It is provides the capability to read GPS PVT data received from the GPS Receiver Hardware.

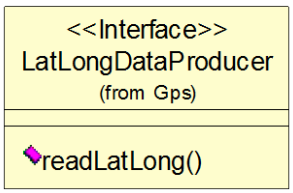

Figure 6 – LatLongDataProducer Interface Class Diagram

B.2.4 Sequence Diagrams

B.2.4.1 Latitude/Longitude 1pps Event Sequence

Description

The *Latitude/Longitude 1pps Event* Sequence specifies the sequence that occurs when a *GPS Device* delivers a 1pps PVT Event. Multiple *Device Users* can connect to a *GPS Device* in order to receive a 1pps PVT event. When the Timebase Device Hardware produces a 1pps event, each *Device User* is issued a *pushLatLong1Pps* to signal the event. Note: The Timebase Device Hardware is the hardware that produces the 1pps event.

Pre-conditions

The *GPS Device* is in the ENABLED state.

Post-conditions

The *GPS Device* has pushed a 1pps event to the *Device Users*.

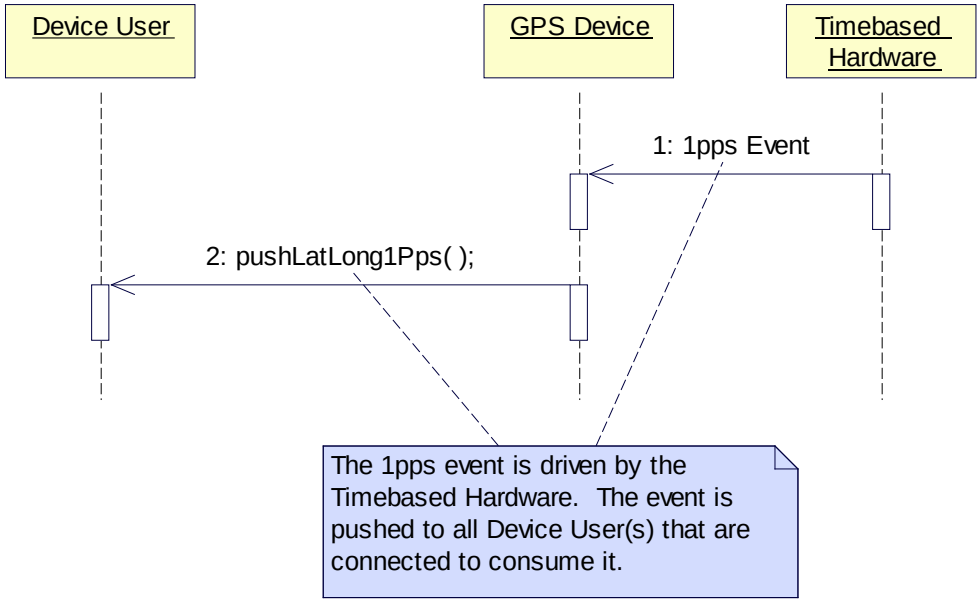

Figure 7 – Latitude/Longitude 1pps Event Sequence Diagram

B.2.4.2 Latitude/Longitude Non-Blocking Read Sequence

Description

The *Latitude/Longitude Non-Blocking Read* Sequence shows the events that occur when a *GPS Device* is issued a non-blocking read. To support this operation, PVT data is read from the GPS Receiver Hardware whenever a Timebase Hardware 1pps event occurs. The *Device User* issues a read command passing in "block" = FALSE to indicate this is a non-blocking read. The PVT data is then read returned directly to the *Device User*. Note: The Timebase Device Hardware is the hardware that produces the 1pps event.

Pre-conditions

The *GPS Device* is in the ENABLED state.

Post-conditions

The *GPS Device* has returned PVT data to the *Device User*.

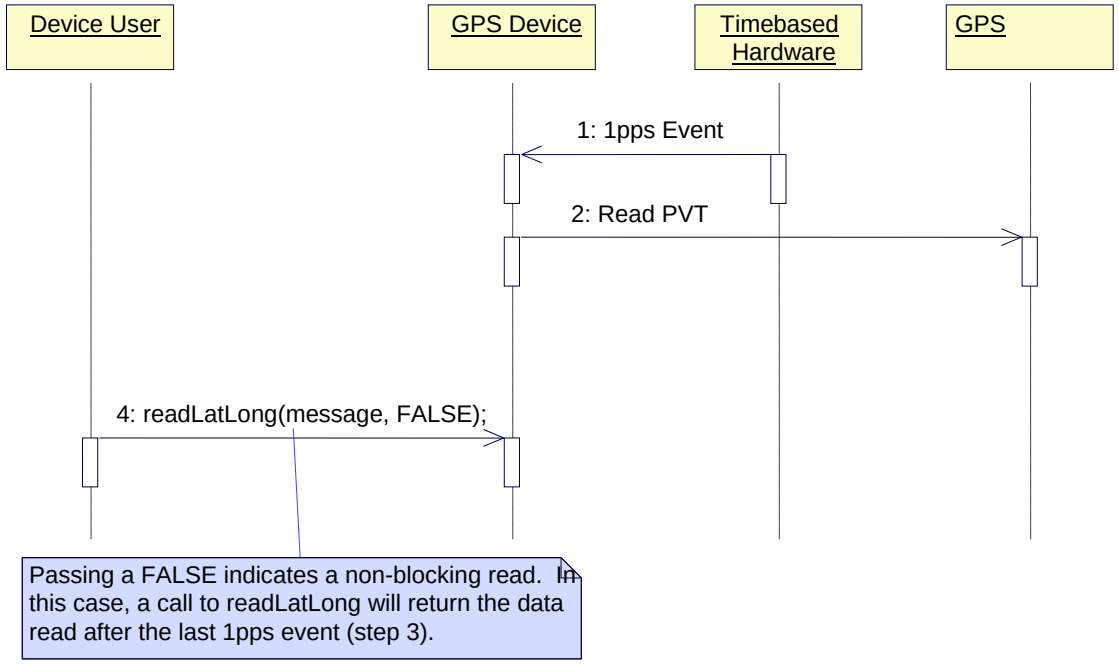

Figure 8 – Latitude/Longitude Non-Blocking Read Sequence Diagram

B.2.4.3 Latitude/Longitude Blocking Read Sequence

Description

The *Latitude/Longitude Blocking Read* Sequence shows the events that occur when a *GPS Device* is issued a blocking read. The *Device User* issues a read command passing in "block" = TRUE to indicate this is a blocking read. The call blocks until a Timebase Device Hardware 1pps occurs or a timeout threshold has been exceeded. If the Timebase Device Hardware 1pps event occurs, PVT data is read from the GPS Receiver Hardware and returned to the *Device User*. If the timeout threshold is exceeded, a *DeviceError* exception is raised and control is returned to the *Device User*. Note: The Timebase Device Hardware is the hardware that produces the 1pps event.

Pre-conditions

The *GPS Device* is in the ENABLED state.

Post-conditions

The *GPS Device* has returned PVT data to the *Device User*.

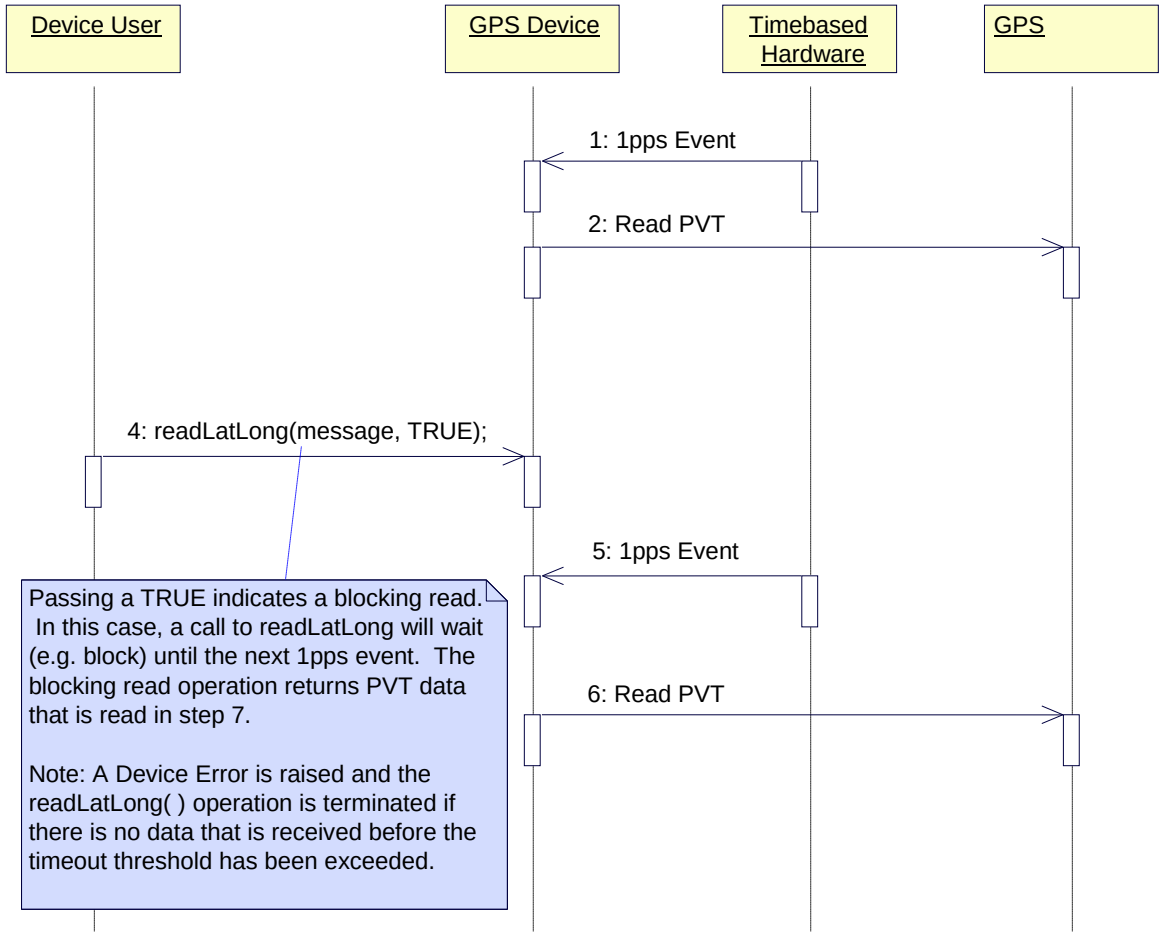

Figure 9 – Latitude/Longitude Blocking Read Sequence Diagram

B.3 SERVICE PRIMITIVES AND ATTRIBUTES

To enhance the readability of this API document and to avoid duplication of data, the type definitions of all structured types (i.e., type definitions, enumerations, exceptions, and structures) used by the Service Primitives and Attributes have been co-located in section B.5 UML. This cross-reference of types also includes any nested structures in the event of a structure of structures or an array of structures.

B.3.1 Gps::LatLongDataProducer

B.3.1.1 *readLatLong* Operation

The *readLatLong* operation is used to read PVT data received from the GPS Receiver hardware.

B.3.1.1.1 Synopsis

```
void readLatLong (
        inout GpsLatLongMessage message,
        in boolean block
        ) raises (DeviceError);
```

B.3.1.1.2 Parameters

Parameter Name	Type	Valid Range	Description
message	GpsLatLongMessage (see B.5.3.1)	Not Applicable	The GPS latitude/longitude message
block	boolean	TRUE = blocking; FALSE = non-blocking	Determines whether to perform a blocking or non-blocking read.

B.3.1.1.3 State

ENABLED CF::Device::operationalState.

B.3.1.1.4 New State

This operation does not cause a state change.

B.3.1.1.5 Return Value

None

B.3.1.1.6 Originator

Service User

B.3.1.1.7 Exceptions

Type	Description
DeviceError (see B.5.2.1)	An exception is raised when the blocking read timed out.

B.3.2 Gps::LatLong1PpsConsumer

B.3.2.1 *pushLatLong1Pps* Operation

The *pushLatLong1Pps* operation provides the ability to push GPS messages to the *GPS Device User* when a 1pps event occurs.

B.3.2.1.1 Synopsis

```
oneway void pushLatLong1Pps (
        in GpsLatLongMessage message
        );
```

B.3.2.1.2 Parameters

Parameter Name	Type	Description
message	GpsLatLongMessage (see B.5.3.1)	The GPS latitude/longitude message

B.3.2.1.3 State

ENABLED CF::Device::operationalState.

B.3.2.1.4 New State

This operation does not cause a state change.

B.3.2.1.5 Return Value

None

B.3.2.1.6 Originator

Service Provider

B.3.2.1.7 Exceptions

None

B.4 IDL

B.4.1 GpsLatLongExt IDL

```
/*
**  GpsLatLongExt.idl
*/

#ifndef __GPSLATLONGEXT_DEFINED
#define __GPSLATLONGEXT_DEFINED

#ifndef __GPSDEVICE_DEFINED
#include "GpsDevice.idl"
#endif

module Gps {

        struct GpsLatLong {
                /* Longitude data. */
                float longitude;
                /* Latitude data. */
                float latitude;
        };

        struct GpsLatLongMessage {
                /* GPS Time. */
                GpsTime time;
                /* GPS velocity data. */
                GpsVel velocity;
                GpsLatLong position;
                /* Figure of merit data. */
                GpsFomData fomData;
        };

        interface LatLongDataProducer {

                exception DeviceError {
                        /* GPS Receive error. */
                        boolean invalidHardware;
```

```
                        /* GPSDevice is in an invalid State. */
                        boolean invalidState;
                        /* GPS Time data is wrong format. */
                        boolean invalidTime;
                        /* The message string describes a reason of exception has been raised by GPSDevice. */
                        string msg;
                };

                void readLatLong (
                        inout GpsLatLongMessage message,
                        in boolean block
                        ) raises (DeviceError);
        };

        interface LatLong1PpsConsumer {

                oneway void pushLatLong1Pps (
                        in GpsLatLongMessage message
                        );
        };

};

#endif // __GPSLATLONGEXT_DEFINED
```

B.5 UML

This section contains the Device component UML diagram and the definitions of all data types referenced (directly or indirectly) by the Service Primitives and Attributes in section B.3.

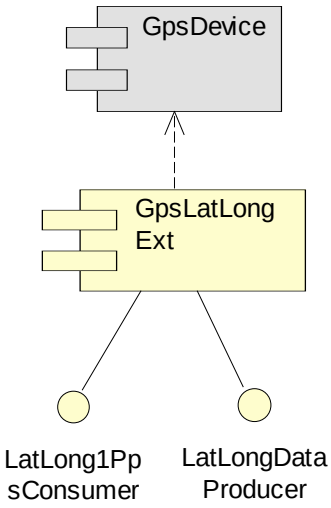

Figure 10 – Latitude/Longitude Extension Component Diagram

B.5.1 Enumerations

None

B.5.2 Exceptions

B.5.2.1 Gps::DeviceError Exception

The *DeviceError* exception is used to indicate that errors were detected during the *readLatLong* operation (see B.3.1.1).

```
exception DeviceError {
        /* GPS Receive error. */
        boolean invalidHardware;
        /* GPSDevice is in an invalid State. */
        boolean invalidState;
        /* GPS Time data is wrong format. */
        boolean invalidTime;
        /* The message string describes a reason of exception has been raised by GPSDevice. */
        string msg;
};
```

Exception	Attributes	Type	Valid Range	Description
DeviceError	invalidHardware	boolean	TRUE= timed out; FALSE=not timed out	Indicates if the blocking read timed out.
	invalidState	boolean	TRUE = invalid; FALSE=valid	Indicates if the GPS state is invalid.
	invalidTime	boolean	TRUE=invalid; FALSE=valid	Indicates if the GPS Time data is in the wrong format.
	msg	string	Not Applicable.	A message of type string indicating that the exception has occurred.

B.5.3 Structures

B.5.3.1 Gps::GpsLatLongMessage Structure

The *GpsLatLongMessage* structure defines the GPS Latitude/Longitude Message (position, velocity, time, and FOM data. This structure is used in the *readLatLong* and *pushLatLong1Pps* operations (see B.3.1.1 and B.3.2).

```
struct GpsLatLongMessage {
        /* GPS Time. */
        GpsTime time;
        /* GPS velocity data. */
        GpsVel velocity;
        GpsLatLong position;
        /* Figure of merit data. */
        GpsFomData fomData;
};
```

struct	Attributes	Type	Description
GpsLatLong Message	time	GpsTime (see A.5.3.1)	This attribute indicates the type of time.
	velocity	GpsVel (see A.5.3.2)	This attribute indicates the type of velocity.
	position	GpsLatLong (see B.5.3.2)	This attribute indicates the type of position.
	fomData	GpsFomData (see A.5.3.3)	This attribute indicates the type of figure of merit data.

B.5.3.2 Gps::GpsLatLong Structure

The *GpsLatLong* structure defines the GPS position using latitude and longitude coordinates.

```
struct GpsLatLong {
        /* Longitude data. */
        float longitude;
        /* Latitude data. */
        float latitude;
};
```

Struct	Attributes	Type	Unit	Valid Range	Description
GpsLatLong	longitude	float	radian	+/-(0 – π)	Longitude data.
	latitude	float	radian	+/-(0 – π/2)	Latitude data.

APPENDIX B.A ABBREVIATIONS AND ACRONYMS

latlong Latitude/Longitude

APPENDIX B.B PERFORMANCE SPECIFICATION

Table 7 provides a template for the generic performance specification for the *Latitude/Longitude Extension* which will be documented in the service or device using the interface. This performance specification corresponds to the port diagram in Figure 3.

Table 7 – Latitude/Longitude Extension Performance Specification

Specification	Description	Units	Value
Worst Case Command Execution Time for latlong_producer_provides_port	*	*	*
Worst Case Command Execution Time for latlong_1pps_uses_port	*	*	*

Note: (*) These values should be filled in by individual developers.

C. MGRS EXTENSION

C.1 INTRODUCTION

The *Military Grid Reference System (MGRS) Extension* is based upon the *GPS Device API*. It extends the functionality of the *GPS Device* to report GPS Position, Velocity, and Time (PVT) data using MGRS coordinates.

C.1.1Overview

This extension contains as follows:

a. Section C.1, *Introduction* of this document contains the introductory material regarding the overview, Service Layer description, Modes, States and Referenced Documents of this document.

b. Section C.2, *Services*, provides summary of service interface uses, interface for each device component, port connections, and sequence diagrams.

c. Section C.3, *Service Primitives and Attributes*, specifies the operations that are provided by the *MGRS Extension*.

d. Section C.4, *IDL*.

e. Section C.5, *UML*.

f. Appendix C.A, *Abbreviations and Acronyms*.

g. Appendix C.B, *Performance Specification*.

C.1.2Service Layer Description

C.1.2.1 GPS Device MGRS Extension Port Connections

The following figure shows the port connections for the *GPS Device* with MGRS reporting capabilities. All port names are for reference only.

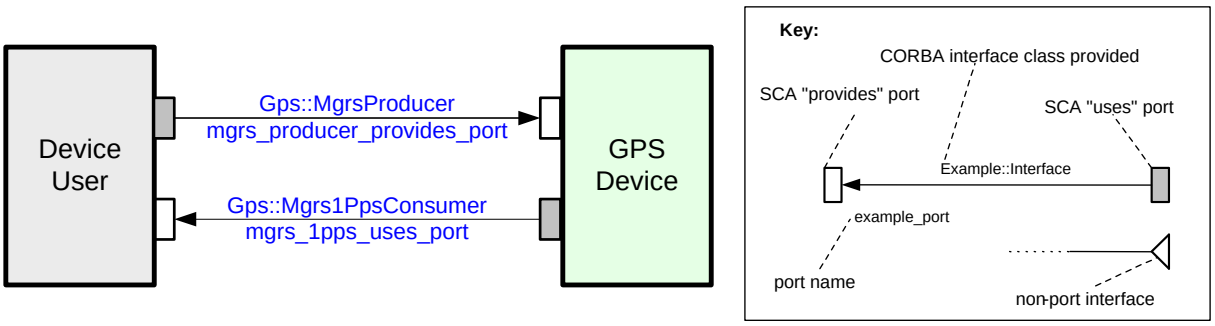

Figure 11 – MGRS Extension Port Diagram

***GPS Device* MGRS Extension Provides Ports Definitions**

mgrs_producer_provides_port is provided by *GPS Device* to produce PVT data.

***GPS Device* MGRS Extension Uses Ports Definitions**

mgrs_1pps_uses_port is used by the *GPS Device* to push PVT data to the *Device User* when a 1pps event occurs.

C.1.3 Modes of Service

Not applicable.

C.1.4 Service States

There are no changes from A. GPS Device.

C.1.5 Referenced Documents

There are no changes from A. GPS Device.

C.2 SERVICES

C.2.1 Provide Services

The *MGRS Extension* Provide Service consists of the following service ports, interfaces, and primitives, which can be called by other client components:

Table 8 – MGRS Extension Provide Service Interface

Service Group (Port Name)	Service (Interface Provided)	Primitives (Provided)
mgrs_producer_provides_port	MgrsDataProducer	readMgrs()

C.2.2 Use Services

The *MGRS Extension* Use Service set consists of the following service ports, interfaces, and primitives. Since the *MGRS Extension* acts as a client with respect to these services from other components, it is required to connect these ports with corresponding service ports applied by the server component. The *MGRS Extension* uses the Port Name as connectionID for the connection.

Table 9 – MGRS Extension Use Service Interface

Service Group (Port Name)	Service (Interface Used)	Primitives (Used)
mgrs_1pps_uses_port	Mgrs1PpsConsumer	pushMgrs1Pps()

C.2.3 Interface Modules

C.2.3.1 MGRS Extension

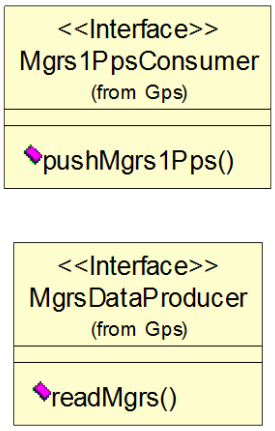

Figure 12 – MGRS Extension Class Diagram

C.2.3.1.1 Mgrs1PpsConsumer Interface Description

The interface design of the *Mgrs1PpsConsumer* is shown in the interface class diagram below. It provides the ability to transfer GPS PVT data to the *Device User* when a 1pps event occurs.

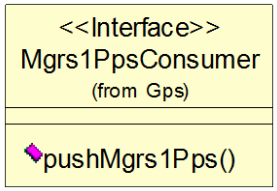

Figure 13 – Mgrs1PpsConsumer Interface Class Diagram

C.2.3.1.2 MgrsDataProducer Interface Description

The interface design of the *MgrsDataProducer* is shown in the interface class diagram. It is provides the capability to read GPS PVT data received from the GPS Receiver Hardware.

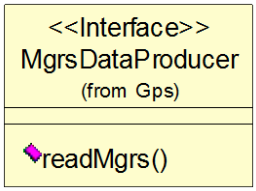

Figure 14 – MgrsDataProducer Interface Class Diagram

C.2.4 Sequence Diagrams

C.2.4.1 MGRS 1pps Event Sequence

Description

The *MGRS 1pps Event* sequence specifies the sequence that occurs when a *GPS Device* delivers a 1pps PVT Event. Multiple *Device Users* can connect to a *GPS Device* in order to receive a 1pps PVT event. When the *Timebase Hardware* produces a 1pps event, each *Device User* is issued a *pushMgrs1Pps* to signal the event. Note: The *Timebase Hardware* is the hardware that produces the 1pps event.

Pre-conditions

The *GPS Device* is in the ENABLED state.

Post-conditions

The *GPS Device* has pushed a 1pps event to the *Device User(s)*.

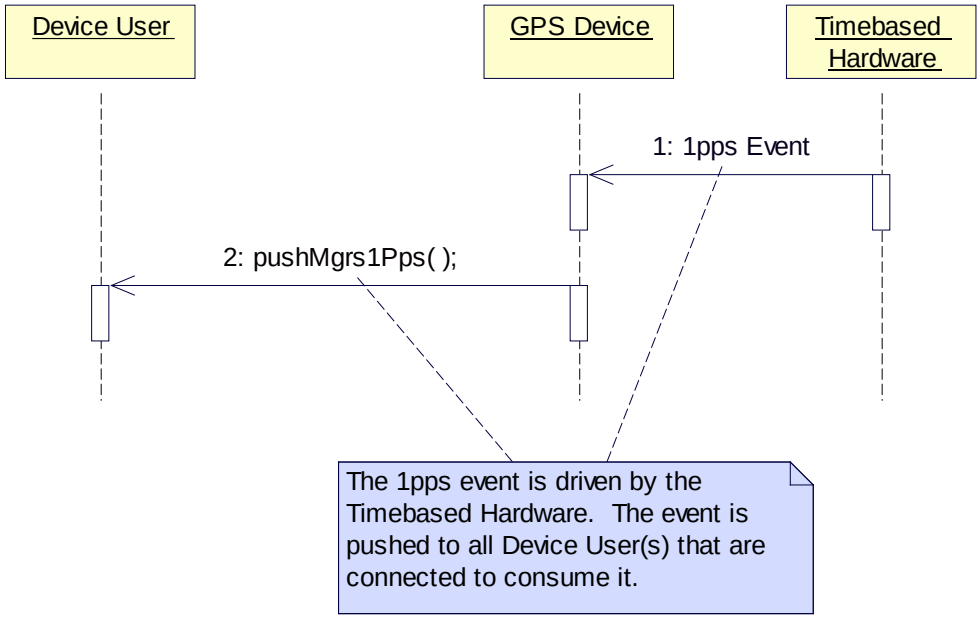

Figure 15 – MGRS 1pps Event Sequence Diagram

C.2.4.2 MGRS Non-Blocking Read Sequence

Description

The *MGRS Non-Blocking Read* sequence shows the events that occur when a *GPS Device* is issued a non-blocking read. To support this operation, PVT data is read from the *GPS Hardware* whenever a *Timebase Hardware* 1pps event occurs. The *Device User* issues a read command passing in "block" = FALSE to indicate this is a non-blocking read. The PVT data is then read and returned directly to the *Device User*. Note: The *Timebase Hardware* is the hardware that produces the 1pps event.

Pre-conditions

The *GPS Device* is in the ENABLED state.

Post-conditions

The *GPS Device* has returned PVT data to the *Device User*.

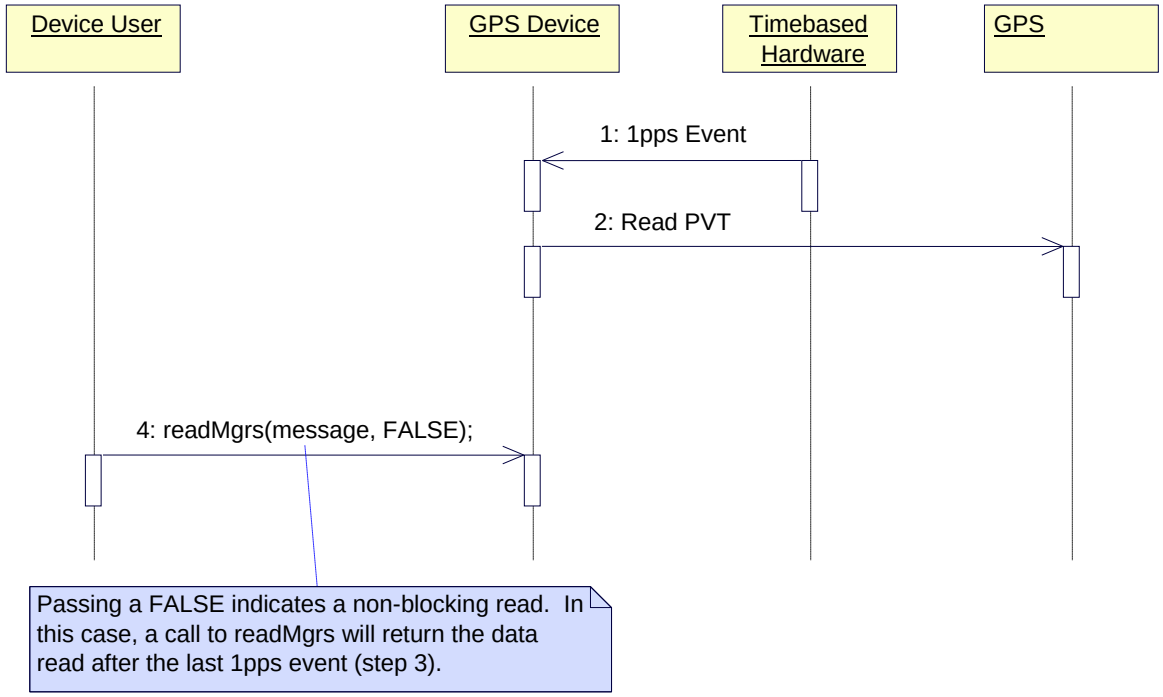

Figure 16 – MGRS Non-Blocking Read Sequence Diagram

C.2.4.3 MGRS Blocking Read Sequence

Description

The *MGRS Blocking Read* sequence shows the events that occur when a *GPS Device* is issued a blocking read. The *Device User* issues a read command passing in "block" = TRUE to indicate this is a blocking read. The call blocks until a *Timebase Hardware* 1pps occurs or a timeout threshold has been exceeded. If the *Timebase Hardware* 1pps event occurs, PVT data is read from the *GPS Hardware* and returned to the *Device User*. If the timeout threshold is exceeded, a *DeviceError* exception is raised and control is returned to the *Device User*. Note: *The Timebase Hardware* is the hardware that produces the 1pps event.

Pre-conditions

The *GPS Device* is in the ENABLED state.

Post-conditions

The *GPS Device* has returned PVT data to the Device User.

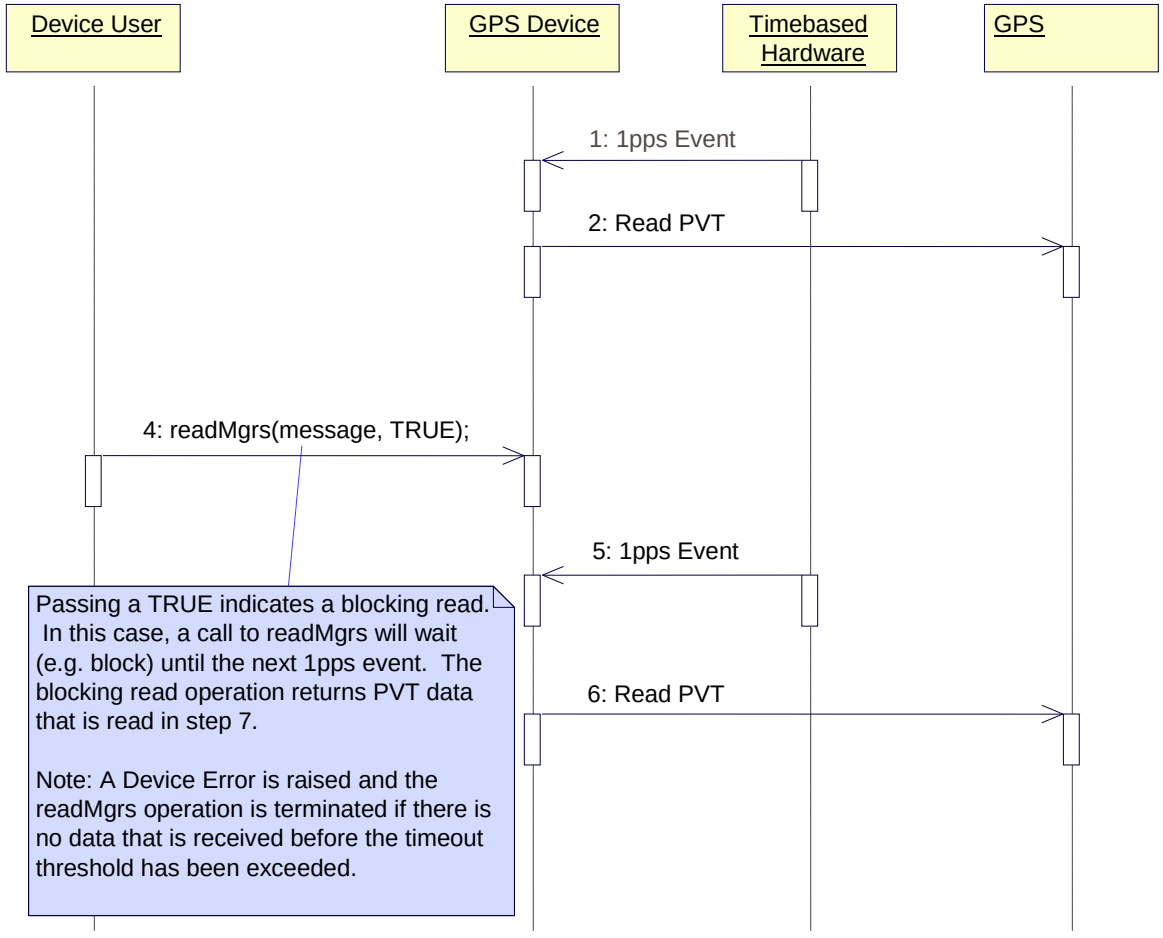

Figure 17 – MGRS Blocking Read Sequence Diagram

C.3 SERVICE PRIMITIVES AND ATTRIBUTES

To enhance the readability of this API document and to avoid duplication of data, the type definitions of all structured types (i.e., type definitions, enumerations, exceptions, and structures) used by the Service Primitives and Attributes have been co-located in section C.5 UML. This cross-reference of types also includes any nested structures in the event of a structure of structures or an array of structures.

C.3.1Gps::MgrsDataProducer

C.3.1.1 *readMgrs* Operation

The *readMgrs* operation is used to read PVT data received from the GPS Receiver hardware.

```
void readMgrs (
        inout GpsMgrsMessage message,
        in boolean block
        ) raises (DeviceError);
```

C.3.1.1.1 Parameters

Parameter Name	Type	Valid Range	Description
message	GpsMgrsMessage (see C.5.3.1)	Not Applicable	A structure specifying the GPS MGRS message (position, velocity, time, FOM data).
block	boolean	TRUE = blocking; FALSE=non-blocking	This flag indicates whether a blocking or non-blocking *readMgrs* operation should be performed (see C.2.4.2 and C.2.4.3 for additional information).

C.3.1.1.2 State

ENABLED CF::Device::operationalState.

C.3.1.1.3 New State

This operation does not cause a state change.

C.3.1.1.4 Return Value

None

C.3.1.1.5 Originator

Service User

C.3.1.1.6 Exceptions

Type	Description
DeviceError (see C.5.2.1)	Indicates that errors were detected during the *readMgrs* operation.

C.3.2Gps::Mgrs1PpsConsumer

C.3.2.1 *pushMgrs1Pps* Operation

The *pushMgrs1Pps* operation provides the ability to push a GPS message to the *Device User* when a 1pps event occurs.

C.3.2.1.1 Synopsis

```
oneway void pushMgrs1Pps (
        in GpsMgrsMessage message
        );
```

C.3.2.1.2 Parameters

Parameter Name	Type	Description
Message	GpsMgrsMessage (see C.5.3.1)	The GPS MGRS message.

C.3.2.1.3 State

ENABLED CF::Device::operationalState.

C.3.2.1.4 New State

This operation does not cause a state change.

C.3.2.1.5 Response

None

C.3.2.1.6 Originator

Service Provider

C.3.2.1.7 Exceptions

None

C.4 IDL

C.4.1GpsMgrsExt IDL

```
/*
**  GpsMgrsExt.idl
*/

#ifndef __GPSMGRSEXT_DEFINED
#define __GPSMGRSEXT_DEFINED

#ifndef __GPSDEVICE_DEFINED
#include "GpsDevice.idl"
#endif

module Gps {

        struct GpsMgrs {
                string mgrs;
        };

        struct GpsMgrsMessage {
                /* GPS Time. */
                GpsTime time;
                /* GPS velocity data. */
                GpsVel velocity;
                GpsMgrs position;
                /* Figure of merit data. */
                GpsFomData fomData;
        };

        interface MgrsDataProducer {

                exception DeviceError {
                        /* GPS Receive error. */
                        boolean invalidHardware;
                        /* GPSDevice is in an invalid state. */
                        boolean invalidState;
                        /* GPS Time data is wrong format. */
```

```
                boolean invalidTime;
                /* The message string describes a reason of exception has been raised by GPSDevice. */
                string msg;
        };

        void readMgrs (
                inout GpsMgrsMessage message,
                in boolean block
                ) raises (DeviceError);

    };

    interface Mgrs1PpsConsumer {

        oneway void pushMgrs1Pps (
                in GpsMgrsMessage message
                );

    };


};

#endif // __GPSMGRSEXT_DEFINED
```

C.5 UML

This section contains the Device component UML diagram and the definitions of all data types referenced (directly or indirectly) by the Service Primitives and Attributes in section C.3.

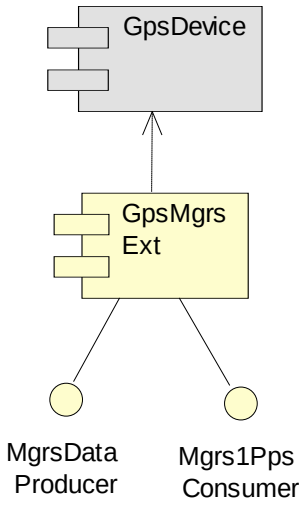

Figure 18 – MGRS Extension Component Diagram

C.5.1Enumerations

None

C.5.2Exceptions

C.5.2.1 Gps::DeviceError Exception

The *DeviceError* exception is used to indicate that errors were detected during the *readMgrs* operation (see C.3.1).

```
exception DeviceError {
        /* GPS Receive error. */
        boolean invalidHardware;
        /* GPSDevice is in an invalid state. */
        boolean invalidState;
        /* GPS Time data is wrong format. */
        boolean invalidTime;
        /* The message string describes a reason of exception has been raised by GPSDevice. */
        string msg;
};
```

Exception	Attributes	Type	Valid Range	Description
DeviceError	invalidHardware	boolean	TRUE= timed out; FALSE=not timed out	Indicates if the blocking read timed out.
	invalidState	boolean	TRUE = invalid; FALSE=valid	Indicates if the GPS state is invalid.
	invalidTime	boolean	TRUE=invalid; FALSE=valid	Indicates if the GPS Time data is in the wrong format.
	msg	string	Not Applicable.	A message of type string indicating that the exception has occurred.

C.5.3Structures

C.5.3.1 Gps::GpsMgrsMessage Structure

The *GpsMgrsMessage* structure defines the GPS MGRS Message which is comprised of the GPS position, velocity, time, and FOM data. This structure is used in the *readMgrs* and *pushMgrs1Pps* operations (see C.3.1 and C.3.2.1).

```
struct GpsMgrsMessage {
        /* GPS Time. */
        GpsTime time;
        /* GPS velocity data. */
        GpsVel velocity;
        GpsMgrs position;
        /* Figure of merit data. */
        GpsFomData fomData;
};
```

struct	Attributes	Type	Description
GpsMgrsMessage	time	GpsTime (see A.5.3.1)	This attribute indicates the type of time.
	velocity	GpsVel (see A.5.3.2)	This attribute indicates the type of velocity.
	position	GpsMgrs (see C.5.3.2)	This attribute indicates the type of position.
	fomData	GpsFomData (see A.5.3.3)	This attribute indicates the type of figure of merit data.

C.5.3.2 Gps::GpsMgrs Structure

The *GpsMgrs* structure defines the GPS position in MGRS formats.

```
struct GpsMgrs {
        string mgrs;
};
```

Struct	Attributes	Type	Description
GpsMgrs	mgrs	String	MGRS position data.

APPENDIX C.A ABBREVIATIONS AND ACRONYMS

MGRS Military Grid Reference System

APPENDIX C.B PERFORMANCE SPECIFICATION

Table 10 provides a template for the generic performance specification for the *MGRS Extension* which will be documented in the service or device using the interface. This performance specification corresponds to the port diagram in Figure 11.

Table 10 – MGRS Extension Performance Specification

Specification	Description	Units	Value
Worst Case Command Execution Time for mgrs_producer_provides_port	*	*	*
Worst Case Command Execution Time for mgrs_1pps_uses_port	*	*	*

Note: (*) These values should be filled in by individual developers.