United Kingdom Fishing Authorities IVMS

Interface

Table of Contents

Document Control

Version	Date	Control	Distribution
0.1	28/10/2014	Initial Version	MMO
0.2	20/12/2018	General Update	MMO

1. Introduction

The following is a “high level” view of the implementation of the IVMS system interface, whereby IVMS suppliers may interface to the DEFRA VMS system for the purpose of both testing and live operation. Access to the IVMS interface is only for suppliers accepted by DEFRA for the purposes of testing, or formally approved as providing an accepted solution for live reporting.

The interface may be subject to change, any such change will be backwards compatible where possible. All changes will be subject to approval by DEFRA and will only be released on their approval. IVMS Suppliers will be notified of any interface changes by DEFRA.

2. Definitions

Unless the context otherwise requires, the following words and expressions shall have the following meanings (to be interpreted in the singular or plural as the context requires):

TERM	MEANING
GUID	Globally Unique IDentifier
IVMS	Inshore Vessel Monitoring System (or hardware device for reporting to this system)
WSDL	Web Service Definition Language
XML	Extensible Markup Language

3. Servers and Interfaces

There will be both a Production and a Test Server, suppliers will use the Test server for their own development purposes and to gain DEFRA approval. The details of the Test Server are provided below. Once approved by DEFRA the details of the Production Server will be provided. The Test and Production Servers will implement the same methods, only the address of the server will change.

Each server implements 2 interfaces, a Web Service accessible via the published WSDL and an Email address.

TEST SERVER

Interface	Address
WSDL	https://81.143.218.190:8084/IVMSService.asmx?WSDL https://81.143.218.190:8084 (homepage)
Email	Ivms.test@saffire-online.com

NOTE: The test server is currently running with a self-signed SSL certificate. Suppliers should accept the certificate when connecting. Where necessary, sample code is supplied below to indicate a method for accepting these certificates programmatically.

4. WSDL Methods

The methods available on the WSDL are broken into 3 main types; Supplier Reporting, Supplier Testing & Supplier Maintenance. The functions are defined below. In calling each method, if the supplied parameters fail validation (e.g. Supplier Authentication, Asset/Device identification) the webservice will throw an exception that the client is expected to handle, in such cases do not resubmit the same report and seek clarification from your technical support contact. See Code Samples below for examples of the errors thrown.

Supplier Reporting

These methods are used to submit reports from iVMS devices to the DEFRA VMS.

public VMSReturn AssetReport(SupplierAuth Auth, AssetReport Report)

- call this method to submit an IVMS device report to the DEFRA VMS system

Supplier Testing

These methods are supplied so that a supplier can verify either their vessel devices, or position reports, have been received by the server.

public List<Asset> GetAssets(SupplierAuth Auth)

- call this method for the Webservice to return a list of all the suppliers Assets currently on the server.

public List<AssetReport> GetAssetReports(SupplierAuth Auth, Asset asset, DateTime FromDate, DateTime ToDate)

- Call this method to return a list of all AssetReport received between the specified FromDate & ToDate range. NOTE: The difference between FromDate and ToDate must be 24 hours or less, anything larger will throw a "SoapException.ClientFaultCode"

Supplier Maintenance

These methods are made available to suppliers in order to allow them to setup and maintain vessel/device details on the DEFRA VMS database. Access to these functions may not be available to all suppliers.

Where suppliers are permitted to update Assets within the DEFRA VMS, the functions below may be called as required as device installations are performed. If a supplier is not authorised to maintain the Asset details, DEFRA will specify the Asset fields for submitting an AssetReport against each DeviceID installed.

public VMSReturn AddAsset(SupplierAuth Auth, Asset asset)

- Call this function to add a new Asset & Device, this is expected to only be called once per vessel as each supplier installs devices. If a supplier call this for a DeviceID already used, then a "SoapException.ClientFaultCode" is thrown.

public VMSReturn RemoveDevice(SupplierAuth Auth, Asset asset)

- Call this function if a device has been removed from an Asset, for example replacing a faulty unit or decommissioning a device. The DeviceID may be re-used on another Asset via the AddAsset or AddDevice calls.

public VMSReturn AddDevice(SupplierAuth Auth, Asset asset)

- Call this method to add a DeviceID to an existing Asset. The DeviceID must not be assigned to an existing Asset, and the Asset must already exist. For example, after replacing a device on an Asset.

public VMSReturn UpdateAsset(SupplierAuth Auth, Asset asset)

- Call this method to update details of an Asset. The Asset is identified by the DeviceID specified, resulting in the AssetName, CallSIgn, PLN & RegNumber being updated. Subsequent webservice calls are expected to use the new Asset fields where specified. For example, Asset details were incorrectly recorded during installation.
- If a Device is being moved to a different Asset, do not use UpdateAsset as it will overwite VMS details of the Asset, instead use RemoveDevice and AddDevice specifying the appropriate Asset for each operation.

** NOTE **
Suppliers may NOT delete Assets, only devices. In case of any problems with submitted Assets which cannot be resolved by the supplier, contact either DEFRA or your technical support representative for assistance.

5. Email Methods

(T.B.C)
Email interface is currently under development, documentation will be updated when available.

6. Object definitions

In the definitions below, all fields are considered Mandatory unless specified otherwise.

SupplierAuth – Suppliers Authentication to IVMS service

Field	Use
SupplierID	Unique Supplier ID
SupplierPIN	Supplier PIN

SupplierID and SupplierPIN will be supplied by DEFRA and will not be updatable by the supplier.

Asset – an Asset/Vessel device

Field	Use
AssetName	The name of the vessel
CallSign (Optional)	The vessel Radio Call Sign
RegNumber (Optional)	The vessel Registry Number
PLN (Optional)	The vessel Port Licence Number
DeviceID	The supplier defined unique identifier
VMSID	The VMS GUID

Each supplier will identify their administered devices by a unique (to them) identifier a.k.a. DeviceID. The DEFRA VMS will maintain a GUID for each device, unique across the entire VMS. When suppliers send reports, they need only populate their own DeviceID and AssetName. When suppliers receive data, via the Supplier Testing functions, the DeviceID will be populated as well as the VMSID for each device. Suppliers are not expected to make any use of the VMSID number, it is for reference only.

CallSign, RegNumber & PLN are not mandatory, but must be supplied on each Reporting function call if used at all.
DEFRA may define the values of these fields to suppliers as required.

AssetPosition – a position reported by an IVMS device

Field	Use
Latitude	Decimal degrees – WGS84
Longitude	Decimal degrees – WGS84
Course	Degrees
Speed	Knots
GPSQuality	Freetext – indicative of the GPS quality
DeviceTimestamp	Time of GPS fix

StatusCode – a device may report one of an enumerations of device status

Status Code	Use
SC_NULL	No status code being reported
SC_EXT_POWER_LOSS	Device has lost external power, using battery
SC_EXT_POWER_RESTORED	Device external power is reconnected
SC_LOW_BATTERY	Device has low internal battery
SC_NETWORK_LOSS	Device lost GPRS coverage (reported on GPRS recovery)
SC_GPS_LOST	Device has lost GPS
SC_TAMPER	Device tamper activated (unit opened or other unauthorised access)
SC_POSITION_LATE	AssetReport was not sent immediately

NOTE: a device may submit an AssetReport containing a StatusCode without an AssetPosition.

ReportChannel – an enumerator of the reporting channels a device may use

Field	Use
RC_GPRS	Report received via GPRS (IVMS device are expected to use GPRS only)
RC_SATELLITE	Report received by Satellite

ReportFlag – an enumeration of the Report Flag options

Field	Use
RF_National	Device is reporting under National Flag reporting requirements (IVMS devices are expected to use RF_National reporting only)
RF_EU	Device is reporting under EU Flag reporting requirements

AssetReport – a report from an IVMS device for the DEFRA VMS system

Field	Use
Asset	Asset object
AssetPosition (optional)	AssetPosition object
SupplierTimestamp	The time the device supplier received the report from their IVMS device.
ReportReason	Freetext – will indicate on VMS the reason for report, e.g. “Zone Entry”, “Zone Exit” etc.
StatusCode	StatusCode enumeration above
ZoneID	0 = no IVMS zone being reported. +ve number = Zone entry -ve number = Zone exit IVMS zone numbers will be provided by DEFRA
Channel	ReportChannel enumeration – RC_GPRS normally
Flag	ReportFlag enumeration – RF_National normally
SupplierReportID	A unique (per supplier) identifier for the supplied AssetReport
VMSReportID	A GUID issued by the VMS for each AssetReport received

When submitting an AssetReport, each supplier must populate the SupplierReportID with a unique (to them) identifier for each report submitted. When suppliers receive data, via the Supplier Testing functions, the VMSReportID will be populated as well as the SupplierReportID for each AssetReport. Suppliers are not expected to make any use of the VMSReportID number, it is for reference only.

VMSReturn -

Field	Use
Code	0 = Error. > 0 = GUID issued by VMS for each report received.
Text	“OK” or details of error.

7. Sample Code

All code examples are in C#, unless otherwise detailed.

7.1 Handling Self Signed Certificates

To accept any certificate

```
ServicePointManager.ServerCertificateValidationCallback = new RemoteCertificateValidationCallback(
  delegate(Object obj, X509Certificate certificate, X509Chain chain, SslPolicyErrors errors)
  {
      return true;
  }
);
```

To accept a single certificate – given example is the TEST Server SSL certificate serial number

```
ServicePointManager.ServerCertificateValidationCallback = new RemoteCertificateValidationCallback(
  delegate(Object obj, X509Certificate certificate, X509Chain chain, SslPolicyErrors errors)
  {
      string serverSSL = certificate.GetSerialNumberString();
      return serverSSL.Equals("6841809044948DA34E80EC81B1ADBB10", StringComparison.InvariantCultureIgnoreCase);
  }
);
```

7.2 AddAsset

```
public void TestSaveVessel()
{
    Log.Info("Adding new vessel");

    IVMSServiceReference.SupplierAuth myAuth = new IVMSServiceReference.SupplierAuth();
    myAuth.SupplierID = 100;
    myAuth.SupplierPIN = "TEST";

    IVMSServiceReference.Asset myAsset = new IVMSServiceReference.Asset();
    myAsset.AssetName = "My Vessel";
    myAsset.CallSign = "V1";
    myAsset.RegNumber = "V123";
    myAsset.DeviceID = "1234";
    myAsset.PLN = "P321";

    IVMSServiceReference.VMSServiceSoapClient serviceiVMS = new IVMSServiceReference.VMSServiceSoapClient();
    // handle SSL certificate as required
    IVMSServiceReference.VMSReturn returnVMS = serviceiVMS.AddAsset(myAuth, myAsset);
}
```

7.3 RemoveDevice

```
public void TestRemoveDevice()
{
    IVMSServiceReference.SupplierAuth myAuth = new IVMSServiceReference.SupplierAuth();
    myAuth.SupplierID = 100;
    myAuth.SupplierPIN = "TEST";

    IVMSServiceReference.Asset myAsset = new IVMSServiceReference.Asset();
    myAsset.AssetName = "My Vessel";
    myAsset.CallSign = "V1";
    myAsset.RegNumber = "V123";
    myAsset.DeviceID = "1234";
    myAsset.PLN = "P321";

    IVMSServiceReference.VMSServiceSoapClient serviceiVMS = new IVMSServiceReference.VMSServiceSoapClient();
    // handle SSL certificate as required
    serviceiVMS.RemoveDevice(myAuth, myAsset);
}
```

7.4 AddDevice

```
public void TestAddDevice()
{
    IVMSServiceReference.SupplierAuth myAuth = new IVMSServiceReference.SupplierAuth();
    myAuth.SupplierID = 100;
    myAuth.SupplierPIN = "TEST";

    IVMSServiceReference.Asset myAsset = new IVMSServiceReference.Asset();
    myAsset.AssetName = "My Vessel";
    myAsset.CallSign = "V1";
    myAsset.RegNumber = "V123";
    myAsset.DeviceID = "1234";
    myAsset.PLN = "P321";

    IVMSServiceReference.VMSServiceSoapClient serviceiVMS = new IVMSServiceReference.VMSServiceSoapClient();
    // handle SSL certificate as required
    serviceiVMS.AddDevice(myAuth, myAsset);
}
```

7.5 UpdateAsset

```
public void TestUpdateAsset()
 {
     IVMSServiceReference.SupplierAuth myAuth = new IVMSServiceReference.SupplierAuth();
     myAuth.SupplierID = 100;
     myAuth.SupplierPIN = "TEST";

     IVMSServiceReference.Asset myAsset = new IVMSServiceReference.Asset();
     myAsset.AssetName = "Daves Dingy II";
     myAsset.CallSign = "V1";
     myAsset.RegNumber = "V123";
     myAsset.DeviceID = "1234";
     myAsset.PLN = "P321";

     IVMSServiceReference.VMSServiceSoapClient serviceiVMS = new IVMSServiceReference.VMSServiceSoapClient();
     // handle SSL certificate as required
     serviceiVMS.UpdateAsset(myAuth, myAsset);
 }
```

7.6 GetAssets

```
public void TestGetAssets()
{
    IVMSServiceReference.SupplierAuth myAuth = new IVMSServiceReference.SupplierAuth();
    myAuth.SupplierID = 100;
    myAuth.SupplierPIN = "TEST";

    IVMSServiceReference.VMSServiceSoapClient serviceiVMS = new IVMSServiceReference.VMSServiceSoapClient();
    // handle SSL certificate as required
    IVMSServiceReference.Asset[] myAssets = serviceiVMS.GetAssets(myAuth);
    foreach (IVMSServiceReference.Asset asset in myAssets)
    {
            Log.InfoFormat("Asset {0} {1} {2} {3} {4} {5}", asset.AssetName, asset.CallSign, asset.PLN, asset.RegNumber,
      asset.DeviceID, asset.VMSID);
    }
}
```

7.7 GetAssetReports

```
public void TestGetAssetReports()
{
    IVMSServiceReference.SupplierAuth myAuth = new IVMSServiceReference.SupplierAuth();
    myAuth.SupplierID = 100;
    myAuth.SupplierPIN = "TEST";

    IVMSServiceReference.Asset myAsset = new IVMSServiceReference.Asset();
    myAsset.AssetName = "My Vessel";
    myAsset.CallSign = "V1";
    myAsset.RegNumber = "V123";
    myAsset.DeviceID = "1234";
    myAsset.PLN = "P321";

    IVMSServiceReference.VMSServiceSoapClient serviceiVMS = new IVMSServiceReference.VMSServiceSoapClient();
    // handle SSL certificate as required
    DateTime fromDate = new DateTime(2014, 09, 26, 00, 00, 00);
    DateTime toDate = new DateTime(2014, 09, 26, 23, 59, 59);
    IVMSServiceReference.AssetReport[] myReports = serviceiVMS.GetAssetReports(myAuth, myAsset, fromDate, toDate);

    Log.InfoFormat("Reports for Asset {0} {1} {2} {3} {4}", myAsset.AssetName, myAsset.CallSign, myAsset.PLN,
    myAsset.RegNumber, myAsset.DeviceID);
    foreach (IVMSServiceReference.AssetReport myReport in myReports)
    {
        Log.InfoFormat("{0} {1} {2} {3} {4} {5} {6} {7} {8} {9} {10} {11} {12}",
                       myReport.Channel, myReport.Flag,
                       (null == myReport.Position) ? 0 : myReport.Position.Latitude,
                       (null == myReport.Position) ? 0 : myReport.Position.Longitude,
                       (null == myReport.Position) ? 0 : myReport.Position.Course,
                       (null == myReport.Position) ? 0 : myReport.Position.Speed,
                       (null == myReport.Position) ? "" : myReport.Position.GPSQuality,
                       myReport.StatusCode, myReport.SupplierReportID, myReport.SupplierTimestamp,
                       myReport.VMSReportID, myReport.ZoneID, myReport.ReportReason);
    }
}
```

7.8 AssetReport

```
public void SaveAssetReportAndZoneExit()
{
    IVMSServiceReference.SupplierAuth myAuth = new IVMSServiceReference.SupplierAuth();
    myAuth.SupplierID = 100;
    myAuth.SupplierPIN = "TEST";

    IVMSServiceReference.Asset myAsset = new IVMSServiceReference.Asset();
    myAsset.AssetName = "My Vessel";
    myAsset.CallSign = "V1";
    myAsset.RegNumber = "V123";
    myAsset.DeviceID = "1234";
    myAsset.PLN = "P321";

    IVMSServiceReference.AssetReport myReport = new IVMSServiceReference.AssetReport();
    myReport.Asset = myAsset;

    IVMSServiceReference.AssetPosition myPosition = new IVMSServiceReference.AssetPosition();
    myPosition.Latitude = 51.216;
    myPosition.Longitude = -2.663333333333333;
    myPosition.Course = 0;
    myPosition.Speed = 0.0;
    myPosition.GPSQuality = "3D Fix";
    myPosition.DeviceTimestamp = DateTime.UtcNow; // for DEMO only, must be GPS fix time
    myReport.Position = myPosition;

    myReport.SupplierTimestamp = myPosition.DeviceTimestamp.AddMinutes(2); // for DEMO only, must be supplier receive time
    myReport.ReportReason = "Zone IO";
    myReport.StatusCode = IVMSServiceReference.StatusCode.SC_NULL;
    myReport.ZoneID = -5; // leaving zone 5
    myReport.Channel = IVMSServiceReference.ReportChannel.RC_GPRS;
    myReport.Flag = IVMSServiceReference.ReportFlag.RF_National;
    myReport.SupplierReportID = 123458; // unique SupplierReportID

    IVMSServiceReference.VMSServiceSoapClient serviceiVMS = new IVMSServiceReference.VMSServiceSoapClient();
```

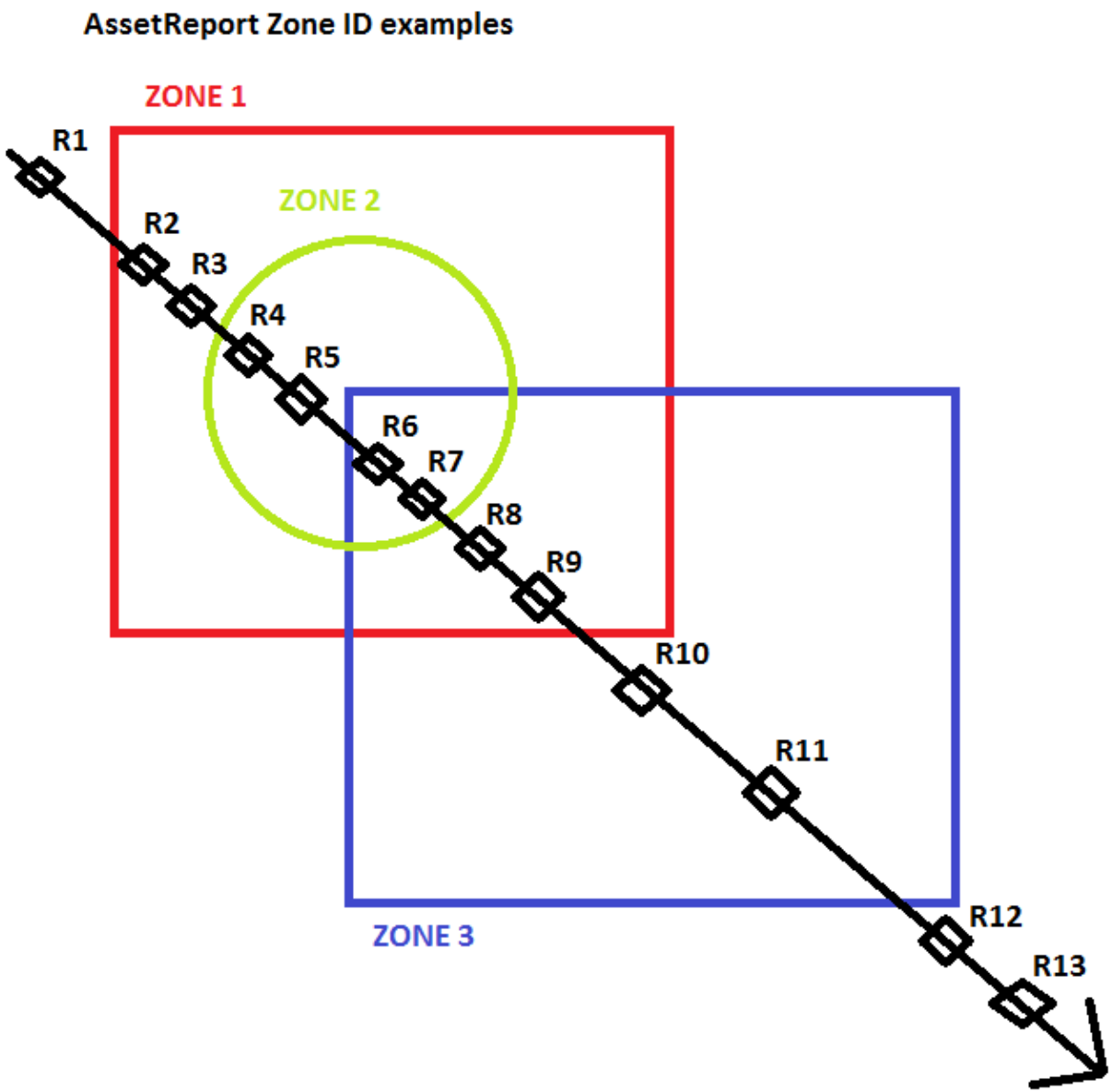
```
    // handle SSL certificate as required
    IVMSServiceReference.VMSReturn returnVMS = serviceiVMS.AssetReport(myAuth, myReport);
}
```

7.9 Client handling server exceptions

```
public void TestInvalidPIN()
{
    IVMSServiceReference.SupplierAuth myAuth = new IVMSServiceReference.SupplierAuth();
    myAuth.SupplierID = 100;
    myAuth.SupplierPIN = "INVALIDPIN";

    IVMSServiceReference.Asset myAsset = new IVMSServiceReference.Asset();
    myAsset.AssetName = "My Vessel";
    myAsset.CallSign = "V1";
    myAsset.RegNumber = "V123";
    myAsset.DeviceID = "1234";
    // handle SSL certificate as required
    IVMSServiceReference.VMSServiceSoapClient serviceiVMS = new IVMSServiceReference.VMSServiceSoapClient();

    try
    {
        IVMSServiceReference.VMSReturn returnVMS = serviceiVMS.AddAsset(myAuth, myAsset);
    }
    catch (FaultException ex) // the Webservice has thrown a “SoapException.ClientFaultCode”
    {
        Log.InfoFormat("Client Error {0} {1}", ex.Action, ex.Code);
    }
    catch (Exception ex)
    {
        Log.Fatal("Unexpected error");
        Log.Fatal(ex);
        throw;
    }

}
```

Asset Report ZONE ID examples

The following diagram explains how the ZONE ID parameter is set for a number of Zone Entry/ Exit messages.


AssetReport Zone ID examples
ZONE 1
ZONE 2
ZONE 3
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13

Report	AssetReport ZONE ID Parameter	Reason
R1	0	Unit is not reporting Zone Entry / Exit event
R2	1	Unit is reporting Zone 1 Entry
R3	0	Unit is not reporting Zone Entry / Exit event
R4	2	Unit is reporting Zone 2 Entry
R5	0	Unit is not reporting Zone Entry / Exit event
R6	3	Unit is reporting Zone 3 Entry
R7	0	Unit is not reporting Zone Entry / Exit event
R8	-2	Unit is reporting Zone 2 Exit
R9	0	Unit is not reporting Zone Entry / Exit event
R10	-1	Unit is reporting Zone 1 Exit
R11	0	Unit is not reporting Zone Entry / Exit event
R12	-3	Unit is reporting Zone 3 Exit
R13	0	Unit is not reporting Zone Entry / Exit event