Interface Control Document

Web Services provided by FMI

Title	Web services provided by FMI
Date	March 25, 2015
Editor	Lasse Häkkinen, Finnish Meteorological Institute
Contributors	Esa Kallio, Aalto University
Distribution	*IMPEx*
Level	*Low level*
Roles	*Data producer/ Web service developer*
User Requirements	
Related Workpackages	*WP2, WP3*

Version History

Version	Date	Released by	Detail
0.1	March 19, 2014	Lasse Häkkinen	Initial version
0.9	June 3, 2014	Lasse Häkkinen	Added getDataPointSpectra, getDataPointSpectraSpacecraft and isAlive methods. Renamed getDataPointValue_Spacecraft as getDataPointValueSpacecraft. Corrected some typos.

EC Project Nr.: **262863**

0.91	June 11, 2014	Lasse Häkkinen	Updated ResourceID in examples according to Spase naming rules.
1.0	December 15, 2014	Lasse Häkkinen	Updated text on error handling. Added text about GUMICS 10^5 runs.
1.0.1	March 25, 2015	Lasse Häkkinen	Added missing default values for optional parameters in method calls.

1. Introduction

1.1 Scope of the document

This document contains the definition of web services provided by the Finnish Meteorological Institute (FMI) in the framework of the FP7 IMPEx project. These web services provide access to HYB and GUMICS simulation data bases at FMI's servers and give users simple interfaces to download simulation run results with user defined parameter sets.

The IMPEx web services are described in WSDL language (version 1.1). The FMI's web services definition file (Methods_FMI.wsdl) lists all services (methods) provided by FMI and is available at

`http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl`

1.2 About FMI simulation run data bases

Currently FMI simulation run data bases consists of two different simulation platforms:

HYB (hybrid and kinetic plasma) simulation platform for planetary interactions. This platform is designed for unmagnetized or weakly magnetized solar system bodies (Mercury, Venus, Mars, comets, etc.). More information may be found from

```
http://hwa.fmi.fi/hyb
```

GUMICS (Grand Unified Magnetosphere Ionosphere Coupling Simulation) platform. This platform is designed to model the plasma environment of the Earth. More information is available at

```
http://gumics.fmi.fi
```

In the IMPEx context each institute with simulation model data base (SMDB) provides an XML file (tree.xml) which describes the metadata for all simulation runs in the data base. This metadata includes relevant input data for the runs along with information about the run results. The HYB and GUMICS models are different in many respects and therefore we have decided to have separate tree files for the two models. They are available at:

```
http://impex-fp7.fmi.fi/ws/Tree_FMI_HYB.xml

http://impex-fp7.fmi.fi/ws/Tree_FMI_GUMICS.xml
```

Although the tree files are separate for the two simulation models the wsdl methods file (Methods_FMI.wsdl) is common for both of them. The user does not have to specify which tree file he/she is using.

Internally at FMI the results of each simulation run are stored in large data files (called .hc files) which contain values of physical quantities (e.g. density, velocities, magnetic field strengths) computed in predefined grids. These grids may be constant or adaptive. The .hc files are not directly available to user but instead user requested values are interpolated from these pre-computed values. So everything at FMI is pre-computed, there are no runs-on-demand.

1.3 Miscellaneous issues

- **Temporary result files**: Most of the methods return an url to randomly named file which is stored at FMI's server. These result files are temporary and will be stored at the server for 30 days.

- **Error situations**: All FMI methods throw a SOAP Fault element on error situations. The fault element's subelement <faultcode> will be either "SOAP-ENV:Client" or "SOAP-ENV:Server". A Client error occurs when some of the input parameters are illegal. A server error indicates that the the data processing failed because of some internal reason. The element <faultstring> gives some information about the error. The element <faultfactor> is always 'FMI'.

- **Units**: SI units are used everywhere in our tree files. Many methods expect an url to a VOTable file as an input parameter. These VOTable files must use SI units for all dimensional quantities. Especially m (meter) should be used as length unit and m/s as velocity unit. Some multiplied units (e.g. km) are recognized but one should be careful in using them. In any case do not use planetary radius as length unit even if the radius is defined somewhere else in the input file.

- **GUMICS inner boundary**: For GUMICS runs one should note that there is a concept called 'inner boundary'. Computed values inside this inner boundary are not reliable and therefore the methods should automatically set those values to missing values (NaN). The value of the inner boundary is 4.2 Earth radii.

1.4 GUMICS 10^5 runs

At FMI the GUMICS simulation model (version 4) has been used to simulate the dynamics of the Earth's magnetosphere-ionospheric system under realistic solar wind conditions. The analysis covers the period 30.1.2002 - 2.2.2003 with input parameters taken from real solar wind data as measured by the ACE satellite. The whole simulation data set consists of a time series of about 100000 snapshots (every five minutes) each describing the state of the magnetosphere at certain time. This set has been produced by combining the output of a large number of simulation runs (about 1800, not 10^5). Each new run starts in a state where the previous run ends and thereafter saves the state of the magnetosphere at five minute intervals. The input solar wind parameters also change dynamically during the run. So strictly speaking the question is not about adding 10^5 independent runs into the tree.xml but adding a single time dependent run with 10^5 snapshots.

In the tree.xml file (Tree_FMI_GUMICS.xml) the inclusion of these runs is implemented as follows:

- A <SimulationRun> element is created for each month separately. So there are 12 elements (1.2.2002-28.2.2002, …. , 1.1.2003 – 31.1.2003).
- The <TemporalDependence> element is set to value 'Yes'.
- The solar wind input parameters for a single simulation run are defined in the elements <InputField>, <InputPopulation> and <InputParameter>. These elements contain sub-element <InputTableURL> which points to a VOTable format file containing the values of the input parameters for each snapshot.
- The <NumericalOutput> element is similar to the static simulation run case. It describes what physical quantities can be computed as output from the simulation run.

In most of the IMPEx method calls the user has to specify the ResourceID of a <NumericalOutput> element (not the <SimulationRun> element) as the input parameter. If the user wants to make a reference to a certain snapshot of a dynamical GUMICS run he/she must add a 'sub-ResourceID' to the <NumericalOutput> element's ResourceID. This 'sub-ResourceID' may be found from the VOTable file defined in the <InputTableURL> element. It may also be found by using the method 'getMostRelevantRun'. The 'sub-ResourceID' is typically a datetime string (e.g. '20020701_123000).

An example: if we want to compute magnetic field lines for the GUMICS time dependent run on 2002-07-01 at 12:30 we would set the ResourceID in the getFieldLine method call as:
ResourceID =
spase://IMPEX/NumericalOutput/FMI/GUMICS/earth/synth_dynamical/200207/Mag?20020701_123000

The methods getDataPointValueSpacecraft and getDataPointSpectraSpacecraft are different in nature as they already contain time information in the spacecraft orbit data. If these methods are used with a time dependent run then the method itself must be intelligent so that it knows which snapshot is used at each spacecraft position. With the GUMICS runs this requires loading and interpolating a new big data file (.hc file) at each time step. This will considerable slow down the processing time as compared to static runs where only a single data file is loaded and analysed.

As mentioned above the method getMostRelevantRun is able to analyse all static and time dependent runs and may be used to search for runs with suitable input parameters. However, for time dependent runs it must be remembered that the run is not a static run with given input parameters but only one snapshot in a continuously changing system.

2. List of web services from FMI

Web services from all SMDS's in the IMPEx project are described in the file "IMPEx Methods Definitions" in Google drive. FMI does not support all these methods. Following table lists those services that are available at FMI's server.

Name of the method	Description	Delivered
getDataPointValue	Returns interpolated values of physical quantities at a set of 3D points specified by the user.	23.04.2013
getDataPointValueSpacecraft	Returns interpolated values of physical quantities along a given spacecraft orbit.	23.04.2013
getSurface	Returns interpolated values of physical quantities on a user specified 2D surface.	30.08.2013
getVOTableURL	Returns an URL to a VOTable format file that is created on the fly.	13.12.2013
getMostRelevantRun	Returns an ordered list of runs that have input values as close as possible to the provided set of values.	17.09.2013
getFieldLine	Returns magnetic field lines or stream lines for given set of initial 3D points.	20.12.2013
getParticleTrajectory	Returns particle trajectories for given set of initial particle 3D positions and velocities.	14.03.2014
getDataPointSpectra	Returns the energy spectra of given particle populations at given 3D points.	03.06.2014
getDataPointSpectraSpacecraft	Returns the energy spectra given particle populations along a given spacecraft orbit.	03.06.2014
isAlive	Returns boolean true. This method is intended for checking web server status.	03.06.2014

The descriptions, input parameters, returned output data and working examples of all these web services are presented below.

2.1 getDataPointValue

This method is used to compute values of various physical quantities (e.g. density, magnetic field strengths etc.) for given simulation run at user given set of 3D points.

Input parameters:

• **ResourceID** (String, Mandatory)

ResourceID of a <NumericalOutput> element in the tree.xml file. This NumericalOutput element defines at the same time the simulation run and the set of physical parameters (e.g. density, magnetic field) which are the output from this simulation run. The correspondence in the data model is: “Spase/NumericalOutput/ResourceID”.

• **Variable** (List, Optional)

List (string) of individual physical parameters (e.g. Density, Pressure, Bx) whose values are interpolated in given points. The tokens for the physical parameters are given in the <ParameterKey> element in the <NumericalOutput> element.
Correspondence in the data model:
“Spase/NumericalOutput/Parameter/ParameterKey”.
If two or more parameters are given they must be separated by a comma (e.g. “Bx,Btot”).

With this parameter user may limit the set of physical parameters returned by the method.
Default value: All parameters included in the given NumericalOutput.

• **url_XYZ** (url, Mandatory)

URL of the VOTable file containing the 3D coordinates of the points in which the interpolation is performed. The coordinate axis labels used in the VOTable file must be those given in Spase/SimulationRun/SimulationDomain/CoordinatesLabel . The coordinates of the points must be expressed in the coordinate system given in Spase/SimulationRun/SimulationDomain/CoordinatesSystem.

• **extraParams** (Optional)

• InterpolationMethod (enumerated String, Optional)

Method used in interpolation. Possible values are:
Linear : Field value at given point is linearly interpolated from the field values at the corner points of the enclosing grid cube.
NearestGridPoint : Field value is taken as the value at the nearest grid point.

Default value: Linear.

• OutputFileType (enumerated String, Optional)

Type of the output file. Possible values are VOTable (default) and netCDF.

Default value: VOTable.

Output:

• **url_Param** (uri)

URL of the result data file containing the 3D position coordinates and the interpolated values of the requested variables.

Notes:

- The input VOTable file may contain additional fields (e.g. time) besides the 3D coordinate fields (X,Y,Z). These additional fields are preserved in the final output VOTable file.
- Due to memory limitations in PHP server the maximum number of input data points that can currently be handled is around 200 000. In this case the processing time is around 10 seconds. For larger input data sets the user should make several independent calls with smaller data sets.

Example:

PHP client code:

```
 // Define the method file and create client

 $methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
 $client = new SoapClient($methods_file);

 // Define input parameters for getDataPointValue

 $params = array (
     'ResourceID' =>
         'spase://IMPEX/NumericalOutput/FMI/HYB/mars/
          spiral_angle_runset_20130607_mars_20deg/Mag',
     'url_XYZ' =>
         'http://impex-fp7.fmi.fi/ws_tests/input/getDataPointValue_input.vot',
     'Variable'    => 'Bx, Btot',
     'extraParams' => array('OutputFileType' => 'VOTable')
);

 // Call the server and display the resulting url

 $data_url = $client->getDataPointValue($params);
 echo $data_url;
```

Input VOTable file:

```
<?xml version='1.0'?>
<VOTABLE version="1.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
        http://www.ivoa.net/xml/VOTable/v1.2"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
<!--
 ! VOTable written by FMI web service getVOTableURL
 ! 2014-03-18T08:29:57+0000
 !-->
<RESOURCE>
<TABLE name="Points" nrows="2">
<DESCRIPTION>
    Test input VOTable file for getDataPointValue method
</DESCRIPTION>
<FIELD ID="col_X" name="X" ucd="pos.cartesian.x" unit="m" datatype="float" />
<FIELD ID="col_Y" name="Y" ucd="pos.cartesian.y" unit="m" datatype="float" />
<FIELD ID="col_Z" name="Z" ucd="pos.cartesian.z" unit="m" datatype="float" />
<DATA>
<TABLEDATA>
    <TR>
        <TD>4000000</TD>
        <TD>-4000000</TD>
        <TD>-800000</TD>
    </TR>
    <TR>
        <TD>4000000</TD>
        <TD>4000000</TD>
        <TD>800000</TD>
    </TR>
</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

Output VOTable file:

```
<?xml version='1.0'?>
<VOTABLE version="1.2"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
       http://www.ivoa.net/xml/VOTable/v1.2"
   xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
<!--
  !  VOTable written by STIL version 3.0-3 (uk.ac.starlink.votable.VOTableWriter)
  !  at 2014-03-18T14:38:20
  !-->
<RESOURCE>
<TABLE nrows="2">
<DESCRIPTION>
Test input VOTable file for getDataPointValue method
</DESCRIPTION>
<FIELD ID="col_X" datatype="float" name="X" ucd="pos.cartesian.x" unit="m"/>
<FIELD ID="col_Y" datatype="float" name="Y" ucd="pos.cartesian.y" unit="m"/>
<FIELD ID="col_Z" datatype="float" name="Z" ucd="pos.cartesian.z" unit="m"/>
<FIELD datatype="float" name="Bx" ucd="phys.magField" unit="T">
    <DESCRIPTION>Magnetic field components</DESCRIPTION>
</FIELD>
<FIELD datatype="float" name="Btot" ucd="phys.magField" unit="T">
    <DESCRIPTION>TotalMagnetic field</DESCRIPTION>
</FIELD>
<DATA>
<TABLEDATA>
    <TR>
        <TD>4000000.0</TD>
        <TD>-4000000.0</TD>
        <TD>-800000.0</TD>
        <TD>1.4051E-8</TD>
        <TD>1.8813E-8</TD>
    </TR>
    <TR>
        <TD>4000000.0</TD>
        <TD>4000000.0</TD>
        <TD>800000.0</TD>
        <TD>8.48494E-9</TD>
        <TD>8.82154E-9</TD>
    </TR>
</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

2.2 getDataPointValueSpacecraft

This method is used to compute values of various physical quantities (e.g. density, magnetic field strengths etc.) along the path of given spacecraft.

Input parameters:

• **ResourceID** (String, Mandatory)

ResourceID of a <NumericalOutput> element in the tree.xml file. This NumericalOutput element defines at the same time the simulation run and the set of physical parameters (e.g. density, magnetic field) which are the output from this simulation run. The correspondence in the data model is: "Spase/NumericalOutput/ResourceID".

• **Variable** (List, Optional)

List (string) of individual physical parameters (e.g. Density, Pressure, Bx) whose values are interpolated along the spacecraft path. The tokens for the physical parameters are given in the <ParameterKey> element in the <NumericalOutput> element. Correspondence in the data model: "Spase/NumericalOutput/Parameter/ParameterKey".
If two or more parameters are given they must be separated by a comma (e.g. "Bx,Btot").

With this parameter user may limit the set of physical parameters returned by the method.

Default value: All parameters included in the given NumericalOutput.

• **Spacecraft_name** (enumerated string, Mandatory)

A string indicating the name of the spacecraft as defined by AMDA. The following names are currently recognized: VEX, MEX, MGS, MAVEN, MESSENGER, CLUSTER1, CLUSTER2, CLUSTER3, CLUSTER4, GEOTAIL, IMP-8, POLAR.

• **StartTime** (dateTime, Mandatory)

Time of first point in the spacecraft path to be included. Time format as defined by ISO 8601 standard (e.g. "2010-05-22T12:50:00.000").

• **StopTime** (dateTime, Mandatory)

Time of last point in the spacecraft path to be included. Time format as defined by ISO 8601 standard (e.g. "2010-05-22T18:10:00.000").

• **Sampling** (duration, Mandatory)

Time interval between two consecutive points along the spacecraft path. Time format as defined by ISO 8601 standard (e.g. “PT60S”).

- **extraParams** (Optional)
  - InterpolationMethod (enumerated String, Optional)

    Method used in interpolation. Possible values are:
    Linear : Field value at given point is linearly interpolated from the field values at the corner points of the enclosing grid cube.
    NearestGridPoint : Field value is taken as the value at the nearest grid point.

    Default value: Linear.
  - OutputFileType (enumerated String, Optional)

    Type of the output file. Possible values are VOTable and netCDF.

    Default value: VOTable.

Output:

- **url_Param** (uri)

  URL of the result data file containing time, 3D position coordinates of the spacecraft and the interpolated values of the requested variables.

Notes:

- The spacecraft orbital positions are retrieved from AMDA by using their web services (http://cdpp1.cesr.fr/AMDA/php/AMDA_METHODS_WSDL.php?wsdl). The list of possible ‘Spacecraft_name’s may be found by using the web service ‘getParameterList’. See documentation ‘Web Services provided by AMDA’ in IMPEx folder in google drive.

Example:

PHP client code:

```
 // Define the method file and create client

 $methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
 $client = new SoapClient($methods_file);

 // Define input parameters for getDataPointValueSpacecraft

 $params = array (
      'ResourceID' =>
        'spase://IMPEX/NumericalOutput/FMI/GUMICS/earth/synth_stationary/solarmin/
EARTH___n_T_Vx_Bx_By_Bz__7_100_600_3p_03_15m/tilt15p/H+_mstate',
      'Variable'  => 'Density',
      'Spacecraft_name' => 'CLUSTER3',
      'StartTime' => '2010-08-02T00:00:00',
      'StopTime'  => '2010-08-02T01:00:00',
      'Sampling'  => 'PT600S',
      'extraParams' => array('OutputFileType' => 'VOTable')
 );

 // Call the server and display the resulting url

 $data_url = $client->getDataPointValueSpacecraft($params);
 echo $data_url;
```

Output VOTable file:

```
 <?xml version='1.0'?>
 <VOTABLE version="1.2"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
         http://www.ivoa.net/xml/VOTable/v1.2"
     xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
 <!--
  ! VOTable written by FMI web service
  ! at 2014-03-18T15:10:58+0200
  !-->
 <RESOURCE>
 <TABLE name="Earth_H+_mstate" nrows="6">
 <DESCRIPTION>
     SimulationModel             : FMI_GUMICS
     SimulationModel_ResourceID : spase://IMPEX/SimulationModel/FMI/GUMICS/
     SimulationRun_ResourceID   :
spase://IMPEX/SimulationRun/FMI/GUMICS/earth/synth_stationary/solarmin/EARTH___n_T
_Vx_Bx_By_Bz__7_100_600_3p_03_15m/tilt15p
     NumericalOutput_ResourceID : spase://IMPEX/NumericalOutput/FMI/GUMICS/
earth/synth_stationary/solarmin/EARTH___n_T_Vx_Bx_By_Bz__7_100_600_3p_03_15m/tilt1
5p/H+_mstate
     Content description        : H+ computed physical quantities
     Object                     : Earth
     Object radius              : 6371200 m
     Spacecraft                 : c3_xyz
     Start time                 : 2010-08-02T00:00:00.000
     Stop time                  : 2010-08-02T01:00:00.000
     Sampling                   : PT600S
 </DESCRIPTION>
 <FIELD ID="col1" arraysize="*" datatype="char" name="Time" ucd="time.epoch"
xtype="dateTime"/>
 <FIELD ID="col2" datatype="float" name="x" ucd="pos.cartesian.x" unit="m"/>
 <FIELD ID="col3" datatype="float" name="y" ucd="pos.cartesian.y" unit="m"/>
 <FIELD ID="col4" datatype="float" name="z" ucd="pos.cartesian.z" unit="m"/>
 <FIELD ID="col5" datatype="float" name="Density" ucd="phys.density"
unit="1/m^3"/>
 <DATA>
 <TABLEDATA>
     <TR>
         <TD>2010-08-02T00:05:00.000</TD>
         <TD>-3.714230e+7</TD>
         <TD>8.569960e+6</TD>
         <TD>-2.286330e+7</TD>
         <TD>1.194940e+5</TD>
     </TR>
     <TR>
         <TD>2010-08-02T00:15:00.000</TD>
         <TD>-3.544070e+7</TD>
         <TD>9.267550e+6</TD>
```

```
        <TD>-2.158910e+7</TD>
        <TD>1.210100e+5</TD>
    </TR>
    <TR>
        <TD>2010-08-02T00:25:00.000</TD>
        <TD>-3.373830e+7</TD>
        <TD>9.919580e+6</TD>
        <TD>-2.032370e+7</TD>
        <TD>1.283250e+5</TD>
    </TR>
    <TR>
        <TD>2010-08-02T00:35:00.000</TD>
        <TD>-3.196350e+7</TD>
        <TD>1.054990e+7</TD>
        <TD>-1.901500e+7</TD>
        <TD>1.425640e+5</TD>
    </TR>
    <TR>
        <TD>2010-08-02T00:45:00.000</TD>
        <TD>-3.010950e+7</TD>
        <TD>1.115350e+7</TD>
        <TD>-1.765910e+7</TD>
        <TD>1.616280e+5</TD>
    </TR>
    <TR>
        <TD>2010-08-02T00:55:00.000</TD>
        <TD>-2.826760e+7</TD>
        <TD>1.169680e+7</TD>
        <TD>-1.632380e+7</TD>
        <TD>1.852490e+5</TD>
    </TR>
</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

2.3 getSurface

This method is used to compute values of various physical quantities (e.g. density, magnetic field strengths etc.) for given simulation run in points which are located on a user specified surface. The surface is specified by defining a vector which is normal to the plane plus a single point on the surface. The method generates a meshgrid on the surface and then calls the getDataPointValue method to compute the required quantities.

Input parameters:

• **ResourceID** (String, Mandatory)

ResourceID of a <NumericalOutput> element in the tree.xml file. This NumericalOutput element defines at the same time the simulation run and the set of physical parameters (e.g. density, magnetic field) which are the output from this simulation run. The correspondence in the data model is: "Spase/NumericalOutput/ResourceID".

• **Variable** (List, Optional)

List (string) of individual physical parameters (e.g. Density, Pressure, Bx) whose values are interpolated in given points. The tokens for the physical parameters are given in the <ParameterKey> element in the <NumericalOutput> element. Correspondence in the data model: "Spase/NumericalOutput/Parameter/ParameterKey". If two or more parameters are given they must be separated by a comma (e.g. "Bx,Btot").

With this parameter user may limit the set of physical parameters returned by the method.

Default value: All parameters included in the given NumericalOutput.

• **PlaneNormalVector** (List of floats, Mandatory)

List (string) of three floats which represent the components of a normalized vector normal to the requested plane. The components must be separated by a space " " or comma ",". The coordinate axis labels used in the VOTable file must be those given in Spase/SimulationRun/SimulationDomain/CoordinatesLabel . The coordinates of the points must be expressed in the coordinate system given in Spase/SimulationRun/SimulationDomain/CoordinatesSystem.

• **PlanePoint** (List of floats, Mandatory)

List (string) of three floats which represent the 3D coordinates of a point lying in the requested plane. The components must be separated by a space " " or comma ",".

• **extraParams** (Optional)

- Resolution (float, Optional)

  Spatial resolution of the grid mesh.

  Default value: Grid cell size used in the simulation run (Spase/SimulationRun/SimulationDomain/GridCellSize).

- InterpolationMethod (enumerated String, Optional)

  Method used in interpolation. Possible values are:
  Linear : Field value at given point is linearly interpolated from the field values at the corner points of the enclosing grid cube.
  NearestGridPoint : Field value is taken as the value at the nearest grid point.

  Default value: Linear

- OutputFileType (enumerated String, Optional)

  Type of the output file. Possible values are VOTable and netCDF.

  Default value: VOTable

Output:

- **url_Param** (uri)

  URL of the result data file containing the 3D position coordinates of the mesh points on the plane and the interpolated values of the requested variables.

Notes:

- Currently the surface must be parallel to XY-, YZ- or ZX-plane i.e. the normal vector must be either “0 0 1”, “1 0 0” or “0 1 0”.

Example:

PHP client code:

```
 // Define the method file and create client

 $methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
 $client = new SoapClient($methods_file);

 // Define input parameters for getSurface

 $params = array (

'ResourceID'=>'spase://IMPEX/NumericalOutput/FMI/HYB/mars/Mars_testrun_lowres/O+_a
ve_hybstate',
      'Variable'     => 'Density',
      'PlaneNormalVector' => '1.0 0.0 0.0',     // YZ plane
      'PlanePoint'  => '3.7e6 0.0 0.0',         // x = 3 700 km
```

```
     'extraParams' => array('OutputFileType' => 'VOTable')
);

 // Call the server and display the resulting url

 $data_url = $client->getSurface($params);
 echo $data_url;
```

Output VOTable file:

```
 <?xml version='1.0'?>
 <VOTABLE version="1.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
    http://www.ivoa.net/xml/VOTable/v1.2"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
 <!--
  ! VOTable written by FMI web service
  ! at 2014-03-19T09:40:43+0200
  !-->
 <RESOURCE>
 <TABLE name="Mars_O+_ave_hybstate" nrows="1681">
 <DESCRIPTION>
     SimulationModel             : FMI_HYB_SimulationModel
     SimulationModel_ResourceID : spase://IMPEX/SimulationModel/FMI/HYB/
     SimulationRun_ResourceID   :
spase://IMPEX/SimulationRun/FMI/HYB/mars/Mars_testrun_lowres
     NumericalOutput_ResourceID : spase://IMPEX/NumericalOutput/FMI/HYB/
mars/Mars_testrun_lowres/O+_ave_hybstate
     Content description        : O+ computed physical quantities
     Object                     : Mars
     Object radius              : 3393500 m
 </DESCRIPTION>
 <FIELD ID="col1" datatype="float" name="x" ucd="pos.cartesian.x" unit="m"/>
 <FIELD ID="col2" datatype="float" name="y" ucd="pos.cartesian.y" unit="m"/>
 <FIELD ID="col3" datatype="float" name="z" ucd="pos.cartesian.z" unit="m"/>
 <FIELD ID="col4" datatype="float" name="Density" ucd="phys.density"
unit="1/m^3"/>
 <DATA>
 <TABLEDATA>
     <TR>
         <TD>3.700000e+6</TD>
         <TD>-1.357400e+7</TD>
         <TD>-1.357400e+7</TD>
         <TD>6.952160e+0</TD>
     </TR>
     <TR>
         <TD>3.700000e+6</TD>
         <TD>-1.357400e+7</TD>
         <TD>-1.289530e+7</TD>
         <TD>5.222300e+0</TD>
     </TR>
.
.
.
.


</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

2.4 getVOTableURL

This method creates a VOTable format file from user defined data, stores the file in FMI server for later use and returns an URL to this VOT file. This is an auxiliary routine that does not use any tree.xml files and is not related to the simulation run data base in any way. There are two basic motivations for the creation of this service:

1. VOTable files are used in many IMPEx web services as input files so users who are not familiar with VOTable format may now easily create .vot files from their own data.
2. Users don't have to set up their own web server for storing the input files as the files are now stored at FMI's server.

The method is able to create simple .vot files and no attempt is made to provide any sophisticated features of the VOTable format.

Input parameters:

• **Table_name** (String, Optional)

'Table_name' parameter will appear as the name attribute of the <TABLE> element in the VOTable file.

Default value : "VOTable"

• **Description** (String, Optional)

'Description' will be set to the <DESCRIPTION> field in the <TABLE> element.

Default value : "VOTable file generated by FMI web service getVOTableURL"

• **Fields** (array of Field's, Mandatory)

The 'Fields' parameter is an array of structures called 'Field'. Each 'Field' corresponds to VOTable <FIELD> element and describes a single quantity (e.g. X coordinate, Bx, Temperature, …). The 'Field' element itself is an associative array where the key names are the allowed attributes of the <FIELD> element ('name', 'ID', 'datatype', 'unit', 'ucd', 'xtype', ' utype'). Furthermore the 'Field' element contains element with name 'data'. This is the array where the actual data for the given field is located.

Output:

• **url_Param** (uri)

URL of the result VOTable format data file.

Notes:

- Each of the data arrays must be of same length. The only mandatory elements in the ‘Field’ element are ‘name’ and ‘data’. The method recognizes the names ‘Time’, ‘X’, ‘Y’, ‘Z’, ‘Ux’, ‘Uy’, ‘Uz’, ‘Utot’, ‘Bx’, ‘By’, ‘Bz’ and ‘Btot’ and will set the other attributes accordingly. If ’Time’ field is included it must be given as an array of ISO8601 format strings.

Example:

PHP client code:

```
 // Generate the input data

 $count = 1000;       // Number of points
 $radius = 6.371e6;   // = 6371 km
 $StartTime = strtotime(“2013-12-11T16:04.12”);

 $My_Time = array();
 $My_X = array();
 $My_Y = array();
 $My_Z = array();
 $My_Accl = array();


 for ($i=0; $i < $count; $i++) {
     My_Time[$i] = date(“Y-m-d\TH:i:s”, $StartTime + 60*$i);
     $My_X[$i] = $radius*cos(2*M_PI*$i/$count);
     $My_Y[$i] = $radius*sin(2*M_PI*$i/$count);
     $My_Z[$i] = 0.0;
     $My_Accl[$i] = 10*rand()/getrandmax(); // Just random values in (0,10)
 }

 // Define the method file and create client

 $methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
 $client = new SoapClient($methods_file);

 // Define input parameters for getVOTableURL

 $params = array (
     'Table_name'   =>'My test run',
     'Description'  => 'VOTable format demo',
     'Fields' => array(
         array( 'name'  => 'Time', 'data' => $My_Time ),
         array( 'name'  => 'X', 'data' => $My_X ),
         array( 'name'  => 'Y', 'data' => $My_Y ),
         array( 'name'  => 'Z', 'data' => $My_Z ),
         array( 'name'  => 'Accl', 'data' => $My_Accl, 'unit'  => 'm/s^2',
                'datatype' => 'double', 'ucd'  => 'phys.acceleration' ),
     )
);

 // Call the server and display the resulting url

 $data_url = $client->getVOTableURL($params);
 echo $data_url;
```

Output VOTable file:

```
<?xml version='1.0'?>
<VOTABLE version="1.2"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
   http://www.ivoa.net/xml/VOTable/v1.2"
   xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
<!--
 ! VOTable written by FMI web service
 ! at 2014-03-19T09:40:43+0200
 !-->
<RESOURCE>
<TABLE name="My test run" nrows="1000">
<DESCRIPTION>
    VOTable format demo
</DESCRIPTION>
<FIELD ID="col_Time" name="Time" xtype="dateTime" arraysize="*" ucd="time.epoch"
       datatype="char" />
<FIELD ID="col_X" name="X" ucd="pos.cartesian.x" unit="m" datatype="float" />
<FIELD ID="col_Y" name="Y" ucd="pos.cartesian.y" unit="m" datatype="float" />
<FIELD ID="col_Z" name="Z" ucd="pos.cartesian.z" unit="m" datatype="float" />
<FIELD ID="col_Accl" name="Accl" unit="m/s^2" datatype="double"
       ucd="phys.acceleration" />
<DATA>
<TABLEDATA>
    <TR>
        <TD>2013-12-11T16:04:12</TD>
        <TD>6371000</TD>
        <TD>0</TD>
        <TD>0</TD>
        <TD>9.8937996895489</TD>
    </TR>
    <TR>
        <TD>2013-12-11T16:05:12</TD>
        <TD>6370874.2419144</TD>
        <TD>40029.910204576</TD>
        <TD>0</TD>
        <TD>2.5540249713482</TD>
    </TR>
    <TR>
        <TD>2013-12-11T16:06:12</TD>
        <TD>6370496.9726225</TD>
        <TD>80058.240096839</TD>
        <TD>0</TD>
        <TD>7.312020779267</TD>
    </TR>
    <TR>
        <TD>2013-12-11T16:07:12</TD>
        <TD>6369868.2070181</TD>
        <TD>120083.40942687</TD>
        <TD>0</TD>
        <TD>6.0990113048344</TD>
    </TR>
.
.
.


</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

2.5 getMostRelevantRun

This method is intended to help users to find those simulation runs in SMDB's tree.xml file whose input parameters best match the given solar wind conditions. For each simulation run in the tree.xml file the method computes a 'difference index'. This 'difference index' is a sum of individual difference indices (one for each solar wind parameter defined in the input parameter set). If $value_{sw}$, $scale_{sw}$ and $weight_{sw}$ denote the value, scale and weight, respectively, of given input value for solar wind parameter 'sw' and $value_{sw}(i)$ is the value of this solar wind parameter for i'th run then the difference index for this run (i) is computed as

$$S_{diff}(i) = \sum weight_{sw} * [(value_{sw} - value_{sw}(i))/scale_{sw}]^2$$

where the sum runs over all given solar wind parameters 'sw'. The smaller the 'difference index' the better the match. A perfect match will give 'difference index' zero.
As an output the method returns the list of matching runs ordered according to the total difference index in ascending order. So the 'most relevat run' is the first element. The output list is in json format.

Input parameters:

• **Object** (String, Mandatory)

Name of the object (e.g. 'Mercury', 'Venus', 'Earth', …). The correspondence in the data model is: "Spase/SimulationRun/SimulatedRegion".

• **RunCount** (positiveInteger, Optional)

Number of most relevant runs returned.

Default value : 1.

• **SW_parameters** (array of solar wind parameters, Mandatory)

Array of solar wind parameters included in matching process. At least one parameter must be defined. The elements of this array are associative arrays with element names 'value', 'weight', 'scale' and 'function'. 'value' is mandatory and defines the numerical value of given parameter. The unit must be the base SI unit. 'weight' is optional and defines the numerical weight factor used for given parameter (weight*[(value-value_run)/scale]^2) in the least-squares fitting. Default = 1.0. 'scale' is optional and is used to make the terms in the least-squares fitting dimensionless (value - value_run)/scale. The unit must be the same as in 'value'. Default scales in various planetary environments are from Slavin & Holzer: Solar Wind Flow About the Terrestrial Planets Modeling Bow Shock Position and Shape, Journal of Geophysical Research, Vol 86, No. A13, pp. 11401-11418, December 1, 1981. 'function' is only used with parameter 'SW_Function' where it is used to describe the mathematical expression of a user defined quantity.

List of solar wind parameters:

SW_density	Solar wind H+ number density (1/m^3). Simulation run data from SW H+ population in Spase/SimulationRun/inputPopulation.
SW_Utot	Solar wind H+ flow velocity (m/s). Simulation run data from SW H+ population in Spase/SimulationRun/inputPopulation.
SW_Temperature	Solar wind H+ temperature (K). Simulation run data from SW H+ population in Spase/SimulationRun/inputPopulation.
SW_Btot	Solar wind magnetic field, total field (T). Simulation run data from IMF field in Spase/SimulationRun/inputField.
SW_Bx	Solar wind magnetic field, X component (T) in the "Object-Sun-Orbital plane" coordinate system, e.g. VSO for Venus and MSO for Mars. Simulation run data from IMF field in Spase/SimulationRun/inputField.
SW_By	Solar wind magnetic field, Y component (T) in the "Object-Sun-Orbital plane" coordinate system, e.g. VSO for Venus and MSO for Mars. Simulation run data from IMF field in Spase/SimulationRun/inputField.
SW_Bz	Solar wind magnetic field, Z component (T) in the "Object-Sun-Orbital plane" coordinate system, e.g. VSO for Venus and MSO for Mars. Simulation run data from IMF field in Spase/SimulationRun/inputField.
Solar_F10.7	Solar 10.7cm flux (unit = SFU). Simulation run data from SolarFlux data in Spase/SimulationRun/inputParameter
SW_Function	User defined quantity contructed from other 'SW_xxx' quantities. The mathematical formula is described as a string in field 'function'. E.g. SW_Function => array('value' => 0.5 , 'function' => 'SW_Bx/SW_Btot').

Output:

• **json_string** (string)

List of most relevant runs in json format. The json object contains an object ‘input’ which lists the given input parameters and an array “runs” which lists the most relevant runs (best fit first). Each run object contains the ResourceID of the run, total difference factor, individual difference factors for each SW parameter and the values of SW parameters in the <SimulationRun> element. It also contains the ResourceIDs of the NumericalOutput elements associated to this SimulationRun.

Notes:

- A demo web page for this method is located at: http://impex-fp7.fmi.fi/ws_tests/getMostRelevantRun

Example:

PHP client code:

```
 // Define the method file and create client

 $methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
 $client = new SoapClient($methods_file);

 // Define input parameters for getMostRelevantRun

 $params = array (
     'Object'   => 'Earth',
     'RunCount' => 2,
     'SW_parameters' => array(
         'SW_Density'  => array( 'value' => 5e6,   'weight' => 2 ),
         'SW_Utot'     => array( 'value' => 4.5e5, 'weight' => 1 ),
         'SW_Function' => array( 'value' => 0.5, 'scale' => 1,
                                 'function' => 'abs(SW_Bx/SW_Btot)')
     )
);

 // Call the server and convert json string into a php object

 $data_json = $client->get getMostRelevantRun ($params);
 $data = json_decode($data_json);
```

Output JSON string:

```
 {
    "input" : {
        "Object" : "Earth",
        "RunCount" : 2,
        "SW_parameters" : {
            "SW_Density" : {
                "value" : 5000000,
                "weight" : 2,
                "scale" : 7000000
            },
            "SW_Utot" : {
                "value" : 4500,
                "weight" : 1,
                "scale" : 430000
            },
            "SW_Function" : {
                "value" : 0.5,
                "weight" : 1,
                "scale" : 1,
                "function" : "abs(SW_Bx/SW_Btot)"
            }
        }
    },
    "runs" : [
        {
            "ResourceID" :
"spase://IMPEX/SimulationRun/FMI/GUMICS/earth/synth_dynamical/200301?20030111_0436
00",
            "ResourceName" : "GUMICS_Earth_run_20030111_043600",
            "NumericalOutput" : [

"spase://IMPEX/NumericalOutput/FMI/GUMICS/earth/synth_dynamical/200301/H+_mstate?2
0030111_043600",

"spase://IMPEX/NumericalOutput/FMI/GUMICS/earth/synth_dynamical/200301/Mag?2003011
1_043600"
            ],
            "S_diff" : "0.00010845229722884",
            "S_diff_n" : {
                "SW_Density" : "8.8890158734694E-5",
                "SW_Utot" : "1.0005003854846E-5",
                "SW_Function" : "9.5571346393E-6"
            },
            "Param_values" : {
                "SW_Density" : "4953333",
                "SW_Temperature" : "220238.3",
                "SW_Utot" : "448639.88044174",
                "SW_Btot" : "7.8638384910233E-9",
                "SW_Bx" : "3.95623E-9",
                "SW_By" : "-6.79E-9",
                "SW_Bz" : "2.9E-10",
                "SW_Function" : "0.50309146157008"
            }
        },
        {
            "ResourceID" :
"spase://IMPEX/SimulationRun/FMI/GUMICS/earth/synth_dynamical/200211?20021116_1931
00",
            "ResourceName" : "GUMICS_Earth_run_20021116_193100",
            "NumericalOutput" : [

"spase://IMPEX/NumericalOutput/FMI/GUMICS/earth/synth_dynamical/200211/H+_mstate?2
0021116_193100",

"spase://IMPEX/NumericalOutput/FMI/GUMICS/earth/synth_dynamical/200211/Mag?2002111
6_193100"
            ],
            "S_diff" : "0.00015429032235757",
            "S_diff_n" : {
                "SW_Density" : "8.265306122449E-5",
                "SW_Utot" : "5.092996233805E-5",
                "SW_Function" : "2.070729879503E-5"
            },
            "Param_values" : {
                "SW_Density" : "4955000",
                "SW_Temperature" : "52151",
                "SW_Utot" : "446931.29505552",
                "SW_Btot" : "7.0511731116177E-9",
                "SW_Bx" : "3.4935E-9",
                "SW_By" : "-5.72E-9",
                "SW_Bz" : "-2.19E-9",
                "SW_Function" : "0.49544947269044"            }
        }
    ]
}
```

2.6 getFieldLine

This method computes field lines or stream lines inside the simulation box. The user will provide a set of 3D position coordinates which act as starting points for the field/stream line tracing. The tracing may be performed forwards, backwards or both. The method will return an URL to a VOTable file which contains the field/stream lines as set of 3D position coordinates and the requested physical quantities at these points.

Input parameters:

• **ResourceID** (String, Mandatory)

ResourceID of a <NumericalOutput> element in the tree.xml file. This NumericalOutput element defines at the same time the simulation run and the set of physical parameters (e.g. density, magnetic field) which are the output from this simulation run. The correspondence in the data model is: "Spase/NumericalOutput/ResourceID".

• **Variable** (List, Optional)

List (string) of individual physical parameters (e.g. Density, Pressure, Bx) whose values are interpolated in given points. The tokens for the physical parameters are given in the <ParameterKey> element in the <NumericalOutput> element. Correspondence in the data model: "Spase/NumericalOutput/Parameter/ParameterKey". If two or more parameters are given they must be separated by a comma (e.g. "Bx,Btot").

With this parameter user may limit the set of physical parameters returned by the method.

Default value: All parameters included in the given NumericalOutput element.

• **url_XYZ** (url, Mandatory)

URL of the VOTable file containing the 3D coordinates of the points which represent the starting points for field/stream line tracing. The coordinate axis labels used in the VOTable file must be those given in Spase/SimulationRun/SimulationDomain/CoordinatesLabel . The coordinates of the points must be expressed in the coordinate system given in Spase/SimulationRun/SimulationDomain/CoordinatesSystem.

• **extraParams** (Optional)

• Direction (enumerated String, Optional)

Direction of field/stream line tracing. Possible values are 'Forward', 'Backward' or 'Both'.

Default value: 'Forward'

- StepSize (float, Optional)

  The spatial distance of consequtive points along the field/stream line in the output data file.

  Default value = ¼ * smallest grid cell size in the simulation run.

- MaxSteps (positiveInteger, Optional)

  The maximum number of steps per field/stream line.

  Default value: 100

- StopCondition_Radius (float, Optional)

  The field/stream line tracing is stopped if the distance from the center of object is less than 'StopCondition_Radius'.

  Default value: 0

- StopCondition_Region (list of floats, Optional)

  The field/stream line tracing is stopped if the point is outside of the box defined by StopCondition_Region = "X_min X_max Y_min Y_max Z_min Z_max".

  Default value: Simulation run box. (Defined by Spase/SimulationRun/SimulationDomain/ValidMin and Spase/SimulationRun/SimulationDomain/ValidMax).

- OutputFileType(enumerated string, Optional)

  Type of the output file. Currently only VOTable is supported.

  Default value: VOTable.

Output:

- **url_Param** (uri)

  URL of the result data file containing the 3D position coordinates of points along the field lines and the interpolated values of the requested variables. The VOTable field name 'Line_no' identifies the field lines.

Notes:

- In the current version the input parameter 'Variable' is neglected. If the 'ResourceID' points to magnetic field then field lines are computed and Bx,By,Bz and Btot values are included in the resulting VOTable file. If 'ResourceID' points to a particle species then velocity stream lines are computed and Ux,Uy,Uz and Utot are returned.

Example:

PHP client code:

```
// Define the method file and create client

$methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
$client = new SoapClient($methods_file);

// Define input parameters for getFieldLine

$params = array (
    'ResourceID'=>'
spase://IMPEX/NumericalOutput/FMI/HYB/mars/spiral_angle_runset_20130607_mars_90deg
/Mag',
    'url_XYZ' =>
        'http://impex-fp7.fmi.fi/ws_tests/input/getFieldLine_input.vot',
    'extraParams' => array('Direction' => 'Forward')
);

// Call the server and display the resulting url

$data_url = $client->getFieldLine($params);
echo $data_url;
```

Input VOTable file:

```
<?xml version='1.0'?>
<VOTABLE version="1.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
        http://www.ivoa.net/xml/VOTable/v1.2"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
<!--
 ! VOTable written by FMI web service getVOTableURL
 ! 2014-03-19T13:09:01+0000
 !-->
<RESOURCE>
<TABLE name="Points" nrows="30">
<DESCRIPTION>
    Test input VOTable file for getFieldLine method
</DESCRIPTION>
<FIELD ID="col_X" name="X" ucd="pos.cartesian.x" unit="m" datatype="float" />
<FIELD ID="col_Y" name="Y" ucd="pos.cartesian.y" unit="m" datatype="float" />
<FIELD ID="col_Z" name="Z" ucd="pos.cartesian.z" unit="m" datatype="float" />
<DATA>
<TABLEDATA>
    <TR>
        <TD>4290000</TD>
        <TD0</TD>
        <TD0</TD>
    </TR>
    <TR>
        <TD>4390000</TD>
        <TD>0</TD>
        <TD>0</TD>
    </TR>
.
.
.
</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

Output VOTable file:

```
 <?xml version='1.0'?>
 <VOTABLE version="1.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
    http://www.ivoa.net/xml/VOTable/v1.2"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
 <!--
  ! VOTable written by FMI web service getVOTableURL
  ! at 2014-03-19T13:09:20+0000
  !-->
 <RESOURCE>
 <TABLE name="Mars_Mag" nrows="4910">
 <DESCRIPTION>
     SimulationModel            : FMI_HYB_SimulationModel
     SimulationModel_ResourceID : spase://IMPEX/SimulationModel/FMI/HYB/
     SimulationRun_ResourceID   :
spase://IMPEX/SimulationRun/FMI/HYB/mars/spiral_angle_runset_20130607_mars_90deg
     NumericalOutput_ResourceID :

spase://IMPEX/NumericalOutput/FMI/HYB/mars/spiral_angle_runset_20130607_mars_90deg
/Mag
     Content description        : Magnetic field vector field lines
     Object                     : Mars
     Object radius              : 3393500 m
     Input parameters :
       ResourceID    :
          impex://FMI/HWA/HYB/mars/spiral_angle_runset_20130607_mars_90deg/Mag
       url_XYZ       :
          http://impex-fp7.fmi.fi/ws_tests/input/getFieldLine_input.vot
       Direction     : Forward
 </DESCRIPTION>
 <FIELD ID="col_X" name="X" ucd="pos.cartesian.x" unit="m" datatype="float" />
 <FIELD ID="col_Y" name="Y" ucd="pos.cartesian.y" unit="m" datatype="float" />
 <FIELD ID="col_Z" name="Z" ucd="pos.cartesian.z" unit="m" datatype="float" />
 <FIELD ID="col_Bx" name="Bx" ucd="phys.magField" unit="T" datatype="float" />
 <FIELD ID="col_By" name="By" ucd="phys.magField" unit="T" datatype="float" />
 <FIELD ID="col_Bz" name="Bz" ucd="phys.magField" unit="T" datatype="float" />
 <FIELD ID="col_Btot" name="Btot" ucd="phys.magField" unit="T" datatype="float" />
 <FIELD ID="col_Line_no" name="Line_no" datatype="int" />
 <DATA>
 <TABLEDATA>
     <TR>
         <TD>4290000</TD>
         <TD>0</TD>
         <TD>0</TD>
         <TD>5.09981E-11</TD>
         <TD>1.56563E-8</TD>
         <TD>-8.92275E-12</TD>
         <TD>1.5656385601782E-8</TD>
         <TD>1</TD>
     </TR>
     <TR>
         <TD>4289720</TD>
         <TD>84749.4</TD>
         <TD>-127.926</TD>
         <TD>-1.53618E-10</TD>
         <TD>1.56389E-8</TD>
         <TD>-3.83634E-11</TD>
         <TD>1.5639701514108E-8</TD>
         <TD>1</TD>
     </TR>
.
.
.
</TABLEDATA>
</DATA>
```

```
</TABLE>
</RESOURCE>
</VOTABLE>
```

2.7 getParticleTrajectory

This method computes charged particle trajectories within given simulation run domain. The initial positions, velocities, masses and charges of the test particles must be defined in a VOTable format file. The method returns an url to a VOTable file which contains the computed trajectories for each particle.

Input parameters:

- **ResourceID** (String, Mandatory)

  ResourceID of a <NumericalOutput> element in the tree.xml file. This NumericalOutput element defines the simulation run. As there may be several NumericalOutput elements which belong to a single run any one of those ResourceID's will be accepted. The correspondence in the data model is: "Spase/NumericalOutput/ResourceID".

- **url_XYZ** (url, Mandatory)

  URL of the VOTable file containing initial particle positions, velocities, masses and charges. The coordinate axis labels used in the VOTable file must be those given in Spase/SimulationRun/SimulationDomain/CoordinatesLabel . The coordinates of the points must be expressed in the coordinate system given in Spase/SimulationRun/SimulationDomain/CoordinatesSystem. The VOTable file must contain fields named 'Mass' and 'Charge'. The units for these fields must be 'kg' and 'C', respectively.

- **extraParams** (Optional)

  - Direction (enumerated String, Optional)

    Direction of particle trajectory tracing. Possible values are 'Forward', 'Backward' or 'Both'.

    Default value: 'Forward'

  - StepSize (float, Optional)

    Length of one time step in tracing.

    Default value: ¼ * smallest grid cell size in the simulation run.

  - MaxSteps (positiveInteger, Optional)

    The maximum number of steps per trajectory.

    Default value: 200.

  - StopCondition_Radius (float, Optional)

    The trajectory tracing is stopped if the distance from the center of object is less than ‘StopCondition_Radius’.

Default value: 0

- StopCondition_Region (list of floats, Optional)

  The tracing is stopped if the point is outside of the box defined by StopCondition_Region = “X_min X_max Y_min Y_max Z_min Z_max”.

  Default value: Simulation run box. (Defined by Spase/SimulationRun/SimulationDomain/ValidMin and Spase/SimulationRun/SimulationDomain/ValidMax).

- InterpolationMethod (enumerated String, Optional)

  Method used in interpolation. Possible values are:
  Linear : Field value at given point is linearly interpolated from the field values at the corner points of the enclosing grid cube.
  NearestGridPoint : Field value is taken as the value at the nearest grid point.

  Default value: Linear

- OutputFileType(enumerated string, Optional)

  Type of the output file. Currently only VOTable is supported.

  Default value: VOTable

Output:

- **url_Param** (uri)

  URL of the result data file containing the 3D position coordinates of points along the particle trajectories. Also included is a field ‘Seconds’ which represents the time which has elapsed since the particle was at its initial position. If tracing ‘Direction’ is ‘Backward’ the ‘Seconds’ < 0. Note that the lines in the VOTable file are not necessarily time ordered. The VOTable field name ‘Particle_no’ is an integer and identifies different particles (similar to ‘Line_no’ in getFieldLine method).

Notes:

- For each particle the positions are given with ‘StepSize’ time intervals. The output file does not contain any computed physical quantities along the trajectory. If required they can be computed with other methods (e.g. getDataPointValue) as the output url of getParticleTrajectory may be given as an input url to other methods.

Example:

PHP client code:

```
 // Define the method file and create client

 $methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
 $client = new SoapClient($methods_file);

 // Define input parameters for getParticleTrajectory

 $params = array (
     'ResourceID'=>'

spase://IMPEX/NumericalOutput/FMI/HYB/mars/spiral_angle_runset_20130607_mars_90deg
/Mag',
     'url_XYZ' =>
         'http://impex-fp7.fmi.fi/ws_tests/input/getParticleTrajectory_input.vot',
     'extraParams' => array(
         'Direction' => 'Forward',
         'StepSize'  => 1,
         'MaxSteps' => 200
     )
);

 // Call the server and display the resulting url

 $data_url = $client->getParticleTrajectory($params);
 echo $data_url;
```

Input VOTable file:

```
 <?xml version='1.0'?>
 <VOTABLE version="1.2"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
         http://www.ivoa.net/xml/VOTable/v1.2"
     xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
 <!--
  ! VOTable written by FMI web service getVOTableURL
  ! 2014-03-19T13:09:01+0000
  !-->
 <RESOURCE>
 <TABLE name="Particle trajectories test run" nrows="100">
 <DESCRIPTION>
     Test input VOTable file for getParticleTrajectory method
 </DESCRIPTION>
 <FIELD ID="col_X" name="X" ucd="pos.cartesian.x" unit="m" datatype="float" />
 <FIELD ID="col_Y" name="Y" ucd="pos.cartesian.y" unit="m" datatype="float" />
 <FIELD ID="col_Z" name="Z" ucd="pos.cartesian.z" unit="m" datatype="float" />
 <FIELD ID="col_Ux" name="Ux" ucd="phys.veloc" unit="m/s" datatype="float" />
 <FIELD ID="col_Uy" name="Uy" ucd="phys.veloc" unit="m/s" datatype="float" />
 <FIELD ID="col_Uz" name="Uz" ucd="phys.veloc" unit="m/s" datatype="float" />
 <FIELD ID="col_Mass" name="Mass" ucd="phys.charge" unit="C" datatype="double" />
 <FIELD ID="col_Charge" name="Charge" unit="kg" datatype="double" ucd="phys.mass"
/>
 <DATA>
 <TABLEDATA>
     <TR>
         <TD>8475000</TD>
         <TD>-10170000</TD>
         <TD>-10170000</TD>
         <TD>-430000</TD>
         <TD>0</TD>
         <TD>0</TD>
         <TD>1.67262178E-27</TD>
         <TD>1.602177E-19</TD>
     </TR>
     <TR>
         <TD>8475000</TD>
         <TD>-10170000</TD>
         <TD>-7910000</TD>
         <TD>-430000</TD>
         <TD>0</TD>
         <TD>0</TD>
         <TD>1.67262178E-27</TD>
         <TD>1.602177E-19</TD>
     </TR>
 .
 .
 </TABLEDATA>
 </DATA>
 </TABLE>
 </RESOURCE>
 </VOTABLE>
```

Output VOTable file:

```
 <?xml version='1.0'?>
 <VOTABLE version="1.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
    http://www.ivoa.net/xml/VOTable/v1.2"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
 <!--
  ! VOTable written by FMI web service getVOTableURL
  ! at 2014-03-14T13:11:31+0000
  !-->
 <RESOURCE>
 <TABLE name="Mars_Mag" nrows="4869">
 <DESCRIPTION>
     Particle trajectories for a HYB simulation run computed by
         getParticleTrajectory FMI web service
     SimulationModel             : FMI_HYB_SimulationModel
     SimulationModel_ResourceID : spase://IMPEX/SimulationModel/FMI/HYB/
     SimulationRun_ResourceID   :

spase://IMPEX/SimulationRun/FMI/HYB/mars/spiral_angle_runset_20130607_mars_90deg
     NumericalOutput_ResourceID :

spase://IMPEX/NumericalOutput/FMI/HYB/mars/spiral_angle_runset_20130607_mars_90deg
/H+_ave_hybstate
     Content description        : H+ computed physical quantities particle
                                  trajectories
     Object                     : Mars
     Object radius              : 3390000 m
  Input parameters :
    ResourceID       :

spase://IMPEX/NumericalOutput/FMI/HYB/mars/spiral_angle_runset_20130607_mars_90deg
/Mag
    url_XYZ          :
        http://impex-fp7.fmi.fi/ws_tests/input/getParticleTrajectory_input.vot
    Direction        : Forward
    StepSize         : 1
    MaxSteps         : 200
 </DESCRIPTION>
 <FIELD ID="col_X" name="X" ucd="pos.cartesian.x" unit="m" datatype="float" />
 <FIELD ID="col_Y" name="Y" ucd="pos.cartesian.y" unit="m" datatype="float" />
 <FIELD ID="col_Z" name="Z" ucd="pos.cartesian.z" unit="m" datatype="float" />
 <FIELD ID="col_Seconds" name="Seconds" unit="s" datatype="float" ucd="time" />
 <FIELD ID="col_Particle_no" name="Particle_no" datatype="int" />
 <DATA>
 <TABLEDATA>
     <TR>
         <TD>8475000</TD>
         <TD>-10170000</TD>
         <TD>-10170000</TD>
         <TD>0</TD>
         <TD>1</TD>
     </TR>
     <TR>
         <TD>8044810</TD>
         <TD>-10170000</TD>
         <TD>-10169900</TD>
         <TD>1</TD>
         <TD>1</TD>
     </TR>
.
.
.
</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

2.8 getDataPointSpectra

This method may be used to compute energy spectra of given particle species at user specified set of 3D points for given simulation run.

Input parameters:

- **ResourceID** (String, Mandatory)

  ResourceID of a <NumericalOutput> element in the tree.xml file. This NumericalOutput element defines the simulation run. The correspondence in the data model is: "Spase/NumericalOutput/ResourceID". NumericalOutput elements which contain spectral data have <SimulationProduct> element set as 'Spectra'. The NumericalOutput element contains the energy channels used in the simulation in the element Spase/NumericalOutput/Parameter/Particle/EnergyRange.

- **url_XYZ** (url, Mandatory)

  URL of the VOTable file containing the 3D coordinates of the points in which the spectra are computed. The coordinate axis labels used in the VOTable file must be those given in Spase/SimulationRun/SimulationDomain/CoordinatesLabel . The coordinates of the points must be expressed in the coordinate system given in Spase/SimulationRun/SimulationDomain/CoordinatesSystem.

- **extraParams** (Optional)

  - InterpolationMethod (enumerated String, Optional)

    Method used in interpolation. Possible values are:
    Linear : Field value at given point is linearly interpolated from the field values at the corner points of the enclosing grid cube.
    NearestGridPoint : Field value is taken as the value at the nearest grid point.

    Default value: Linear

  - OutputFileType (enumerated string, Optional)

    Type of the output file. Currently only VOTable is supported.

    Default value: VOTable

  - EnergyChannel (list of channel BandNames, Optional)

    List of energy channel band names as defined in Spase/NumericalOutput/Parameter/Particle/EnergyRang/Bin/BandName. The method returns particle counts only on these energy channels.
    Note: Currently all energy channel counts are returned regardless of the value of this parameter.

    Default value: All energy channels listed in the given NumericalOutput element.

Output:

• **url_Param** (uri)

URL of the result data file containing the 3D position coordinates of points and the particle counts in specified energy channels at these points. The output file (VOTable) contains the following information:

- The energy ranges of individual energy channels are defined in a <PARAM> element named "EnergyRange". The boundary values of the channels are listed in the attribute named "value", e.g.:

  ```
  <PARAM name="EnergyRange" unit="eV" ucd="instr.param" datatype="float"
  arraysize="9" value="1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0"/>
  ```

  If there are N energy channels then the "value" list must contain N+1 elements. In the example above the channels are: Ch1 = 1.0-2.0, Ch2 = 2.0-3.0, ... , Ch8 = 8.0-9.0 .

- Position coordinates of each data point.
- 'Time' field is included if it defined in the input VOTable file (url_XYZ).
- The particle counts of all channels are defined in a field named 'ParticleFlux'. Type of this field is 'list of floats'. The unit of ParticleFlux is 'ions/(m^2*s*str*eV)' where str = steradians.

Example:

PHP client code:

```
 // Define the method file and create client

 $methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
 $client = new SoapClient($methods_file);

 // Define input parameters for getDataPointSpectra

 $params = array (
     'ResourceID'=>'spase://IMPEX/NumericalOutput/FMI/HYB/venus/
            run01_venus_nominal_spectra_20140417/H+_spectra',
     'url_XYZ' =>
         'http://impex-fp7.fmi.fi/ws_tests/input/getDataPointSpectra_input.vot',
     'extraParams' => array(
         'OutputFileType' => 'VOTable'
     )
);

 // Call the server and display the resulting url

 $data_url = $client->getDataPointSpectra($params);
 echo $data_url;
```

Input VOTable file (http:// impex-fp7.fmi.fi/ws_tests/input/getDataPointSpectra_input.vot) :

```
 <?xml version='1.0'?>
 <VOTABLE version="1.2"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
         http://www.ivoa.net/xml/VOTable/v1.2"
     xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
 <!--
  ! VOTable written by FMI web service getVOTableURL
  ! 2014-06-03T11:24:33+0000
  !-->
 <RESOURCE>
 <TABLE name="getDataPointSpectra test run" nrows="21">
<DESCRIPTION>
  getDataPointSpectra output format demo
</DESCRIPTION>
<FIELD ID="col_X" name="X" ucd="pos.cartesian.x" unit="m" datatype="float" />
<FIELD ID="col_Y" name="Y" ucd="pos.cartesian.y" unit="m" datatype="float" />
<FIELD ID="col_Z" name="Z" ucd="pos.cartesian.z" unit="m" datatype="float" />
<DATA>
<TABLEDATA>
       <TR>
              <TD>0</TD>
              <TD>6719900</TD>
              <TD>1000000</TD>
       </TR>
       <TR>
              <TD>0</TD>
              <TD>7636250</TD>
              <TD>1000000</TD>
       </TR>
       <TR>
              <TD>0</TD>
              <TD>8552600</TD>
              <TD>1000000</TD>
       </TR>
       <TR>
              <TD>0</TD>
              <TD>9468950</TD>
              <TD>1000000</TD>
       </TR>
       <TR>
              <TD>0</TD>
              <TD>10385300</TD>
              <TD>1000000</TD>
       </TR>
 .
 .
 </TABLEDATA>
 </DATA>
 </TABLE>
 </RESOURCE>
 </VOTABLE>
```

Output VOTable file (http:// impex-fp7.fmi.fi/ws_tests/output/getDataPointSpectra_output.vot) :

```
 <?xml version='1.0'?>
 <VOTABLE version="1.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
    http://www.ivoa.net/xml/VOTable/v1.2"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
 <!--
 ! VOTable written by FMI web service
 ! at 2014-06-03T15:11:11+0300
 !-->
<RESOURCE>
<TABLE name="Venus_H+_spectra" nrows="21">
<DESCRIPTION>
  SimulationModel             : FMI_HYB_SimulationModel
  SimulationModel_ResourceID : spase://IMPEX/SimulationModel/FMI/HYB
  SimulationRun_ResourceID   : spase://IMPEX/SimulationRun/FMI/HYB/venus/
                               run01_venus_nominal_spectra_20140417
  NumericalOutput_ResourceID : spase://IMPEX/NumericalOutput/FMI/HYB/venus/
                               run01_venus_nominal_spectra_20140417/H+_spectra
  Content description        : H+ energy spectra
  Object                     : Venus
  Object radius              : 6051800 m
</DESCRIPTION>
<PARAM name="EnergyRange" unit="eV" ucd="instr.param" datatype="float"
arraysize="97" value="10 10.86976144985797 11.81517139768183 12.84280945819864
13.95982751565989 15.17399949763863 16.49377547795966 17.92834048529384
19.48767842669749 21.18264157097451 . . . "/>
<FIELD ID="col1" datatype="float" name="x" ucd="pos.cartesian.x" unit="m" />
<FIELD ID="col2" datatype="float" name="y" ucd="pos.cartesian.y" unit="m" />
<FIELD ID="col3" datatype="float" name="z" ucd="pos.cartesian.z" unit="m" />
<FIELD ID="col4" datatype="float" name="ParticleFlux" ucd="phys.flux.density"
unit="m-2.s-1.sr-1.eV-1" arraysize="96"/>
<DATA>
<TABLEDATA>
       <TR>
              <TD>0</TD>
              <TD>6.7199e+06</TD>
              <TD>1e+06</TD>
              <TD>5.1768e+06 406133 375167 390967 392000 363500 383733 365033
344533 356800 … 0 0 0</TD>
       </TR>
       <TR>
              <TD>0</TD>
              <TD>7.63625e+06</TD>
              <TD>1e+06</TD>
              <TD>4.97207e+06 364933 366500 338100 380100 313067 332533 364933
335133 320800 … 0 0 0</TD>
       </TR>
       <TR>
              <TD>0</TD>
              <TD>8.5526e+06</TD>
              <TD>1e+06</TD>
              <TD>5.41133e+06 365133 391233 393533 412700 371967 342767 378533
355667 331200 … 0 0 0</TD>
       </TR>
       <TR>
              <TD>0</TD>
              <TD>9.46895e+06</TD>
              <TD>1e+06</TD>
              <TD>5.41287e+06 273233 308133 329933 327267 326867 364133 381233
343400 257033 … 0 0 0</TD>
       </TR>
       <TR>
              <TD>0</TD>
              <TD>1.03853e+07</TD>
              <TD>1e+06</TD>
              <TD>4.8574e+06 372000 401700 417100 408900 330900 334200 299933
320533 328267 … 0 0 0</TD>
       </TR>
.
.
.
</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

2.9 getDataPointSpectraSpacecraft

This method may be used to compute energy spectra of given particle species along spacecraft path for given simulation run.

Input parameters:

• **ResourceID** (String, Mandatory)

ResourceID of a <NumericalOutput> element in the tree.xml file. This NumericalOutput element defines the simulation run. The correspondence in the data model is: "Spase/NumericalOutput/ResourceID". NumericalOutput elements which contain spectral data have <SimulationProduct> element set as 'Spectra'. The NumericalOutput element contains the energy channels used in the simulation in the element Spase/NumericalOutput/Parameter/Particle/EnergyRange.

• **Spacecraft_name** (enumerated string, Mandatory)

A string indicating the name of the spacecraft as defined by AMDA. The following names are currently recognized: VEX, MEX, MGS, MAVEN, MESSENGER, CLUSTER1, CLUSTER2, CLUSTER3, CLUSTER4, GEOTAIL, IMP-8, POLAR.

• **StartTime** (dateTime, Mandatory)

Time of first point in the spacecraft path to be included. Time format as defined by ISO 8601 standard (e.g. "2010-05-22T12:50:00.000").

• **StopTime** (dateTime, Mandatory)

Time of last point in the spacecraft path to be included. Time format as defined by ISO 8601 standard (e.g. "2010-05-22T18:10:00.000").

• **Sampling** (duration, Mandatory)

Time interval between two consecutive points along the spacecraft path. Time format as defined by ISO 8601 standard (e.g. "PT60S").

• **extraParams** (Optional)

• InterpolationMethod (enumerated String, Optional)

Method used in interpolation. Possible values are:
Linear : Field value at given point is linearly interpolated from the field values at the corner points of the enclosing grid cube.
NearestGridPoint : Field value is taken as the value at the nearest grid point.

Default value: Linear

• OutputFileType (enumerated String, Optional)

Type of the output file. Currently only VOTable is supported..

Default value: VOTable

Output:

• **url_Param** (uri)

URL of the result data file containing time, 3D position coordinates of points and the particle counts in specified energy channels at these points. The output file (VOTable) contains the following information:

- The energy ranges of individual energy channels are defined in a <PARAM> element named "EnergyRange". The boundary values of the channels are listed in the attribute named "value", e.g.:

  ```
  <PARAM name="EnergyRange" unit="eV" ucd="instr.param" datatype="float"
  arraysize="9" value="1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0"/>
  ```

  If there are N energy channels then the "value" list must contain N+1 elements. In the example above the channels are: Ch1 = 1.0-2.0, Ch2 = 2.0-3.0, ... , Ch8 = 8.0-9.0 .

- Time and corresponding position coordinates of each data point.
- The particle counts of all channels are defined in a field named ‘ParticleFlux’. Type of this field is ‘list of floats’. . The unit of ParticleFlux is ‘ions/(m^2*s*str*eV)’ where str = steradians.

Example:

PHP client code:

```
 // Define the method file and create client

 $methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
 $client = new SoapClient($methods_file);

 // Input parameters for getDataPointSpectraSpacecraft

 $params = array (
        'ResourceID'=>'spase://IMPEX/NumericalOutput/FMI/HYB/venus/
              run01_venus_nominal_spectra_20140417/H+_spectra',
       'Spacecraft_name' => 'VEX',
       'StartTime'       => '2010-08-02T06:00:00.000',
       'StopTime'        => '2010-08-02T09:00:00.000',
       'Sampling'        => 'PT60S',
       'extraParams'     => array('OutputFileType' => 'VOTable')
);

 // Call the server and display the resulting url

 $data_url = $client->getDataPointSpectraSpacecraft($params);
 echo $data_url;
```

Output VOTable file
(http:// impex-fp7.fmi.fi/ws_tests/output/getDataPointSpectraSpacecraft_output.vot) :

```
 <?xml version='1.0'?>
 <VOTABLE version="1.2"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.ivoa.net/xml/VOTable/v1.2
    http://www.ivoa.net/xml/VOTable/v1.2"
    xmlns="http://www.ivoa.net/xml/VOTable/v1.2">
<!--
 ! VOTable written by FMI web service
 ! at 2014-06-03T16:23:55+0300
 !-->
<RESOURCE>
<TABLE name="Venus_H+_spectra" nrows="180">
<DESCRIPTION>
  SimulationModel             : FMI_HYB_SimulationModel
  SimulationModel_ResourceID : spase://IMPEX/SimulationModel/FMI/HYB
  SimulationRun_ResourceID   : spase://IMPEX/SimulationRun/FMI/HYB/venus/
                               run01_venus_nominal_spectra_20140417
  NumericalOutput_ResourceID : spase://IMPEX/NumericalOutput/FMI/HYB/venus/
                               run01_venus_nominal_spectra_20140417/H+_spectra
  Content description        : H+ energy spectra
  Object                     : Venus
  Object radius              : 6051800 m
  Spacecraft                 : VEX
  Start time                 : 2010-08-02T06:00:00.000
  Stop time                  : 2010-08-02T09:00:00.000
  Sampling                   : PT60S
</DESCRIPTION>
<PARAM name="EnergyRange" unit="eV" ucd="instr.param" datatype="float"
arraysize="97" value="10 10.86976144985797 11.81517139768183 12.84280945819864
13.95982751565989 15.17399949763863 16.49377547795966 17.92834048529384
19.48767842669749 21.18264157097451 …"/>
<FIELD ID="col1" arraysize="*" datatype="char" name="Time" ucd="time.epoch"
xtype="dateTime"/>
<FIELD ID="col2" datatype="float" name="x" ucd="pos.cartesian.x" unit="m" />
<FIELD ID="col3" datatype="float" name="y" ucd="pos.cartesian.y" unit="m" />
<FIELD ID="col4" datatype="float" name="z" ucd="pos.cartesian.z" unit="m" />
<FIELD ID="col5" datatype="float" name="ParticleFlux" ucd="phys.flux.density"
unit="m-2.s-1.sr-1.eV-1" arraysize="96"/>
<DATA>
<TABLEDATA>
      <TR>
            <TD>2010-08-02T06:00:30.000Z</TD>
            <TD>1.56478e+07</TD>
            <TD>9.18185e+06</TD>
            <TD>-1.90592e+07</TD>
            <TD>897233 66133.3 67766.7 75766.7 77600 59866.7 … 0 0 0</TD>
      </TR>
      <TR>
            <TD>2010-08-02T06:01:30.000Z</TD>
            <TD>1.56141e+07</TD>
            <TD>9.16817e+06</TD>
            <TD>-1.88244e+07</TD>
            <TD>897233 66133.3 67766.7 75766.7 77600 59866.7 … 0 0 0</TD>
      </TR>
      <TR>
            <TD>2010-08-02T06:02:30.000Z</TD>
            <TD>1.55787e+07</TD>
            <TD>9.15365e+06</TD>
            <TD>-1.85839e+07</TD>
            <TD>976733 73466.7 77700 80000 77700 66000 68833 … 0 0 0</TD>
      </TR>

.
```

```
.
.
</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>
```

2.10 isAlive

This method may be used to check the web server status. It returns Boolean true if the web services are available. If the server is not running a SOAP error message is returned.

Input parameters: None

Output: Boolean true

Example:

PHP client code:

```
// Define the method file and create client

$methods_file = "http://impex-fp7.fmi.fi/ws/Methods_FMI.wsdl";
$client = new SoapClient($methods_file);

// Call the server and display the status

$isAlive = false;

try {
      $isAlive = $client->isAlive();
}
catch (Exception $e) {
      echo "Errors : <br />";
      echo $e->getMessage();
}

if ($isAlive)
      echo "We are alive ! " . "\n";
else
      echo "We are dead ? " . "\n";
```