Interface Control Document

List of Webservices provided by LATMOS and its Simulation database LatHyS

Title:*	List of webservices to be provided by the LATMOS
Date:*	26 April 2012
Editor:*	R. Modolo
Contributors:*	S. Hess
Distribution:*	*project*
Level:*	*low level*
Roles:*	Data producer / Webservice Developper
User Requirement:	
Related Workpackage(s):	WP2, WP3

Version History

Version	Date	Released by	Detail
0.0	26/04/2012	S. Hess	draft
0.1	28/05/2012	R. Modolo	Comments
0.2	13/12/2012	R. Modolo	Update of webservices and detail description of method (getdatapointvalue)
0.3	21/01/2013	R. Modolo	Harmonization with FMI for getDataPointValue, update of getDataPointValue LATMOS_Spacecraft
0.4	07/02/2013	R. Modolo	Update of methods with IMPEx harmonization
0.5	15/03/2014	R. Modolo	Adding webservices example and update
0.6	14/12/2014	R. Modolo	Adding getDataPointSpectra and getDataPoitnSpectraSpacecraft
1.0	12/05/2015	R. Modolo	Update option getDataPointSpectra and getDataPointSpectraSpacecraft

Scope of this document

The aim of the present document is to provide the list of webservices implemented by the LATMOS for its Simulation DataBase (LatHyS). The function of each services is detailed, as well as the inputs they need and the outputs they provide.

1. Philosophy proposed by LATMOS (April, 2012)

It has been decided at the Graz ADD Meeting that SMDB would be accessible through a single XML file (tree.xml) which (1) describes all the simulation runs and the static simulations results and (2) points toward the methods.xml defining the methods.

There is however a discrepancy between data providers who have pre-computed runs and those who make runs-on-demand.

For those with run on demand, it seems that the request will not be fully automated (there will be human intervention) so that it will not be a generic web service, and thus will not be covered by the methods.xml file.

For those who have precomputed runs (LATMOS case), users (in IMPEX case: tools) already know the list of available Objects, Models and Runs just from the tree.xml file. The list of runs (with their respective inputs) can be defined in the tree.xml file even if there are no static outputs.

In that case, it does not seem interesting to implement methods such as **'getAvailableObjects'**, **'getAvailableRuns'**, **'getRunDescription'**, **'getAvailableModels'** which were defined in the first ADD draft, because **these informations can be obtained by parsing tree.xml.**

The **'getAvailableProduct'** (which should return 2dCuts,Timeseries,...) should also be avoided since the answer to **this request would be the methods.xml file itself**.

The method '**getAvailableParameters**' seems also **not required**. All parameters (ion species density, temperature, magnetic field, electric field,…) which are available on pre-computed data-product, and therefore listed and described in the tree.xml file, will be available as potential input parameters for the different methods.

Methods which will not be developed at LATMOS according to the ADD are:

- **getAvailableObjects()**
- **getAvailableModels(ObjectID)**
- **getAvailableRuns(ObjectID,ModelID)**
- **getRunDescription(RunID)**
- **getAvailableProducts(RunID)**
- **getAvailableParameters(RunID,ProductID)**

However in order to simplify the "tree.xml" and do not mention all the granules timeseries for a given spacecraft (precomputed data), will be the result of a webservice – getFileURL (more details at 2.4).

The Methods WSDL file is available at
http://impex.latmos.ipsl.fr/Methods_LATMOS.wsdl

The tree XML file is available at
http://impex.latmos.ipsl.fr/tree.xml

2. New List of WebServices from LATMOS (January/February 2013)

Harmonization of a common set of web-services has been investigated; however several inputs parameters are optional for some databases and others which won't be included. Therefore all optional parameters which are not common to all databases providing this webservice are identified as "additional parameters". These "additional parameters" are databases dependent.

The webservices developed by LATMOS are:

Name of the webservice	Functionnality	Delivered
getFileURL	Methods for pre-computed data. This methods indicates the granule of the tree.xml which is now removed(to save space).	19/04/2013
getDataPointValue	This is a generic method which can be used to return parameters for 0D (a given point), 1D (along a curve/trajectory), 2D (in a plane) or 3D (inside a volume).	19/04/2013
getDataPointValueSpacecraft	This Method intepolates the physical simulation parameters along a given spacecraft and for a given time interval	19/04/2013
getSurface	This method is called from 3DView. It is required to generate a meshgrid and compute interpolation for one or several parameters.	19/04/2013
getFieldLine	This method returns a magnetic field line/stream lines/etc. lines	19/12/2013
getDataPointSpectra	This Method returns interpolated ion spectra for a given point in space (0D) or in 1D (along a curve/trajectory) for a given ion species and a given simulation	-
getDataPointSpectraSpacecraft	This Methods generates ion spectrogram along a given spacecraft orbit for a given time interval	-

The details of the different webservices are presented below.

2.1 Method : getDataPointValue (update of D3.10)

The starting point of this method is described in the delivery report D3.10.

This is a generic method which can be used to return parameters for 0D (a given point), 1D (along a curve/trajectory), 2D (in a plane) or 3D (inside a volume). Following D3.10 document, the set of coordinates on which the interpolation is performed is provided as a result of an other webservice (input for our method).

3D output is authorized. Asking 3D data is not efficient since 3D full cube is already pre-computed and provided in the tree.xml.

Inputs :

- **ResourceID: (Mandatory)** A **string** describing which dataset is requested (consistent with the tree.xml file). It provides a unique ID. It replaces the RUN_ID from the D3.10 document. The ResourceID of the NumericalOutput is accessible as easily as the RunID in the tree.xml and provide a much more precise information. **ResourceID** points to the 3D Cubes files in which the interpolation is to be made. The correspondence in the DataModel is "Spase/NumericalOutput/ResourceID"

- **Variable**: **(Optional)** A **string** describing the list of individual parameters from the selected NumericalOutput separated by a comma. Each parameters are identified by the ParameterKey. This allows setting a precise definition for **Variable**. In the D3.10 document, the variable_STR values were not defined, except for simple ones, but did not address the difficulties encountered for the species, which have different names depending on runs and databases. Now, it is made simple as the couple (**ResourceID+ ParameterKey**) clearly defines the output we want to interpolate, in a unique manner. By default, all parameters inside the Numerical Ouput are sent back. The correspondence in the DataModel is:

"Spase/NumericalOutput/ResourceID/Parameter/ParameterKey"

- **url_XYZ: (Mandatory)** URL of the VoTable file containing the 3D coordinates. Suggested field name according to Spase/SimulationRun/SimulationDomain/CoordinatesLabel. After receiving getDataPointValue request the webservice reads the file from the url.

The **coordinate of the points** must be expressed in the **coordinate system defined** by Spase/SimulationRun/SimulationDomain/CoordinateSystem **and with Space Coordinate label names.** Databases (or at least the LATMOS database), will not provide reference frame converters, as this is a wide job which usually is accomplished by tools (using SPICE,...) and not by databases. Time is optional in the Input VOTable with a FIELD ucd="Time"

- **additional_params:** parameter or set or parameter which are databases dependent.

With **additional_params** value for LATMOS :

- **IMFClockAngle: (optional)** a real corresponding to the angle of rotation of a 1D points of coordinate. The rotation will be done around the symmetry axis (typically x-axis). By default, no rotation will be operated.

- **OutputFileType**: **(optional)** the output file format could be selected by the user. According to the ADD document the authorized output format are NetCDF and VOTable.

Outputs:

- **url_Param** : the output of this service will generate an temporary url of a file containing the 3D coordinates and the interpolated parameter values of the requested **Variables**.
  (Units : physical unit of quantity, as defined and described in the datamodel and tree.xml.)

Example :

```
    $client = new SoapClient("http://impex.latmos.ipsl.fr/Methods_LATMOS.wsdl",
array('trace' => 1));

    $param=new IMPExWSClass();
    $param->ResourceID='impex://LATMOS/Hybrid/Mars_13_02_13/Mag/3D';
    $param->url_XYZ='http://impex.latmos.ipsl.fr/Vmvrv5e.xml';


    $extBases = $client->getDataPointValue($param);

    echo $client->__getLastResponse();
class IMPExWSClass{
 public $Variable="Bx,By,Btot";
 public
$extraParams=array('IMFClockAngle'=>120,'OutputFileType'=>'VOTable','DebugShowCm
d'=>1);
}
```

Example of Output file :

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE version='1.2' xmlns='http://www.ivoa.net/xml/VOTable/v1.2'>
<RESOURCE name='LatHySData'>
<TABLE name='Data'>
</DESCRIPTION>
<GROUP ID="PosFrame" ref="MSO">
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C1" ref="col2"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C2" ref="col3"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C3" ref="col4"/>
</GROUP>
<GROUP ID="FieldFrame" ref="MSO">
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C1" ref="col5"/>
```

<FIELDref utype="stc:AstroCoords.Position3D.Value3.C2" ref="col6"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C3" ref="col7"/>
</GROUP>
<FIELD name="Time" ID="col1" ucd="time.epoch" xtype="dateTime" utype="" datatype="char" arraysize="*" />
<FIELD name="X" ID="X" ucd="pos.cartesian.x" unit="Rm" utype="stc:AstroCoords.Position3D.Value3.C1" datatype="float"/>
<FIELD name="Y" ID="Y" ucd="pos.cartesian.y" unit="Rm" utype="stc:AstroCoords.Position3D.Value3.C2" datatype="float"/>
<FIELD name="Z" ID="Z" ucd="pos.cartesian.z" unit="Rm" utype="stc:AstroCoords.Position3D.Value3.C3" datatype="float"/>
<FIELD name="Bx" ID="col5" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<FIELD name="By" ID="col6" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<FIELD name="Btot" ID="col7" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<DATA>
<TABLEDATA>
<TR>
<TD> 2007-07-12T14:00:30 </TD><TD> -1.99981 </TD><TD> 0.175831 </TD><TD> -0.642188 </TD><TD> 6.744 </TD><TD> -0.140 </TD><TD> 8.413 </TD>
</TR>
<TR>
<TD> 2007-07-12T14:01:30 </TD><TD> -1.97182 </TD><TD> 0.171635 </TD><TD> -0.664246 </TD><TD> 6.744 </TD><TD> -0.140 </TD><TD> 8.596 </TD>
</TR>

2.2 Method: getDataPointValueSpacecraft

This Method returns interpolated physical simulation parameters along a given spacecraft. This method returns a VoTable with the different url for each requested parameter.

Inputs :

ResourceID: **(Mandatory)** A string describing which dataset is requested (consistent with the tree.xml file). It provides a unique ID. It replaces the RUN_ID from the D3.10 document. The ResourceID of the NumericalOutput is accessible as easily as the RunID in the tree.xml and provide a much more precise information. ResourceID points to the 3D Cubes files in which the interpolation is to be made. The correspondence in the DataModel is “Spase/NumericalOutput/ResourceID”.

- **Variable**: **(Optional)** A **string** describing the list of individual parameters from the selected NumericalOutput separated by a comma. Each parameters are identified by the ParameterKey. This allows setting a precise definition for **Variable**. In the D3.10 document, the variable_STR values were not defined, except for simple ones, but did not address the difficulties encountered for the species, which have different names depending on runs and databases. Now, it is made simple as the couple (**ResourceID+ ParameterKey**) clearly defines the output we want to interpolate, in a unique manner. By default, all parameters inside the Numerical Ouput are sent back. The correspondence in the DataModel is:

“Spase/NumericalOutput/ResourceID/Parameter/ParameterKey”

- **SpacecraftName**: **(Mandatory)** a string indicating the name of the spacecraft as defined by AMDA and 3Dview.

- **StartTime: (Mandatory)** Time format as defined by ISO 8601 standard.

- **StopTime: (Mandatory)** Time format as defined by ISO 8601 standard.

- **Sampling: (Mandatory)** Time format as defined by ISO 8601 standard.

- **additional_params:** parameter or set or parameter which are databases dependent.

With **additional_params** value for LATMOS :

- IMFClockAngle: (optional) a real corresponding to the angle of rotation of a 1D points of coordinate. The rotation will be done around the symmetry axis (typically x-axis). By default, no rotation will be operated.

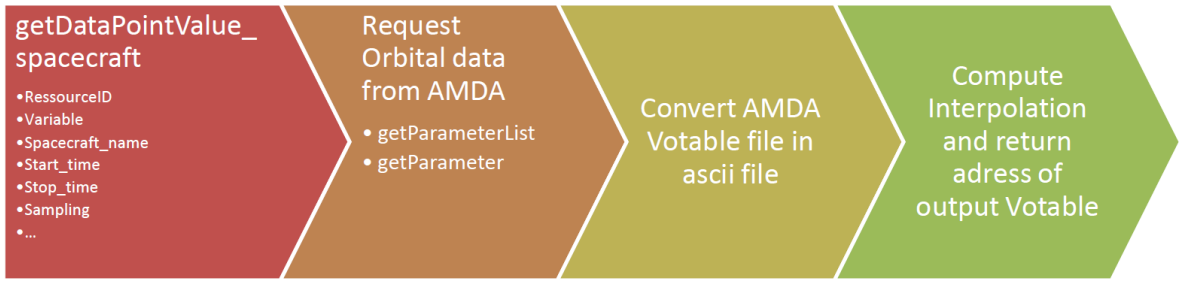
- OutputFileType: **(optional)** the output file format could be selected by the user. According to the ADD document the authorized output format are NetCDF and VOTable.

-

Outputs:

- **url_param: (Mandatory)** the output of the service generates a temporary url of a VoTable containing requested time, 3D coordinates and interpolated requested parameters.

The workflow is the following :



Example :

```
$client = new SoapClient("http://impex.latmos.ipsl.fr/Methods_LATMOS.wsdl",
array('trace' => 1));

$param=new IMPExWSClass();
$param->ResourceID='impex://LATMOS/Hybrid/Mars_13_02_13/The/3D';

$param->StartTime='2007-07-12T00:00:00.000Z';
$param->StopTime='2007-07-13T00:00:00.000';
$param->Spacecraft_name='MEX';
```

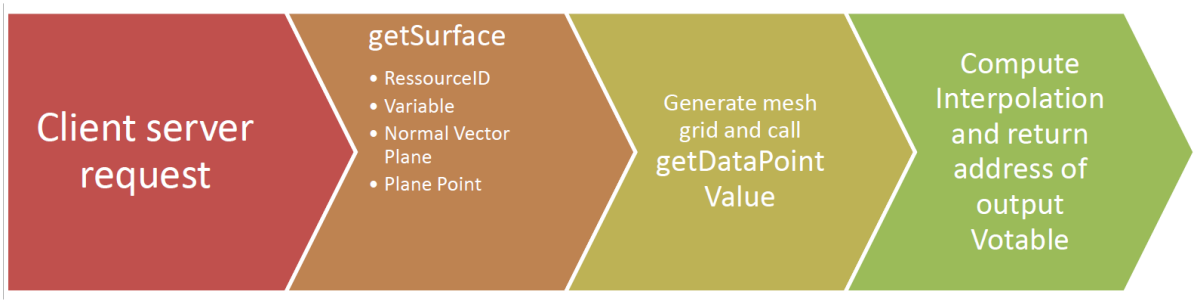
```
    $param->Sampling="60";

    $extBases = $client->getDataPointValue_Spacecraft($param);

    echo $client->__getLastResponse();

class IMPExWSClass{
 public $Variable="Ux";
 public
$extraParams=array('IMFClockAngle'=>120,'OutputFileType'=>'VOTable','DebugShowCmd'=>1);
}
```

Example of output file :

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE version='1.2' xmlns='http://www.ivoa.net/xml/VOTable/v1.2'>
<RESOURCE name='LatHySData'>
<TABLE name='Data'>
</DESCRIPTION>
<GROUP ID="PosFrame" ref="MSO">
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C1" ref="col2"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C2" ref="col3"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C3" ref="col4"/>
</GROUP>
<GROUP ID="FieldFrame" ref="MSO">
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C1" ref="col5"/>
</GROUP><FIELD name="Time" ID="col1" ucd="time.epoch" xtype="dateTime" utype="" datatype="char" arraysize="*" />
<FIELD name="X" ID="X" ucd="pos.cartesian.x" utype="stc:AstroCoords.Position3D.Value3.C1" datatype="float"/>
<FIELD name="Y" ID="Y" ucd="pos.cartesian.y" utype="stc:AstroCoords.Position3D.Value3.C2" datatype="float"/>
<FIELD name="Z" ID="Z" ucd="pos.cartesian.z" utype="stc:AstroCoords.Position3D.Value3.C3" datatype="float"/>
<FIELD name="Ux" ID="col5" ucd="phys.veloc" utype="" datatype="float" unit="km.s-1" />
<DATA>
<TABLEDATA>
<TR>
<TD> 2007-07-12T00:00:30.000 </TD><TD> -2.83433 </TD><TD> 0.302013 </TD><TD> 0.381733 </TD><TD> 0.000 </TD>
</TR>
<TR>
<TD> 2007-07-12T00:01:30.000 </TD><TD> -2.81906 </TD><TD> 0.298793 </TD><TD> 0.351162 </TD><TD> 0.000 </TD>
</TR>
```

2.3 Method: getSurface

This method is called from 3DView. It is required to generate a meshgrid and compute interpolation for one or several parameters. It is therefore linked to getDataPointValue, as the following sketch describes.



Inputs :

ResourceID: (Mandatory) A string describing which dataset is requested (consistent with the tree.xml file). It provides a unique ID. It replaces the RUN_ID from the D3.10 document. The ResourceID of the NumericalOutput is accessible as easily as the RunID in the tree.xml and provide a much more precise information. ResourceID points to the 3D Cubes files in which the interpolation is to be made. The correspondence in the DataModel is "Spase/NumericalOutput/ResourceID".

- **Variable**: **(Optional)** A **string** describing the list of individual parameters from the selected NumericalOutput separated by a comma. Each parameters are identified by the ParameterKey. This allows setting a precise definition for **Variable**. In the D3.10 document, the variable_STR values were not defined, except for simple ones, but did not address the difficulties encountered for the species, which have different names depending on runs and databases. Now, it is made simple as the couple (**ResourceID+ ParameterKey**) clearly defines the output we want to interpolate, in a unique manner. By default, all parameters inside the Numerical Ouput are sent back. The correspondence in the DataModel is:

"Spase/NumericalOutput/ResourceID/Parameter/ParameterKey"

- **NormalVector Plane: (Mandatory)** 3D components of a normalized vector normal to the requested plane. The type of the variable is that of "Spase/NumericalOutput/SpatialDescription/PlaneNormalVector"

- **PlanePoint : (Mandatory)** A 3D coordinate Indicating the position of one point in the plane requested. **Coordinate of the point** must be expressed in the **coordinate system defined** by:
- Spase/SimulationRun/SimulationDomain/CoordinateSystem **and with Space Coordinate label names.** The type of the variable is that of "Spase/NumericalOutput/SpatialDescription/PlanePoint"

- **additional_params:** parameter or set or parameter which are databases dependent.

With **additional_params** value for LATMOS :

- **Resolution: (Optional)** A **real** that specified the spatial resolution of the grid mesh.

- **IMFClockAngle: (optional)** a real corresponding to the angle of rotation of a 1D points of coordinate. The rotation will be done around the symmetry axis (typically x-axis). By default, no rotation will be operated.

- **OutputFileType**: **(optional)** the output file format could be selected by the user. According to the ADD document the authorized output format are NetCDF and VOTable.

Outputs :

- **Url_grid: (Mandatory)** A URL linked to a file (VoTable/netCDF) with the coordinate of each points in a 2D plane on which interpolated parameters are provided.

Example :

```
    $client = new SoapClient("http://impex.latmos.ipsl.fr/Methods_LATMOS.wsdl",
array('trace' => 1));

    $param=new IMPExWSClass();
    $param->ResourceID='impex://LATMOS/Hybrid/Mars_13_02_13/Mag/3D';

    $param->PlanePoint='0 0 0';
    $param->PlaneNormalVector='0 0 1';

    $extBases = $client->getSurface($param);

    echo $client->__getLastResponse();

class IMPExWSClass{
 public $Variable="Bx,By,Bz";
 public
$extraParams=array('IMFClockAngle'=>0,'OutputFileType'=>'VOTable','DebugShowCmd'=
>1);
}
```

Example of output file :

```
<?xml version="1.0" encoding="UTF-8"?>
<VOTABLE version='1.2' xmlns='http://www.ivoa.net/xml/VOTable/v1.2'>
<RESOURCE name='LatHySData'>
<TABLE name='Data'>
```

```
<DESCRIPTION />
<GROUP ID="PosFrame" ref="MSO">
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C1" ref="col1"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C2" ref="col2"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C3" ref="col3"/>
</GROUP>
<GROUP ID="FieldFrame" ref="MSO">
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C1" ref="col4"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C2" ref="col5"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C3" ref="col6"/>
</GROUP>
<FIELD name="X" ID="X" ucd="pos.cartesian.x" utype="stc:AstroCoords.Position3D.Value3.C1"
datatype="float"/>
<FIELD name="Y" ID="Y" ucd="pos.cartesian.y" utype="stc:AstroCoords.Position3D.Value3.C2"
datatype="float"/>
<FIELD name="Z" ID="Z" ucd="pos.cartesian.z" utype="stc:AstroCoords.Position3D.Value3.C3"
datatype="float"/>
<FIELD name="Bx" ID="col4" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<FIELD name="By" ID="col5" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<FIELD name="Bz" ID="col6" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<DATA>
<TABLEDATA>
<TR>
<TD> -8069.904 </TD><TD> -15971.685 </TD><TD> 0 </TD><TD> -1.668 </TD><TD> 2.668
</TD><TD> 0.027 </TD>
</TR>
```

2.4 Method: getFileURL

This method fetches the precomputed data (already calculated values along spacecraft trajectory).

After discussion with CDPP it seems to be more convenient to return a VoTable file with several URL corresponding to all simulated parameters. The selection of parameter (and Spacecraft) will be done by the tools (AMDA and 3Dview).

- **RunID : (Mandatory)** A **string** defining the unique Id of the simulation. It allows selecting a given simulation. The different parameters are not selected. The results of this methods returns VoTable within URL information for all simulated parameters.

- **StartTime: (Mandatory)** Time format as defined by ISO 8601 standard.

- **StopTime: (Mandatory)** Time format as defined by ISO 8601 standard.

Outputs:

VoTable: (Mandatory) the output of the service generates a a VoTable containing a list of granule of Spacecraft matching the Start and Stop time request and for all simulation parameters.

AMDA and 3Dview have more information than required but it will help minimizing the number of calls for this webservice. Since LATMOS have all precomputed interpolated data, the webservice will not last too long (interpolation for all parameters is already made we just return the URL location).

Example :

```
    $client = new SoapClient("http://impex.latmos.ipsl.fr/Methods_LATMOS.wsdl",
array('trace' => 1));

    $param=new IMPExWSClass();
    $param->ResourceID='impex://LATMOS/Hybrid/Mars_14_01_13/Mag/MEX/0.0';

    $param->StartTime='2007-07-12T00:00:00.000Z';
    $param->StopTime='2007-07-13T00:00:00.000';
    $extBases = $client->getFileURL($param);

    echo $client->__getLastResponse();

class IMPExWSClass{
 public $Variable="Btot,Bx";
 public
$extraParams=array('IMFClockAngle'=>0,'OutputFileType'=>'VOTable','DebugShowCmd'=
>1);
}
```

Example of Output description

```
<VOTABLE version="1.2">
<RESOURCE name="LatHySGranules">
<TABLE name="FileURLs">
<DESCRIPTION/>
<FIELD ID="ResourceID" datatype="char" name="ResourceID" ucd="meta.id" arraysize="*"/>
<FIELD ID="StartTime" datatype="char" xtype="dateTime" name="StartTime" ucd="time.epoch;time.start"
        arraysize="*"/>
<FIELD ID="StopTime" datatype="char" xtype="dateTime" name="StopTime" ucd="time.epoch;time.stop"
        arraysize="*"/>
<FIELD ID="URL" datatype="char" name="URL" ucd="meta.ref.url" arraysize="*"/>
<FIELD ID="ParentID" datatype="char" name="ParentID" ucd="meta.id.parent" arraysize="*"/>
<DATA>
<TABLEDATA>
<TR><TD>impex://LATMOS/Hybrid/Mars_14_01_13/Mag/MEX/0.0/Mag_14_01_13_MEX_ephem_2007_0
        7_0.0</TD>
<TD>2007-07-01T00:00:49.000</TD><TD>2007-07-31T23:57:08.000</TD>
<TD>http://impex.latmos.ipsl.fr/Hybrid/Mars_14_01_13/Mag_14_01_13_MEX_ephem_2007_07_0.0.xml
        </TD>
<TD>impex://LATMOS/Hybrid/Mars_14_01_13/Mag/MEX/0.0</TD>
</TR>
</TABLEDATA>
</DATA>
```

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

2.5 Method: getFieldLine

This would provide streamlines, or field lines inside the simulation box. It will be computed by following field lines equally spaced in a plane defined by the user. The inputs are those of 2dCut (to define the plane) + the spacing between the field lines.

The tool defines a series of starting points (close to what is asked for getDataPointValue).

Inputs :

- **ResourceID: (Mandatory)** A **string** describing which dataset is requested (consistent with the tree.xml file). It provides a unique ID. It replaces the RUN_ID from the D3.10 document. The ResourceID of the NumericalOutput is accessible as easily as the RunID in the tree.xml and provide a much more precise information. **ResourceID** points to the 3D Cubes files in which the interpolation is to be made. The correspondence in the DataModel is "Spase/NumericalOutput/ResourceID"

- **Variable**: **(Optional)** A **string** describing the list of individual parameters from the selected NumericalOutput separated by a comma. Each parameters are identified by the ParameterKey. This allows setting a precise definition for **Variable**. In the D3.10 document, the variable_STR values were not defined, except for simple ones, but did not address the difficulties encountered for the species, which have different names depending on runs and databases. Now, it is made simple as the couple (**ResourceID+ ParameterKey**) clearly defines the output we want to interpolate, in a unique manner. By default, all parameters inside the Numerical Ouput are sent back. The correspondence in the DataModel is:

"Spase/NumericalOutput/ResourceID/Parameter/ParameterKey"

- **url_XYZ: (Mandatory)** URL of the VoTable file containing the 3D coordinates. Suggested field name according to Spase/SimulationRun/SimulationDomain/CoordinatesLabel.

- **additional_params:** parameter or set or parameter which are databases dependent.

With **additional_params** value for LATMOS :

- **Direction: (optional)** Options: 'forward' , 'backward','both'
-

- **Stepsize: (optional)** the spatial step integration for the field line computing (by default half of the simulation spatial resolution).

- **OutputFileType**: **(optional)** the output file format could be selected by the user. According to the ADD document the authorized output format are NetCDF and VOTable.

Outputs:

url_Param : the output of this service will generate an temporary url of a file containing requested field lines

Example :

```
$client = new SoapClient("http://impex.latmos.ipsl.fr/Methods_LATMOS.wsdl",
  array('trace' => 1));

    $param=new IMPExWSClass();
    $param->ResourceID='impex://LATMOS/Hybrid/Mars_13_02_13/Mag/3D';
    $param->url_XYZ='http://impex.latmos.ipsl.fr/Vmvrv5e.xml';


    $extBases = $client->getFieldLine($param);

    echo $client->__getLastResponse();
class IMPExWSClass{
   public $Variable="Bx,By,Bz,Btot";
   public
  $extraParams=array('IMFClockAngle'=>0,'OutputFileType'=>'VOTable','DebugShowC
  md'=>1);
  }
```

Example of Output file

```
<?xml version="1.0"?>
<VOTABLE version="1.2" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.ivoa.net/xml/VOTable/v1.2" xmlns:stc="http://www.ivoa.net/xml/STC/v1.30" >
<RESOURCE name="IMPEx Field/Flow line">
<TABLE name="Lines">
<DESCRIPTION> File containing the position of each field/flow lines.
</DESCRIPTION>
<GROUP ID="PosFrame" ref="MSO">
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C1" ref="col2"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C2" ref="col3"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C3" ref="col4"/>
</GROUP>
<GROUP ID="FieldFrame" ref="MSO">
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C1" ref="col5"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C2" ref="col6"/>
<FIELDref utype="stc:AstroCoords.Position3D.Value3.C3" ref="col7"/>
</GROUP>
```

<FIELD name="Line" ID="col1" ucd="meta.number" utype="" datatype="int" width="3" />
<FIELD name="X" ID="col2" ucd="pos.cartesian.x" utype="stc:AstroCoords.Position3D.Value3.C1" datatype="float" unit="km" />
<FIELD name="Y" ID="col3" ucd="pos.cartesian.y" utype="stc:AstroCoords.Position3D.Value3.C2" datatype="float" unit="km" />
<FIELD name="Z" ID="col3" ucd="pos.cartesian.z" utype="stc:AstroCoords.Position3D.Value3.C3" datatype="float" unit="km" />
<FIELD name="Bx" ID="col5" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<FIELD name="By" ID="col6" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<FIELD name="Bz" ID="col7" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<FIELD name="Btot" ID="col8" ucd="phys.magField" utype="" datatype="float" unit="nT" />
<DATA>
<TABLEDATA>
<TR>
<TD> 1 </TD><TD> -6778.355 </TD><TD> 595.979 </TD><TD> -2176.696 </TD><TD> -6.665 </TD><TD> -3.975 </TD><TD> -0.801 </TD><TD> 7.802 </TD>
</TR>
<TR>
<TD> 1 </TD><TD> -6888.837 </TD><TD> 530.083 </TD><TD> -2189.981 </TD><TD> -6.581 </TD><TD> -4.056 </TD><TD> -0.636 </TD><TD> 7.757 </TD>
</TR>

2.6 Method: getDataPointsSpectra

This method provides time series Ion spectra along a spacecraft trajectory

Inputs :

- **ResourceID: (Mandatory)** A **string** describing which dataset is requested (consistent with the tree.xml file). It provides a unique ID.

If xpath='Spase/Simulation/ResourceID' then the method applies to all species.
If xpath='Spase/NumericalOutput/ResourceID' then the method applies to the species identified in the NumericalOutput (same as getDataPointValue)

- **Variable**: **(Optional)** A **string** describing the list of populations from the selected NumericalOutput separated by a comma. Each parameters are identified by the PopulationID. This allows setting a precise definition for **Variable**. In the D3.10 document, the variable_STR values were not defined, except for simple ones, but did not address the difficulties encountered for the species, which have different names depending on runs and databases. Now, it is made simple as the couple (**ResourceID+ PopulationID**) clearly defines the output we want to interpolate, in a unique manner. By default, all parameters inside the Numerical Ouput are sent back. The correspondence in the DataModel is:

"Spase/NumericalOutput/ResourceID/Parameter/Particle/PopulationID"

- **url_XYZ: (Mandatory)** URL of the VoTable file containing the 3D coordinates. Suggested field name according to Spase/SimulationRun/SimulationDomain/CoordinatesLabel. After receiving getDataPointValue request the webservice reads the file from the url. Same comments than 3.1. Time is optional in the Input VOTable with a FIELD ucd="Time"

- **additional_params:** parameter or set or parameter which are databases dependent.

With **additional_params** value for LATMOS :

- **IMFClockAngle: (optional)** a real corresponding to the angle of rotation of a 1D points of coordinate. The rotation will be done around the symmetry axis (typically x-axis). By default, no rotation will be operated.

- **OutputFileType**: **(optional)** the output file format could be selected by the user. According to the ADD document the authorized output format are NetCDF and VOTable.

- **EnergyChannel : (optional)** A list of Energy channel number (Ch0, Ch1, Ch2,…) separated by commas. The Energy of each energy channel has to be defined. Maybe use "Spase/NumericalOutput/Parameter/Structure/Element/ParameterKey".

Outputs:

- **url_Param** : the output of this service will generate an temporary url of a file containing requested time, 3D coordinates and the ion spectra.

Example:

```
 $client = new SoapClient("http://impex.latmos.ipsl.fr/Methods_LATMOS.wsdl",
array('trace' => 1));

$param=new IMPExWSClass();
$param->ResourceID=
'spase://IMPEX/NumericalOutput/LATMOS/Hybrid/Mars_14_03_14/IonSpectra';
$param->url_XYZ='http://impex.latmos.ipsl.fr/Vmrmf2.xml';
$param->EnergyChannel=' '

$extBases = $client->getDataPointSpectra($param);
echo $client->__getLastResponse();

class IMPExWSClass{

public
      $extraParams=array('IMFClockAngle'=>120,'OutputFileType'=>'VOTable','DebugSho
      wCmd'=>1);}
```

Example of Output 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/v1.1 http://www.ivoa.net/xml/VOTable/v1.2' xmlns='http://www.ivoa.net/xml/VOTable/v1.2'>
<RESOURCE>
<DESCRIPTION>DD1ajgsm_mex_xyz</DESCRIPTION>
<TABLE name='DD1ajgsm_mex_xyz'>
<FIELD datatype='float' ID='X' name='X' unit='Rm' ucd='pos.cartesian.x' utype='stc:AstroCoords.Position3D.Value3.C1'>
<VALUES null='NaN'/>
</FIELD>
<FIELD datatype='float' ID='Y' name='Y' unit='Rm' ucd='pos.cartesian.y' utype='stc:AstroCoords.Position3D.Value3.C2'>
<VALUES null='NaN'/>
</FIELD>
<FIELD datatype='float' ID='Z' name='Z' unit='Rm' ucd='pos.cartesian.z' utype='stc:AstroCoords.Position3D.Value3.C3'>
<VALUES null='NaN'/>
</FIELD>
<FIELD name="ParticleFlux" ID="col4" ucd="phys.flux" utype="" datatype="float" unit="Ions.m-2.s-1.str-1.eV-1" arraysize="32" />
<PARAM name="EnergyRange" unit="eV" ucd="instr.param" datatype="float" arraysize="32" value=" 0., 10.0, 13.0, 16.9, 22.0, 28.6, 37.1, 48.3, 62.7, 81.6, 106.0, 137.9, 179.2, 233.0, 302.9, 393.7, 511.9, 665.4, 865.0, 1124.6, 1461.9, 1900.5, 2470.6, 3211.8, 4175.4, 5428.0, 7056.4, 9173.3, 11925.3, 15502.9, 20153.8, 26199.9, 34059.9"/>
<DATA>
<TABLEDATA>
<TR>
<TD> 2010-01-01T17:00:24.000Z </TD><TD> 2.00288 </TD><TD> 2.813075784 </TD><TD> 0. </TD><TD> 0.283E+02 0.246E+02 0.581E+01 0.539E+01 0.332E+00 0.306E+00 0.275E+00 0.317E+00 0.299E+00 0.192E+00 0.173E+00 0.421E-01 0.000E+00 0.367E+02 0.777E+03 0.139E+04 0.968E+03 0.602E+04 0.437E+04 0.140E+03 0.370E-01 0.000E+00 0.000E+00 0.553E+00 0.159E+00 0.969E-02 0.000E+00 0.000E+00 0.112E+00 0.000E+00 0.000E+00 0.187E+00</TD>
</TR>
</TABLEDATA>
</DATA>
</TABLE>
</RESOURCE>
</VOTABLE>

2.7 Method: getDataPointsSpectraSpacecraft

This Method returns interpolated physical simulation parameters along a given spacecraft. This method returns a VoTable with the different url for each requested parameter.

Inputs :

ResourceID: (Mandatory) A string describing which dataset is requested (consistent with the tree.xml file). It provides a unique ID. It replaces the RUN_ID from the D3.10 document. The ResourceID of the NumericalOutput is accessible as easily as the RunID in the tree.xml and provide a much more precise information. ResourceID points to the 3D Cubes files in which the interpolation is to be made. The correspondence in the DataModel is "Spase/NumericalOutput/ResourceID".

- **Variable**: **(Optional)** A **string** describing the list of individual parameters from the selected NumericalOutput separated by a comma. Each parameters are identified by the ParameterKey. This allows setting a precise definition for **Variable**. In the D3.10 document, the variable_STR values were not defined, except for simple ones, but did not address the difficulties encountered for the species, which have different names depending on runs and databases. Now, it is made simple as the couple (**ResourceID+ ParameterKey**) clearly defines the output we want to interpolate, in a unique manner. By default, all parameters inside the Numerical Ouput are sent back. The correspondence in the DataModel is:

"Spase/NumericalOutput/ResourceID/Parameter/ParameterKey"

- **SpacecraftName**: **(Mandatory)** a string indicating the name of the spacecraft as defined by AMDA and 3Dview.

- **StartTime: (Mandatory)** Time format as defined by ISO 8601 standard.

- **StopTime: (Mandatory)** Time format as defined by ISO 8601 standard.

- **Sampling: (Mandatory)** Time format as defined by ISO 8601 standard.

- **additional_params:** parameter or set or parameter which are databases dependent.

With **additional_params** value for LATMOS :

- **IMFClockAngle: (optional)** a real corresponding to the angle of rotation of a 1D points of coordinate. The rotation will be done around the symmetry axis (typically x-axis). By default, no rotation will be operated.

- **OutputFileType**: **(optional)** the output file format could be selected by the user. According to the ADD document the authorized output format are NetCDF and VOTable.

- **EnergyChannel : (optional)** A list of Energy channel number (Ch0, Ch1, Ch2,…) separated by commas. The Energy of each energy channel has to be defined . Maybe use "Spase/NumericalOutput/Parameter/Structure/Element/ParameterKey".

Outputs:

- **url_Param** : the output of this service will generate an temporary url of a file containing requested time, 3D coordinates and the ion spectra.

Example:

```
 $client = new SoapClient("http://impex.latmos.ipsl.fr/Methods_LATMOS.wsdl",
array('trace' => 1));

$param=new IMPExWSClass();
$param->ResourceID=
'spase://IMPEX/NumericalOutput/LATMOS/Hybrid/Mars_14_03_14/IonSpectra';
$param->StartTime='2010-01-01T18:00:00.000Z';
$param->StopTime='2010-01-01T18:10:00.000Z';
$param->Spacecraft_name='MEX';
$param->Sampling='60';
$param->EnergyChannel='‘



$extBases = $client->getDataPointSpectra($param);
echo $client->__getLastResponse();

class IMPExWSClass{

public
      $extraParams=array('IMFClockAngle'=>120,'OutputFileType'=>'VOTable','DebugSho
      wCmd'=>1);}
```

Example of Output 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/v1.1
http://www.ivoa.net/xml/VOTable/v1.2' xmlns='http://www.ivoa.net/xml/VOTable/v1.2'>
```

<DESCRIPTION>Imported from AMDA</DESCRIPTION>
<RESOURCE>
<DESCRIPTION>DD1gzem0_mex_xyz</DESCRIPTION>
<TABLE name='DD1gzem0_mex_xyz'>
<FIELD datatype='char' ID='Time' name='Time' xtype='dateTime' arraysize='*' ucd='time.epoch'>
<DESCRIPTION>Time</DESCRIPTION>
<VALUES null='NaN'/>
</FIELD>
<FIELD datatype='float' ID='X' name='X' unit='Rm' ucd='pos.cartesian.x' utype='stc:AstroCoords.Position3D.Value3.C1'>
<DESCRIPTION>mex_xyz - Type : Local Parameter @ CDPP/AMDA - Name : xyz_mso - Units : Rm - Size : 3 - Frame : MSO - Mission : MEX - Instrument : ephemeris - Dataset : orbit</DESCRIPTION>
<VALUES null='NaN'/>
</FIELD>
<FIELD datatype='float' ID='Y' name='Y' unit='Rm' ucd='pos.cartesian.y' utype='stc:AstroCoords.Position3D.Value3.C2'>
<DESCRIPTION>mex_xyz - Type : Local Parameter @ CDPP/AMDA - Name : xyz_mso - Units : Rm - Size : 3 - Frame : MSO - Mission : MEX - Instrument : ephemeris - Dataset : orbit</DESCRIPTION>
<VALUES null='NaN'/>
</FIELD>
<FIELD datatype='float' ID='Z' name='Z' unit='Rm' ucd='pos.cartesian.z' utype='stc:AstroCoords.Position3D.Value3.C3'>
<DESCRIPTION>mex_xyz - Type : Local Parameter @ CDPP/AMDA - Name : xyz_mso - Units : Rm - Size : 3 - Frame : MSO - Mission : MEX - Instrument : ephemeris - Dataset : orbit</DESCRIPTION>
<VALUES null='NaN'/>
</FIELD>
<FIELD name="ParticleFlux" ID="col5" ucd="phys.flux" utype="" datatype="float" unit="Ions.m-2.s-1.str-1.eV-1" arraysize="32" />
<PARAM name="EnergyRange" unit="eV" ucd="instr.param" datatype="float" arraysize="32" value=" 0., 10.0, 13.0, 16.9, 22.0, 28.6, 37.1, 48.3, 62.7, 81.6, 106.0, 137.9, 179.2, 233.0, 302.9, 393.7, 511.9, 665.4, 865.0, 1124.6, 1461.9, 1900.5, 2470.6, 3211.8, 4175.4, 5428.0, 7056.4, 9173.3, 11925.3, 15502.9, 20153.8, 26199.9, 34059.9"/>
<DATA>
<TABLEDATA>
<TR>
<TD> 2010-01-01T16:00:25.000Z </TD><TD> -1.28777 </TD><TD> -1.18784 </TD><TD> -3.61010 </TD><TD> 0.134E+01 0.187E+02 0.667E-01 0.361E+00 0.281E+00 0.308E+00 0.299E+00 0.375E+00 0.315E+00 0.479E+00 0.237E+01 0.134E+01 0.228E+01 0.703E+02 0.120E+04 0.302E+04 0.140E+04 0.105E+05 0.102E+05 0.555E+03 0.567E+01 0.141E+02 0.000E+00 0.000E+00 0.000E+00 0.000E+00 0.134E+00 0.400E+00 0.354E-01 0.000E+00 0.000E+00 0.000E+00</TD>
</TR>

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