XML Mesh format

Kirill Terekhov

March 7, 2016

Abstract

This document describes the mesh format. It gives a simple example for the layout of the file in XML-compatible format and step by step describes meaning of each tag and attribute.

1 Example of file

The example of the layout of the file is presented below.

```
<!-- For compatibility with browsers
       one can add style-sheet here -->
<?xml version="1.0" encoding="utf-8"?>
<!-- Define parallel mesh with 2 subdomains -->
<ParallelMesh Number="2" [Layers="2" Element="Face"]>
  <!-- Definition of the first mesh-->
  <Mesh [Name="Box"] [RepairOrientation="False"]>
    <!-- Describe data that is defined on the mesh -->
    <!-- Number can be used to match
         the number of entries and
         optimize memory allocation -->
    <Tags [Number="8"]>
      <!-- See further text for description of each
         attribute -->
      <Tag Name="GLOBAL_ID"
           Size="1"
           Type="Integer"
           Sparse="Sets"
           Definition="Cells,Faces,Nodes,Sets" />
     <!-- Size is optional, default: Variable,
          Type is optional, default: Real,
          Sparsity is optional, default: None,
          Name and Definition are mandatory -->
      <Tag Name="PERMEABILITY_TENSOR"
           [Size="6"]
           [Type="Real" ]
         [Sparse="None"]
         Definition="Cells" />
    <Tag Name="BOUNDARY_PRESSURE"
         Size="1"
         Type="Real"
         Sparse="Faces"
         Definition="Faces" />
    <Tag Name="WELL_INDEX_WELL0"
         Size="1"
         Type="Real"
         Sparse="Cells"
         Definition="Cells" />
    <Tag Name="BOUNDARY_DISPLACEMENT"
         Size="3"
         Type="Real"
         Sparse="True"
         Definition="Nodes" />
    <Tag Name="CONNECTIONS"
         Size="Variable"
         Type="Reference"
         Sparse="None"
         Definition="Faces" />
    <Tag Name="FLUX"
         Size="1"
         Type="Variable"
         Sparse="None"
         Definition="Faces" />
    <Tag Name="WHATEVER"
         Size="Variable"
         Type="Bulk"
         Sparse="Sets"
         Definition="Sets" />
  </Tags>
  <!-- Define all the nodes of the mesh -->
  <Nodes Number="100" [Dimension="3"]>
    <![CDATA[
         xyz1 xyz2 xyz3
         ...
         xyz98 xyz99 xyz100
     ]]>
  </Nodes>
  <!-- Optionally define the edges -->
  [<Edges Number="...">...</Edges>]
  <!-- Optionally add 100 faces.
       It is not mandatory to specify all
       the faces of the mesh -->
  <Faces [Number="100"]>
      <!-- 50 faces out of 100 are defined
           by connections to nodes. -->
      <Connections Number="50"
                   Type="Nodes"
                   [HighOrder="False"]
                   [Offset="1"]>
       <!-- Data starts from the number
            of nodes and then node list-->
        <![CDATA[
          3 1 2 3
          3 2 3 4
          4 4 6 5 1
           ...
        ]]>
      </Connections>
      <!-- 50 faces out of 100 are defined
           by connections to edges. -->
      <Connections Number="50" Type="Edges">
       <!-- Data starts from the number
            of edges and then edge list.
            Edges should be explicitly defined
            for the method to work. -->
        <![CDATA[
          4 0 1 2 3
          3 2 1 4
          ...
          ]]>
      </Connections>
  </Faces>
  <!-- Cells are mandatory. -->
  <Cells Number="100">
      <!-- 50 faces are defined by nodes. -->
      <Connections Number="50" Type="Nodes">
      <!-- Data starts from the number of nodes
           and then follows a list of nodes. -->
        <![CDATA[
        4 1 2 3 4
        8 3 2 4 1 5 6 8 9
        ...
        ]]>
      </Connections>
     <!-- Can define a general
          polyhedron built of faces-->
      <Connections Number="50" Type="Faces">
      <!-- Data starts from the number of nodes
```

```
            and then follows a list of nodes. -->
         <![CDATA[
         10 1 2 3 4 5 6 7 8 9 10
          6 1 4 6 8 5 10
          ...
         ]]>
       </Connections>
     </Cells>
  <!-- Sets of elements. -->
  <!-- Offset descibes offset in enumeration of
       Parent, Child and Sibling -->
   <Sets Number="100" [Offset="1"]>
      <!-- Set with boundary elements. -->
      <!-- Offset describes a shift in
           enumeration of set elements -->
      <Set Name="BOUNDARY_SET"
           Size="100"
           [Parent="Unset"]
           [Child="10"]
           [Sibling="2"]
           [Comparator="Unordered"]
           [Offset="1"]>
        <!-- Type and offset as they are provided
             in <Cell> and <Face> tags -->
        <![CDATA[Face:1 Face:4 Face:10 ... Cell:12]]>
       </Set>
       <Set> ... </Set>
   </Sets>
  <!-- Define the data, the number of data sets should be equal to the number of tags -->
   <Data Number="11">
      <!-- TagName maps data set to the tag -->
      <!-- GLOBAL_ID -->
      <!-- Attribute Sparse defines
           how to read in the data -->
      <DataSet SetType="Nodes" TagName="GLOBAL_ID" [Sparse="False"]>
          <![CDATA[ 1 2 3 4 ... 100 ]]>
      </DataSet>
      <DataSet SetType="Faces" TagName="GLOBAL_ID">
          <![CDATA[ 1 2 3 4 ... 100 ]]>
      </DataSet>
      <DataSet SetType="Cells" TagName="GLOBAL_ID">
          <![CDATA[ 1 2 3 4 ... 100 ]]>
      </DataSet>
      <DataSet SetType="Sets" TagName="GLOBAL_ID">
```

```
            <![CDATA[ 1 2 3 4 ... 100 ]]>
        </DataSet>
        <!-- PERMEABILITY_TENSOR -->
        <!-- No need to provide set type
             since there is only one type for the tag.
           -->
        <!-- SetSize parameter in data repetition
             parameter describes total number of cells
           -->
        <DataSet [SetType="Cells"]
                  TagName="PERMEABILITY_TENSOR">
            <![CDATA[ {550,450,0,550,0,1}*SetSize ]]>
        </DataSet>
         <!-- BOUNDARY_PRESSURE -->
         <!-- Sparse="False" here tells that although
              BOUNDARY_PRESSURE is not a dense data
              the data below is listed for each element
              of the set BOUNDARY_SET -->
        <DataSet SetType="SetData"
                 TagName="BOUNDARY_PRESSURE"
                 SetName="BOUNDARY_SET"
                 Sparse="False">
             <![CDATA[500 550 600 ... ]]>
        </DataSet>
        <!-- Next dataset -->
        <DataSet>...</DataSet>
        <!-- BOUNDARY_DISPLACEMENT -->
        <!-- We do not have a set that reffers to
             boundary nodes thus we have to refer
             to each node globally. -->
        <DataSet SetType="Nodes"
                 TagName="BOUNDARY_DISPLACEMENT"
                 Sparse="True">
             <!-- Each entry is preceded by
                  the number of element it
                  belongs to -->
             <![CDATA[
               15 {0.1,0.2,0.3}
               20 {0.3,0.1,0.1}
               30 {0.2,0.3,0.4}
             ]]>
        </DataSet>
        <!-- CONNECTIONS -->
        <!-- Optionally can have Offset for
             enumeration of elements -->
        <DataSet TagName="CONNECTIONS" [Offset="1"]>
```

```
           <!-- List of elements inside of scopes-->
           <![CDATA[
               {Cell:1,Cell:4,Face:5}
               {Cell:4,Cell:5}
               {Cell:10, Cell:2, Cell:4, Cell:9}
            ]]>
       </DataSet>
       <!-- FLUX -->
        <DataSet TagName="FLUX">
           <!-- The value and corresponding
                variations -->
            <![CDATA[
               (0.123;3;1;0.5;4;0.6;105;-1.1)
               (0.234;2;4;0.6;5;-0.6)
               (0.567;10;0.1;2;0.2;4;-0.4;9;0.1)
               ...
               ]]>
        </DataSet>
        <!-- WHATEVER -->
        <DataSet TagName="WHATEVER">
          <Sets>
            <!-- It's a tag represented with binary
                 values of variable size, first comes
                 number of element then size and
                 then values or some interpretable
                 expression. -->
            <![CDATA[
              25 {FF,AF,AE,00,11}
              55 {AF,B0,11,EF}
              60 EE*8
              80 "hello world"
             ]]>
           </Sets>
        </DataSet>
    </Data>
  </Mesh>
  <Mesh>
    <!-- A very similar construct -->
  </Mesh>
</ParallelMesh>
```

2 ParallelMesh

The purpose of this XML tag is to optionally describe the number of meshes contained in the file in attribute ”Number”. As an additional information it can provide number of ghost layers between meshes in attribute ”Layers” and the type of elements used to compute adjacency for ghost layers in attribute ”Element”. This information is reserved for the future use.

3 Mesh

This xml tag wraps all the data of the mesh. The optional attribute ”Name” can be used to address elements of this mesh from another mesh. The optional attribute ”RepairOrientation” if set to ”True” will correct the orientation of the faces.

4 Tags

This xml tag declares the tags of the data that are present on the mesh. Optional attribute ”Number” corresponds to the total number of tags defined.

4.1 Tag

Each xml tag contains attributes that describe the way the data is stored on the mesh. The following attributes can be defined:

Name The name assigned to the data. Cannot be the same for two different tags. Mandatory to provide.

Size Number of records of the data on each element. Can be any positive number or ”Variable”. ”Variable” means that the data may have different number of entries on each element. Optional, default: ”Variable”.

Type Type of the data that tag represents, can be either ”Real” or ”Integer” or ”Bulk”, or ”Variable” or ”Reference” or ”RemoteReference”. ”Real” corresponds to floating numbers; ”Integer” - to integral numbers; ”Bulk” can represent any binary data; ”Variable” can store a floating point value and corresponding variations represented by pairs of integral number and floating point coefficient; ”Reference” and ”RemoteReference” represent links to mesh elements or sets or mesh itself. Optional, default: ”Real”.

Sparse Defines that data is present only on some elements of the indicated element type. Can be either Sets or Cells or Faces or Edges or Nodes or None. Optional, default: None.

Definition Defines types of elements that posses the data. Can be Mesh, Sets, Cells, Faces, Edges or Nodes. Mandatory to provide.

The ”GLOBAL_ID” tag in the example defines a single integer entry on all the nodes, faces and cells, and on some sets. The ”WHATEVER” tag defines a binary data of arbitrary size only to some sets. When one would like to have just a handful of entries unrelated to the mesh elements one can prescribe them to the mesh, i.e. write Definition=”Mesh”.

5 Nodes

This is a mandatory XML tag for the file that describes all the nodes of the mesh. The ”Number” attribute describes the total number of nodes. Optional attribute ”Dimension” tells the number of space dimensions, that is the number of coordinates in each entry, defaults to 3. The contents of the XML tag inside of ”$<![CDATA[]]>$” can be entered in any format suitable to represent vectors as described in § 9. Nodes are mesh elements of dimension 0.

6 Faces, Edges, Cells

XML Tags Faces and Edges are optional and could be used to define only some of the elements of the mesh. For example one may introduce only boundary faces in XML tag Faces. The XML Tag Cells is mandatory and it represents cells of the mesh. One can optionally provide an optional attribute ”Number” that will be used to check that the number of elements red is correct. Edges, faces and cells are mesh elements of dimension 1, 2 and 3 respectively.

6.0.1 Connections

XML tag ”Connections” is used to provide the connection of each listed element to elements of lower dimension. Attributes are ”Type” that describe types of listed elements; ”Number” - total number of listed elements; ”Offset” - optional attribute that describes first position of listed elements, default 0. It’s required that the type of elements in ”Type” has lower dimension than constructed element. It is possible to describe some cells constructed of nodes and some cells constructed of faces by using consecutive XML tags ”Connections”, see example for details.

The content of the XML tag inside of ”$<![CDATA[]]>$” should start from the number of elements connected followed by a list of positions of elements. For example if we have a record 3123 for a face with connection to nodes then this record specifies that the face consists of 3 nodes namely node 1, node 2 and node 3. Enumeration of nodes here corresponds to the order of nodes in which their coordinates are listed in XML tag ”Nodes”.

7 Sets

This XML tag encloses all the mesh sets. An optional attribute ”Number” describes the total number of the sets in the mesh and will be compared to actual number of sets red from file. Optional keyword ”Offset” can be used to determine shift in positions of sets provided in ”Child”, ”Sibling” and ”Parent” attributes of individual sets.

7.1 Set

Describes each set. Sets could be arranged into an arbitrary tree, such as an octree or a kd-tree. Attributes are:

Name Assigns the name to the set. The name cannot be the same for two different sets. Mandatory to provide.

Size Number of elements belonging to the set. Can be zero. Mandatory to provide.

Parent A parent set for tree hierarchy. Can be ”Unset” or a position of the set in the order at which the set appears in the file. Optional, default ”Unset”.

Child A first child of the current set in the tree hierarchy. Optional, default ”Unset”.

Sibling A sibling of the current set or the next child of the parent. Optional, default ”Unset”.

Comparator Represents arrangement of the elements of the set. Can be ”Unsorted”, ”Identificators”, ”Centroid”, ”Hierarchy”, ”Handle”. Optimized algorithms are used when set is sorted. Optional, default: Unsorted.

Offset Sets a shift in the enumeration of elements. Optional, default: 0.

In the contents of the XML tag ”Set” inside of ”$<![CDATA[]]>$” all the mesh elements are listed in whatever order is needed. They will be reordered internally if ordering was prescribed. The ordering happens after the mesh data is attached to the elements. Each element is listed as ”type:position” where type can be either Mesh or Set or Cell or Face or Edge or Node. The position corresponds to the position in the order of elements in which their records are encountered in corresponding XML tags.

8 Data

Encloses all the data of the mesh. Optional attribute ”Number” can be added to control the number of ”DataSet” XML tags encountered.

8.1 DataSet

XML tag ”DataSet” can have the following attributes:

TagName The name of the tag to which the data corresponds. Mandatory to provide.

SetType The domain of definition for the data. Can be either ”Mesh” or ”Sets” or ”Cells” or ”Faces” or ”Edges” or ”Nodes” or ”SetData”. When the value is ”SetData” then it is required to provide ”SetName”. Optional when the corresponding tag in TagName is defined only on one type of elements, then ”SetType” is deduced from that type, otherwise mandatory to provide.

SetName The name of the set whose elements get the data. Must be provided only if attribute ”SetType” has the value ”SetData”.

Sparse Describes the way of data representation. If value is ”True” then each data entry should be preceded by element number to which the data belongs. Optional, deduced to ”False” when ”SetData” is provided and from the attribute ”Sparse” of the corresponding tag otherwise.

Offset Describes shift in positions of references. Used only for tags of type ”Reference” and ”RemoteReference”.

MeshName Describes the mesh for RemoteReference. Used only for tags of type ”RemoteReference”, optional.

In the contents of the XML tag ”DataSet” inside of ”$<![CDATA[]] >$” the data is provided in the format described in § 9.

When only some of the faces or edges are defined, the appearance of their data defined in ”DataSet” should correspond to the order in which they are encountered in XML tags ”Faces” or ”Edges”.

9 Data representation

Data representation by type:

Real Real data is represented by a number with floating point. Floating point should be represented by dot. Example: 3.141592.

Integer Integer data is represented by a number. Example: 12345.

Bulk Binary data is written in hex notation. The notation uses two letters in the range $[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F]$ and represents a number from 0 to 255. Text can be represented by a string enclosed into commas, i.e. ”hello world”.

Variable Variable is represented by a set of pairs of floating point number and integrable number. The set is enclosed into scopes and is separated with semicolons. Example: $(0.123; 3; 0.5; 1; 0.6; 4; -1.1; 105)$. The first 0.123 is the value of the variable, then 3 is the number of derivatives. Coefficients are $0.5, 0.6, -1.1$ for the entries $1, 4, 105$.

Reference Reference is used to address an element of the current mesh and represented by type of element followed by colon and by the number of element. Example: Cell:15. The current mesh can be always addressed with Mesh:0.

RemoteReference Remote reference is used to address an element in another mesh. Represented with mesh name in quotes followed with colon, type of element, then colon again and at last the number of element. Example: ”Box”:Cell:15. When attribute MeshName=”Box” is provided in XML tag ”DataSet” then the representation is just Cell:15.

9.1 Vector data representation

When the data is the array of entries, it should be recorded in scopes {,} and entries should be separated with the comma.

9.2 Repetition

Any data entry can be repeated multiple times when followed with multiplication sign and the number of repetitions. For example 0.1*5 will repeat 0.1 five times and {0.1,0.2,0.3}*10 will repeat the vector {0.1,0.2,0.3} ten times. A keyword SetSize can be used as the multiplier value to indicate that the number of repetitions should correspond to the number of elements contained in the provided set or of the provided type, i.e. {0.1,0.2,0.3}*SetSize. When the representation of data in ”DataSet” is sparse, the data could be repeated only for the current element. An arithmetic operation with SetSize is possible, i.e. {0.1,0.2,0.3}*(SetSize/2).