ProFiDo XML Workflow Format Specification

Falko Bause, Philipp Gerloff, Alparslan Kirman, Jan Kriege, Daniel Scholtyssek
Informatik IV, TU Dortmund
profido@ls4.cs.uni-dortmund.de

July 24, 2014

ProFiDo (Processes Fitting Toolkit Dortmund) [4] is a graphical toolkit that provides a consistent use of commandline-oriented tools for the fitting of stochastic processes and distributions. ProFiDo itself provides only a core functionality to handle graphs including support for storing and loading graphs and the parameter specification of graph elements. This document describes the file format in which graphs/workflows created with ProFiDo are saved. Each supported fitting tool corresponds to a node in ProFiDo and since those tools are not hard-coded into ProFiDo's Java code, but are read from a configuration file, this document is closely related to [2] where the format of the configuration file is specified.

1 Document Structure

A workflow in ProFiDo consists of different types of nodes for the different fitting tools and edges that connect the nodes and define in which direction results are propagated from one node to the other. Consequently, the file format has to account for graphical information (e.g. coordinates), that are available for all types of nodes and edges, and tool-specific attributes, that depend on the fitting tool which the node corresponds to and are specified in ProFiDo's XML configuration file [2].
The document description should begin with an XML declaration like for example `<?xml version="1.0"?>`. The single root element of the workflow description is called workflow with the start tag `<worklow>` and the end tag `</workflow>`. The whole workflow specification is expected between these tags and consists of three parts:

- `<general>`... `</general>`: Between these tags some general information about the workflow is expected. Possible contents are specified in Sect. 2.
- `<nodes>`... `</nodes>`: These tags enclose a list of all nodes of the workflow. This part of the XML description is defined in Sect. 3.
- `<edges>`... `</edges>`: Similar to nodes all edges of the workflow are listed between these tags. More details on specifying edges can be found in Sect. 4.

Example 1 (Sketch of Workflow Specification) *The general outline of a workflow specification in XML format is as follows.*

```
<?xml version="1.0"?>
<workflow>
  <general>
    ...
  </general>
  <nodes>
    ...
  </nodes>
  <edges>
    ...
  </edges>
</workflow>
```

2 General Workflow Information

A workflow contains some general information not related to edges or nodes that is stored in the `<general>` section of the XML description. This includes the name of the workflow (within the tags `<name>`), the default folder (`<defaultdir>`), the path of the exported script (`<exportdir>`) and the boolean settings that control whether the node information, the edge information and the order keys are displayed and the debug settings that control if all node or edge information is displayed (`<settings>`). Additionally the ProFiDo version that was used to create the workflow is saved within the tag <**version**>.

Example 2 (Specification of General Information) *The general part of the XML specification is structured as follows.*

```
<general>
  <name> ... </name>
  <defaultdir> ... </defaultdir>
  <exportdir> ... </exportdir>
  <settings> true true false false false</settings>
  <version> 1.3</version>
</general>
```

3 Nodes

The description of nodes is enclosed within the `<node>` tag. Each node has a unique identifier (specified by the tag `<id>`), a name (`<name>`) and coordinates (`<gridcoords>`) that determine its position on the canvas. `<gridcoords>` specifies the coordinates of

the grid in which the node is placed. Older versions of ProFiDo (prior to 1.2) used point coordinates within the tag `<coords>` to store the position of the node. The `<coords>` tag is deprecated and `<gridcoords>` should be used instead. However, ProFiDo can still load models with the old tag, if no `<gridcoords>` tag is found.
In addition, each node can have a `<comment>` tag. This tag is optional and appears only if a comment for the node is provided. `<comment>` has an attribute "visible" which can be set to "true" or "false" to control whether this information is displayed by ProFiDo.
ProFiDo distinguishes three types of nodes, input, output and job nodes, that all have different parameters. Consequently, a node description may contain one of the three tags `<input>`, `<output>` or `<job>` to specify the parameters of the corresponding node type.
For input and output nodes information about the file can be defined by using the `<file>` tags. The file information consists of the name (`<name>`), the full path (`<path>`) and the type of the file (`<type>`).
For job nodes the specification is more complicated and closely linked to ProFiDo's configuration file [2]. The configuration file lists all tools that are supported by ProFiDo and their parameters. Hence, the description of a job node must contain a reference to identify the corresponding fitting tool declared in the configuration file and the actual values for its parameters. The `<name>` tag is used for identification of the fitting tool. The content of this tag must match one of the names declared in the `<program><general><name>`... `</name>`... sections of the XML configuration file. The values for the parameters of the job can be defined within the `<parameterlist>` tags. Each parameter description is enclosed by a `<parameter>` tag. For each parameter a name (`<name>`) and a value (`<value>`) have to be specified. Moreover, the tags `<enable>` and `<visible>` may contain boolean values to specify whether the parameter is enabled, i.e. used when executing the tool, and visible, i.e. printed as information, respectively. The name must match one of the names that have been defined within a `<parametergroup>` tag of the corresponding tool within the configuration file. An exception are parameters of type filename. These parameters are specified within the description of the corresponding edge.

Example 3 (Node Specification) *The part of the XML specification describing the nodes is structured as follows.*

```
<nodes>
  <node>
    <id> ... </id>
    <name> ... </name>
    <comment visible="false"> ... </comment>
    <gridcoords> ... </gridcoords>
    <input>
      <file>
        <name> ... </name>
        <path> ... </path>
        <type> ... </type>
      </file>
    </input>
  </node>
  ...
  <node>
    <id> ... </id>
    <name> ... </name>
    <comment visible="false"> ... </comment>
    <gridcoords> ... </gridcoords>
    <output>
      <file>
        <name> ... </name>
        <path> ... </path>
        <type> ... </type>
      </file>
    </output>
  </node>
  ...
  <node>
    <id> ... </id>
    <name> ... </name>
    <comment visible="false"> ... </comment>
    <gridcoords> ... </gridcoords>
    <job>
      <name> ... </name>
      <parameterlist>
        <parameter>
          <name> ... </name>
          <value> ... </value>
          <enable> ... </enable>
          <visible> ... </visible>
        </parameter>
        ...
      </parameterlist>
    </job>
  </node>
  ...
</nodes>
```

4 Edges

The definition of an edge in XML format which is enclosed by `<edge>` tags has to contain the start and end node of the edge and a set of points between start and end. Furthermore, an order key and the file that is propagated along the edge from start to end node can be specified, in particular file name, path and type are expected.
The start and end node are defined by the tags `<from>` and `<to>`, respectively. In both cases a reference to a node is expected within those tags, i.e. the identifier specified by `<id>` for that node (cf. Sect. 3). The tag `<orderkey>` is used to assign a key for ordering the edges. The order key is necessary if a job node can process several incoming edges of the same type (i.e. with the same file type) and one wants to specify

which file is passed first to the job node.
The edge points of each edge are stored within the `<points>` tags. Each point must be defined by `<gridpoint>`X Y `</gridpoint>` specifying the grid coordinates of the point. If its value is set to -1 -1 the point is automatically positioned on the canvas. Older versions of ProFiDo (prior to 1.2) used point coordinates within the tag `<point>` to store the position of the point. The `<point>` tag is deprecated and `<gridpoint>` should be used instead. However, ProFiDo can still load models with the old tag, if no `<gridpoint>` tag is found.
Finally, information about the file can be defined by using the `<file>` tags. The file information consists of the name (`<name>`) and the type of the file (`<type>`). The tags `<orderkey>`, `<name>` and `<type>` may contain an attribute visible="true" or visible="false" to control whether this information is displayed by ProFiDo.

Example 4 (Edge Specification) *The part of the XML specification describing the edges is structured as follows.*

```
<edges>
  <edge>
    <from> ... </from>
    <to> ... </to>
    <orderkey visible="false"> ... </orderkey>
    <points>
      <gridpoint> ... </gridpoint>
      ...
    </points>
    <file>
      <name visible="false"> ... </name>
      <type visible="true"> ... </type>
    </file>
  </edge>
  ...
</edges>
```

5 Example XML Workflow Specification

To clarify the previous XML definitions we will present the full XML description of a small example workflow that is shown in Fig. 1. The Workflow consists of an input node, two job nodes and an output node. The corresponding XML configuration file is given as an example in [2].

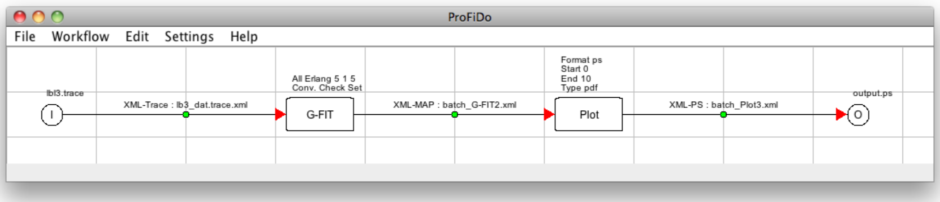

Figure 1: Example Workflow

Example 5 (XML Specification for the Workflow from Fig. 1) *In the following we will list the complete XML specification for the example workflow from Fig. 1:*

```
<?xml version="1.0"?>
<workflow>
  <general>
    <name> Example </name>
    <defaultdir> /home/fred/ </defaultdir>
    <exportdir>  </exportdir>
    <settings> true true false false false</settings>
    <version> 1.3</version>
  </general>
  <nodes>
    <node>
      <id> 0 </id>
      <name> I </name>
      <gridcoords> 0 1 </gridcoords>
      <input>
        <file>
          <name> lbl3.trace </name>
          <path> /home/fred/example/ </path>
          <type> Trace </type>
        </file>
      </input>
    </node>
    <node>
      <id> 3 </id>
      <name> O </name>
      <gridcoords> 9 1 </gridcoords>
      <output>
        <file>
          <name> output.ps </name>
          <path> /home/fred/example/ </path>
          <type> PS </type>
        </file>
      </output>
    </node>
    <node>
      <id> 1 </id>
      <name> G–FIT </name>
```

```
        <gridcoords> 3 1 </gridcoords>
        <job>
          <name>G–FIT</name>
          <parameterlist>
            <parameter>
              <name> Input Filename </name>
              <value>  </value>
              <enable> true </enable>
              <visible> true </visible>
            </parameter>
            <parameter>
              <name> Output Filename </name>
              <value>  </value>
              <enable> true </enable>
              <visible> true </visible>
            </parameter>
            <parameter>
              <name>Single Erlang</name>
              <value> 3 1 1 1 </value>
              <enable> false </enable>
              <visible> false </visible>
            </parameter>
            <parameter>
              <name>All Erlang</name>
              <value> 5 1 5 </value>
              <enable> true </enable>
              <visible> true </visible>
            </parameter>
            <parameter>
              <name>Conv. e</name>
              <value> 0.00005 </value>
              <enable> true </enable>
              <visible> false </visible>
            </parameter>
            <parameter>
              <name>Conv. Check</name>
              <value> 1 </value>
              <enable> true </enable>
              <visible> true </visible>
            </parameter>
            <parameter>
              <name> Logarithmic Trace aggregation </name>
              <value> 100 </value>
              <enable> false </enable>
              <visible> false </visible>
            </parameter>
            <parameter>
              <name> Uniform Trace aggregation </name>
              <value> 100 </value>
              <enable> false </enable>
              <visible> false </visible>
            </parameter>
        </parameterlist>
      </job>
    </node>
    <node>
      <id> 2 </id>
      <name> Plot </name>
      <gridcoords> 6 1 </gridcoords>
      <job>
        <name> Plot </name>
        <parameterlist>
          <parameter>
            <name> Format </name>
            <value> ps </value>
            <enable> true </enable>
            <visible> true </visible>
          </parameter>
          <parameter>
            <name> Start </name>
            <value> 0 </value>
            <enable> true </enable>
            <visible> true </visible>
          </parameter>
          <parameter>
            <name> End </name>
            <value> 10 </value>
            <enable> true </enable>
            <visible> true </visible>
          </parameter>
          <parameter>
            <name> Type </name>
            <value> pdf </value>
            <enable> true </enable>
            <visible> true </visible>
          </parameter>
          <parameter>
            <name> Output Filename </name>
            <value>  </value>
            <enable> true </enable>
            <visible> true </visible>
          </parameter>
          <parameter>
            <name> Trace Input Filenames </name>
            <value>  </value>
            <enable> true </enable>
            <visible> true </visible>
          </parameter>
          <parameter>
            <name> PH Input Filenames </name>
            <value>  </value>
            <enable> true </enable>
            <visible> true </visible>
          </parameter>
```

```
            <parameter>
              <name> Map Input Filenames </name>
              <value>  </value>
              <enable> true </enable>
              <visible> true </visible>
            </parameter>
            <parameter>
              <name> ARIMA Input Filenames </name>
              <value>  </value>
              <enable> true </enable>
              <visible> true </visible>
            </parameter>
            <parameter>
              <name> ARTA Input Filenames </name>
              <value>  </value>
              <enable> true </enable>
              <visible> true </visible>
            </parameter>
            <parameter>
              <name> CAPP Input Filenames </name>
              <value>  </value>
              <enable> true </enable>
              <visible> true </visible>
            </parameter>
          </parameterlist>
        </job>
      </node>
    </nodes>
    <edges>
      <edge>
        <from> 0 </from>
        <to> 1 </to>
        <orderkey visible="false"> 10 </orderkey>
        <points> <gridpoint> -1 -1 </gridpoint> </points>
        <file>
          <name visible="false"> lbl3.trace.xml </name>
          <type visible="true"> XML-Trace </type>
        </file>
      </edge>
      <edge>
        <from> 1 </from>
        <to> 2 </to>
        <orderkey visible="false"> 10 </orderkey>
        <points> <gridpoint> -1 -1 </gridpoint> </points>
        <file>
          <name visible="false"> batch_G-FIT1.xml </name>
          <type visible="true"> XML-MAP </type>
        </file>
      </edge>
      <edge>
        <from> 2 </from>
        <to> 3 </to>
        <orderkey visible="false"> 10 </orderkey>
        <points> <gridpoint> -1 -1 </gridpoint> </points>
        <file>
          <name visible="false"> batch_Plot2.xml </name>
          <type visible="true"> XML-PS </type>
        </file>
      </edge>
    </edges>
  </workflow>
```

References

[1] Falko Bause, Peter Buchholz, and Jan Kriege. ProFiDo - The Processes Fitting Toolkit Dortmund. In *Proc. of the 7th International Conference on Quantitative Evaluation of SysTems (QEST) 2010*, pages 87–96, 2010.

[2] Falko Bause, Philipp Gerloff, Alparslan Kirman, Jan Kriege, and Daniel Scholtyssek. ProFiDo XML Configuration Format Specification, 2014. http://www4.cs.uni-dortmund.de/profido.

[3] Falko Bause, Philipp Gerloff, Alparslan Kirman, Jan Kriege, and Daniel Scholtyssek. ProFiDo XML Workflow Format Specification, 2014. http://www4.cs.uni-dortmund.de/profido.

[4] Falko Bause, Philipp Gerloff, and Jan Kriege. ProFiDo - A Toolkit for Fitting Input Models. In Bruno Müller-Clostermann, Klaus Echtle, and Erwin P. Rathgeb, editors, *Proceedings of the 15th International GI/ITG Conference on Measurement, Modelling and Evaluation of Computing Systems and Dependability and Fault Tolerance (MMB & DFT 2010)*, volume 5987 of *LNCS*, pages 311–314. Springer, 2010.

[5] Falko Bause and Jan Kriege. ProFiDo XML Interchange Format Specification, 2014. http://www4.cs.uni-dortmund.de/profido.