The Design of an XML-Based Experiment and Model Description Language

Andrew W. Hallagan

Department of Computer Science
Bucknell University

November 5, 2010
Big Idea

XML languages developed for

- **Experiment Description**: allows researchers to describe an experiment space in a way that is intuitive, elegant and can easily be transformed into a series of individual design points.

- **Model Description**: allows researchers to describe the structure of the components they are using without working directly with a C++ or Python script.
Roadmap

Experiment Description
- What it does.
- A conceptual example (i.e., no code).
- An implemented example.
- Results.

Model Description
- What it does.
- A conceptual example (i.e., no code).
- An implemented example.
Running Example: Csma Bridge

The OnOffHelper component in C++:

```cpp
OnOffHelper onoff ( /* ... */ );
onoff.SetAttribute ("OnTime",
                   RandomVariableValue (ConstantVariable (1)));
onoff.SetAttribute ("OffTime",
                   RandomVariableValue (ConstantVariable (0)));
onoff.SetAttribute ("DataRate",
                   DataRateValue (DataRate(50000)));
onoff.SetAttribute ("PacketSize",
                   UintegerValue (uint32 (512)));
```
What the Experiment Description Language Does

- Lists each experimental factor in turn and describes some list of parameter values which the factor will take on.
- Provides constructs for building different types of lists.
- Describes a factorial experiment design.
- Prunes factorial experiment design.
Conceptual Example

Suppose the following experiment space:

\begin{align*}
onTime & \in [10, 16, 27, 31] \\
offTime & \in [10 \times i : 0 \leq i \leq 3] \\
packetSize & \in \text{ListA} \\
dataRate & \in \text{ListA} \\
\text{ListA} & = [1, 2, 3]
\end{align*}

With the restriction on each design point:
\begin{align*}
\offTime & > \onTime \quad \text{and} \quad \packetSize = \dataRate.
\end{align*}
Implemented Example

The OnOffHelper experiment space in XML:

```xml
<factor>offTime</factor>
<test>EQUALS</test>
<lconst>10</lconst>
<op>MULT</op>
<rvar>i</rvar>
<where>
  <range>
    <var>i</var>
    <lo>0</lo>
    <hi>3</hi>
    <delta>1</delta>
  </range>
</where>
```
## Results: Pruned Experiment Space

<table>
<thead>
<tr>
<th>Index</th>
<th><code>packetSize</code></th>
<th><code>offTime</code></th>
<th><code>onTime</code></th>
<th><code>dataRate</code></th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>1</td>
<td>20</td>
<td>10.0</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>30</td>
<td>10.0</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>2</td>
<td>10</td>
<td>10.0</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>20</td>
<td>10.0</td>
<td>2</td>
</tr>
<tr>
<td>22</td>
<td>2</td>
<td>30</td>
<td>10.0</td>
<td>2</td>
</tr>
<tr>
<td>32</td>
<td>3</td>
<td>20</td>
<td>10.0</td>
<td>2</td>
</tr>
<tr>
<td>35</td>
<td>3</td>
<td>30</td>
<td>10.0</td>
<td>2</td>
</tr>
<tr>
<td>42</td>
<td>1</td>
<td>20</td>
<td>16.0</td>
<td>1</td>
</tr>
<tr>
<td>45</td>
<td>1</td>
<td>30</td>
<td>16.0</td>
<td>1</td>
</tr>
<tr>
<td>55</td>
<td>2</td>
<td>20</td>
<td>16.0</td>
<td>2</td>
</tr>
<tr>
<td>58</td>
<td>2</td>
<td>30</td>
<td>16.0</td>
<td>2</td>
</tr>
<tr>
<td>68</td>
<td>3</td>
<td>20</td>
<td>16.0</td>
<td>2</td>
</tr>
<tr>
<td>71</td>
<td>3</td>
<td>30</td>
<td>16.0</td>
<td>2</td>
</tr>
<tr>
<td>81</td>
<td>1</td>
<td>30</td>
<td>27.0</td>
<td>1</td>
</tr>
<tr>
<td>94</td>
<td>2</td>
<td>30</td>
<td>27.0</td>
<td>2</td>
</tr>
<tr>
<td>107</td>
<td>3</td>
<td>30</td>
<td>27.0</td>
<td>3</td>
</tr>
</tbody>
</table>
What the Model Description Language Does

- Initially, model descriptions will have very little flexibility in order to serve more like templates.
- Eventually, provides an overarching *model* built as a composition of sub-components.
- Separate from simulation script.
Model Description XML

```xml
<component>
    <name>OnOffHelper</name>
    <id>onoff</id>
    <attrlist>
        <attr>onTime</attr>
        <attr>offTime</attr>
        <attr>dataRate</attr>
        <attr>packetSize</attr>
    </attrlist>
</component>
```
Results: A Single Model Instance

<OnOffHelper id="onoff">
   <attr name="onTime" value="10.0"/>
   <attr name="offTime" value="0"/>
   <attr name="dataRate" value="1"/>
   <attr name="packetSize" value="1"/>
</OnOffHelper>

From here we can transform into executable C++ or Python!
Questions, comments?