Visualizing Your Domain Model Using Reladomo Metadata

26 September 2006

Table of Contents

1. Introduction

When a persistent set of objects is specified in Reladomo metadata, the objects can be visualized. The output can be used as documentation, or simply browsed through to gain understanding of the domain.

2. Required Tools

To use the visualization, you will need two things:

- Run the MithraGraphGenerator using Ant.
- Load the graph in yEd.

2.1. Running MithraGraphGenerator

MithraGraphGenerator is invoked via Ant. Here is an example target:

```
<target name="generate-mithra-graph-product">
    <taskdef name="mithra-graphgen"

  classname="com.gs.fw.common.mithra.generator.MithraGraphGenerator"
        loaderRef="mithraGenerator">
       <classpath refid="mithragen.classpath"/>
    </taskdef>
    <mithra-graphgen xml="${mithra.home}/xml/mithra/para/
ParaClassList.xml"
        outputFile="${generated.src.dir}/paraproduct.graphml"

  includeList="Product,PositionQuantity" followRelationshipDepth="1"
        showAttributes="primaryKey"/>
</target>
```

The elements for the `mithra-graphgen` tag above are:

The `xml` element specifies the location of the Reladomo class list -- a list of all of the classes to be graphed by Reladomo. By default, the graph generator will put all the classes in the graph if `includeList` is

not specified. The combination of `includeList` and `followRelationshipDepth` further defines what objects are shown on the graph.

The `includeList` is a comma separated list of objects to be graphed. This is an optional attribute, and omitting it will cause all the objects to be graphed. These objects, if specified, will be the root from which relationships can be followed. The example above will graph the Product class, the PositionQuantity class as well any objects that have a direct relationship from those two.

The `followRelationshipDepth` attribute specifies how deep relationships should be followed from the classes specified in `includeList`. This attribute has no effect if `includeList` is not specified. The default value is 1. To see just the classes specified in `includeList`, set this to zero.

The `showAttributes` value determines what object attributes will be visible in the output. The valid values are: "none", "all", "primaryKey" or a number (e.g. "7"). If a number is specified, the primary keys are given precedence and appear at the top of the list.

2.2. Running yEd

You can download a copy of yEd from http://www.yworks.com/products/yed/. After running yEd, you can open the file you generated (paraproduct.graphml in the above example). You'll see a jumble of boxes. Don't panic. You have to use yEd to layout the diagram. yEd and a large number of options for layout. You should experiment a bit to see what fits your model best. All these options appear under the "Layout" menu. Here are some recommendations to get you started:

- Small number of objects: Hierarchical or Orthogonal works well.
- Medium number of objects: Organic works well.
- Large number of objects: Circular works ok.

Once you have the diagram in it's final form, you can export to it to a variety of file type, including PDF, SVG and GIF. For large diagrams, you can also print the file on multiple pages.