Diagram Based UIs in Oracle ADF

Duncan Mills
Oracle Development Team
Cloud and Mobility Development Tools

Keywords:

ADF DVT DIAGRAM VISUALIZATION

Introduction

The 12.1.3 release of Fusion Middleware sees the rather exciting introduction of the new DVT Diagram component. This paper aims to give a high level picture of what diagram is all about, and what can be achieved with it. In concept, diagram is slightly different from normal ADF/ JSF components where all you do is to provide the data and set a few properties. With diagram you may have a little more work to do, but the payback is a huge in terms of flexibility and power.

Starting With the Basics

From a data perspective, diagram is pretty simple; it will take a collection of nodes that describe the shapes on the diagram (the boxes, diamonds, cloud and robots in the example screenshot below) and a collection of links that describe the edges (lines) that connect those shapes together. Unlike a tree or hierarchy viewer component there does not have to be any hierarchy in the data, relationships can run in any direction, and indeed there don't even need to be relationships.
The third partner in the diagram story is the layout. This highlights one of the big differences between diagram and many other ADF components. We realized early on that it would be impossible to come up with a definitive list of all of the ways that developers might want to lay out the shapes and links on the diagram. Just imagine the possible combinations here you have here in the way that nodes are physically laid out and how far apart they are. Given that you have a very open unstructured set of data there is really no "correct" way to do a layout. Inevitably if we had just come up with a set of common layouts they might work for a small percentage of the community but even more users would be asking for different views or more configurability to what we already had. Every diagram is different!
So the key point about the DVT diagram is that the layouts are not picked from a fixed set. We ship a set of demo layouts that you can re-use, but in most cases we expect that you'll want to create your own. This is a moderately complex task so I have linked to a series of articles at the end of this paper that can help you through this. The good news is that this ability to precisely control layout just makes the component even more useful and powerful!
Here's a screen shot of one of the diagram demos. This example embodies a classic diagram style of visualization, however, it's just the start of what you are able to achieve.

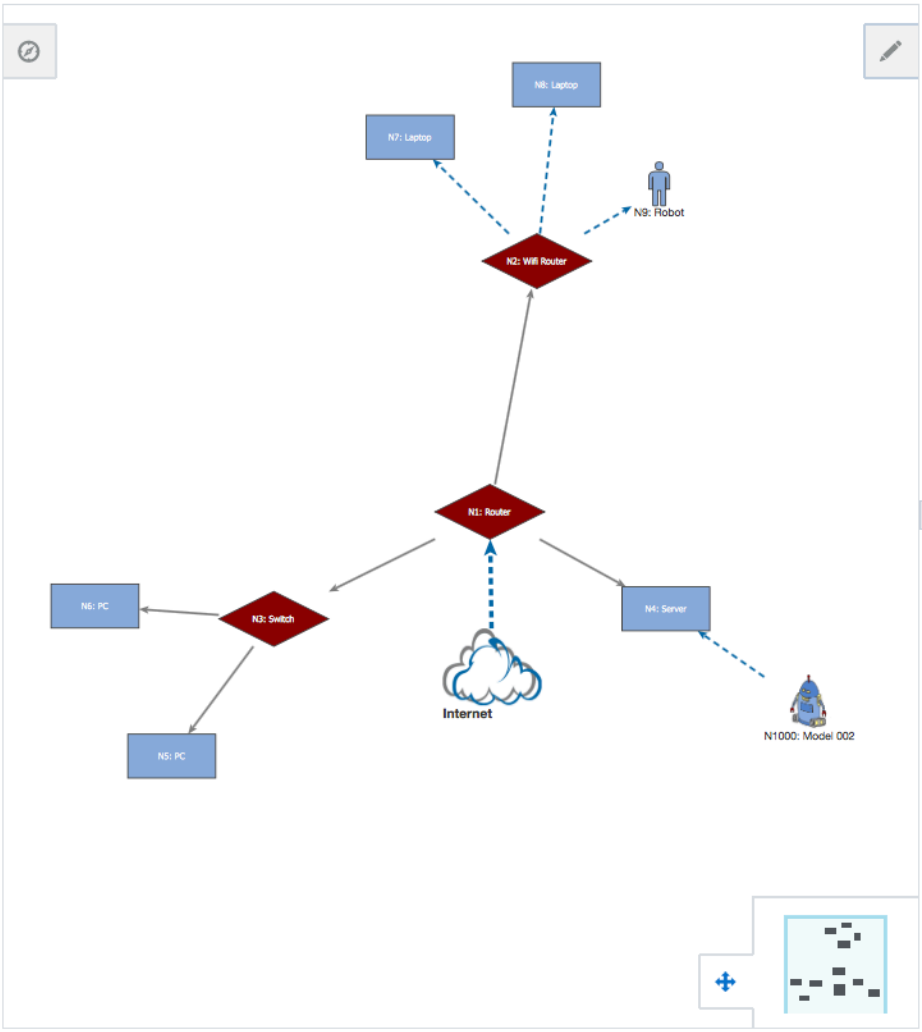


Illustration. 1: Network Style Visualization using the ADF diagram

In contrast we can create very different visualizations using diagram, which really look nothing like the classic network or flow diagram that you might expect, for example, a Sankey diagram showing energy flows from production to consumption in the UK:

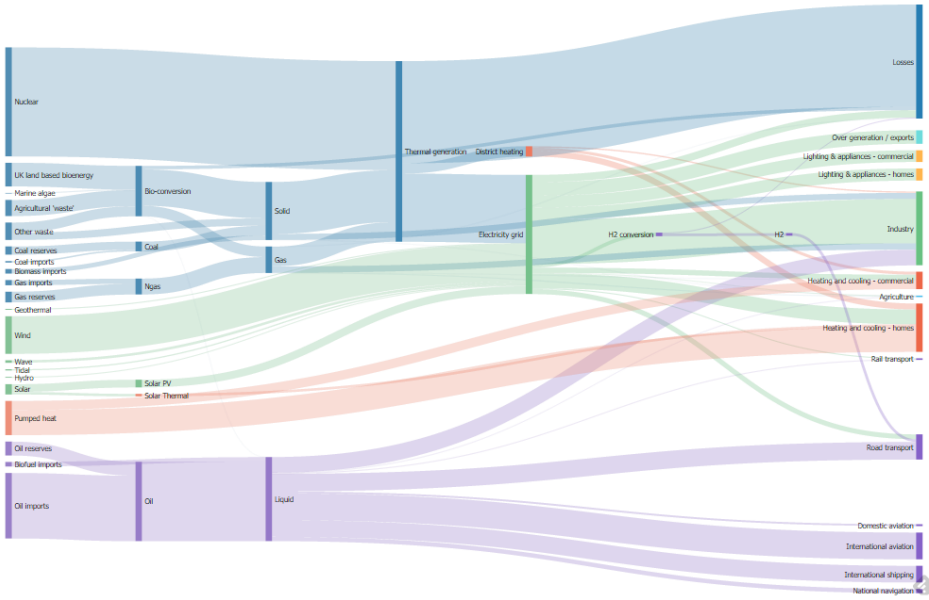


Illustration. 2: A Sankey Visualization Created with the ADF diagram

What Diagram Is Not

Although the diagram can be interactive to a degree, for example it supports drag and drop and the creation of new nodes and links, it is not a design surface. Don't pick up diagram and expect to be able to move nodes around on the surface to random positions. It's primarily a visualization tool and although, in the future, you may be able to fully emulate a freeform vector graphics program with it, we're not there yet. Saying that, with a little imagination, you can step way beyond simple visualization into something that works as an input device. The *Simple Rules Editor Demo* is an example of this. The user of the page can build up complex expressions by dragging and dropping "facts" onto the diagram, which then displays a visual representation of the rule-set as it is being built.

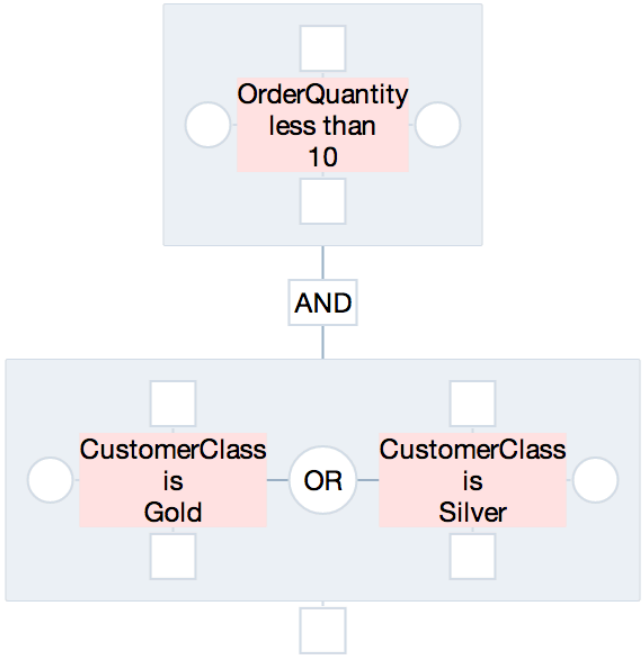


Illustration. 3: A Rule Builder Created Using ADF diagram

What the Diagram Can Show

As you can see from some of the screen shots included in this paper, there is quite a wide range of things that can be displayed on the diagram nodes. They can just be simple shapes or scalable images with a label, but they can also be more complex and like the DVT hierarchy viewer component, the nodes can contain differing amounts of information depending on the zoom level and also rich content on the nodes including the use of DVT panelCards and other layout components.

Interactivity

You have a try for yourself with the online diagram demos at http://jdevadf.oracle.com to get a feel for the component in action, but the basics are simple. The user can pan around the diagram using the mouse or the thumbnail view, and zoom, using the specialist zoom bar or mouse. Of course if the user is on a tablet then they can use their fingers with standard drag and pinch gestures to pan and zoom as well. This mobile browser support is a key feature of diagram, which will allow it to be used in the most modern of application user interfaces.

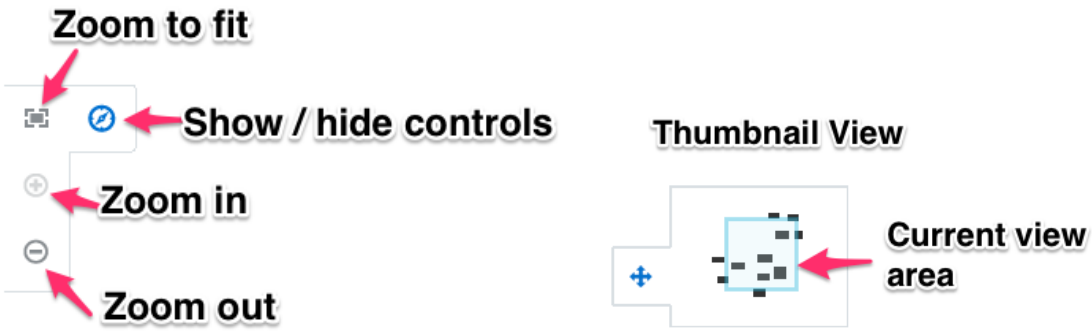
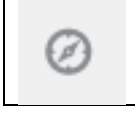


Illustration. 4: Interaction Controls on the Diagram

The nodes in the diagram are also keyboard navigable if you don't use a mouse.
Diagram nodes themselves can be grouped and stacked in various ways, if the developer chooses to allow it, when you click on a particular node you may well see an activity icon which allows you to carry out some additional operations. Here's a quick summary of the kinds of visual hints that you will see in the diagram for reference:

	**Isolate**	Allows you to "zoom in" to just view this node (and it's containees) and hide all other nodes
	**Additional Links**	When you have isolated a node clicking this icon will pull in other nodes that are directly linked to this node
	**Restore**	When you have isolated a node, this icon appears at the top is the diagram and undoes the isolation, restoring the previous view
	**Drill**	Used when nodes contain other nodes this icon allows you to drill into the container and just focus on it's contents. The diagram renders a breadcrumb trail for you to navigate back up through the hierarchy.
	**Delete**	if your diagram node is marked as editiable (ie. readOnly="false") then this icon will appear and will trigger the nodeDeleteListener associated with the diagram
	**Preview Stack**	When several nodes of the same type are "stacked together", this icon will overlay an exploded version of the stack so that you can see the individual members
	**Close Stack**	Closes the preview overlay of the stacked node and returns back to the main diagram view
	**Unstack**	Explodes the stack to show each node individually
	**Control Panel**	Located in the top / left of the diagram clicking this icon will reveal or hide the zoom controls

	**Overview Panel**	Located in the bottom / left of the diagram clicking this icon will reveal or hide the thumbnail view
	Legend Panel	Located in the top / right of the diagram clicking this icon will reveal or hide the legend for the diagram
	Palette Panel	Located in the top / right of the diagram clicking this icon will reveal or hide the palette that you can use for node types and links to drag onto an updatable diagram

More Information

In this paper I've only been able to provide you with just a short introduction to the ADF diagram and how it might be used, so I encourage you to consider how you could enhance you user interfaces using diagram then go and learn more with the following resources:

- Interactive Demos showing many of the features discussed here: http://jdevadf.oracle.com/adf-richclient-demo/faces/feature/diagram/index.jspx
- The ADF Data Visualization Blog, where I have written a whole series of articles about diagram in detail, including how to build your own layouts: https://blogs.oracle.com/data-visualizations

Contact address:

Name
Duncan Mills
Oracle Cloud and Mobility Development Tools Team

Email duncan.mills@oracle.com
Internet: http://blogs.oracle.com/groundside