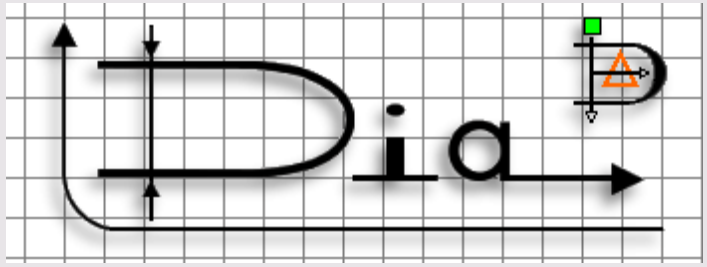
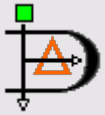



James Henstridge - james@daa.com.au
Perth, Western Australia

# What is Dia?

- Dia was originally written by Alexander Larsson.
- Dia is a structured drawing editor
- Structured diagrams are:
  - Usually composed of from a set of standard drawing elements.
  - Elements of diagram usually represent something.

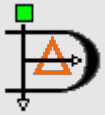


# Supported Diagram Types

- UML
- Entity-Relationship
- Network Diagrams
- Flowcharts
- Circuit diagrams
- and more ...

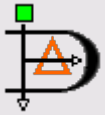
# Diagram Concepts

- Dia provides a number of concepts to support the different diagram types:
  - `Objects' (eg. a shape or line)
  - Connection points
  - Handles -- a connected handle will stick to its connection point
  - Base classes for commonly used objects (eg. orthogonal lines)



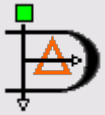
# The Toolbox

- For each diagram type, a number of objects (shapes and lines) are written.
- Objects are organised into sheets in the toolbox.
- Contents of each sheet is determined by an XML file stored in $(prefix)/share/dia/sheets or ~/.dia/sheets.



# The Diagram Display

- User interface patterned after the GIMP.
- Don't forget the right mouse button :)
- Can have multiple views of the one diagram. Updates on one view are reflected in other views.
- Can zoom in and out from the view submenu of the right mouse button menu.
- Has a support for a grid, including a snap to grid feature.
- Can do anti aliased rendering (using Raph's libart library)

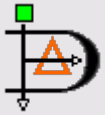


# Objects

- Objects are placed by selecting the appropriate tool from the toolbox and clicking, or clicking and dragging, on the diagram display.
- If you prefer the selected tool to reset to the arrow when an object is placed, set the "Reset tools after create" option in the preferences.
- Double clicking on an object will bring up its properties dialog.
- Middle clicking on an object will bring up an object specific menu.

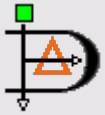
# Loading and Saving Diagrams

- Dia uses XML as its native format. By default, it also gzips the output. This gives us the readability benefits of text without the size problems.
- Dia can also export diagrams to a number of formats:
  - EPS (Encapsulated Postscript)
  - CGM (Computer Graphics Metafile)
  - SVG (Scalable Vector Graphics)
  - PNG -- uses the anti-aliased renderer
- Currently, Dia can only read its own native format.



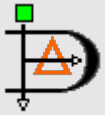
# Adding New Diagram Types

- First work out what objects make up the diagram.
- For each object, write an implementation in C. This involves:
  - Writing code to render the object. This is done in terms of an abstract Renderer object, which may render to the screen, postscript or something else.
  - Write code to react to movement of its handles
  - Implement a properties dialog and optionally an object menu.



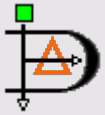
# New Diagram Types (continued)

  - Implement a distance function
  - Implement loading and saving for the object.
  - implement a few other house keeping functions
- Write a sheet file which contains all the new objects.



# Custom shapes

- There is an easier way of adding new shape type objects, which doesn't require any knowledge of C.
- Custom shapes plugin provides a way of adding new objects whose behaviour is controlled through a simple XML file.
- A custom shape XML file consists of:
  - The name of the object type.
  - the icon to use in the toolbox.

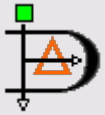


# Custom shapes (continued)

  - A description of how to draw the object. This is done using a subset of the SVG specification.
  - Positions of connection points for the object.
  - Constraints on how the object may be scaled.
- The custom shape code is suitable for any shape that scales affinely
- Does not help with adding new line types.
- Most of the flowchart shapes use the custom shape code.

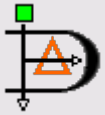
# Properties Interfaces

- Before Dia-0.84, there is no way to programatically get or set the value of an arbitrary object in a diagram.
- With the properties API, each object provides describe_props, get_props and set_props methods. By providing a standard way to modify the properties of an object, the following is possible:
  - Automatically generating a properties dialog
  - Generate a properties dialog for a group of selected objects.



# Properties Interfaces (continued)

  - Automatically load and save properties when loading/saving a diagram.
- Dia-0.84 contains a partially complete implementation of these APIs.
- The standard objects, custom shapes and groups have properties implementations.
- I am planning to add the ability to specify extra properties for custom shapes (eg. resistance value for a resistor shape).



# The Future

- A python scripting interface is currently being developped. This will allow programatic manipulation of diagrams, and with the properties interface should be powerful enough to generate XMI from Dia UML diagrams, for instance.
- Bonobo support is planned for the future. This will allow embedding Dia diagrams in other Gnome Office applications, and make it easier to embed Dia in other applications (such as a CASE tool).

# Links

- Dia web site:
  http://www.lysator.liu.se/~alla/dia/
- Dia mailing list subscription address:
  dia-list-request@lysator.liu.se
  (put subscribe in the subject).