



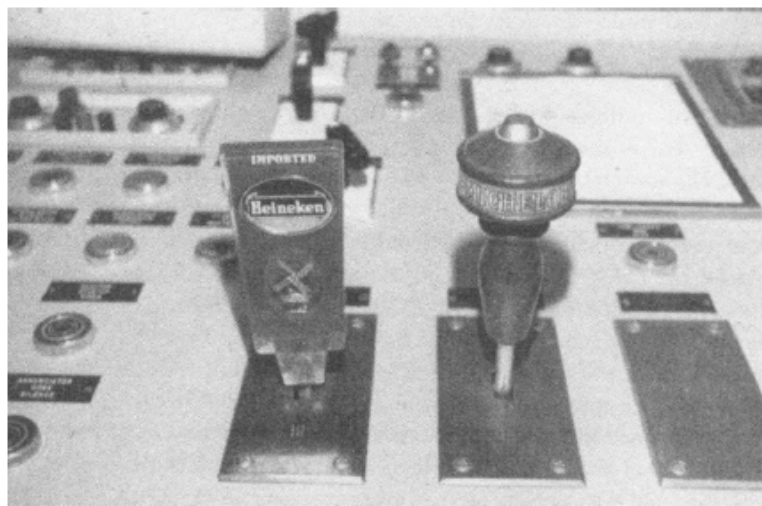
Chapter 2



Module 2 - Design



Some fun





General issues of design



- ✓ Must design large things
- ✓ Design approaches an art form
- ✓ Principal concern with *user*



How not to design



The screenshot shows a web search interface for real estate. The navigation bar at the top includes 'HOME', 'SEARCH', 'OPEN HOUSES', and 'FIND REP'. Below this, there are several overlapping and cluttered buttons for property types: 'FARM', 'COMM', 'MFAM', 'RENT', 'RESIDENTIAL', 'CONDO', and 'LOT'. The 'COMM' button is highlighted in purple. Below the buttons, there are search filters for price ('-Low Price-' and '-Hi Price-'), number of bedrooms ('-# Beds-'), and number of bathrooms ('-# Baths-'). A 'Neighbourhood:' dropdown menu is open, showing options: 'All', 'Regional Municipality', 'Airport', 'Alfred Twp', and 'Alta Vista'. Below that, a 'Property Type:' dropdown menu is open, showing options: 'All', '2 Storey', and '3 Storey'. The interface is cluttered with many overlapping elements and a lack of clear visual hierarchy.



Rules



- Avoid doing things just because you know how to do them.
- Make your designs be driven by requirements.



The design process



The design process involves both

- specification of the behaviour of a product, and
- specification of the detailed techniques used to implement the product.

In each area, there exist a range of tools and techniques that can benefit any software product.



Role of designer



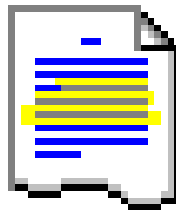
- ✓ Designer interacts with people
- ✓ Must relate designs back to requirements and constraints.
- ✓ Design must be readable, understandable, implementable.
- ✓ Designer uses *abstraction*.



Iconic abstraction



Text file using an icon:

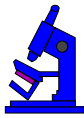




Higher level abstractions


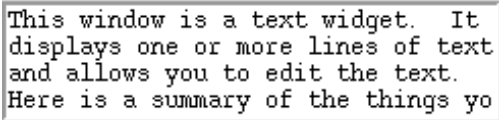



- ✓ Desktop
- ✓ Wastebasket



Building blocks



Button	Textbox	Label
	 <p>This window is a text widget. It displays one or more lines of text and allows you to edit the text. Here is a summary of the things yo</p>	 <p>Perl/Tk</p>



Building blocks



Menu	Checkbox	Radiobutton



Building blocks



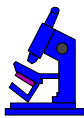
Scrollbar	Graph	Directory Tree
		<p>DirTree, display directory tree.</p>



Containers



Width	Height
<input checked="" type="radio"/> Default	<input checked="" type="radio"/> Default
<input type="radio"/> cm	<input type="radio"/> cm
<input type="radio"/> inches	<input type="radio"/> inches
<input type="radio"/> % of Page	<input type="radio"/> % of Page
<input type="radio"/> % of Column	
<input type="text" value="0"/>	<input type="text" value="0"/>



Sundries



- ✓ Cursors
- ✓ Fonts
- ✓ Colours
- ✓ Drag-n-drop
- ✓ Cut-n-paste.



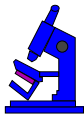
Use cases



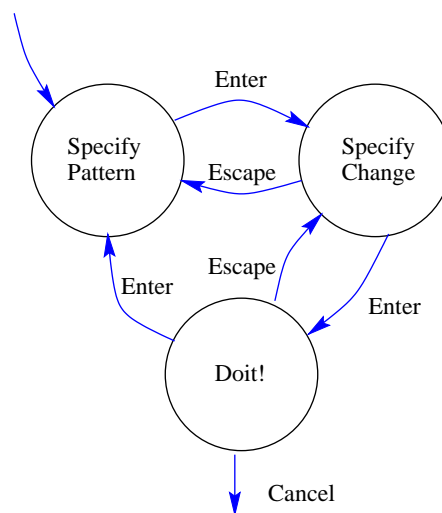
Designer imagines and proposes common scenarios², and

1. checks to see if the scenarios are *consistent*, and *complete*,
2. tries out the scenarios on people to see if they work,
3. tests the scenarios and attempts to quantify their behaviour.

²Scenarios=Use_cases. Use_cases=scenarios.

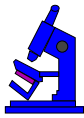
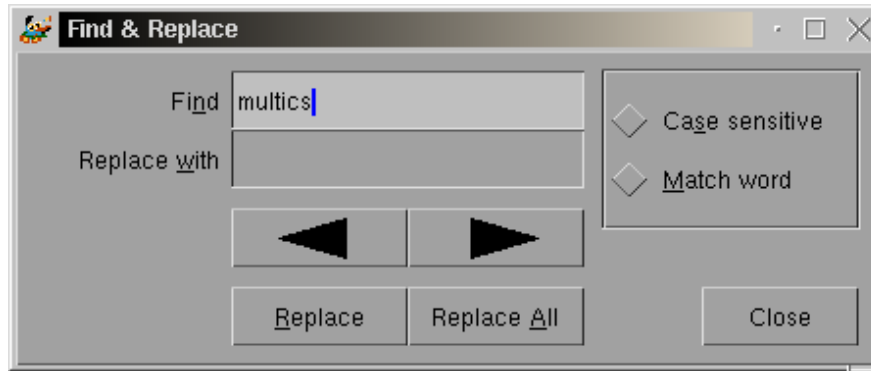


F&R state diagram

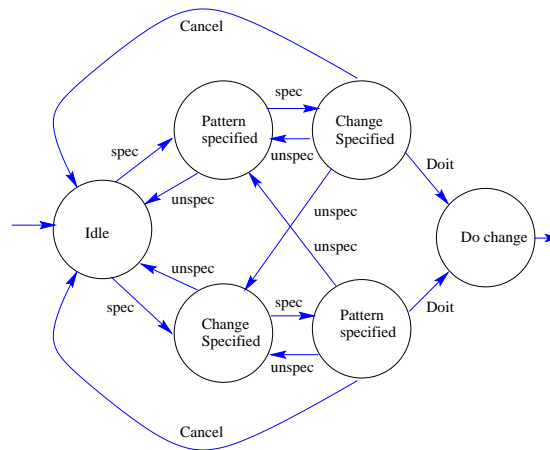




GUI-style *find-and-replace*:



State diagram





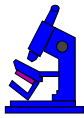
Therac25



The focus on detail related to the state of a dialog is not trivial.

- ✓ There is a well known example of a poorly constructed dialog, that contributed to the death of cancer patients in the US

<http://sunnyday.mit.edu/therac-25.html>



Modelling

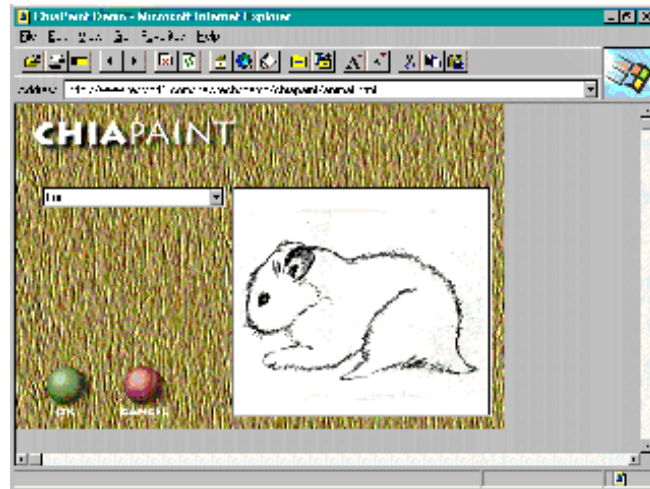


Modelling: - Used to demonstrate the UI, without actually implementing the *core* software.

Dan Bricklen's demo program (a demo copy is available at <http://www.brickin.com/>) is worth looking at for modelling a user interface. There is an amusing demo called **chiapaint**.



Chiapaint



Modelling



It is also relatively easy to model a new UI using Tcl/Tk.



OO technology



The principle features of OO technology are as follows:

1. Abstraction,
2. Information hiding,
3. Inheritance,
4. Polymorphism, and
5. Genericity



OO technology and design



Some of these OO features provide a mechanism which supports the construction of better software...

- ✓ Create library
- ✓ Generalize it
- ✓ Design becomes the detailing of new classes derived from the generalized ones...



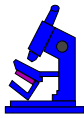
GUI design



GUI design has to meld four possibly conflicting elements:

1. **Software model** - structure of data and software
2. **User profile** - the types of users
3. **Product perception** - the mental image developed by user
4. **Product image** - the GUI - screenshots, descriptions or specifications

In general, a GUI is successful when the product perception matches the product image.



GUI specification/design



Our concern is to:

Develop a functional and behavioural response specification in terms of its cognitive aspects.

The functional and behavioural response specification is turned inside-out from a normal *software* specification. With a *software* behavioural model, we start with an analysis of states, events and actions, and specify the expected views as a result. With GUI specification, our orientation is to start with the views, and specify the states, events and actions associated with those views.



Basis for GUI design



One of the most characteristic elements of many GUI programs is the use of the event-driven software architecture. When the designer adopts this paradigm, the GUI program is viewed as a series of response routines for particular events.



Design document



- User requirement
- Environment
 - Software constraints
 - Other constraints
- Interface design
 - Overview
 - Interface description
 - * Prototype screens
 - * Functional specifications
 - * Behavioural specifications
- Testing methodology

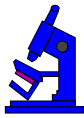


Formal GUI design



- ✓ Z can *formally* specify complex GUI interactions.
- ✓ Z tools test the specification

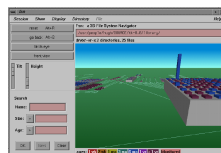
More details may be found in the handout, found at
<http://www.cs.virginia.edu/~jck/publications/zum.97.pdf>
It describes the interface to a nuclear reactor.



GUI designs



- ✓ With Therac-25, we saw how a poorly constructed interface led to deaths of patients...
- ✓ By contrast, a good interface may result in the reverse...

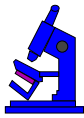




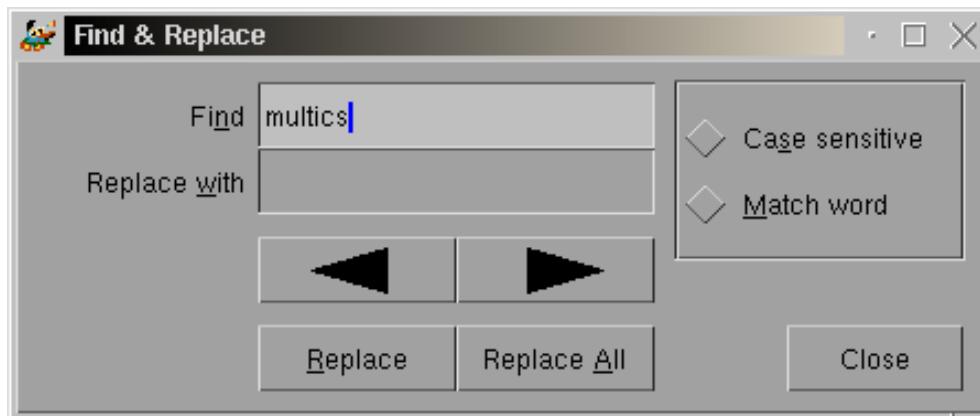
Examples of GUI designs



Here are some examples of different designs for similar things

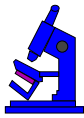
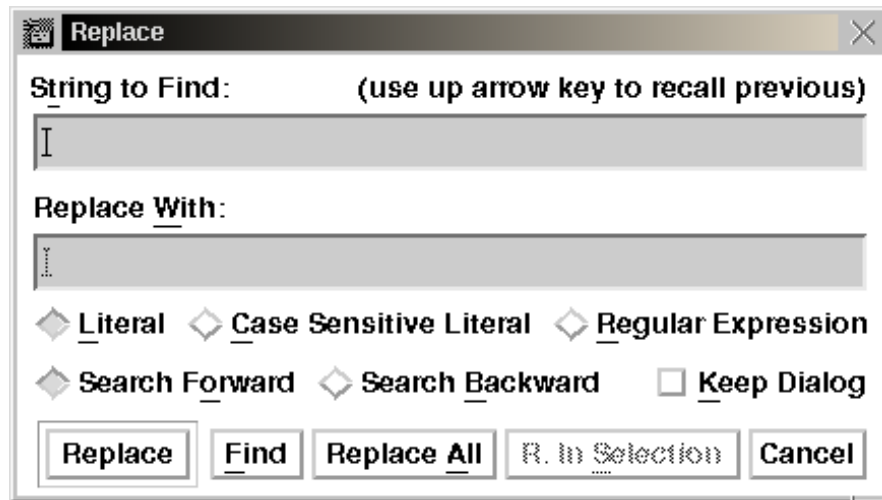


Lyx Find-and-replace:

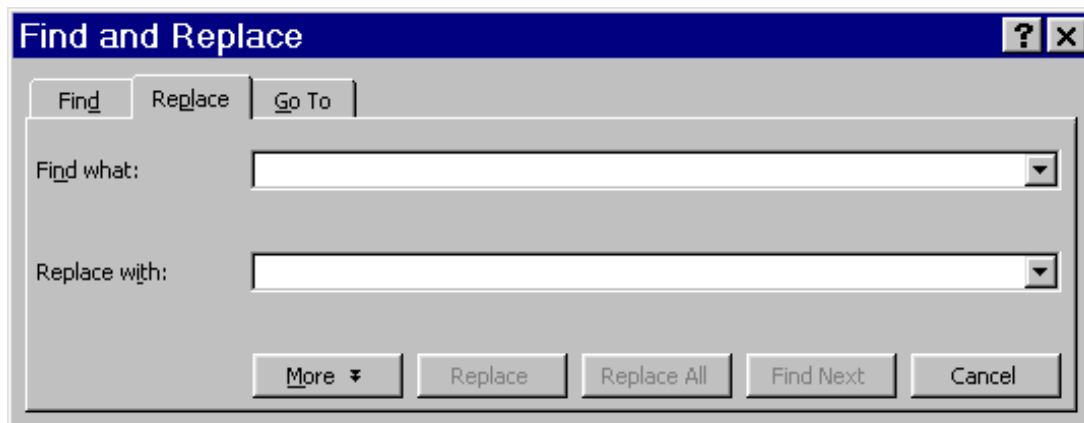




Nedit find-and-replace

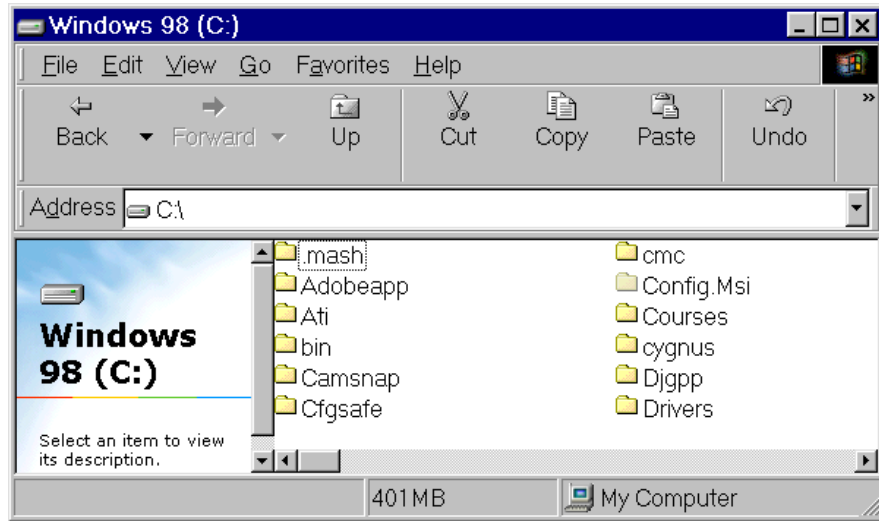


Word find-and-replace

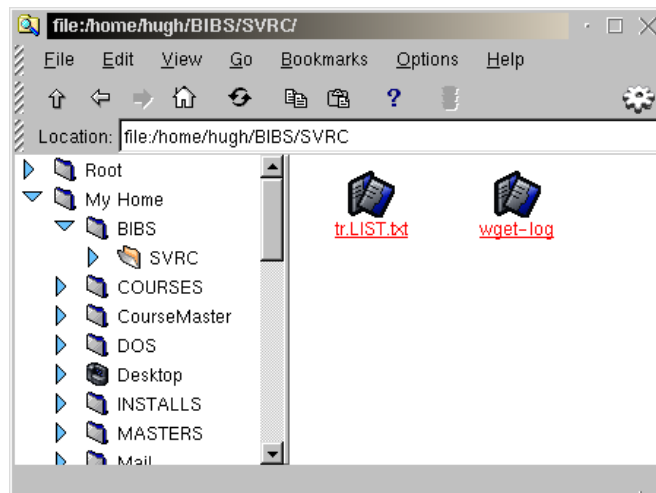




Win98 file manager

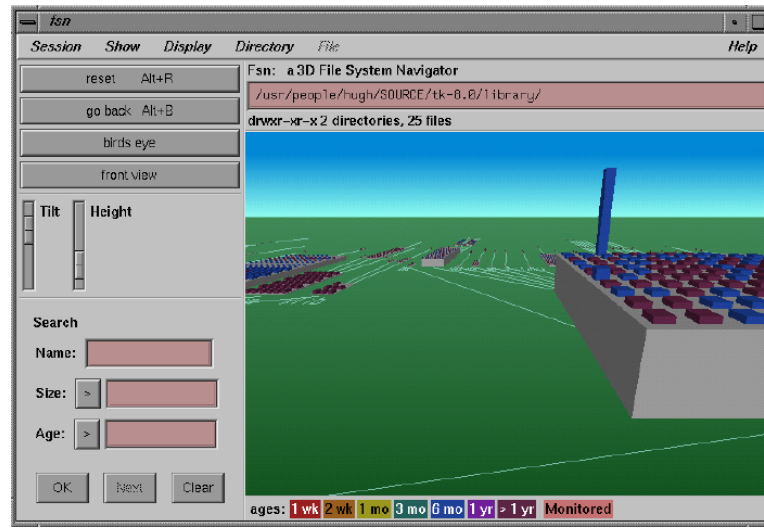


KDE file manager





FSN



Vizualization design



Visualization design has a similar structure to GUI design - a difference being the focus on the use of *analogy*.



Basis for visualization design



Eick proposes the following guidelines:

1. Focus the visualization on task-specific user needs.
2. Use a whole-database overview display.
3. Encode the data using colour, shape, size, position.
4. Use drill-down, filters and multiple linked views
5. Use smooth animation for time



Visualization document



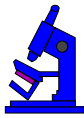
- User requirement
- Environment
 - Software constraints
 - Other constraints
- Interface design
 - Overview
 - Interface description
 - * Drill-down and other displays
 - * Encoding
- Testing methodologies



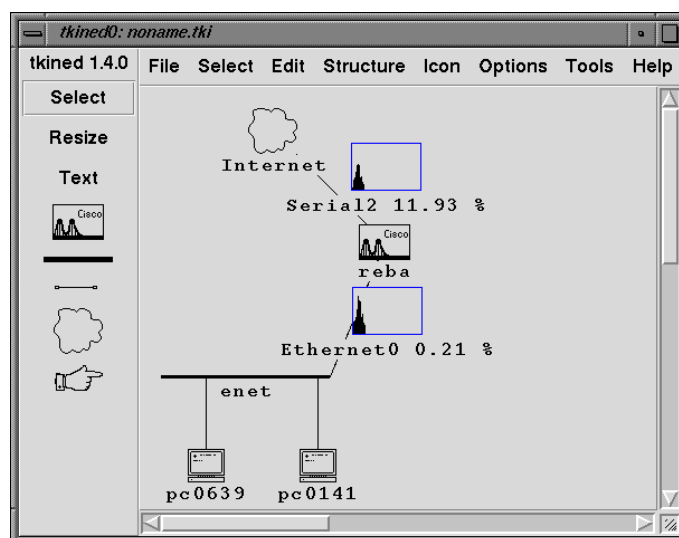
Examples of visualization



There are many examples of data visualizations, and I have just taken some from the world of network management - starting from simple graphical displays through to 3D images.

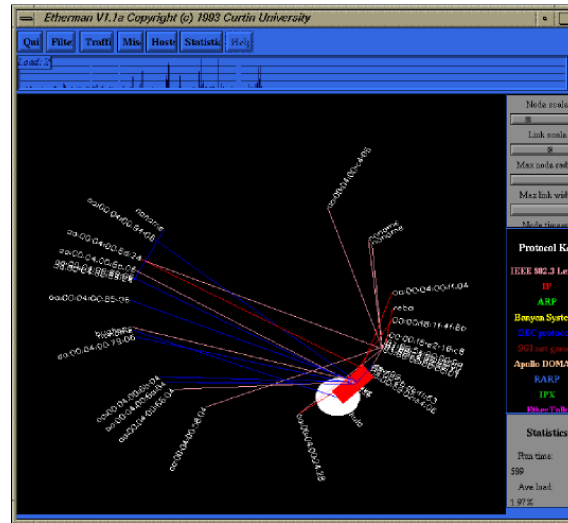


Graphs and diagramming





Compact visualization

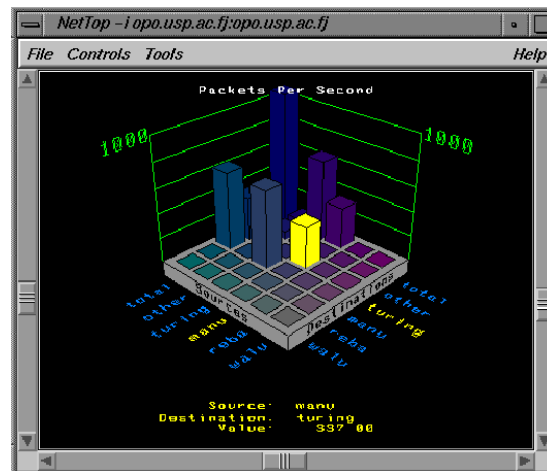


CS3283 - Hugh Anderson's notes.

Page number: 91



3D graph

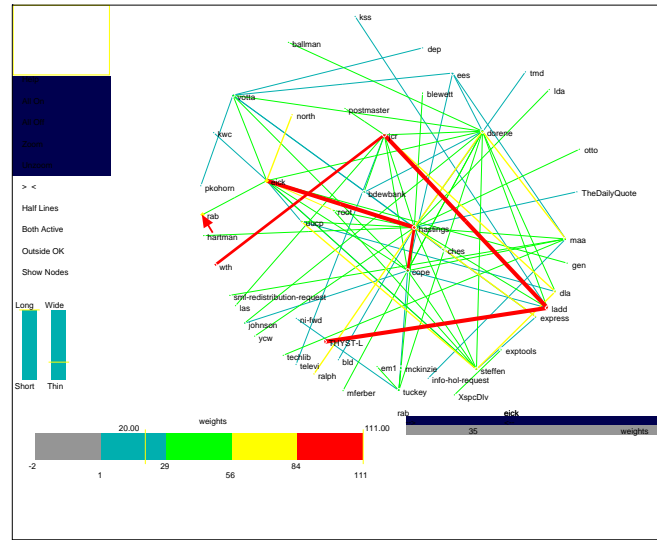


CS3283 - Hugh Anderson's notes.

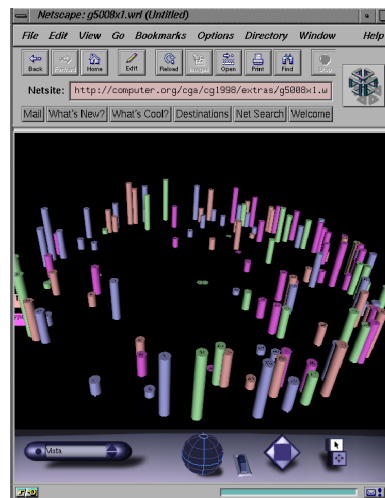
Page number: 92



Abstract 3D view - SeeNet

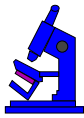
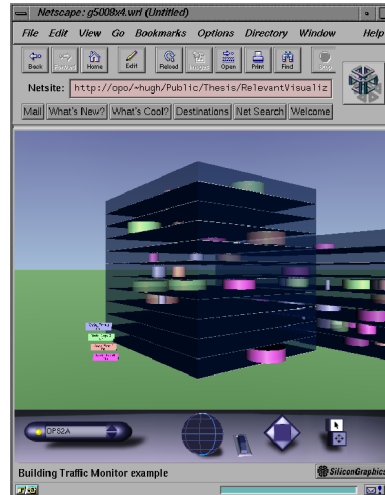


Abstract 3D view - Flodar





3D world-view



Summary of topics



In this module, we introduced the following topics:

- The designer's mindset
- Specification and design, tools and methods
- Examples of successful designs