LAN Bridge Spanning Tree Animation
Final Year Dissertation
Deliverable two

Simon Arlott
Supervisor: Peter King

March 17, 2006

Contents

1 Overview **2**

2 Aims and Objectives **3**
- 2.1 Network configuration . . . 3
- 2.2 Simulation . . . 3
- 2.3 Display . . . 4
- 2.4 Timetable change . . . 4

3 Functional Specification **5**
- 3.1 Network Configuration . . . 5
- 3.2 Simulation . . . 6
- 3.3 Display . . . 7

4 Design **9**
- 4.1 User interface . . . 9
  - 4.1.1 Layout edit mode . . . 9
  - 4.1.2 Scripting mode . . . 10
  - 4.1.3 Simulation mode . . . 11
  - 4.1.4 Display . . . 12
  - 4.1.5 Help . . . 12
- 4.2 Internal data structures . . . 12
- 4.3 Simulation . . . 14

5 Preliminary Implementation **15**

6 Testing Strategy **18**
- 6.1 Network Configuration . . . 18
- 6.2 Simulation . . . 18
- 6.3 Display . . . 19

Chapter 1

Overview

The IEEE 802 committee has developed a series of algorithms for enabling bridges to be placed between LANs without the stations being aware of it. If the network formed has loops, the bridges run an election protocol to ensure that only sufficient bridges to form a spanning tree remain active. This project is to develop a system for specifying a configuration of LANs, simulating the operation of the spanning tree algorithm, and displaying the simulation in real-time using an interactive graphical environment.

Chapter 2

Aims and Objectives

Networks use bridges to forward packets transparently from one part of the network to another, allowing a large network to be split up into sections improving performance and efficiency. For redundancy there will usually be multiple physical paths that can be used and when these create a loop the potential for broadcast packets to repeat indefinitely occurs. The spanning tree algorithm was developed to identify a suitable loop-free network and enable forwarding on only the bridge ports required to implement it.

The aim of this project is to create a program to set up a network configuration and run a simulation of the spanning tree algorithm on it to demonstrate what the algorithm does, and the status of the bridges and network when the algorithm is running.

2.1 Network configuration

A graphical interface will be developed allowing the configuration to be created, edited, and displayed. Since the algorithm will be demonstrated with the same configuration multiple times a feature to save and reload will be provided.

2.2 Simulation

The speed of the simulation will be variable and can be stopped and resumed at any time. It will be interactive, allowing the current status of any part of the network to be obtained and modified to see how the algorithm reacts to changes. Automated operation must also be provided to allow the simulation to run through certain situations on its own.

2.3 Display

The current spanning tree that has been produced from running the algorithm should be shown on the display. When displaying the simulation, packets between one part of the network and another should be shown to add animation to the system and to indicate where traffic is flowing. The program should be easy to use, without being too simple and unable to show enough detailed information at the same time.

2.4 Timetable change

The timetable for the implementation has changed, the network configuration implementation will be completed much later than planned, partly due to the change of network component placement method (Chapter 5). This will not affect the outcome of the project adversely since the time allocated for implementation was originally specified to be more than was required. The simulation and display implementation is therefore also moved to be finished two weeks later than planned, but still early enough to allow for thorough testing.

Deadline	**Activity/Deliverable**
31st March	Network configuration requirements implemented
5th April	Simulation and Display requirements implemented
7th April	Software ready for user testing

Chapter 3

Functional Specification

3.1 Network Configuration

Loading and saving

The network layout and current configuration values are kept by the program while it is running (Section 4.2), this information can be reset to a blank network after prompting the user to save it. Saving the network involves writing the details of the network to a file. The network can be loaded from or reverted to a saved file. Loading or reverting the network will reset the simulation.

Graphical edit interface

The LANs, hosts and bridges in the network are shown in the same way as used for the simulation. A grid to which the mouse pointer snaps is used for positioning components and simplifying the layout. New network components can be created by selecting the type of component and then clicking where to start and finish placing it.

Existing components can be moved with differing freedom depending on the type; bridges, and LANs with attached hosts can be moved as a single group while hosts can be moved along the attached LAN or relocated to a different LAN. Components can also be deleted, removing them from the simulation.

Components with configurable settings such as hosts and bridges can be selected which will update a list of properties showing the current settings and allowing them to be changed.

Ability to reconfigure network while the simulation is paused

While the simulation is paused it is possible to modify the network without affecting the simulation unnecessarily – moved components will still have any packets in transit at the new location, components extended in length will not remove packets currently moving on them. Deleting components will result in packets on those components being lost. New LANs can be connected to bridge ports which will be detected by the simulation when it is resumed.

Bridge identification

Each bridge and bridge port's ID can be set while the simulation is paused or running using the same method as other network component settings, but any changes will only have an effect the next time the spanning tree algorithm is run.

Bridge parameters

The bridge parameters for the spanning tree algorithm can be changed or set to suggested defaults at any time.

3.2 Simulation

Variable speed

The speed of the simulation can be changed using a standard slider interface.

Pause function

A pause button will pause or resume the running simulation. While paused the current state of the bridges can be inspected and modified without additional events occurring. The network editing mode will automatically pause the simulation.

Stepping function

To allow the simulation to be run incrementally the paused simulation can be run until a significant event, such as a new bridge configuration packet being received that changes the state of a bridge.

Automation using scripting

The scripting system is able to define a list of states the simulation should run through, optionally being made to pause at each state and allowing network component settings to be changed when each state is reached and the simulation is started or resumed.

Random packet sending state for hosts

An option in the configuration of each host will enable or disable the random sending of packets to other hosts known to be reachable at some point on the network. This can be changed while the network is being edited and also when the simulation is running.

Trigger a packet from a host to another host

Individual packets to other hosts can be triggered from the options available while the simulation is running when a host is selected.

3.3 Display

Show/modify current status of a bridge

When a bridge is selected the current status is shown and configurable settings can be changed.

View log of a bridge's status

The most recent state changes are logged and shown when the bridge is selected.

Visual indication of packets

Packets are shown as coloured lines moving along the LAN and host connections, at intersections they will be doubled to reflect the broadcast nature of the network.

Spanning tree

The current spanning tree set up by the algorithm is shown by changing the colour of the connections between LANs and bridge ports to grey instead of black when they are not used in the tree.

Control hosts

Selecting a host will show a button to make it send a packet to a specified host and an option to enable or disable the random packet sending feature.

Control simulation

The simulation can be paused or resumed and also run incrementally using the stepping function through toolbar buttons.

Chapter 4

Design

The program will be implemented in Java and use the Swing API for the graphical user interface.

4.1 User interface

The user is presented with a menu for accessing the file-related commands and help, and a toolbar for the simulation functions. Three modes are available for working with the network, the simulation mode (Figure 4.3), the layout edit mode (Figure 4.1), and the scripting mode (Figure 4.2). Toolbar buttons for changing to the edit and scripting modes will pause the simulation (if running) and change to that mode.

In both modes the mouse pointer will snap to positions splitting up the available space into 10 by 10 pixel squares. In the edit mode a grid will be displayed underneath the network showing where the mouse pointer will snap to, helping with alignment of components.

For the layout edit and simulation modes a panel interface for modifying selected network components will be shown at the side of the window (optionally on the left or right). In the scripting mode this will show the list of states.

4.1.1 Layout edit mode

Additional toolbar buttons will be displayed which will facilitate switching between four options: selecting components, adding LANs, adding hosts, adding bridges. The configuration panel will change to be oriented towards changing network settings instead of showing the current simulated network state.

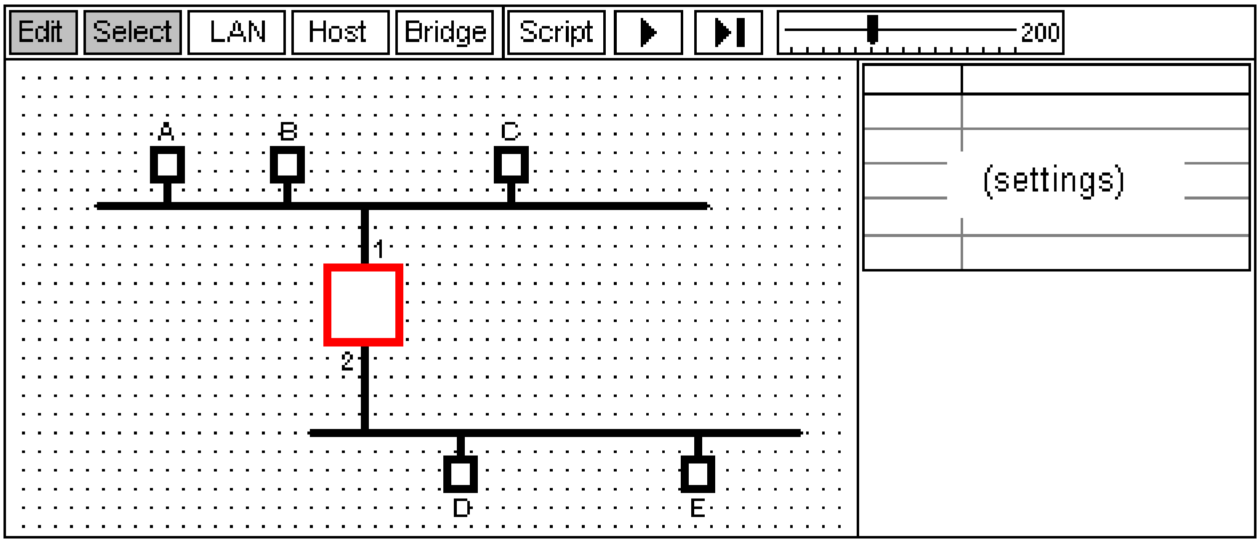

Figure 4.1: Layout edit mode

To make the interface simpler to use for modifying the network layout, the creating and moving of components will be available using only the mouse buttons without the need for using the toolbar to select an action as well as a component type.

Clicking with the primary mouse button will initiate placement of a new component and clicking with the secondary mouse button while hovering over an existing component will initiate moving components. While placing a component the primary mouse button will finish and the secondary mouse button will cancel the operation. For movement the primary mouse button will move the component to its new position and the secondary mouse button will cancel the operation. Pressing the *delete* key on the keyboard while a component is selected for moving will delete that component (and any connected hosts in the case of LANs).

When selecting components the primary mouse button will change the currently selected component to the one under the mouse pointer and the configuration panel will change to show the settings for that component and allow them to be changed. For convenience moving and deleting will also work in this mode.

4.1.2 Scripting mode

In this mode the states, component setting changes and the option to pause when the state is reached will be shown as a list in the configuration panel. The network will still be visible to allow

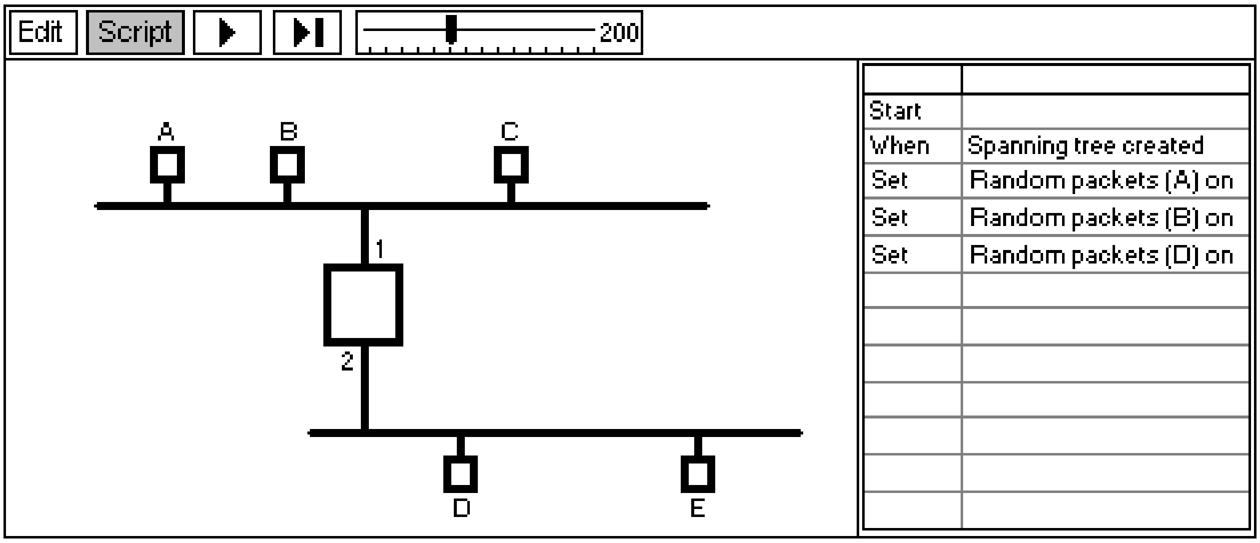

Figure 4.2: Scripting mode

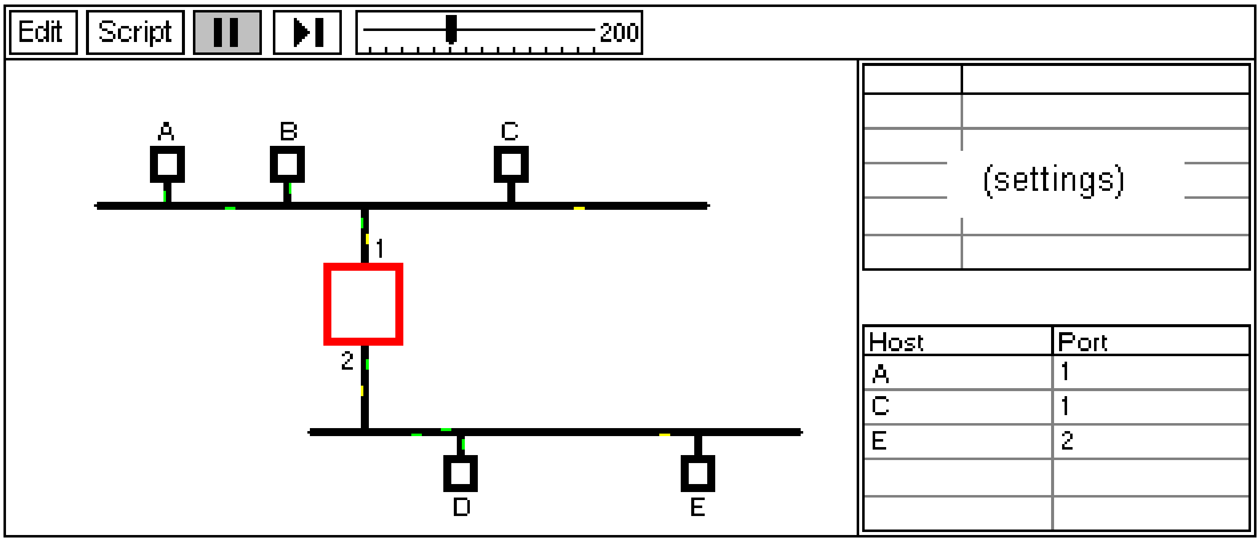

Figure 4.3: Simulation mode

components whose settings should be changed to be selected instead of having to navigate through a drop-down list of all components.

4.1.3 Simulation mode

Toolbar buttons for pausing, resuming and stepping through one event or state change of the script will be used to control the simulation state. A slider will control the overall speed of the simulation.

When a component is selected using the primary mouse button the configuration panel will change to show the status, commands and simulation settings for that component. For hosts this will show buttons to send out a packet to a specified host and the option to enable random packet sending. For bridges the current spanning tree parameters (as set by the root bridge or from the default settings) will be shown, along with the current host address table and a log of recent bridge activity. An option to show the host address table as a separate on-top window or as part of the displayed network should be made available so that the host tables of multiple bridges can be shown at once.

4.1.4 Display

The display of the network will use thick lines for LANs and connections (to allow for an area for animated packets to be shown on), and boxes for the hosts and bridges with text labels to identify them. Packets will be shown as three different colours, one for host to host communication, one for bridge configuration message packets and another for topology change notification packets (Figure 4.3).

4.1.5 Help

Context-sensitive online help will be provided through the use of tool-tips, a link to the user manual will be available from the help menu.

4.2 Internal data structures

The network components will be stored internally as objects linked to each other (Figure 4.5) as well as with graphical co-ordinates (Figure 4.4) to allow the simulation to be run on it without complex detection of connected components and the display process to be as simple as possible for better performance. This linking process will be carried out at the point of adding it to the network layout using the edit mode, or when loading the data from a file.

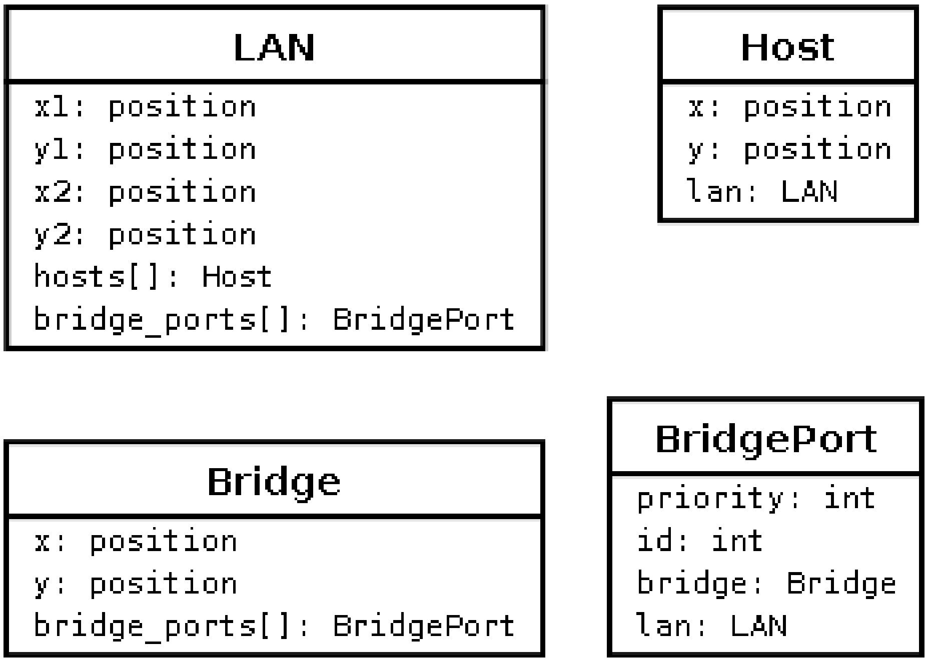

Figure 4.4: Objects used to represent network components

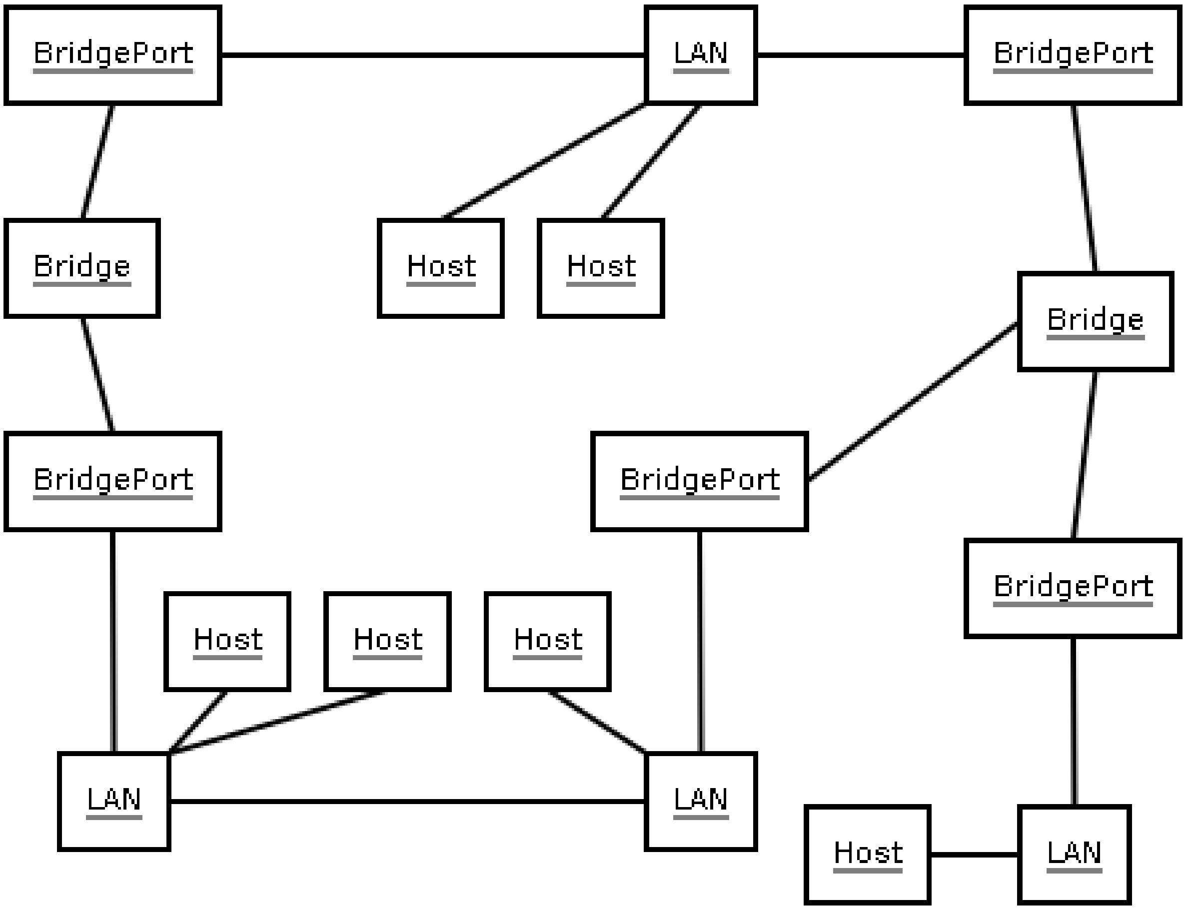

Figure 4.5: Internal data structure of network components

4.3 Simulation

The simulation process will keep track of all the packets on the network and run in a loop at the speed set by the user. With each iteration of the loop each packet will be moved along the LAN duplicating it at every connection (a host, another LAN or a bridge port) until it has nowhere else to go and is discarded. The LANs which receive the duplicate packet send it along in the same way. Hosts and bridges will receive their copy of the packet and process it, with the bridges forwarding it to another LAN or generating new packets if necessary. Bridge timers will also be checked each time the loop is run and react as required by the spanning tree protocol.

Information about the current state of the algorithm will be kept in each bridge object, instead of globally, although information about the global state will be available for the scripting mode. When running from a script the simulation will check it after each loop iteration to find out if the simulation should stop or settings should be changed.

Chapter 5

Preliminary Implementation

Loading/saving networks

Editing

The network components can be created using the mouse, by clicking to start placing and then again to end – Figures 5.1, 5.2.

A network of LANs and hosts can be created and loaded from or saved to a file using the menu options **Open** and **Save** which display standard file selection dialog boxes.

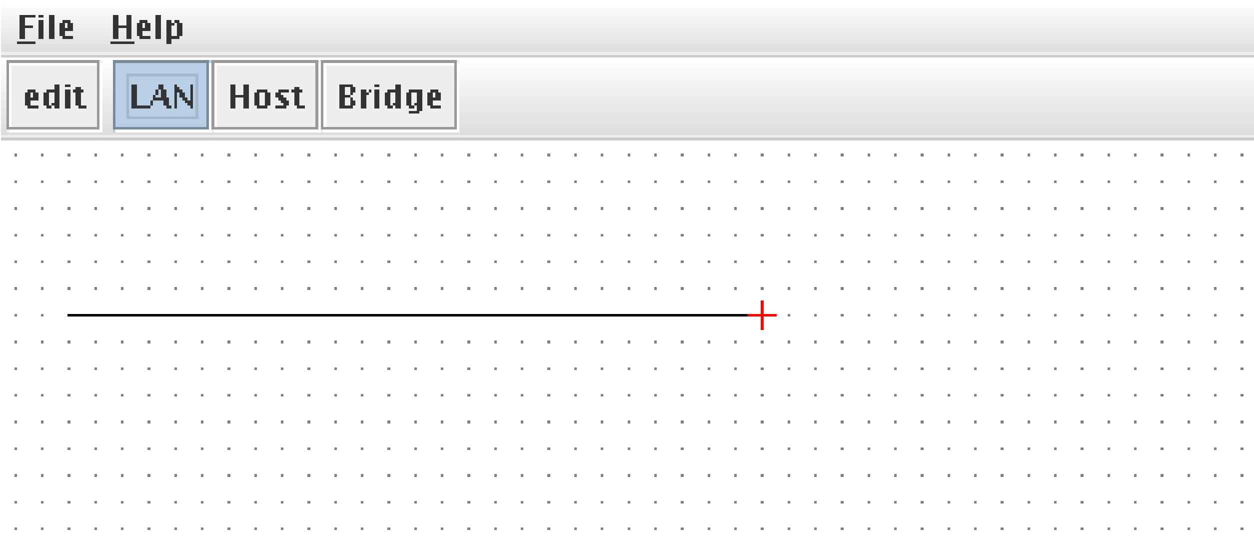

Figure 5.1: Creating a LAN

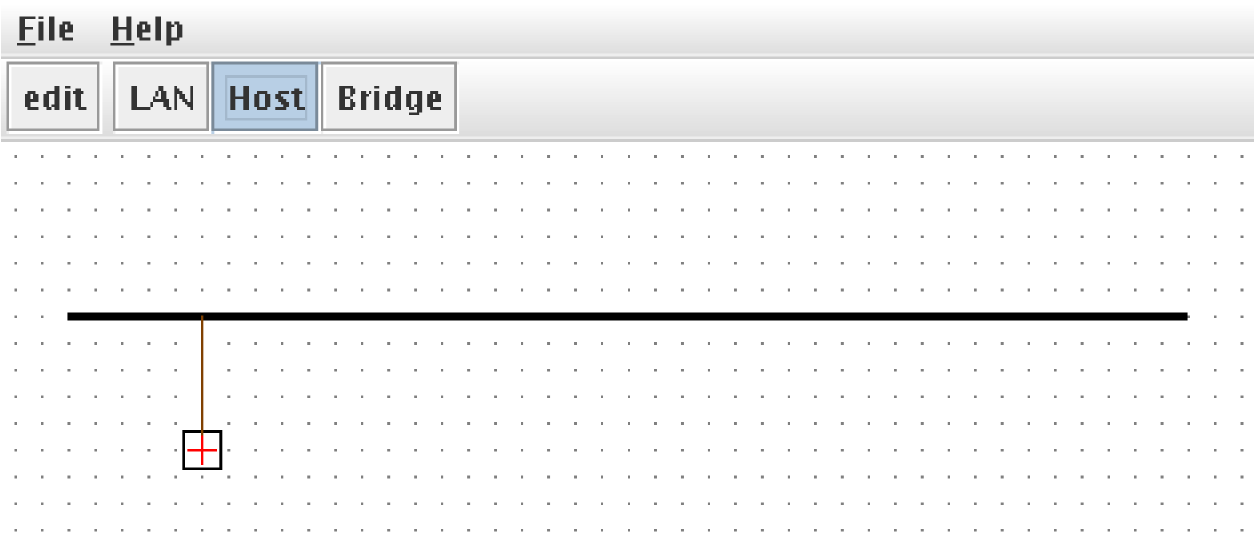

Figure 5.2: Creating a host

Data file format

The information about the network is stored in XML form for easy manual editing:

```
<?xml version="1.0" encoding="UTF-8"?><lbsta><network>
<lan id="0" x1="255" x2="255" y1="75" y2="225"/>
<lan id="1" x1="125" x2="255" y1="225" y2="225"/>
<lan id="2" x1="255" x2="465" y1="225" y2="225"/>
<lan id="3" x1="25" x2="255" y1="75" y2="75"/>
<lan id="4" x1="255" x2="485" y1="75" y2="75"/>
<host lan="3" x="85" y="115"/><host lan="3" x="165" y="115"/>
<host lan="3" x="215" y="15"/><host lan="0" x="315" y="125"/>
<host lan="0" x="315" y="165"/><host lan="4" x="365" y="35"/>
<host lan="1" x="195" y="285"/><host lan="2" x="375" y="285"/>
</network></lbsta>
```

Design

The first attempt at producing an interface for editing the network was a system allowing free placement of LANs with the mouse pointer snapping to appropriate positions on and in line with LANs to keep it connected properly and lined up. However this method proved to be quite complicated to implement and was not particularly easy to use so the current grid-based system was used instead. The circle in figure 5.3 is the current position snapped to based on the position of the mouse pointer and the thin line representing the parallel LAN.

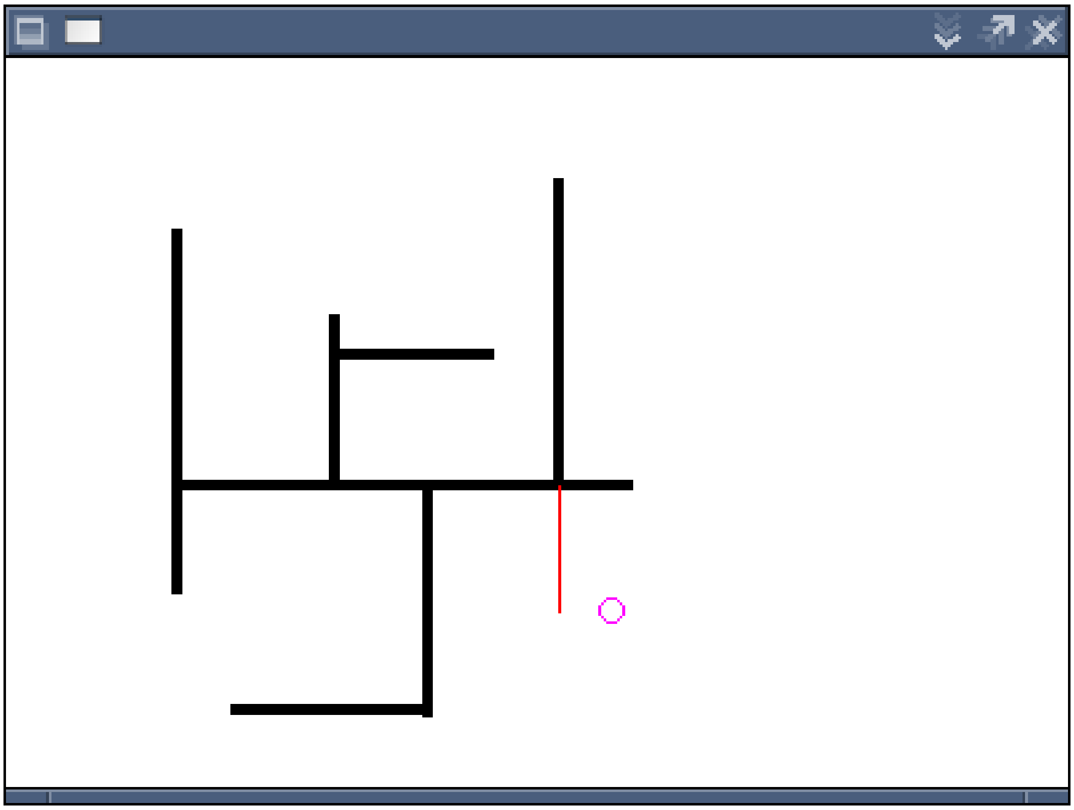

Figure 5.3: Freeform LAN placement with snap feature

Chapter 6

Testing Strategy

6.1 Network Configuration

The network configuration testing will involve ensuring that loops involving only LANs cannot be created at any time, and checking that packets are handled correctly if the network is modified while the simulation is running (but paused). It is also important to test the ability of the program to avoid ambiguity in the network by not allowing network components to overlap, otherwise the displayed behaviour could differ from the expected one.

The ease of use of the program must be evaluated by developing a list of tasks the user would want to perform when creating and modifying a network, and examining how this would be done without having any prior knowledge of the software.

6.2 Simulation

Implementation of the spanning tree algorithm

The simulation will be tested by creating a set of diverse test networks and running them with debug output recording the result to check that it responds correctly in all possible situations that the algorithm was designed to handle such as bridge ports connected to the same LAN, varying spanning tree algorithm settings on each bridge, multiple bridge failures or hosts moving LANs to ensure the simulation will run as expected.

User interface

The control of the simulation will be tested to ensure it behaves as expected, with no undesired behaviour resulting from pausing and running the simulation, or from using the stepping function in conjunction with the pause function.

6.3 Display

Testing of the display will be an evaluation of how well the displayed network components and packet animation serve the aims and objectives of the program.