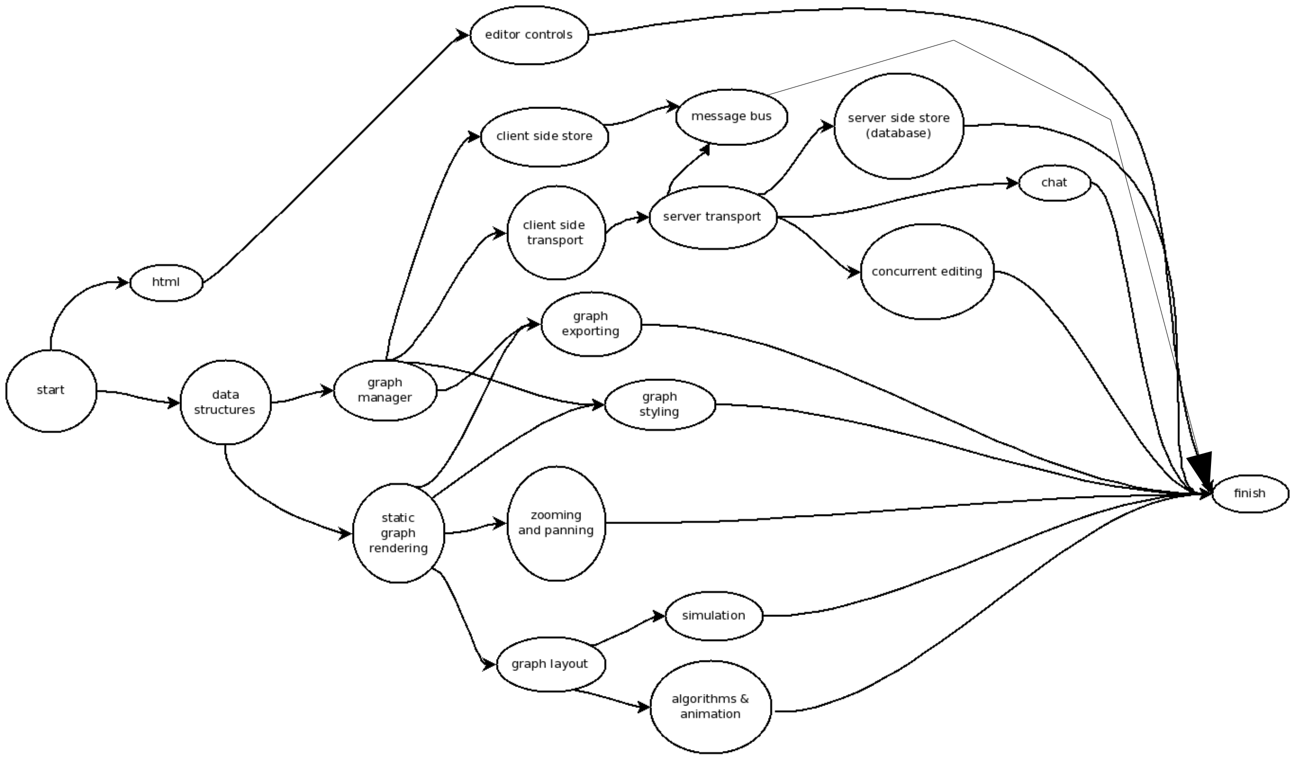

editor controls
message bus
client side store
server side store
(database)
chat
client side
transport
server transport
html
concurrent editing
graph
exporting
start
data
structures
graph
manager
graph
styling
finish
static
graph
rendering
zooming
and panning
simulation
graph layout
algorithms &
animation

Section	Module Name	Responsibility	Description
Editor GUI	high level html + framework	John	Provides a coherent, consistent, and aesthetically pleasing GUI with all the buttons and controls required to link to the "Graph editor controls" module and make graph creation and editing as intuitive and quick as possible.
	Graph editor controls	Ed + Ben	Connects the high level user interface buttons to the Graph Manager. Essentially changes what the mouse function does when it acts upon the canvas (fires an event).
	Algorithm	Everyone	A collection of algorithms that can be run on graph data to perform tasks such as shortest path finding and travelling salesman
	Styling	John	Allows customised styling of graphs, such as colouring, node (vertex) size and edge (line) thickness.
	Simulation	John,Ben	Simulates the execution of automata. Communicates with the animation module so the current state can be visually indicated.
Rendering	Static Graph	Mike and Max	Manages the rendering of a static graph, takes data from the Graph Manager and represents it. Renders vertices and edges using style data from the style module.
	Zooming/Panning	Mike	Module responsible for maintaining the current state of zoom and/or pan of the graph viewer. So other graph modules can refer to.
	Animation	Ben	Module responsible for maintaining the current state of an animation if necessary and contains details for possible animations.
	Layout	Ed	Module is responsible for the representation of the graph. It is responsible for simple things such as the layout and type of graph but also more complex operations such as optimisation, restructuring the graph in the most efficient manner
Graph Manager	Manager	Sam	Manages internal representation of graphs and provides data to the rendering layer
	Data Structure	Sam	Simple data structures for representing graphs, vertices and edges and data structures for drawing them
Communication Layer	Server Side	Ed	A serversocket which listens for incoming connections and passes each one off to a handler
	Local	Max	A websocket which binds to the server remotely, allowing for fast bidirectional communication between the server and client
Server	Database Storage	Max,Sam	Part of the back end of the server, this database will hold the graph information that is imputed from the user. It stores information such as lists of vertices, edges, type and the current form of styling used, and an id linking it to a specific user.
	Real Time Editing	Max,Sam	This section is responsible for mediating the interaction between multiple users and the information about the graph. By storing changes as transformations and processing all incoming requests, it will allow all copies of the graph to maintain an identical state even when changed simultaneously
	Online Chat	Ed	A simple chat interface designed to be implemented alongside real time collaboration in order to facilitate easy communication between the editors. It will just display imputed text to all users connected to the current graph.
	Message Bus	Sam	A bus over which all modules in the frontend can communicate. This will allow greater flexibility on the frontend. Communication layer will communicate with this
	Local Store	Mike	Module responsible for enabling the application to work offline by copying all the graph information from the server backend.
	Image Recognition	Everyone	Subject to time remaining after other features are implemented (Extension). This module will incorporate several features: 1. The scribble pad, the user will be able to use a freehand drawing tool to draw a rough representation of the graph, this drawing will then be analysed and transformed into an optimised graph. 2. Map reading, user inputs some map which will then be analysed and presented as a graph (similar to the map of the London underground)
	Exporting Graphs	Mike	Module responsible for generating exported graphs (either as an image or a vector graph). It can export the current view of graph, the entire graph, the current state of animation or a series of states of animation.

	Thing	Time		Depends	Earliest Start	Earliest finish	Latest Start	Latest Finish	Float (week)
1	high level html + framework	1	29th november	None	10/8/2010	10/15/2010	11/22/2010	11/29/2010	6
2	Graph editor controls	2	29th november	1	10/15/2010	10/29/2010	10/15/2010	10/29/2010	0
3	Algorithms	8	1st march	9	12/17/2010	2/11/2011	1/4/2011	3/1/2011	3
4	Styling	7	21st january	10,6	10/29/2010	12/17/2010	12/3/2010	1/21/2011	5
5	Simulation	9	1st march	9	12/17/2010	2/18/2011	12/28/2010	3/1/2011	2
6	Static Graph	2	29th November	11	10/15/2010	10/29/2010	10/15/2010	10/29/2010	0
7	Zooming/Panning	7	16th january	6	10/15/2010	12/3/2010	11/28/2010	1/16/2011	6
8	Animation	7	1st march	6	10/29/2010	12/17/2010	1/11/2011	3/1/2011	11
9	Layout	7	21st january	6	10/29/2010	12/17/2010	12/3/2010	1/21/2011	5
10	Manager	2	29th november	11	10/15/2010	10/29/2010	10/15/2010	10/29/2010	0
11	Data Structure	1	29th november	None	10/8/2010	10/15/2010	11/22/2010	11/29/2010	6
12	Server Transport	2	16th january	13	11/26/2010	12/10/2010	1/2/2011	1/16/2011	5
13	Local Transport	4	16th january	10	10/29/2010	11/26/2010	12/19/2010	1/16/2011	7
14	Database Storage	5	16th january	12	12/10/2010	1/14/2011	12/22/2010	1/26/2011	2
15	Concurrent Editing	11	1st march	12	12/10/2010	2/25/2011	12/14/2010	3/1/2011	1
16	Online Chat	6	1st march	12	12/10/2010	1/21/2011	1/18/2011	3/1/2011	6
17	Message Bus	6	21st january	12,18	11/26/2010	1/7/2011	12/10/2010	1/21/2011	2
18	Local Store	6	1/21/2010	10	10/29/2010	12/10/2010	12/10/2010	1/21/2011	6
19	Image Recognition	18	4th march	10	10/15/2010	2/18/2011	10/29/2010	3/4/2011	2
20	Exporting Graphs	7	28th january	10,6	10/29/2010	12/17/2010	12/10/2010	1/28/2011	6

Milestones:
editor, the editor and drawing functionality - 29th november
simple algos, simple algorithms that can operate on the graph - 29th november
integration, integration between the already existing components - 2nd december
alpha-feature-freeze, feature freeze before the deadline for bug testing - 3rd december
alpha, the alpha release deadline - 6th december
transport, server and client side transport as well as server and client side storage - 16th january
styling different graphs, flow charts, class diagrams ect, the entire styling system - 21st january
integration, integration between the already existing components - 25th january
beta-feature-freeze, feature freeze before the deadline for bug testing - 28th january
beta, the beta release - 31 january
simulation, simulation of DFA, DPDA, turing machine ect - 1st march
layout, algorithms for automatically layouting graphs - 1st march
concurrency, real time editing between multiple users - 1st march
integration, integration between already existing components - 3rd march
gamma-feature-freeze - 4th march
gamma - 7th march
report - 2nd may