Software Design and Architectures
SE-2 / SE426 / CS446 / ECE426
Fall 2003

Assignment 2 : Modeling Software

This assignment is due in class (1pm RCH 205) on Monday September 22. Each student should submit his own work, not that of a group.

Draw four simple models of a word-processing desktop application. Include a model from each of the four categories of "view" (logical/data, process/functional/behavioural, physical/behavioural, and constructional/development). Include at least one black-box and one white-box model. Use a graphical representation for each model.

Explain how each of the following scenarios operates in each of your models. (Some scenarios may be unrelated or "invisible" in some models.)

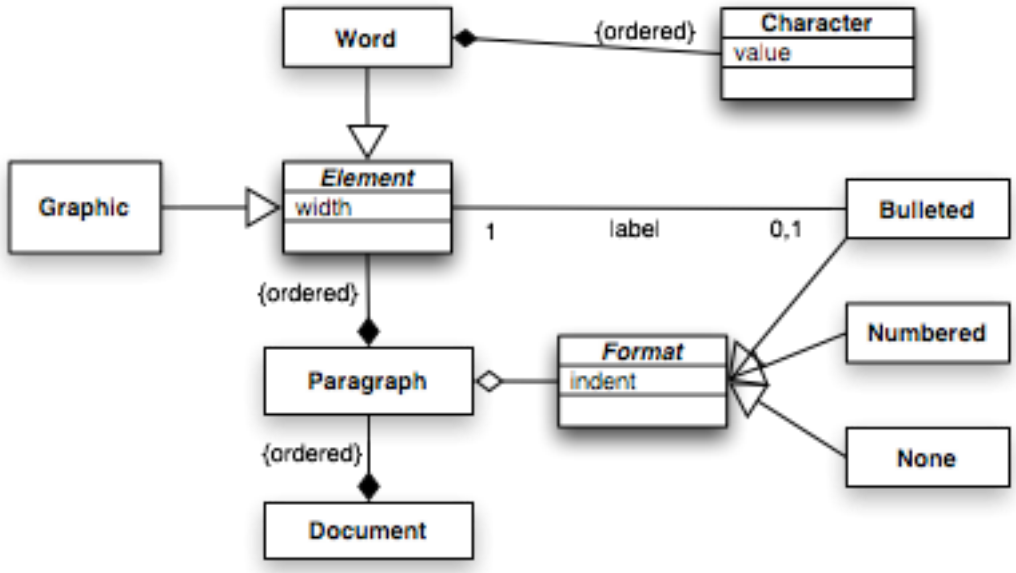
- Create a bulleted list from a selection of paragraphs in the text.
- Print an existing document.
- Install a new version of the application.
- Build the application from source

Since you have four models and four scenarios, there will be sixteen explanations.

The emphasis here is on breadth not depth. Don't try to make your models complete.

Sample Solution
I should have marked the scenarios for future reference. Let's call them (a), (b), (c), and (d).

1. Here is a class diagram for some part of a document involving paragraphs and formats.



In this model, for scenario (a) the selection of paragraphs would be assigned a new Bulleted Format,

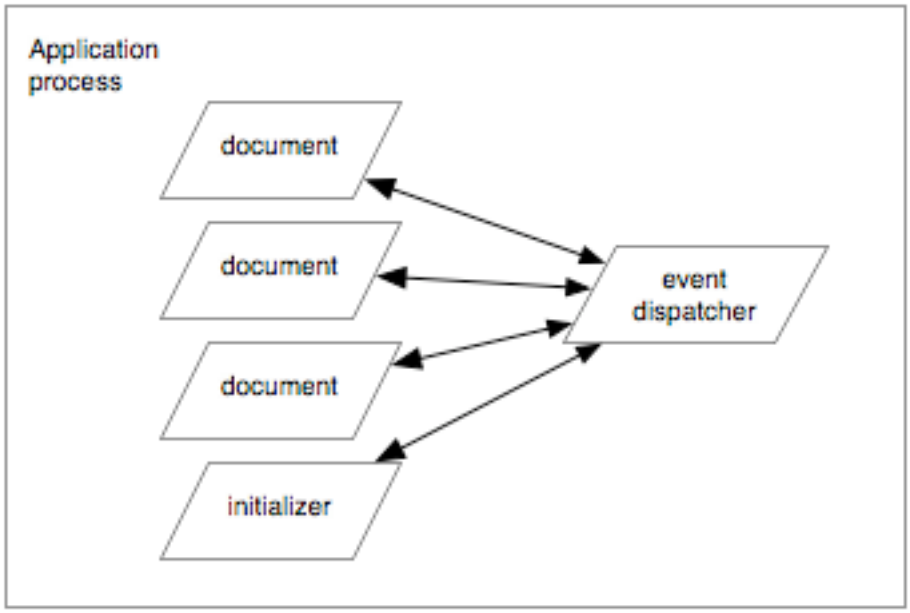
each having a label which was a Graphic Element that was a bullet symbol.

For scenario (b), a print method or procedure (not shown) would walk the structure of the Document (which is a sequence of Paragraphs) and the structure of each Paragraph (being a sequence of Elements), and for each Element obtain a printable image. (This is greatly simplified because there is a great deal omitted from this model.)

For scenario (c) there is not much to say. A new version of the application might change the logcal structure (in which case we would be concerned about access to legacy documents stored in the old logical format).

For scenario (d) there is nothing to say about this model.

2. Word processing applications typically have no major-task multi-processing, so there isn't much point modeling that. However, they often have substantial (i.e. long) initialization tasks (loading fonts, plugins, environment stuff, *etc.*) which shouldn't interrupt the starting of the GUI. In addition there *may* be facilities which require to be run in the background because they take so long: for example, reading and writing large documents across the network. Here is a multithreaded model which allows initialization and each document to be processed on its own thread so that the user can start work on a document before initialization has finished or while another document is loading or saving. (Typical GUI managers, like Swing or GTK, are not designed to be thread-safe all the way down, so the dispatching of events and the receiving of graphics operations must be done on a single “UI thread”.



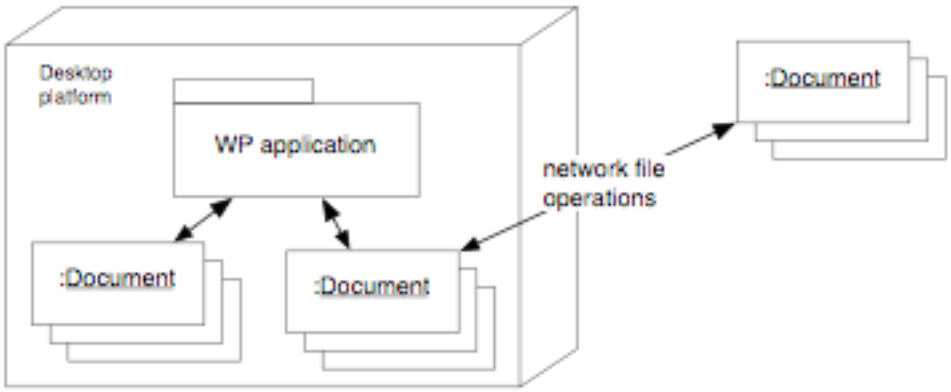
For scenario (a) the event dispatcher would send the user's request to the appropriate document thread. (Changes would be sent to the presentation layer, not shown.) Of course the operations in scenario (a) would take place so quickly that multithreading would have no effect whatever on performance.

For scenario (c) nothing can be said about this model, assuming that the application must be not running when a new version is installed.

For scenario (d) there is nothign to say about this model.

Other appropriate models might include the modal state of parts of the user interface, or data flow at a lower level (*e.g.* translating the document structure into Postscript or PDF for printing).

3. There is really very little to model in the physical domain for a desktop application. Applications run on a single machine and the documents are stored in files in the file system. In other words, this (boring) picture (arrows represent data flow).



For scenario (a), a document may be expected to be already in memory, and so all the activity related to the creation of a bulleted list takes place within the application package.

For scenario (b), the same remarks are true as above: the printer software and hardware are not shown.

For scenarios (c) and (d), there is nothing to say about this model.