WORKING NOTE AC76

Alan Creak
December 1989

THREE HYPERFORTY :

The structure of a course in operating systems.

By making certain assumptions about desirable characteristics of a university lecture course, I was led to propose an unusual course structure for the stage 3 operating systems course[1]. This note continues the argument with a proposal for a way of representing the course structure explicitly.

There are (at least) three reasons, none of them as securely based as I'd like, why this explicit structuring might be a good idea.

1. It will be good for the lecturers concerned to analyse the course. It will give us a better understanding of how the parts fit together, and a better basis for assessing the value of different topics in the course. I have argued elsewhere[2] that our courses should be subjected to such scrutiny as a part of efforts to design our overall course of study more carefully, and as a means of keeping individual courses up to date.

2. It will be good for the students. A clear understanding of the way in which different topics treated in the course fit together makes it much easier to comprehend the course, and to make sense of its subject matter. I have not found it easy to make much sense of operating systems given the traditional, bottom-up, approach; I would like to believe that we can do a much better job of presenting a coherent subject with the new, top-down, approach, and this cannot but benefit the students.

3. It will give me an excuse to get to grips with Hypercard[3], or something of the sort. Keeping up to date with new technology is not one of my stronger points, and it would be nice to do it once – if only to convince myself that it isn't worth the effort.

HOW TO DO IT.

The structure of the course, as I see it at the moment, manifests itself in two ways. To begin with, there is the conventional tree structure : a root node representing the topic of operating systems is linked to children representing its component subtopics (perhaps processor management, memory management, file systems, and so on, to take a traditional view of the course). This development is then continued recursively until the "leaf" topics are reached. This analysis is traditional, and so far makes no distinction between the "old" and "new" views.

That comes with the second form of structure. I see this as represented by "horizontal" links in the tree, typically connecting requirements identified in one subtree with ways of satisying them in other subtrees. For example, the requirement of confidentiality identified while analysing people's requirements for data storage is linked to the topics of security in data transmission, protection and security in disc file systems, and protection in memory management. It is the insistence that the requirements links in this tree should go forward in time that dictates the "new" approach, and imposes the top-down order on the subject matter.

Other links might be useful. Obvious ones are to textbook references, other literature references, and distributed handouts. I do not know whether it would be feasible to attempt to build all the documentation, handouts and all, into the system : I suspect not, but a system which could in principle expand to include this material would be useful.

A piece of software which appears to offer the facilities required is Hypercard. Perhaps there are others, but that's the only one living on my computer just now. I know that isn't a very good reason for using it, but I doubt if I can do much better at the moment, because I'm not yet sure just what I want to do. At least Hypercard will let me do some experiments, and clarify the issues involved.

REFERENCES.

1 : G.A. Creak : *Operating systems course structure*, Unpublished Working Note AC75 (4 December 1989).

2 : G.A. Creak : *Computing – a course structure*, Unpublished Working Note AC73 (15 November 1989).

3 : *Hypercard User's Guide*, Apple Computer Inc., 1988.