WORKING NOTE AC119

Alan Creak
10 December 1997

GOING BACK AGAIN

In this note I describe the operations involved in reproducing the mark-up files used in constructing the department's handbooks from the final versions of the handbooks themselves.

WHY THERE IS A QUESTION.

An important part of data management for the handbooks (or anything else, for that matter, but here I'm discussing the handbooks) is to make sure that all relevant files are maintained in a consistent state, and that amendments made in one place are correctly distributed through the system to wherever they are supposed to be. That's obvious.

It is less obvious that the amendments happen at different times. I described the requirements in an earlier note[1], but some further changes turned out to be necessary even after the "Step 7" described there. It is noteworthy that these were, in several cases, not consequences of inadequate checking earlier, but of interactions between topics and broader implications which were not so obvious when the topics were handled separately. In some cases, considerable blocks of text were moved bodily, and new handbook sections were introduced; these changes were seen as desirable when the handbooks were read as a whole, and weren't obviously good ideas at the more fragmented level.

The result of incorporating these necessary amendments in 1997 was a final Word file for each handbook which differed in several respects from the data files from which it was originally composed. We had not made changes frivolously, and each represented some improvement on the first Word version, so it was obviously important to carry the amendments back into the data files somehow. If only because there wasn't time to do it any other way, I planned to carry out this operation manually, copying and pasting snippets here and there to bring the data files and handbook into correspondence.

Necessity apart, I had worked through most of the transformations required for the 1998 handbooks by hand (appropriately enough) with the intention of understanding the processes involved at first hand (ditto). I regard 1997 (by which I really mean the preparation of the 1998 handbooks) as a trial run, and my proposal to carry out this fairly final transaction manually for that year was reasonable enough – except that by the time the handbooks were safely in print and it was possible to meddle with the files without immediate deleterious consequences for the production I had forgotten which bits had been changed. I could still perform the operations by hand, doing it for the whole text rather than selected parts, but that was a far bigger task than I'd expected, and I didn't have time for that either. Suddenly it seemed to be a good time to consider possibilities for the future.

WHAT IS THE QUESTION ?

I speculated in the earlier note that automating this backward transfer might not be useful. Now with the experience gained through the year, I am persuaded that some automatic means of ensuring that the data files do really reflect the handbook contents precisely is necessary. That isn't quite the same statement, because it doesn't presuppose any particular implementation method, and there are a few ways in which the required end could in principle be achieved. For example :

- All editing changes to the handbooks could be made to the various source files rather than to the handbook Word files, and the Word files reconstructed each time. (Too late for 1997, but possible as a future policy.) As the reconstruction should not in fact be a very demanding task, this is not a silly idea, but a more serious consideration is that the changes are found by perusing the handbook text as it is in the Word version, and it's therefore a lot easier, and probably more reliable, to edit the Word files themselves. To go a step back is more likely to cause errors, as even now there are several different files involved, and it might not be clear just where the data come from in any particular case.

- Editing can be performed on the Word file, and the results then copied back into appropriate places in the original files where changes had occurred. The changes could be found by comparing before and after versions of the files, or possibly by exploiting Word's ability to record changes in some way. I didn't work out any details, because it seemed clear that if the editing operations were likely

to insert whole new sections of text then it might simply be impossible to pack the material back into the original form. In fact, this objection is probably overstated; unless the file structure depends on details as small as sections – which was sometimes the case with the files I first acquired, but is no longer – it is likely that means could be found, but by the time I worked that out I had become convinced by the advantages of the next possibility, and it didn't seem that this one would be as good even if it worked.

- Editing can be performed on the Word file, and new master files wholly reconstructed therefrom automatically. This avoids even the possibility of minor conflicts which might arise in trying to change the original files, and offers the considerable advantage of leaving old and new versions, which can be compared to check that differences have been properly handled. It also removes the need for keeping a record of all changes, which is a considerable advantage. On the other hand, if the files are rebuilt from scratch the possibility of using the original as a template is made much harder. Provided that the structure is kept very simple, though, there is little to gain from the previous copy, and this is unlikely to be a serious problem.

There is a limit to the possible reconstruction; we cannot reconstruct more from the Word files than the Word files contain. This constraint is significant if the material presented in the handbook is only a subset of the material in the source files. This does not cause trouble with the second proposal, as all the original information remains available in the original files, but in the last proposal this source of additional information is not available; a further operation is therefore necessary, in which the reconstituted partial files are merged with the real source files. By this time, though, the data are in comparatively simple mark-up form, and merging should be much simplified.

Despite the complication of the additional merge step, I decided to follow the third prescription. It has two big advantages : it separates the Word-to-mark-up and merging operations, and it gives great freedom in editing Word or mark-up files as the occasion demands. Provided that a working mark-up file distinct from the permanent files is used in preparing the handbook, repeated transformation back and forth between Word and mark-up forms is possible, and this flexibility could be valuable.

It did not seem necessary to be too pedantic about the details, and I decided that I didn't mind starting off the process with a little manual work by sorting out at least some of the parts of the data which originated from different data files. I concentrated mainly on the ordinary part of the handbook[2] and the Courses file. This left me with a number of problems to solve, which I describe below. First, though, some general points which have become clear during my efforts are worth recording.

GENERAL CONSIDERATIONS.

The primary difficulty in carrying out the backward transformation automatically is finding the information required to do it. It is all there in the nicely formatted mark-up files, but the field labels are lost in the Word form, and must be reconstructed somehow. It is clear enough to someone reading the handbook how to reformat the text, but structure which is obvious to me depends on my understanding what the handbook is about, and is not accessible to a programme.

I had expected that this year any attempt at automatic conversion would need more than a little help, because of the severe loss of information in a form easily accessible to a computer programme. In the event, I was surprised how little intervention was necessary. The main difficulties were with material set in tables, or with several levels of indenting, but most of the rest worked very smoothly. Nevertheless, there is no guarantee that this ease of transformation will persist, and for future robust effectiveness it would be wise to provide means for preserving the information in the Word files somehow – where "somehow" means something rather like "invisibly and indestructibly".

It is self-evident that we don't want formatting information to obtrude on the presentation of the handbook, so however it's done it mustn't affect the text, nor in any way hinder the freedom of composition of the text. Microsoft, in its less-than-infinite wisdom, has provided a possible means of achieving something like the desired end in a Word file by using "hidden text"; the character formatting machinery provides means for denoting any area of text as hidden. Such text is not displayed or printed unless you explicitly direct that it should be. By incorporating formatting information as hidden text, it seems possible that my requirements could be satisfied. Indeed, that was my original idea, and it lasted until I realised that it might be invisible, but is certainly not indestructible. I knew that because the files

which I inherited when I took over the job showed signs of hidden text, which appeared to have been inserted by some previous handbook curator in an attempt to combine printed and HTML versions. What remained showed very clearly how the hidden text could be mangled during editing unless appropriate care was taken, and it hadn't been.

Which brings us to the requirement for indestructibility. Whatever form the information takes, it should not be possible (or, at least, it should not be too easy) to destroy it by editing. Recall that the object of the exercise is to make it possible to edit the Word files in the final stages of handbook preparation, and then to ensure that changes made are transferred safely back to the long-term data files; editing will therefore happen, and the formatting information must be preserved. Clearly, that isn't possible in any absolute sense : if someone edits the handbook by deleting everything in it, some details are likely to be lost. A more plausible goal is to make sure that minor changes – insertions, deletions, small cut-and-paste operations, etc. – will be unlikely to cause trouble, and to make it easy to ensure that the correct information is supplied in cases of more extensive change.

My answer at the moment is to rely on paragraph styles to carry the information. These have the tremendous advantage that they are not localised in any obvious way, so they can't easily be deleted or garbled (the main difficulty with the hidden text), and simple changes to existing text make no difference. In fact, in Word the styles are localised, unobviously, in the end-of-paragraph markers, but if one of these is deleted the text format is likely to change, in which case it will be obvious that the style must be restored. Finally, text of a particular style can be specifically identified in searching and replacing operations, so the method does give usable formatting information.

An unrelated potential difficulty is that of information which originates in files other than the immediate sources for the handbooks. An example is the list of lecturers in the information on courses; this will eventually be derived from the Functionaries file. For the time being, at least, I do not propose to automate the feedback to the extent of such more remote sources. That isn't primarily because of the added difficulty; it's because the more remote files are more likely to be authoritative, and not to require amendments. These files are usually primary files from which other information is drawn, and are subject to their own revision methods.

In developing my preferred method, I therefore plan that there will be an initial stage in which the working source files are constructed from the real handbook source and such other files as are needed, after which further reference to the primary files will be unnecessary. For example, the lecturers' details can be constructed from the Functionaries file when initial requests for course information are distributed. In reconstructing the immediate sources from the handbooks, it will be necessary to reconstruct appropriate text for the lecturers field as it appears in the intermediate file, but it will not be checked further or automatically transmitted to the primary file.

A DIGRESSION TO VTA.

It is interesting (to me, anyway) that my pet technique of VTA[3] (Vocabulary Translation Analysis) offers clarification in some parts of this discussion. One of the features of VTA is an emphasis on how information is carried by different channels within a communication. In the case of the handbook, we might describe information channels in something like these terms :

Text : Obviously. All the information for people must appear in text form somewhere. This is not a source of difficulty provided that it is simply carried about from file to file; it can be troublesome if formatting depends on content.

Mark-up : All the formatting information in the main data files is encoded by mark-up symbols of one sort or another. (That's why I usually call the main data files the mark-up files.) The most prominent are the field markers like !!lecturers||, but other more localised markers (!!italicon||, for example) are used to identify local features requiring special treatment. Other markers might denote files to be inserted, URL definitions, and other structural or textual features. No such markers appear in the printed text; the problem of going back again is to identify what markers should be inserted, and where. HTML files also contain mark-up symbols, which is what you'd expect from a HyperText Mark-up Language.

Paragraph styles : These happen in the Word files. They determine the format of many aspects of a paragraph of text, except insofar as they are overridden by local changes such as character styles. Though strictly associated with the end-of-paragraph mark, they behave as though distributed throughout the paragraph – so additional text can be inserted, or material can be deleted, within the paragraph without changing the style. An interesting consequence of this organisation is that any forced change of paragraph style imposed on any component of the paragraph – by a replacement, for example – necessarily changes the style of the whole paragraph.

Character styles : These also happen in the Word files, and generally resemble the paragraph styles, but they are more commonly used to mark some localised text string for attention in some way. These styles are associated with individual characters, so local changes are possible. The usual convention for inserting new characters from the keyboard within a section formatted with a particular character style is to give each new character the same character style as its left-hand neighbour, so the "distributed" effect of the paragraph style is maintained to some degree, but text transferred by copying and pasting into the middle of such a section is likely to retain its original character style.

Hidden text : This is text which can be incorporated in a Word file with the particular property that it can be omitted when the text is displayed or printed. It is implemented as a special sort of character style, but is perhaps better thought of as a textual tag which can be attached to a particular point in the file and referred to when desired. An **annotation** is a special sort of hidden text which can be made rather more secure, but at the cost of making editing harder.

Layout : This is an essential information-carrying channel in the printed handbook and in the displayed world-wide web pages, but of itself it is not easily perceptible at all to the computer system. The use of paragraphs, white space, tabular form, and other devices is very important for easy human comprehension of the printed material; it carries information about grouping, emphasis, relationships between data, and the like. The object of the formatting process is to convert the mark-up symbols into layout so that the text can be understood by people.

The special property of the Layout channel is perhaps at the bottom of all the problems in this process of handbook organisation. Somehow the mechanically imperceptible layout information must be constructed from information carried in the other channels, which is fiddly but tedious rather than excessively hard, and then converted back again, which is the topic of this note.

THE COURSES FILE.

The Courses file is probably the easy one. It has lots of well defined structure, it is peppered with useful keywords, and its parts are clearly identified with fields and records in the file. It is also much more completely represented in the handbooks than some of the other files; all the courses turn up in one of the handbooks, and there is very little difference in content between handbook material and filed material.

Looking to the future, though, there is no guarantee that this simple state of affairs will continue. It is certain that the near-identity of Courses file and handbook material will not last, as I intend that the Courses file will contain more material about courses, aiming eventually at a single file covering all course-related material. Further, even within the descriptive bits changes are likely; the main files should record information about such things as HTML links which are useful in the HTML handbook but not in the printed version, and these could not be recovered from the Word files. (It was this sort of information which had been added to the Word files as hidden text, and then comprehensively shredded by subsequent editing.) It might also be that some of the current Courses material will vanish, as it is already held elsewhere – the list of lecturers and tutors is an obvious example which I have already mentioned.

The aim should therefore be to produce, first, a file similar to, but not intended to be identical with, the Courses file. I can imagine editorial aids which will compare the recovered file with the old Courses file and point out differences, leaving it to the human editor to decide which version should be adopted. That will be later, if ever. (Even more remote is the prospect of really automating almost the whole of the Courses file reconstruction; it seems likely that almost all the changes will be sufficiently simple to manage automatically, leaving only gross changes for editorial attention, but that's speculation.)

For the moment, then, the intention is to produce something very like the Courses file. In 1997, it will be essentially identical, but divergences are expected as time goes by. A second difference in 1997 is that the whole conversion, from formatted Word file to reborn Courses file, was carried out with a Word macro. This was a matter of expediency, but a better strategy for the long term is probably to reflect as far as possible the procedures used to construct the Word file[4], doing the minimum possible in Word to reconstruct the prepared text file, then to complete the conversion using the simpler text processing facilities which might some day materialise. While the programming was surprisingly easy using Word, the more symmetrical arrangement is likely to be much easier to maintain, as complementary, and simply related, changes can be made to two corresponding files rather than having to develop two complementary procedures in quite different contexts.

Here is a summary of the operations used :

Start at the beginning			
replace all –	*condition –*	*by –*	*Notes*
^p		‖2^p‖1	Make line ends visible.
‖1‖2			Forget empty lines.
‖1	Style Course heading	^p!!number‖^p	Split the first line into number –
^t	Style Course heading	^p^p!!title‖^p	- and title lines.
‖1Prerequisites:^t		^p!!prerequisites‖^p	Change each item title to the appropriate mark-up symbol.
‖1Restrictions:^t		^p!!restrictions‖^p	
‖1Assessment:^t		^p!!assessment‖^p	
‖1Where:^t		^p!!where‖^p	
‖1When:^t		^p!!when‖^p	
‖1Lecturers:^t		^p!!lecturers‖^p	
‖1Tutors:^t		^p!!tutors‖^p	
‖1Texts required:^t		^p!!texts required‖^p	
‖1Texts recommended:^t		^p!!texts recommended‖^p	
‖1Description:		^p!!description‖^p	
‖1Contents:		^p!!contents‖^p	
!!number‖^p‖2^p		^p	Tidying up.
‖1			
‖2			
^p^t		^p	
^p^p^p		^p^p	
^p^p^p		^p^p	
select all			
Normal style			
remove italic			

Some specific comments :

- The curious insertion of ‖1 and ‖2 as end-of-line markers is occasioned by the even more curious invisibility of the paragraph marks (Word for ends-of-lines) when replacing text using pattern matching. Perhaps there's a way, but the paragraph marks aren't listed in the special characters which one can use in patterns.

- The repeated replacement of three end-of-line characters by two towards the end of the programme was sufficient in the example; it means "any sequence of more than two end-of-line characters can be reduced to two".

- Further consideration should be given to the lists of texts. At present, I accept whatever format the functionaries give me, but the result is a mess. It would be sensible to define some standard style, and to use a corresponding mark-up notation in the files. Bibtek is perhaps the most obvious, unless I use my Croak format.

It is clear that almost all the required formatting can be managed using material found in the text alone. The only exception in the scheme above is the use of the Course heading style to identify the title line; this could almost be replaced by a text operation based on the appearance of "415." at the beginning of a paragraph. The style was retained because "415." is of fairly common occurrence in the files, and might well appear at the beginning of a paragraph by accident, but more because one course (the BTech project) has a different prefix.

Some other features should have been there to ensure correct operation, though they were not needed in 1997; obvious examples (which means the examples I can think of at the moment) are indications of character style, so that boldness and italicity can be preserved. (A means of carrying out the required transformations appears in the following description of converting the mark-up file.) Doubtless other omissions will come to light in the course of time, but the simple and clearly marked form makes this transformation very easy.

THE HANDBOOK MARK-UP FILE.

Large tracts of the handbook mark-up file are simple in structure, and pose few problems; this is what I have called the *ordinary* handbook material[2]. The features significant in the transformation are chapter headings, section headings, and tables (all identifiable by paragraph style), and character styles. Some of the fiddly details gave me some trouble, mainly in precisely identifying the strings which should be treated as units, but some of that was probably because of my unfamiliarity with the style of programming required for a Word macro.

In almost all cases, the required information was comparatively easy to extract from the Word file, and the rest was then straightforward. Perhaps the most difficult task was that of marking several consecutive entities with the same style as a single block; my general solution is described below. It's clumsy, but it works. This rather artificial step can probably be avoided when the Word macro is replaced by a combination of Word and text processing, as described above.

Word tables remain as a significant problem, and I have not yet had time to work out how best to deal with them. These are major constituents of the *extraordinary* handbook material[2], and major reasons for my deferring the treatment of this material for the time being. Clearly, they must be converted into text, and Word provides means of so doing. This only works reliably if the contents of each table cell are restricted to simple text, and if there are no clever, but very useful, tricks such as merging cells, and ends-of-lines appearing within cells. The problems are not (so far as I can see) difficult to solve, but they are tedious and messy. The obvious solutions – replacing all offending characters by mark-up symbols – are clumsy in the extreme, and lead to a very cluttered mark-up representation which is difficult to read, and therefore hard to edit with confidence. Investigations continue.

Here is a slightly edited version of the sequence of operations used to convert the file. The editing has (I assert) not changed the essential function of the programme, but has tidied it up a little. The original grew more by evolution than by careful design; in consequence, components did not always appear in a very comprehensible order, some operations were needlessly duplicated, and so on. Precise reproduction of the original is not very important, as, all being well, much of it will never be used again, but the set of operations required remains of interest for designing the replacement.

replace all –	*condition –*	*by –*	*Notes*
"^p "		"^p"	Clear awkward spaces.
" ^p"		"^p"	
"^p"		"||2^p||1"	Mark line ends.
"||2^p||1"	Style "Bulletlist"	"^p"	Identify, bracket, and mark bulleted lists.
"||1"	Style "Bulletlist"	"!!Bulletliston||^p||1"	

"||2"	Style "Bulletlist"	"||2^p!!Bulletlistoff||"	
"||1||2"		""	Eliminate line end markers for empty lines. (Keep the lines – they might be separators.)
"(||1*||2)"	PatternMatch Style "Chapter heading" → not bold	"!!chapter||^p\1^p"	Label chapter headings.
"(||1*||2)"	PatternMatch Style "Section heading" → not bold	"!!heading||^p\1^p^p!!body||"	Label section headings.
"||1"		""	Remove line end markers.
"||2"		""	
"(<*>)"	PatternMatch Italic → not italic	"||italicon\1||italicoff"	Identify, bracket, and mark italic strings.
"([/,.:;\(\)\-])"	PatternMatch Italic → not italic	"||italicon\1||italicoff"	
"||italicoff||italicon"		""	
"||italicon"		""	Use HTML brackets.
"||italicoff"		""	
"(<*>)"	PatternMatch Bold → not bold	"||boldon\1||boldoff"	Identify, bracket, and mark bold strings.
"([/,.:;\(\)\-])"	PatternMatch Bold → not bold	"||boldon\1||boldoff"	
"||boldoff||boldon"		""	
"||boldon"		""	Use HTML brackets.
"||boldoff"		""	
"^m"		"^p"	Remove page breaks.
"!!"		"^p!!"	Tidy layout
"||^p^p"		"||^p"	

Specific comments :

- End-of-line markers appear again.
- Chapter and section headings get special treatment. Notice that character formatting implied by the style (bold text in both cases) must be removed, or it will be dealt with again when isolated character styles are marked later on.
- The only significant special format was the bulleted list – and even this wasn't marked very well in the Word files, as arbitrary other styles had been used. Changing them to the single style *Bulletlist* was a great improvement. It seems likely that as the system improves more special formatting styles will be defined, though at present the printed handbook is rather low on layout.
- Word does not have structures to identify maximal contiguous blocks of text with the same character or paragraph styles. (That's reasonable enough, when you think about it.) Because of that, I had to indulge in some acrobatics to insert mark-up symbols bracketing such blocks – in effect, I insert symbols round the contiguous units which I can detect, and then eliminate internal bracket pairs. The units are lines – Word paragraphs – for paragraph styles, which is easy, but for character styles a neat solution is hard to find. The obvious unit is the character, and that works, but it's excruciatingly slow, as two mark-up symbols are inserted between every consecutive pair of characters of the style concerned, and then almost all are deleted. The method shown above first converts italic words, and then specific characters which are omitted. This method isn't pretty, but it works; a better way would be very welcome. (These operations should happen in the Courses file as well, but didn't.)
- Instructions to replace my mark-up symbols by HTML equivalents for bold and italic character styles appear. I'm not sure about them, and on the whole believe now that they shouldn't be there. I

vacillated for some time between retaining my symbols for consistency, and using the HTML versions for more general comprehensibility. Now I think that I should stick to my version for internal use, though it might be sensible to change to HTML or other notation when offering the text to functionaries for amendment.

- Contrived formats are a nuisance. Special styles, reflecting function as well as form, should be used wherever necessary. For example, for additional formatting within a bulleted list it is silly to use an existing style such as *midp2*, even if the form is right; if necessary, invent a new style (perhaps *midp2 bull*) which denotes both the required layout and the fact that it appears in a bulleted list. Generally, the style should tell you what's happening in sufficient detail to replace the formatting symbols.

The contrast with the Courses file is striking. In this case, there is no convenient textual flag to identify significant parts of the file, and any formatting not directly attached to the character styles must be done using paragraph styles.

OTHER FILES.

Not yet : see above. The mark-up file is a satisfactory intermediate state for the time being. I shall work on procedures for interconverting mark-up and other source files in good time.

Whether this is a good strategy for ever is not so clear. What do I do in case of amendments during the year ? Any changes made should be applied to the appropriate primary files, and from there distributed through the system. It is not clear whether that should affect these mark-up files. If they are seen as essentially handbook material, then perhaps they should not be changed – but in that case I must construct a further set of conversions between primary files and various HTML files.

But perhaps that's right. I've been obsessed with handbook files for a while now, and perhaps I'm seeing them as more important than is in fact sensible. The primary files are, after all, primary, and, while intermediate representations can be useful if they make life easier, it isn't obvious that something designed in order to produce the handbook is well adapted for use as a more general intermediate file.

Further thought is required. At present, I lean to the idea that the mark-up file is an artefact concerned with the handbooks (including the www version) alone, so that apart from changes to the handbook text nothing is likely to require transmission back to more remote sources – it's up to the people to make sure that they're changed accordingly. This implies that any change to primary files can only get to the handbooks (including the www version) by reconstructing the mark-up file. That seems fair enough, really, though it might be worth investigating means of doing it in modules.

REFERENCES.

1 : G.A. Creak : *Background for a document generator*, unpublished Working Note AC115 (September, 1997).

2 : G.A. Creak : *Making the HTML version of the handbooks*, unpublished Working Note AC118 (December, 1997).

3 : VTA is all in the mind, as I don't seem to have written anything but an unpublished paper to define it. Something will happen eventually, but at the moment this is as good as I can manage : G.A. Creak : *Reaching beyond words in rehabilitation computer systems*, unpublished Working Note AC96 (May, 1996).

4 : G.A. Creak : *Designing the document factory*, unpublished Working Note AC117 (December, 1997).