WORKING NOTE AC120

Alan Creak
19 December 1997

HANDBOOK PREPARATION : THE FUTURE (PERHAPS)

Experience gained in constructing the department's handbooks for 1998 is reviewed, and a fairly plausible proposal for how to do it in future is put forward.

In an uncomfortably large number of recent Working Notes[1, 2, 3, 4], I have recorded experience and occasional thoughts relating to my activities in preparing the handbook material for 1998. Now it is time (he wrote, pretentiously) to formulate a structure for the future.

In fact, that isn't a pretentious statement – it's a dire necessity. If I don't settle on a firm design rather soon, I'll have another year of muddling through without gaining any benefits from my experience of 1997. I've now worked through the complete handbook cycle, and probably understand it as well as I ever shall, so it's a good time to consider how to proceed.

NOMENCLATURE

In the various documents I have produced so far, I have striven to use vocabulary consistently, and usually failed. That's because when I started I didn't know what I was talking about, and on the journey to a better understanding I talked about it in so many different ways and with so many different temporary models of the structure of the overall system that I simply lost track.

All being well, such confusion is now behind us. I therefore hereby promulgate a set of definitions of terms which I shall use consistently for at least a week, and – I hope – for a lot longer than that. Notice that some statements within this set of definitions are at variance with statements I have made with equal confidence earlier in the story. These override earlier statements; they are justified (I hope) in the later parts of this note. I shall at least attempt to use them consistently throughout this document. I think that, provided that the general outlines of the file structure described below remain intact, the terminology will work. It is functional rather than beautiful, but that isn't necessarily a bad thing.

Composite files : Files of any nature containing material from more than one source. Such files are usually prepared for publication, or as intermediates in constructing files for publication.

HTML files : Mark-up files written in HTML for display using World-Wide Web technology. These are usually modular in nature, and written in unformatted text.

Mark-up files : Any text files in which structure is indicated by reserved mark-up symbols. HTML files are examples, but the files of the Reference collection are formatted as mark-up files using a specially designed mark-up language.

Modular files : Files containing information on single topics, typically used as the atoms from which the handbook is constructed. The Reference collection is composed of modular files.

Prepared text : A special form of mark-up file in which mark-up symbols determine only paragraph and character styles to be used when the file is converted into a Word file.

Reference collection : The "real" database for the department's information. Parts of it are identified in my notes[1] on the structure of the handbook review procedure. The files of the Reference collection are modular mark-up files.

Word files : Files prepared from the database for presentation with Word. They are usually intended for printing or other distribution – perhaps as PDF files, for example. The handbook files are the most obvious (to me, at the moment); these are quite large modular files. In fact, they are converted into Pagemaker form for printing.

Working collection : A set of modular files, originally copied from the reference collection, used as working material for constructing the handbook. These files are sent to functionaries for checking and amendment, eventually converted into prepared text, and even more eventually reconstructed from the Word files.

WHAT HAS TO HAPPEN.

The outlines, and a bit more, were described in an earlier note[1]; most of the description given there still applies. I now have a clearer view of the whole process, so this description should be seen as a refinement of the earlier attempt.

The major component of the overall operation is to get material from the Reference collection and to make it available to people who might want it in the form of the department handbook. An important subsidiary component is to get material added or changed during the editing back from the Word files into the Reference collection[4]. Strictly speaking, it might be sensible to think of the handbook review process, where all the material is perused by the responsible functionary and corrected if necessary, as a separate operation, but in practice it is only the immediate requirement to print the handbook which spurs the functionaries to act, so the two operations are at least closely linked, and it makes sense to try to organise them as part of the same activity.

Nevertheless, however good the motive in theory, in practice the mixing makes it harder to see just what's going on. The view is further obscured by the original almost total dependence on Word, and – perhaps related to the mixture of activities – ill-defined time relationships. Perhaps that's why my early document ostensibly on a document generator[1] is almost all about the review process. In any case, it was not until towards the end of the 1997 exercise that the picture became clear.

Here is a list of significant points about the process. It is to be read together with the list headed "The handbook task" from the original specification[1].

Eliciting information from functionaries : I have expressed, or at least implied, some measure of confidence that this part of the handbook operation can be, at least in part, automated[1]. I still think that's right, but it will take a rather simpler structure than I used this year if full advantage of automation is to be gained. This year, sticking to a standard identification convention based on handbook section numbers simplified matters such as sending out requests for revision, identifying answers when they returned, and recording the events. It faltered significantly when the section numbers changed part way through the process, but the principle is good.

Editing changes to the Word files[1, 4] : These can happen up to the date of printing the handbooks. Eventually, these amendments must find their way back into the Reference collection, and thence to wherever they might be needed.

Late changes to the Skeleton file[4] : These are not so common, but more far-reaching in their effects. Only one such event happened in 1997, when it was decided that some material was inappropriately placed in the handbook, and should be moved from its original position into a different place, where it would form a new section. Consequent changes were simple text editing in the text files (easy), renumbering sections (not so easy), and changing references to sections (easy in the contents; I just hoped there were no others). Ideally, the change to the Skeleton file should be automatic; part of it at least is possible in principle, just as it was possible to construct the handbook contents pages automatically[3], but it might be that not all the required information is present in the Word file.

Make the Prepared text from the mark-up files : This operation is governed by the Skeleton files. Components from the files, both (components and files) selected according to the requirements of the handbooks, are combined with appropriate markings into Word files which can then be converted into the final required form by a Word macro. This is the job of the Document Factory, which has been worked out in some detail for the Courses file[2], but not so carefully for the rest of the material.

Feedback from Word to mark-up[1, 4] : The exercise so far has carried the conversion back to a file format which in the long term is likely to be at best an intermediate file. It has been a composite mark-up file, closer in form to the Reference collection than to the Prepared text, but containing only handbook information. In 1997, there's nothing else in the Reference collection, but in future the restriction means that I need a "merging" programme to transfer material from the regenerated files into the Reference collection.

Inserting HTML links[3] : this sounds like a fairly trivial problem in the context of the department's information system, and perhaps it is, but it has to be done and turns out to be curiously complicated. There are conflicting requirements from different parts of the system :

- Text defining links to be used as links, names, etc. must be present in the HTML files, but not in the Word files.
- Text defining links to be used as information must be present in both HTML files and Word files, but should probably also be presented as active links in the HTML files.
- Some links, particularly those relating different sections of the handbook, must be inserted automatically because the actual addresses will not be known until the handbook is completed.
- Mark-up symbols defining these characteristics, whatever they are, must not be confusing for functionaries amending the files, and unnecessary duplication should be avoided.
- "Mailto" links will also be necessary, and pose generally similar problems.

Inserting graphics : All my discussion so far has been about text. While most of the handbook material is in text form, there are a few diagrams, and some means of dealing with them is necessary. At the most primitive, one could simply insert notes to mark spaces for manual operations, so it is always possible to cope somehow. The important point here is to ensure that some appropriate notation is inserted into the files at the required place[3].

Subroutine calls : At some point, it is necessary to combine the modular files in order to construct composite files; the most obvious example is in converting the skeleton file into text for the handbook. There is no reason why this operation should not be used recursively – as indeed is likely, because (for example) we know that some information about the Courses records is kept in the Functionaries file. Something amounting to a subroutine call is therefore needed[3]. (If you prefer to avoid commitment to a particular implementation, you might like to think of it as a reference for late binding.) It is similar in principle to a file include operation, where the included file is treated in the same way as the parent, so can included text and further instructions if required.

Continuing changes to the Reference collection : Mistakes in the handbook are identified and reported, people come and go, their qualifications change and interests develop, textbooks are found or go out of print, details of some courses turn out to be wrong, information not available at the time of printing comes to hand. Generally, we might want to change information in any part of the handbook at any time in the year. This year, the Functionaries list was altered after the handbooks were printed, so that details of academics' connections with three courses were changed. All this must be fed back into the Reference collection in an orderly way, and made available as required – particularly in the HTML information system, which should be done as quickly as possible.

Continuing changes to the HTML files : We can't do much about the printed handbooks, but we can try to keep the World-Wide Web version up to date, and the rest of the department's WWW material. At present there's a lot of duplication, which should be sorted out – for example, there are several independent lists of people in the department, and they're not the same. Whether the handbook lists should be the primary lists, or instead the handbook (at least, the HTML handbook) should refer to other files, is to be decided.

IMPLICATIONS FOR DESIGN.

This is not an attempt at formal design, desirable though that would be. It is instead a collection of notes pointing to desirable structural features of the file system supporting the handbook preparation, and perhaps of the programmes which deal with the conversions between them.

BIG FILES OR LITTLE ?

In 1997 I used composite mark-up files – indeed, I took some trouble to combine the rather haphazard collection of files which I inherited, some modular and some composite, into a single modular file for each handbook. The HTML files remained largely modular to fit in with the department's not very documented conventions for WWW files, but otherwise composite files ruled. They worked well, by and large, but reflection suggests that this was only in part a consequence of good design. The part in question was that the composite nature of the files was very well adapted to the use of "Change All" operations with Word macros, and operations of that sort did most of the hard work in 1997, even on the mark-up files. This will not last.

It's clear from quite recent writings[4] that I still thought in that way until about when I reviewed the overall process, but particularly in the later stages of the operation it became clear that dealing with large composite files, while occasionally necessary, was inconveniently clumsy, and made it much harder to deal with separate items in different ways.

The composite files were not at all good for operations requiring that complete sections be moved from one place to another in the handbook. It was not difficult to do by hand, but any reduction in manual operations is a good thing, and not having to do things at all is better than having to do them. Composite files are also inconvenient if many people contribute to one of the files; repeatedly editing a large file to alter fragments is harder than simply replacing small files. It is also rather less safe, as mistakes made by one functionary can easily flow on to later parts of the material. I have therefore concluded that it will be better to keep the mark-up and Reference collection files as modules.

For the most part, using modular files is no more difficult than using composite files. (The few exceptions are to do with searching for particular items, but that isn't an activity which I require in the handbook preparation. It's useful for occasional manual jobs, but they can probably be done just as well using Word operations with the Word files[1].) In illustration, consider the Document Factory[2]; changing the input from a composite to a modular file requires only that an input stream be provided identifying the files to be used, and that the "machine" which moves to the next record of the Courses file is replaced by one which reads the required file. Modular files also lend themselves to comparatively easy incorporation of amendments to the handbook through the year. The amendments are easier to make, but it is more significant that as each module corresponds directly to an HTML file, it isn't necessary to redo all the HTML handbook.

NAMES, NOT NUMBERS.

Files should have simple names for reference. This year's experience shows why. I started with two sets of inherited files, one for each printed handbook, and a set of HTML files which I hardly used. Each file in a set contained the material for one or more sections of the handbook; in some cases the files represented material to be checked by an identifiable functionary, but in other cases the reason for a file's composition was not clear. Each file was named according to its handbook chapter name and first section number, but there was no indication of the number of the final section contained in the file. This sometimes made it fairly difficult to find required material.

Being cautious, and unwilling to break anything which I didn't know about, I retained these files for the first editing run of the handbooks. During this exercise, I sent parts of the handbook by electronic mail to the various functionaries responsible, and collected and collated their replies. To make this feasible, I gave titles to the messages which identified both the file and the handbook section concerned – so a message entitled "UG 02.0 2.3" was about section 2.3 in file "02.0 Ugrad Courses of Study" in the UnderGraduate handbook collection. (I added the leading zero in the file name after becoming exasperated at repeatedly losing files with two-digit chapter numbers.)

This is clumsy, but it worked – until I wanted to change the handbook format by shuffling the sections a bit. Then I had to choose between deferring the shuffle, which would be confusing, or using two sets of numbers, which would be confusing. (I chose to use two sets of numbers; it was confusing.)

There are other requirements too : for example, we want to be able to define cross references (both in words and HTML links) within the handbook, and it would be helpful to be able to identify files without knowing their positions in the handbook. All in all, there seems to be a fairly clear case for identifying files by name rather than by handbook position.

Is there anything to say in favour of retaining the handbook-based names ? The only point of which I can think is that the numbering method does make it easy to list the files in their order of appearance in the handbook – provided, of course, that you remember to use two-digit chapter numbers, which was not the case with the files I inherited. So far as I can remember, I never wanted to list the files in that order for any reason connected with the order itself.

Finally, there is a question about the file names : will it be possible to devise short, but usefully descriptive, names ? My guess is that it probably will, though some care will be needed. Time will tell.

The question of names is also important in identifying people. In this case, the difficulty is not that we don't have a recognisable name; it is that we have too many. There have been moves in the past to cause us to settle on one form of name as standard, but this has never really worked, because people aren't like that. I am G.A. Creak, A. Creak, Alan Creak, G. Alan Creak, Dr Creak, Alan, alan, a.creak, a_creak, G.A.C., A.C., Creak, G. Creak (rarely, and by accident), and gcre003 (my "UPI", by courtesy of the university, and ridiculous). Oh, and 6046069. At least, both "Alan" and "Creak" are, at the moment, unique in the department – but we have multiple Johns, Peters, and Andrews (very apostolic). And Roberts, though they have ingeniously called themselves Robert, Rob, and Bob, so we don't notice. And we have two Caludes, and two Lennons, and occasionally two Lobbs. Then there are Gary and Garry, and Christian, Cristian, Cris, and Chris, not all of whom are distinct.

Which name do we use when writing our documents ? It doesn't matter much if it's purely descriptive and destined to remain as simple text; if it works, it will do. It matters more if we try to make automatic links to people's files or to electronic mail addresses, when it's important to get the right sort of name.

In the handbooks, it turns out to be more important than I expected. Although people's names turn up quite often, it is almost always in circumstances where they can be automatically generated from the Functionaries or People files, so I thought that, apart from a few instances which could be managed by hand, and preferably not by me, all would be well. In practice, it is not so. The Functionaries file, from which most of the operations start, identifies people by surname, with an initial (afterwards) if necessary. To automate the system, these must be identified with entries in the People file, which must then provide at least a preferred name for printing in the handbook, and an electronic mail address. A fairly versatile name resolver therefore seems to be quite a good idea.

Finally, for all sorts of arbitrary name, we will need directories so that functionaries can use them when constructing their text.

CONSISTENT WORD STYLES.

It is obviously important to define styles – particularly paragraph styles – to achieve the desired layout in the Word files[1]. One of the significant problems with the files I received was that sensible styles had not been defined; spaces and tabulation had been used to bend simple styles, and local styles had been defined but not renamed, so the formatting information they represented was not available generally. It took me a very long time to disentangle the mess.

Style definitions are also important in identifying the nature of the formatted material, so that it can be converted back into mark-up form when handbook editing is complete[4]. To this end, a format might be given more than one name in order to carry this sort of type information.

There are two immediate conclusions : first, styles are very significant components of the Word files; and, second, that it is important to design a set of standard styles to be used throughout the Word files. Without design, or at least acceptable guidelines, there is a risk of proliferating styles needlessly. This is to be avoided, for every style must be defined in the Word templates used, and requires instructions for conversion back into mark-up form.

Here I shall restrict myself to guidelines, because I think determining the details of the styles required needs more careful analysis than I have time to perform now. The guidelines are in any case more important, because, if well constructed, they are likely to preserve good structure through any future developments that might be required. Here, then, are some guidelines. They apply to format in the Word files; HTML is a different problem, but if there is sufficient information for Word there will be sufficient for HTML.

Guideline 1 : ALL layout should be controlled by paragraph styles. Multiple spaces, tabulation characters, and ends-of-lines in the mark-up text should not be significant except as explicitly defined in certain cases. (The convention that a double end-of-line represents a paragraph break[2] is an example.)

Guideline 2 : There should be a standard base of simple styles which can be used in any document. These should determine overall attributes such as indenting, text position, text size, and so on. They should have no tabulation stops. (At the moment, the handbooks have almost no clever formatting, and hardly any indenting, so rather few basic styles will be needed. Simple text, centred text, and bulleted lists cover most of the requirements. Others will doubtless develop.)

Guideline 3 : Special styles required, which should be as little as possible, should be defined in terms of the closest base style. They should always be named.

Guideline 4 : Styles may be defined as identical with other styles if the new name is required to carry other information[4] (see above).

Guideline 5 : If redefining styles, care must be taken not to mess up the style structure. Dependencies between styles, such as might be set up by following Guidelines 3 and 4, should be considered carefully.

Guideline 6 (Occam's razor) : Styles should not be proliferated without necessity. Always use an existing style unless there's good reason to define a new one.

And so on. Doubtless experience will suggest other guidelines, probably too late.

CONSISTENT MARK-UP CONVENTIONS.

The principles for mark-up are much the same as those for styles. Consistency is exceedingly important, as the mark-up symbols represent not only printed layout but also the field structure inside file records.

Mark-up symbols must be easily identifiable. I have chosen the form !!....|| as the basis for my notation because it is rather unlikely to turn up by accident. It is slightly vulnerable, because people do occasionally use double exclamation marks in text, so a change to |!...|| might be a worthwhile improvement.

The symbols should also be reasonably comprehensible. That's why I've used rather long symbols (!!section heading||, !!bulletliston||, !!italicon||, etc.). As against that, it is easier to type, copy, or otherwise manhandle a long string wrong; that's one reason why I wondered whether to use the HTML conventions for character styles (italic, bold, etc) instead of my more cumbersome form. I'll keep the longer forms in the Reference collection, but perhaps use HTML forms, or both, in material I distribute for checking.

Several sorts of mark-up symbol can be used, and they have different sorts of effect on the files in which they are used. Here's a list of some existing and potential examples.

Class of symbol	*Symbol type*	*Description*	*Example*
Mark-up file structure	Field markers	These symbols identify fields in files. They are only significant in the files in which they are defined[2]. A field extends from its opening marker to the beginning of the next recognised field marker, or to the end of the file.	!!section heading||
Text formatting	Paragraph formatting	Two sorts of paragraph style mark-up symbols appear to have evolved. One is a single symbol, and unconditionally switches the paragraph style until further notice; the other comes as a pair of brackets, and identifies a range of text – always a set of paragraphs – which is to be presented in a special way.	!!text|| !!bulletliston||, !!bulletlistoff||
	Text styles	These symbols are always (so far) brackets, and identify arbitrary strings of text which are to be presented in other than normal text form.	!!italicon||, !!italic off||
Operations	Functions	These are (so far non-existent) means of converting some text into some other text. They cover such operations as references to other parts of the handbook, people's home pages, people's electronic mail addresses, who does what. So far, they are all search-and-retrieval operations. The "examples" are pure invention.	!!reference "contents"|| !!functionary "340 supervisor"||
	Instructions	These are (also so far non-existent) means of causing the handbook machine to do something special. Uses so far observed are for inserting files or pictures. More invention, but not original[3].	!!represents stage2courselist|| !!picture Tamaki||

THE SKELETON FILE.

The Skeleton file turns up as a notional entity in my early discussion[1], but has never had a real physical existence apart from its approximate embodiment in the handbooks' contents pages. This is particularly clearly demonstrated by the curious fact that I produced the contents pages by extensively editing the Word or (for the HTML contents) mark-up files, which is the wrong way round. It is clear that in a more automated system the Skeleton file should be used much more actively.

Just how it should be constructed is a different question. Perhaps the obvious organisation (to people accustomed to hierarchic structures) is to restrict a master file to a list of chapters, and to define sections within files for the chapters. It seems likely that in practice this organisation would be too fiddly. The Skeleton files should be simple documents which can be read easily, and give a clear picture of the organisation of the handbooks. Experience suggests that the optimum description level is probably to list the handbook sections rather than chapters.

It seems obvious that chapter and section numbering can be done automatically. It is less obvious that it need be done at all. The numbers are perhaps useful for remote references, where searching a printed handbook for a number is guided by other section numbers, and perhaps they could be retained for that purpose. On the other hand, they are merely confusing for HTML files, because section sequence is not significant. The answer is perhaps to ensure that all such references are produced automatically, and to generate numbers only for the Word files.

CONCLUSIONS OF A SORT.

The notes which follow are tentative answers to some of the points made above, and other things that came to mind. They are not intended to be a complete account of anything at all; I'm writing them down so that I don't forget them.

THE FILE FORMAT.

All files in the Reference collection should contain their names, used for identification, and who is responsible for them, which can be either a name or a functionary.

```
!!file|| xyz

!!owner|| alan
!!owner|| !!functionary "340 supervisor"||
```

Handbook files will also contain

```
!!document|| <which handbook>
!!section number|| <number in current printed handbook>
!!section heading|| <section title>
```

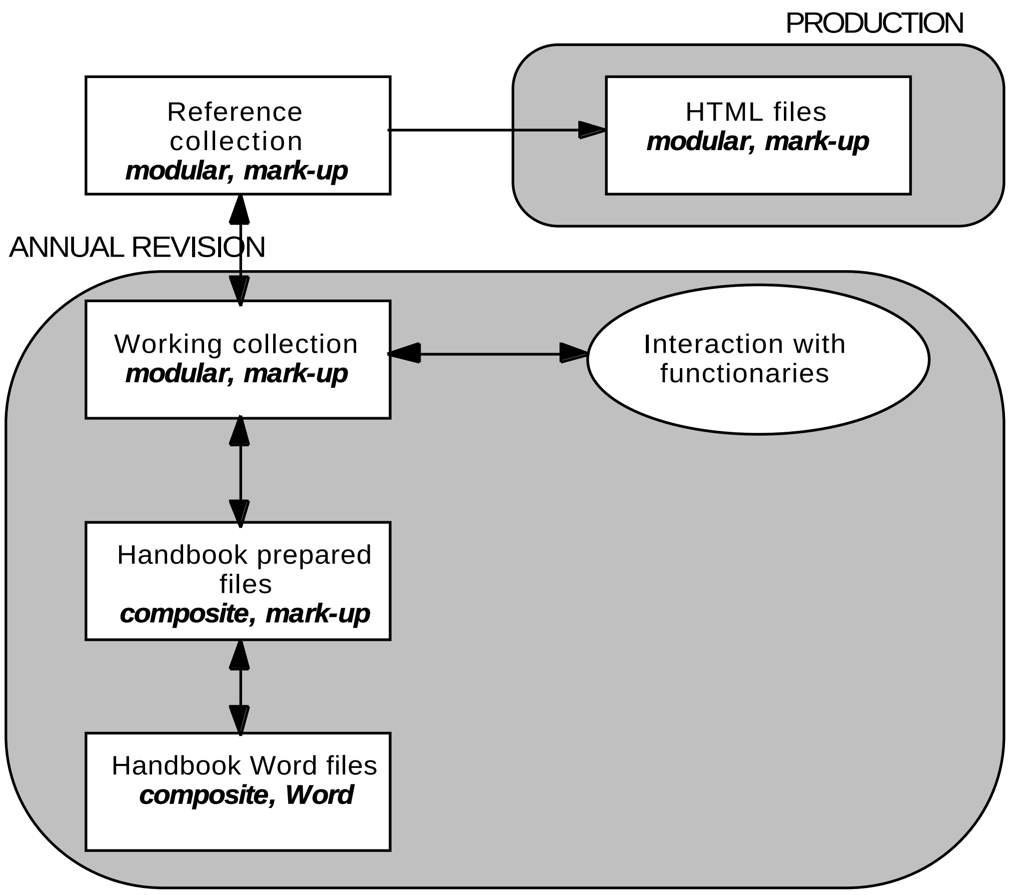
Notice that there are not only undergraduate and graduate handbooks; there is a faculty handbook, and we have some entries in the commerce faculty handbook. These are not particularly demanding in handbook processing terms, and do not affect the arguments (such as they are) presented for the department's handbook system, but it is reasonable to stick to the same general format if we have one. More generally, information is required for other documents, so it seems reasonable to include a general document identifier.

THE HANDBOOK PROCESS.

A sketch of the expected process appears on the next page. It is not to be taken too seriously, but represents my current ideas reasonably well. Moving downwards in the diagram corresponds roughly to increasing visible formatting, and possibly to more precise selection of material; moving upwards implies return of information gathered at the more detailed level, and possibly merging with the parent files.

The two branches are headed *Annual Revision* and *Production* to emphasise their different natures. It is noteworthy that the results from the Annual Revision flow back to the Reference collection, but the connection from Reference collection to HTML files is one-way. Changes to the Reference collection might happen at any time, whereupon it is possible to reflect these stages quickly in the HTML files without further ado[3]; the modularity of the files concerned simplifies this operation. My depiction of the printed handbooks as a by-product of the annual revision, but might not be an unreasonable view of possible future developments. Even without the printed handbooks, something like the Annual Revision would be a sensible exercise; as it is, the need for the printed handbooks gives us a convincing trigger.

The Working collection is the base for the handbook preparation activities. It begins as a copy of the Reference collection, for obvious reasons. (The Reference collection itself is, I hope, protected by the normal system back-up operations, and the one-way-ness of the connection with the HTML files preserves it from any harm in that direction.) As a general principle, only material needed for the revision exercise need be copied, and it might be that this becomes less in quantity as we work down the diagram. The complementary implication is that moving upwards might be a merging operation rather than a simple regeneration and replacement. At the very sensitive level, it is obvious that merging data with the existing Reference collection must be carried out with great care. Just what that means is a lot less obvious.


PRODUCTION
Reference collection modular, mark-up
HTML files modular, mark-up
ANNUAL REVISION
Working collection modular, mark-up
Interaction with functionaries
Handbook prepared files composite, mark-up
Handbook Word files composite, Word

The horizontal branch involving "Interaction with functionaries" covers the operations described when I thought that was the whole story apart from a few tidying-up operations[1]. Ah, innocence. It is still true that, because of its dependence on people rather than machinery, this component is likely to be the most ticklish, and to remain strongly labour-intensive; while the process as outlined earlier is probably a good guide for the future, much remains to be determined at the level of ways and means.

The transitions from Working collection to Prepared files, and then to Word files, are those performed by the "Document Factory"[2], and the reverse operations[4] have been explored. Again, it is certain that much development work remains to be done, but there are reasonable grounds for confidence that no insurmountable problems lie ahead.

REFERENCES.

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

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

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

4 : G.A. Creak : *Going back again*, unpublished Working Note AC119 (December, 1997).