Experiences in document conversions performed during the MeDoc project

Hubert Feyrer <hubert.feyrer@rz.fh-regensburg.de>

March 17, 1998

Abstract

This document summarizes the experiences made in converting electronically available documents in various formats for use in the MeDoc project's electronic library. After giving an overview of available formats and tools to manipulate them, choices are outlined for design questions that will arise during document format conversion. Guidelines are given to straighten the conversion process described next as much as possible. Further ways of improvement are discussed at the end of this document.

Contents

1 Introduction: Value of document format conversions for electronic libraries **3**
- 1.1 Need for a Common Document Format . . . 3
- 1.2 Possible Common Formats . . . 3
- 1.3 Drawbacks of Using a Common Format . . . 3
- 1.4 The Way to Be Gone . . . 3

2 Document Formats **4**
- 2.1 Overview . . . 4

3 Tools: converters and others **5**
- 3.1 Intro . . . 5

4 Guidelines for Document Conversion **5**
- 4.1 Target Format . . . 6
- 4.2 Language . . . 7
- 4.3 Fonts . . . 7
- 4.4 Colors . . . 8
- 4.5 Usage of Images . . . 9
- 4.6 Document Segmentation . . . 10
- 4.7 Design of the Navigation Bar . . . 10
- 4.8 Links . . . 11
- 4.9 Table of Contents . . . 14
- 4.10 References . . . 17
- 4.11 Index . . . 18
- 4.12 Titlepage . . . 19

5 Communication and Exchange of Experiences **19**

6 Workflow of a Conversion **19**
- 6.1 Obtaining Sources . . . 19
- 6.2 Specifying the Goal . . . 21
- 6.3 Intercepting the Converter . . . 21
- 6.4 Feedback . . . 22
- 6.5 Document delivery . . . 22

7 Conclusion **23**

1 Introduction: Value of document format conversions for electronic libraries

1.1 Need for a Common Document Format

Libraries have to keep their contents in a uniform format to facilitate reading, indexing and searching for their users. Also, the choice has to be done depending on these users' needs. And with a library as open to its customers as the MeDoc library, a whole variety of platforms have to be adressed as clients to the library. So the format has to be something that can be easily read on any of those platforms, with enforcing no or only small restrictions on these clients. Only one format has to be chosen, to avoid redundancy and inconsistencies.

The format(s) used for the library have to be open so implementing services for reading and searching the texts are possible for the users on any platforms, and noone is forced to use (and pay!) any proprietary product ported to his platform due to the marketing strategies of a single vendor.

1.2 Possible Common Formats

Open standards that are in common use today and that are available on a wide range of platforms are SGML (ISO 8879) and HTML (URL: http://www.w3-.org/) as document description languages as well as PostScript and its successor, the Portable Document Format (PDF), both of which belong to the category of page description languages/formats.

For the MeDoc library, HTML and PDF have been chosen as the formats to be used, see [1].

1.3 Drawbacks of Using a Common Format

But hardly any book or text today is written in HTML, not to speak of hand-coded PDF. Rather, text-processors ranging from rather archaic to high-tech are used. Some of these text-processors do offer converting into HTML and/or PDF, and some don't, so one probably has to go a more complicated way in reaching the wanted target format. And even if there's a straight way for conversion, today's tools aren't always 100% perfect. Bugs and immaturity often prevent correct conversion into the target format in a single step, so it's often necessary to help these tools, sometimes even fixing what they broke.

1.4 The Way to Be Gone

As well as there's a whole lot of document formats to convert from, via and to, there are several tools and techniques that can be used to convert a document into the given format. Besides that, often the author of the document can help to ease the conversion by paying attention to the fact that the book isn't (only) going to be printed on paper.

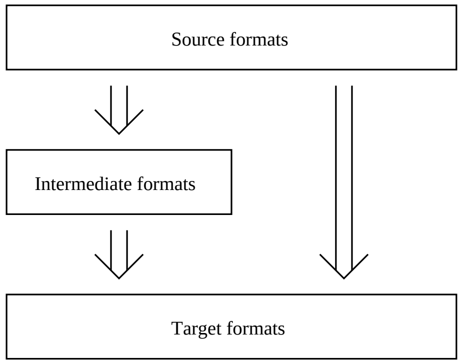

Figure 1: The three groups of document formats and their relationship

This report is about these formats, techniques, their interchangeability and application.

2 Document Formats

2.1 Overview

Many different electronic document formats are in use today, they can be devided into three groups, as shown in Figure 1.

Target formats are the ones which existing documents are converted to and which are used in keeping documents for reading and other usage. Existing documents are produced in one of several source formats, usually by the author of a book, using some text processor. Some of these source formats can be brought into the desired target format either directly or via some intermediate format that can then be converted into the final (target) format in a second step.

Table 1 describes some of the attributes of the formats in these three categories. The “Name” column indicates the name of a certain document format. The category into which the document format belongs (see Fig. 1) is indicated by the “Category” column. “Standard” indicates whether there is an open standard either by some independent group or a vendor (‘open’), the format's fixed according to some reference implementation (‘source’), or the format is defined by some vendor and may change at any time (‘vendor’). The “Structure” column tells if the contents of a file confirming to any of the formats listed can be read without a special tool (‘ascii’), or if special software is needed to process

Name	Category	Standard	Structure	More information
HTML	Target	open	ascii	http://www.w3.org/
PDF	Target	open	binary	[12]
PostScript	Intermediate	open	ascii	[2], [3], [4]
DVI	Intermediate	source	binary	[9]
RTF	Intermediate	open	ascii	[11]
LaTeX	Source	source	ascii	[9]
WinWord	Source	vendor	binary	http://www.microsoft.com/
FrameMaker	Source	vendor	binary	http://www.frame.com/

Table 1: Summary on document formats

documents in that format ('binary'). Finally, the "More information" colunm lists some references on where to get more information on the given document format.

3 Tools: converters and others

3.1 Intro

Having that many different document formats demands a choice of programs for converting one into another, and doing other manipulations on documents in any of these formats. The availability and quality of such tools for a given format also make it more or less easy to deal with that format.

Table 2 gives a list of the most commonly used tools that are available for manipulating the document formats described in the previous chapter. Given are the name of the tool, the possible input- and output-formats as well as a URL where to get more information on that tool.

4 Guidelines for Document Conversion

Having a set of tools available is like an instrument to a musician, but there's more than just knowing how to play this instrument. Without the knowledge on how to play to the notes, disharmonies will come out, and the result will look (sound) bad[1]. And just as a musician has more than one way to interpret a certain piece of music, the converter of a document has to decide by which stylistic means he arranges the given fragments.

The following list of keypoints is intended to ease the process of identifying points which need attention, and to aid in deciding on how to solve them.

[1]Of course it's also possible the composer caused some disharmonies - during document conversion, this usually means more work for the converter. But that's another story.

Tool: Where to get	Inputformat	$\mapsto$ Outputformat
Word Internet Assistant: http://www.microsoft.com/word/internet/	ASCII, Word, RTF, ...	$\mapsto$ HTML
latex2html: http://www-dsed.llnl.gov/files/programs/unix/latex2html/	TeX, LaTeX	$\mapsto$ HTML
frame2html: ftp://ftp.nta.no/pub/fm2html/fm2html.v.0.8.9a.tar.Z	Native FrameMaker, MIF	$\mapsto$ HTML
WebWorks HTML Lite: http://www.frame.com/	Native FrameMaker, MIF	$\mapsto$ HTML
rtf2html: http://www.sunpack.com/RTF/latest/	RTF	$\mapsto$ HTML
TeX: ftp://ftp.tug.org/tex/	TeX, LaTeX	$\mapsto$ DVI
dvips: http://www.radicaleye.com/dvips.html	DVI	$\mapsto$ PostScript
Adobe Distiller: http://www.adobe.com/prodindex/acrobat/main.html	PostScript	$\mapsto$ PDF
Adobe Exchange: http://www.adobe.com/prodindex/acrobat/main.html	PDF	$\mapsto$ PDF
Ghostscript: http://www.cs.wisc.edu/ ghost/	PostScript	$\mapsto$ TIFF, PDF, ...
HotDog Pro: http://www.sausage.com/store/indexST.htm	HTML, ASCII	$\mapsto$ HTML
FrontPage: http://www.microsoft.com/frontpage/	HTML, ASCII, RTF, WinWord	$\mapsto$ HTML

Table 2: Summary on document manipulation utilities

4.1 Target Format

The choice of the target document format, into which to convert a given electronic document depends mainly on the contents of this document. Given the impact of the usage pattern of a document, HTML is usually preferrable, but there are reasons which speak for the PDF format in some cases.

With HTML, some things are easy to do, some are difficult to do, and others aren't possible at all, given the current state of the available converters and browsers commonly used, and the definitions on HTML they implement. E. g. documents with lots of formulas end up in including lots of small images into the HTML text, as there's currently no HTML reader in widespread use that supports formulas and mathematical symbols. In addition, available converters as latex2html do not work properly in translating all those formulas into images, so PDF is the way to go here.

Other features such as comments and notes on the margin that the inventors of HTML didn't have in mind but that are common on printed media can be realized in HTML with some effort with tables, frames, and other advanced features. If not, one should investigate if switching over to PDF can solve the problem.

Besides visible aspects, the way in which the document will be used has some impact on what target format to use. While PDF documents are usually downloaded in one piece, documents in the HTML format are downloaded page by page. So if a document is expected to be read from front to back, PDF might be the choice. If it's more likely that only few parts of a document are accessed frequently (like e.g. table of contents and index) and the text is accessed on a on-demand basis, choosing HTML will support this usage pattern in a less network saturating way.

4.2 Language

Sometimes a book exists in more than one language, probably the author's native language and english. In such a case, the decision has to be made which version of the document to convert for further use.

Sometimes the answer is given by the fact which version of the document is more up to date. In any other case the language should be used that the intended user community is most likely to accept preferrably. If the audience is international, english is the language of choice.

4.3 Fonts

When reading text on the screen, the choice of the font used is important. If it's too small, people may have to wear glasses to read it. Furthermore, with a small font, a lot of information can be put on a single page, which can reduce readability, too. On the other hand, choosing a font too big may need the user to scroll too much, which will be a burden when reading long texts, too. Therefore, one should be careful to use any extreme, and not use a certain size just because it fits the converter or it's default.

The main goal to maintain and remember is readbility, taking the most common settings into consideration that the user of a document viewer can tune.

Besides giving general rules, some document formats need special attention for fonts.

When converting TeX-documents into PDF (via DVI and PostScript), one should try to replace the pixel-fonts TeX uses into PostScript fonts, as these are clearer to read and faster to display on screen. Furthermore, when printing, the document is more resolution independent of the output device.

When converting into PDF, one should also choose a font size that would be ok for printing, too. Just because a PDF reader provides the facility to zoom into every word, the page might still get too crowded with a small font.

When converting into HTML, this problem doesn't exist, since the user of the document can set the font size via his browser, so he can adjust it so it fits his reading abilities. However, even here there are two points that should be

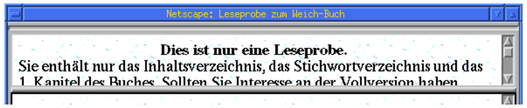

Figure 2: Example of a bad designed (font-insensitive) Frame. Even worse ones don't even provide a scroll bar.

Figure 3: Table and horizontal rule on white and grey background.

remembered. When using images with text in them, e.g. as in many technical drawings etc., the size of that font can not be chanced any more, as it's part of the images. Care must be taken so that this font is displayed ok, too. Second, when using frames, these are usually specified in fixed relations relative to the window size. When very small frames are used, e.g. for a navigation bar or page header/footer, it's possible text (or even worse, links) get scrolled outside of the visible area of the frame when the user has chosen a very big font. It's usually better to use some kind of tables for these purposes as they are font sensitive, in contrast to frames.

4.4 Colors

When using color, a few aspects must be take into account as well, else the result may not be easy to read or even nice to look at.

When going to PDF, using color is usually no matter, as the underlying color models from PostScript ensure that one color displays the same everywhere, besides that, PDF documents are usually in a uniform "black on white" style.

This might not be the case with documents in HTML. The "black on grey" that many browsers offer today is often deemed as boring, and that at least a different background color has to be chosen for the pages. White is easily chosen and also pretty to look at, but one should use it with care if the document uses tables with borders (<TABLE BORDER>), horizontal rules (<HR>) and other objects which have a three-dimensional look by default, as the white background will harm this look, see Figures 3. In such a case, shades of grey and pastel may provider a better look and also keep the 3D look of the said objects.

Figure 4: Characters with and without anti-aliasing

4.5 Usage of Images

When images are part of a document, effort must be taken so the result looks good in the converted document. Often, more than one step is necessary to bring an image from a source to the target document, usually GIF and/or JPEG when heading for HTML. A good set of image manipulation tools will help in making this step easier. But still, one has to know a little bit on what's important when generating images.

First and most important is, that every detail on the picture is visible. Ways to archive this are switching on anti-aliasing if the tools and image formats support it. Anti-aliasing has a positive effect not only on fonts and their readability, but also lines, circles and other graphical objects look less crude, whether they are color or black and white. See Figure 4 for an example.

When a image contains text, the notes on fontsize explained in the previous section apply.

Furthermore, when an image consists of line drawing, the width of these lines shouldn't be taken too narrow or to wide, to keep details and not stroke them out by too thick lines. This is usually no problem when going for PDF, as this format contains good mechanisms for sizing images itself, as well as anti-aliasing is usually done automatically by the document viewer.

But when converting into a pixel format such as GIF and/or JPEG for use in an HTML document, keeping details may be a problem because to have all the details visible on the picture, the dimension of the image may exceed the size of what can be viewed even on a 640x480 display. To solve this problem, it's easiest to put a small (thumbnail) image on the page itself, and then create a link to the image with a higher resolution. That way, everyone can watch the details if he wants, without wasting time to download big image.

The same kind of time-saving can be archived (in addition) by reducing the number of colors an image uses. It's of little use to insert any true- and highcolor-images with millions of colors, when most of the users have displays which hardly support 256 different colors, and even these colors are shared by several images, documents and/or applications.

Beside all those visible aspects of the contents of an image, there are also some guidelines to regard on inserting them into documents.

Most important, every image should have some text outlining what it's good for, either directly above of below it. (Of course it should be only above or below throughout the whole book, not mixed!) When converting into HTML, one might also set an additional ALT-parameter to the IMG-tag (<IMG ALT="some-text" ...>) to give a textual description of the picture in case the browser used to read the document has images disabled or doesn't support them at all.

4.6 Document Segmentation

Document segmentation means dividing a large document into smaller pieces. This technique is used to reduce download time, so only the parts of a documents need to be downloaded that are actually of interest, not the whole document. Furthermore, especially with documents in HTML, having a document divided in small pages prevents that the user has to scroll a lot, and it's easier to bookmark a certain location than if that interesting place is buried somewhere in a long document.

Document segmentation in HTML is usually done one page for each (sub)section, resulting in many small pages that can be loaded quickly. Also, only the images referenced in that page are downloaded, not every one in the whole document. In sections only consisting of the section heading and no text, it's recommended to merge this heading with the next section (and it's text), so that no whole page is used for only a heading. Figure 5 illustrates this merging process.

With PDF, the story is a bit different, as the linking between different files isn't very well handled by the existing processing tools, and therefore rather big segments are recommended, usually containing a whole part or chapter of the document. If an index or working table of contents is essential for a PDF document, one might even considering not segmenting the document at all, to keep the link information.

4.7 Design of the Navigation Bar

With having the big document broken up into many little pages, it's necessary to provide a common method to skip through all these pages, going back and forth. Also, when reading some page, it might be useful to be able to go to the index and/or table of contents as well. The mechanism providing this is called a "navigation bar". In PDF, this is built into the document format and the reader itself, so nothing has to be done there, but in HTML this has to be

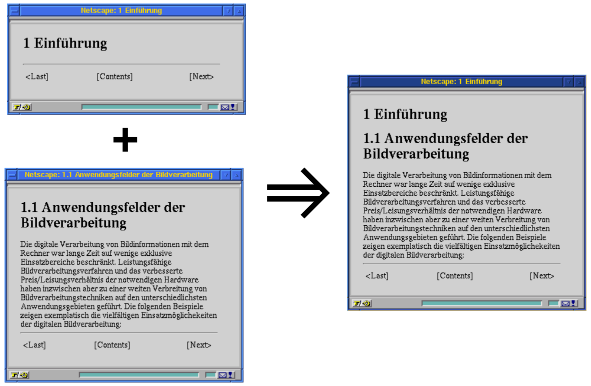

Figure 5: Example of merging two pages after segmentation

realized by hand. The navigation bar provides a doubly linked list of all the pages, for going to the next and previous page, see Figure 6.

The buttons a navigation bar should provide at the very least links to the next and last page. The "back"-link from the table of contents and the "next"-link from the last page are usually omitted. Additional links to the table of contents and index make it much more usable.

Also, a "up" link may provide a way to go up one step in the document hierarchy, as is shown by Figure 7.

The "up"-links only go *one* level up to the next hightest page, as for example the "up"-link for 1.2.1 goes to 1.2.

Aside from the logical structure of the links, the actual look of these links in the navigation bar may vary between documents (although consistency is a must within a single document!). The links may consist of pictures, chapter numbers, chapter names, or both, see Figures 8, 9, 10 and 11 for examples.

When placing navigations bars, attention should be paid that there is one at the beginning of the page, and also a second one at the end, so the reader of the page doesn't have to scroll up when he's done reading the page, but instead can go to the next page directly.

4.8 Links

Links are the essence of every hypertext document. Besides referencing related information, it can be (and is!) used to provide navigation inside a document in HTML. Therefore, it's important to make sure that every link is working properly, to ensure correct navigation, especially in the navigation bar, but also

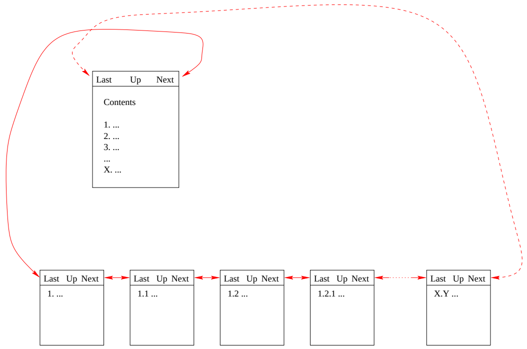

Figure 6: Connecting the document segments with a navigation bar.

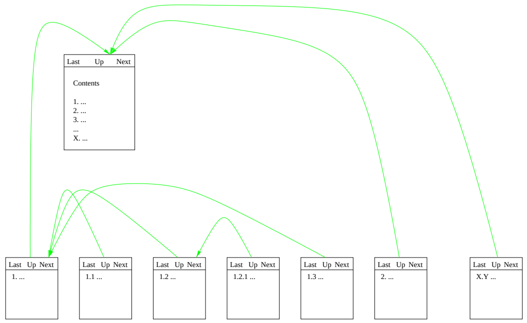

Figure 7: Navigating in the document hierarchy with a "up" link

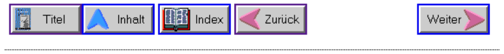

Figure 8: Navigation bar with images.

<BACK] [CONTENTS] [INDEX] [FORWARD>

Figure 9: Navigation bar with plain text.

Figure 10: Navigation bar with chapter names

Figure 11: Navigation bar with chapter numbers and names.

in the table of contents and the index. While HTML-browsers display in detail that a link is broken when followed, PDF readers will act as if the link wasn't there if the linked destination is not available.

Also to provide smooth navigation, the navigation bar should be visible when following a link to a new page. This is not the case when converting LaTeX documents to HTML with latex2html, as the links produced by that tool rather address the first heading on a page, resulting in the navigation bar being scrolled out. When the user of the document decides he wants to go to the next page immediately, he has to use the scoll bar first, which is bad. Rather, these links of the form “xxx#yyy” need to be converted into simple links of the form “xxx”, so the navigation bar's also visible.

As far as possible and supported by source format and converter tools, cross references inside a document should be converted into links as well as any available index.

For links of the form “see chapter XXX”, “see section XXX”, “see figure XXX”, the chapter-, section- or figure-number “XXX” can be the anchor of the link, pointing at the referenced object. For references naming a page (“see page XXX”), it is hardly possible to name a page numer at all, and even in those cases it's unlikely that the named location has anything to do with a physical page. Therefore, it's ok to use some sort of button as an anchor for the link, instead of the page number, as shown in Figure 12.

When using the icon provided by latex2html and choosing a non-standard background color, the icon must be fixed so only the edges are transparent, not

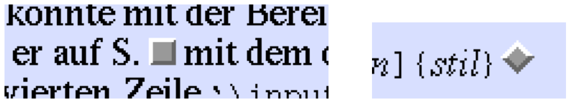

Figure 12: Page references with images as anchor

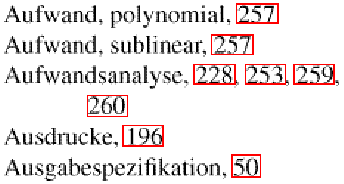

Figure 13: Virtual page numers as anchors for links in index.

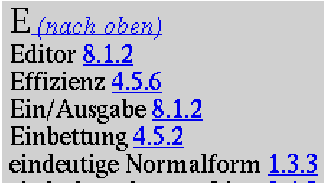

Figure 14: Section numbers as anchors for links in index.

the center.

In the index, there's a greater variety on how to display a anchor. Possible solutions are a virtual page-number that's only there to provide an anchor, an image just as described above, or listing the referenced section's number and/or heading. Figures 13, 14, 15, 16 and 17 show some examples.

When the `\ref` and `\pageref` are used in a LaTeX document that is then converted to HTML using latex2html, a small square is placed as anchor, instead of any page/section number. To turn this into plain text, information written into the .aux file can be used to gain these page/section number. An example script for this can be found in the conversion report from Graf/Eibisch [5].

4.9 Table of Contents

To get an overview of a documents contents, a table of contents should be available, from which every section can be accessed directly. This can be done with PDF and HTML, when converting LaTeXto HTML, the hyperref macros can do this job automatically.

Sometimes the table of contents gets quite long. When converting into HTML, it's recommended to divide the table of contents into two levels. The main table on the first level only displays chapter headings, linking to a table of contents only for this chapter on the second level. Figure 18 illustrates this splitting of the table of contents.

arithmetic operations
    Arithmetic operations, Arithmetic operations
arithmetic/logic unit
    Switches and symbols
arrays
    Arrays, Arrays
    assigning
        Assignment, Assignment
    assigning
        Assignment, Assignment

Figure 15: Section names as anchors for links in index.

Ergänzungspakete, Struktur der
    2.5 Zur Struktur von to 2.5.6 Der Hauptteil
  Hauptteil
    2.5.6 Der Hauptteil | 2.5.6 Der Hauptteil | 2.5.6 Der Hauptteil | 2.5.6 Der Hauptteil
  Identifikationsteil
    2.5.1 Identifikationsteil
  Initialisierungsteil
    2.5.2 Initialisierungsteil

Figure 16: Section numbers and names as anchors for links in index.

<P>
<PARAM>
Parisi, Anthony
PASSWORD
    TYPE
PATH_INFO
PATH_TRANSLATED
Perl

Figure 17: Keywords as anchors for links in index.

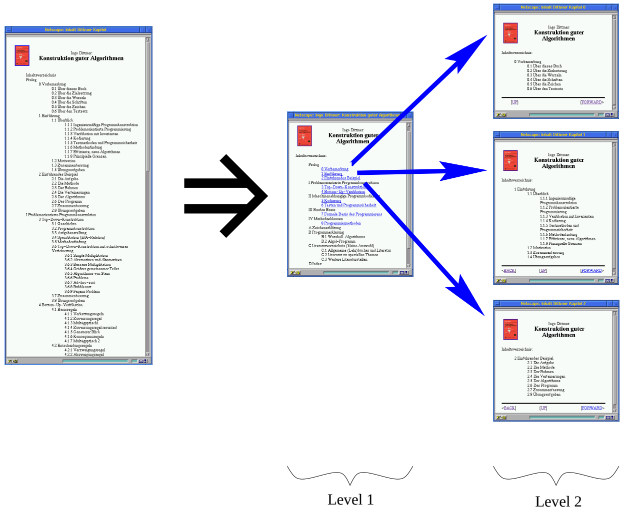

Figure 18: How to split the table of contents into two levels.

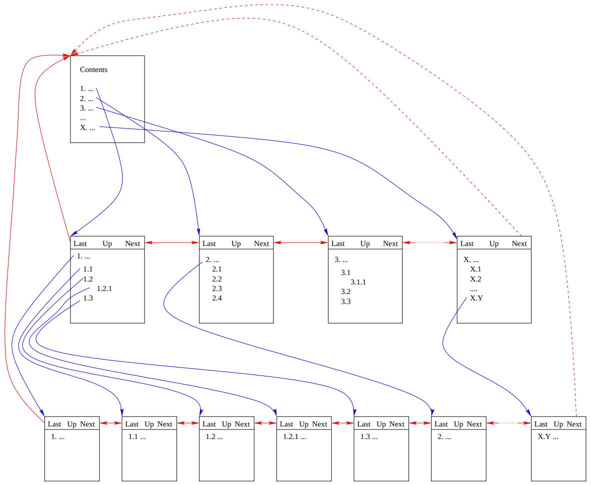

Figure 19: Impact of a second level of table of contents on navigation.

As for navigating, the chapter's tables of contents on the second level should be linked into a circular list (with references to the next/last chapter), just as regular pages are, see Figure 19.

As with the regular pages, a "back"-link from the first page and a "next"-link from the last page may be omitted. However, "up"-links should be available, pointing to the first level table of contents.

4.10 References

Documents may contain collections of references to books, papers, magazines and other type of information, that is references throughout the document. In the referenced section, it's possible to create links to other available documents, if they are available, their URL is know, *and* known to persist. Also, email-references (mailto:-links) may be inserted to contact an author of a book.

Furthermore, so-called "backlinks" can be inserted from the reference of a book back into the places in the document, where this book (or whatever) was references. This makes it easier to find out what for a certain book was being references for. When using the hyperref-macros for a LaTeX-to-PDF conversion, this can be done automatically by giving the "backref"-option when calling the hyperref macros:

1 2 3 4 7 A B C D E F G H I J K L M N O P Q R S T U V W Z

A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

[A] [B] [C] [D] [E] [F] [G] [H] [I] [J] [K] [L] [M] [N] [O] [P] [Q] [R] [S] [T] [U] [V] [W] [X] [Z]

Figure 20: Various character-jump-bars for an index

```
\RequirePackage[backref]{hyperref}
```

4.11 Index

The last part of a document is usually the index, containing keywords and references into the document, where these keywords first appear or are described. Many source formats and converters support index creation, either when converting to PDF or HTML.

When going for PDF from a LaTeXsource and using the hyperref macro package, a hyperlinked index can be generated by giving the "hyperindex" option:

```
\RequirePackage[hyperindex]{hyperref}
```

When going from FrameMaker to HTML using frame2html, this converter needs to be fixed to recognize index links at all, and then the index needs to be generated with the help of an external script. See the conversion report for Jörg Raasch's book for further details.[6]

As noted above during discussion of the look of links in the previous section, there are a number of ways to display links in the index, ranging from (virtual) page numbers, to images and chapter number/names as anchor, see Figures 13, 14, 15, 16 and 17.

Besides the look of the links (or better, their anchors) in the index, there's another point to pay attention to when generating an index, especially in HTML. When the index gets too long, it becomes slow to load and awkward to use. In such a case, at least a character-jump-bar has to be inserted, which allows jumping to the beginning of every set of index marks that begin with that character. The look of this bar may differ, but if present, it needs to be on the top of the index as well as at the end, similar to the navigation bar. Figure 20 shows some examples.

Of course, inserting a character-jump-bar just helps navigating inside the index, it doesn't make it shorter or speed up loading. If required, the index can be broken into a single file for every set of index words starting with the same letter, with a jump-table for every file as the starting point, similar to the table of contents with two levels, see 4.9

4.12 Titlepage

For every document, either converted to PDF or HTML, there needs to be a starting page in HTML. The page should contain a picture of the book's cover, and the text of the book's back. This titlepage should be designed after the guidelines of the publisher of the book, so all the books of one publisher look the same. If the conversion is into PDF, there should also be a table of contents in HTML besides the titlepage, so anyone interrested in the book doesn't have to download the whole document for having a first look on that's in there.

5 Communication and Exchange of Experiences

So far, a lot of topics have been addressed that are important when converting a document into either PDF or HTML. However, if one starts working on a conversion, it's easy to loose the overview. Therefore, it's good to have some documents available at which too look for reference, and to see how the one or other detail has been realized. If other people at the same site are working on converting documents, it can be very useful to not let them work independent of each other, but to give them a base for communications, either through a (local) mailing list or some meetings scheduled to inform everyone about the state, problems and solutions encountered.

If unsolveable problems arise, there's a wealth of information on document conversion collected in the conversion reports during the MeDoc project and hosted by Springer, see [8].

On the same site, there's also a list of email addresses of people who've worked on the various document conversions during the MeDoc project, sorted by source and target format they've converted from and to. So this may give another way of getting help when someone doesn't know how to handle a certain problem.

6 Workflow of a Conversion

After explaining the formats of electronic documents, how to convert them and by using which guidelines, by explaining what's important and where traps and pitfalls are, this section gives an overview of the actual workflow for a document conversion. Ranging from the aquisition of the document sources, ranging over to pinning the desired goal down and what two principal ways exist to realize the end product if the converter doesn't do its job 100%, until the final delivery of the produced document.

6.1 Obtaining Sources

The first thing before converting is getting the sources of the document to convert. They are usually obtained after consulting with the publisher or directly

from the author of the book in question. Regardless if the source comes from the author or the publisher, it's good to have both addresses for later feedback.

Ways to get the source include traditional media such as floppy disks and tapes, or electronic ways such as email or FTP. In the first case, it's very important that the media is in a format that can be read at the converters site. E. g. if a disk is shipped in Macintosh or Amiga file format, it's highly unlikely anyone can read it. The least common nominator for this job is probably still the MS-DOS filesystem, with files compressed in some way to retain the filenames, their case and the directory structure. The same goes with tapes, where one should agree on block size and backup software used before attempting to ship anything.

Problems can arise when using an electronic transport way, too, though. When sending through FTP or email, the files should also be compressed for the reasons given above. If the network in use is too slow, has a high rate of packet loss or connections are terminated often and on a random basis, the archive files containing the source should be chopped into little pieces of 1 megabyte, 100 or even 10 kilobytes, and each of these little pieces can then be transported seperately.

When getting the sources for the Weich-book, this method had to be applied, as the author of the sources lived in Austria, and the network between austrian schools and the german education network was then routed via the EBone network, going from Regensburg to Vienna via Duesseldorf, Amsterdam and Paris, and it was impossible to transfer the 15 megabytes file via FTP in one piece. Several attempts have been taken during all day- and night times, but no chance. The transfer always terminated after some random time. The file was then cut into tiny pieces and send to Regensburg via email, using the automatic retransmit-on-failure mechanisms employed in the mail protocol (SMTP). With this method, the little packets arrived one by one, slowly, though, but surely, until all pieces were there and could be glued together to a big archive again, which then could be processed. With today's Ten34 network connection european educational networks, this problem wouldn't arise anymore, but still it was a problem at the end of 1996!

After getting the source for a document to convert, it's important to check if it's complete, i.e. containing the files for all chapters, index, in case of TEX documents any additional font, class and style files as well as any bibliographic data and information on the order in which to process all the files, if necessary. Any omissions should be reported back to the supplier of the sources immediately, so these can be received as fast as possible. Also, if the source format is unknown this kind of information should (really! :-) be obtained, together with which tools may be helpful, e.g. with which software images were created, which platform the author has used, any special configuration, etc.

What's left to get is a printed version of the document. This can be received usually either from the local library, or if unavailable, often the publisher will supply an issue. The printed version is important before and after the conversion: before, so one can see what's coming up without going through all the source files, and having to figure out a probably non-trivial building process.

During and after the conversion, the paper version of the document is most helpful to compare if the converted document is similar enough to the printed version, and if every detail is realized in a satisfactory way.

6.2 Specifying the Goal

The next step after obtaining the sources is to make some decisions. The first and most important is into which format to convert the document in question, see 4.1 for a discussion.

After the target format is nailed down, other decisions have to be made on which language to use (if there are several), what colors to use, how to divide the chapters into small pieces (segmentation), design of the navigation bar, table of contents, index, as well as the display of cross references within the document. Guidelines and solutions for all these questions have been given in the previous chapter and won't be repeated here.

6.3 Intercepting the Converter

Doing the conversion is no problem if the conversion tools do what they should. But this is rarely the case. Often, the tools produce incorrect output, that can be fixed in one of two ways:

- early interception, i.e. tuning the converter
- late interception, i.e. fixing the output manually

Early interception in the conversion process means to recognize any errors produced by a converter, and to fix the converter to no longer produce this error. This is usually an iterating process, running the converter, fixing, running it again, ... until all bugs have been identified and fixed. There's still a number of fixed to do by hand afterwards, which are either not possible to fix in the converter or source at all, or which would cost too much time and knowledge to fix, both of which may not be available. When fixing the converter, this requires having the source code, which is usually no problem when using freely available products. When using commercial converters, this is a problem, though. Bug reports can be sent to the vendor, but usually the time until a fix comes out is way to long to wait for it. Nevertheless fixing the converter means one only has to do a change in one place, and not many places all over the document (possibly leaving something out by mistake). Another advantage of fixing the converter is that the made fix can be contributed to everyone having the same problem, resulting in the enhancement and support of freely available software.

In case the source for a converter is not available or one doesn't have the knowledge to modify it, the resulting output of the conversion step has to be post processed, either by hand or by a (set of) scripts which fix the mistakes the converter introduced. Often, these fixes can be applied by using a (multi-file) search-and-replace-operation, but sometimes the fix is not so easy. Some scripting language for processing the HTML file is needed then, usually perl,

as it's available on all major conversion platforms. With such a language, it's e.g. possible to find out the first heading (number and/or name) and to write it into the title of a HTML document, to find out which page or chapter number a given symbolic reference in a LaTeX of FrameMaker document corresponds to, etc. See [6] and [5] for some examples.

If it's not possible to employ such a language to fix a problem in all files, the output has to be fixed by hand. This is no problem if a mistake only occurs in a few places, but if a mistake occurs all over the document, fixing by hand is an endless waste of time that should be avoided if possible.

6.4 Feedback

Feedback by people not directly working on a document conversion may be helpful, as these people usually see problems not obvious to the converter himself. Also, the earlier there's feedback, the earlier any fundamental problems can be fixed, and the less damage is done by a wrong decision during the specification of the conversion goals.

Such helpful feedback may come from collegues either working on book conversions, too, or not, from any local supervisor or even from the project leader. The persons who have to be asked for in any case are the author of the book and the publisher, as it is their document that is being converted, and the work is being done for them.

As mentioned above, feedback is welcome at every point during document conversion, but when the conversion is done, the publisher and the author should have a look again to give their final ok before the document is delivered.

6.5 Document delivery

When everything is converted, the title page is done, etc. a final check should be done that all the files fit into one single directory which can be moved around without breaking anything (i.e. if there are no absolute links used by accident). If there's a reading rehearsal, it goes into it's own directory, just as the full version does. Any additional files (title page, table of contents, misc. images, ...) should be kept in one directory as well.

All the data can then be delivered via the same ways as the source of the document was received, either electronically or via traditional media, most probably in the form of compressed archive files when using electronical delivery. If there's a written conversion contract, it may specify this further.

If the publisher or project leader requires a conversion report, as was mandatory during the MeDoc project, this report should be included on the media for final delivery as well.

If a media such as CD-ROM is used for delivery, it should contain conversion report, title-page (and any auxiliary files), reading rehearsal and full version each in it's own directory, together with a README file describing which file/directory belongs to what, and (with directories full of HTML files) where the starting file is (if there is no index.html).

7 Conclusion

To sum up all the conversions, it can be said that the source texts obtained in the various formats can rarely be converted without doing some substantial manual work on them first; detailled guidelines for authors will help avoiding this. Also, guidelines for document converters as given in this document help a lot, as many experiences and ideas can be given without re-inventing the wheel. Tools are available, but they don't always do what they promise. Knowing their strengths and flaws is essential for efficient work.

Given the experiences in document conversion performed during the MeDoc project, it can be hoped that future documents comply to then-given standards for authoring, as well as existing tools get improved to leave less work that has to be done manually.

References

[1] Brüggemann-Klein, A., "Präsentation auf dem MeDoc-Verleger-Treffen anläßlich der Frankfurter Buchmesse", TU München.

URL: http://medoc.informatik.tu-muenchen.de/deutsch/organisation/-projektbibliothek/folien/brueggem.ps

[2] Adobe Systems Inc., "PostScript language tutorial and cookbook" (the blue book), Addison-Wesley, Reading, MA, 1985.

PostScript language tutorial and cookbook is the most common tutorial book on PostScript. It provides an easy, informal introduction to the PostScript language and graphics primitives. The cookbook is a collection of programs that are offered as examples of PostScript usage. These samples have been chosen both as illustrations of the functional range of PostScript and as useful ingredients for inclusion in application packages.

[3] Adobe Systems Inc., "PostScript language reference manual" (the red book), 2nd ed, Addison-Wesley, Reading, MA, 1990.

PostScript language reference manual is the book that defines the PostScript language. The second edition not only defines Level 1 PostScript, but also encompasses the color, composite font, file system, and DPS extensions and the PostScript language Level 2.

[4] Adobe Systems Inc., "PostScript Language Program Design" (the green book), Addison-Wesley, Reading, MA, 1988.

PostScript Language Program Design is intended to teach the fundamentals of designing PostScript language programs and to show how the language works, so the your programs will be fast, well-behaved, easy to understand, and portable.

[5] Graf, S., Eibisch, C.: "Konvertierungsbericht zum Weich-Buch", FH Regensburg, FB Informatik, Prüfeningerstr. 58, 93049 Regensburg, Germany.

URL: http://medoc.springer.de/konvertierung/konvertierungsberichte/-weich_eibisch_kurz.html, http://medoc.springer.de/konvertierung/-konvertierungsberichte/weich_eibisch_lang.html

[6] Feyrer, H.: "Konvertierungsbericht zum Raasch-Buch", FH Regensburg, FB Informatik, Prüfeningerstr. 58, 93049 Regensburg, Germany.

URL: http://medoc.springer.de/konvertierung/konvertierungsberichte/-raasch_feyrer.html

[7] Mayerhofer, J.: "Konvertierungsbericht zum Kopka, 1. Band", FH Regensburg, FB Informatik, Prüfeningerstr. 58, 93049 Regensburg, Germany.

URL: To be placed somewhere below http://medoc.springer.de/-konvertierung/konvertierungsberichte/ when ready.

[8] Various: Collection of all conversion reports given for the book conversions during the MeDoc project.

URL: http://medoc.springer.de/konvertierungen/konvertierungsberichte/

[9] Knuth, Donald E., "The TEX Book", Addison-Wesley, Reading, MA, 1986.

[10] Klorz: "Konvertierung des Buches 'Theoretische Informatik' von Werner Brecht ins HTML-Format - Erfahrungsbericht", Lehrstuhl Informatik 5, RWTH Aachen.

URL: http://medoc.springer.de/konvertierung/konvertierungsberichte/-brecht_sklorz.html

[11] Microsoft Product Support Services: "Rich Text Format (RTF) Specification", Microsoft Corp., 1994.

URL: http://night.primate.wisc.edu:70/0/software/RTF/RTF-Spec-1.3.ps

[12] Tim Bienz, Richard Cohn, James R. Meehan: "Portable Document Format Reference Manual, Version 1.2", Adobe Systems Incorporated, 1996.

URL: http://www.adobe.com/supportservice/devrelations/PDFS/TN/-PDFSPEC.PDF