

Making of ... 'Technik der IP-Netze' (3. Auflage)

Erwin Hoffmann

7. Mai 2016

1 Der Herr schuf am Anfang ...

1.1 ... Papyrus und Tinte

Nein nicht wirklich. Am Anfang stand – so Mitte der 90er Jahre – ein Buchprojekt über Netzwerke. Das Internet war erst im Entstehen. Und so hiess dies auch schlicht 'High Speed Internetworking' und sollte bei *Addison-Wesley* ein relativer Verkaufserfolg werden.

Die Welt damals war vom raschen Verseuchen mit Windows-Software gekennzeichnet. Es gab 'Coral Draw' und 'Page Maker'. Word gab es in der Version 2.0. Für die Unverbesserlichen gab es AMI Pro. Obwohl: Der Nimbus auch unter IBM's OS/2 laufen zu können, war eher ein Menetekel als ein Vorteil.

So ausgerüstet, beschlossen Prof. Badach, Olaf Knauer (sein Student) und ich dieses Buchprojekt. Mir ist heute noch unklar, wie man mit einer eher nicht bestehenden Informationsbasis (Zugang zu Originaldokumenten war schwierig, das Internet in der heutigen Form gab es noch nicht) so ein Projekt anfangen kann. Aber die Welt war im Aufbruch und lechzte nach Informationen.

Nach einer verkappten 2. Auflage, die nur ein Schatten der ersten war – das Buchprojekt sollte nun die Bestandteile 'LANs', 'WANs' und 'TCP/IP' sauber trennen (der zweite Band ist immer noch unveröffentlicht und behandelt damals so dringende Fragen wie 'X.25', 'Frame Relay' und 'ATM') – war nach dem Abmarsch von Addison-Wesley die Frage offen: Wie geht es weiter?

Prof. Badach nutzte seine Kontakte zum *Hanser-Verlag* und wir hatten Ende der 90er Jahre eine neuen Verleger ... und neue Herausforderungen.

2 TIPNv1 – die erste Auflage

Nachdem wir 'High Speed Internetworking' noch unter Windows 3.1 mit Word 2.0 gestaltet hatten, nutzte ich für den Satz nun Windows NT 3.51 sowie Word 6.0. Wow! Nicht mehr 1024 KByte Speicher sondern 4 MByte

und echte 32 Bit! Wie? Word 6.0 ist nur 16 Bit? Weit gefehlt. Zusammen mit Windows NT 3.1 brachte Microsoft ein echtes 32 Bit Word 6 heraus – sowohl in Englisch als auch in Deutsch. Und das Ding war brauchbar (trotz Beta-Status)! Kein HTML; kein UNICODE. Aber Textsatz und Import von EPS-Graphiken ging ohne Probleme. Eine Produktivitätsbombe.

Ein Problem blieb: Wie ist der Satz, also Seitenspiegel und auch die Fonts zu wählen, dass es den Vorstellungen des Verlags entspricht und so auch bei der Druckerei ankommt?

Der geneigte Leser sollte wissen, dass im Gegensatz zu Verlagen wie Springer, der Hanser-Verlag das komplett gesetzte Manuskript von den Autoren anfordert: Also PDF in die Druckmaschine und gut ist!

Im Umkehrschluss heisst dies, dass sich die Autoren über Satzspiegel, Layout, Grösse und Fonts, Farben und natürlich der Gesamtgestaltung des Werks Gedanken machen müssen. Da ist niemand im Backoffice, der diese Aufgaben übernimmt.

Vom Hanser-Verlag habe ich damals eine Word-Vorlage erhalten; keine Fonts – nichts. Die Word-Vorlage war fehlerhaft. Mittels [1] konnte ich letztlich ein sinnvolles Layout erstellen. Naja fast. Ein wichtiger Punkt waren damals Randmarginalien. Der Hanser-Verlag hat meine modifizierte Vorlage anschliessend für weitere Buchprojekte genutzt.

2.1 TIPNv2 – die zweite Auflage und Word 2000

Im Nachhinein produzieren wir alle 7 Jahre eine neue Auflage; sprich Buch. 2005 war es wieder soweit. IPv6 war in den RFCs komplett überarbeitet worden. Ein Update war fällig. Auf drängen meines Kollegen haben wir dann Word 2000 benutzt. Ein Desaster. Word 2000 gibt es in mindestens 5 verschiedenen Versionen. Schlimmer noch: Im Gegensatz zu Word 6(c) hat bei Word 2000 die 'interne' Dokumentenvorlage Vorrang vor der allgemeinen. Dass die aktuelle Formatierung (aus z.B. der Zwischenablage) die Einstellungen des mit einer Dokumentenvorlage verbundenen Dokuments (z.B. eines Absatz) überschreibt, ist mittlerweile eine Pest, die auch MacOS übernommen hat¹.

Schlussendlich mussten wir das (gesetzte) Buch dem Verlag zur Weiterverarbeitung übergeben, der dann z.B. das Index-Verzeichnis erstellt hat.

2.2 TIPNv2 – die dritte Auflage mit T_EX

Wer bereits die ersten Auflagen unseres Buches in den Händen hält, hat mit den ca. 600 Seiten und etwa ebenso vielen Abbildungen den Eindruck, dass dies kaum mit Word zu schaffen ist. Der Eindruck täuscht nicht. Und so

¹Soweit ich verstanden haben, wird der Text zur Kopie in die Zwischenablage generell HTML formatiert.

beschloss ich mit der dritten Auflage eine komplette Kehrtwendung Richtung $\text{T}_{\text{E}}\text{X}$.

$\text{T}_{\text{E}}\text{X}$ kenne ich sehr gut seit meiner Promotion. Meine Dissertation ist komplett in $\text{T}_{\text{E}}\text{X}$ (LaTeX gab es damals noch nicht) verfasst. Mit dem Aufkommen der PCs und dem WYSIWYG fiel dies dann doch ausserhalb meines Fokus.

Mit meiner Professur für Informatik an der FH Frankfurt hiess es zunächst meinen in Word und Powerpoint (3.0) bestehenden Datenbestand zu nutzen und zu konvertieren. Erste Versuche mit Softmaker und OpenOffice waren aber nicht nachhaltig. Die Wahl auf $\text{T}_{\text{E}}\text{X}$ war unausweichlich.

3 Wie mache ich $\text{T}_{\text{E}}\text{X}$?

Am Anfang steht die Frage des Editors. Ein Editor der

- $\text{T}_{\text{E}}\text{X}$ -Kommandos kennt und
- Syntax-Fehler minimiert

ist hochwillkommen. Meine Wahl fiel auf Pascal Brachet's 'TeXmaker'. Die richtigen $\text{T}_{\text{E}}\text{X}$ -Typen sind entweder Physiker (Michel Goosons, Peter Breitenlohner) oder Mathematiker (wie Pascal Brachet) ... oder?

TeXmaker² ist eine gute Wahl – kommt aber bei Filialdokumenten an seine Grenzen – und ist bei grossen Dokumenten aufgrund seines 'Multithreading' Konzepts kaum brauchbar.

Wo liegen die Problem bei TeXmaker? Meine persönliche Liste:

- TeXmaker erkennt aufgrund des

```
\usepackage[english]
```

nicht, welche Sprache das Dokument hat, sondern man muss das im Editor global für alle Dokumente umstellen.

- TeXmaker kennt kein 'Search & Replace' über mehrere Filialdokumente.
- TeXmaker's log Output ist praktisch unbrauchbar und aufgrund der multithreading Fähigkeit findet man Fehler in den Filialdokumenten auch nicht gut.
- Tritt ein Fehler bei $\text{T}_{\text{E}}\text{X}$ -compilieren auf, springt TeXmaker an den Anfang des Dokuments. Sehr lästig und unnötig.

²<http://www.xm1math.net/texmaker/>

Kurz gesagt: TeXmaker kam bei unserer Buchprojekt an seine Grenzen. Mehrmals musste ich mittels 'kill -term ...' das Programm dazu bringen, einen Vorgang zu beenden. Nicht unbedingt ein Fehler in TeXmaker; aber lästig in der gesamten Verarbeitungskette.

Unter MacOS X sind allgemein Text-Editoren nicht sehr üppig gesät³. Ja; Textwrangler⁴, BBEdit⁵ sind die Könige. Aber für T_EX? Im Grunde ist ja T_EX eine Programmiersprache für Worte: Klammern, Escape-Zeichen .. mehr braucht man eigentlich nicht. Schön ist ein Programm mit Syntax-Unterstützung. Ein Versuch mit Tincta⁶ endete böse: Das Programm zerschoss die ASCII-Datei und fort war sie. Der zweite Ansatz war Textmate⁷. Trotz Beta-Status funktioniert das Programm sauber und erzeugt in weiteren Prozessen (und nicht Threads!) sowohl das T_EX-Compilat als auch die Log-Datei. Toll und schnell – aber die Belegung von Short-Cuts war mir zu aufwändig. In der Bedienung ist TeXmaker unübertroffen; obwohl die Shortcuts manchmal klemmen und das underscore entfernt wurde.

3.1 Altlasten: Von .doc nach .tex

Ein Grossteil des Buchs lag ja im Word-Format vor. Nur wie verarbeitet ich das weiter unter T_EX? Kurz gegoogelt: Es gibt ein Windows-Programm 'Word2TeX'. Aber: Funktioniert als Plug-In bei einer ganz bestimmten Word-Version, die ich nicht habe. Lediglich AMI-Pro bietet eine sauber T_EX-Schnittstelle. Lang, lang ist's her.

Bei Word gibt es zwei Alternativen:

1. Export der Datei als ASCII. Geht.
2. Paste und Copy der Datei von Word nach TeXmaker. Geht.

Beim ersten Ansatz habe ich die Datei mit Textmate zunächst verarbeitet, weil das ein geniales Werkzeug zum Absatzumbruch bietet. Word exportiert eine Absatz als eine Zeile; und die kann seeehhr lang sein. Absätze werden zudem nicht – wie bei T_EX üblich – mit Leerzeilen getrennt.

Ein bisschen Acht geben auf hoch- und tiefgestellte Zeichen und die Formatierung und es geht. Ein Grossteil des Textes lässt sich gut konvertieren. Naja, fast.

³Unter Windows gibt es notepad++ und den genialen PFE (<http://www.lancaster.ac.uk/staff/steveb/cpaap/pfe/>)

⁴<http://www.barebones.com/products/textwrangler/>

⁵<http://www.barebones.com/products/bbedit/>

⁶<http://www.mr-fridge.de/software/tincta/>

⁷<https://macromates.com/>

3.2 Die Abbildungen

Die zweite Auflage hatte mehr hundert Abbildungen; teils mit sehr grosser Sorgfalt hergestellt. Wie importiere ich die in T_EX?

Antwort 1: Nicht als Bitmap. Das bläht das Buch auf. PNG und Konsorten scheiden daher aus.

Antwort 2: Die Bilder wurden meistens mit dem Programm *Micrografx* 'Disigner' (heute bei Corel) erstellt. Das Dokumentenformat ist eine Vektorgraphik. Export nach SVG? Fehlanzeige. SVG in T_EX auch Fehlanzeige! Was bleibt ist der PDF (= Postscript) Export.

Antwort 3: Die meisten Programme (so auch *Word*) machen PDF Export. Nur: Intern wird ein Bitmap über den Drucker/Graphik-Treiber erzeugt, der mit einer PDF-Hülle ummantelt wird: Unbrauchbar.

Antwort 4: Der Trick besteht darin, die Datei zu drucken und als PDF abzulegen. Die PDF-Erzeugung geht in der Regel über *Ghostscript* – und das macht es richtig. Die als Druckdatei hinterlegte PDF behält ihre Vektorgraphik Attribute. Nicht immer richtig; aber immerhin.

Antwort 5: Neben PNG versteht T_EX auch PDF als Graphikformat. Wer dies bereitstellt hat gewonnen.

Antwort 6: Die aus dem *Designer* extrahierten PDFs müssen aber angepasst und ggf. modifiziert werden. Die Graphiken waren teils sehr 'altbacken' und Fehler waren auch nicht unüblich. Wir brauchten also ein Programm, dass PDF-Dateien sinnvoll als Vektorgraphiken weiter verarbeiten kann – und wieder per PDF verfügbar macht.

Lösung 7: *Omnigraffle*⁸ (Pro). Hätte es diese Programm nicht gegeben, wäre die dritte Auflage des Buchs nicht zustande gekommen. Die ca. 100 Euro für eine Lizenz des Programms haben sich mehr als gelohnt. OpenOffice bzw. LibreOffice erwiesen sich beim PDF-Export als Katastrophe.

Was jetzt noch fehlte, war ein Konzept, wie die Zeichnungen aufzubereiten sind und wie vor allem auch Elemente wie Farben und Verläufe einheitlich dargestellt werden sollten. Aufgrund der Vielzahl der Zeichnungen war dies aber nicht immer konsistent. *Omnigraffle* liefert zwar eine 'Library' in Form von 'Stencils'; aber eine 'Dropbox' für gängige Elemente war in der Version 5.x nicht vorgesehen. Exemplarisches leistet da z.B. Papyrus⁹.

4 Der Satz: Das T_EX-Layout

Wer von euch hat schon mal ein Buch komplett layoutet – also Satzfertig? Ich gebe zu, dass war mein 5ter Versuch – und der ist immer noch anspruchsvoll. Was gilt es zu berücksichtigen?

- *Satzspiegel:* Der Text muss in den 'Rahmen' passen.

⁸<https://www.omnigroup.com/omnigraffle>

⁹<https://www.papyrus.de/>

- *Fonts*: Das Buch soll gut lesbar sein. Keine Brüche in den Fonts und sinnvolle Fontgrößen.
- *Umbruch*: Keine halbleeren Seiten, Graphiken an der richtigen Stelle.
- *Rechtschreibung*: Technische Termini sollen eindeutig und im Buch identisch verwendet werden.
- *Proportionen*: Zeichnungen, Text und Fonts der Zeichnungen sind aufeinander abzustimmen, damit sie homogen wirken.

Dazu kommen noch folgende 'Kleinigkeiten':

- Wie gestalte ich das Inhaltsverzeichnis sinnvoll?
- Wie sieht es mit dem Abbildungs- und Tabellenverzeichnis aus?
- Und die Quellen, sprich Literaturverzeichnis?
- Nicht zuletzt: Für die Leser der gedruckten Ausgabe das Index-Verzeichnis?

Ich habe ja bereits erwähnt, dass das Buch aus älteren Quellen stammt, die nicht unbedingt 'TeX-compatibel' sind. Im Gegensatz zu einheitlichen Werkzeugen, wie *Framemaker*¹⁰ oder *InDesign*¹¹ ist die Umgestaltung nicht trivial. Schauen wir uns mal an, wie wir das gelöst haben.

TeX-Layout

Das Layout von TeX ist technisch, aber nicht unbedingt ästhetisch. Um das letzte zu gewährleisten gibt es TeX-Klassen, von denen wir die Klasse 'Superklasse' zumindest partiell genutzt haben.

Der TeX-Einstieg in unser Buch sieht daher folgendermassen aus:

```
\documentclass[%
openright,%
twoside,10pt,%
numbers=noenddot,%
%DIVcalc,%
toc=index]{scrbook}
```

Eigentlich nichts besonders: 2-seitiges Layout; Default-Fontgröße ist 10 Punkt. Bei der Nummerierung wird auf einen abschliessenden Punkt verzichtet. Der Verweis auf den Index wird ins Inhaltsverzeichnis aufgenommen. Die Defaults werden aus der Klasse 'srcbook' abgeleitet.

¹⁰<http://www.adobe.com/de/products/framemaker.html>

¹¹<http://www.adobe.com/de/products/indesign.html>

```

\renewcommand{\labelitemi}{${\color{gray!80}\raisebox{.1ex}{\scalebox{.70}{$\blacktriangleright$}}
\renewcommand{\labelitemii}{--}
\renewcommand{\labelitemiii}{$\triangleright$}
%
\let\origitemize\itemize
\def\itemize{\origitemize\itemsep -3pt}
\def\itemize{\origitemize\setlength{\topsep}{1ex plus0.5ex minus0.5ex}\itemsep -3pt}
%
\let\origenumerate\enumerate
\def\itemize{\origitemize\itemsep -3pt}
\def\enumerate{\origenumerate\setlength{\topsep}{1ex plus0.5ex minus0.5ex}\itemsep -

```

Der Satzspiegel beschreibt, wie der Seitenaufbau aussieht: Das zu beschreibende Testfeld, die (leeren) Kanten recht und links (wir haben ja ein Buch, das aufgeschlagen werden will), die Randmarginalie, der Bereich der Überschriften und die der Fussnoten und ggf. Seitenangaben.

Gut, der gewählte Satzspiegel ist konservativ:

- Die Seitenüberschriften enthalten Angaben über `section` und `subsection` sowie die Seitennummer.
- Wir nutzen Randmarginalien, um Aussagen und Terme in den Abschnitten hervorzuheben.
- Der untere Teil des Seitenspiegels wird nur für Fussnoten (wenige!) verwendet.

Die Irritation, die sich für den normalen $\text{T}_\text{E}\text{X}$ -Benutzer ergibt ist, dass es für jedes Satz-technische Problem mehrere Lösungen gibt. So haben wir z.B die Darstellung der Seiten-Überschriften nicht über die Möglichkeiten des Koma-Scripts gelöst, sondern selbst per Package `fancyhead` Hand angelegt:

```

% -- Header Zeilen
%
\usepackage{fancyhdr}
\pagestyle{fancy}% muss vor \renewcommand{\sectionmark} stehen
\fancyhf{}
\fancyfoot{} % Hack geht nicht (Seite-Nr auf Kapitel Seite)
\fancyhead[EL]{\sffamily \thepage}% gerade Seiten, links
\fancyhead[ER]{\sffamily \leftmark}% gerade Seiten, rechts
\fancyhead[OL]{\sffamily \rightmark}% ungerade Seiten, links
\fancyhead[OR]{\sffamily \thepage}% ungerade Seiten, rechts
\renewcommand{\chaptermark}[1]{
% \markboth{\thechapter.\ #1}{
\markboth{\thechapter{} #1}{

```

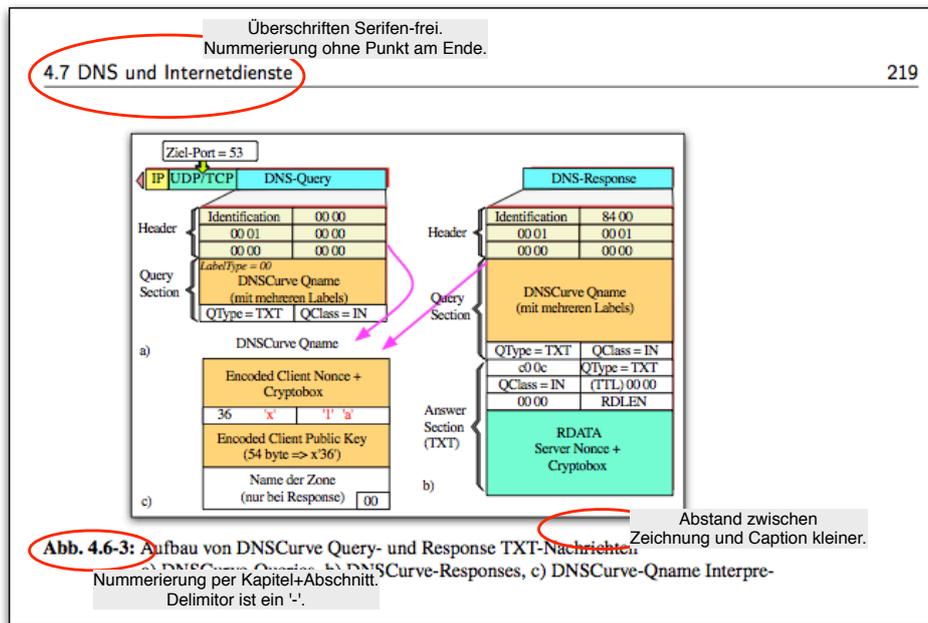


Abbildung 1: Angepasste Merkmale für Überschriften, Nummerierung und Abstände

```

}
\renewcommand{\sectionmark}[1]{
\markright{\thesection{} #1}{-}
}

\fancypagestyle{plain}{%
\fancyhf{} % clear all header and footer fields
\fancyfoot{} % except the center
\renewcommand{\headrulewidth}{0pt}
\renewcommand{\footrulewidth}{0pt}
}

```

Abbildung 1 zeigt das Ergebnis und gibt noch Erläuterungen, wie die Zähler für Section, Subsection und die Nummerierung eingesetzt wurde.

Aus diesem Beispiel ist deutlich erkennbar, dass T_EX eigentlich eine Programmiersprache für Worte ist, wo Variablen (#1) definiert, zugewiesen und auf diese zurückgegriffen werden kann.

Mit den Randmarginalien gab es einen besonderen Spass: T_EX hat eigentlich überhaupt kein Verständnis für Randmarginalien und diese müssen quasi als 'Aliens' in den Text eingefügt werden, indem neben dem Haupttext eine extra Spalte eingefügt wird. Die Positionierung der Zeichen in dieser Spalte kann erst postum, also nach Erzeugung des Haupttexts bestimmt

werden. Werden Einträge in der Randmarginalie erzeugt oder der Umbruch ändert sich von der linken auf die rechte Seite, sind zumindest 2 T_EX-Läufe notwendig damit das Ergebnis stimmt ... fast.

Am Anfang sieht das noch sehr übersichtlich aus:

```
\setlength{\marginparwidth}{20mm}
\setlength{\marginparsep}{4mm}
\addtolength{\headwidth}{\marginparsep}
\addtolength{\headwidth}{\marginparwidth}
```

Damit ist also die Lage der Marginalie bestimmt. Nur: Die Marginalie wird beim Seitenumbruch nicht berücksichtigt! Es bleiben somit das, was die Schriftsetzer früher als 'Hurenkinder' und 'Schusterjungen' bezeichnet haben. Wer unser Buch aufschlägt, findet diese nicht. Um dies zu vermeiden muss, der T_EX-Kernel angepasst werden. `makeakletter` und `makeatother` sind hier die 'magic commands', von denen ich vor diesem Buchprojekte noch nichts gehört hatte.

Was mir auch nicht bei T_EX gefallen hat, sind die Abstände zwischen Überschriften und Abschnitt, das Einrücken der ersten Zeile und ... die Darstellung von Listen.

Die Abstände bei Abschnitte relativ zu den Überschriften lassen sich global recht einfach umdefinieren:

```
% -- Absaetze
%
\parindent=0pt
\parskip=5pt
\topsep=-15pt
```

Bei den Aufzählungen, d.h. nummerierten und 'Bullditem'-Listen wird es etwas komplizierter. Wir haben folgendes genutzt, um die 'Word'-ähnlichen Boxen zu erzielen:

```
\renewcommand{\labelitemi}{\color{gray!80}%
\raisebox{.1ex}{\scalebox{.70}{\blacksquare}}}\$}
\renewcommand{\labelitemii}{--}
\renewcommand{\labelitemiii}{\triangleright}
%
\let\origitemize\itemize
\def\itemize{\origitemize\itemsep -3pt}
\def\itemize{\origitemize\setlength%
{\topsep}{1ex plus0.5ex minus0.5ex}\itemsep -3pt}
%
\let\origenumerate\enumerate
\def\itemize{\origitemize\itemsep -3pt}
```

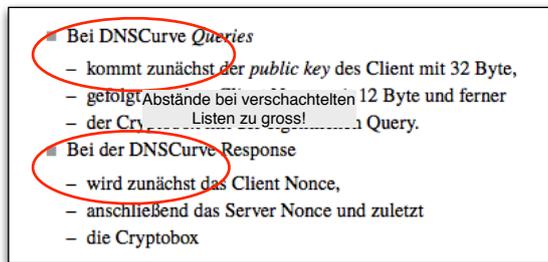


Abbildung 2: Zu grosse Abstände zwischen den Zeilen verschachtelter Liste

```
\def\enumerate{\origenumerate\setlength%
{\topsep}{1ex plus0.5ex minus0.5ex}\itemsep -3pt}
```

Verschwiegen habe ich hier die notwendigen $\text{T}_{\text{E}}\text{X}$ -Packages für Farben und Symbole.

Nur: Gut ist das alles nicht. Wer sich Aufzählungen im Buch kritisch anschaut, wird feststellen, dass die Abstände nicht abhängig vom Grad der Einrückung sind, sondern konstant. Das schaut dann folgendermassen aus [Abbildung 2]:

Bis heute habe ich keine (einfache) Lösung gefunden. Klar, man kann die Hierarchie (und die Parameter für den Satz) explizit angeben; aber $\text{T}_{\text{E}}\text{X}$ berücksichtigt das nicht dynamisch.

4.1 Wahl der Fonts und der leidige Umbruch

Die Frage der Fonts, also der Buchstabenformen hat eine lange Geschichte. Angefangen vom ersten Schriftsatz in Blei mit speziellen Schriftzeichen am Kapitelanfang ... bis; ja dem Font 'Comic Sans' von Microsoft, der häufig als die Verballhornung jeder Font-Ästhetik bezeichnet wird.

Ehrlich gesagt: Ich mag (wie viele meiner Kollegen vom CERN) Comic Sans. Er erzeugt nämlich Aufmerksamkeit, da 'gegen den Strich gebürstet'. Das einfache, unangestregte Lesen von Text ist die zentrale Anforderung an einen guten Font. Generell wird unterschieden zwischen Fonts mit 'Schnörkeln' an den Buchstaben – also Serifen – oder Serifen-freie Fonts.

Times (New) Roman und *Arial (Swiss, Helvetica)* sind typischer Vertreter je einer Philosophie. Für das weniger anstrengende Lesen sind Fonts mit Serifen aufgrund der erhöhten Redundanz besser geeignet. Das Mischen unterschiedlicher Font-Familien und -Typen sollte keinen Bruch im Text verursachen.

Leider kommen da noch erhebliche technische Schwierigkeiten dazu. Trotz umfangreichen Bemühens konnte ich z.B. Comic Sans nie für meine $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ -Beamer Folien nutzen.

In unserm Buch habe ich es daher mit Schlichtheit gehalten. Meine Wahl fiel auf:

```
% -- Fonts
%
\usepackage{lmodern} % Überschriften besser
\usepackage{tgtermes} % bestes Schriftbild
%\usepackage{palatino} % nicht so schön + 8% mehr Seiten
```

Palatino war der Font, den uns z.B. Addison-Wesley vorgeschrieben hat.

Kerning

Wie man im Ausschnitt oben sehen kann, habe ich mit verschiedenen Fonts experimentiert; auch auf dem Hintergrund der Kompaktheit des Buchs, d.h. unter Einhaltung des Umfangs von 800 Seiten. Hierbei kommt auch dem 'Kerning' ein wesentlicher Beitrag zu. 'Kerning' ist das Einrücken von Zeichen die Nahe beieinander stehen. Beispiele sind 'll', 'Sch' und andere. Sind die Abstände zu gross, wird das Schriftbild unregelmässig.

TeX kann das Kerning schon sehr gut; aber mit folgendem Trick wurde es dann doch noch besser:

```
\usepackage[activate={true,nocompatibility},
%spacing=nonfrench, %
final,tracking=true,kerning=true,spacing=true, %
factor=1100,stretch=10,shrink=10]{microtype} % whow
\microtypecontext{spacing=nonfrench}
```

Font-Grössen

Wie dargestellt, war das Einhalten des Umfang des Buchs eine zentrale Anforderung des Verlags. Problematisch war, dass in der Entwicklung des Inhalts immer mehr Stoff untergebracht werden sollte. Die Font-Grösse kann auch nicht beliebig verkleinert werden. Während bei Briefen eine Font-Grösse von 12 Pt (Punkten) üblich ist, sind Bücher häufig in 11 Pt gesetzt.

Das war für unser Buch zu gross. Nicht nur, dass die Seitenzahl nicht zu halten gewesen wären; auch das Gleichgewicht zwischen Texten und Abbildungen wäre verloren gegangen. Dies galt auch für die Tabellen. Eine Font-Grösse von 10 Pt erschien daher angemessen.

Nachdem das Buch weitgehend fertig gestellt war und die ersten Probeausdrucke vorlagen, kam dann der Schock: Zwischen Kapitel 4 und 5 stimmen die Font-Grössen nicht, was auf den Einzelausdrucken der Kapitel nicht aufgefallen war. Was war passiert?

Im Grunde genommen ganz einfach: Ein nicht abgeschlossenes Statement

```
\begin{small}
\end{small}
```

verursacht durch eine fehlende Klammer.

Der Fehler war beim Kompilieren mit TexMaker untergegangen. Er sorgte aber dafür, dass ab Kapitel 5 die Fonts kleiner waren (T_EX verkleinert mittels dieser Anweisungen die nachfolgenden Fonts um einen Pt) als in den ersten vier Kapiteln. Da die Seitenzahl bereits auf die Forderungen des Hanser-Verlags angepasst war, blieb nur eine Wahl: Nach Beseitigung des Fehlers müssen die gesamten Fonts per T_EX-Kernelpatch so skaliert werden, dass sowohl die Lesbarkeit als auch der Umfang des Buchs stimmt. Ergo: Die Default-Font-Grösse ist 9.6 Pt. Nicht schön, aber praktikabel.

4.2 Bilder und Tabellen

Wer das Problem mit 'float'-Objekte in T_EX kennt, weiss, dass T_EX einen eigenen Willen zur Positionierung von Bildern besitzt. Es gibt zwar Packages, mit denen die Bilder an genauer dieser Stelle festgenagelt werden können, aber das ist weder im T_EX-Sinne noch für grössere Texte praktikabel.

Abbildungen

Eine typische Abbildung im Buch wird folgendermassen eingebettet:

```
\begin{figure}[h!]  
\centering  
\includegraphics[width=0.9\textwidth]{./Kap_09/Bild/9-3_06.pdf}  
\caption[Aufbau eines AYIYA-Header und Einsatz bei UDP und TCP]  
{Aufbau eines AYIYA-Header und Einsatz bei UDP und TCP\  
\footnotesize{HashM: HashMet, AuthM: AuthMet; weitere Abkürzungen siehe [\ref{f9-3-  
\label{f9-3_06}  
\end{figure}
```

Für die T_EX-Spezialisten brauche ich nicht zu wiederholen, dass das '!' im `figure` Statement aussagt: 'Liebes T_EX blende das Objekt (möglichst) genau an dieser Stelle ein!'

Beim `includegraphics` Statement halte ich es so, dass ich die Grösse der Graphik auf 80% oder – wie im Beispiel – auf 90 % der Seitengrösse beschreibe; also nicht absolut wie mit einem `scale`.

Das `caption`-Statement ist zweigeteilt: In der rechteckigen Klammer kommen die Aussagen hinein, die später im *Abbildungsverzeichnis* erscheinen soll – auf das der Verlag im Buch verzichtet hat. In den geschweiften Klammern ist der eigentliche Text zu sehen; wobei ich Abkürzungen nochmals eine Pt kleiner gesetzt habe.

Zur Referenzierung von Abbildungen und Tabellen habe ich mir ein eigenes Makro gebastelt; wie zu sehen `fref` und logischerweise `tref`.

Zu erwähnen ist noch, dass die Abstände und die Position der *Captions* (natürlich) speziell angepasst sind [Abbildung 1]:

```

% -- Graphik
%
\usepackage{graphicx}
%\usepackage{float}
\usepackage{caption}
%\setlength\belowcaptionskip{-2pt}
\setlength{\intextsep}{8pt plus 2pt}

```

TeX liefert hier ansonsten auch ein eher unbefriedigendes Ergebnis.

Tabellen

Wer die Tabellen im Buch gesehen hat, weiss, dass die eher von bescheidenem Aussehen sind. Gelegentlich habe ich aus von `tabular` Gebrauch gemacht, was aber nicht erläutert werden muss.

Tabellen in TeX haben wiederum ihr eigenes (Font-Grössen) Environment. Generell plädiere ich dafür, entweder das Package `tabularx` oder `tabulary` zu verwenden, da diese den Umbruch in Zeilen zulassen. Um verschiedene Textgrössen zu ermöglichen, habe ich folgende Definition genutzt:

```

% -- Tabellen
%
\usepackage{tabularx}
\usepackage{tabulary}
\usepackage{multirow}
\usepackage{longtable}
\usepackage{colortbl}
%\usepackage{multicolumn}
%
\newenvironment{tabular10}{\fontsize{8}{10}\selectfont\tabular}{\endtabular}
\newenvironment{tabularx10}{\fontsize{9}{10}\selectfont\tabularx}{\endtabularx}
\newenvironment{tabulary10}{\fontsize{8}{10}\selectfont\tabulary}{\endtabulary}
\newenvironment{tabular9}{\fontsize{7}{9}\selectfont\tabular}{\endtabular}
\newenvironment{tabularx9}{\fontsize{7}{9}\selectfont\tabularx}{\endtabularx}
\newenvironment{tabulary9}{\fontsize{7}{9}\selectfont\tabulary}{\endtabulary}
\setlength{\tabcolsep}{2pt}

```

Abgesehen von der Unterstützung von Farben, habe ich die Tabellen mit `tabcolsep` etwas enger gesetzt als üblich. Mittels der `fontsize` Option, sind die Grössen der Zellentexte und die der Überschriften definiert.

4.3 Inhaltsverzeichnis, Quellen und Index

Ehrlich gesagt, habe ich mich um die Gestaltung des Inhaltsverzeichnisses nicht sonderlich bemüht, sondern das TeX machen lassen.

Wichtig war mir hier die logische und technische Unterteilung des Buchs in vernünftige Abschnitte. Im Haupt-TeX-File sieht das dann folgendermaßen aus:

```
%----- doc specifics -----
\begin{document}

\pagenumbering{Roman}
\setcounter{page}{5}

\tableofcontents
\include{./Kap_00/kap_00} % Vorwort

\clearpage
\pagenumbering{arabic}

\setpartpreamble[u,r][0.5\textwidth]{ %<preamble>
\vspace{200pt}
Wilkommen in CSNET!
\medskip

Michael,
\smallskip

This is your official welcome to CSNET.
We are glad to have you aboard. \\
\hrule
\smallskip
From: Laura Breeden \\
breeden\%csnet-sh.arpa@csnet-relay.csnet \\
To: rotert\%germany@csnet-relay.csnet \\
Via: csnet-relay; 3 Aug 84 10:44-MET}

\part{Klassisches IPv4/UDP/TCP}
\include{./Kap_01/kap_01} % Einführung
\include{./Kap_02/kap_02} % Alte Welt
\include{./Kap_03/kap_03} % Layer 4
\include{./Kap_04/kap_04} % DNS
\include{./Kap_05/kap_05} % DHCP; APIPA, NAT, IPSec
```

Also, es gibt *Parts*, *Sections*, *Subsections* und *Paragraphs* für die Gliederung, wobei die letzteren keine Einzugs ins Inhaltsverzeichnis finden.

Wichtig war mir die Untergliederung in Abschnitte (*Parts*) und ein kleiner Einleitungstext, wie oben beschrieben. Bei den *Paragraphs* habe ich TeX ein wenig auf die Sprünge geholfen.

Inhaltsverzeichnis	Inhaltsverzeichnis mit römischen Ziffern	IX
6.4.3	Aufbau von SIP-Adressen	320
6.4.4	Funktion eines SIP-Proxy bei der IP-Videotelefonie	322
6.4.5	Trapezoid-Modell von SIP	323
6.4.6	Unterstützung der Benutzermobilität bei SIP	325
6.4.7	Beschreibung von Sessions mittels SDP	327
6.5	Multipath TCP (MPTCP)	330
6.5.1	Typischer Einsatz von MPTCP	331
6.5.2	Transportschicht mit MPTCP	333
6.5.3	Multipath TCP im Hypertext-Verlinkung	336
6.5.4	MPTCP-Angaben im Hypertext-Verlinkung	340
6.5.5	Aufbau einer MPTCP-Verbindung	342
6.5.6	Anpassung des TCP-Headers für MPTCP	343
6.5.7	Abbau einer MPTCP-Verbindung	344
6.5.8	Middleboxen als Störfaktoren bei MPTCP	346
6.6	Schlussbemerkungen	347
II	Internet Protocol Version 6	349
7	Das Protokoll IPv6	351
7.1	Neuerungen bei IPv6 gegenüber IPv4	352
7.2	Header-Struktur bei IPv6	353
7.3	Erweiterungs-Header	355
7.4	IPv6-Flexibilität mit Options-Headern	359
7.4.1	Aufbau von Options-Headern	359
7.4.2	Belegung des Option-Feldes	360
7.5	Einsatz von Jumbo Payload	362
7.6	Source Routing bei IPv6	362
7.7	Fragmentierung langer IPv6-Pakete	364
7.8	Aufbau von IPv6-Adressen	365
7.8.1	Darstellung von IPv6-Adressen	366
7.8.2	IPv6-Adressensystematik und -Gültigkeitsbereiche	369
7.8.3	Interface-ID in IPv6-Adressen	370
7.8.4	Interface-Index bei Link-Local IPv6-Adressen	372

Abbildung 3: Inhaltsverzeichnis in T_EX

```

\def\paragraf#1{%
% \vspace{\ShiftPT}%
\medskip%
\begin{normalsize}
\textsf{\textbf{#1}}
% \label{\thechapter.\thesection: #1}
\end{normalsize}
}

```

Abbildung 3 illustriert, wie T_EX das Inhaltsverzeichnis aufbaut. Nicht schön, aber praktisch. Text und alle Abbildungen, einschliesslicher aller Referenzen sind per Hyperlinks indiziert und durchsuchbar.

Ansonsten 'kleben' die Überschriften der Paragraphen zu sehr am nachfolgenden Text. `ShiftPT` ist eine Variable die ich gezwungenerweise einführen musste, da ja die Fontgrösse 9.6 Pt ist und die diesen Wert trägt. Jedes Filialdokument muss mit diesem Wert 'getagged' werden.

Quellen

Bei den Quellen habe ich das so gehalten, wie unter [2] beschrieben. Beim Erstellen des Literaturverzeichnis gab es dann wieder Überraschungen, die ich nicht erwartet hatte.

Zunächst verfügt T_EX über mehrere Literatur-Backends. Der Standard `bibtex` kommt mit Umlauten nicht klar (UTF-8). Dies ist ein Kennzeichen vieler T_EX-Packages, der unbedingt aufgeräumt gehört.

Dann besteht das Problem darin, wie die Ausgabe der Literaturquellen im Buch erfolgen soll. Wir haben im Buch folgenden Eintrag:

```
[FMNORS11] I. Fikouras, C. Mulligan, R. Noldus, U. Olsson, A. Ryde und M. Stille.  
IMS Application Developer's Handbook: Creating and Deploying Innovative IMS Applicat
```

Sofern ich mich richtig erinnere, machte T_EX daraus ein `Fik11` ...

Nachdem ich alle Sonderzeichen aus dem T_EX-Dateien entfernt hatte und nur nur UTF8 (und keine UTF8x) Zeichen in den Dokumenten hatte, konnte ich schliesslich mit

```
%----- Bibliographie -----
```

```
\usepackage[backend=bibtex,bibencoding=utf8,style=alphanumeric]{biblatex}  
\addbibresource{TIPN03.bib}
```

glücklich werden.

Indices

Die Erstellung eines qualifizierten Index habe ich von Anfang an unterschätzt; und das Ergebnis im Buch ist daher eher bescheiden. Generell muss die Erstellung des Index als eigenes Projekt aufgefasst werden, was ähnlichen Umfang wie der Inhalt des Buchs selbst aufweist.

Ich hatte angenommen, über die Randmarginalien quasi automatisch einen Index erstellen zu können. So hatte auch die Einträge hierin programmiert. No way.

Als nächstes hatte ich die Idee, quasi zunächst ein Meta-Index-Verzeichnis zu erstellen und dann den Inhalt der Dateien dem gegenüber laufen zu lassen und Einträge in den Dateien automatisch zu erstellen. Ja, wäre gegangen: Aufwand ca. 1 Mann-Monat.

T_EX liefert aber einen hierarchischen Index. Dieser liesse sich ohne ergänzende Intelligenz nicht abbilden. Wäre aber ein cooles T_EX-Projekt, was unter den Titel 'Complex Event Processing' feile.

Daher war (wieder mal) manuelles Eingreifen erforderlich. Das Ergebnis unseres Indexes ist daher – obwohl umfangreich – weit entfernt von Systematik.

4.4 Trennung und Rechtschreibung

Die Worttrennung in T_EX immer noch unbefriedigend. Gerade bei zusammen geschriebenen Substantiven (per Bindestrich) tut sich T_EX schwer. Natürlich habe ich eine Hyphenation Liste geführt:

```
%----- hyphenation -----
```

```
\hyphenation{  
Ver-bin-dungs-orien-tiert  
An-wen-dun-gen  
Standard-an-wen-dun-gen  
In-for-ma-tion-en  
in-for-ma-tion  
Vor-stel-lung  
Re-ferenz-modell  
Re-ferenz-modells  
OSI-Re-ferenz-modells  
Check-summe  
Hash-summe
```

Aber auch hier gilt: Eigentlich sollte die Software das beherrschen. Dass sie es nicht tut, liegt wohl in der fehlenden Logik der Deutschen Sprache begründet.

Rechtschreibung

Ich bin mit der alten Rechtschreibung 'gross' geworden. In Deutschland ist die Verwendung der Sprache immer noch ein Klassenprivileg. Die Rechtschreibung ist im Hinblick auf Logik kaum nachvollziehbar.

Mit den Schweizern halte ich es so (und das liegt nicht zuletzt daran, dass ich für meine Schreibmaschinenkurs ein schweizerisches Gerät Zuhause hatte, wo das 'ß' fehlte), dass ich diesen Buchstaben für komplett überflüssig halte.

Dem Hanser-Verlag war mein Stil nicht beizubringen. Allerdings hat der Korrektor (Herr Dubau) seine Arbeit ernst genommen und wirklich geholfen, nicht nur Fehler zu finden, sondern auch Ausdrücke zu systematisieren.

Als Unix-Anwender habe ich das dann auch wieder automatisiert, und T_EX sei Dank mittels einiger `sed`-Skripten realisiert. Immer wenn es eine Festlegung auf eine Schreibweise gab (die im Manuskript an *einer* Stelle angemerkt wurde) habe ich dies in *allen* Filialdokumenten übernommen:

```
#!/bin/sh
```

```
TEXFILES=$(find ../ -name "kap_*.tex" | grep -v Dubau)  
INPUT=$1
```

```

if [ -f "${INPUT}" ]; then
    echo "Using sedfile $INPUT ..."

    echo $TEXFILES

    for FILE in $TEXFILES; do
        if [ -f $FILE ]; then
            echo "Index File: $FILE"
            cp ${FILE} ${FILE}.keep
            sed -f ./${INPUT} $FILE > ${FILE}.new \
                && mv ${FILE} ${FILE}.old && mv ${FILE}.new ${FILE}
            L=$(diff ${FILE} ${FILE}.old | grep -c "^>")
            echo "Replaced pattern in $FILE on $L lines."
        fi
    done
else
    echo "No sed Input file given ..."
    exit 2
fi

```

Die Erzeugung einer Kopie hat mir bei einem manchmal notwendige 'Roll-Back' das Leben gerettet ... Typische 'kanonische' Umsetzungen waren z.B.:

```

s;Web-Browser;Webbrowser;g
s;Web-Server;Webserver;g
s;Web-Seite;Webseite;g
s;des Internets;des Internet;g
s;Internet-Zugang;Internetzugang;g
s;Internet-Verbindung;Internetverbindung;g
s;weiss ;weiß ;g
s;muß;muss;g
s;Buchs;Buches;g
s;die Schwitche ;die Schwitches ;g

```

Naturgemäss schießt `sed` auch (bei mangelnder Information) über das Ziel hinaus und verändert Ausdrücke, die eigentlich so bestehen sollen. So geschehen beim Begriff des 'Mediums' bzw. der 'Media'.

Was ich im Buch vermieden haben, sind die englischen Pluralformen: Es gibt also nur Bit und Byte; keine Bits und Bytes.

5 Zusammenfassung

Obwohl \TeX -er der ersten Stunde, besitzt \TeX nunmehr aufgrund der Packages eine so grosse Komplexität, die schier undurchschaubar ist. Die mangelnde durchgehende UTF-8 Unterstützung ist ein weiteres grosses Problem.

Obwohl ich in \LaTeX gearbeitet habe, wäre vielleicht \XeLaTeX eine bessere Alternative gewesen. Allerdings geht mein Kenntnisstand hierüber gegen Null; was eine enorme Lernkurve bedeutet hätte.

Und genau ist das Problem mit \TeX : '50 Ways to leave your Lover'. Und der heisst nunmal 'Word'. Der neue Lover muss mit seinen guten Eigenschaften erkennbar sein. Dazu gehört Dokumentation, die aber einfach konsumierbar sein muss.

Bei der Nutzung von \TeX ergibt sich ein weiteres Problem: Die Fehlermeldungen müssen verständlich und umsetzbar sein. Auch fehlt ein Editor, der eine Syntax-Überprüfung bei wichtigen Kommandos vornimmt.

Beim Vergleich der Word-Variante (TIPNv2) mit der \TeX -Variante (TIPNv3) fällt aber der Fortschritt durchaus ins Auge. Und deshalb hat sich der Umstieg gelohnt.

Danke Donald K.!

6 Bibliographie

Literatur

- [1] Nicol, Natascha und Albrecht, Ralf *Wissenschaftliche Arbeiten schreiben mit Word* Addison-Wesley, München, 2000
- [2] Hoffmann, Erwin *Bezüge, Fussnoten, Referenzen, Zitate* <https://www.fehcom.net/fh-frankfurt/zitate/Zitate.pdf>