A Directory Structure for TEX Files

TUG Working Group on a TEX Directory Structure (TWG-TDS)

draft version 0.104 November 15, 1995

This document is available on any CTAN host (Appendix D has a complete reference). Please send questions or suggestions by email to `twg-tds@shsu.edu` or by postal mail to Karl Berry / 135 Center Hill Road / Plymouth, MA 02360 / USA. We welcome all comments.

Contents

1 Introduction

TeX is a powerful, flexible typesetting system used by thousands of people around the world. It is extremely portable and runs on virtually all operating systems. One unfortunate side effect of TeX's flexibility is that there is no single "right" way to install it. This has resulted in many sites having different installed arrangements.

The primary purpose of this document is to describe a standard TeX Directory Structure (TDS): a directory hierarchy for macros, fonts, and the other implementation-independent TeX files. As a matter of practicality, this document also suggests ways to incorporate the rest of the TeX files into a single structure. It has been designed to work on all modern systems. In particular, this Technical Working Group (TWG) believes it is usable under Unix, MS-DOS, Windows NT, OS/2, MacOS, and VMS. We hope that administrators and developers of both free and commercial implementations of TeX will adopt this standard.

This document is intended both for the TeX system administrator at a site and for people preparing TeX distributions—everything from a complete runnable system to a single macro or style file. It may also help TeX users find their way around systems organized this way. It is not, however, a tutorial: we necessarily assume knowledge of the many parts of a working TeX system. If you are unfamiliar with any of the programs or file formats we refer to, consult the references in Appendix D.

1.1 The role of the TDS

The role of the TDS is to stabilize the organization of TeX-related software packages that are installed and in use, possibly on multiple platforms simultaneously.

At first glance, it may seem that the Comprehensive TeX Archive Network (CTAN) archives fulfill (at least) part of this role, but this is not the case. The role of CTAN is to simplify archiving and distribution, not installation and use.

In fact, the roles of the TDS and CTAN are frequently in conflict, as you will see elsewhere in this document. For distribution, many different types of files must be combined into a single unit; for use, it is traditional to segregate files (even similar files) from a single package into separate, occasionally distant, directories.

1.2 Conventions

In this document, "/" is used to separate filename components; for example, `texmf/fonts`. This is the Unix convention but the ideas are in no way Unix-specific.

In this document, "TeX" generally means the TeX system, including METAFONT, DVI drivers, utilities, etc., not just the TeX program itself.

The word "package" in this document has its usual meaning: a set of related files distributed, installed, and maintained as a unit. This is *not* a LaTeX 2ε package, which is a style file supplementing a document class.

We use the following typographic conventions:

`literal` Literal text such as `filename` is typeset in typewriter type.

⟨*replaceable*⟩ Replaceable text such as ⟨*package*⟩, identifying a class of things, is typeset in italics inside angle brackets.

2 General

This section describes common properties throughout the TDS tree.

2.1 Subdirectory searching

Many TeX installations store large numbers of related files in single directories, for example, all `TFM` files and/or all TeX input files.

This monolithic arrangement hinders maintenance of a TeX system: it is difficult to determine what files are used by what packages, what files need to be updated when a new version is installed, or what files should be deleted if a package is removed. It is also a source of error if two or more packages happen to have input files with the same name.

Therefore, the TWG felt each package should be in a separate directory. But we recognized that explicitly listing all directories to be searched would be unbearable. There are dozens of packages a site may wish to install. Aside from anything else, listing that many directories would lead to paths many thousands of characters long, overflowing the available space on some systems.

Also, if all directories are explicitly listed, installing or removing a new package would mean changing a path as well as installing or removing the actual files. This would be a time-consuming and error-prone operation, even with implementations that provide some way to specify the directories to search at runtime. On systems without runtime configuration, it would require recompiling software, an intolerable burden.

As a result, the TWG concluded that a comprehensive TDS required that implementations of TeX must support some form of implicit subdirectory searching. More precisely, implementations must make it possible to specify that TeX, METAFONT, and their companion utilities search in both a specified directory and recursively through all subdirectories of that directory when looking for an input file. (Other forms of subdirectory searching, for example recursive-to-one-level searches, may also be provided.) We encourage implementors to provide subdirectory searching (at the option of the installer and user) for all paths.

The TDS does not specify a syntax for specifying recursive searching, but we encourage implementors to provide interoperability (see Section B.2).

2.2 Rooting the tree

In this document, we shall designate the root TDS directory by '`texmf`' (for "TeX and METAFONT"). We recommend using that name where possible, but the actual name of the directory is up to the installer. On PC networks, for example, this could map to a logical drive specification such as `T:`.

Similarly, the location of this directory on the system is site-dependent. It may be at the root of the file system; on Unix systems, `/usr/local/share`, `/usr/local`, `/usr/local/lib`, and `/opt` are common choices.

The name `texmf` was chosen for several reasons: it reflects the fact that the directory contains files pertaining to an entire TeX system (including METAFONT, METAPOST, BibTeX, etc.), not just TeX itself and it is descriptive of a generic installation rather than a particular implementation.

A site may choose to have more than one TDS hierarchy installed (for example, when installing an upgrade). This is perfectly legitimate.

2.3 Local additions

The TDS cannot specify precisely when a package is or is not a "local addition". Each site must determine this according to their own conventions. One site might wish to consider "nonlocal" only those files that came with the particular TeX distribution they installed; another site might consider "local" only those files that were actually developed at the local site and not distributed elsewhere.

We recognize two common methods for local additions to a distributed `texmf` tree. Both have their place; in fact, some sites may employ both simultaneously:

1. A completely separate tree which is a TDS structure itself; for example, `/usr/local/umbtex` at the University of Massachusetts at Boston. This is another example of the multiple `texmf` hierarchies mentioned above.
2. A directory named '`local`' at any appropriate level, for example, in the ⟨*format*⟩, ⟨*package*⟩, and ⟨*supplier*⟩ directories discussed in the following sections. The TDS reserves the directory name `local` for this purpose.

   We recommend using `local` for site-adapted configuration files, such as `language.dat` for the Babel package or `graphics.cfg` for the graphics package. Unmodified configuration files from a package should remain in the package directory. The intent is to separate locally modified or created files from distribution files, to ease installing new releases.

One common case of local additions is dynamically generated files, e.g., PK fonts by the `MakeTeXPK` script originated by Dvips. A site may store the generated files directly in any of:

- their standard location in the main TDS tree (if it can be made globally writable);
- an alternative location in the main TDS tree (for example, under `texmf/fonts/tmp`);
- a second complete TDS tree (as outlined above);
- any other convenient directory (perhaps under `/var`, for example `/var/spool/fonts`).

No one solution will be appropriate for all sites.

2.4 Duplicate filenames

Different files by the same name may exist in a TDS tree. The TDS leaves unspecified which of two files by the same name in a search path will be found, so generally the only way to reliably find a given file is for it to have a unique name. However, the TDS requires implementations to support the following exceptions:

- Names of TeX input files must be unique within each given first-level subdirectory of `texmf/tex` and `texmf/tex/generic`, but not within all of `texmf/tex`; i.e., different TeX formats may have files by the same name. (Section 3.1 discusses this further.)

  Therefore, no single format-independent path specification, such as a recursive search beginning at `texmf/tex` specifying no other directories, suffices. So an implementation must provide format-dependent path specifications, for example via wrapper scripts or configuration files.

- Many font files will have the same name (e.g., `cmr10.pk`), as discussed in Section 3.2.2. An implementation must therefore distinguish these files by mode and resolution.

All implementations we know of already have these capabilities.

One place where duplicate names are likely to occur is not an exception:

- Names of METAFONT input files (as opposed to bitmap fonts) must be unique within all of `texmf/fonts`. In practice, this is a problem with those variants of Computer Modern which contain slightly modified files named `punct.mf`, `romanl.mf`, and so on. We believe the only feasible solution here is simply to rename the derivative files to be unique.

3 Top-level directories

The directories under the `texmf` root identify the major components of a TeX system. A site may omit any unneeded directories.

Although the TDS by its nature can specify precise locations only for implementation-independent files, we recognize that installers may well wish to place other files under `texmf` to simplify administration of the TeX tree, especially if it is maintained by someone other than the system administrator. Therefore, additional top-level directories may be present.

The top-level directories specified by the TDS are:

`tex` for TeX files (Section 3.1).

`fonts` for font-related files (Section 3.2).

`metafont` for (non-font) METAFONT files (Section 3.3).

`metapost` for METAPOST files (Section 3.4).

`bibtex` for BibTeX files (Section 3.5).

`doc` for user documentation (Section 3.6).

`source` for sources. This includes both traditional program sources (for example, Web2c sources go in `texmf/source/web2c`) and LaTeX `dtx` sources (which go in `texmf/source/latex`).

`source` is intended for files which are not needed at runtime by any TeX program; it should not be included in any search path. For example, `plain.tex` does not belong under `texmf/source`, even though it is a “source file” in the sense of not being derived from another file, but rather in `texmf/tex/plain/base`, as explained in Section 3.1.

⟨*implementation*⟩ for implementations (examples: `emtex`, `web2c`), to be used for whatever purpose deemed suitable by the implementor or TeX administrator. Files that cannot be shared between implementations, such as pool files (`tex.pool`) and memory dump files (`plain.fmt`) go here, in addition to implementation-wide configuration files. See Section B.3 for examples of real ⟨*implementation*⟩ trees.

⟨*program*⟩ for individual configuration files and program-specific input files for TeX-related programs (examples: `mft`, `dvips`). In fact, the `tex`, `metafont`, `metapost`, and `bibtex` directories above may be seen as instances of this case.

3.1 Macros

TeX macro files shall be stored in separate directories, segregated by TeX format and package name. (We use 'format' in its traditional TeX sense to mean a usefully `\dump`-able package.)

`texmf/tex/`⟨*format*⟩`/`⟨*package*⟩`/`

⟨*format*⟩ is a format name (examples: `plain`, `amstex`, `texinfo`, `latex`). By providing a ⟨*format*⟩ directory, path searching can be limited to only those directories that contain relevant files.

The TDS allows distributions that can be used as either formats or packages (e.g., Texinfo, Eplain) to be stored at either level, at the option of the format author or TeX administrator. We recommend that packages used as formats at a particular site be stored at the ⟨*format*⟩ level: by adjusting the TeX inputs search path, it will be straightforward to use them as macro packages under another format, whereas placing them in another tree completely obscures their use as a format.

The TDS reserves the following ⟨*format*⟩ names:

- `generic`, for input files that are useful across a wide range of formats (examples: `null.tex`, `path.sty`). Generally, this means any format that uses the category codes of Plain TeX and does not rely on any particular format. This is in contrast to those files which are useful only with Plain TeX (which go under `texmf/tex/plain`), e.g., `testfont.tex` and `plain.tex` itself.
- `local`, for local additions. See Section 2.3.

Thus, for every format, it is correct to search at least the ⟨*format*⟩ directory and then the `generic` directory (in that order). Other directories may need to be searched as well, depending on the format. When using AMS-TeX, for example, the `amstex`, `plain`, and `generic` directories should be searched, because AMS-TeX is compatible with Plain.

⟨*package*⟩ is a TeX package name (examples: `texdraw`, `babel`). The TDS reserves the following ⟨*package*⟩ names:

- `base`, for the base distribution of each format, including files used by `INITEX` when dumping format files. For example, in the standard LaTeX distribution, the `ltx` files created during the build process shall be stored in the `base` directory.
- `hyphen`, for hyphenation patterns, e.g., `hyphen.tex`. These are typically used only by `INITEX`. In most situations, this directory need exist only under the `generic` format.
- `images`, for image input files, such as Encapsulated PostScript figures. Although it is somewhat non-intuitive for these to be under a directory named "`tex`", TeX needs to read these files to glean bounding box or other information. A mechanism for sharing image inputs between TeX and other typesetting programs (Interleaf, FrameMaker, etc.) is beyond the scope of the TDS. In most situations, this directory need exist only under the `generic` format.
- `local`, for local additions and configuration files. See Section 2.3.
- `misc`, for packages that consist of a single file. An administrator or package maintainer may nevertheless choose to create directories even for single-file packages at their discretion.

In the case where a format consists of only a single file and has no auxiliary packages, that file can simply be placed in the ⟨*format*⟩ directory, instead of ⟨*format*⟩`/base`. For example, Texinfo goes in `texmf/tex/texinfo/texinfo.tex`, not `texmf/tex/texinfo/base/texinfo.tex`.

3.2 Fonts

Font files shall be stored in separate directories, segregated by file type, font supplier, and typeface name. `PK` and `GF` files need additional structure, as detailed in the next section.

`texmf/fonts/`⟨*type*⟩`/`⟨*supplier*⟩`/`⟨*typeface*⟩`/`

⟨*type*⟩ is the type of font file. The TDS reserves the following ⟨*type*⟩ names:

- `afm`, for Adobe font metrics.
- `gf`, for generic font bitmap files.
- `pk`, for packed bitmap files.
- `source`, for font sources (METAFONT files, property lists, etc.).
- `tfm`, for TeX font metric files.
- `type1`, for Type 1 fonts (in any format).
- `vf`, for virtual fonts.

As usual, any of these directories may be omitted at a site not needing them (`gf` is a particularly likely candidate for omission).

⟨*supplier*⟩ is a name identifying font source (examples: `adobe`, `ams`, `public`). The TDS reserves the following ⟨*supplier*⟩ names:

- `ams`, for the fonts distributed by the American Mathematical Society in the AMS-fonts collection.
- `local`, for local additions. See Section 2.3.
- `public`, for freely redistributable fonts where the supplier neither (1) requested their own directory (as with, e.g., `ams`), nor (2) made a great number of fonts (e.g., `adobe`). It does not contain all extant freely distributable fonts, nor are all files therein necessarily strictly public domain.
- `tmp`, for dynamically-generated fonts, as is traditional on some systems. It may be omitted on systems which do not need it (like all other directories in the TDS).

⟨*typeface*⟩ is the name of a typeface family (examples: `cm`, `euler`, `times`). The TDS reserves the following ⟨*typeface*⟩ names:

- `cm` (within `public`), for the 75 fonts defined in *Computers and Typesetting, Volume E*.
- `latex` (within `public`), for those fonts distributed with LaTeX in the base distribution.
- `local`, for local additions. See Section 2.3.

Some concrete examples:

`texmf/fonts/tfm/public/cm/cmr10.tfm`
`texmf/fonts/type1/adobe/utopia/putr.pfa`

For complete supplier and typeface name lists, consult *Filenames for TeX fonts* (see Appendix D).

3.2.1 Font bitmaps

Font bitmap files require two characteristics in addition to the above to be uniquely identified: (1) the type of device (i.e., mode) for which the font was created; (2) the resolution of the bitmap.

Following common practice, the TDS segregates fonts with different device types (modes) into separate directories. Consult `modes.mf` (see Appendix D) for recommended mode names.

Some printers operate at more than one resolution (e.g., at 300 dpi and 600 dpi), but each such resolution will necessarily have a different mode name. Nothing further is needed, since implicit in the TeX system is the assumption of a single target resolution.

Two naming strategies are commonly used to identify the resolution of bitmap font files. On systems that allow long filenames (and in the original METAFONT program itself), the resolution is included in the filename (e.g., `cmr10.300pk`). On systems which do not support long filenames, fonts are generally segregated into directories by resolution (e.g., `dpi300/cmr10.pk`).

Because the TDS cannot require long filenames (see Section A.1), we must use the latter scheme for naming fonts, via two more subdirectory levels under `pk` and `gf`:

`texmf/fonts/pk/`⟨*mode*⟩`/`⟨*supplier*⟩`/`⟨*typeface*⟩`/dpi`⟨*nnn*⟩`/`
`texmf/fonts/gf/`⟨*mode*⟩`/`⟨*supplier*⟩`/`⟨*typeface*⟩`/dpi`⟨*nnn*⟩`/`

⟨*mode*⟩ is a name which identifies the device type (examples: `cx`, `gsftopk`, `ljfour`). Usually, this is the name of the METAFONT mode used to build the PK file.

For fonts rendered as bitmaps by a program that does not distinguish between different output devices, the ⟨*mode*⟩ name shall be that of the program (e.g., `ps2pk`, `gsftopk`).

`dpi`⟨*nnn*⟩ specifies the resolution of the font (examples: `dpi300`, `dpi329`). '`dpi`' stands for dots per inch, i.e., pixels per inch. We recognize that pixels per millimeter is used in many parts of the world, but dpi is too traditional in the TeX world to consider changing now.

The integer ⟨*nnn*⟩ is to be calculated using METAFONT arithmetic and then rounded; i.e., it is the integer METAFONT uses in its output `gf` filename. We recognize small differences in the resolution are a common cause of frustration among users, however, and recommend implementors follow the level 0 DVI driver standard (see Appendix D) in bitmap font searches by allowing a fuzz of $\pm 0.2\%$ (with a minimum of 1) in the ⟨*dpi*⟩.

Implementations may provide extensions to the basic naming scheme, such as long filenames and font library files, provided that the basic scheme is also supported.

3.2.2 Valid font bitmaps

The TWG recognizes that the use of short filenames has many disadvantages. The most vexing is that it results in the creation of dozens of different files with exactly the same name. At a typical site, `cmr10.pk` will be the filename for Computer Modern Roman 10 pt at 5–10 magnifications for 2–3 modes. (Section 2.4 discusses duplicate filenames in general.)

To minimize this problem, the TDS strongly recommends PK files contain enough information to identify precisely how they were created: at least the mode, base resolution, and magnification used to create the font.

This information is easy to supply: a simple addition to the local modes used for building the fonts with METAFONT will automatically provide the required information. If you have been using a local modes file derived from (or that is simply) `modes.mf` (see Appendix D), the required information is already in your PK files. If not, a simple addition based on the code found in `modes.mf` can be made to your local modes file and the PK files rebuilt.

3.3 Non-font METAFONT files

Most METAFONT input files are font programs or parts of font programs and are thus covered by the previous section. However, a few non-font input files do exist. Such files shall be stored in:

`texmf/metafont/`⟨*package*⟩`/`

⟨*package*⟩ is the name of a METAFONT package (for example, `mfpic`).

The TDS reserves the following ⟨*package*⟩ names:

- `base`, for the standard METAFONT macro files as described in *The METAFONTbook*, such as `plain.mf` and `expr.mf`.
- `local`, for local additions. See Section 2.3.
- `misc`, for METAFONT packages consisting of only a single file (for example, `modes.mf`).

3.4 METAPOST

METAPOST is a picture-drawing language developed by John Hobby, derived from Knuth's METAFONT. Its primary purpose is to output Encapsulated PostScript instead of bitmaps.

METAPOST input files and the support files for METAPOST-related utilities shall be stored in:

`texmf/metapost/`⟨*package*⟩`/`

⟨*package*⟩ is the name of a METAPOST package. At the present writing none exist, but the TWG thought it wise to leave room for contributed packages that might be written in the future.

The TDS reserves the following ⟨*package*⟩ names:

- `base`, for the standard METAPOST macro files, such as `plain.mp`, `mfplain.mp`, `boxes.mp`, and `graph.mp`. This includes files used by `INIMP` when dumping mem files containing preloaded macro definitions.
- `local`, for local additions. See Section 2.3.
- `misc`, for METAPOST packages consisting of only a single file.
- `support`, for additional input files required by METAPOST utility programs, including a font map, a character adjustment table, and a subdirectory containing low-level METAPOST programs for rendering some special characters.

3.5 BibTeX

BibTeX-related files shall be stored in:

`texmf/bibtex/bib/`⟨*package*⟩`/`
`texmf/bibtex/bst/`⟨*package*⟩`/`

The `bib` directory is for BibTeX database (`.bib`) files.
The `bst` directory is for BibTeX style (`.bst`) files.

⟨*package*⟩ is the name of a BibTeX package. The TDS reserves the following ⟨*package*⟩ names (the same names are reserved under both `bib` and `bst`):

- `base`, for the standard BibTeX databases and styles, such as `xampl.bib`, `plain.bst`.
- `local`, for local additions. See Section 2.3.
- `misc`, for BibTeX packages consisting of only a single file.

3.6 Documentation

Most packages come with some form of documentation: user manuals, example files, programming guides, etc. In addition, many independent files not part of any macro or other package describe various aspects of the TeX system.

The TDS specifies that these additional documentation files shall be stored in a structure that parallels to some extent the `fonts` and `tex` directories, as follows:

`texmf/doc/`⟨*category*⟩`/...`

⟨*category*⟩ identifies the general topic of documentation that resides below it; for example, a TeX format name (`latex`), program name (`tex`, `bibtex`), or other system components (`web`, `fonts`).

The TDS reserves the following categories:

- Within each ⟨*category*⟩ tree for a TeX format, the directory `base` is reserved for base documentation distributed by the format's maintainers.
- `general`, for standalone documents not specific to any particular program (for example, Joachim Schrod's *Components of TeX*).
- `help`, for meta-information, such as FAQ's, David Jones' macro index, etc.
- `html`, for HTML documents.
- `info`, for processed Texinfo documents. (Info files, like anything else, may also be stored outside the TDS, at the installer's option.)
- `local`, for local additions. See Section 2.3.

The `doc` directory is intended for implementation-independent and operating system-independent documentation files. Implementation-dependent files shall be stored in their respective directory, as provided for by the implementation and/or TeX administrator (for example, VMS help files under `texmf/vms/help`).

The documentation directories may contain TeX sources, `DVI` files, PostScript files, text files, example input files, or whatever form of documentation will best explain the package.

See Section 4.1 for a summary.

4 Summary

A skeleton of a TDS `texmf` directory tree:

`bibtex/`	BibTeX input files
`bib/`	BibTeX databases
`base/`	base distribution (e.g., `xampl.bib`)
`misc/`	single-file databases
`bst/`	BibTeX style files
`base/`	base distribution (e.g., `plain.bst`, `acm.bst`)
`misc/`	single-file styles
`doc/`	see Section 3.6 and the summary following
`fonts/`	font-related files
⟨*type*⟩`/`	file type (e.g., `pk`)
⟨*mode*⟩`/`	type of output device (for `pk` and `gf` only)
⟨*supplier*⟩`/`	name of a font supplier (e.g., `public`)
⟨*typeface*⟩`/`	name of a typeface (e.g., `cm`)
`dpi`⟨*nnn*⟩`/`	font resolution (for `pk` and `gf` only)
⟨*implementation*⟩`/`	TeX implementations, by name (e.g., `emtex`)
`metafont/`	METAFONT (non-font) input files
`base/`	base distribution (e.g., `plain.mf`)
`misc/`	single-file packages (e.g., `modes.mf`)
⟨*package*⟩`/`	name of a package (e.g., `mfpic`)
`metapost/`	METAPOST input and support files
`base/`	base distribution (e.g., `plain.mp`)
`misc/`	single-file packages
⟨*package*⟩`/`	name of a package
`support/`	support files for METAPOST-related utilities
`mft/`	MFT inputs (e.g., `plain.mft`)
⟨*program*⟩`/`	TeX-related programs, by name (e.g., `dvips`)
`source/`	program source code by name (e.g., `web2c`, `latex`)
`tex/`	TeX input files
⟨*format*⟩`/`	name of a format (e.g., `plain`)
`base/`	base distribution for format (e.g., `plain.tex`)
`misc/`	single-file packages (e.g., `webmac.tex`)
`local/`	local additions to or configuration files for ⟨*format*⟩
⟨*package*⟩`/`	name of a package (e.g., `graphics`, `mfnfss`)
`generic/`	format-independent packages
`hyphen/`	hyphenation patterns (e.g., `hyphen.tex`)
`images/`	image input files (e.g., Encapsulated PostScript)
`misc/`	single-file format-independent packages (e.g., `null.tex`).
⟨*package*⟩`/`	name of a package (e.g., `babel`)

4.1 Documentation tree summary

A skeleton of a TDS directory tree under `texmf/doc`:

`ams/`	
`  amsfonts/`	`amsfonts.faq`, `amfndoc`
`  amslatex/`	`amslatex.faq`, `amsldoc`
`  amstex/`	`amsguide`, `joyerr`
`bibtex/`	BibTeX
`  base/`	`btxdoc.tex`
`fonts/`	
`  fontname/`	*Filenames for TeX fonts*
`  oldgerm/`	`corkpapr`
⟨*format*⟩`/`	name of a TeX format (e.g., `generic`, `latex`)
`  base/`	for the base distribution
`  misc/`	for contributed single-file package documentation
⟨*package*⟩`/`	for *package*
`general/`	across programs, generalities
`  errata/`	`errata`, `errata[1-8]`
`  texcomp/`	*Components of TeX*
`generic/`	for non-format-specific TeX packages
`  babel/`	
`  german/`	`germdoc`
`help/`	meta-information
`  ctan/`	info about CTAN mirror sites
`  faq/`	FAQs of `comp.text.tex`, etc.
`html/`	HTML files
`info/`	GNU Info files, made from Texinfo sources
`latex/`	example of ⟨*format*⟩
`  base/`	`ltnews*`, `*guide`, etc.
`  graphics/`	`grfguide`
⟨*program*⟩`/`	TeX-related programs, by name (examples follow)
`metafont/`	`mfbook.tex`, `metafont-for-beginners`, etc.
`metapost/`	`mpman`, `manfig`, etc.
`tex/`	`texbook.tex`, *A Gentle Introduction to TeX*, etc.
`web/`	`webman`, `cwebman`

A Unspecified pieces

The TDS cannot address the following aspects of a functioning TeX system:

1. The location of executable programs: this is too site-dependent even to recommend a location, let alone require one. A site may place executables outside the `texmf` tree altogether (e.g., `/usr/local/bin`), in a platform-dependent directory within `texmf`, or elsewhere.
2. Upgrading packages when new releases are made: we could find no way of introducing version specifiers into `texmf` that would do more good than harm, or that would be practical for even a plurality of installations.

3. The location of implementation-specific files (e.g., TeX `.fmt` files): by their nature, these must be left to the implementor or TeX maintainer. See Section B.3.
4. Precisely when a package or file should be considered "local", and where such local files are installed. See Section 2.3 for more discussion.

A.1 Portable filenames

The TDS cannot require any particular restriction on filenames in the tree, since the names of many existing TeX files conform to no particular scheme. For the benefit of people who wish to make a portable TeX distribution or installation, however, we outline here the necessary restrictions.

It turns out that ISO-9660 meets the stringent limitations of all operating systems in use today. It is the only universally acceptable file system format for CD-ROMs. It specifies the following:

- File and directory names, not including any directory path or extension part, may not exceed eight characters.
- Filenames may have a single extension. Extensions may not exceed three characters. Directory names may not have an extension.
- Names and extensions may consist of *only* the characters `A`–`Z`, `0`–`9`, and underscore. Lowercase letters are excluded. (The only common place where mixed-case names occur in the TeX system is in LaTeX font descriptor files, and LaTeX does not rely on case alone to distinguish among these files.)
- A period separates the filename from the extension and is always present, even if the name or extension is missing (e.g., `FILENAME.` or `.EXT`).
- A version number, ranging from 1–32767, is appended to the file extension, separated by a semicolon (e.g., `FILENAME.EXT;1`).
- Only eight directory levels are allowed, including the top-level (mounted) directory (see Section 2.2). Thus, the deepest valid ISO-9660 path is:

  ```
  texmf/L2/L3/L4/L5/L6/L7/L8/FOO.BAR;1
  1     2  3  4  5  6  7  8
  ```

  The deepest TDS path needs only seven levels:

  ```
  texmf/fonts/pk/cx/public/cm/dpi300/cmr10.pk
  1     2     3  4      5  6  7
  ```

Some systems display a modified format of ISO-9660 names, mapping alphabetic characters to lowercase, removing version numbers and trailing periods, etc.

B Implementation issues

We believe that the TDS can bring a great deal of order to the current anarchic state of many TeX installations. In addition, by providing a common frame of reference, it will ease the burden of documenting administrative tasks. Finally, it is a necessary part of any reasonable system of true "drop-in" distribution packages for TeX.

B.1 Adoption of the TDS

We recognize that adoption of TDS will not be immediate or universal. Most TeX administrators will not be inclined to make the switch until:

- Clear and demonstrable benefits can be shown for the TDS.
- TDS-compliant versions of all key programs are available in ported, well-tested forms.
- A "settling" period has taken place, to flush out problems. The public release of this document is the first step in this process.

Consequently, most of the first trials of the TDS will be made by members of the TDS committee and/or developers of TeX-related software. Indeed, some of this has taken place during the course of our deliberations (see Appendix D for a sample tree available electronically). They will certainly result in the production of a substantial number of TDS-compliant packages.

Once installable forms of key TDS-compliant packages are more widespread, some TeX administrators will set up TDS-compliant trees, possibly in parallel to existing production directories. This testing will likely flush out problems that were not obvious in the confined settings of the developers' sites; for example, it should help to resolve OS and package dependencies, package interdependencies, and other details not addressed by this TDS version.

After most of the dust has settled, hopefully even conservative TeX administrators will begin to adopt the TDS. Eventually, most TeX sites will have adopted the common structure, and most packages will be readily available in TDS-compliant form.

We believe that this process will occur relatively quickly. The TDS committee spans a wide range of interests in the TeX community. Consequently, we believe that most of the key issues involved in defining a workable TDS definition have been covered, often in detail. Most TeX developers have been consulted about implementation issues, and have been trying out suggested TDS formats. Thus, we hope for few surprises as implementations mature.

Finally, there are several (current or prospective) publishers of TeX-based CD-ROMs. These publishers are highly motivated to work out details of TDS implementation, and their products will provide inexpensive and convenient ways for experimentally-minded TeX administrators to experiment with the TDS.

Efforts are under way to set up a "TDS Registry" that will coordinate assignment of TDS-compliant directory names and provide a definitive database of TDS-compliant software distributions. (Perhaps this could also serve many sites as the definition of when a package is local.) For now, distribution through CTAN serves as an imprecise registry.

B.2 More on subdirectory searching

Recursive subdirectory searching is the ability to specify a search not only of a specified directory $\langle d \rangle$, but recursively of all directories below $\langle d \rangle$.

Since the TDS specifies precise locations for most files, with no extra levels of subdirectories allowed, true recursive searching is not actually required for a TDS-compliant implementation. We do, however, strongly recommend recursive searching as the most user-friendly and natural approach to the problem, rather than convoluted methods to specify paths without recursion.

This feature is already supported by many implementations of TEX and companion utilities, for example DECUS TEX for VMS, Dvips(k), emTEX (and its drivers), PubliC TEX, Web2c, Xdvi(k), and Y&YTEX.

Even if your TEX implementation does not directly support subdirectory searching, you may find it useful to adopt the structure if you do not use many fonts or packages. For instance, if you only use Computer Modern and AMS fonts, it would be feasible to store them in the TDS layout and list the directories individually in configuration files or environment variables.

The TWG recognizes that subdirectory searching places an extra burden on the system and may be the source of performance bottlenecks, particularly on slower machines. Nevertheless, we feel that subdirectory searching is imperative to a well-organized TDS, for the reasons stated in Section 2.1. Implementors are encouraged to provide enhancements to the basic principle of subdirectory searching to avoid performance problems, e.g., the use of a filename cache (this can be as simple as a recursive directory listing) that is consulted before disk searching begins. If a match is found in the database, subdirectory searching is not required, and performance is thus independent of the number of subdirectories present on the system.

Different implementations specify subdirectory searching differently. In the interest of typographic clarity, the examples here do not use the ⟨*replaceable*⟩ font.

Dvips: via a separate `TEXFONTS_SUBDIR` environment variable.

emTEX: `t:\subdir!!`; `t:\subdir!` for a single level of searching.

Kpathsea: `texmf/subdir//`

VMS: `texmf:[subdir...]`

Xdvi (patchlevel 20): `texmf/subdir/**`; `texmf/subdir/*` for a single level of searching.

Y&Y TEX: `t:/subdir//` or `t:\subdir\\`.

B.3 Example implementation-specific trees

The TDS cannot specify a precise location for implementation-specific files, but for informative purposes, we provide here the default locations for some implementations. Please contact us with additions or corrections. These paths are not definitive, may not match anything at your site, and may change without warning.

We recommend all implementations have default search paths that start with the current directory (e.g., '`.`'). Allowing users to include the parent directory (e.g., '`..`') is also helpful.

B.3.1 Public DECUS TEX

If another VMS implementation besides Public DECUS TEX appears, the top level implementation directory name will be modified to something more specific (e.g., `vms_decus`).

```
texmf/
  vms/          VMS implementation specific files
    exe/        end-user commands
      common/   command procedures, command definition files, etc.
      axp/      binary executables for Alpha AXP
```

`vax/`	binary executables for VAX
`formats/`	pool files, formats, bases
`help/`	VMS help library, and miscellaneous help sources
`mgr/`	command procedures, programs, docs, etc., for system management

B.3.2 Web2c 7.0

(Email `tex-k@cs.umb.edu` to contact the maintainer of this implementation.)

All implementation-dependent TEX system files (`.pool`, `.fmt`, `.base`, `.mem`) are stored by default directly in `texmf/web2c`. The configuration file `texmf.cnf` and various subsidiary `MakeTeX...` scripts used as subroutines are also stored there.

Non-TEX specific files are stored following standard GNU practice. Given a root directory ⟨*prefix*⟩ (`/usr/local` by default), we have default locations as follows:

⟨*prefix*⟩`/`	installation root (`/usr/local` by default)
`bin/`	executables
`man/`	man pages
`info/`	info files
`lib/`	libraries (`libkpathsea.*`)
`share/`	
`texmf/`	TDS root
`web2c/`	implementation-dependent files (`.pool`, `.fmt`, `texmf.cnf`, etc.)

See `prep.ai.mit.edu:/pub/gnu/standards.*` for the rationale behind and descriptions of this arrangement. A site may of course override these defaults; for example, to put everything under a single directory such as `/usr/local/texmf`.

C Is there a better way?

Defining the TDS required many compromises. Both the overall structure and the details of the individual directories were arrived at by finding common ground among many opinions. The driving forces were feasibility (in terms of what could technically be done and what could reasonably be expected from developers) and regularity (files grouped together in an arrangement that "made sense").

Some interesting ideas could not be applied due to some or all current implementations lacking the necessary support:

- Path searching control at the TEX level. If documents could restrict subdirectory searching to a subdirectory via some portable syntax in file names, restrictions on uniqueness of filenames could be relaxed considerably (with the cooperation of the formats), and the TEX search path would not need to depend on the format.
- Multiple logical `texmf` trees. For example, a site might have one (read-only) location for stable files, and a different (writable) location for dynamically-created fonts or other files. It would be reasonable for two such trees to be logically merged when searching, etc.

C.1 Macro structure

The TWG settled on the ⟨*format*⟩/⟨*package*⟩ arrangement after long discussion about how best to arrange the files.

The primary alternative to this arrangement was a scheme which reversed the order of these directories: ⟨*package*⟩/⟨*format*⟩. This reversed arrangement has a strong appeal: it keeps all of the files related to a particular package in a single place. The arrangement actually adopted tends to spread files out into two or three places (macros, documentation, and fonts, for example, are spread into different sections of the tree right at the top level).

Nevertheless, the ⟨*format*⟩/⟨*package*⟩ structure won for a couple of reasons:

- It is closer to current practice; in fact, several members of the TWG have already implemented the TDS hierarchy. The alternative is not in use at any known site, and the TWG felt it wrong to mandate something with which there is no practical experience.
- The alternative arrangement increases the number of top-level directories, so the files that must be found using subdirectory searching are spread out in a wide, shallow tree. This could have a profound impact on the efficiency of subdirectory searching.

C.2 Font structure

The TWG struggled more with the font directory structure than anything else. This is not surprising; the proliferation of PostScript fonts being used with TeX is what made the standard single-directory arrangements untenable, and therefore what initiated the TDS effort.

C.2.1 Font file type location

We considered the supplier-first arrangement currently in use at many sites:

`texmf/fonts/`⟨*supplier*⟩`/`⟨*typeface*⟩`/`⟨*type*⟩`/`

This improves the maintainability of the font tree, since all files comprising a given typeface are in one place, but unless all the programs that search this tree employ some form of caching, there are serious performance concerns. For example, in order to find a `TFM` file, the simplest implementation would require TeX to search through all the directories that contain `PK` files in all modes and at all resolutions.

In the end, a poll of developers revealed considerable resistance to implementing sufficient caching mechanisms, so this arrangement was abandoned. The TDS arrangement allows the search tree to be restricted to the correct type of file, at least. Concerns about efficiency remain, but there seems to be no more we can do without abandoning subdirectory searching entirely.

We also considered segregating all font-related files strictly by file type, so that METAFONT sources would be in a directory `texmf/fonts/mf`, property list files in `texmf/fonts/pl`, the various forms of Type 1 fonts separated, and so on. Although more blindly consistent, we felt that the drawback of more complicated path constructions outweighed the benefits. The TDS merges file types (`mf` and `pl` under `source`, `pfa` and `pfb` and `gsf` under `type1`) when a benefit could be found.

C.2.2 Mode and resolution location

We considered having the `mode` at the bottom of the font tree:

`texmf/fonts/pk/`⟨*supplier*⟩`/`⟨*typeface*⟩`/`⟨*mode*⟩`/`⟨*dpi*⟩`/`

In this case, however, it is difficult to limit subdirectory searching to the mode required for a particular device.

We then considered moving the `dpi`⟨*nnn*⟩ up to below the mode, as seemed natural:

`texmf/fonts/pk/`⟨*mode*⟩`/`⟨*dpi*⟩`/`⟨*supplier*⟩`/`⟨*typeface*⟩`/`

But then it is not feasible to omit the `dpi`⟨*nnn*⟩ level altogether on systems which can and do choose to use long filenames.

C.2.3 Modeless bitmaps

The TDS specifies using utility names as mode names for those utilities which generate bitmaps, e.g., `texmf/fonts/pk/gsftopk/`. An alternative was to introduce a single directory `modeless/` below which all such directories could be gathered. This has the considerable advantage of not requiring each such directory name to be listed in a search path.

But it has disadvantages, too: it would use the last available directory level, preventing any future expansions; and it would put PK files at different depths in the tree, possibly hindering existing search strategies and certainly a likely source of confusion. We decided to stay with existing practice and keep the utility name at the same level as the mode names.

We are making an implicit assumption that METAFONT is the only program producing mode-dependent bitmaps. If this becomes false we could add an abbreviation for the program to mode names, as in `mfcx` vs. `xyzcx` for a hypothetical program Xyz, or we could at that time add an additional program name level uniformly to the tree. For our present purposes, it seems more important to concisely represent the current situation than to take account of hypothetical possibilities that may never be realized.

C.3 Documentation structure

Additional documentation files have always been problematic and often ignored or placed in obscure places.

We considered placing them in the same directory as the source files for the packages, but we felt it was important that users be able to find documentation separately from sources, since most users have no interest in sources.

We hope that a separate, but parallel, structure for documentation would (1) keep the documentation together and (2) make it as straightforward as possible for users to find the particular documentation they were after.

D Related references

This appendix gives pointers to related files and other documents.

In this document, ⟨*CTAN:*⟩ means the root of an anonymous ftp CTAN tree. This is both a host name and a directory name. For example:

    ftp.dante.de:/tex-archive
    ftp.shsu.edu:/tex-archive
    ftp.tex.ac.uk:/tex-archive

Finger `ctan@ftp.shsu.edu` for a complete list of CTAN sites; there are mirrors worldwide.

Here are the references:

- The TDS mailing list archives can be retrieved via ftp from `shsu.edu:[twg-tds]` and `vms.rhbnc.ac.uk:/archives/twg.tds.*`.
- A sample TDS tree: ⟨*CTAN:*⟩`tds`.
- A collection of BibTeX databases and styles: `ftp.math.utah.edu:/pub/tex/bib`.
- *Components of TeX* by Joachim Schrod: ⟨*CTAN:*⟩`documentation/components-of-TeX`.
- The level 0 DVI driver standard: ⟨*CTAN:*⟩`dviware/driv-standard/level-0`.
- *Filenames for TeX fonts*: ⟨*CTAN:*⟩`documentation/fontname`. This distribution includes recommended supplier and typeface names.
- ISO-9660 CD-ROM file system standard: `http://www.iso.ch/cate/cat.html`.
- A complete set of METAFONT modes: ⟨*CTAN:*⟩`fonts/modes/modes.mf`. This file includes recommended mode names.

E Contributors

The TWG had no formal meetings; electronic mail was the primary communication medium.

Sebastian Rahtz is the TeX Users Group Technical Council liaison. Norman Walsh is the committee chair.

Original contributors:

barbara beeton	Karl Berry	Vicki Brown	David Carlisle
Thomas Esser	Alan Jeffrey	Jörg Knappen	Pierre MacKay
Rich Morin	Sebastian Rahtz	Joachim Schrod	Christian Spieler
Elizabeth Tachikawa	Philip Taylor	Ulrik Vieth	Paul Vojta
Norman Walsh			

Additional contributors:

David Aspinall	Nelson Beebe	Harriet Borton	Bart Childs
Damian Cugley	Alan Dunwell	Michael Ferguson	Erik Frambach
Bernard Gaulle	Jeffrey Gealow	George Greenwade	Thomas Herter
Berthold Horn	Charles Karney	David Kastrup	David Kellerman
Wonkoo Kim	Richard Kinch	Robin Kirkham	Alex Kok
Eberhard Mattes	Bob Morris	Lenny Muellner	Oren Patashnik
David Rhead	Mark Sinke	Andrew Trevorrow	Doug Waud
Chee-Wai Yeung			