MPlib: MetaPost as a reusable component

Taco Hoekwater, Hans Hagen
http://tug.org/metapost

Abstract

This short article introduces MPlib, a project that will be started in the autumn of 2007. The goal of this project is to turn MetaPost into a modern, re-entrant system library that can be used by many different applications programs at the same time.

1 The MetaPost workflow

Probably the most common use of MetaPost today is as a batch drawing program to create graphics that are then included inside a pdfTEX document. It is even quite normal for those graphics to be included inside the TEX source, with the MetaPost input file created on the fly by macro processing: for LATEX, this functionality is provided by the packages `emp`, `feynmf`, and `mfpic`; in ConTEXt, extensive in-line MetaPost support is built into the core engine.) In that case, MetaPost is often run on the fly by means of Web2C's `\write18` TEX extension.

In figure 1 you will see a typical flowchart of that process, and you will notice that it is a fairly complex affair.

The left column is the process that is immediately visible to the user: you run pdfTEX on a `.tex` file, and it generates a `.pdf` file.

The next column shows the execution of MetaPost, along with the required pre-processing (the TEX macro package has to create a temporary input file for MetaPost) and the post-processing (converting MetaPost's output from EPS to PDF format). In the figure, `mptopdf` is represented as a single program for the sake of simplicity. Various solutions exist for this, and the most common one is based on a set of TEX macros that ship with the ConTEXt distribution. These macros can be executed via a standalone program, or (most often) as a macro package that is included by the main TEX document.

The whole right-hand side of the flowchart is taken up by `makempx`, the program that handles TEX-based labels inside images. The MetaPost executable calls `makempx` automatically when it discovers that there are TEX-based labels in the document. `makempx` itself is just a dispatcher: it runs the separate program `mpto` to extract those labels from the MetaPost input file and place them in a TEX file, then it runs TEX on that file, and finally it runs `dvitomp` to convert the DVI file back into low-level drawing routines that MetaPost understands.

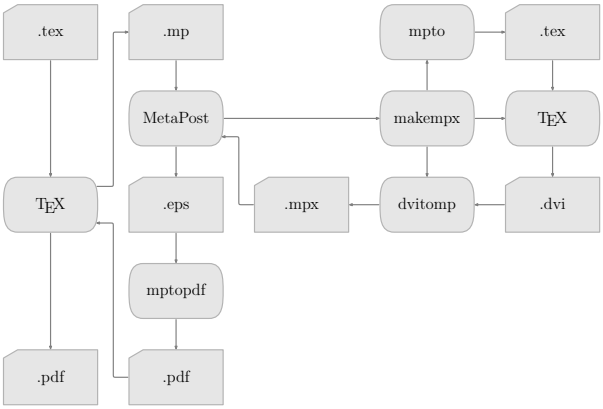

Figure 1: A typical workflow for MetaPost images inside a PDF document.

In this workflow, there are half a dozen programs called and the same number of intermediate files created.

2 Rationale

From looking at the workflow figure, it should be clear that all of this is not very efficient. In particular, the whole `makempx` block is wasteful of system resources, especially when MetaPost is executed on-the-fly.

It would be much nicer if MetaPost behaved like other system library components such as XML parsers and OpenGL engines: just link your application to the library, and there should be no more need for all those intermediate files and external programs.

Unfortunately, updating the label handling and creating system integration requires massive changes to the source code as well as the build system, and therefore it was very unlikely that this would ever get done without extra incentives. A significant amount of time and effort has to be invested to fix those particular problems.

It was clear to us that, to get these tasks done within a reasonable time frame, at least some of the

work would have to be done under an organized project umbrella, and that is why we started the MPlib project.

3 Goals

What we want is to convert MetaPost into a reusable component library that is fully re-entrant and whose functionality can be easily embedded into other programs. To reach this primary goal, MetaPost not only has to be converted to a form suitable for library use, but also a set of new components needs to be added: an indirection layer for input and output, a configurable system for strategies regarding error handling, and a re-engineered labeling system.

4 Implementation

Work will start in the autumn of this year, and it is our current estimate that the project will be complete by the summer of 2008. The actual programming will be carried out by Taco. Hans Hagen will lead the project, and Bogusław Jackowski will be in charge of quality control.

The current version of MetaPost is a mix of WEB (Pascal) and C code, that is compiled using a complex build system based on the Web2C Pascal converter. One of the implementation tasks for MPlib is to convert MetaPost into a more mainstream distribution package. For that, all of the source code will be converted into C, using either CWEB or NOWEB to retain the literate programming quality of MetaPost.

Some parts of the internals of MetaPost will be opened up and a documented application interface will be offered. Besides a MetaPost-compatible standalone executable based on MPlib, a Lua language binding to the library will be provided. This binding will allow the immediate use of MPlib within LuaTEX, as well as function as an example for other language bindings.

5 Acknowledgments

This project is supported by the worldwide TEX User Groups. In particular, we want thank the following user groups that have already promised financial and other support: DANTE e.V., TUGIndia, TUG, NTG, CSTUG, and GUST.