Multiple Changefiles: The Adventure Continues

E. W. (Wayne) Sewell
Software Engineering Specialist

The subject of multiple changefiles for the same WEB program has appeared several times in recent issues of TUGboat (Appelt and Korn, Vol 7, #1, and Guntermann and Rülling, Vol 7, #3). Even though the general TUG readership may be tiring of this subject, it is still a valid concern for anyone actively involved in WEB programming. This article describes yet another approach to the subject, embodied in a program called WEBMERGE.

A perfect example of the use of a program such as WEBMERGE is the Modula-2 WEB system described elsewhere in this issue (page 118). MWEB was implemented as a pair of changefiles applied to TANGLE and WEAVE, containing the modifications to allow WEB to work with the language Modula-2. However, virtually every implementation of a WEB program written for portability requires a changefile to tailor the program to the target system. These two sets of changes are independent and (hopefully) mutually exclusive, since the MWEB changes have to do with the program logic and the implementation changes deal primarily with the interface to the operating environment, but both must be applied to the same WEB source file. TANGLE and WEAVE, the WEB processors, expect only one changefile containing all of the changes to be incorporated. WEBMERGE can combine the two sets of changefiles into single files acceptable to the two programs.

Another valid use of WEBMERGE is in changes to be permanently applied to the main WEB file of a program, such as the updates to METAFONTware made by Tom Rokicki in the October 1986 TUGboat. These changefiles were not intended to be used as input to TANGLE and WEAVE, but were printed as if they were changefiles to guide the installer in making the changes directly to the WEB files with a text editor. WEBMERGE could have been used to process them like any other changefile to create a new WEB file.

The implementation of WEBMERGE is conceptually closer to the stand-alone TIE program described by Guntermann and Rülling than to the modifications Appelt and Korn made directly to TANGLE and WEAVE. Virtually all of Guntermann and Rülling's article applies equally well to WEBMERGE. Both programs apply multiple changefiles to a WEB file and generate either a new WEB or a composite changefile containing the combined changes.

The basic difference between TIE and WEBMERGE is in how the multiple changefiles relate to each other. The operation of the two programs should be pretty much the same when there is no change conflict (the case where more than one of the changefiles tries to modify the same lines of code), but the programs operate very differently when conflicts occur. In the sequential approach taken by Guntermann and Rülling, "the addition of changefile f_{i+1} behaves as if the changefiles f_1 to f_i had been merged into the WEB program before". The problem I see with this approach (assuming I am understanding it correctly) is that it requires the changefiles to be aware of the existence of each other. In other words, if changefiles f_1 and f_2 modify the same parts of a program, file f_2 must be written to modify f_1 rather than the WEB file itself. This precludes using f_2 without f_1. If the changes made by the two changefiles are truly independent, then it should be possible to handle them independently as well. It might be desirable to apply f_2 with a different f_1 or by itself. To reuse the MWEB example from above, changefiles WEAVE.VAX and MWEAVE.CH exist, both based on WEAVE.WEB. Applying MWEAVE.CH to WEAVE.WEB produces the generic version of MWEAVE (Modula-2 WEAVE). Applying WEAVE.VAX to WEAVE.WEB results in the VAX-specific version of regular WEAVE. Merging WEAVE.VAX and MWEAVE.CH together results in MWEAVE.VAX, which can then be used to create the VAX version of MWEAVE. Either of these changefiles can be used alone or with the other with no modifications. Also, MWEAVE.CH can be merged with a completely different implementation changefile to produce MWEAVE for another environment, without changing either file.

In contrast to TIE, WEBMERGE applies all of the changefiles to the original WEB file in parallel. If a conflict occurs, one of the changefiles is selected to apply that change, and the others are flushed (in this case, "flushing" a changefile means discarding the current change section for that file and moving ahead to the next one). A warning message is sent to the screen identifying which two files had a conflict, which file was flushed, and the source line on which the conflict occurred. Both the line number and the contents of the line are displayed so it is easy to determine exactly where the conflict occurred. Which file is used and which ones are flushed depends on how the conflict occurs. Two rules apply: a changefile with a matching operation already in progress has precedence over any others which match later lines; if no change is currently in progress and more than one file matches on the same

line of the WEB file, the higher priority changefile is used. Priority refers to position within the list of changefiles (f_1 would have a higher priority than f_2).

Conflicts when merging changefiles are inevitable. While significant conflicts are not very likely, since the changes being merged are normally for different purposes and modify different portions of the code, conflicts of a trivial nature occur often. For instance, many WEB programs follow the example of Stanford and output a "banner line" to the terminal to identify the program and its version level, as in:

```
@d banner=='This is WEAVE,
   Version X.X'
```

Nearly all changefiles modify this line to reflect what change they are making to the program, such as:

```
@d banner=='This is WEAVE
   with hyperspace option, ...'
@d banner=='This is MWEAVE,
   Modula-2 WEAVE, ...'
```

for modifications to the logic of the program itself or

```
@d banner=='This is WEAVE,
   VAX/VMS Version ...'
@d banner=='This is WEAVE,
   Microsoft Pascal Version ...'
```

for the various implementation changefiles. However, when multiple changefiles are being merged, the banner line of none of them is correct, since the version of the program actually executing is a combination of the two:

```
@d banner=='This is MWEAVE,
   VAX/VMS Version ...'
```

The \title command in the "limbo" portion of a WEB program falls in the same category as the banner line, since it is also a target common to many changefiles.

The solution to this problem is to create a *third* changefile containing nothing but conflict resolutions. Its change sections would consist only of the composite banner line and title. It should be placed first in the list, so that its changes will override all of the others. Since the conflicts it addresses are expected, the warning messages can be ignored. (It goes without saying that any *unexpected* conflicts which surface must be analyzed to insure that they don't change the logic of the program to an uncompilable or unexecutable state.)

If the sequential approach of TIE is truly needed, the case where one changefile needs to be fully applied before the second one is applied to the result of the first, this can be accomplished serially by using WEBMERGE to create an intermediate WEB file and then applying the second changefile to it. Of course, this does require additional steps, but that's what batch files and command procedures are for.

Hopefully, WEBMERGE should be available from Stanford on the regular distribution tape by the time this reaches print. The WEB files and the VAX implementation files should be available from Stanford and additionally from Kellerman and Smith. For the people who have absolutely no way of reading a magnetic tape, the IBM PC version is available from me on PC floppies for a handling fee. Additionally, the original TANGLE and WEAVE, the MWEB system described elsewhere in this issue, and several of the TEX and METAFONT utility programs (sometimes referred to as TEXware and METAFONTware) are also available on floppy. All of these have change files targeted for Microsoft Pascal running under MS-DOS on the IBM PC, which is my development system. As far as other target computers are concerned, WEBMERGE was cannibalized from TANGLE, so it should be possible to adapt the current implementation-specific changefile for TANGLE without too much difficulty. If you have TANGLE running, you should have no trouble with WEBMERGE.

How to MANGLE Your Software: The WEB System for Modula-2

E. W. (Wayne) Sewell
Software Engineering Specialist

Standard Pascal is an incomplete language from a real-world production software point of view. This is not surprising, since the language was originally designed by Niklaus Wirth as a tool for teaching structured programming, and was never intended for development of production code. The only reason for the widespread use of Pascal is that the various implementors extended the language tremendously when they developed their compilers. VAX Pascal is a good example of a full-featured production compiler. Its many extensions to Pascal allow sophisticated systems to be developed with it. Virtually *every* implementation of Pascal has to extend it in some way, since standard Pascal (as described in Jensen & Wirth) is absolutely