J11.3

TECHNIQUES FOR SELF-MAINTAINING WEB SITES

Julia A. Collins*
NOAA-CIRES Climate Diagnostics Center
University of Colorado
Boulder, Colorado

1. INTRODUCTION

In this paper, we describe our experiences using Cascading Style Sheets (CSS), document templates, and Apache Server-Side Includes (SSI) to create a flexible architecture for the Climate Diagnostics Center web site. Many software systems exist for creating and deploying web hierarchies. At CDC, only a small fraction of our staff is dedicated purely to Web service design and maintenance. We therefore rely on many different individuals throughout CDC to maintain a substantial amount of our Web content. These individuals have a variety of computing environments on their desk tops and have varying levels of expertise. Our challenge was to create a development and maintenance environment that enables this diverse "support staff" to produce a web site that is reliable, documented, secure, accessible and consistent in its appearance and navigational structure.

2. ANATOMY OF A WEB PAGE

Figure 1 shows a very general web document. All of the information presented in the document is considered to be content. For purposes of this discussion, the content will be further broken down into a navigation component and a body content component. The specific geometry and positioning of these components make up the layout of the document.

3. WHY MANAGE CONTENT?

Web sites seem to grow in an exponential manner, and what was once a small, contained site can quickly turn into an overwhelming collection of documents and script output. Usability of a site is heavily dependent upon consistency in page appearance (the “look and feel”) and the use of hyperlinks which guide the user to their desired information (navigation links). As the size of a site grows, updating the look and feel and navigation links on a file-by-file basis becomes unrealistic.

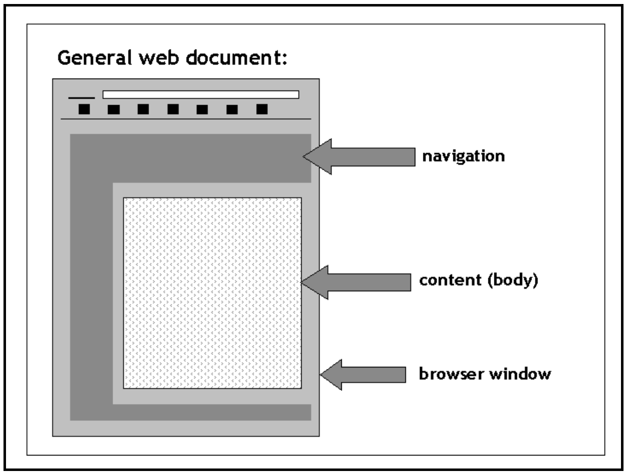


Figure 1

Additionally, the page author may not have the knowledge or expertise to create or maintain those navigation links. Managing the navigation content in a centralized fashion minimizes the effort needed to make any changes to the document navigation and look and feel. Individual page authors are relieved from dealing with those details. This approach is scalable in that the effort needed to manage the page layout and navigation content does not change appreciably as the site grows.

4. OUR REQUIREMENTS

In some cases page authors need to include additional navigation information specific to their page contents. Additionally, contributors to our web site vary widely in their technical expertise and knowledge of document editing tools. As a result, we wanted to minimize the HTML knowledge needed by page authors, while still providing a means for more HTML-literate contributors to take advantage of Javascript and CSS in the body of their web documents.

We currently serve CDC web documents via an Apache web server running on a UNIX platform. Our content management approach needed to be able to work

**Corresponding author address:* Julia A. Collins, NOAA/ERL/CDC (R/E/CD1), 325 Broadway, Boulder, CO 80303. E-mail: jac@cdc.noaa.gov

within that environment. Finally, the documents served from this system need to adhere to the standards for the technologies they use (e.g., HTML, XML) and to conform to the policies established by governing organizations (e.g., Section 508 of the Rehabilitation Act Amendments in the U.S.).

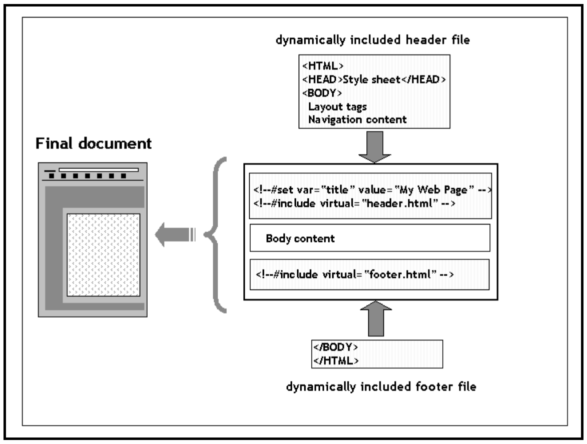
5. COMMERCIAL TOOLS

Although commercial site and document management tools exist, their expense and concerns about the quality of the resulting HTML prove to be disadvantages. As a result, we focused on tools already at hand.

6. TAKING ADVANTAGE OF SERVER CAPABILITIES

The Apache mod_include handler provides a simple means of dynamically determining the content of an HTML file. The mod_include features include the ability to set variables, test variables for particular values, parse environment variables, include the contents of other HTML files, and apply conditional statements to these actions. To use server side includes (SSI), the mod_include module must be included in the Apache build (or dynamically loaded), and the server must be configured to parse those files which contain SSI. Enabling SSI for a collection of HTML documents does not necessarily provide the ability to execute Common Gateway Interface (CGI) scripts. For security reasons, we do not enable the “#exec” option except in a limited number of well-defined cases.

Figure 2 is a highly-simplified schematic of the file construction process. The “set” option is used to declare variables which define the page title, its relationship to other files at our site, author identification information, and pointers to style sheet or JavaScript code needed for the body content. The “include” command is then used to include a header file. The header file contents handle the positioning of the navigation elements and provide the HTML tags needed to start the document. After the body content, a footer file is included which “closes” the HTML tags and inserts a standard footer text.


dynamically included header file
Final document
Body content
dynamically included footer file

```
<HTML>
<HEAD>Style sheet</HEAD>
<BODY>
  Layout tags
  Navigation content
```

```
<!--#set var="title" value="My Web Page" -->
<!--#include virtual="header.html" -->
```

```
<!--#include virtual="footer.html" -->
```

```
</BODY>
</HTML>
```

Figure 2

7. THE NEXT GENERATION

To achieve our immediate content management goals we compromised between the extreme of using a single software package and that of hand creating all of our documents. By taking full advantage of Apache server side includes, we were able to separate the body content of our documents from the layout instructions and navigation content. This allows page authors to use their preferred editing tool to create their page content, and then insert the content into a simple template file which provides the navigation elements and graphics that give the site a consistent look.

One drawback to relying on Apache SSI is the lack of database integration. PHP, a preprocessor using an HTML-embedded scripting language, provides this capability. Other technologies which may prove to be more flexible and robust than SSI in the long term include Java servlets and Java Server Pages (JSP). We plan to investigate these options as part of our ongoing site maintenance efforts.

8. REFERENCES

Apache HTTP Server Version 1.3, cited 2001: Apache module mod_include. [Available on-line from http://httpd.apache.org/docs/mod/mod_include.html]

Apache Week, cited 2001: Using Server Side Includes. [Available on-line from http://www.apacheweek.com/features/ssi]