A Hypertext for Literate Programming

M. Brown
Department of Computer Science
University of Alabama
Box 870290, Tuscaloosa, AL 35487-0290
(205) 348-6363
marcus@fred.cs.ua.edu

B. Czejdo
Department of Mathematical Sciences
Loyola University
New Orleans, Louisiana 77004

Abstract

In this paper we describe a hypertext presentation for the WEB system for literate programming. The requirements for an environment for modern literate programming are analyzed and the WEB hypertext system is proposed. Different types of windows for displaying text, indices and graphical representations are discussed. All semantic links and operators using them are analyzed. The proposed architecture of the WEB hypertext system includes a general purpose relational database management system. Mapping between hypertext queries and relational database queries is provided. The system allows the user to define new hypertext operations by providing the corresponding SQL queries for the underlying relational database. The described system can be easily extended by many other types of queries and applications.

Keywords: Literate Programming, Hypertext, Database

I. INTRODUCTION

One current research topic in the area of information systems is concentrated on devising mechanisms to extend the traditional notion of "flat" text to a hypertext presentation [10]. As a result of this extension the text is treated as a structure with complex organizational links allowing direct references from one part of the text to another. While many texts do not have such links as a natural part of their presentation, computer source programs have a structure which often allows for relatively straightforward division into fragments with connecting semantic links. In particular, the WEB [5] approach to literate programming requires extensive usage of semantic links. An experimental environment for browsing and editing WEB computer source programs has been proposed [3]. In this paper we present a WEB Hypertext System (WHS) with extendible database and user defined operations.

The paper is organized as follows. In the next section, we describe the concept of literate programming together with the requirements for an environment for modern literate programming. Section 3 presents a WEB hypertext environment for literate programming while the operators for the environment are discussed in Section 4. The architecture of a WHS with the underlying database is described in Section 5. Mapping between hypertext queries and relational database queries is also provided in this section. The Summary presents some conclusions and directions for further work.
II. THE WEB SYSTEM FOR LITERATE PROGRAMMING

Donald E. Knuth coined the term Literate Programming to describe his concept that programming should produce works of literature. Literate programming should concentrate on explaining to the human reader/programmer what the computer is supposed to do, rather than concentrating our effort on simply writing instructions for the computer [5]. Knuth insists that the format and structure of a literate program should be designed to communicate primarily with the humans who read the source program rather than the computers which perform the program. Programming in this way should produce better programs with better documentation of those programs.

Knuth created the WEB system in 1981 in order to rewrite his \TeX program in the most portable and understandable manner possible [6], the basic concept behind literate programming. WEB allows a single source program to produce two different results: a running program on a computer and a typeset document which defines and describes that program for human readers. WEB consists of a combination of a high level programming language and a typesetting language (\TeX) with additional commands to control the relationship between the two languages and to allow for modularization: The WEB system applies two preprocessors to the source. TANGLE creates programming code suitable for the compiler, while WEAVE creates \TeX code which produces the typeset documentation, as is shown in Figure 1.

![WEB File Processing Diagram](image)

A WEB source program contains WEB modules. Each module contains up to three parts: documentation, macro definitions, and programming code. The documentation section may include any relevant commentary. The macro definitions are similar to the \#define statement in C, but are programming language independent. The programming code may be in any high-level language for which WEB has been implemented (currently Pascal[5], C[8], Modula-2[11] and Ada[9].) For the purposes of this discussion the terms “function,” “procedure” and “subroutine” will be used interchangeably to refer to a syntactically complete subprogram as defined by the programming language. “Module” on the other hand, refers to a section of WEB code which may or may not be functionally equivalent to a “subroutine.”

The listing of a WEB source program provides a full index of all the variable names, module names, functions, and procedures in the source program. This is not a new feature, but it is usually not an integral part of the source program listing. One new aspect of the WEB index is that the programmer can include strings in the index as he chooses. These selected strings or “topics” (also called “control texts”) are used to indicate some special feature or problem of the current module.

Literate programming in WEB provides important advantages over traditional programming in four different ways:

1) WEB encourages organization of code based on psychological rather than syntactic divisions. The code can be modularized as the programmer chooses by separating