# A Database of Course Materials in Computer Architecture

Edward F. Gehringer
Department of Electrical and Computer Engineering
and Department of Computer Science
North Carolina State University
Raleigh, NC 27695-7911
efg@ncsu.edu

## Abstract

Teaching computer architecture, like any course at the college level, requires the instructor to spend much time in preparation outside class. This time is spent in writing lectures, making up homework problems and labs, and devising and grading exams. This involves much duplication of effort, since the same problems and labs are, in principle, usable in courses at different universities. Although a small amount of sharing has gone on an ad hoc basis, until now, there has been no way to search for and retrieve a very wide variety of course materials.

Instructors teaching computer architecture have been surveyed, and several dozen have expressed interest in participating in this project. Contributors to the database will be asked to supply a pathname(s) for the location of their course materials. For security's sake, problem solutions will be kept in encrypted archives. The unencrypted portion of the database will be managed very similar to the technical-report databases now available for browsing over the World-Wide Web. It will be possible to search for problems, lectures, or other material, containing arbitrary keywords. The encrypted portion of the database will require new interfaces, or possibly new browser software to be written. Development of this database will encourage reuse, and aid in improving the quality of courses and decreasing the time required to teach them well.

## 1. Teaching Productivity

Productivity is a basic measure of performance in most industries. Manufacturing companies measure it by output per unit of labor. In the academic community, we have derived measures for research productivity, such as number of publications, number of refereed papers, and amount of research funding. To be sure, considerable controversy surrounds these measurements; although they may be easy to compute, they are often seen as inappropriate measures of quantity rather than quality. But the fact remains that the question of productivity is at least being addressed.

In teaching, the opposite problem seems to prevail. At the university level, teaching evaluations by students are widely used as a measure of quality. Controversy besieges this metric too, but it at least exists, and a considerable literature has developed around it. Quality, however, is very different than productivity; a straightforward application of the manufacturing definition would imply that high productivity is synonymous with large classes, and almost everyone "knows" this to be false. To the extent that a notion of teaching productivity exists, it seems to be associated with the time that students and faculty are in personal contact. The public (and many legislators), at least, seem to view teaching this way, and this leads to persistent demands for faculty to spend more hours "in the classroom."

Now, anyone who has ever taught at the university level, particularly in technical fields, realizes that behind every hour of lecture there stand many hours of preparation—in my case, six to ten for every new lecture assembled, and eight to twelve for every exam. The fact, however, that this effort is so little appreciated suggests that we look for ways to reduce it. Not only would this improve our profession's public image; it would also free more time for research, which would allow faculty to spend more time on new ideas, and thereby achieve promotion and advance in their careers.

Much of the routine activity associated with teaching is susceptible to optimization via sharing and cooperation. Not every instructor who teaches a course need write a new lecture from scratch; chances are that the lecture of another instructor with a similar teaching style could be adapted more quickly than a new lecture could be written. Faculty spend enormous amounts of time designing tests, homework problems, and (especially) labs, and these are even more reusable. Instructors get little credit for repetitive activity like creating yet another problem on a familiar topic, and they do not particularly like to do them. More enjoyable are tasks that pique student interest, like synthesizing course material and presenting lectures. It is not particularly interesting to do this "from the ground up." It is better to do this by assembling parts that someone else has written, just as it is more rewarding to program with high-level libraries than in assembly language. Indeed, by cutting the time needed to integrate new material into a course, reuse makes it possible to incorporate new topics more quickly, enhancing the students' educational experience.

This author is aware of only one experiment in reusing course materials—the University of California at Berkeley/University Video Communication Pilot Program to Aid New Faculty in Lecture Preparation. Announced by Dave Patterson in 1993, this program made videotapes and notes for all lectures by outstanding Berkeley instructors available at a modest fee to faculty teaching similar courses at other schools. What is most surprising is the uniqueness of this effort; a search of two on-line databases turned up no references to articles on teaching productivity, where "productivity" was defined in a manner even remotely similar to that described above. Most of what has been written focuses on teaching vs. research productivity. In fact, among the articles with the word "productivity" in the abstract, there were no articles at all on saving faculty time in course preparation. The Berkeley/UVC program is a laudable beginning, but it focuses solely on lectures. This is an important foray in improving productivity, but by no means the whole battle.

## 2. Barriers to Sharing and Cooperative Work

Instructors in a course rarely share many materials with others in similar courses. Often this is because problems are not completely written up, existing in bare outline form that is not intelligible without access to a complete (unpublished) lecture. Even when such lectures are published completely (as in several cases where instructors have put their lectures on the World-Wide Web), they may consist only of an electronic version of transparencies, or some similar outline form, and be understandable only when the instructor orally explains the topics and fills in the gaps. Beyond that, there are differences in teaching styles. Some instructors approach a topic in a much more mathematical way than others, even when using the same textbook. Others emphasize the practical knowledge that comes from writing or modifying programs to illustrate a concept. Still others choose to give design problems, which challenge the student's analytic skills. To a great degree, these choices reflect the instructor's own knowledge and background, and cannot be effectively used by an instructor whose background and therefore style is different. Even when styles are similar, a good problem may require familiarity with several concepts, some of which another instructor may not have covered. Or a prerequisite, such as a course in probability, may be required at one school but not another.

Cooperation is meant to save time, but it has a high overhead itself. It is necessary for an instructor to locate another instructor with similar a similar background and style. Sharing must be with someone from a different school, because at the same school, many students would have access to files of problems from previous terms. It takes time to gather together a whole set of materials, package them, and mail them. Then the recipient must manually search for problems that are appropriate to his/her course.

## 3. Advantages of Exploiting the World-Wide Web

Enter the Internet and the World-Wide Web. The widespread use of wordprocessing has resulted in a wealth of machine-readable course materials, and the availability of networked computers for instruction has led to many of these materials being placed on line. Our survey reported below shows that dozens of instructors have most or all of their materials on line. Enough material in computer architecture is now on the Internet to make up a database of gratifyingly comprehensive proportions. With the creation of the necessary indexes and links, it will be possible to search quickly for materials matching a given style. One will be able to do a keyword search, and then eliminate materials that do not appear to be the right style. Cooperation is facilitated: it is nearly as easy to share with instructors across the world as to share with one other site. Transmission is nearly instantaneous, and searching for and selecting problems will not take much longer.

## 4. Feasibility of the Database

To determine the interest in a database of course materials, 144 computer architects at universities throughout the world were surveyed by e-mail. The list was drawn from previous ISCA attendees, attendees at computer-architecture education workshops, and people to whom the author was referred by members of the above groups. Results of the survey are shown in Table 1. Response was very gratifying. Fifty-eight instructors at 39 schools have agreed to participate. Only one declined (he cited lack of time). Many of the others responded that they were not teaching computer architecture; some of these were students (although an attempt had been made to filter out students by `finger`ing people before sending mail, and removing those whose `.plan` file mentioned a thesis topic or whose home directory was on a suspicious structure like `/students` or `/grad`). Some of the others were research faculty, or teaching faculty teaching courses in other areas. Several of the 58 participants had already put their lectures on the Web, and two-thirds of the 58 have more than half their material in machine-readable format. Of the 20 who have no more than half in electronic form, what is on the computer is usually homework and exams—just what we are most interested in sharing.

A few formats predominate. LaTeX is most common, followed by FrameMaker and Microsoft Word. Seventy-one of 103 responses mention one of these three (there are more than 58 responses because many instructors use more than one wordprocessor). If PowerPoint is included, the top four formats comprise more than 80% of total responses. The decline of older formatters is not unexpected; only three responses mention `troff` or its derivatives. More surprisingly, WordPerfect and Interleaf were mentioned only once each.

Other observations on the survey results are that courses are about equally divided between computer organization and architecture. If you count the parallel-architecture courses, the balance shifts in favor of computer architecture. Some respondents noted that there doesn't seem to be a clear dividing line between organization and architecture, but most had no trouble classifying the focus of their course as one or another. Among the courses that fell in category e ("other") were a course on interconnection networks and a general electronics lab that included interfacing and assembly-language programming. The courses are about half undergraduate and half graduate. The most popular texts are the two by Hennessy and Patterson, with Hamacher, Vranesic, and Zaky coming in in third place (or fourth place, if you count "no text" or "own notes," which would have come in second). For labs, programming assignments were about twice as common as hardware-oriented labs; fewer than half the courses included the latter. Those who assigned programs tended to use C and/or C++. VHDL appeared to be the most popular hardware-description language (but the responses were very sparse; this was not a digital-design survey). For parallel programming, the ANL macros were used at four schools. Surprisingly, PVM was used in only three schools (but remember, the survey focused on architecture, not parallel programming).

## 5. How the Database Will Work

With respect to the software involved, the computer-architecture database is very similar to computer-science technical report databases available currently on the Web [Fox 95]. Information is maintained at sites around the Web. Directory information (abstracts, etc.) is gathered at one site where it is inserted into a database which can then be searched.

*Construction*. Participants will furnish a list of directories where their course materials are located. Information will be `ftp`ed from these directories. One way would be to make the directories available via anonymous `ftp` (except for sensitive material like solutions, which would have to be protected). The simplest way to protect sensitive material would be to encrypt it using the public key of the database program. The database program will periodically `ftp` all the files that have changed since they were last `ftp`ed. We will devise scripts that will parse the files, discovering the boundaries between problems. A database entry will be created for each problem (lecture, etc.).

*Retrieval*. A Web page will be provided, with a form that can search for entries (e.g., problems, lectures) that include certain keywords. Only problems will be displayed; to fetch solutions will require authorization, as described below. The user will be able to scroll through the list of problems in the browser, and `ftp` any that are desired.

**Table 1:** Answers to Survey Questions

1. What is the main topic of your course?

| | |
|---|---|
| a. Computer organization | 33 |
| b. Computer architecture | 34 |
| c. Parallel computer architecture | 18 |
| d. Parallel architecture and programming | 13 |
| e. Other | 8 |

2. What level is your course taught at?

| | |
|---|---|
| a. Sophomore-junior | 22 |
| b. Senior | 22 |
| c. Beginning graduate | 34 |
| d. Graduate, research oriented | 12 |

3. What textbook(s) do you use?

| | |
|---|---|
| Hennessy & Patterson, *Computer Architecture: A Quantitative Approach* | 22 |
| Own notes and/or papers | 15 |
| Hennessy & Patterson, *Computer Organization & Design: The Hardware/Software Interface* | 7 |
| Hamacher, Vranesic, and Zaky, *Computer Architecture and Organization* | 5 |
| Kai Hwang, *Advanced Computer Architecture* | 3 |
| Mike Johnson, *Superscalar Microprocessor Design* | 3 |
| Andrew Tanenbaum, *Structured Computer Organization* | 3 |

21 other books received one mention each.

4. What kind of homework do you assign? (“Check” all that apply.)

| | |
|---|---|
| a. Labs (hardware oriented) | 22 |
| b. Non-mathematical problems to be solved | 47 |
| c. Mathematical problems to be solved | 38 |
| d. Design problems | 48 |
| e. Programming assignments | 42 |

5. What programming languages and/or software packages are used in your course?

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| C | 28 | VHDL | 5 | C/C++ | 3 | C-Linda | 1 |
| C++ | 11 | ANL macros. | 4 | PVM | 3 | FabricFactory | 1 |
| M68000 | 7 | DLX | 4 | MIPS | 3 | Occam | 1 |
| Dinero | 6 | Pascal | 4 | | | | |

6. How many of your course materials (e.g., lecture notes, handouts, problems, exams) are in machine-readable format?

| | |
|---|---|
| None | 4 |
| Less than half | 14 |
| Half | 2 |
| More than half (but < 99%) | 32 |
| At least 99% | 15 |

7. Which wordprocessor or document formatter was used to produce the (e.g., LaTeX, Scribe, FrameMaker, Interleaf, WordPerfect, Microsoft Word)?

| | | | |
|---|---|---|---|
| Latex | 32 | troff, ditroff | 3 |
| FrameMaker | 20 | ASCII | 2 |
| Word | 19 | Interleaf | 1 |
| PowerPoint | 11 | Postscript | 1 |
| HTML | 5 | WordPerfect | 1 |

## 6. Options for Access Control

A database that contains problems *and* solutions will be much more useful than one that contains problems alone. This is not only because it saves the instructor the trouble of solving the problems that he assigns, but also because it is sometimes not possible to judge the suitability of the problem unless one knows what the solution entails. So, without solutions, one might have to solve, say, ten problems in order to assign five.

However, the presence of solutions in a database requires access control, to prevent students from "doing" their homework via `ftp`. There are two options:

1. Do not incorporate privileged materials into the database; requests to retrieve them must be directed to the server on which they are stored.
2. Store privileged materials locally to the database; requests to retrieve them are directed to the database server.

Option 1 requires secure communications between database users and various sites. All sites will have to "know" about all authorized users, and authenticate all requests to retrieve solutions. This seems to be a less viable option than the following one.

Option 2 requires more space on the database server, but only the database server ever needs to authenticate a request. Authentication can be performed with secure Netscape Navigator, which is free to nonprofit organizations. Secure Netscape Navigator uses the Secure Sockets Layer (SSL). A reference implementation, Netscape SSLRef, is also freely available. It is designed to aid efforts to provide secure communications in applications that use SSL. Use of secure Netscape Navigator would limit retrieval of privileged materials to those who can run Netscape. Netscape, however, is freely available for most platforms.

While security software for the clients is freely available, the same is not true of the server. To operate using security features, the Netscape Commerce Server requires a digitally signed certificate, which is available for a fee, probably $5,000–$10,000. In essence, the security problem posed by this database is the opposite of the problem faced by commercial transactions on the Internet. Commercial applications require information (e.g., credit-card numbers) to be sent securely to the server. The user does not normally care if the confirmation returned by the server is sent securely. The database, on the other hand, requires information to be returned securely *from* the server. It would not be much of a problem if the request for privileged material were intercepted. (It would only inform a student that the instructor had requested certain problems.) However, the information needs to be sent back securely. It is not clear at present whether everyone who needs to receive secure information has to be running on a secure server. If so, a significant cost would be imposed on everyone. In this case, a viable alternative would be to return solutions via e-mail using PGP encryption.

## 7. Responsibilities of the Participant

The responsibilities of the participant in the database depend on whether privileged or unprivileged access is requested, and whether information is to be made available ("published") or just retrieved.

- For unprivileged retrieve access, no work is necessary beyond running the database browser.
- To publish on the database, the participant will have to tell the administrator where the relevant files are located. The administrator will also need `ftp` access to this site (anonymous `ftp` will be sufficient). If privileged information (e.g., solutions) is to be published, it must be encrypted with the public key of the database administrator before being put in a public place.
- To retrieve privileged information, the participant must have in place software to decrypt the information. This could either be incorporated into a secure Web browser or be PGP decryption software used on privileged information returned encrypted in e-mail.

## 8. Plan of Action

The first phase in constructing the database will be to bring up the unprivileged part. The software is very similar to that used in tech-report browsers; one of these is currently under construction at NCSU. If privileged information is desired during this first phase, it will have to be obtained by sending e-mail directly to the author of the problem in question.

In this first phase, students will be retained to program "stanzas" to parse the published files into problems (and later, solutions), and insert them into the database. These stanzas are short scripts that can identify the beginning of a problem, and the beginning of a solution.

The second phase will involve secure transmission of privileged information. Funding will be obtained to acquire commercial security mechanisms, if necessary. Software to send encrypted e-mail will be written. A "secure client" will also be written, if necessary.

## 9. Benefits of the Database

Availability of the database will provide a host of benefits and opportunities for further research. First, it will save time for instructors, allowing more time for research and contact with students. It will tend to raise the quality of homework and exams, since instructors will be able to obtain access to good problems developed by others. Instructors will be able to concentrate on making up just a few good problems in their area of expertise, rather than devise all of the homework for a course every time it is taught. Problems within the database could serve as source material for textbook authors; the contributor of a problem should, however, control the usage of that problem for commercial purposes, possibly negotiating royalties for its inclusion in a text.

This project opens several new vistas for future work. Instructors can keep track of whose materials they reuse most frequently; eventually, software may even be devised to keep such tallies automatically. This may suggest possible collaborators for work on developing future courses or course materials. Also, instructors can be encouraged to report which problems or other materials they have actually reused. Counts can be kept, presumably automatically, and in the not-too-distant future, awards could be given to the most-reused materials. This could be one approach to the problem of devising quantitative measures for accomplishment in teaching.

One might characterize the "style" of an instructor by the problems (s)he reuses. Do teaching styles tend to fall into certain categories? Do certain styles receive higher student evaluations than others? Is it possible to predict what problems an instructor will be most interested in from the problems he has used in the past? If so, such problems could be displayed first in the browser.

The tenet of reuse has long been respected in hardware and software architectures alike, even when honored in the breach. The time is ripe to apply this principle not only in what we are teaching about, but also in how we go about teaching.

## References

[Fox 95] Fox, Edward A., "World-Wide Web and computer science reports," *Comm. ACM* 38:4, April 1995, pp. 43–44.