CS593 Transaction Processing
Course Information — Winter (January – March) 1999

Instructor: Philip A. Bernstein, philbe@microsoft.com, (425)936-2838
http://www.research.microsoft.com/~philbe/
Meeting Time: Tuesday 6:30 – 9:20 PM, January 5 – March 9. Final exam, probably March 16.
Locations: (live) Microsoft Redmond West F/1002, phone 703-6132
(video) Univ. of Washington, EE1 Rm.3
Jan. 12 only: Whole class meets at EE1 003 (i.e. no lecture at Microsoft).
Discussion session: One per week with teaching assistant and/or instructor, as needed, to discuss homework and projects. Proposed times, to discuss at the January 5 course meeting:
Thursdays, 6-7 PM, at UW (likely Sieg 422)
Fridays, 6-7 PM, at Microsoft (Building 31/1027)
Textbook: "Principles of Transaction Processing for the System Professional,"
by Philip A. Bernstein and Eric Newcomer, Morgan Kaufmann Publishers, 1996
Teaching Assistant: Neil Spring, nspring@cs.washington.edu
Web Site: http://www.cs.washington.edu/education/courses/cse593/CurrentQtr/
Mailing List: cse593@cs.washington.edu. as usual, mail majordomo@cs.washington.edu with subscribe cse593 yourmailaddress

Lecture Outline

1. Introduction (textbook, Chapter 1)
2. Transaction Processing Monitors (textbook, Chapter 2)
3. Database Concurrency Control (textbook, Chapter 6)
4. Database Recovery (textbook, Chapters 7 and 8)
5. Two-Phase Commit (textbook, Chapter 9)
6. Transaction Processing Communications (textbook, Chapter 3)
7. Queuing (textbook, Chapter 4)
8. Replication (textbook, Chapter 10)
9. Advanced Transaction Models and Workflow

The above is meant to indicate only the sequence. The exact mapping of topics to course meetings is hard to predict.

Prerequisites

There are no particular prerequisites for the course, besides having a good general understanding of software systems. It's also helpful to have some knowledge of SQL (be able to write simple selection and join queries).

Assignments and Grading

There will be 4 problem sets during the first 5 weeks. Each problem set will be due in one week. There will also be a more substantial project, of your choice, which can be done individually or in groups. The second week and the last 4 weeks of the course will have no assignments to leave you time to focus on the project, though you'll need to start earlier. There will be a final examination. The grade will be a weighted sum of assignments (35%), project (35%), and final exam (25%) (plus an extra 5% weight to your best of the 3).

Any weekly assignment may be handed in up to one week late; no explanation required. There will be a 10% penalty for any assignment that is not handed in by the beginning of the class in which it is due. Assignments will *NOT* be accepted after the solution is distributed (which will never be sooner than one week after the due date). If a major personal problem is interfering with your doing the coursework, please see the instructor to make special arrangements.

Please e-mail your assignment solutions to both the instructor and teaching assistant (philbe@microsoft.com, nspring@cs.washington.edu). Use either Word 97 for Windows or ASCII text. If this plan causes you a serious problem, please talk to the instructor.

Projects

The main goals of the project are to allow you to dig deeper into some aspect of transaction processing (TP) of interest to you and to learn something useful. Initial due dates:

- January 12 – email message outlining one or more project ideas of interest, as a basis for discussion
- January 19 – submit detailed proposal (1-2 pages) of a particular project with particular partners. There will be no assignment during the second week, to allow you to do this.
- January 26 – final plan agreed to by all parties. You should have some written specs by now, or comparable accomplishments depending on the project.

If your project entails working with new software tools, you should be reading documentation and running examples by January 19.

Here are several project ideas:

1. Design and implement a TP application using a TP monitor. Keep it simple enough to get an end-to-end system working well enough to run a demo. In particular, keep the user interface very simple. If possible, do a little performance measurement. Only a short write-up is required to summarize what you have done, as long as the code is readable. A final demonstration is required.

   Some TP monitors make it easy to develop a basic application. In such a case, to receive a high grade , you need to tackle an especially challenging feature. For example, if you use Microsoft Transaction Server, then you could develop a resource dispenser and/or integrate a simple resource manager with the Distributed Transaction Coordinator (2-phase commit). *Make sure you have a solid working solution to your base features before immersing yourself in the challenging feature*.

2. Implement a distributed transaction system in Java. The server will allow customers to make/cancel/query travel reservations, airlines to add/cancel/modify flight information, and hotels and car rental agencies to post room and car availability. We will provide clients, the (likely RMI or something more J++ friendly) interface the clients expect, and a general design framework for the (ultimately distributed) server. Students will build mechanisms for persistence, recovery, locking, and two-phase commit. The transaction system will concurrently support multiple clients and multiple servers.

3. If there is TP dimension to your current job, you could write a paper that describes and evaluates it from a TP perspective. For example, in past offerings, papers were written about the replication algorithms in an email system, the transactional aspects of a queuing communications product, and a large financial trading application.

4. Analyze a TP application and installation currently in production. Interview the application designers, users, and system managers. Read documentation if it's available. Write up what you learned.

   We unfortunately do not have any applications lined up. So to pick this project, you need a personal contact with an appropriate company and the company's agreement to access the people and documentation.

5. If you have access to the software and documentation of a TP monitor not covered in Chapter 5 of the textbook, study it and write a description of at least comparable detail. This could be a home-grown infrastructure specific to a particular application system. Compare the monitor's capabilities to some of those described in the text.