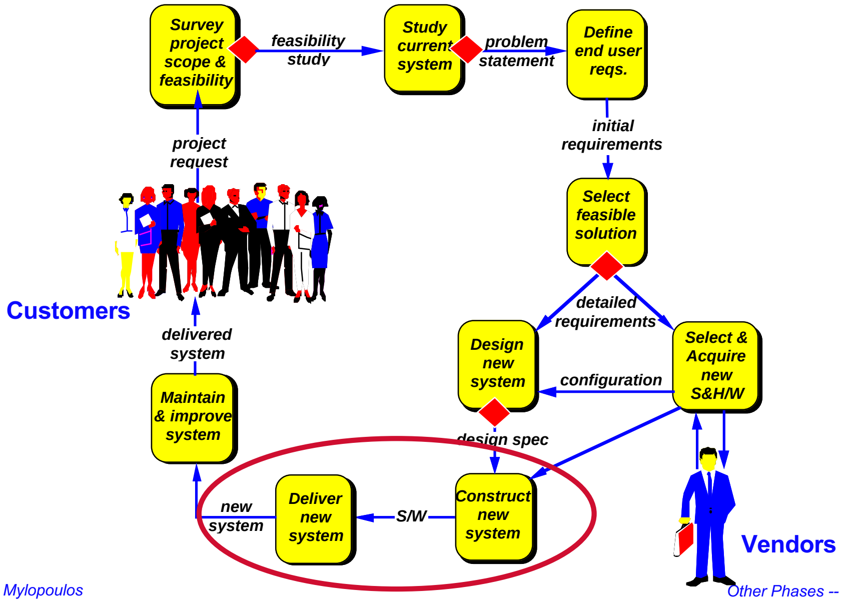
XXIV. Other Phases

Implementation Phase
Testing and Installation Phase
Post-Installation Tasks
Course Summary

The Software Lifecycle Revisited



Survey project scope & feasibility

feasibility study

Study current system

problem statement

Define end user reqs.

initial requirements

Select feasible solution

detailed requirements

Design new system

configuration

Select & Acquire new S&H/W

design spec

Construct new system

S/W

Deliver new system

new system

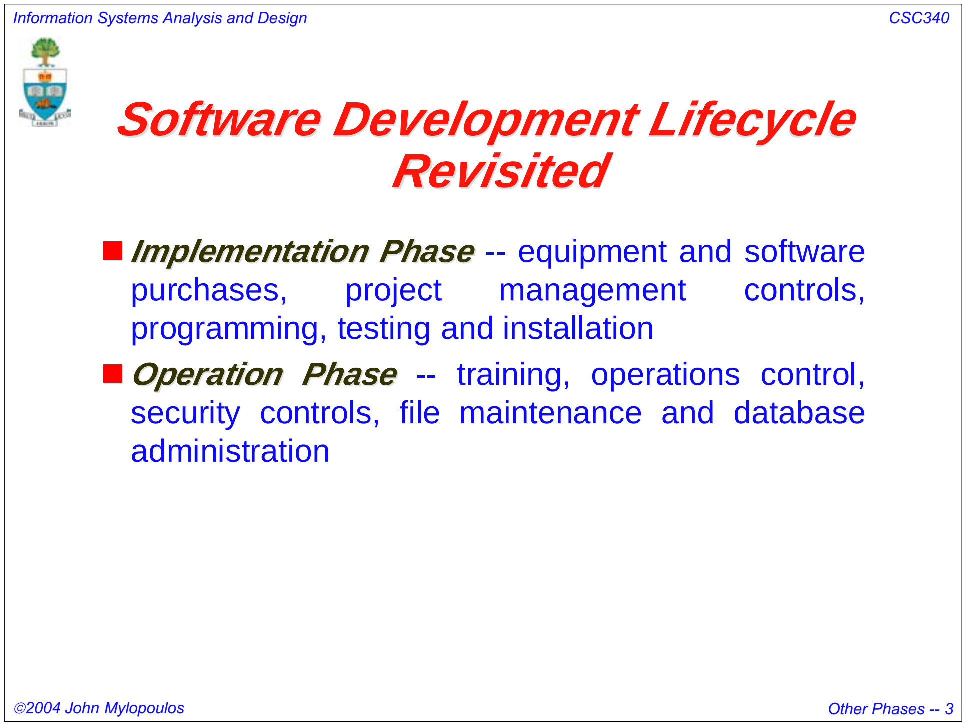
Maintain & improve system

delivered system

Customers

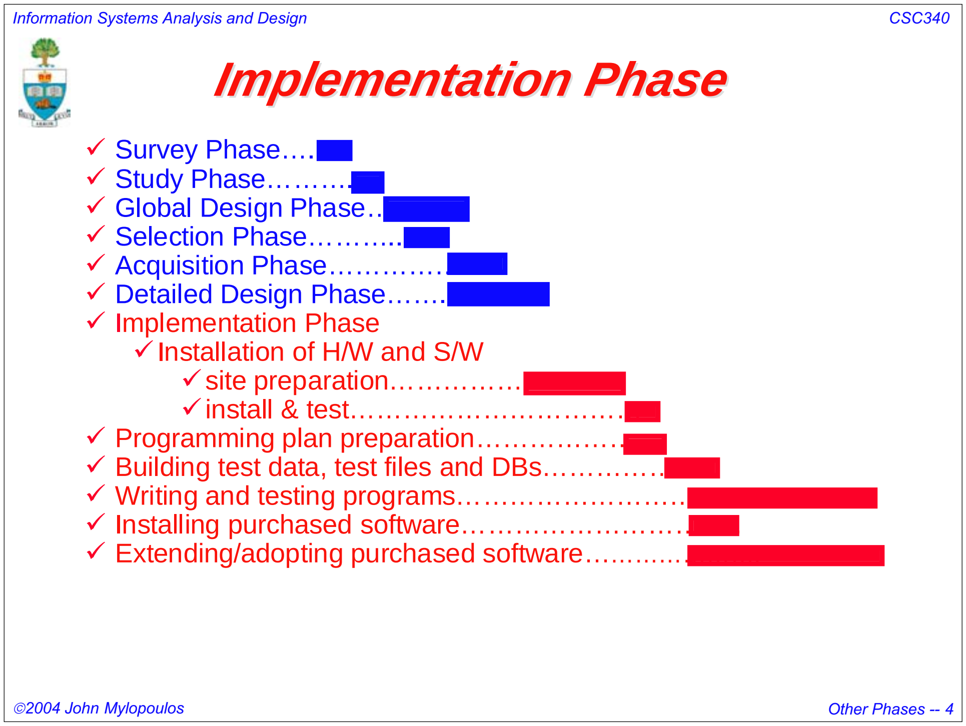
project request

Vendors



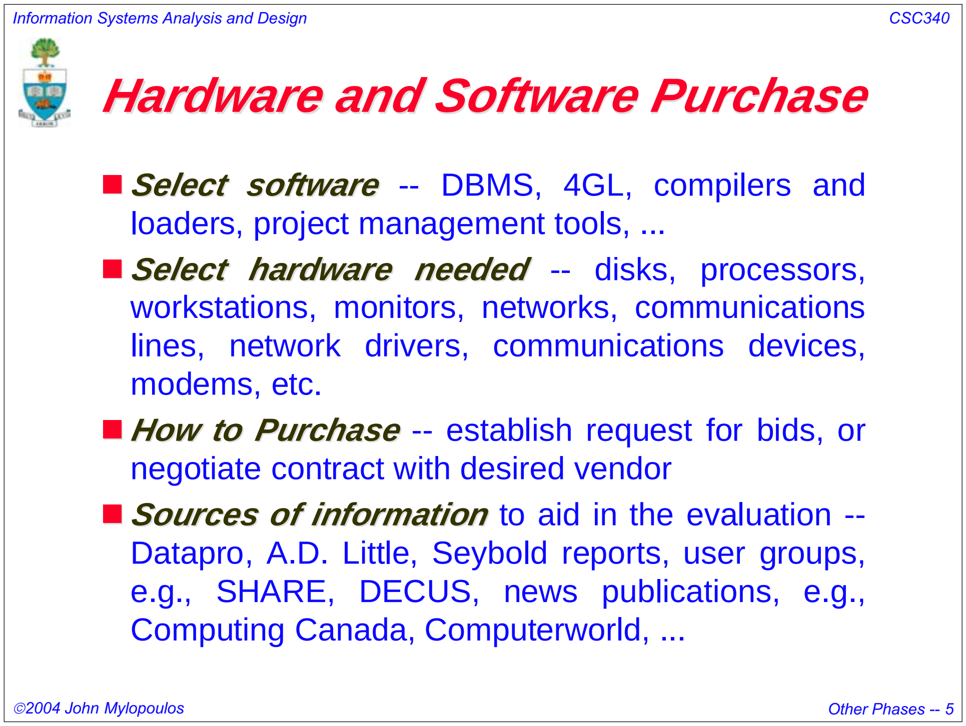
Software Development Lifecycle Revisited

- ***Implementation Phase*** -- equipment and software purchases, project management controls, programming, testing and installation
- ***Operation Phase*** -- training, operations control, security controls, file maintenance and database administration



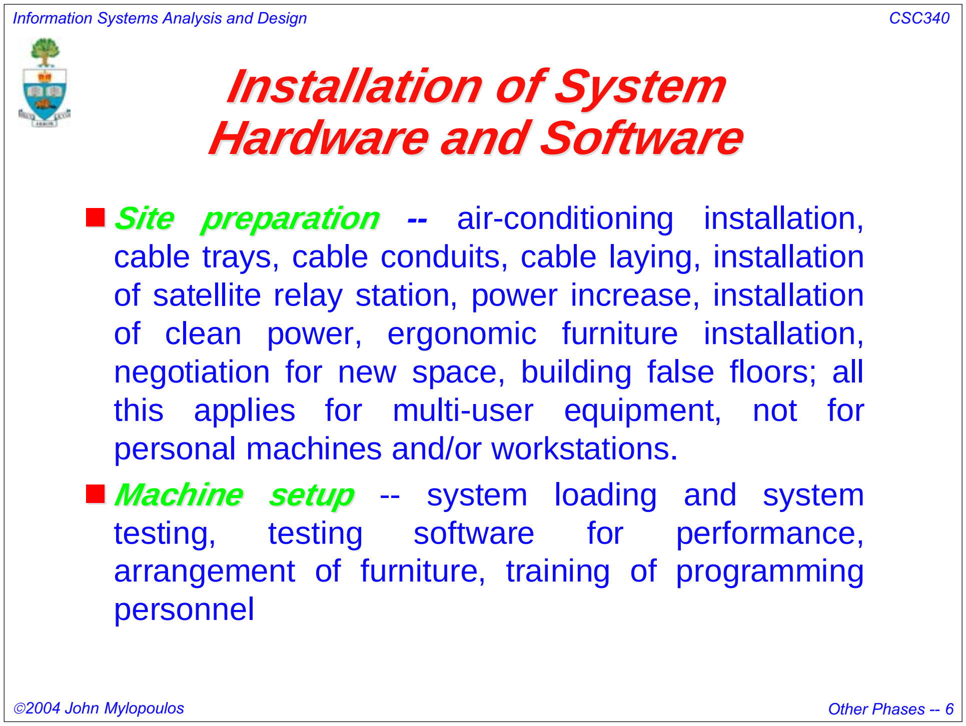
Implementation Phase

- ✓ Survey Phase….
- ✓ Study Phase………
- ✓ Global Design Phase..
- ✓ Selection Phase………..
- ✓ Acquisition Phase………….
- ✓ Detailed Design Phase…….
- ✓ Implementation Phase
  - ✓ Installation of H/W and S/W
    - ✓ site preparation……………
    - ✓ install & test……………………………
- ✓ Programming plan preparation………………
- ✓ Building test data, test files and DBs…………..
- ✓ Writing and testing programs………………………
- ✓ Installing purchased software………………………
- ✓ Extending/adopting purchased software…………



Hardware and Software Purchase

- ***Select software*** -- DBMS, 4GL, compilers and loaders, project management tools, ...
- ***Select hardware needed*** -- disks, processors, workstations, monitors, networks, communications lines, network drivers, communications devices, modems, etc.
- ***How to Purchase*** -- establish request for bids, or negotiate contract with desired vendor
- ***Sources of information*** to aid in the evaluation -- Datapro, A.D. Little, Seybold reports, user groups, e.g., SHARE, DECUS, news publications, e.g., Computing Canada, Computerworld, ...



Installation of System Hardware and Software

- ***Site preparation*** **--** air-conditioning installation, cable trays, cable conduits, cable laying, installation of satellite relay station, power increase, installation of clean power, ergonomic furniture installation, negotiation for new space, building false floors; all this applies for multi-user equipment, not for personal machines and/or workstations.
- ***Machine setup*** -- system loading and system testing, testing software for performance, arrangement of furniture, training of programming personnel

Programming Plan Preparation

- Review the design specifications
- ***Organization of the programmer team*** -- chief programmer, librarian, specialists, programmers
- ***Team Manager*** -- is the person who is in charge of the team and directs the whole implementation.
- ***Development of detailed construction plan*** -- order in which modules will be built and tested; specification of naming conventions and parameter passing conventions; specification of version numbers of system and development accomplishments; specification of control procedures

A Note: Chief Programmers

- In the seventies, software teams were defined around chief programmers.
- ***Chief programmer*** -- is the person who ***conceives*** and directs the whole implementation.
- Chief programmer team concept has never been validated:
  - ✓Some tasks best dealt with by assigning them to one very qualified programmer;
  - ✓Other complex tasks done by a team of equal programmers each contributing their specialty;
  - ✓Mundane tasks fit best the chief programmer team model.

Building Test Data, Test Files and Databases

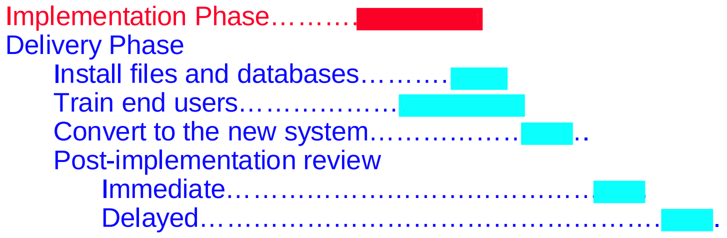
- Have users generate test data, if possible
- Generate full range of data, even non-key values
- Generate enough data to test size decisions of programs, e.g., have reports printed on more than one page
- Generate data that test the full range of potential values, e.g., generate the maximum and minimum input values allowed by a program

Note: This stage may include the conversion of an existing database

Programming and Testing

- Here is a top-down programming strategy:
  - ✓Review program structure
  - ✓Design user interface modules - prototyping
  - ✓Test user interface
  - ✓Design top modules
  - ✓Code and test top modules (stub testing)
  - ✓Design lower modules
  - ✓Code and test lower modules (stub testing)
- ***Stub testing*** -- test individual modules by simulating the interfaces to other modules
- ***Unit or program testing*** -- test all modules that have been coded
- ***System testing*** -- test the whole system

Delivery Phase



System Acceptance Testing

- ***Alpha Testing*** - testing on friendly users, often in-house
- ***Beta Testing*** - testing on less friendly users, usually an outside group who wants to use the system early
- ***End-user testing*** (or ***verification testing***) - test the system in a simulated environment to see whether it meets user specifications and usability requirements; often done during alpha testing
- ***Validation testing*** -- run the system in a live environment, testing system performance, peak workload performance, human engineering test, methods and procedures test, backup and recovery test, audit testing, i.e., is system free of errors

Post-Implementation Review

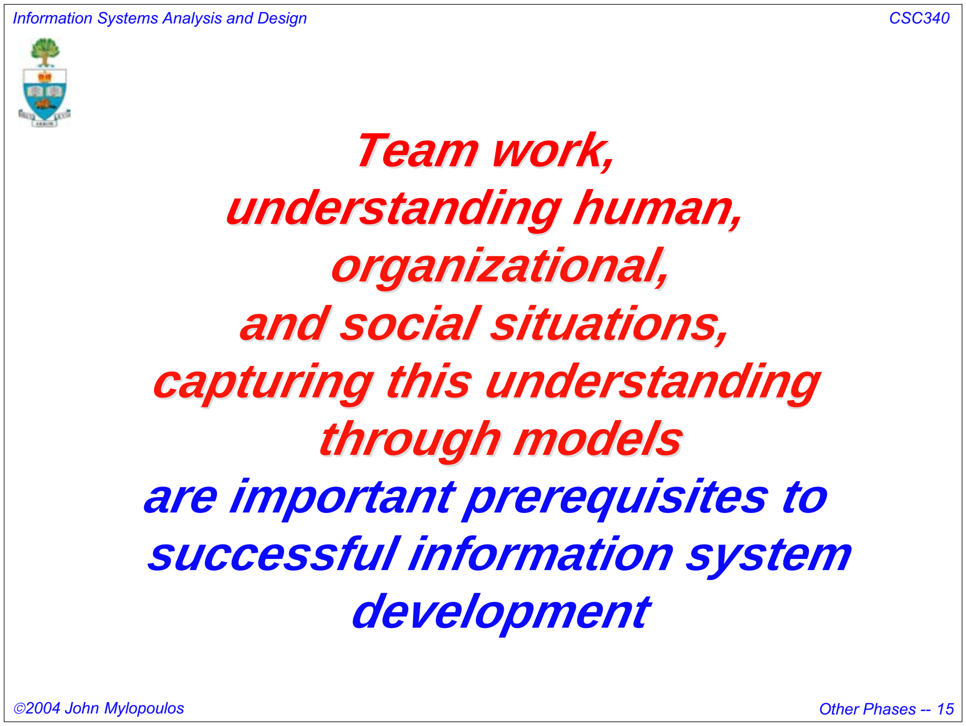
Involves two subtasks:

- ***Evaluate operational information system*** -- Does it fulfill the objectives set out? Does it adequately support transaction processing, management reporting...? Are the projected benefits being realized? How do end-users feel about the system? Should there be any enhancements? When should they be implemented? Are the internal controls working adequately?
- ***Evaluate development processes*** -- Did system costs match budgeted amounts? Was system completed on time? What was the performance of each individual on the project? What problems did we encounter? What would we do differently?

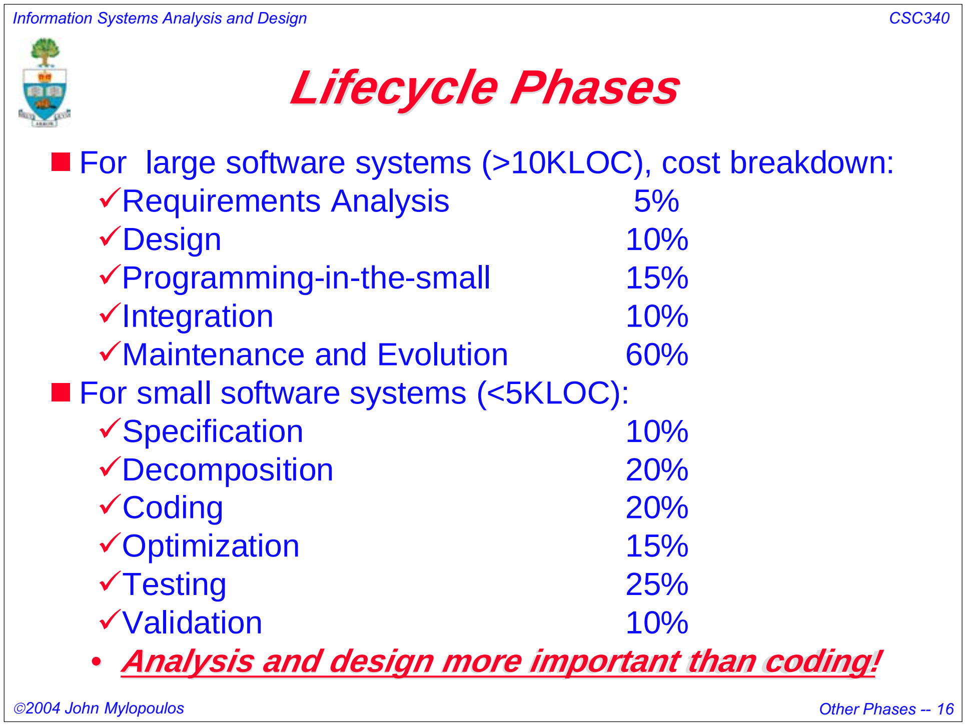
Course Summary

Building information systems involves three phases:

- ***Feasibility Study*** **--** understand the problem, come up with an information system solution.
- ***Requirements Analysis*** **--** specify functional and non-functional requirements of selected alternative.
- ***Design*** -- define the overall architecture, databases, interface and program structures.
- All three phases can be supported by modeling languages such as UML.



Team work, understanding human, organizational, and social situations, capturing this understanding through models are important prerequisites to successful information system development



Lifecycle Phases

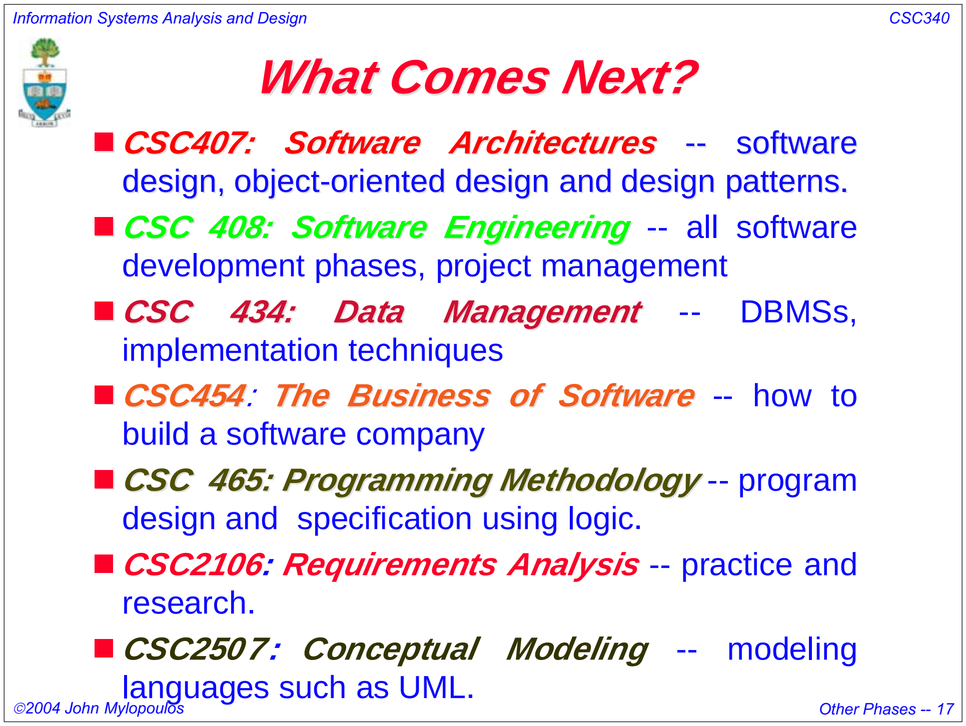
- For large software systems (>10KLOC), cost breakdown:

✓Requirements Analysis	5%
✓Design	10%
✓Programming-in-the-small	15%
✓Integration	10%
✓Maintenance and Evolution	60%

- For small software systems (<5KLOC):

✓Specification	10%
✓Decomposition	20%
✓Coding	20%
✓Optimization	15%
✓Testing	25%
✓Validation	10%

- ***Analysis and design more important than coding!***



What Comes Next?

- ***CSC407: Software Architectures*** -- software design, object-oriented design and design patterns.
- ***CSC 408: Software Engineering*** -- all software development phases, project management
- ***CSC 434: Data Management*** -- DBMSs, implementation techniques
- ***CSC454: The Business of Software*** -- how to build a software company
- ***CSC 465: Programming Methodology*** -- program design and specification using logic.
- ***CSC2106: Requirements Analysis*** -- practice and research.
- ***CSC2507: Conceptual Modeling*** -- modeling languages such as UML.