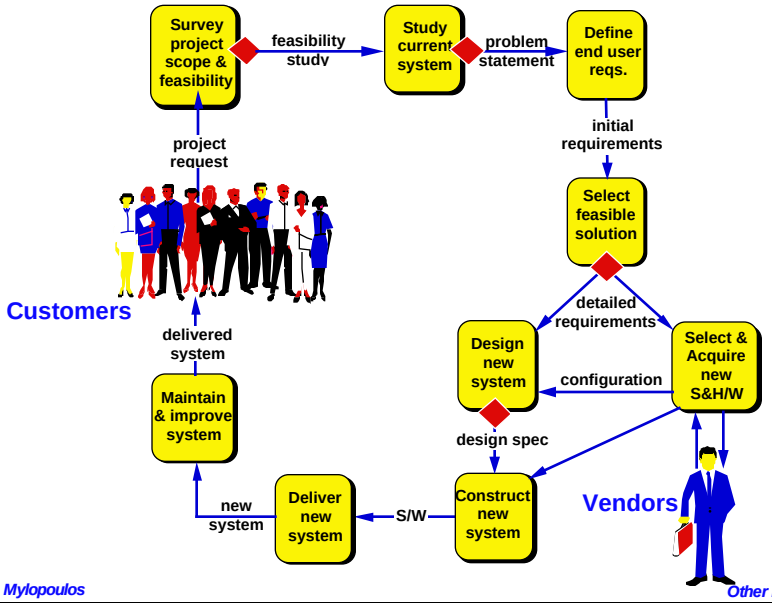
XXVI. Other Phases

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

The Software Lifecycle Revisited



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……

Installation of System Hardware and Software - Purchasing

- **Select specific software needed** -- database packages, fourth generation languages, compilers and loaders, project management tools, ...
- **Select specific hardware needed** -- additional disk storage, more powerful processors, workstations/terminals, cabling for networks, external 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 of hardware and software purchase decisions -- Datapro reports, A.D. Little reports,, Seybold reports, user groups, e.g., SHARE, DECUS, news publications, e.g., Computing Canada, Computerworld, ...

equipment must be ordered at this point if it is to arrive in a timely fashion

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
- **Chief programmer** -- is the person who conceives and directs the whole implementation.
- Note: Chief programmer team concept has never been validated; some tasks best done by one very qualified programmer; other complex tasks done by a team of equal programmers each contributing their specialty; mundane tasks fit best under chief programmer team concept
- **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

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

Writing and Testing Computer Programs

Here is a top-down programming strategy:

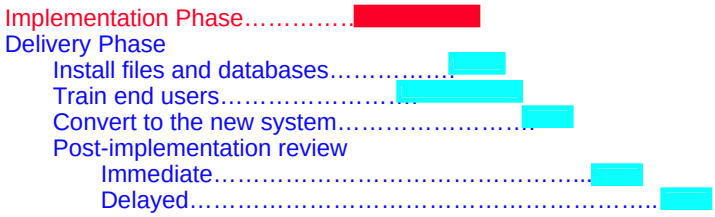
- Review program structure
- Design user interface modules - prototyping
- Test user interface
- Design top modules
- Code and test top modules (stub testing)
- Design lower modules
- Code and test lower modules (stub testing)
- Program Test -- test each individual program consisting of several modules (unit or program testing)
- System Test -- test all programs working together (system 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 Test

- **Alpha Testing** - testing system on friendly users, usually in-house
- **Beta Testing** - testing system on less than 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 the operational information system** -- Does it fulfill the goals and 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 the system 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

- **Information System Analysis** involves understanding a problem and coming up with an information system solution within an organization
- **Information System Design** involves defining the overall architecture, databases, interface and program structures of the information system to be built, given functional and non-functional requirements.
- Both analysis and design phases can be supported by modeling languages such as UML.
- **Team work**, **understanding human, organizational, and social situations** an important prerequisite to successful information system analysis and design

Relative Importance of Lifecycle Phases

- For large software systems, involving >10K lines of code (LOC), the breakdown of costs between different phases is as follows:

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

- The breakdown of costs per phase for small software systems (<5K LOC) has as follows:

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

Systems analysis and design more important than coding!

What Comes Next?

- **CSC407 -- Software Architectures** (a new course)
- **CSC 408 -- Software Engineering**, covers analysis, design, implementation etc., emphasis on project management
- **CSC 434 -- Data Management**, covers database management systems, implementation techniques for databases, data engineering
- **CSC454 -- The Business of Software**, gives an overview of the software industry, how to build your own software company
- **CSC 465 -- Programming Methodology**, covers program design and specification using logic.
- **CSC2106 -- Requirements Analysis**, focuses on requirements analysis, covers practice and research.
- **CSC2510 -- Conceptual Modelling**, covers in depth modeling languages such as UML.