Lecture Objectives

- At the completion of this lecture you should:
  - be aware of the tasks involved in the implementation phase of information system development
  - be aware of the responsibilities of the systems analyst, the client and the users in this phase
  - be able to develop a test plan for an information system and to perform testing according to that plan
  - be able to develop a suitable plan for conversion from an existing system to a new system
  - be aware of training and other user-oriented issues in installation of a new system

Lecture 11
Implementation Issues

Systems Implementation

<table>
<thead>
<tr>
<th>Task</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acceptance Checklist, Implementation Schedule, Training Schedule, Re-estimate</td>
<td>REVIEW</td>
</tr>
<tr>
<td>Training Guides, User Manuals</td>
<td>REVIEW</td>
</tr>
<tr>
<td>Test Data Preparation, System Test: Functional &amp; Performance, Test Conversion</td>
<td>REVIEW</td>
</tr>
<tr>
<td>Acceptance Test</td>
<td>REVIEW</td>
</tr>
<tr>
<td>Computer Documents, I/O Documents, Operating Guide</td>
<td>REVIEW</td>
</tr>
</tbody>
</table>

Testing

- Testing is ...
  - "the process of exercising or evaluating a system by manual or automatic means to verify that it satisfies specified requirements or to identify differences between expected and actual results."
    (IEEE, 1983)
  - "Anyone who believes that his or her program will run correctly the first time is either a fool, an optimist, or a novice programmer."
    (Anon.)

Principles of Testing

- Testing is the process of executing a program with the intention of finding errors
  - an attempt to 'break' the program
  - It is impossible to completely test any nontrivial module or any system
  - when do you stop testing ?

version: Jan 97 updated by H. Smith
Topic: Implementation

Software Errors

- Can arise for any of several reasons
  - the specification may be wrong
  - the specification may specify something that is physically impossible given the H/W and S/W
  - the system design may be at fault
  - the program design may be at fault
  - the program code may be wrong

Testing Steps

- All testing involves the following steps:
  - select what is to be measured by the test
  - decide how it is to be tested
  - develop the test cases
  - determine the expected or correct results (you must ensure that expected results can be measured - vagueness does not encourage adequate testing)
  - execute the test cases
  - compare actual results to expected results

Software Errors

- All testing involves the following steps:
  - select what is to be measured by the test
  - decide how it is to be tested
  - develop the test cases
  - determine the expected or correct results (you must ensure that expected results can be measured - vagueness does not encourage adequate testing)
  - execute the test cases
  - compare actual results to expected results

Testing Approaches

- Any software can be tested in two ways:
  - White box (or glass box) Knowing the internal workings of a module so that its logical structure and operations can be systematically tested.
  - Black box Knowing functions that the systems is supposed to perform and testing to see system to see if it performs the functions properly.

Module or Unit Testing

- Each module is tested individually
  - Lists what is being tested
  - Lists expected outcome
  - Identifies data to be used .. all possible combinations
- Who carries out Module Testing?
  - Programmer - tests at code level
  - Analyst - tests at application level
Integration Testing

- Verifies that the components of a system work together as described in the program design and system design specifications. It is necessary because:
  - data can be lost across interfaces
  - a function may not perform as expected when combined with another function
  - one module can have an adverse effect on another
- Integrating modules is best done using an incremental approach - easier to detect and correct errors.

Integration Testing

- There are a number of strategies that can be used to carry out integration testing:
  - Big-bang testing
  - Incremental Approaches:
    - Top-down testing
    - Bottom-up testing
    - Sandwich testing
- Any incremental integration testing needs a combination of stubs and drivers to work.

Using Stubs and Drivers

- Stubs and drivers link modules to enable them to run in an environment close to the real one of the future.

Stubs: take the place of modules that are called but have not yet been coded may be invoked or receive or transmit data to the test module as required.

Drivers: call the module under test and pass it test data.

Big Bang Testing

- Throw them all together at once
  - Advantages:
    > None - perceived to be faster
  - Disadvantages:
    > difficult to find and isolate the cause of any errors that appear
    > interface errors cannot easily be distinguished from other errors.

Incremental Approach to Testing

- **REPEAT UNTIL** the system is complete
  - Implement and unit test a module
  - Add the module to the existing combination
  - Test and debug the new combination
- **END REPEAT**
- Deliver the system
- Each time through the loop, the part of the system implemented will be working
  - crucial interfaces are not left till the end
  - resource usage is better distributed

Top Down Testing

- Implement the top module of a structure chart first
  - Each subordinate module is simulated by a stub or dummy module.
  - Each stub is replaced by a real module and the structure re-tested until the bottom level of the chart has been reached.
**Top Down Testing**

- **Advantages**
  - Feedback to users
  - Skeleton versions
  - Project less likely to be axed
  - Major system interfaces are tested
  - Testing resources are distributed more evenly
  - Implementers can see early results
  - If time is short, can begin other parts of the development cycle
  - Implementation can monitor progress - working modules vs. kilos of code
- **Disadvantages**
  - A large number of stubs may be required
  - Writing realistic lower level stubs may be difficult and time consuming, i.e., more costly

**Bottom Up Testing**

- **Advantages**
  - Project less likely to be axed
  - Testing resources are distributed more evenly
  - Implementers can see early results
  - Feedback to users (to some degree)
  - Driver modules are generally easier to develop than stubs...
- **Disadvantages**
  - No working program can be demonstrated until the last module is tested
  - Major top-level interfaces that may be critical are tested late
  - Cannot implement intermediate versions of the system

**Sandwich Testing**

- **Advantages**
  - A target layer is chosen based on the structure and characteristics of the module hierarchy
  - The target layer is usually the one just above all the general purpose utility modules
  - A top-down approach is used above the target layer
  - A bottom-up approach is used below the target layer
  - Testing converges on the target layer

**System Testing**

- **The process of testing the integrated software in the context of the total system it supports**
  - performed after all unit and integration testing is complete

**Who carries out System Testing?**

- systems analyst, systems implementers, technical support

**Tests conducted at this stage include**

- Function tests - demonstrate that all the functions specified for the system in the requirements specification are operational
- Performance tests - demonstrate that the system meets the non-functional requirements specified.

---

version: Jan 97 updated by H. Smith
## Implementation

### Function Testing

- **Performed after all programming and integration testing is finished**
  - Test cases
    - must cover every aspect of the system’s functionality
    - should have a high probability of detecting errors
  - Test plan
    - should be developed from the original specification
    - must include expected results that are measurable

### Performance Testing

- **Compares the integrated modules with the non-functional system requirements such as speed, performance**
  - Stress tests
  - Configuration tests
  - Regression tests
  - Timing tests
  - Quality tests
  - Maintenance tests
  - Human factors tests

### Acceptance Testing

- **Commences when the developers are confident that the system is ready to be used**
- **Is where the user decides if the system is ready for use**
- **Similar to system testing but politically very different**
- System testing can dispose of bugs while no one is watching
- Acceptance testing is done under a spotlight, with the user watching (when you wish you had done more and better system testing)

- **May be completely in user’s hands, but often shared between analyst and user**
- **Criteria for acceptance**
  - Is specification
    - presented to the user
    - signed by the user
  - Or
    - produce a definite plan for agreement on the criteria in the specification before you begin - must include results that can be measured

- **Involves installing the system at user sites and is required when acceptance testing has not been performed on site**
- The test focuses on completeness of the installed system and verification of any functional or nonfunctional characteristics that may be affected by site conditions
- **Testing is complete**
  - When the customer is satisfied with the results
  - The system can then be formally delivered

---

version: Jan 97 updated by H. Smith
Topic: Implementation

Implementing the System

• Other implementation tasks
  – implementation planning
  – finalise documentation
  – prepare the site
  – convert data into required form and media
  – conduct training
  – install system
  – monitor system
  – transition to maintenance mode
  – post-implementation review

Implementation Planning

• Implementation stage of the project
  – requires a great deal of co-ordination with professionals outside the development team
• Implementation plan
  – will have been developed at earlier stage of project
  – will need to be extended in greater detail
  – must be updated to reflect the current situation
• Poor planning can cause significant delays in deadline!
• Tasks
  – finalise acceptance checklist
  – complete and confirm training schedule
  – review and revise implementation plan

Finalise Documentation

• Documentation describes how a system works to a wide audience
• The four main areas are
• Training documentation
  – used specifically during the training sessions
  – especially designed to put the novice user at ease
• User documentation
  – tells users how to work with the system and perform their tasks
  – may be a user manual, on-line help, quick reference guide etc

Finalise Documentation

• System documentation
  – a communications tool and to review and revise the system during development
  – also facilitates maintenance and enhancement of the system
• Operations documentation
  – aimed at a centralised operations group (not on-line operators)
  – details what tasks an operator needs to carry out for a particular program

Prepare the Site

• Ensure that facilities are adequate
  – varies in complexity
  – may require new facilities or re-modelling of current facilities for first-time computer systems
  – consider issues such as
    > adequate space for all resources, ergonomic furniture, noise reduction, privacy, security, appropriate electrical connections, uninterrupted power, etc.

Prepare the Site

• Ensure that facilities are adequate
  – install the hardware and software required to run the system
    > usually done to a specification
    > must be tested to ensure no damage during transportation, product not defective, product changes between purchase and delivery are acceptable
  – People responsible
    > Vendor Engineer
    > Technical Support Group

version: Jan 97 updated by H. Smith
**Data Conversion**

- Current production data could be converted in 3 ways
  - Format, Content, Storage Medium
  - Done according to the conversion plan
  - Manual file conversion is a time-consuming task
  - Often needs specially written conversion programs e.g.
    - Database Load Program
    - Record Transformation Program
  - Data must be confirmed to be correct

**Data Conversion**

- May be simple or complex
  - Depends on system
- May need to support both files
  - Can introduce time lag
  - Files may be out of step
- General procedures involved
  - Prepare existing files ... no errors, up-to-date
  - Prepare manual files
  - Build new files and validate
  - Begin maintenance of new and old files
  - Work towards established cut-off date
  - Final check of accuracy

**Training**

- “If you think education is expensive and time-consuming - try ignorance.”
  - Bok, 1978

**Conduct Training**

- Need to consider:
  - Who is the audience?
  - What level of detail should be imparted to the audience?
  - Who should conduct the training?
  - Where should the training be conducted?
  - When should the training be conducted?

**Building User Understanding**

- Training - a complete and concentrated course in system use at the time of delivery
- Training must be planned
  - Methods
  - Resources
  - Should also consider help during and after installation for new users, infrequent users and users who want to “brush up”

**Building User Understanding**

- Training aids
  - Must be easy to use
  - Reliable
  - Demonstrations and classes
  - Documentation
  - On-line help and icons
  - Expert users
- Supportive User Manager who provides training, motivation, support

version: Jan 97 updated by H. Smith
Install the System

- Method of installation depends on several criteria
  - Cost - if there are cost constraints certain choices are not viable
  - System criticality - if system failure would be disastrous, the safest approach should be selected regardless of cost
  - User computer experience - the more experience the users have, the less necessary it is to delay changeover
  - System complexity - the more complex the system, the greater the chance of flaws ... a safer approach is better
  - User resistance - need to consider what the users are best able to cope with

Install the System

- Alternatives
  - Direct installation or Abrupt cut-over
  - Parallel installation
  - Phased installation or Staged installation
  - Pilot installation or Single Location conversion

Direct Installation (Abrupt Cutover)

- Old system stops and new system starts

  Old system

  Total cutover

  New system

Direct Installation (Abrupt Cutover)

- This approach is meaningful when
  - the system is not replacing any other system
  - the old system is judged absolutely without value
  - the old system is either very small and/or very simple
  - the new system is completely different from the old and comparisons would be meaningless

  Advantages
  - costs minimised
  - Disadvantages
  - high risk

Parallel Installation

- Old and new systems operated concurrently

  Old system

  Total cutover

  New system

Parallel Installation

- Old and new systems operated concurrently
- Cut-over at end of a business cycle
- Balancing between both systems
- Advantages
  - risks low if problems occur
- Disadvantages
  - cost of operating both systems 2.5 times the resources

version: Jan 97 updated by H. Smith
Phased Installation (Staged Installation)

- System installed in stages
- Total cutover

Old system

New system

Phased Installation (Staged Installation)

- System installed in stages
- Subsequent stages provide more features
- Phases or stages need to be identified at general design
- Advantages
  - lower costs for earlier results
  - benefits can be realised earlier
  - rate of change for users minimised

Pilot Installation

- Old and new systems operated concurrently
- Total cutover

Old system

Old system

New system

Pilot Installation

- Old and new systems operated concurrently
- Only part of the organisation tries out the new system
- The pilot system must prove itself at the test site
- Advantages
  - risks relatively low if problems occur
  - errors are localised
  - can be used to train users before implementation at their own site
- Disadvantages
  - lack of consistency between different parts of organisation

Monitor Operations

- Monitor user satisfaction
  > with functional requirements
  > with system performance
- Run benchmark tests
- Tune system
### Topic: Implementation

#### Transition to Maintenance

- Most organisations have formal procedures set up
- A "maintenance" section is responsible!
- Procedures should be set up to request maintenance
- Owners of the new system must be informed of relevant procedures

#### Post Implementation Review

- A PIR analyses what went right and wrong with a project. It is conducted 2 to 6 months after conversion by a team which includes user reps, development staff, internal auditors and sometimes external consultants - development team is not in charge!
  - look at original requirements and objectives
  - evaluate how well they were met
  - compare costs of development and operation against original estimates (maintenance costs ??)
  - compare original and actual benefits
  - new system reviewed to see whether more of original or additional benefits can be realised

### References


---

version: Jan 97 updated by H. Smith