Information System Provisioning

(a) IT Project Management (Development)

(b) Purchasing systems

Systems Provisioning

- we use the term “provisioning” because systems may be provided in a range of different ways - not just as new developments
- systems development remains a major focus but it is important to remember that development is just one alternative way to obtain a new system
- IS skills have been defined and developed in the context of new systems development, but note that:
  > systems development skills are relevant in all cases
  > systems analysis skills are particularly important

IS Provisioning

- systems provisioning refers to the set of processes used by IT managers in order to fulfill an organisation’s IT requirements
  - identification of an applications requirement
  - definition of the problem
  - selection of the appropriate approach to providing a solution
  - completion of the selected processes
  - installation of the solution as operational software

Provisioning Methods

- Systems Development
  > Traditional approach (formal SDLC)
  > Rapid systems development
  > RAD (JAD)
  > Prototyping
  > End user systems development
- Systems Purchase
  > Purchase of a packaged solution
  > ERP a special case
  > Purchase and customization
Traditional Developments

- The traditional approach was to build a system to meet specific user requirements.
  - Advantages included:
    - Installation of “purpose-built” solutions which were specific to organizational requirements.
    - Being able to utilize and retain organizational knowledge.
  - Risks with traditional developments: They can be expensive, time-consuming, and quite risky to build and install.

Systems Development

- "Software development remains an individual craft lacking in solid measurement systems, and based on weak theoretical foundations" does not ensure success.
  - "It is humanly impossible to control large projects" the more complex the project becomes, the less easy it is to keep to schedule/budget.
  - These comments are from practice-oriented guidelines and indicate the extent to which IT project management remains a discipline highly reliant on “gut feel” and experience.

Information Systems Integration

- A key question in IS is that of "integrated systems". An integrated systems structure works off:
  - Standard data structures
  - Standard data definitions
  - Standard process definitions
  - Standard business rules.

- Many theorists misunderstand the complexities involved and dramatically underestimate the complexity of these issues.

Benefits of Systems Integration

- Standardized work practices/processes:
  - Easier introduction of new staff to the organization.
  - Interchangeable staff.
  - Standard customer interface.
  - Simplified management processes.
  - Simplified change management.
  - Reduced system redundancy.
  - Economies of scale.

- Standardized data structures and definitions:
  - Reliable data.
  - Consistent management information.
  - Consistent reporting formats and processes.
  - "Single-point" system changes.
  - Improved communications.

Integrated Information Systems

- "Note that there are no 'disadvantages' to integrated functional systems. The amazing that more companies didn’t have the foresight to design integrated systems" (Mike Rewald - quoted Turban, McLean & Wetherbe p371).
Systems Integration

- there has been no really definitive research into systems integration, but a large company could have ~200 data entities and 5-6 major processes (plus many sub-processes) to consider when planning a systems integration activity.
- change is the biggest enemy of large-scale systems planning - organizational data and process requirements reflect changes in the business environment and are therefore not stable.

System Provisioning: a) IT Project Management (Development)

- good IT management consists in being aware of the critical issues and focusing on managing them well.

Methodologies

- what is a methodology?
  - an organized collection of methods and techniques comprising a way to deliver a required output.
- why is a methodology?
  - the provision of an expert “shell” to guide actions.
- does a methodology impact on the need for experience and/or specialised skills?

a Good Systems Development Methodology

- provides a checklist of required actions.
- identifies appropriate sequences of actions.
- is more like a set of heuristics (rules-of-thumb) than a set of laws.
- focuses on deliverables as much as processes (some new methodologies emphasise deliverables more than processes).

Systems Development

- during the life of a development project, managers should visit (revisit) the following issues at all critical review points (review points may be at the start or finish of methodology stages, or accompany the production of specific major deliverables).
  - economic justification.
  - resource allocation.
  - the adequacy of the quality control processes in place (quality reviews and walkthroughs).
  - risk assessment.
  - progress reviews.

Managing Systems Development

- the following is a brief overview of the pragmatics of managing systems development.
- NB:
  - managers have to deal with complex, partially abstract products.
  - reliable precedents to provide the basis for estimating are hard to find.
  - the problems stem from a form of “combinatorial explosion”.
  - the three easy variables of project management......
The Three Key Variables

- **functionality**
  - effectiveness of new system: what it can do

- **resources**
  - money, people, machinery expended in development

- **time**
  - hours spent

Managing the three variables

- ideally a project manager will retain a substantial degree of project freedom

- “fixing” time, resources and functionality is a management recipe for disaster

- effective project management requires the ability to allow at least one of the key variables to vary

Systems Development - Resources

- managing resources
  - software products including languages, development aids, DBMS
  - hardware including processing platforms and networks
  - money
  - people including developers, user analysts and support staff

- issues affecting the resources required
  - size of the application
  - complexity of the requirements
  - availability of IT skills and expertise
  - availability of user skills and expertise
  - novelty factors
  - scalability

Systems Development - Time

- managing time
  - usually significantly constrained in practice
  - the 95% complete syndrome has to be accommodated
  - 1 year is usually the maximum length of time for which reasonably reliable estimates can be made in practice (in many cases a lot shorter period)
  - despite that, IS managers must have forecasts which are as accurate as possible
  - milestones etc. must be set to pace progress

- when the timeframe doesn’t fit the time required
  - one view is that if the timeframe won’t shift, something else has to give
  - a contrasting view is that there is a general tendency to fit the work to the timeframe rather than the reverse
  - Parkinson’s Law: “work expands to fit the time available to do it”
  - Gersick’s Law: “there is a transition point where having lots of time becomes being short of time”
  - Human propensity for “muddling through” despite normal predictions
  - the only solution is high-quality estimating plus tight management
Systems Development - Functionality

- managing functionality
- functionality is the area of greatest uncertainty in any development project
  - functionality must be accurately documented and agreed
  - differences in understanding can surface quite late - one of the strengths of the formal SDLC is the emphasis on user agreement
  - the later the problems surface, the more expensive they are to fix ("band-aid" solutions are why there are problem areas in most completed applications)
  - reducing functionality is the basic IT management contingency strategy...

Some Notorious IS Problems

- controlling scope creep
  - documentation
  - estimating
  - risk management

"Scope Creep"

- the tendency for the scope to increase because
  - some requirements were overlooked during the early stages
  - new possibilities are revealed as the project progresses and the role of technology becomes clearer
  - the processes of analysis and design help users to "frame" the problem better
  - problems of mutual understanding surface and have to be resolved

"Documentation"

- a lack of adequate documentation is a frequent issue with information systems
  - the major problems with documentation are
    - maintaining its accuracy and currency when system changes are being made is difficult (unlike other engineering disciplines, many changes are made directly to the system rather than first documented)
    - much documentation is meaningful only to the original system developers, and as staff changes occur the motivation to update the documentation weakens
    - time pressure

Documentation

- managers must balance the requirement for documentation against the likelihood it will be maintained
  - documentation should assist the process not hinder it
  - the documentation of agreed functionality is the fundamental requirement
  - ambiguity is inevitable in the wording of requirements
  - most disagreements on IT projects are about what the requirements actually mean

Estimating

- perhaps the most problematic area in IT management
  - experience is vital but relevant experience takes time to acquire
  - estimates need to be reviewed and revised as frequently as possible: the key to reliable estimates is constant revision
  - estimates should be prepared on a staged basis (ie first up to system requirements, then for delivery)
  - the relationship between people and progress is not linear

  > the "mythical man-month"

- formal estimating methods
  - LOC estimating
  - function point estimating (ASMA)
  - defect identification and correction rates

- "defensive" (contingency-oriented) estimating is good practice...
Risk Management

- managing risk
  - the objective is to understand and manage the key risk factors
  - progress is measured according to whether the total level of risk is falling
- major risk areas
  - clients
  - management skills
  - technical skills
  - application characteristics
  - hardware requirements
  - system software requirements
  - project importance

Managing Systems Development

- IS project management remains an art rather than a science
  - automated aids
    - can be very useful for keeping track of progress
    - can assist with estimates and timetabling
    - are not a substitute for common-sense and attention to detail
    >ie many automatically generated timetables and project charts have a very bad signal-to-noise ratio
    - “sanity” checks are critical

System Provisioning

(b) Purchasing systems

The Argument for Purchase:

- IT is not a "core" organizational competence
  - which means that?
- if an organization is not an IT company, why should it build or even manage its own applications?
- but even when purchasing software, it is arguable that full-scale systems analysis should be conducted within the purchasing organization anyway

Purchasing Systems

- we are living in an increasingly automated and systematized world
  - this is even more true of the business world than the world in general
  - virtually all business processes are highly automated
  - relatively standard "solutions" now exist for many business processes including
    - general ledger
    - financial accounting
    - personnel systems
    - materials supply and inventory management
    - telemarketing

Purchase versus Build?

- in simple terms, the question is whether to change the system or the organization
  - BUT using systems to drive major organizational changes is a risky proposition

Managing Vendor Interactions

- criteria for evaluating a candidate software package include those relating to the vendor
  - reputation
  - organizational size and market presence
  - support
  - expertise
  - comparison sites
  - guarantees
  - upgrades
  - customisation support policy
### Customization

- this is where a company buys a packaged solution but wishes to make changes & retain some of the organization’s own “flavor”
  - customization requires the establishment of a systems development project
  - >10% change is usually high-risk and very difficult to manage in practice
  - the calculation is difficult because it is usually very difficult to tell what else will be affected after you start making changes
  - scope-creep is very important here because of the tendency for users to prefer their own way of doing things

### Controlling Customization

- It is extremely hard to control the effects of customization, irrespective of scope creep
- It is difficult to see the ramifications of making a change to the visible outputs of a package
- for instance, changes to data items impact on database design, and lead to radical changes in data structures and definitions (e.g. the customer file/entity)
- once work on customization has commenced, developers frequently find that one thing is connected to another (hip-bone to the hip) and that change begets change
- changes amounting to more than 100% of the original code have been recorded

### Risks of Customization

- many of the greatest benefits of buying packages are lost
  - support arrangements
  - package-related technology upgrades
  - software enhancements
  - reliability and performance guarantees