Chapter 4: Selection of an appropriate project approach

NET481: Project Management

Afnan Albahli
Outline of lecture

- Building OR buying software
- Taking account of the characteristics of the project
- Process models
  - Waterfall
  - Prototyping and iterative approaches
  - Incremental delivery
- Agile approaches
This lecture concerned with choosing the right approach to a particular project: variously called technical planning, project analysis, methods engineering and methods tailoring.

In-house: means that the developers and the users of the software are in the same organization.
- often the methods to be used dictated by organizational standards

Suppliers: means that the developers and the users of the software are in the different organization.
- need for tailoring as different customers have different needs
Build or buy?

- In-house development?
- Outsource?

Build or Buy
In-House

Developing a new IT application in-house:

- Time is needed to develop the software
- Would often require the recruitment of new technical staff to do the job
- Usually, the new staff won’t be needed after the project is completed
- Sometimes due to the novelty of the project there may be lack of executives to lead the effort
Contracting the project out to an external IT development company (outsourcing):

- Time is needed to develop the software
- The conducting company will have technical and project expertise not readily available to the client
- The client would still do management effort to establish and manage the contracts
Some advantages of off-the-shelf (OTS) software

- Cheaper as supplier can spread development costs over a large number of customers
- Software already exists
  - Can be trialled by potential customer
  - No delay while software being developed
- Where there have been existing users, bugs are likely to have been found and eradicated
Some possible disadvantages of off-the-shelf

- Customer will have same application as everyone else: no competitive advantage, but competitive advantage may come from the way application is used

- Customer may need to change the way they work in order to fit in with OTS application

- Customer does not own the code and cannot change it

- Danger of over-reliance on a single supplier
Steps of Project Analysis

- Identify project as either objective driven or product driven.

- Analyze other project characteristics by asking:
  - Will we implement a data-oriented or a process oriented system?
  - Will the software to be produced be a general tool or application specific?
  - Are there specific tools available for implementing the particular type of application?
    - E.g.: – does it involve concurrent processing?
    - Is the system knowledge-based?
    - Will the system to be produced makes heavy use of computer graphics?
Steps of Project Analysis (cont’d)

- Is the system to be created safety critical?
- Is the system designed to carry out predefined services or to be engaging and entertaining?
- What is the nature of the hardware/software environment in which the system will operate?
Steps of Project Analysis (cont’d)

- Identify high-level project risks.
- The more uncertainty in the project the more the risk that the project will be unsuccessful.
- Recognizing the area of uncertainty allows taking steps towards reducing its uncertainty.
- Uncertainty can be associated with the products, processes, or resources of a project.
Steps of Project Analysis (cont’d)

- **Product uncertainty:**
  - How well are the requirements understood.
  - The users themselves could be uncertain about what the system is to do.

- **Process uncertainty:**
  - For the project under consideration, the organization will use an approach or an application building-tool that it never used before.

- **Resource uncertainty:**
  - The main area of resource uncertainty is the availability of the staff with the right ability and experience.
General approach

- Look at risks and uncertainties e.g.
  - are requirements well understood?
  - are technologies to be used well understood?

- Look at the type of application being built e.g.
  - information system? embedded system?
  - criticality? differences between target and development environments?

- Clients’ own requirements
  - need to use a particular method
Structure versus speed of delivery

Structured approach

- Also called ‘heavyweight’ approaches
- Step-by-step methods where each step and intermediate product is carefully defined
- Emphasis on getting quality right first time
- Example: use of UML (Unified Modelling Language)
- Future vision: Model-Driven Architecture (MDA). UML supplemented with Object Constraint Language, press the button and application code generated from the UML/OCL model
Structure versus speed of delivery

Agile methods

- Emphasis on speed of delivery rather than documentation
- RAD Rapid application development emphasized use of quickly developed prototypes
- JAD Joint application development. Requirements are identified and agreed in intensive workshops with users
Software Process Models

- Waterfall Model.
- V-process Model.
- Spiral Model.
- Software prototyping.
- Phased Development Model.
  - incremental development model.
  - iterative development model.
Waterfall

1. Feasibility study
2. User requirements
3. Analysis
4. System design
5. Program design
6. Coding
7. Testing
8. Operation
Waterfall

- Classical model of system development.
- Called one-shot or once-through model.
- Limited scope of iteration. Is this a strength or a limitation??
  - This is a strength for the WF-model.
  - Because it is suitable for some projects especially for large projects, we want to avoid reworking tasks that are thought to be completed.
  - Reworking tasks could result in late delivery.
- Suitable for systems with well defined requirements.
- Not suitable for systems of high uncertainty.
V-process Model

- An extension of the waterfall model.
- V-process model expands the activity box “testing” in the waterfall model.
- Each step has a matching validation process.
- Validation process can cause a Loop back to the corresponding stage and reworking the following steps in case of discrepancy.
V-process Model (cont’d)

- learning by doing
- improved communication
- improved user involvement
- a feedback loop is established
- reduces the need for documentation
- reduces maintenance costs i.e. changes after the application goes live
- prototype can be used for producing expected results
A greater level of detail is considered at each stage of the project.

Represented as a loop or a spiral where the system is considered in more detail.

This means greater confidence about the probability of success.

Each sweep is terminated by an evaluation before the next iteration is embarked upon.
Spiral Model (cont’d)
Prototype is a working model of one or more aspects of the projected system.

Goal

- Gain knowledge
- reduce risk and uncertainty
- verify a design or implementation approach

The prototype is constructed and tested, quickly and inexpensively to test assumptions.
Classification of a Prototype

- **Throw-away**
  - Tests out some ideas.
  - Discarded when the true development of the operational system is started.
  - The prototype could be developed using a different SW and HW environment than those that will be used for the final system.

- **Example: user interface**
  - **Prototype**: use a desktop application builder to produce an acceptable user interface.
  - **Final system**: use a procedural programming language.
Prototyping Model

1. Build prototype
2. User feedback
3. User satisfaction
   - YES
   - NO

Flowchart: Build prototype → User feedback → User satisfaction → YES/NO
Benefits of Prototyping

- Learning by doing.
- Improved communication.
- Improved user involvement.
- Clarification of partially-known requirements.
- Demonstration of the consistency and completeness of a specification
Benefits of Prototyping (cont’d)

- Reduced need for documentation.
- Reduced maintenance costs.
- Feature constraint.
Drawbacks of Prototyping

- Users sometimes misunderstand the role of the prototype.
- Lack of project standards possible.
- Lack of control.
- Additional expense.
- Machine efficiency.
- Close proximity of developers.
Different projects will have uncertainties at different stages.

Thus, prototypes can be used at different stages.

Examples:

At the requirements gathering stage: to pin down requirements that seem blurred and shifting.

At the design stage: to test out the user's ability to navigate through a sequence of input screens.
To what extent is the prototyping done?

- The prototyping usually simulates only some aspects of the target application, thus there might be:
  - **Mock-ups**
    - e.g. Copies of input screens shown to the users on a terminal.
    - They cant actually be used.
  - **Simulated interaction**
    - A user can type in a request to see a record in a database and an example of a result is shown.
    - There is no real access is made to the database.
Forms of Prototypes

- **Partial working model**
  - **Vertical:** only some features are fully prototyped
  - **Horizontal:** all features are prototyped but not in detail.
Incremental Model

- Break the system into small components.
- Implement and deliver small components in sequence.
- Every delivered component provides extra functionality to the user.
Incremental Model (cont’d)

- Requirements Analysis
- Arrange requirements in increments
- Design and develop increment
  - Validate increment
  - Integrate increment
  - System OK?
    - YES
    - NO
      - Design and develop increment

(The McGraw-Hill Companies, 2009)
Iterative Model

- Deliver full system in the beginning.
- Enhance existing functionality in new releases.
Iterative Model

- Design system version n
- Develop system version n
- Validate system version n

If System complete is NO, then:
- n = n+1
- Go back to Design system version n

If System complete is YES, then:
- System complete
Combined Incremental and Iterative Model

- Every new release includes:
  - extra functionality.
  - enhancement of existing functionality.

Popularly used in software industry.