Coping with Change and Risk

Chapter 2.3 & 2.4

CMPT 276
© Dr. B. Fraser
Based on slides from Software Engineering 9th ed, Sommerville.
Topics

• How can software projects manage change?
  – What is prototyping?
  – What is incremental development?
Coping with change

- Change is inevitable in all large software projects:
  - Business changes lead to new (or changed) system requirements.
  - open up new possibilities.

- Cost of change = Cost of reworking completed work (re-analyzing requirements, design, recoding) + Cost of..
Reducing the cost of rework

- **Change avoidance:**
  - software development process includes.. before significant rework is required.
  - **Example:** develop a prototype system to show a key (uncertain?) features to customers.

- **Change tolerance:**
  - software development process is designed to..
    - Usually **incremental development**.
    - Changes may be in a **future increment** (no rework), or may have to **alter part of existing system**.
Change avoidance with
(Throwaway)
Software Prototyping
Throwaway Software Prototyping

- **Prototype:**
  - used to try out options.

- **"Throw-away" code:**
  - Prototypes could ignore things like code quality, error-handling, or testability.
  - Built to **answer a specific question**, not to see if the whole system will work.
Software prototyping

- A prototype can be used in:
  - to help with requirements elicitation and validation;
  - to explore options;
  - For example, a paper prototype of the UI.
Benefits of Prototyping:
- Improved **system usability**.
- A closer match to users’ **real needs**.
- Improved **design quality**.
- Improved **maintainability**.
- Reduced development effort.
Prototype development

- Focus on **poorly understood** areas of the product;
- **Error checking** and **recovery** may be omitted;
- Focus on **rather than** requirements.

- **Ex**: Accessing hardware, screen layouts, database access.
- **Ex**: Security, performance, etc.

- **Prototypes** are not a good basis for a production system:
  - Likely below software **quality** standards.
  - Normally **undocumented**;
  - **Degraded structure** from rapid change (no refactoring);
  - Hard to tune it to meet **non-functional requirements**.
Change tolerance with Incremental Delivery
Incremental delivery

- **Development and delivery are**
  - Each increment delivers **some required functionality**.

- **Prioritized user requirements**
  - highest priority ones included in early increments.

- **Requirement changes**
  - Once the development of an increment is started,
    - Requirements for later increments continue to evolve.
Incremental development and delivery

- **Incremental development**
  - Develop the system in **increments**.
  - increment before proceeding to development of next increment;
  - Normal approach used in..

- **Incremental delivery**
  - Deploy an increment for..
  - More realistic evaluation because of..
  - Difficult to implement for replacement systems as increments have **less functionality than old system**.
The increment could fit into a larger system plan (BDUF), or be developed on the fly with evolutionary planning (Agile).
Incremental delivery advantages

- **Benefits Include:**
  - New functionality delivered with each increment so system ***functionality is available earlier***.
  - Early increments act.. to help elicit requirements for later increments.
  - Lower risk of overall project failure.
  - Highest priority requirements implemented first and..
Incremental delivery problems

• **Common Functionality:**
  – Most systems require a set of basic facilities that are used by different parts of the system.
  – Hard to identify common facilities because requirements are not defined in detail until..

• **Contracts:**
  – Specification developed iteratively with the software.
  – Complete system specification can be needed as part of the..
Summary

• Processes should cope with change.
  - Change avoidance:
    • Throwaway prototyping helps avoid poor decisions on requirements and design.
  - Change tolerance:
    • Iterative development and delivery allows changes without disrupting whole system.