System Analysis and Design

L02. Iterative and Evolutionary Development

Topics

- Iterative and Evolutionary Development
- Unified Process

The Waterfall Model

- In a **waterfall** (or sequential) lifecycle process there is an attempt
  - to define (in detail) all or most of the requirements before programming.
  - And often, to create a thorough design (or set of smodels) before programming.
  - Likewise, an attempt to define a "reliable" plan or schedule near the start - not that it will be.

Drawbacks of the Waterfall Model

- The reality is that not only does software change, but change happens *during* the process 改变
  - Realistic models are not strictly linear, but allow for cycles
  - Bear in mind, however, that more cycles mean more costs
- Offers no insight into how does each activity transform one artifacts (documents) of one stage into another 没有考虑文档的转换.
  - For example, requirements specification → design documents?
- Fails to treat software development a problem-solving process
  - Software development is not a manufacturing but a creative process
  - Manufacturing processes really can be linear sequences, but creative processes usually involve back-and-forth activities such as revisions
  - Software development involves a lot of communication between various human stakeholders
- Nevertheless, more complex models often embellish the waterfall
  - incorporating feedback loops and additional activities

Phased Development

- Nowadays, customers are less willing to wait years for a software system to be ready
  - So it's necessary to reduce the **cycle time** of software products
  - In 1996, 80% of HP's revenues derived from products developed in previous two years
  - *How is this accelerated cycle time made possible?*
- Phased development reduces cycle time
  - Design a system so it can be delivered in pieces, letting users have some functionality while the rest is under development
- So there are usually two or more systems in parallel:
  - The **operational** or **production** system in use by customers
  - The **development** system which will replace the current release
  - As users use Release *n*, developers are building Release *n* + 1

Iterative and incremental process

- **Incremental development** partitions a system by functionality
  - Early release starts with small, functional subsystem, later releases add functionality up to full functionality 功能递增.
- **Iterative development** improves overall system in each release
  - Delivers a full system in the first release, then changes the functionality of each subsystem with each new release 性能递增
- Suppose a customer wants to develop a word processing package
  - **Incremental approach**: provide just Creation functions in Release 1, then both Creation and Organization in Release 2, finally add Formatting in Release 3, …
  - **Iterative approach**: provide primitive forms of all three functions in Release 1, then enhance (making them faster, improving the interface, etc.) in subsequent releases
  - Pros and cons of these two approaches?
- Many organizations combine iterative and incremental approaches

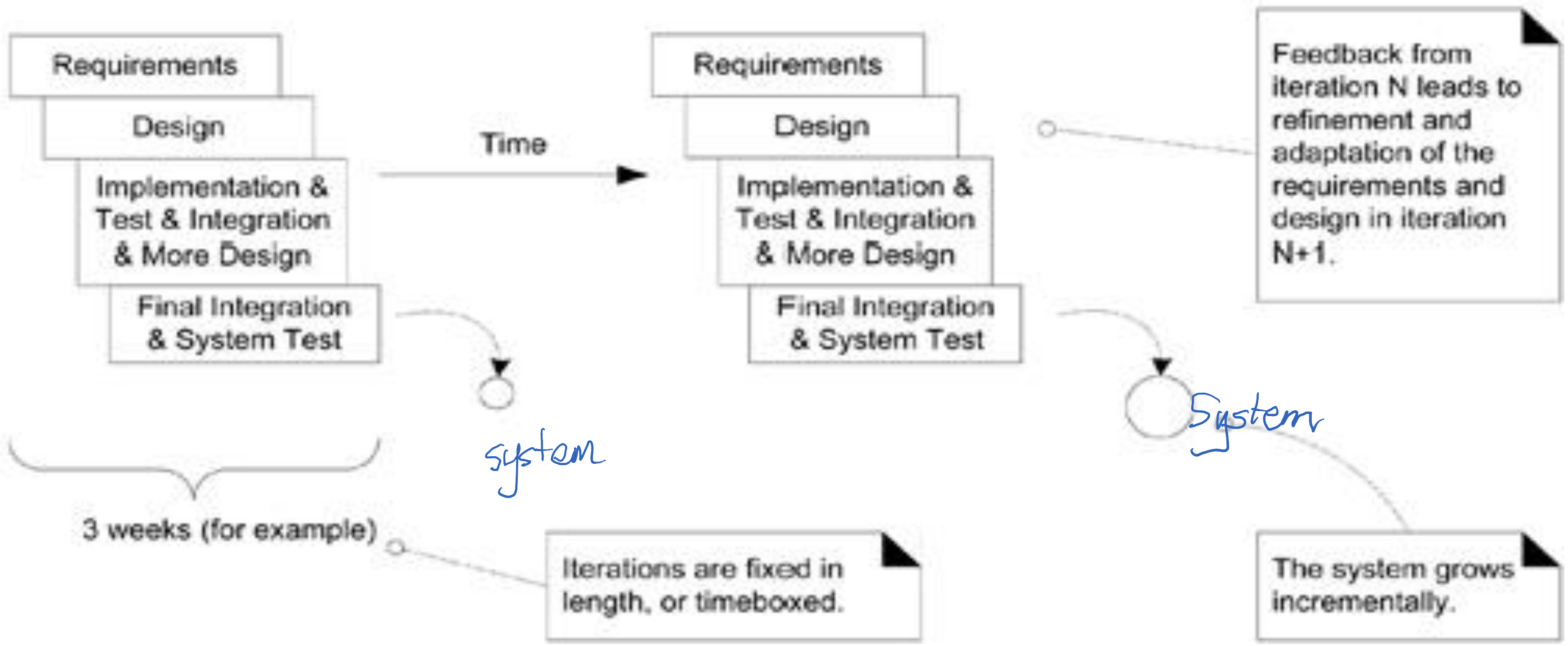
两种方法的结合

Iterative and Evolutionary Development

- It is an iterative and incremental development 是迭代和递增式开发.
- Lifecycle involves early programming and testing of a partial system, in repeating cycles.
  - Development is in **short cycles, or iterations** 迭代周期
  - Each one is **tested and integrated**
  - Each one gives **an executable partial system**
- Feedback from each iteration leads to **refinement and adaptation** of the next.
  细化和修正
- Normally assumes development starts before all the requirements are defined in detail

- An example of such a process is the **unified process** (UP). 统一过程

Iterative and Evolutionary Development


Requirements
Design
Implementation & Test & Integration & More Design
Final Integration & System Test
Time
Requirements
Design
Implementation & Test & Integration & More Design
Final Integration & System Test
Feedback from iteration N leads to refinement and adaptation of the requirements and design in iteration N+1.
system
System
3 weeks (for example)
Iterations are fixed in length, or timeboxed.
The system grows incrementally.

Handling Change in Iterative and Evolutionary Development

应对变化的基本思想.

- "Change is good. Hard cash is better." John Cole
- Don't fight changes to the software
- Let users guide the development
- They cannot tell you what they do not know

Handling Change in Iterative and Evolutionary Development

- Each iteration involves choosing a **small subset** of the requirements, and **quickly** designing, implementing, and testing.
- In early iterations the choice of requirements and design may not be exactly what is ultimately desired.

初期的选择
可能不是最终
希望的

- But the act of swiftly taking a small step, before all requirements are finalized, or the entire design is speculatively defined, **leads to rapid feedback** from the users, developers, and tests (such as load and usability tests).

Handling Change in Iterative and Evolutionary Development

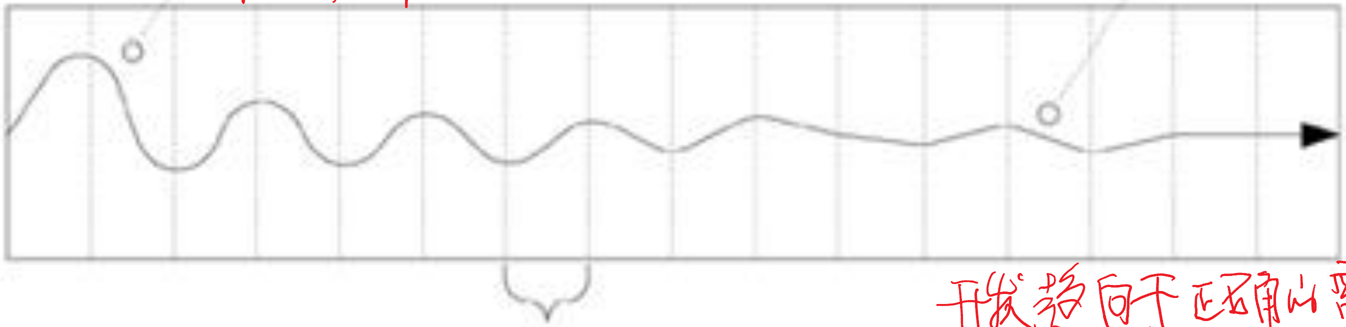
- In addition to requirements clarification, activities such as load testing will prove
  - if the partial design and implementation are on the right path, or
  - if in the next iteration, a change in the core architecture is required.
- Better to resolve and *prove* the risky and critical design decisions **early** rather than late
- And iterative development provides the mechanism for this.

Iterative Feedback and Evolution

Early iterations are farther from the "true path" of the system. Via feedback and adaptation, the system converges towards the most appropriate requirements and design.

开始可能偏离正确的路径，

In late iterations, a significant change in requirements is rare, but can occur. Such late changes may give an organization a competitive business advantage.



开发趋向于正确的需求与设计.

one iteration of design, implement, integrate, and test

一个迭代

Benefits of Iterative and Evolutionary Development

- Fewer failures, lower defect rates 瀑布开发导致80%以上失败
- Early mitigation of high risks 减少高风险
- Early visible progress
- Early feedback, user engagement, and adaptation
- Managed complexity: team sees small pieces at a time 可控的复杂度，
- Use of learning within an iteration to improve the development process itself

Summary

- Why Iterative and incremental Development?
- Iterative and evolutionary development Process
- Handling Change in Iterative and Evolutionary Development
- Benefits of Iterative Development