Foundations of Software Engineering

Process: Linear to Iterative

Michael Hilton
Learning goals

• Understand the need for process considerations
• Select a process suitable for a given project
• Address project and engineering risks through iteration
• Ensure process quality.
Administrivia

• HW4 due today
Interview

• Sean McDonald
The Waterfall Model
Win Royce and Barry Boehm, 1970

**Why was this an important step?**

**What are limitations?**
Key challenge: Change

• Software seems changeable ("soft")
• Developers prone to changes and "extra features"
• Customers often do not understand what is easy to change and what is hard
• "Good enough" vs. "optimal"
The "V" Model (80s, 90s)
When is waterfall appropriate?

1. The requirements are known in advance.
2. The requirements have no unresolved, high-risk risks such as due to cost, schedule, performance, safety, security, user interfaces, organizational impacts, etc.
3. The nature of the requirements will not change very much.
4. The requirements are compatible with all the key system stakeholders’ expectations.
5. The architecture for implementing the requirements is well understood.
6. There is enough time to proceed sequentially.
Early improvement: sequencing

• Enforce earlier software considerations
• Waterfall instituted at TRW (Aerospace Govt Contractor) in 70s, with several additional recommendations for iterations (like prototypes).
• Modeled after traditional engineering
  – blueprints before construction
  – decide what to build, build it, test it, deploy
  – Reduce change
• Successful model for routine development
• Problematic at large scale
  – Requirements -> Delays -> Surprise!
A natural engineering process?

• Decide what to build
• Build it
• Test it
• Deploy it

• Don't know what to build in advance
• Don't know all details how to build
• Struggling with testing and evaluation
• Deploy, evolve, redeploy

- Early and frequent feedback
- Support for constant adaptation
Iteration!

- Early and frequent feedback
- Support for constant adaptation
- Address risks first
Software Engineering *Risks*

• Project risks
  – Projects late, buggy, cost overruns

• System risks
  – Security and safety issues
  – e.g. Toyota case

• Engineering risks
  – Unsuitable technology choices, validation issues, usability issues, scalability issues...
Cone of Uncertainty

Mitigation of risk through process interventions (examples)

• Risk-driven process
  – Prioritization and prototyping

• Architecture and design
  – Isolate/encapsulate risks
  – Follow industry standards

• Design for assurance
  – Preventive engineering
  – Codevelopment of system and evidence

• Functionality and usability
  – Prototypes, early usability labs
The Role of Architecture

Source: Boehm, Valerdi, Honour, The ROI of Systems Engineering. 2008
Key: Iterative Processes

• Interleaving and repeating
  – Requirements engineering, Risk assessment
  – Architecture and design
  – Implementation
  – Quality assurance
  – Deployment

• But when, in which sequence, and how often?

• What measurements can ground decisions?
Iteration in Project Management

- Identify constraints
  - Budget, Personal, Deadlines
- Estimate project parameters
- Define milestones
- Create schedule

Activities begin

- Check progress
  - Done?
    - yes
    - no
      - Reestimate project parameter
        - Refine schedule
          - Problem?
            - yes
            - Technical review
              - new feature requests
            - no
              - renegotiate constraints
                - renegotiate constraints
                  - Abort?
The Spiral Model (Barry Boehm)

Drive from **engineering risks:**
- Requirements
- Design
- Implementation
OODA Loop

Unfolding Circumstances

Outside Information

Unfolding Interaction With Environment

Observations

Observe

Feed Forward

Implicit Guidance & Control

Orient

Decision (Hypothesis)

Feed Forward

Feed Forward

Feedback

Feedback

Feedback

Cultural Traditions

Genetic Heritage

Analysis & Synthesis

New Information

Previous Experiences

Act

Action (Test)

Implicit Guidance & Control

Unfolding Interaction With Environment

John Boyd's OODA Loop

cc (3.0) Moran
Iteration decision

• Too slow?

• Too fast?

• -> Drive by risks and measurement data; per project decision

• Contracts?
Iteration decision

• Too slow?
  – Late reaction, reduce predictability
• Too fast?
  – Overhead, reduce innovation
• "Death spiral"
  – Deferred commitment, prototypes without conclusions, missing feedback loops
• -> Drive by risks and measurement data; per project decision
• Contracts?
Rational Unified Process (UP)
(more on Agile, XP, Scrum, Kanban in a later lecture...)
Iterative vs. Incremental?
Change Control
Change Control Board
Change Request Form

Project: SICSA/AppProcessing             Number: 23/02
Change requester: I. Sommerville        Date: 20/01/09
Requested change: The status of applicants (rejected, accepted, etc.) should be shown visually in the displayed list of applicants.

Change analyzer: R. Looek               Analysis date: 25/01/09
Components affected: ApplicantListDisplay, StatusUpdater

Associated components: StudentDatabase

Change assessment: Relatively simple to implement by changing the display color according to status. A table must be added to relate status to colors. No changes to associated components are required.

Change priority: Medium
Change implementation:
Estimated effort: 2 hours
Date to SGA app. team: 28/01/09          CCB decision date: 30/01/09
Decision: Accept change. Change to be implemented in Release 1.2
Change implementor: Date of change:
Date submitted to QA: QA decision:
Date submitted to CM: Comments:
Change Impact Analysis

• Estimate effort of a change
• Analyze requirements, architecture, and code dependencies
• Tractability very valuable if available
• Various tools exist, e.g., IDE call graphs
Feature Freeze

• Pre-release phase
• Do not allow any changes except bug fixes
• Avoid destabilization
Release Planning with Branches

Development

End of Release 1 development

New feature 1 (for Release 2)

End of Release 2 development

New feature 2 (for Release 2)

New feature 3 (for Release 3)

Release 1

Release 2

QA passes - goes alpha

Public release

Bug fix

QA passes - goes alpha

Public release

Bug fix

Bug fix

Project milestone

End of branch

Create branch/merge changes
Case Study: Microsoft

• Microsoft plans software in features.
• 3-4 milestones per release.
• After each milestone, reconsider which features should be implemented.
• Stabilization and freeze at end of milestone.

Cusumano and Selby. Microsoft Secrets.
Prepare servicing strategy for Windows 10 updates

07/26/2017 • 6 minutes to read • Contributors
Microsoft slows down Windows 10 update pace for businesses following complaints

Relief for IT admins

By Tom Warren | @tomwarren | Sep 6, 2018, 11:00am EDT
How much iteration? How much change control? (3 cases)
Discussion: what is the purpose of tracking process?
Burn Down Charts

Sample Burndown Chart

- Remaining effort
- Ideal burndown
- Remaining tasks
- Completed tasks

Remaining effort (hours)

Remaining and completed tasks

Day

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21

0 5 10 15 20 25

0 5 10 15 20 25

Completed tasks

Remaining effort

Ideal burndown

Remaining tasks

Institute for Software Research
Milestone Trend Analysis

- Quickly rising?
  - estimations too optimistic
- Changing trends?
  - unreliable early estimations
- Ziz-zag pattern?
  - unreliable estimations
- Falling?
  - overly large buffers
Process metrics: Quality

• Bugs reported?
• Bugs fixed?
• Evidence of completed QA activities
  – "Test coverage", inspection completed, usability study, ...
• Performance analysis?
Goodhart's law

"When a measure becomes a target, it ceases to be a good measure."
Discussion: what makes a good process?
Process evaluation

• How predictable are our projects?

• 33% of organizations collect productivity and efficiency data

• 8% collect quality data

• 60% do not monitor their processes
Process improvement loop

High-level approaches:
• Opportunistic, based on double-loop learning.
• Analytic, based on measurement + principles
• Best practices frameworks

acting

training and enforcement

analyzing difference

monitoring
Defect Prevention Process, IBM 1985

- When a mishap occurs:
  1. Take corrective action
  2. Conduct root cause analysis (Root cause(s): Management, people, process, equipment, material, environment):
     • Why did the mishap occur? Why was it not detected earlier?
     • Is there a trend indicating a broader problem? Can we address it?
     • What went right during this last stage? What went wrong?
  3. Implement preventive actions within the team context

- Successful changes are percolate up to corporate level.
Google SRE Process

- blame-free postmortem culture
- “Unless we have some formalized process of learning from these incidents in place, they may recur ad infinitum.”

Six Sigma, Motorola 1985

“Six Sigma seeks to improve the quality of process outputs, reducing the defects to 3.4 per million, by identifying and removing their causes and minimizing variability. It is applicable to manufacturing and services. It uses statistical methods, and creates a special infrastructure of people within the organization ("Champions", "Black Belts", "Green Belts") who are experts in them.”

**DMAIC, Existing products and services**
- Define
- Measure
- Analyze
- Improve
- Control

**DMADV & DFSS, New or redesigned products and services**
- Define
- Measure
- Analyze
- Design
- Verify
Process standards...
SEI’s Capability Maturity Model Integration

• Not a process, but a meta-process
  – Primarily used by the US government to control estimates from software vendors
  – Would prefer to accept a higher, more stable estimate.

• CMMI measures how well a company measures their own process
The CMMI Framework

1. Process unpredictable and poorly controlled
2. Projects can repeat previously mastered tasks
3. Process characterized, fairly well understood
4. Process measured and controlled
5. Focus on process improvement

Optimizing

Quantitatively Managed

Defined

Managed/Repeatable

Initial
Process Tradeoffs

• (Note: Success stories in many industrial settings, eg. automobile industry.)

• Process vs product quality. Process Quality influences Product Quality, but does not guarantee it

• Following "best practices" as legal defense strategy
  – “Check box compliance”? 
Scenario 1

• You work at an internet company on a large, existing code base.
• A bug manifests in a client-facing product, affecting profits.
• What do you do?
Scenario 2

• You run a small software firm with a handful of really smart engineers.
• Your employees keep having great ideas and building awesome products!
• ...but they’re consistently beat to market by your competitors.
• What do you do?
Scenario 3

• You’re a large consulting firm that works on fixed cost engagements.

• A major client is threatening to cancel such a contract and cease contracting work to you in the future because you are late on several key milestones on two of your engagements with them.

• What do you do?
Summary

• Sequential process models emphasized "think before coding"
• Often too rigid, with changing requirements and environments
• Iteration to address risks
• Change management to control change
• Measure process, continuously improve process