Software Engineering talk

• Title: Lean or Agile software engineering process: an industry perspective
• By Keith Hanson, CEO, Twin Engine Labs
• Time & Place: 5:30pm, Jan 9 2014, Bogard Hall
Product Life cycle (RUP)
View of Software Engineering

User Needs/Inputs – Specification or Requirement

Technical:
Design/Analysis Implementation

Project/Product Management:

Manufacturing/Quality Control

Marketing

Sale
Typical SE life cycle process

- **User Inputs -> Requirements and Specification**
- **Design & Analysis -> Analysis & Design documents**
- **Implementation -> Programming, integration, Tools -> Code or system**
- **Testing -> Test Plan & Test results**
- **Manufacturing, installation/Deployment and Quality Control -> Change Management & Configuration Management.**

- **A classic water model.. Not good..why?**
- **the above SE workflow can be divided into smaller iterations**
Recap:
Implementation Work Flow
Introduction

• Fundamental goal is to build a working version of system.
• Craft out the Architecture
• Implement the Design in terms of components.
• Plan System Integrations
• Implement Design classes and subsystems
• Unit test components
• Integrate Components.
Implementation Workflow Activities and Workers

- Architect & Developers
  - Architectural Implementation
  - Integrate System
  - Implement a Class
  - Implement a Subsystem/component
  - Perform a Test

- System Integrator

- Component Engineer
Artifacts

• Implementation Model
  - Component source code, executable files, libraries, configurations, database etc..
    - Implementation subsystem
    - Interface

• Architecture Description
  - View of Implementation Model

• Integration Build Plan
Implementation

• Describes how elements of design model are implemented in terms of components such as source code, executable, etc.

• Write programs, write or acquire framework or components (IDE, J2EE or AMPP MVC etc)

• Describe how components are organized using the structuring and modularization mechanisms of the implementation environment and programming languages

• Hierarchy of Implementation (sub) Systems containing Components and Interfaces
Component implementation with build tools & version control

- **Build tools**
  - Unix make
  - Ant: XML based build tool
  - Organize implementation into modules
    - source
    - Document
    - Output (binary)
    - Package

- **Version control or change Management**
  - Help manage every change to your implementation or any artifact without any lost
Integration Build Plan

- Describes the build (SW product) that will occur within a given iteration.
- Build plan for each increment
  - list of functionality: use cases and/or scenarios
  - list of implementation subsystems and components
- Test each build, including regression tests
- Configuration management with the build plan
System Integrator

• Is responsible for designing integration build plan and performing incremental integration & perhaps regression test.
The Test Workflow
Main Goals

• To ensure that the system offers a high degree of quality before it’s delivered to customers
• Result: Test Model
• Use-case oriented testing
• Quality Gate
  – Entrance Criteria
  – Exit Criteria
Artifacts

• Test Case
  ➢ Black-box testing-use cases
  ➢ White-box testing-use case realizations-design
  ➢ Integration testing-system level

• Test Procedure
  How to perform
Sample of test reports/results

<table>
<thead>
<tr>
<th>Case ID</th>
<th>Description</th>
<th>Release no</th>
<th>p/f</th>
<th>comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Registration sunny day</td>
<td>1</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Registration with illegal data</td>
<td>1</td>
<td>F</td>
<td>It allows illegal data. Reported in TR001</td>
</tr>
<tr>
<td>3</td>
<td>Login</td>
<td>1</td>
<td>P</td>
<td></td>
</tr>
</tbody>
</table>
• Evaluate Test
  - evaluating results of integration and system (use-cases)
  - comparing to appropriate quality goals
Key Measure

- Completeness of testing or Coverage
- Reliability
- Performance
How to make decision: Readiness of your project

• Requirement & Test Plan
• Objective and Criteria
  – Entrance Criteria
  – Exit Criteria
• Quality Model and Measurement (Software Metrics)
Example of Acceptance Chart by FIO
RUP First half summary
Quality Improvement via Change Control and Configuration Management
Source Code Control

- A tool to track changes in your files.
- Allows to roll back to any changes based on versioning
- Example of these tools are CVS, SVN, GIT
- More covered later
Change Management

• (wiki) is a structured approach to shifting/transitioning individuals, teams, and organizations from a current state to a desired future state.

• In sw engineering, an approach to track, control and transition software/artifact quality toward defined goals.
Why change Management.

- to **maximize** productivity by regulating a systematic way to control changes and **minimizing** mistakes caused by confusion in software development life cycle
- To identify, organize, and control **modifications** to the software being built for a target release
- Must be formulated to **identify** change, **control** change, **ensure** that change is being properly implemented, and **report** changes to others who may have an interest
Why Change Management

• initiated when the project begins and terminates when the software is discontinued

• Roles & Responsibilities
  – Project manager -> regular milestones to changes in the CM and use it as an auditing mechanism
  – Change control manager -> a controlling, tracking, and policy making mechanism
  – Software engineer -> a changing, building, and access control mechanism
  – Customer -> a quality assurance and product identification mechanism
Elements of a Change & Configuration Management

• Configuration & Change Management elements
  – A set of tools coupled with a file management (e.g., database) system that enables access to and management of each software configuration item
  – Version controls

• Process elements
  – A collection of procedures and tasks that define an effective approach to change management for all participants

• Construction elements
  – A set of tools that automate the construction of software by ensuring that the proper set of valid components (i.e., the correct version) is assembled

• Human elements
  – A set of tools and process features used by a software team to implement effective SCM
What can be changes in Software life cycle?

- New Feature Request
- Bug Reports to fix problems
- Artifacts for change control
  - Source code
  - Object code
  - Documents
  - Configurations
Automated SCM Repository (Functions and Tools)

- Versioning
- Dependency tracking
- Change management
- Requirements tracing
- Configuration management
- Audit trails

SCM Repository
- Functions
- Data integrity
- Information sharing
- Tool integration
- Data integration
- Methodology enforcement
- Document standardization

These slides are from Dr. Jay-Evan J. Tevis's software engineering lecture ppt
Functions of an SCM Repository

- Data integrity
  - Validates entries, ensures consistency, cascades modifications
- Information sharing
  - Shares information among developers and tools, manages and controls multi-user access
- Tool integration
  - Establishes a data model that can be accessed by many software engineering tools, controls access to the data
- Data integration
  - Allows various SCM tasks to be performed on one or more CSCIs
- Methodology enforcement
  - Defines an entity-relationship model for the repository that implies a specific process model for software engineering
- Document standardization
  - Defines objects in the repository to guarantee a standard approach for creation of software engineering documents

These slides are from Dr. Jay-Evan J. Tevis's software engineering lecture ppt
Toolset Used on a Repository

- **Versioning**
  - Save and retrieve all repository objects based on version number
- **Dependency tracking and change management**
  - Track and respond to the changes in the state and relationship of all objects in the repository
- **Requirements tracing**
  - (Forward tracing) Track the design and construction components and deliverables that result from a specific requirements specification
  - (Backward tracing) Identify which requirement generated any given work product
- **Configuration management**
  - Track a series of configurations representing specific project milestones or production releases
- **Audit trails**
  - Establish information about when, why, and by whom changes are made in the repository

These slides are from Dr. Jay-Evan J. Tevis's software engineering lecture ppt
Example of Change request in Software engineering

- Per feature or usecase
- Change ID
  - New development
  - Maintenance/ Enhancement
  - Bug report
- Priority
- Severity
- State
Example of change request in Software engineering

• Quality states & the state diagram
• Examples
  – Initial/created
  – Assigned
  – Submitted (developer)
  – Unit tested
  – IT_passed, SVV_passed/final
Example of Quality Transition in CR

Code submission to repository

Assign to a developer

init → as → su → ut → IT → SV

- Unit test Passed
- Intg Passed
- SVV Passed

F

F

F

F
Main Task in Change Management

- Identify changes
- Change Control Task
- Version Control Task
- Configuration Auditing Task
- Report Task
Identification Task

- Identification separately names each CSCI and then organizes it in the SCM repository using an object-oriented approach
- Objects start out as basic objects and are then grouped into aggregate objects
- Each object has a set of distinct features that identify it
  - A name that is unambiguous to all other objects
  - A description that contains the CSCI type, a project identifier, and change and/or version information
  - List of resources needed by the object
  - The object realization (i.e., the document, the file, the model, etc.)
Change Control Task

- Change control is a *procedural* activity that ensures quality and consistency as changes are made to a configuration object.
- A change request is *submitted* to a configuration control authority, which is usually a change control board (CCB)
  - The request is *evaluated* for technical merit, potential side effects, overall impact on other configuration objects and system functions, and projected cost in terms of money, time, and resources.
- An engineering change order (ECO) is *issued* for each *approved* change request
  - Describes the *change* to be made, the constraints to follow, and the *criteria* for review and audit.
- The baselined CSCI is *obtained* from the SCM repository
  - *Access control* governs which software engineers have the authority to access and modify a particular configuration object.
  - *Synchronization control* helps to ensure that *parallel* changes performed by two different people don't overwrite one another.
Version Control Task

- Version control is a set of procedures and tools for managing the creation and use of multiple occurrences of objects in the SCM repository
- Required version control capabilities
  - An SCM repository that stores all relevant configuration objects
  - A version management capability that stores all versions of a configuration object (or enables any version to be constructed using differences from past versions)
  - A make facility that enables the software engineer to collect all relevant configuration objects and construct a specific version of the software
  - Issues tracking (bug tracking) capability that enables the team to record and track the status of all outstanding issues associated with each configuration object
- The SCM repository maintains a change set
  - Serves as a collection of all changes made to a baseline configuration
  - Used to create a specific version of the software
  - Captures all changes to all files in the configuration along with the reason for changes and details of who made the changes and when
Configuration Auditing Task

- Configuration auditing is an SQA activity that helps to ensure that quality is maintained as changes are made.
- It complements the formal technical review and is conducted by the SQA group.
- It addresses the following questions:
  - Has the change specified in the ECO been made? Have any additional modifications been incorporated?
  - Has a formal technical review been conducted to assess technical correctness?
  - Has the software process been followed, and have software engineering standards been properly applied?
  - Has the change been "highlighted" and "documented" in the CSCI? Have the change data and change author been specified? Do the attributes of the configuration object reflect the change?
  - Have SCM procedures for noting the change, recording it, and reporting it been followed?
  - Have all related CSCIs been properly updated?
- A configuration audit ensures that:
  - The correct CSCIs (by version) have been incorporated into a specific build.
  - That all documentation is up-to-date and consistent with the version that has been built.
Status Reporting Task

• Configuration status reporting (CSR) is also called status accounting
• Provides information about each change to those personnel in an organization with a need to know
• Answers what happened, who did it, when did it happen, and what else will be affected?
• Sources of entries for configuration status reporting
  – Each time a CSCI is assigned new or updated information
  – Each time a change is approved by the CCB and an ECO is issued
  – Each time a configuration audit is conducted
• The configuration status report
  – Placed in an on-line database or on a website for software developers and maintainers to read
  – Given to management and practitioners to keep them appraised of important changes to the project CSCIs
Configuration Management

• Wiki “software configuration management (SCM) is the task of tracking and controlling changes in the software. Configuration management practices include revision control and the establishment of baselines.”

• Change control + software manufacturing (building/construction process
Sample tools for SCM

- IBM ClearQuest & ClearCase
- AT&T Sablme
- MS sourceSafe
- GIT (open source change/version control)
- CVS, SVN
- Make/ANT etc..
Building process/tool

- Define your project & dependencies
- Smart compilation with changes & dependency
- Tools such as Make & Ant
Summary

- SCM = Software Change/configuration Management is a vital process + tools to help ensuring to gradually control changes to your software products with quality assurance
- The concept can be applied to other product development