Leadership
Release Management
Continuous Integration

october 9, 2013
Significant Semester Requirements

- 15% Participation: in-class, at meetings, setup meetings in timely manner, weekly status reports (individual grade)
- 15% Writing: executive summary, project plan, design documentation (team grade)
- 15% Deadlines: sprint deadlines met and meeting planning done in timely manner (team grade)
- 15% User Testing: evidence users have tested, feedback report (team grade)
- 15% User Friendly Design: easy, non-programming, way for user to change client data (team grade)
- 25% Client Satisfaction: client consistently reports good progress (team grade)
Discussion Topics

- Pre-Sprint Planning
  - Define tasks/stories and estimates
- Sprint Planning
- Management vs Leadership
- Software Release Management
- Continuous Integration
Pre-Sprint Planning

Before Sprint Planning can begin, the team needs to have the tasks/stories defined and estimated. This is done as part of pre-sprint planning.

- **Break Requirements into Tasks**
  - Go through each Product Backlog item selected for the Sprint. Break the requirements into tasks.

- **Shrink Tasks to Improve the Task-Based Burndown.**
  - A good, informative task-based burndown chart depends on there being many small tasks to burn down.
Sprint Planning

The purpose of Sprint Planning is for the Product Owner and the team to negotiate what should be accomplished during the sprint.

- **Set the Sprint Budget**
  - First calculate the team’s Sprint Budget. This is the available number of hours the team has to work on the Sprint.
  - Then, make any reasonable deductions for time that team members will not be able to spend working on the Sprint. Deduct holidays, any known meetings, any time likely to be spent working on other projects, etc. Based on past experience, deduct a reasonable amount of time for support, if appropriate.

- **Establish Stable Velocity**
  - Use your normal sprint planning process for each sprint until you can demonstrate stable velocity.

- **Build the Sprint**
  - Add tasks to satisfy the Sprint Budget and velocity.
  - Identify stretch tasks to cover times when the team under-commits or over-estimates.
Calculating Velocity

Velocity is how fast you are developing your software.

In Scrum, Velocity is how much product backlog effort a team can handle in one Sprint.

Let's consider the following example.

A team of 5 developers worked on the project we are estimating. The Sprint length is 2 weeks. In this Sprint they committed to completing 25 story points.

<table>
<thead>
<tr>
<th>Team Member</th>
<th>Hours Worked</th>
</tr>
</thead>
<tbody>
<tr>
<td>Julie</td>
<td>16</td>
</tr>
<tr>
<td>Adam</td>
<td>10</td>
</tr>
<tr>
<td>Sam</td>
<td>14</td>
</tr>
<tr>
<td>Mary</td>
<td>16</td>
</tr>
<tr>
<td>Ann</td>
<td>14</td>
</tr>
<tr>
<td><strong>Sum:</strong></td>
<td><strong>70</strong></td>
</tr>
</tbody>
</table>

The team delivered software worth 25 story points in 70 hours.
Management vs Leadership

What is the Difference Between Management and Leadership?

- Leadership and management must go hand in hand.
- Workers need their managers not just to assign tasks but to define purpose.
- Managers must organize workers, not just to maximize efficiency, but to nurture skills, develop talent and inspire results.
Management vs Leadership

- The manager maintains; the leader develops.
- The manager focuses on systems and structure; the leader focuses on people.
- The manager relies on control; the leader inspires trust.
- The manager has a short-range view; the leader has a long-range perspective.
- The manager asks how and when; the leader asks what and why.
- The manager has his or her eye always on the bottom line; the leader’s eye is on the horizon.
- The manager imitates; the leader originates.
- The manager accepts the status quo; the leader challenges it.
- The manager is the classic good soldier; the leader is his or her own person.
- The manager does things right; the leader does the right thing.
Software Release Management

Software release management encompasses the identification, packaging, and delivery of the elements of a product. This includes executable program, documentation, release notes, and configuration data.

Benefits

- Helps make software builds simple, quick and reliable
- Helps take care of configuration management.
- Helps drive software quality.
- Helps optimize development and QA time.
- Speeds up time to market.
Release Mgmt in Repository

- main development branch
- stable branch
- development branches
- release candidates
Software Release Management

Software release management encompasses the identification, packaging, and delivery of the elements of a product. This includes executable program, documentation, release notes, and configuration data.

Benefits

- Helps make software builds simple, quick and reliable.
  - This is achieved by employing the best tools for the job. This means understanding all the various build tools, seeing how they integrate with the systems that already exist in the workplace, and making an informed choice.
Software Release Management

Software release management encompasses the identification, packaging, and delivery of the elements of a product. This includes executable program, documentation, release notes, and configuration data.

Benefits

- Helps take care of configuration management.
  - Software configuration management (SCM) is the task of tracking and controlling changes in the software, part of the larger cross-discipline field of configuration management.”
  - SCM practices include revision control and the establishment of baselines. If something goes wrong, SCM can determine what was changed and who changed it.
  - If a configuration is working well, SCM can determine how to replicate it across many hosts.
Software Release Management

Software release management encompasses the identification, packaging, and delivery of the elements of a product. This includes executable program, documentation, release notes, and configuration data.

Benefits

▪ Helps optimize development and QA time.
  ▪ By giving the dev team the feedback on the quality of their code and telling them where they’re going right and going wrong, we’re helping them target their efforts.
  ▪ By providing these solutions for them, doing the builds, configurations and releases, the developers can get busy doing the stuff they’re skilled at doing.
  ▪ For the QA team, we’re finding defects and failing releases before the releases even get to them.
Software Release Management

Software release management encompasses the identification, packaging, and delivery of the elements of a product. This includes executable program, documentation, release notes, and configuration data.

Benefits

- Speeds up time to market.
  - Making builds quicker, easier and more reliable, accelerates the process of fine tuning code quality.
  - This also helps identify defects before QA starts and the process of releasing the software into production quicker and simpler.
  - This saves significant development, QA and operations and so our new, higher quality software, can be released efficiently into production.
Software Release Management

Software release management encompasses the identification, packaging, and delivery of the elements of a product. This includes executable program, documentation, release notes, and configuration data.

Benefits

- Helps drive software quality.
  - Thanks to the Continuous Integration process, and the tools that have been built around it, it’s now possible for us to build software every single time a piece of code is checked in, run a suite of unit tests, analyze the code for lazy programming and report on the amount of test coverage a project has.
  - Jenkins is an open-source continuous integration software tool written in the Java programming language for testing and reporting of defects in an automated fashion.
Continuous Integration

Continuous integration (CI) is the practice, in software engineering, of merging all developer working copies with a shared mainline several times a day. Its primary purpose is to prevent integration problems.

Continuous Integration with automated test execution has seen broad adoption in recent years. The ideas behind Continuous Integration have changed how companies look at Build Management, Release Management, Deployment Automation, and Test Orchestration.

Jenkins is a popular industry CI tool which provides an easy-to-use continuous integration system, making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. The automated, continuous build increases the productivity.
Continuous Integration Example 1
Basic configuration.
Continuous Integration Example 2

Basic configuration plus automated testing.