AN INTRODUCTION TO THE AGILE METHODOLOGIES AND CRYSTAL

S. Salamah
CS 5389-Software Engineering Practicum
Spring 2018

Overview

- What is Agile?
- Misconceptions of Agile
- Crystal Clear
  - What is it?
  - Fundamental Properties of Crystal
  - Processes
  - Tools
- Expectations
What is Agile?

- Agile is an alternative methodology used to define engineering process
  - Initially, Agile was used for software engineering projects
  - Now, Agile is applied to other project types
    - Systems Engineering
    - Marketing
    - General team-based project management

Agile Manifesto

- **Manifesto for Agile Software Development**
  
  We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

  - Individuals and interactions over processes and tools
  - Working software over comprehensive documentation
  - Customer collaboration over contract negotiation
  - Responding to change over following a plan

  That is, while there is value in the items on the right, we value the items on the left more.

From: AgileManifesto.org
### Examples of Agile Methodologies

- Scrum
- Extreme Programming
- Crystal (Clear, Yellow, Orange, etc.)
- Rationale Unified Process
- Etc.

### Key and Common Attributes

- Initial team process defined and agreed upon
- Process lead, manager, coach designated
- Iterative development
  - Short work periods called iterations, sprints, etc.
    - 2 weeks to 1 month are common
  - Team determines features to implement and sticks to the plan
  - Delivery of a working (but perhaps incomplete) system to customer
- Before each work session, a standup meeting occurs to determine session’s goals
- After each iteration, team reflects upon changes
Common Misconceptions

- You do not need a process because you are Agile
- You do not have to produce documentation
- Missing deliverables is ok because you are “Agile”
- Teams do not need to designate a lead

WRONG!!!

Agile is a light-weight process to allow teams to focus on design and requires vigilance to ensure the defined process is being followed.

Documentation is not optional. It should be considered a part of the deliverable.
Introduction to Crystal Clear

- What is it?
- Fundamental Properties of Crystal
- Processes
- Tools

Crystal: What is it?

- Agile and lightweight process for teams
  - Crystal Clear – Small teams (<15)
  - Crystal Yellow – Medium size teams

- Alistair Cockburn, 2005

- Starts with more of guidelines, suggestions, and tools and allows the team to develop the process

- Comprised of seven fundamental properties
Properties of Crystal

1. Frequent delivery
2. Reflective improvement
3. Osmotic communication
4. Personal safety
5. Focus
6. Access to expert users / customers
7. Technical environment w/ automated testing and frequent integration

Frequent Delivery

- Break up the development period into iterations
  - Month, week, two weeks, etc.
  - We will use two week iterations.
- Integration, iteration, user viewing, and release
  - Differences?
- Delivery to customer / user at the end of every iteration
- “Walking Skeleton” approach works quite well
  - Create an initial skeleton of the project and build upon the skeleton with each iteration until the last iteration delivers a final product.
  - Skeleton is up and running for the duration of development (different than a spike or throw-away prototype)
  - Requires discipline to make sure that you don’t lose a limb in the process
Reflective Improvements

- Things never work out as planned, but we can/should learn from them.
- As we collect data, we may modify the process to improve efficiency.
- Likewise, we each have our own concerns, suggestions, etc.
- Reflection workshop provides an opportunity to reflect upon the past iteration and make adjustments (maybe a complete reverse of course) to the process

  Pair programming? Type of unit testing? TDD?, seating type? ...

An hour every week or around the end of an iteration will do

Reflection Workshop

- Open and safe discussion
- Three elements:
  - Keep these
  - Problems / Issues
  - Try These
- Focus is not on technical issues, but on process issues
Reflection Workshop (2)

**Iteration 1**

<table>
<thead>
<tr>
<th>Keep These</th>
<th>Try These</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Team leader meetings&lt;br&gt;• Episode scripts</td>
<td>• Meeting agenda&lt;br&gt;• Unit test&lt;br&gt;• Integration test&lt;br&gt;• Py-Unit&lt;br&gt;• Team redesign</td>
</tr>
</tbody>
</table>

**Problems**

• Code commenting<br>• Naming conventions

---

Osmotic Communication

Process of gradual or unconscious assimilation of ideas, knowledge, etc.

- Single work environment
- Frequent inter and intra team communication
- Pair-programming
- Display all reflections, planning, design, task assignments, etc. for all to see

Reduces the cost of communication and increases rate of feedback
Osmotic Communication

• Not the same as pair-programming
• Facilitates osmotic communication
• Might replace design and code reviews

Side-by-Side Programming
Standup Meetings

- At the beginning of each work episode, a standup meeting should be held.
  - Form a circle
  - Everyone STANDS UP!!!
  - Discuss your current progress
  - Disclose if you are behind
  - Discuss problem resolutions with group

- Personal safety is critical here!
  - This is used to help you feel a sense of accountability and to communicate issues
  - Never attack, berate, accuse, point fingers, etc. during this time

Personal Safety

- Everyone is given the opportunity to speak freely and openly
- Frequent delivery allows you to identify strengths and weaknesses of self and team
- A level of trust is developed as product is delivered at each iteration
- Competitive spirit may arise, but always remember it is a team effort

- Exposure is necessary to establish trust:
  - Revealing your ignorance
  - Revealing a mistake
  - Revealing incapability

- Try to expose yourself and your team to these early on and get it done with
Focus

- Important that everyone knows what tasks are being worked upon during this iteration.
- By doing initial planning at the beginning of iteration, the tasks are identified, understood, and assigned.
- You are committed to only delivering the tasks called for during the iteration.
  - However, if free time becomes available, it is up to you to take advantage of it!!!

Access to expert users

- It is important to have access to expert users and/or customers
- Delivery and demonstration to customer at the end of every iteration
Technical environment

- Your technical environment must be established to support your work
- Lab layout and equipment
- Version control software (SVN)
- Unit testing tools (if available)
- Closed lab
- Resolve issues at the beginning of an iteration

Suggested Tools

- Blitz planning
- Technology spike
- Walking skeleton
- Bazaar Planning
- Any tools you wish to carry over from PSP/TSP
  - Data collection is still important and required
  - Time logs are a must
Blitz Planning

- Based on XPs, “planning game”
- Use note cards to list out tasks and create a high-level plan for entire project

<table>
<thead>
<tr>
<th>Task</th>
<th>Owner</th>
<th>Time Estimate (Iterations)</th>
</tr>
</thead>
</table>

1. Assemble Team
   - Team members, expert users, instructors, etc.
2. Brainstorm
   - Individually, create note cards for presumed tasks.
   - Tasks are not only project-oriented but also process-oriented
   - Do this in no more than 20 minutes
3. Each team organizes tasks into a sequence
   - Parallel tasks are placed in parallel vertically
4. Layout tasks globally
   - Identify initial walking skeleton and build from there
5. Review and add lost tasks if any
6. Break timeline into iterations and identify dependencies
7. Mount post cards and display
Bazaar Planning

- For your iteration, look at your feature cards
  - Break down features into a small set of deliverables
  - Each deliverable should be achieved by one person
  - Each deliverable should be testable!!!

- Now, write each of these on slips of paper and put them in a hat (or other vessel of your choice)
- Now, play the “Bazaar” game

Bazaar Planning (2)

- Now, play the bazaar game…
  - Team member pulls task out of hat
    - If they want it, they keep it
    - If they don’t want it, they put it back and go to the back of the line
    - You can only do this once
  - Go to the next person.
  - Repeat until everyone has at least one task and all tasks are out of the hat.
  - If anyone so desires, they may trade so long as both parties are in agreement.
    - Negotiate; e.g., one hard task for 2-3 easy tasks is fair.

- Post the task assignments somewhere where everyone can see
  - (OSMOTIC COMMUNICATION!!!) & (INDIVIDUAL ACCOUNTABILITY)
Notes on Iteration

- Practicum team is encouraged to create own process script
  - Nothing set in stone. Your process may change.
- Commit to your deliverables at the beginning of the iteration
- Stick to the plan
- Failure to delivery
  - Should be avoided, but sometimes is inevitable. Do not alter deadlines
  - Notify others. Move to next iteration.
  - Tasks at the last iteration must be moved out of the project
    - i.e. lower overall grade due to failure to meet requirements

References

- Good reference on how to manage and agile process
- Will be placed in the lab.
  - Use as you wish, but do not remove from the lab.
# Semester Teams

<table>
<thead>
<tr>
<th>Team 1</th>
<th>Team</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jesus</td>
<td>Daniel</td>
</tr>
<tr>
<td>Jesus</td>
<td>Daniel</td>
</tr>
<tr>
<td>Marco</td>
<td>Angel</td>
</tr>
<tr>
<td>Victoria</td>
<td>Lennox</td>
</tr>
</tbody>
</table>