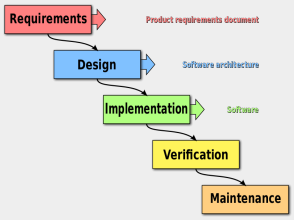
Lab: Agile

Experiencing Waterfall vs Agile

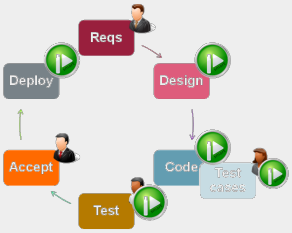
The traditional method of developing complex programs used the waterfall method:

1. Analysis
2. Design and Specifications
3. Write the code
4. Deploy
5. Maintenance (debug, debug, debug)



This process, developed after World War II by Deming, works well for designing factory assembly lines or highways. However, when designing software there are variations that develop, similar to linguistics, that make this predictive approach to be less effective.

One alternative is the agile approach. Many quick versions working toward an end goal that allows variations to become part of the finished product.



This lab will demonstrate one reason why the agile approach is effective for software development.

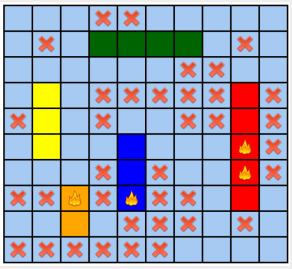
Getting Setup

Pair off into teams. Each team should have access to a computer and this site Battleship.

Note: If working online, video conference with a partner, with one person sharing their screen. For group presentations online, have each student running the simulation as the instructor tabulates the results for each round.

Round One - Waterfall

Set the "Shots per iteration" to 40 and click "New game". The first programmer uses the waterfall method by analyzing the problem (hit all the hidden ships representing product features) and selecting all 40 shots (predictive model).

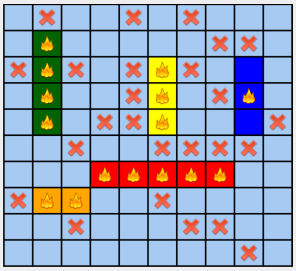


Together count how many were successful by the number of flame icons.

- How many features did you successfully complete? (How many ships did you actually sink?)

- What percentage of your hits were accurate? For example, if you had 4 hits and there were 17 places to hit a ship, 4/17 = .2352 or 24%.
- Take a screen shot of the screen for reference. Write this information down

Round Two - Agile



Refresh the page. Set the "Shots per iteration" to 1 and click "New game". The second programmer uses the agile method by taking a shot and getting immediate feedback.

- How many features did you successfully complete? (How many ships did you actually sink?)
- What percentage of your hits were accurate?
- Take a screen shot of the screen for reference. Write this information down.

Round Three - Modified Agile

Refresh the page. Set the "Shots per iteration" to 20 and click "New game". The first programmer uses a modified agile method by taking several shots. Run the iteration twice, keeping track of your successes each time. (40 = 20*2)

- How many features did you successfully complete? (How many ships did you actually sink?)
- What percentage of your hits were accurate?
- Take a screen shot of the screen for reference. Write this information down

Round Four - What if?

Refresh the page. Decide together how many shots would be practical, getting feedback as you solve the overall software problem without spending time on a meeting after each iteration. Whatever number you choose, run the simulation enough times to equal 40 shots total. For example, if you choose 5 shots per iteration, then run the simulation 8 times (40=5*8).

- How many features did you successfully complete? (How many ships did you actually sink?)
- What percentage of your hits were accurate?
- Take a screen shot of the screen for reference. Write this information down.

Analysis

Round One uses predictive planning or the use of the waterfall methodology.

Round Two uses the agile approach, with immediate feedback after each cycle or version.

Round Three still uses the agile method, but is more practical, with less feedback (client meetings) but still tracking the process as it develops.

Round Four allows you to "noodle around" (a technical programming term) to see how different levels of feedback can affect the success of a project.

Answer these questions as a team

1. Which round was most effective on "hitting the target"? Why is that?
2. A development team, along with the manager and client, might not be able to meet multiple times each day. Based on your experimentation with different "Shots per iteration", how often should there be feedback in this simulation?
3. What online tools can you think of that can be useful in making feedback less cumbersome?
4. How important do you think the first stage "Requirements" in the waterfall method is? How beneficial would it be to include this in the agile development cycle?
5. If things are constantly changing as a project is completed, how can teams prevent scope creep without resorting to the waterfall method?

Summary

The waterfall methodology is useful for designing processes that are consistent with little variation as they are being built. Examples include designing a manufacturing assembly line or building a highway.

The agile methodology is useful for designing processes that have lots of variation introduced during the development cycle. Examples would include writing complex software, raising children, and the continual improvement of business areas such as customer service, human resources, and professional development and training.

Special thanks to Tasty Cupcakes for outlining this game and the JavaScript version out on GitHub by Mark Suurmond of ZilverLine.com.

Images courtesy of Wikipedia, https://www.wikiwand.com/en/Waterfall_model and, https://upload.wikimedia.org/wikipedia/commons/8/8d/SPADE_automates_Software_development_activities.png

Written by Peter Johnson, WebExplorations.com

Revised: 01-02-17