

Agile Product Planning and Estimation

Welcome & Introductions

- Steve Ropa
 - Agile Coach
 - Product Consultant
 - Certified Scrum Master
 - Certified Scrum Product Owner
 - 17 years software development
 - 11 years programming
 - 8 years director of development
 - 10 years Agile experience
 - XP
 - Scrum

Planning Software Projects is Hard



How did it get this way?

- Dr. Winston Royce – Planning the Development of Large Software Systems

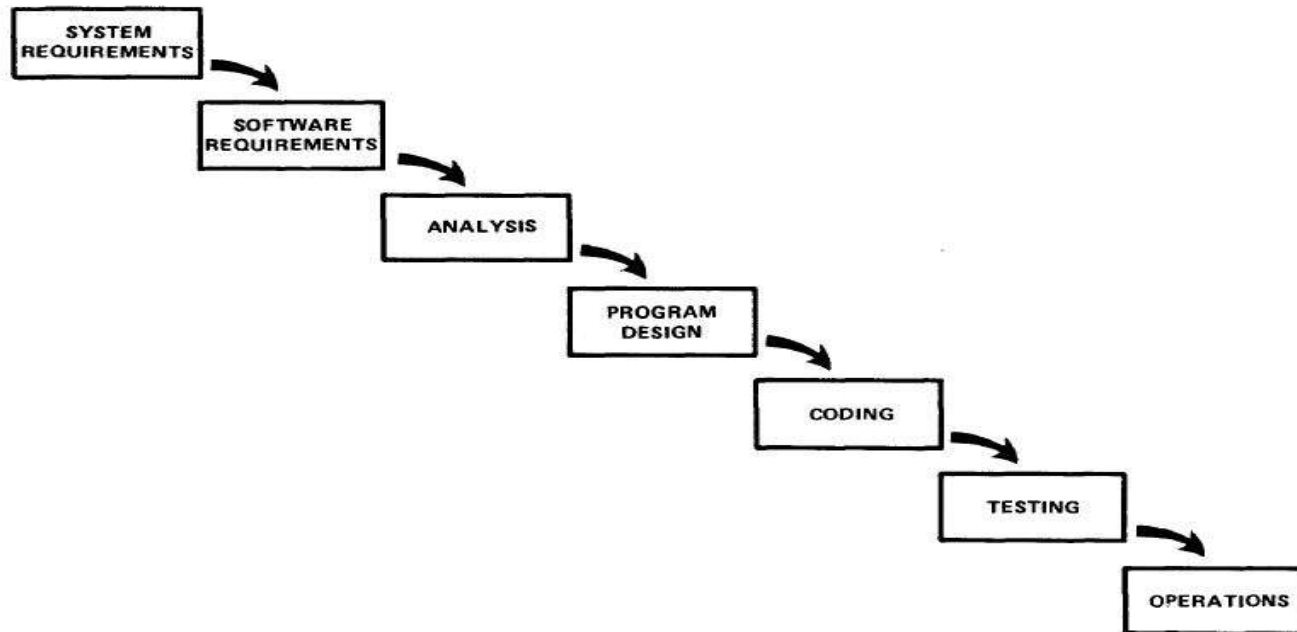


Figure 2. Implementation steps to develop a large computer program for delivery to a customer.

Many people took this as the “revealed truth”

- But that isn't what the rest of the paper said:
- “I believe in this concept, but the implementation described above is risky and invites failure. “



How a waterfall becomes an avalanche

- All of the requirements were done up front, prior to really knowing what the system will do and how it will grow.
- Once the flow starts, entropy sets in – no room for change
 - But things need to change anyway
- Just as the phases cascade, any delay will cascade from phase to phase, snowballing along the way



So we tried to fix it...

- Once the requirements are written, they aren't allowed to change
 - But they do
- Everyone will write the requirements in the exact same way, so everything is clear
 - But it really isn't ever clear enough
- We will pay very experienced people a lot of money and give them exalted titles so they can tell us exactly how a system will look and how long it will take.



Ok, that didn't work, now what?

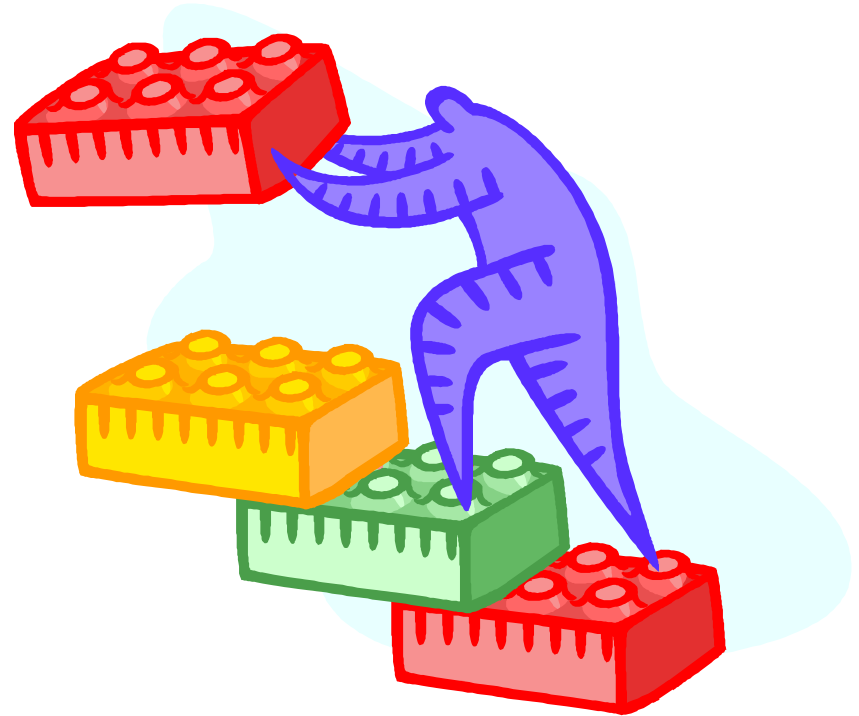


Agile Software Development

- We are uncovering new 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.

Agile Planning Building blocks

- User Stories
- Acceptance Tests
- Estimation
- Release Planning
- Iteration Planning
- Tasks (optional)
- Review

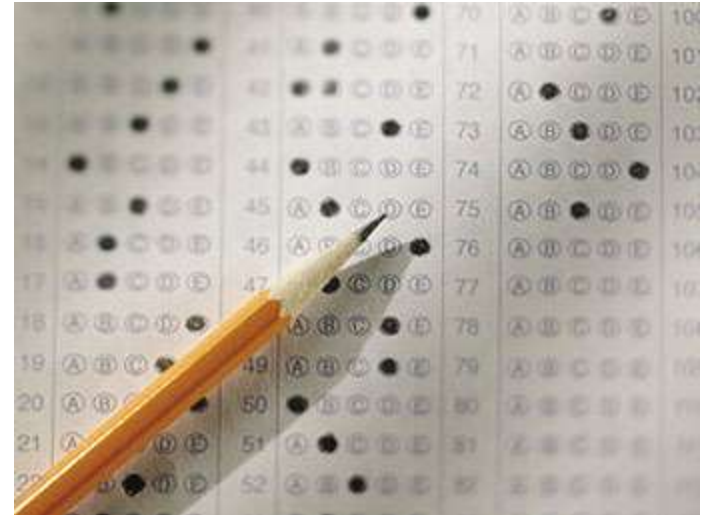


User Stories



- A brief description of a piece of functionality that will provide value
- Should be small enough to fit on a 3x5 index card
- A story is merely a reminder to have a conversation about a feature – it is not a specific requirement
- A story should be able to be completed in one iteration.

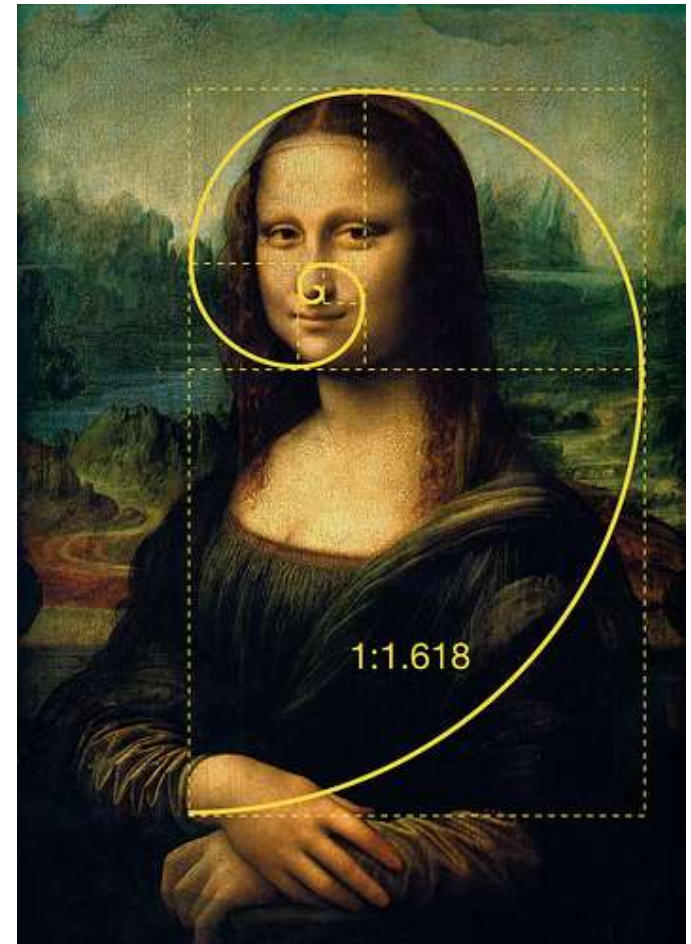
Acceptance Tests



- What is “done”?
- Product owner/customer and development team work together to define done.
- Acceptance Tests are a list of criteria that, when met indicate that the story is complete.
- Whenever possible, these should be automated

Estimation

- In Agile methods, we estimate size, not duration
- Estimates are intentionally vague, and often unit-less
- Common estimate values are
 - T-shirt sizes
 - Scale of 1-10
 - Fibonacci Sequence



Planning Poker



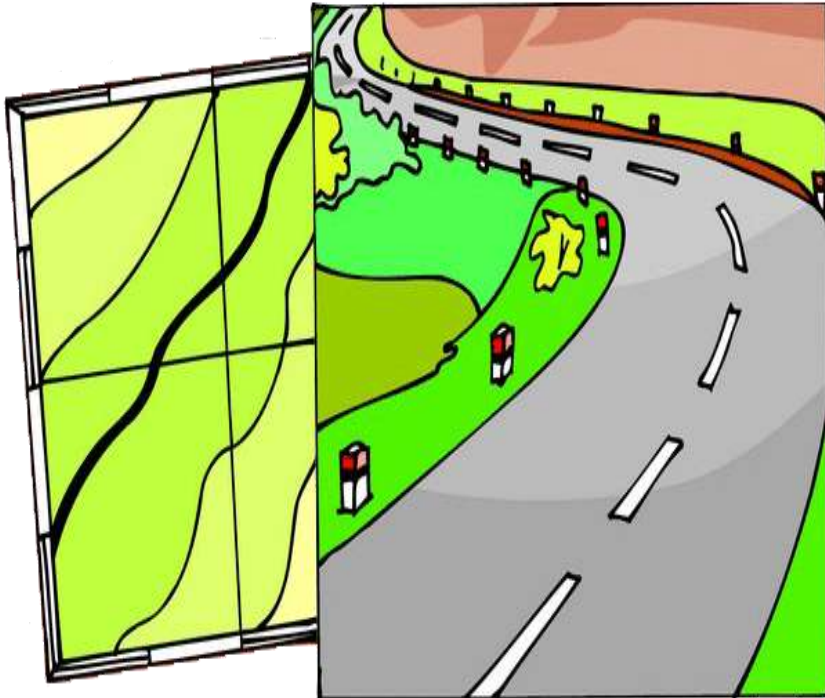
- Product Owner reads the story
- Team asks questions and discusses for 5 minutes
- Each team member selects a card for the size they believe the story is
- High and low values get 5 minutes to explain why they believe so
- Repeat until consensus is reached

Velocity

- How many points worth of stories do we get truly done in a given iteration?
- Yesterday's Weather
 - If we got 25 points done last iteration, we can probably do 25 this one.
- Don't forget sustainable pace



Release Planning



- Product Owner/Customer prioritizes stories in order of when she would like to see them.
- Product Owner identifies the release date.
- Team estimates size of stories using agile estimation techniques such as planning poker
- Based on velocity, identify how many stories should be completed by the release date.
- This release plan will change over the course of the iterations.

Iteration Planning

- Determine Capacity for this iteration
 - What can the team comfortably commit to?
- Start with the top story in the list
- Discuss what it will take to get this story done
- Is the current estimate still good, or should we re-estimate?
- If we are using tasks, this is a good time to identify and estimate them
- Can we commit to accomplishing these stories this iteration?



Tasks

- Break down the story into actionable pieces that can get done in a day or two
- Avoid tasks like –
 - Write the code
 - Test the feature
- Some folks like to treat tests like tasks also
- Estimating tasks
 - Ideal hours
 - Simple count



