Are We There Yet?
An Agile Planning Workshop


Your Coach: Paul Hodgetts
- Team coach, trainer, consultant, developer
- Founder and CEO of Agile Logic (Fullerton)
- 26 years overall, 10 years agile experience
- Certified Scrum Trainer
- Worked with a lot of “enterprise” teams
- Author (Extreme Programming Perspectives)
- Speaker at conferences (Agile 200x, SD East/West, JavaOne)
- Active in Scrum Alliance, Agile Alliance (Program Director)
- Member of CSUF agile advisory board
- Contact info: phodgetts@agilelogic.com
Agile Framework

- Iteration/Sprint
  - Daily
- Release/Milestone

“Backlog” (Queue)

Agile Team

Agile Cycles

“Predictive” Planning

- Predict the tasks required to deliver
- Usually combined with “waterfall” approach
  - Big batch, long cycle, phased activities
- Goal is to map out the plan, manage to it
Why Predictive Planning Fails

- **Descartes Theory**
  - Analysis leads to convergence on a single answer
  - Assumes high degree of stability in environment
  - Requires early choices without concrete feedback

- **Chaos Theory**
  - Complexity creates convergence on probability
  - Learning in complex environment spawns emergence
  - Software development complex enough to be chaotic
  - We will discard significant amount of predictive plans

“Agile” Planning

- Set goals, continually steer to best outcome
- Progressive refinement based on feedback
- Low WIP, short nested cycles, iterate on plans
- Highsmith cycle:

  - Envision
  - Speculate
  - Close
  - Adapt
  - Explore
Levels of Agile Planning

From a time perspective:
- Business Cycle
- Product Cycle
- Product Release Cycle
- Development Milestone (integration event)
- Development Cycle (iteration / sprint)
- Construction Cycle (synch event, daily scrum)

Levels of Agile Planning

From a granularity perspective:
- Product Capability (features sets / areas)
- Feature (related capability cluster)
- Feature Capability (story)
- Work Element (task cluster)
- Work Task
- Work Episode
Keys to Making Agile Planning Work

1. Use probability and statistics in our favor
2. Know what we’re delivering
3. Establish our knowledge base
4. Leverage team collaboration and expertise

Estimates Are Just That...

- Estimates are guesses, some better, some worse
- Each estimate has a probability of correctness
  - “Log Normal” probability
  - Complexity of software biases to it more often taking longer than less time
“Precision” in Estimating

- Learning must be concrete to be effective
- Must actively reduce uncertainty / risk

“Precision” of Estimates

- We do not have point precision in estimates
- Estimates tend to form probability clusters (“buckets”)
- Size affects distribution – scale intervals increase
- Tune the scale to your product / team aspects
Central Limit Theorem

- The more samples we have, the more the overall result converges on an attractor
- The more items we estimate, our overall estimation variance will decrease and stabilize

2. Know What We’re Delivering

- We need a common understanding of “done”
- Consider each gating point:
  - Task Completion
  - Story Completion
  - Iteration / Sprint Completion
  - Release Completion
- Build a checklist of what’s expected at each
Example Completeness Criteria

Abstract Estimating

Understand Deliverable Evaluate Criteria Synthesize Potential To Use Our Capacity Work Time "Ideal Time" "The Plan" Calendar Time Elapsed Time "The Date"

Abstract Estimating "Story Points" Concrete Estimating This can all be automatic!
Evaluating Criteria

- Goal is to generate a rating of “potential to consume our capability”
  - Ratings must have relative size
- What’s needed to be effective?
  - Understanding of the item
  - Understanding of development strategy
- Some prep may be needed
  - Collaborative exploration (“meet and greet”)

3. Agile Estimating Knowledge Base

- Keep a team criteria check list
  - Draw on the broad team expertise
  - Use it for cross-learning
- Keep the criteria list relevant
  - Use retrospectives to explore exceptions
  - Metrics of estimates vs. actuals are not as relevant as root cause analysis

- Estimation by limited individuals is problematic:
  - Smaller knowledge base to draw on
  - Less discussion, more potential for blind spots
  - Lack of buy-in from those responsible to deliver
- We want wider involvement when estimating
- But larger group activities have issues:
  - Hard to reach a group decision
  - Easy to drift off to tangent issues
  - Difficult to gain participation of everyone
- We need a protocol to control the activity

“Planning Poker”

- “Wide-band” estimating technique
- Each team member gets a set of estimate cards
- Go through the backlog items, one-by one:
  1. Discuss and understand each one
     - Ensure it’s “good”
     - Ensure we have a strategy for implementing
     - Consider criteria, synthesize
  2. Each team member simultaneously reveals rating
  3. Discuss highs and lows, until alignment reached
- More useful when abstraction level is higher
- Less useful when items are more specialized
Rate the “Dog-ness”

Labrador Retriever

Dachshund
Rate the “Dog-ness”

Poodle

Rate the “Dog-ness”

German Shepherd
Rate the “Airplane-ness”

- Criteria (in rough order of influence):
  - Size
  - Speed
  - Passenger capacity
  - “Cool-ness”

Boeing 737
Rate the “Airplane-ness”

Cessna 150

Rate the “Airplane-ness”

C-5A Galaxy
Rate the “Airplane-ness”

Airbus A-380

Rate the “Airplane-ness”

Sopwith Camel
Rate the “Airplane-ness”

SR-71 Blackbird

“Wall of Wonder” Planning

- Make sure relative sizes feel right
- Set a cut-off size and stick to it
- Set a time-box for discussing each item
### Steps for Developing an Agile Plan

- Generate goals and target items (backlog)
- Estimating size of items
  - Abstract vs. concrete estimates
- Estimating capacity to deliver
  - Velocity (size units per time box)
- Prioritizing items
  - Rank ordering is the most concrete
- Mapping to a plan
  - Dates are mapped, not estimated!

### Release Planning

- Planning for release milestones from the team
  - Could be either internal or external release
- At a longer time scale
  - Usually multiple iterations / sprints
- At a higher level of abstraction
  - At the feature (epic story) / story level
Estimating Velocity

- First velocity estimate is a guess
- Gain additional feedback from first iteration / sprint planning session
- Actual iteration / sprint results establish actual velocity feedback
- For ongoing sprints:
  - Use “yesterday’s weather”
  - Moving average of prior actual velocities
  - Velocity should stabilize
    - But not if team or project is changing

Prioritizing Product Backlog

- We need to break the “all or nothing” mindset
- Priority is based on many factors:
  - Value of the capability to stakeholders
  - Relative cost vs. value (ROI)
  - Risks and unknowns
  - Dependencies from sagas or epics
- Mixed priorities may require splitting
- Backlog items are sorted by priority
- Sequencing reflects incremental strategy
Mapping to a Release Plan

**Product Backlog**

- **Velocity**
  - **Target Date Planning**
    - How many stories fit into the total number of iterations up to the target date?
  - **Target Scope Planning**
    - In which iteration does the target story land?
    - What is the end date?

Iteration / Sprint Planning

- Planning for a development time box
  - Represents a complete integration event
- At a shorter time scale
  - Usually 1 to 4 week time boxes
- At a lower level of abstraction
  - At the small story level
    - Must be “good” stories
  - Often decomposed to tasks
Iteration / Sprint Planning Flow

- Target Stories
- Work Elements
- Individual Tasks
- Tasks with Work Time
- Tasks with Ownership

Minimum Required for Iteration Planning
Can be part of Planning (Scrum)
Must be kept current
Can be just-in-time (XP)
Should be just-in-time Specialization!

Task Breakdown and Sizing

- Target tasks to deliver something concrete
  - Consider milestones toward story complete
  - Specialization forces specific breakdowns
- Make tasks useful for tracking
  - Tasks should be completed in about a day
  - Breakdown larger task clusters
- Task estimates typically in “ideal hours”
  - Consider completeness criteria
  - Limited wide-band estimating can be useful
Estimating Capacity

- Capacity to deliver is not just available hours!
  - Velocity should drive committed work
  - But hour capacity data can be useful
- Create a matrix (spreadsheet):
  - Each team member’s available days
  - Standard hours per day (consider overhead)
  - Adjustment for shared / part-timers
  - Calculate each team member’s hours, total
  - Specialization forces additional calculations

Sample Capacity Matrix

<table>
<thead>
<tr>
<th>Team Member</th>
<th>Available Days</th>
<th>Hours per Day</th>
<th>% Dedicated</th>
<th>Total Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fred</td>
<td>10.00</td>
<td>6.50</td>
<td>100%</td>
<td>65.00</td>
</tr>
<tr>
<td>Sanjiv</td>
<td>10.00</td>
<td>6.50</td>
<td>100%</td>
<td>65.00</td>
</tr>
<tr>
<td>Carla</td>
<td>10.00</td>
<td>6.50</td>
<td>50%</td>
<td>32.50</td>
</tr>
<tr>
<td>Jose</td>
<td>8.00</td>
<td>6.50</td>
<td>100%</td>
<td>52.00</td>
</tr>
<tr>
<td>Wu</td>
<td>10.00</td>
<td>6.50</td>
<td>100%</td>
<td>65.00</td>
</tr>
<tr>
<td>Bob</td>
<td>10.00</td>
<td>6.50</td>
<td>100%</td>
<td>65.00</td>
</tr>
<tr>
<td>Dimitri</td>
<td>10.00</td>
<td>6.50</td>
<td>33%</td>
<td>21.45</td>
</tr>
</tbody>
</table>

**Total Hours:** 365.95
Prioritization Within Iterations

- Prioritization should follow story priority
  - Team may optimize ordering within iteration
- But be careful of risking completing stories
  - Watch out for “fan-out” (“Scrummerfalls”)
  - Team should “swarm” on each story in turn
  - Variance should be limited to one story

Incorporating Feedback

- Iteration / Sprint Plan Feedback
  - Stories completed vs. stories committed
  - Task completion vs. estimated tasks
  - Actual team member availability
- Release Plan Feedback
  - Actual velocity for each sprint
  - Actual story size vs. estimated size
  - Changes to other stories from feedback
Tracking: Burndown w/Status

Project Burn Down Chart

Completed Features
Detailed Features
Preliminary Features

Question?
Thank You For Attending!

Presentation materials will be available in the Resources section of www.agilelogic.com

Paul Hodgetts
Agile Logic
www.agilelogic.com
phodgetts@agilelogic.com
(714) 577-5795