Agile Software Development

Lecturer: Raman Ramsin

Lecture 18

Scrum: High-Level Planning
Multilevel Planning in Scrum

- Scrum defines only \textit{sprint} planning and \textit{daily} planning (via the daily scrum).
  - However, organizations typically require higher-level planning activities as well, in the form of \textit{strategic}, \textit{portfolio}, \textit{product}, and \textit{release} planning.

[Rubin 2012]
High-Level Planning Activities

1. **Strategic Planning**: An activity for determining sets (portfolios) of related software development projects that an organization needs to carry out.

2. **Portfolio Planning/Management**: An activity for determining which products of a portfolio to work on, in what order, and for how long.

3. **Product Planning (Envisioning)**: An activity for capturing the essence of a potential product, and creating a rough plan for its creation.
   - Envisioning begins with the creation of a **Vision**, followed by the creation of a high-level **Product Backlog**, and frequently a **Product Roadmap**.

Multilevel Planning in Scrum: Details

<table>
<thead>
<tr>
<th>Level</th>
<th>Horizon</th>
<th>Who</th>
<th>Focus</th>
<th>Deliverables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portfolio</td>
<td>Possibly a year or more</td>
<td>Stakeholders and product owners</td>
<td>Managing a portfolio of products</td>
<td>Portfolio backlog and collection of in-process products</td>
</tr>
<tr>
<td>Product (envisioning)</td>
<td>Up to many months or longer</td>
<td>Product owner, stakeholders</td>
<td>Vision and product evolution over time</td>
<td>Product vision, roadmap, and high-level features</td>
</tr>
<tr>
<td>Release</td>
<td>Three (or fewer) to nine months</td>
<td>Entire Scrum team, stakeholders</td>
<td>Continuously balance customer value and overall quality against the constraints of scope, schedule, and budget</td>
<td>Release plan</td>
</tr>
<tr>
<td>Sprint</td>
<td>Every iteration (one week to one calendar month)</td>
<td>Entire Scrum team</td>
<td>What features to deliver in the next sprint</td>
<td>Sprint goal and sprint backlog</td>
</tr>
<tr>
<td>Daily</td>
<td>Every day</td>
<td>ScrumMaster, development team</td>
<td>How to complete committed features</td>
<td>Inspection of current progress and adaptation of how best to organize the upcoming day’s work</td>
</tr>
</tbody>
</table>

[Rubin 2012]
Planning Hierarchy

[Rubin 2012]
Portfolio Planning: Process

- **Inputs:** Newly Envisioned Products (candidates for inclusion in the portfolio backlog) and In-Process Products.

- **Activities:**
  - **Scheduling:** Determining the proper sequence of the products in the portfolio backlog.
  - **Managing Inflows:** Determining when to insert items into the portfolio backlog (each item is a product to be developed in the future).
  - **Managing Outflows:** Determining when to pull a product out of the portfolio backlog (and start developing it).
  - **Managing In-Process Products:** Determining when to preserve, pivot, deliver, or terminate a product that is currently in process.

- **Outputs:** Portfolio Backlog (prioritized list of future products) and In-Process (Active) Products.
Portfolio Planning: Process

[Rubin 2012]

Department of Computer Engineering
Portfolio Planning: Strategies

Inflows:
- Economic filter
- Arrival rate
- Emergent opportunities
- Smaller, more frequent releases

Portfolio backlog:
- Product A
- Product B
- Product C
- Product D

Outflows:
- Idle work, not idle workers
- WIP limit
- Complete engaged teams

Scheduling:
- Lifecycle profits
- Cost of delay
- Accuracy, not precision

Marginal economics

In-process

[Rubin 2012]
**Product Planning: Process**

- **Inputs:** Initial Idea or Pivoted Idea (for reenvisioning), Planning Horizon (how far into the future should we consider), Expected Completion Date (for envisioning), Budget/Resources (available to conduct envisioning), and Confidence Threshold (the “definition of done” for envisioning).

- **Activities:** Create Vision, Create Product Backlog, Create Roadmap, and Acquire Other Knowledge.

- **Outputs:** Product Vision, Product Roadmap, and High-Level Product Backlog.
Product Planning: Process

[Rubin 2012]
Product Planning: Product Vision

- The product vision provides a clear description of the areas in which the stakeholders, e.g. users and customers, get value from the product.

- Visions, even of complex products, should be simple and should provide a coherent direction to those involved in the project.

- Popular formats:
  - Elevator statement
  - Product datasheet
  - Product vision box
  - User conference slides
  - Press release
  - Magazine review
Product Vision: Areas of Stakeholder Value

- **Entry conditions**
  - Achieve parity with competition
  - Deliver minimum required features
  - Get compliant (SOX, FDA, HIPAA)

- **Enablement**
  - Target a new market
  - Enable sales of other products or services

- **Differentiator**
  - Differentiate from competitors
  - Delight the customer

- **Spoiler**
  - Eliminate competitors’ differentiator
  - Raise the parity bar
  - Redefine the game by changing market focus

- **Cost reducer**
  - Shorten time to market
  - Reduce the number of people or their time allocation
  - Improve margins
  - Increase expertise

[Rubin 2012]
Product Vision Example: Press Release Format

Review Everything, Inc., announced today the successful launch of its new SmartReview4You service. This service provides all online users with their own trainable agent to scour the Internet and identify unbiased, relevant product or service review information.

Remarked Doris Johnson, an avid user of online reviews, “I now have my very own personal assistant that mimics how I find and filter online reviews. It’s amazing—I teach it what I like and don’t like about reviews, then SmartReview4U tears across the Internet finding product or service reviews and automatically weeds out the biased or bogus ones. It does at lightning speed what used to take me forever. This service is a huge timesaver!”

C. J. Rollins, CEO of Review Everything, Inc., said, “We are pleased to offer the world’s first truly smart review service. Since the inception of the Internet people have leveraged the wisdom of the online crowd. However, the crowd can get very noisy at times and it is hard to separate the wheat from the chaff. Our super-smart service does the laborious work of sifting through the huge volume of online review information, eliminating suspicious reviews and returning only relevant ones. You read only the reviews you’d choose to consider if you spent hours searching on your own.”

The new SmartReview4You is available free of charge at the following website: www.smartreview4you.com.

[Rubin 2012]
Product Planning: High-Level Product Backlog

- Consists of a set of initial epics, typically identified during a brainstorming session. Example:

As a typical user I want to teach SR4U what types of reviews to discard so that SR4U will know what characteristics to use when discarding reviews on my behalf.

As a typical user I want a simple, Google-like interface for requesting a review search so that I don’t have to spend much time describing what I want.

As a typical user I want to have SR4U monitor the Internet for new reviews on products or services of interest and automatically filter and report them to me so that I don’t have to keep asking SR4U to do it for me.

As a sophisticated user I want to tell SR4U which sources to use when searching on my behalf so that I don’t get back reviews from sites I don’t like or trust.

As a product vendor I want to be able to show an SR4U-branded review summary for my product on my website so that people can immediately see what the marketplace thinks of my product as determined by a trusted source like SR4U.

[Rubin 2012]
Product Planning: Product Roadmap

- Produced after the initial vision and the high-level product backlog.
- Consists of a series of releases for achieving some or all of the product vision.

<table>
<thead>
<tr>
<th></th>
<th>Q3—Year 1</th>
<th>Q4—Year 1</th>
<th>Q1—Year 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Market map</td>
<td>Initial launch</td>
<td>Better results</td>
<td>Sophisticated users</td>
</tr>
<tr>
<td></td>
<td></td>
<td>More platforms</td>
<td></td>
</tr>
<tr>
<td>Feature/benefit map</td>
<td>Basic learning</td>
<td>Improved learning</td>
<td>Define sources</td>
</tr>
<tr>
<td></td>
<td>Basic filtering</td>
<td>Complex queries</td>
<td>Learn by example</td>
</tr>
<tr>
<td>Architecture map</td>
<td>100K concurrent</td>
<td>iOS</td>
<td>Web services</td>
</tr>
<tr>
<td></td>
<td>web users</td>
<td>and Android</td>
<td>interface</td>
</tr>
<tr>
<td>Market events</td>
<td>Social Media Expo</td>
<td>Review Everything</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>User Conference</td>
<td></td>
</tr>
<tr>
<td>Release schedule</td>
<td>1.0</td>
<td>2.0</td>
<td>3.0</td>
</tr>
</tbody>
</table>

[Rubin 2012]
Product Planning: Guidelines

- Target a realistic confidence threshold
- Focus on a short horizon
- Act quickly
- Pay for validated learning
- Use incremental/provisional funding
- Learn fast and pivot

Guidelines for economically sensible envisioning

[Rubin 2012]
Release Planning: Process

- **Inputs:** Product Vision, High-Level Product Backlog, Product Roadmap, and Velocity (of the team or teams that will work on the release).

- **Activities:** Reviewing Release Constraints (scope, date, and budget), Reviewing Minimum Releasable Features (MRFs of the release), Product Backlog Grooming, and Sprint Mapping (indicating in which sprint some or many of the product backlog items might be created).

- **Output:** Release Plan (showing when we will finish, what features we will get, what the cost will be, what the desired MRFs are for the release, and how some of the product backlog items map to sprints within the release).
Release Planning: Process

[Rubin 2012]

Department of Computer Engineering

Sharif University of Technology
# Fixed-Date Release Planning

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Determine how many sprints are in this release.</td>
<td>If all sprint lengths are equal, this is simple calendar math because you know when the first sprint will start and you know the delivery date.</td>
</tr>
<tr>
<td>2</td>
<td>Groom the product backlog to a sufficient depth by creating, estimating the size of, and prioritizing product backlog items.</td>
<td>Because we are trying to determine which PBI’s we can get by a fixed date, we need enough of them to plan out to that date.</td>
</tr>
<tr>
<td>3</td>
<td>Measure or estimate the team’s velocity as a range.</td>
<td>Determine an average faster and an average slower velocity for the team.</td>
</tr>
<tr>
<td>4</td>
<td>Multiply the slower velocity by the number of sprints. Count down that number of points into the product backlog and draw a line.</td>
<td>This is the “will-have” line.</td>
</tr>
<tr>
<td>5</td>
<td>Multiply the faster velocity by the number of sprints. Count down that number of points into the product backlog and draw a second line.</td>
<td>This is the “might-have” line.</td>
</tr>
</tbody>
</table>

[Rubin 2012]
Fixed-Date Burnup Chart

[Diagram showing a burnup chart with backlog intentionally inverted and fixed ship date is end of sprint 6. The chart categorizes features into must have, might have, won't have, and will have based on sprints.]
Fixed-Scope Release Planning

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Groom the product backlog to include at least the PBI we would like in this release by creating, estimating the size of, and prioritizing PBI.</td>
<td>Because this is a fixed-scope release, we need to know which PBI's are in the fixed scope.</td>
</tr>
<tr>
<td>2</td>
<td>Determine the total size of the PBI's to be delivered in the release.</td>
<td>If we have a product backlog of estimated items, we simply sum the size estimates of all of the items we want in the release.</td>
</tr>
<tr>
<td>3</td>
<td>Measure or estimate the team's velocity as a range.</td>
<td>Determine an average faster and an average slower velocity for the team.</td>
</tr>
<tr>
<td>4</td>
<td>Divide the total size of the PBI's by the faster velocity and round up the answer to the next integer.</td>
<td>This will tell us the lowest number of sprints required to deliver the features.</td>
</tr>
<tr>
<td>5</td>
<td>Divide the total size of the PBI's by the slower velocity and round up the answer to the next integer.</td>
<td>This will tell us the highest number of sprints required to deliver the features.</td>
</tr>
</tbody>
</table>
Fixed-Scope Burndown Chart
Fixed-Scope Burnup Chart

- Target
- Actual
- Average
- High
- Low

Story points vs. Sprints within release

[Rubin 2012]
References
