Software Development*

Process, Models, Methods, Diagrams
Software Development Life Cycles

Part - IV

*from http://www.cs.washington.edu/403/
What is Scrum?

Scrum:

- Is an agile, lightweight process
- Can manage and control software and product development
- Uses iterative, incremental practices
- Has a simple implementation
- Increases productivity
- Reduces time to benefits
- Embraces adaptive, empirical systems development
- Is not restricted to software development projects

- Embraces the opposite of the waterfall approach...
Scrum Origins

• Jeff Sutherland
  – Initial scrums at Easel Corp in 1993
  – IDX and 500+ people doing Scrum

• Ken Schwaber
  – ADM
  – Scrum presented at OOPSLA 96 with Sutherland
  – Author of three books on Scrum

• Mike Beedle
  – Scrum patterns in PLOPD4

• Ken Schwaber and Mike Cohn
  – Co-founded Scrum Alliance in 2002, initially within Agile Alliance
Agile Manifesto

- Individuals and interactions over Process and tools
- Working software over Comprehensive documentation
- Customer collaboration over Contract negotiation
- Responding to change over Following a plan

Source: www.agilemanifesto.org
Project Noise Level

Source: Strategic Management and Organizational Dynamics by Ralph Stacey in Agile Software Development with Scrum by Ken Schwaber and Mike Beedle.
Scrum at a Glance

Daily Scrum Meeting

24 hours

Backlog tasks expanded by team

30 days

Sprint Backlog

Product Backlog

As prioritized by Product Owner

Potentially Shippable Product Increment

Source: Adapted from *Agile Software Development with Scrum* by Ken Schwaber and Mike Beedle.
Sequential vs. Overlap

Rather than doing all of one thing at a time...

...Scrum teams do a little of everything all the time
Scrum Framework

Roles
- Product owner
- Scrum Master
- Team

Ceremonies
- Sprint planning
- Sprint review
- Sprint retrospective
- Daily scrum meeting

Artifacts
- Product backlog
- Sprint backlog
- Burndown charts
Scrum Roles

– Product Owner
  • Possibly a Product Manager or Project Sponsor
  • Decides features, release date, prioritization, $$$

– Scrum Master
  • Typically a Project Manager or Team Leader
  • Responsible for enacting Scrum values and practices
  • Remove impediments / politics, keeps everyone productive

– Project Team
  • 5-10 members; Teams are self-organizing
  • Cross-functional: QA, Programmers, UI Designers, etc.
  • Membership should change only between sprints
"Pigs" and "Chickens"

- **Pig**: Team member committed to success of project
- **Chicken**: Not a pig; interested but not committed

A pig and a chicken are walking down a road. The chicken looks at the pig and says, "Hey, why don't we open a restaurant?" The pig looks back at the chicken and says, "Good idea, what do you want to call it?" The chicken thinks about it and says, "Why don't we call it 'Ham and Eggs'?" "I don't think so," says the pig, "I'd be committed but you'd only be involved."
Sprint Planning Mtg.

Sprint planning meeting

Sprint prioritization
- Analyze/evaluate product backlog
- Select sprint goal

Sprint planning
- Decide how to achieve sprint goal (design)
- Create sprint backlog (tasks) from product backlog items (user stories / features)
- Estimate sprint backlog in hours

Team capacity
Product backlog
Business conditions
Current product
Technology

Sprint goal
Sprint backlog
Daily Scrum Meeting

- **Parameters**
  - Daily, ~15 minutes, Stand-up
  - Anyone late pays a $1 fee

- **Not for problem solving**
  - Whole world is invited
  - Only team members, Scrum Master, product owner, can talk
  - Helps avoid other unnecessary meetings

- **Three questions answered by each team member:**
  1. What did you do yesterday?
  2. What will you do today?
  3. What obstacles are in your way?
Scrum's Artifacts

• Scrum has remarkably few artifacts
  – Product Backlog
  – Sprint Backlog
  – Burndown Charts

• Can be managed using just an Excel spreadsheet
  – More advanced / complicated tools exist:
    • Expensive
    • Web-based – no good for Scrum Master/project manager who travels
    • Still under development
Product Backlog

- The requirements
- A list of all desired work on project
- Ideally expressed as a list of user stories along with "story points", such that each item has value to users or customers of the product
- Prioritized by the product owner
- Reprioritized at start of each sprint
User Stories

• Instead of Use Cases, Agile project owners do "user stories"
  – **Who** (user role) – Is this a customer, employee, admin, etc.?
  – **What** (goal) – What functionality must be achieved/developed?
  – **Why** (reason) – Why does user want to accomplish this goal?

  As a [user role], I want to [goal], so I can [reason].

• Example:
  – "As a user, I want to log in, so I can access subscriber content."

• **story points**: Rating of effort needed to implement this story
  – common scales: 1-10, shirt sizes (XS, S, M, L, XL), etc.
## Sample Product Backlog

<table>
<thead>
<tr>
<th>Backlog item</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow a guest to make a reservation</td>
<td>3 (story points)</td>
</tr>
<tr>
<td>As a guest, I want to cancel a reservation.</td>
<td>5</td>
</tr>
<tr>
<td>As a guest, I want to change the dates of a reservation.</td>
<td>3</td>
</tr>
<tr>
<td>As a hotel employee, I can run RevPAR reports (revenue-per-available-room)</td>
<td>8</td>
</tr>
<tr>
<td>Improve exception handling</td>
<td>8</td>
</tr>
<tr>
<td>...</td>
<td>30</td>
</tr>
<tr>
<td>...</td>
<td>50</td>
</tr>
<tr>
<td>Sprint</td>
<td>ID</td>
</tr>
<tr>
<td>--------</td>
<td>----</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
</tr>
</tbody>
</table>
Sprint Backlog

- Individuals sign up for work of their own choosing
  - Work is never assigned
- Estimated work remaining is updated daily
- Any team member can add, delete change sprint backlog
- Work for the sprint emerges
- If work is unclear, define a sprint backlog item with a larger amount of time and break it down later
- Update work remaining as more becomes known
## Sample Sprint backlog

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code the user interface</td>
<td>8</td>
<td>4</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code the middle tier</td>
<td>16</td>
<td>12</td>
<td>10</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Test the middle tier</td>
<td>8</td>
<td>16</td>
<td>16</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>Write online help</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write the Foo class</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Add error logging</td>
<td></td>
<td></td>
<td>8</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
### Sample Sprint Backlog

#### Sprint 1

01/11/2004

<table>
<thead>
<tr>
<th>Backlog Item</th>
<th>Backlog Item</th>
<th>Owner</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Minor</td>
<td>Remove user kludge in .dpr file</td>
<td>BC</td>
<td>0</td>
</tr>
<tr>
<td>2 Minor</td>
<td>Remove cMap/cMenu/cMenuSize from disciplines.pas</td>
<td>BC</td>
<td>8</td>
</tr>
<tr>
<td>3 Minor</td>
<td>Create &quot;Legacy&quot; discipline node with old civs and E&amp;I content</td>
<td>BC</td>
<td>8</td>
</tr>
<tr>
<td>4 Major</td>
<td>Augment each tbl operation to support network operation</td>
<td>BC</td>
<td>80</td>
</tr>
<tr>
<td>5 Major</td>
<td>Extend Engineering Design estimate items to include summaries</td>
<td>BC</td>
<td>16</td>
</tr>
<tr>
<td>6 Super</td>
<td>Supervision/Guidance</td>
<td>CAM</td>
<td>32</td>
</tr>
</tbody>
</table>

#### 19 days work in this sprint

<table>
<thead>
<tr>
<th>Backlog Item</th>
<th>Backlog Item</th>
<th>Owner</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Minor</td>
<td>Remove user kludge in .dpr file</td>
<td>BC</td>
<td>0</td>
</tr>
<tr>
<td>2 Minor</td>
<td>Remove cMap/cMenu/cMenuSize from disciplines.pas</td>
<td>BC</td>
<td>8</td>
</tr>
<tr>
<td>3 Minor</td>
<td>Create &quot;Legacy&quot; discipline node with old civs and E&amp;I content</td>
<td>BC</td>
<td>8</td>
</tr>
<tr>
<td>4 Major</td>
<td>Augment each tbl operation to support network operation</td>
<td>BC</td>
<td>80</td>
</tr>
<tr>
<td>5 Major</td>
<td>Extend Engineering Design estimate items to include summaries</td>
<td>BC</td>
<td>16</td>
</tr>
<tr>
<td>6 Super</td>
<td>Supervision/Guidance</td>
<td>CAM</td>
<td>32</td>
</tr>
</tbody>
</table>
Sprint Burndown Chart

- A display of what work has been completed and what is left to complete
  - one for each developer or work item
  - updated every day
  - (make best guess about hours/points completed each day)

variation: Release burndown chart
- shows overall progress
- updated at end of each sprint
Sample Burndown Chart
<table>
<thead>
<tr>
<th>Tasks</th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code the user interface</td>
<td>8</td>
<td>4</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code the middle tier</td>
<td>16</td>
<td>12</td>
<td>10</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Test the middle tier</td>
<td>8</td>
<td>16</td>
<td>16</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>Write online help</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12</td>
</tr>
</tbody>
</table>

![Graph showing hours worked per day](image-url)
Burndown Example 1

No work being performed
Burndown Example 2

Work being performed, but not fast enough

Sprint 1 Burndown

Days in Sprint vs Hours remaining
Work being performed, but too fast!
The Sprint Review

- Team presents what it accomplished during the sprint
- Typically takes the form of a demo of new features or underlying architecture
- Informal
  - 2-hour prep time rule
  - No slides
- Whole team participates
- Invite the world
Scalability

- Typical individual team is 7 ± 2 people
  - Scalability comes from teams of teams

- Factors in scaling
  - Type of application
  - Team size
  - Team dispersion
  - Project duration

- Scrum has been used on multiple 500+ person projects
### Scrum vs. Other Models

#### Process Comparison

<table>
<thead>
<tr>
<th>Defined processes</th>
<th>Waterfall</th>
<th>Spiral</th>
<th>Iterative</th>
<th>SCRUM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Required</td>
<td>Required</td>
<td>Required</td>
<td>Planning &amp; Closure only</td>
</tr>
<tr>
<td>Final product</td>
<td>Determined during planning</td>
<td>Determined during planning</td>
<td>Set during project</td>
<td>Set during project</td>
</tr>
<tr>
<td>Project cost</td>
<td>Determined during planning</td>
<td>Partially variable</td>
<td>Set during project</td>
<td>Set during project</td>
</tr>
<tr>
<td>Completion date</td>
<td>Determined during planning</td>
<td>Partially variable</td>
<td>Set during project</td>
<td>Set during project</td>
</tr>
<tr>
<td>Responsiveness to environment</td>
<td>Planning only</td>
<td>Planning primarily</td>
<td>At end of each iteration</td>
<td>Throughout</td>
</tr>
<tr>
<td>Team flexibility, creativity</td>
<td>Limited - cookbook approach</td>
<td>Limited - cookbook approach</td>
<td>Limited - cookbook approach</td>
<td>Unlimited during iterations</td>
</tr>
<tr>
<td>Knowledge transfer</td>
<td>Training prior to project</td>
<td>Training prior to project</td>
<td>Training prior to project</td>
<td>Teamwork during project</td>
</tr>
<tr>
<td>Probability of success</td>
<td>Low</td>
<td>Medium Low</td>
<td>Medium</td>
<td>High</td>
</tr>
</tbody>
</table>