A cornucopia of agile processes

**Agile Processes**
- Extreme Programming (XP)
- Scrum
- Crystal
- DSDM
- Lean software development
- Unbranded “agile”

**Semi-Agile Processes**
- Feature-Driven Development (FDD)
- Unified Process
- OpenUP

© 2003–2008 Mountain Goat Software®
A closer look at Scrum

• No specific engineering practices prescribed
  • But many Scrum teams are adopting much of XP
• 2- to 4-week iterations called “sprints”
• Self-organizing, cross-functional teams
• Uses generative rules to create an agile environment

The Scrum project community

Programmer Programmer
Analyst DBA
Tester Programmer
Tester User Experience Designer

Scrum Master
Process coach and Sheepdog for the team

Product Owner
Defines what to build and in what order
Scrum

- Sprint goal
- Return

Sprint backlog

- Cancel
- Coupons
- Gift wrap

Product backlog

24 hours

Sprint
2-4 weeks

Potentially shippable product increment

Seven Sins of Project Management

1. Gluttony
2. Lust
3. Sloth
4. Opaqueness
5. Pride
6. Wastefulness
7. Myopia
Sin #1: Gluttony

- Definition
  - Fixing all dimensions (scope, schedule, resources, and quality) of a project
  - A project management sin of excess
- Experienced as
  - Impossible schedules
  - Death marches
- Leads to
  - Trying to do too much for the resources (time, people) available
  - Cutting quality to meet other goals

The iron triangle

Which dimension is best to adjust to achieve project success?
Timeboxes help avoid gluttony

- Agile iterations are timeboxed
  - So the schedule of an iteration is fixed
  - But the number of iterations is variable
  - Focus is always on “what can we accomplish next?”

Timeboxed iterations increase predictability

- Over time a team learns how much it can complete per iteration (its “velocity”)
  - Knowing this prevents the temptation to overcommit
What expectation of future velocity should this team set?

<table>
<thead>
<tr>
<th>Feature</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature A</td>
<td>5</td>
</tr>
<tr>
<td>Feature B</td>
<td>3</td>
</tr>
<tr>
<td>Feature C</td>
<td>5</td>
</tr>
<tr>
<td>Feature D</td>
<td>5</td>
</tr>
<tr>
<td>Feature E</td>
<td>5</td>
</tr>
<tr>
<td>Feature G</td>
<td>3</td>
</tr>
<tr>
<td>Feature I</td>
<td>3</td>
</tr>
<tr>
<td>Feature H</td>
<td>5</td>
</tr>
<tr>
<td>Feature J</td>
<td>2</td>
</tr>
<tr>
<td>Feature K</td>
<td>5</td>
</tr>
<tr>
<td>Feature L</td>
<td>3</td>
</tr>
</tbody>
</table>

Sin #2: Lust

- **Definition**
  - Intense or unrestrained craving for features

- **Experienced as**
  - Trying to put too many features into a product during the time allowed
  - Treating all features as “critical”

- **Leads to**
  - Overtime, reduced quality, surprises
Three ways agile deals with lust

1. Developing features in priority order

2. Incremental gratification

3. Working at a sustainable pace

“Overtime is a symptom of a serious problem on the project. The XP rule is simple—you can’t work a second week of overtime. For one week, fine, crank and put in some extra hours. If you come in on Monday and say ‘To meet our goals, we’ll have to work late again,’ then you already have a problem that can’t be solved by working more hours.”

~Kent Beck
Old habits die hard

Sin #3: Sloth

- **Definition**
  - Failing to do high quality work at all times

- **Experienced as**
  - Testing quality in at the end
  - Instability during development

- **Leads to**
  - Big delays
  - Unpredictable schedules
Agile practices related to quality

- Simple design
- Automated testing
- Test-driven development
- Continuous integration
- Pair programming
- Refactoring

Case Study 1

Cosmodemonic Biotech

<table>
<thead>
<tr>
<th></th>
<th>Waterfall</th>
<th>Scrum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Case pages</td>
<td>3,000</td>
<td>1,400</td>
</tr>
<tr>
<td>User Stories</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calendar months</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>Person months</td>
<td>540</td>
<td>54</td>
</tr>
<tr>
<td>Lines of Java code</td>
<td>58,000</td>
<td>51,000</td>
</tr>
<tr>
<td>Lines of Java code per person-month</td>
<td>120</td>
<td>840</td>
</tr>
</tbody>
</table>
Case Study 2

**ePS**

**Productivity (NCSS / month)**

- US average: 270
- Three years prior to introducing agile: 389
- First nine months after starting agile: 1206

NCSS = Non-Comment Source Statement (Java)

**Defects per KNCSS**

- Three years prior to introducing agile: 10
- First nine months after starting agile: 2.9

But wait, there’s more...

- Results achieved without any targeted rewrite of existing (buggy) code
- Many of the post-agile defects continued to be in the old code
- True results would be even better (if we had measured them)
Sin #4: Opaqueness

• Definition
  • Obscuring the progress, quality or other attribute of a project

• Experienced as
  • Not knowing the true state of the project

• Leads to
  • Surprises
  • Poor decisions

Three types of opaqueness

1. How agile addresses Quality opaqueness

• Don’t let bugs accumulate
• Continuous integration
• Features are either “Done” or “Not Done”
  • Avoids the 90% Syndrome
2. How agile addresses Schedule opaqueness

A release burndown chart

![Burndown chart with story points over iterations]

Mean (Worst 3) = 28
Mean (Last 8) = 33
Mean (Best 3) = 37

3. How agile addresses Scope opaqueness

![Bar chart with velocity over iterations]

Mean (Best 3) = 37
Mean (Last 8) = 33
Mean (Worst 3) = 28
Predicting what will be in

Assume:
There are five sprints left.

At our slowest velocity we’ll finish here (5×28)

At our long-term average we’ll finish here (5×33)

At our best velocity we’ll finish here (5×37)

A Backlog Burndown Chart

Returns
Gift Wrap
Returns
Exchanges
Cart
... Coupons

© 2003–2008 Mountain Goat Software®
Sin #5: Pride

- Definition
  - Believing that we know everything to build the product
- Experienced as
  - A lack of stakeholder and user involvement
- Leads to
  - Failure to solicit feedback
  - Failure to learn

Where are opportunities for feedback?

Sprint goal

Return

Cancel

Gift wrap

Coupons

Product backlog

24 hours

Sprint 2-4 weeks

Potentially shippable product increment
Agile requirements

“User stories” facilitate working closely with users & customers

As a user, I want to reserve a hotel room.
As a vacation traveler, I want to see photos of the hotels so that I can choose the best one for me.
As a user, I want the site to be available 99.999% of the time I try to access it.

Progressive refinement

As a VP Marketing, I want to review the performance of historical promotional campaigns so that I can identify and repeat profitable ones.

As a VP Marketing, I want to select the timeframe to use when reviewing the performance of past promotional campaigns, so that I can identify and repeat profitable ones.

As a VP Marketing, I can select which type of campaigns (direct mail, TV, email, radio, etc.) to include when reviewing the performance of historical promotional campaigns.
An example

As a VP Marketing, I want to see information on **direct mailings** when reviewing historical campaigns.

As a VP Marketing, I want to see information on **television advertising** when reviewing historical campaigns.

As a VP Marketing, I want to see information on **email advertising** when reviewing historical campaigns.

---

**Sin #6: Wastefulness**

- **Definition**
  - Misuse of critical resources

- **Experienced as**
  - Losses of creativity, motivation, and time

- **Leads to**
  - Project malaise
  - Delays
  - Doing it the same way (again)
How agile handles waste

1. Timeboxing
2. Daily standups
3. Iteration retrospectives
4. Self-organizing teams

5. Spreading intensity evenly

© 2003–2008 Mountain Goat Software®
Sin #7: Myopia (Shortsightedness)

- **Definition**
  - Not seeing beyond your own work
- **Experienced as**
  - Teams who don’t see the big picture
  - Individuals who work only within their roles
- **Leads to**
  - Unsuccessful products
  - Delays
Seeing the forest and the trees at the same time

**Release Plan**

- **Sprint 1**
- **Sprint 2**
- **Sprint 3**
- **Sprints 4–7**

<table>
<thead>
<tr>
<th>Task</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task A</td>
<td>8 hours</td>
</tr>
<tr>
<td>Task B</td>
<td>16 hours</td>
</tr>
<tr>
<td>Task C</td>
<td>5 hours</td>
</tr>
<tr>
<td>Task D</td>
<td>8 hours</td>
</tr>
</tbody>
</table>

Iteration Plan

Cross-functional team

- All disciplines necessary to go from idea to implementation
- Improves creativity and ownership
- Whole team commitment

But does that make everyone a generalist?
Is everyone a generalist?

Meconi’s

Bruno’s

Ferentino’s

Key
- Order-taker
- Floater
- Sandwich-maker

Screw that; where’s lunch?

Mike Cohn
mike@mountaingoatsoftware.com
www.mountaingoatsoftware.com
(720) 890–6110 (office)
(303) 810–2190 (mobile)