Harvesting Mob Programming Patterns: Observing How We Work

Michael Keeling
LendingHome
@michaelkeeling

Joe Runde
IBM
@joerunde
This is a tale of two teams...
Key Challenges

- Team was new – most senior person had been at LendingHome for 9 months
- Inexperienced in the mortgage domain
- Large codebase with pockets of technical debt

- Large legacy codebase to refactor
- Tons of DevOps work
- Two teams merged together with different work styles and expectations
<table>
<thead>
<tr>
<th></th>
<th>Platform PGH Team</th>
<th>Doc Ranker Team</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>FinTech</td>
<td>Search + Machine Learning</td>
</tr>
<tr>
<td>Technologies</td>
<td>Ruby on Rails web app</td>
<td>Java, Go, microservices</td>
</tr>
<tr>
<td>Team Size</td>
<td>9 Engineers</td>
<td>9 Engineers</td>
</tr>
<tr>
<td>Location</td>
<td>Co-Located in Pittsburgh</td>
<td>Partially Distributed</td>
</tr>
<tr>
<td>Average Experience</td>
<td>7+ years</td>
<td>3+ years</td>
</tr>
</tbody>
</table>
Mob Programming worked pretty well on my previous team... maybe it could help with my new team?
MOB PROGRAMMING 101
“Mob programming is a software development approach where the whole team works on the same thing, at the same time, in the same space, and at the same computer.”

Pair Programming
Mob Programming Advantages

• Reduce communication and management overhead
• Bust knowledge silos
• Decrease task thrashing
• Improve team alignment
• Involve the whole team
• Promote self-organization
Worries...

“They’ll find out that I don’t know any terminal shortcuts and laugh at me.”

“I’m really bad at spelling and everybody will see.”

“I don’t know the domain and I won’t be able to keep up.”
Complaints...

“There’s no way we can keep up our velocity, we’ll miss all our deadlines!”

“I could do this so much faster by myself if I didn’t have to explain everything to the group!”
“So, we’re going to sit around in a circle and... program?”
Woody Zuill, *Mob Programming – A Whole Team Approach, Agile 2014*
A recent mobbing session at LendingHome...
The Core Principle of Mob Programming

Treat each other with kindness, consideration, and respect.
There is no “one right way” to mob program.
But there are plenty of wrong ways...
PATTERNS:
A BRIEF INTERLUDE
“Each pattern describes a problem that occurs over and over again in our environment and then describes the core of the solution to that problem in such a way that you can use this solution a million times over without ever doing it the same way twice.”

Christopher Alexander, A Pattern Language, 1977
Software without patterns is like a big pile of spaghetti:
Patterns help organize software and communicate intent using shared human experience.
What if we also used patterns to describe the way that we work together to write code?
Patterns from the Agile2014 Experience Report

• Driver and navigator roles
• Switch drivers using a timer
• Driver rotation list
• Mob retrospective
• Lots of other great ideas...
PATTERN HARVESTING (A/K/A AFTER HOURS SOCIALIZING)
Comparing stories is a fantastic way to uncover patterns...
“Oh boy my mob went off the rails today. I left for an hour and they were building things in left field!”
“Huh, sometimes my teammates switch tasks every time they have a new idea. It can be a mess too.”
“I've started drawing pictures to keep us all on the same page. What are you doing?”
“Carol started keeping a list of all the ideas so others could stay focused. It's working really well.”
“Oh, like a punch list! I'm gonna add that below my pictures.”
We do a lot of different things in our team. It can be difficult to tell when something is interesting or mundane.

Solution

Share a story about how your team works with a friend not on the team.

Consequences

Outside perspective can help put ways of working in broader context. The person listening can ask probing questions and "play back" your experiences.

A Basic Pattern Outline

- **Context**: Describes the problem and general state of the world.
- **Solution**: Describes how you’ll solve the problem.
- **Consequences**: Describes how the world changes as a result of applying the solution.
We do a lot of different things in our team. It can be difficult to tell when something is interesting or mundane.

**Solution**

Share a story about how your team works with a friend not on the team.

**Consequences**

Outside perspective can help put ways of working in broader context. The person listening can ask probing questions and "play back" your experiences.

**How do I know when to use this pattern?**

**What should I do?**

**What are the benefits?**

**What problems might I encounter when applying this solution?**
Pattern Harvesting Meta-Pattern: Share Stories

**Context**  We do a lot of different things in our team. It can be difficult to tell when something is interesting or mundane.

**Solution**  What should I do?

**Consequences**  What are the benefits? What problems might I encounter when applying this solution?
Pattern Harvesting Meta-Pattern: Share Stories

**Context** We do a lot of different things in our team. It can be difficult to tell when something is interesting or mundane.

**Solution** Share a story about how your team works with a friend not on the team.

**Consequences**

- What are the benefits?
- What problems might I encounter when applying this solution?
Pattern Harvesting Meta-Pattern: Share Stories

**Context**  We do a lot of different things in our team. It can be difficult to tell when something is interesting or mundane.

**Solution**  Share a stories about how your team works with a friend not on the team.

**Consequences**  Outside perspective can help put ways of working in broader context. The person listening can ask probing questions and “play back” your experiences.
What did we learn about how to be a better mob?
OUR MOB PROGRAMMING PATTERNS
Insight #1

Mob participants take on different roles during a mob programming session.
<table>
<thead>
<tr>
<th>Pattern</th>
<th>Gist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Facilitator</td>
<td>Volunteer who helps the group stay focused and to help resolve differences of opinion. Typically the driver does not take this role.</td>
</tr>
<tr>
<td>Recorder</td>
<td>Volunteers who capture notes such as context or design decisions on behalf of the mob.</td>
</tr>
<tr>
<td>Researcher</td>
<td>Volunteers who seek out information in real-time that is required for the mob to move forward.</td>
</tr>
<tr>
<td>Navigator</td>
<td>Direct the driver in what to do. Members of the mob not currently driving are assumed to be a navigator. Navigators can contribute to the mob in many ways.</td>
</tr>
<tr>
<td>Driver</td>
<td>The person currently at the keyboard, capturing the thoughts of the navigators as everyone works to solve a particular problem.</td>
</tr>
<tr>
<td>Devil’s Advocate</td>
<td>A navigator who takes a contrarian position to help make the mob’s designs stronger.</td>
</tr>
</tbody>
</table>
Pattern: Recorder (a/k/a Archivist)

**Context**  Many ideas are discussed during a mob, often people will think out loud about an idea, an execution-focused mob can easily forget great ideas.

**Solution**  Assign someone to write stuff down.

**Consequences**  Focus mob participants on the present. Discussions are captured for later use. Artifacts are available to share with people who were not in the mob (e.g. ADRs).
Recorder: Tips and Hints

• Kick this pattern off by setting a good example
• Create shared document (e.g. DropBox)
• Create a focal point for distributed mobs
• Draw pictures!
• Don’t be afraid to assign a volunteer
• Pay attention when a decision is made from multiple options
Effective collaboration is one of the greatest barriers to effective mob programming.
Nearly half the patterns we identified are directly related to how teammates work together.
Collaboration Patterns

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Gist</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Punch List</strong></td>
<td>A task list used by the mob to track work items and select what to work on next. Punch lists can be text-based or graphical.</td>
</tr>
<tr>
<td><strong>Splinter Group</strong></td>
<td>One or more members of the mob who break away from the main group to complete a routine task. Splinter groups can also investigate alternatives for review by the whole mob.</td>
</tr>
<tr>
<td><strong>Ridin’ Shotgun</strong></td>
<td>A navigator who solely dictates the mob’s work to the exclusion of other navigators.</td>
</tr>
<tr>
<td><strong>Mute your mic</strong></td>
<td>A navigator chooses to temporarily remain silent as a navigator to give other navigators a chance to contribute. Used as a way to kick start a slow mob or prevent one person from dominating the mob.</td>
</tr>
<tr>
<td><strong>Fight Club</strong></td>
<td>A situation in which two or more participants fight over the direction of the mob with total disregard for the guiding principles of mutual respect and consideration. Extremely harmful to the mob, considered an anti-pattern.</td>
</tr>
<tr>
<td><strong>Natural Swap</strong></td>
<td>A new driver takes the keyboard without prompting by either a member of the mob or timer at a “natural” break in the work, such as after a test passes or a refactoring step is completed.</td>
</tr>
<tr>
<td><strong>Forced Swap</strong></td>
<td>A new driver takes the keyboard after being prompted by either a member of the mob or a timer.</td>
</tr>
<tr>
<td><strong>Distracted non-Participant</strong></td>
<td>Navigators who are present in the mob but otherwise do not participate, perhaps distracted by other work.</td>
</tr>
</tbody>
</table>
Pattern: Splinter Group

**Context**  Multiple options present themselves to the mob, some individuals express strong interest in a specific task, routine work creates a bottleneck to “meaningful” work.

**Solution**  Break off a small splinter group to work on something independently of the full mob.

**Consequences**  Mob focus can be directed toward activities that are likely to most benefit from collective input. Individuals can dive into areas of interest. The mob breaks slightly but intentionally.
• Splinter groups can be individuals, pairs, or new mobs.
• Pay attention when side conversations start happening. Does it need the mob’s attention or is a splinter group forming?
• It's OK to for people to split off from the mob!
• Great for deep investigations and debugging
Insight #3

Driving style has strong influence over how the mob operates.
## Driver Patterns

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Gist</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thinking Out Loud</td>
<td>The driver articulates their current thinking as they are the prevailing expert in the room or see the path forward.</td>
</tr>
<tr>
<td>Tell me what to write</td>
<td>A prompt drivers will sometimes use to engage help from navigators, inviting someone from the mob to direct the driver.</td>
</tr>
<tr>
<td>Driving on Autopilot</td>
<td>A driver who proceeds without inputs from the rest of the mob.</td>
</tr>
<tr>
<td>Plowing Through</td>
<td>A driver who, with the support of the mob, works on the task at hand with the intent of completing it as quickly and painlessly as possible. Often combined with the <em>thinking out loud</em> pattern.</td>
</tr>
</tbody>
</table>
Pattern: Thinking Out Loud

**Context**  The current driver holds most of the knowledge required for the task at hand.

**Solution**  Have the driver talk out what they are doing as they code.

**Consequences**  Allows the mob to move forward while still sharing knowledge. The mob can still direct general direction.

Alternative: "Tell me what to write" pattern
Pattern: Tell me what to write

**Context** The driver is new to a framework, language, or domain.

**Solution** The navigator(s) give detailed instructions, sometimes at the keystroke level.

**Consequences** The mob moves slower but knowledge sharing is amplified. There is a risk that participants may tune out if one navigator dominates the mob (see "ridin' shotgun" pattern)
Pattern: Plowing Through

**Context**  The mob is working through a frustrating task that requires tacit knowledge held by only the driver

**Solution**  The mob lets the driver plow through a solution as fast as possible

**Consequences**  The mob moves faster to a working solution, but less knowledge is transferred and participants may disengage
Anti-Pattern: Driving on Autopilot

**Context** The mob isn't providing focused input to the driver, or the driver forgot the mob was there.

**Disruptive Behavior** The driver writes code while the group watches silently.

**Consequences** The feedback loop is disrupted. Code review and knowledge sharing benefits of mobbing aren't realized.
WRAP-UP
“We didn't set out to do mob programming. We set out to create an environment where good things can happen.”

– Woody Zuill,
Yesterday @ Agile2019
Mob programming is an excellent tool for amplifying awesomeness.
It takes practice to find the mobbing styles that work for your team and context.
Silver Toolbox
What cool mob programming patterns have you discovered?

https://github.com/michaelkeeling/mob-programming-patterns
https://github.com/Austin117/MobPatterns
https://github.com/willemlarsen/mobprogrammingrpg
Mob Mentality Show on YouTube
Pattern Harvesting Meta-Pattern: Share Stories

**Context**  We do a lot of different things in our team. It can be difficult to tell when something is interesting or mundane.

**Solution**  Share stories about how your team works with a friend not on the team.

**Consequences**  Outside perspective can help put ways of working in broader context. The person listening can ask probing questions and “play back” your experiences.
Share Stories: Tips and Hints

• Hit the bar (or some other low stress locale)
• Share something – anything!
  • Problems work great, whether you have an answer yet or not
• Ask open questions, probe for details
  • Help the story teller be specific
• Name it
• Write it down
Get 25% off with the code Agile2019_Keeling at https://pragprog.com/book/mkdsa/design-it

Also available at the Agile2019 Book Store
Thank you!

Michael Keeling
@michaelkeeling
neverletdown.net

Joe Runde
@joerunde