Automating Continuous Planning in SAFe

Darius Foo, Jonah Dela Cruz, Subashree Sekar, Asankhaya Sharma
Veracode
Agenda

• Motivation
• Sapling
• Demo
• Future work
Agile teams

single site  multi-site  satellite workers  remote-first

Source: Martin Fowler, http://martinfowler.com/articles/remote-or-co-located.html
PI Planning

- Meet face-to-face
- Align goals
- Cross-team dependencies
- Match demand to capacity
Limitations

- Visibility across geos
- Cross-geo dependencies
- Manual digitization
Continuous Planning

- Similar in spirit to CI/CD
- Global visibility into a continuously-updated plan
- Automate away overhead
- Transparency and accessibility
Sapling

A tool for Continuous Planning

Special support for SAFe PI Planning
Teams begin with a prioritized list of epics

Time is divided into sprints

Epics are divided into user stories and assigned to sprints

Stories are assigned story points

Sprint capacities should be respected
Story dependencies

• Within teams
  • e.g. “Research” item should always precede an “Implementation” item

• Across teams
  • e.g. platform team should finish API before frontend team uses it
  • Requires communication

• PI Planning is all about surfacing and negotiating cross-team dependencies
## Enabling Continuous Planning

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sapling supports the workflow</td>
<td>Allowing users to edit stories, sprints, epics, and boards</td>
</tr>
<tr>
<td>Stories are assigned to sprints automatically</td>
<td>So the plan is always up to date</td>
</tr>
<tr>
<td>Teams can request to add stories to each other’s boards</td>
<td></td>
</tr>
<tr>
<td>Dependencies across the entire organization are always visible</td>
<td></td>
</tr>
<tr>
<td>JIRA import/export</td>
<td></td>
</tr>
</tbody>
</table>
Story Assignments

• Hard constraints (integrity constraints):
  • Sprint capacity
  • Dependencies
  • Pins

• Soft constraints (optimization):
  • Epic priority: minimize stories in lower-priority epics before higher ones
  • Context: minimize distance between dependent stories
  • Parallelism: maximize number of people working at the same time
  • Conservativeness: maximize earlier sprint assignments over later ones
Story Requests

• Structures the process of negotiating dependencies
Story Requests

• Visualization of cross-team dependencies
Demo
Future work

• Stability and explainability (UX) of solver decisions
• Previews
• Incremental solving (allowing non-optimal solutions)
• Deeper JIRA integration
Thanks!

https://github.com/srcclr/sapling