Balancing Dev Agility With your IT Ops Mission Using Kanban

Dominica DeGrandis
@Dominicad
Cycle of Frustration

THINGS TAKE TOO LONG!
THERE’S NO VISIBILITY

TOO MUCH WORK!
CONFLICTING PRIORITIES
When everything is a priority 1
Context switching is expensive.

And irritating!

because …

• Multitasking “stop & go” breaks the on-a-roll flow.
• Churn and rework create waste.
Kanban practices

1. Make work visible
Bringing visibility to Ops work

because …

- It’s hard to manage invisible work.
- Humans absorb info faster visually than any other sense.
- Visualization helps us see patterns & connections that matter.

![Diagram showing project management with categories like Xpedites, Project, Misc., CODB, and status stages like On Deck, Prep doing, Prep done, Implement, Validate, Closed. Each category is color-coded and has icons representing different projects and statuses.]

- Incident (live issue, security breach)
- Project A
- Project B
- Project C
- CODB
When the front end gets fixed without addressing the backend.
Kanban practices

1. Make work visible
2. Limit work-in-progress
### Bringing limits to amount of work in play

<table>
<thead>
<tr>
<th>Incident (live issue, security breach)</th>
<th>Project A</th>
<th>Project B</th>
<th>Project C</th>
<th>Misc.</th>
<th>On Deck</th>
<th>Prep doing</th>
<th>Prep done</th>
<th>Implement</th>
<th>Validate</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xpedites</td>
<td>3</td>
<td>6</td>
<td></td>
<td></td>
<td>4</td>
<td>8</td>
<td>10</td>
<td>5</td>
<td>96</td>
<td>3</td>
</tr>
</tbody>
</table>

because …

- Knowledge work is perishable.
- Churn and rework create waste.
- Increased cycle time increases cost.
Time weighs heavily on heroes

"UNBALANCED"
11/09

I CAN’T HOLD ON MUCH LONGER
Kanban practices

1. Make work visible
2. Limit work-in-progress
3. Make process policies explicit
Rules contribute to flow
Infrastructure contributes to congestion
Consider options that allow for improved flow

Roundabouts decrease “stop & go” increasing throughput by 20%. What rules would increase Ops throughput?
# Ops workflow sample rules

<table>
<thead>
<tr>
<th>Backlog</th>
<th>On Deck</th>
<th>Prep doing</th>
<th>Prep done</th>
<th>Implement</th>
<th>Validate</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xpedites</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Misc</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Items in backlog remain uncommitted and un-prioritized until there is capacity to absorb them.**
- **Prioritized work limited to available capacity**
- **done**
  - InfoSec
  - Arch
  - CAB

Auto closed > 5 days

Address at Ops Review
Calculating **Cost of Delay**

Consists of the sum of –

- Missed Revenue
  - New revenue
  - Protecting Existing revenue
- Extra costs
  - Running costs (dev staff, ops staff, ops cost)
  - Fines and penalties (regularity, contractual)
- **Sum of Cost of Delay of Other Projects (often missed)**

Kanban practices

1. Make work visible
2. Limit work-in-progress
3. Make process policies explicit
4. Measure flow
because ...

- Historical cycle time provides ability to set customer expectations.
- Discovering where and why work gets bottlenecked is useful.
- Looking at trends over time reveals improvement (or not).
How long has work been idle?

![Bar chart showing the number of tickets for different periods of work sitting in implementation.](chart.png)
Rate of incoming vs. outgoing

Issues: 1483 created and 1413 resolved
Period: last 90 days (grouped Daily)
Measure for objectivity

Time spent using Tupperware

- Storing fresh food
- Storing rotten food
- Searching for the right $@#$%ing lid
Kanban practices

1. Make work visible
2. Limit work-in-progress
3. Measure flow
4. Make process policies explicit
5. Develop feedback loops
Opportunities to hear and to be heard

Lean Coffee
http://leancoffee.org/
Operations Review

Reflect on quantitative performance measures
Quantitative performance measures

• WIP (work-in-progress)
• Cycle time/Lead time
• Throughput
• Quality
• Issues & blocked work

Across:

Trend & Variability
Kanban practices

1. Make work visible
2. Limit work-in-progress
3. Measure flow
4. Make process policies explicit
5. Develop feedback loops
6. Evolve Experimentally using models & theory
Experiment with applicable models & theory
No.
Balance Dev Agility with your IT Ops Mission Using *Kanban*

www.ddegrandis.com
*Kanban for Devops & IT Ops*
dominica@ddegrandis.com

Dominica DeGrandis
@Dominicad