Building an Agile Release Plan Using Forecasting

Chris Shinkle
Director of Innovation
@chrissshinkle | chrissshinkle@sep.com
The session has started...

1. Consider your last project plan or release plan...

2. Find someone to pair with.

3. Share how accurate the plan was. Did you deliver all the features as scheduled?
WHAT’S THE DIFFERENCE?

ROADMAP vs RELEASE PLAN

@chrisshinkle
Person who's a potential customer + Your product = Awesome person who can do rad shit!
ROADMAP
MANAGES
OUTCOMES

RELEASE PLAN
MANAGES
OUTPUTS

@chrisshinkle
Agile Planning Onion

We struggle here

VISION & STRATEGY
ROADMAP
RELEASE
ITERATION
DAILY

High-level Strategic Planning
Release Planning
Short-term Planning

@chrisshinkle
COMPONENTS OF A RELEASE PLAN

1. **WHAT** features, epics, or stories to deliver.
2. **WHEN** the Product Backlog Items will be delivered.

@chrisshinkle
WHAT TO DELIVER

RELEASE PLAN COMPONENTS

@chrisshinkle
TO PRIORITIZE FEATURES YOU
ALIGN WITH YOUR OUTCOMES
@chrisshinkle
Feature Prioritization Techniques

1. Kano Model

2. Buy a Feature

3. Story Map

4. Outcome Driven Innovation (ODI)

5. MoSCoW

6. Outcomes -> Personas -> Stories
Better Backlog Prioritization (from random to lifetime cost of delay)

The problem with any prioritization decision is that it is a decision to service one job and delay another.

**The Goal**
- what to start next

Given a set of things we could do next, is one more economically advantageous to start first.

**The Challenge**
- Doing one thing delays others

Every item has a different economic impact by being delayed. The impact will be a balance of lost value, and how long they are delayed.

**The basic concept**
- balance $ & time

$Value lost due to delay$ = delay duration

Same duration, do > lost $ first

Same lost $, do < duration first

**Harder to decide**
- when both value of what NOT done and duration change

Small to medium impact decisions
- Gets close when the impact of being wrong is moderate

Medium to large impact decisions
- Start understanding the total cost of delay (products and major features)

Random choice

Crystal Ball

HiPPO
- Just make a choice

Highest Value First
- Common in Scrum. Some attempt as estimating “value”

Relative Value & Effort (delay) Matrix
- Balance duration and value

SAFe “Arnold”
- Make urgency dominate

SAFe (real estimates)
- Normalize & weight values against $ and time range scales

Cost of Delay Curve Categorization
- (e.g. Kanban’s class of service)

Economic Impact of delay over a product’s entire lifespan is modeled (long term value)

SAFe and SAFe Arnold Mod

SAFe (Scaled Agile Framework) uses a weighted shortest job first balancing technique. It uses subjective measures and approximates optimal starting order using the following formula (highest first):

$Value + Criticality + Risk Reduction or Opportunity Enablement$ / $Job Size$

Joshua Arnold offers the following modification to make time criticality more dominant:

$Criticality \times (Value + Risk Reduction or Opportunity Enablement)$ / $Job Size$

__Economic Models and WSJF__

Donald Reinertsen in his book *Principles of Product Development Flow* offers a variety of scheduling techniques. The most popular is Weighted Shortest Job First where optimal starting order is calculated using delay impact in dollars and size. Optimal order (highest to lowest) is calculated using the formula:

$Cost of delay$ / $Duration of delay$

Reinertsen suggests it’s prudent to consider the total market impact of a delay, not just the immediate lost value.

Jointly designed in an online conversation by Martin Burns, Don Reinertsen, Chris Matts, Joshua Arnold, Tony Grout and Troy Magennis sometime during 2016. It is, and will remain a work in progress. Trademarks used NOT ours.

__Chance of making a sub-optimal decision__

- Troy Magennis

__Effort required to get optimal decision__

Note: a naïve assumption is that good analysis is performed in every technique shown here. It’s possible to do all of these techniques poorly and make poor decisions. Use care!
DATES OFTEN DRIVE DECISIONS

REGARDLESS OF CONTENT…

@chrisshinkle
WHEN CAN WE DELIVER

@chrisshinkle
How long = \frac{Size}{Velocity}

distance and pace model

@chrisshinkle
Sprint = 2 weeks

15 pts per sprint

300 story pts

How many weeks =

15 pts per sprint

300 story pts
How many weeks? = \[
\frac{300 \text{ story pts}}{[10-20] \text{ pts per sprint}}
\]

Sprint = 2 weeks
How many weeks? = $\frac{[280-320] \text{ story pts}}{[10-20] \text{ pts per sprint}}$

Sprint = 2 weeks

@chrisshinkle
How many weeks?

\[
\begin{align*}
\text{Sprint} &= \frac{[280-320]}{[10-20]} \times [1 - 2] \\
\text{Sprint} &= 2 \text{ weeks}
\end{align*}
\]
How many weeks?

\[
\frac{[194-251]}{[13-29]} \times [1 -1.75] \times \ldots
\]

Sprint = 2 weeks

@chrissinkle
HOW MANY PEOPLE USE...

VELOCITY AND STORY POINTS?

@chrisshinkle
Story Point Manipulation

One day in 2004 Jim exhorted the team to go faster. This team had an average velocity around 52 points per iteration. Their velocity would fluctuate by a few points, but generally remained steady. Yet just weeks after Jim asked the team to "sprint", the team's velocity jumped up into the high 80s!

I asked someone what happened…
Story Point Manipulation

She looked at me funny and said, "These days around here if you sneeze, you get a story point." I shook my head, amazed at how a mature agile team could so suddenly inflate their story point estimates to appear like they were going faster.

My confidence in story points and velocity calculations began to erode after that experience.

- Joshua Kerievsky
Teams Struggle with Story Points

Many teams struggle with story points. I have certainly seen teams devalue those points when their project managers try to get something for nothing by asking the team to go faster. I've also seen teams fake their velocity charts for the benefit of project managers who were more interested in form over function. For such teams, a different approach may be warranted.

- Uncle Bob Martin

@chrisshinkle
Velocity is not Value

Velocity can be misused, and usually is.
Velocity can be gamed, and usually is.
A team that is focusing on velocity is not focusing on value. **I wish I had never invented velocity**, if in fact I did.

- Ron Jefferies
Planning Fallacy

Executives tend to “make decisions based on delusional optimism rather than on a rational weighing of gains, losses, and **probabilities**. They overestimate benefits and underestimate costs. They spin scenarios of success while **overlooking the potential mistakes and miscalculations**. As a result, they pursue initiatives that are unlikely to come in on budget or on time or to deliver the expected returns – or even to be completed.”

- Daniel Kahneman
FORECASTING
Practical Metrics for Agile Teams

Wednesday, June 13, 2018

Troy Magennis

Focused Objective

info.sep.com/TroyMagennnis2018
Forecasts Must 3 Things:

1. Statement about some future outcome or unknown event
2. Statement about the level of uncertainty, but not guaranteed
3. Way of eventually testing the actual outcome against the forecast
1. Question: “Will it snow tomorrow?”
2. Uncertainty: “80% chance of snow.”
3. Feedback: “Did it snow?” No? Why did we get it wrong? Are we wrong the appropriate amount of expected times?
QUIZ TIME!
Forecasting is about answering the right questions, to a transparent degree of certainty, with as little effort as possible.
What is a “good” forecast?

Good is a relative term. If you have no information, even a rough idea of what might happen in the future can help make a better decision.

- Troy Magennis, Forecasting using data, Chapter 1
1. It returned **multiple options**, not just the one it defined as “best”

2. It didn’t give my arrival time, but a **duration**

3. The forecast is constantly **updated**
Contrast this with most IT

1. We give a **single option** - one the team has given based on current team size and understanding of the project.
2. We give an **actual date** result, ignoring that the start date may change.
3. Once we begin, the original date forecast is often defined as a commitment and **new information discarded**

@chrisshinkle
Forecasts document our assumptions

Forecasts aren’t just numerical or date values. A forecast is often a numerical value in addition to the embedded assumptions that allow that forecast to be reliable.
A minimum set of assumptions should cover these aspects:

1. A measure of the ability to start delivery
2. A measure of initial scope and size.
3. A measure of expected rework and scope growth.
4. A measure of expected progress of scope delivery over time.
5. A measure of acceptable quality to be able to deliver to customers.
6. A measure of the ability to deliver to customers (environments, process, logistics).
3. A measure of expected rework and scope growth.

Any time we use a distance and pace based model for forecasting, we must consider what might increase the distance or modify the pace. Not doing so means poor forecasts.

The same with software projects, we need to know the original size and how that might increase as we begin delivery.
TYPES OF GROWTH TO CONSIDER

**Time based** - The longer we go the more alterations to original scope get added.

**Rate based** - The more work we complete the more we learn about what we need to do to deliver. (e.g. defects and rework)

**Scale based** - Often work items are split as the team understands the feature story in more detail.

**Event based** - Feedback or things that go wrong in the approval to release process.

- Troy Magennis, Forecasting using data, Chapter 8
## TIME BASED GROWTH

<table>
<thead>
<tr>
<th>Release frequency to customers</th>
<th>Technically easy</th>
<th>Technically hard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous to every 2 weeks</td>
<td>1x</td>
<td>1.25x</td>
</tr>
<tr>
<td>3 weeks to every 6 weeks</td>
<td>1.25x (+25%)</td>
<td>1.5x</td>
</tr>
<tr>
<td>7 weeks to every 12 weeks</td>
<td>1.5x (+50%)</td>
<td>1.75x</td>
</tr>
<tr>
<td>13 weeks to every 26 weeks</td>
<td>1.75x (+75%)</td>
<td>2x</td>
</tr>
<tr>
<td>26 + weeks</td>
<td>2x (+100%)</td>
<td>4x</td>
</tr>
</tbody>
</table>

- Troy Magennis, Forecasting using data, Chapter 8

@chrissinkle
## Rate Based Growth

<table>
<thead>
<tr>
<th>Growth vector</th>
<th>Occurrence Estimate</th>
<th>Story Count Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defects</td>
<td>Every story</td>
<td>1 to 3</td>
</tr>
<tr>
<td>Localization</td>
<td>U.I. stories, 20-30% total</td>
<td>3 to 4 (string translation, adding to deployed resources, testing)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

@chrisshinkle

- Troy Magennis, Forecasting using data, Chapter 8
### Event Based Growth

- Troy Magennis, Forecasting using data, Chapter 8

<table>
<thead>
<tr>
<th>Risk</th>
<th>Probability Estimate</th>
<th>Story Count Estimate (if this risk happens)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance under load &lt; 1 second page load time</td>
<td>50-75%</td>
<td>20 to 30 stories to add an indexing server</td>
</tr>
<tr>
<td>Performance under load &lt; 1 second page load time</td>
<td>More likely than not</td>
<td></td>
</tr>
<tr>
<td>Browser compatibility. Major errors in Safari, Chrome or IE 10+ browsers</td>
<td>20-40%</td>
<td>6-15 stories</td>
</tr>
<tr>
<td>Browser compatibility. Major errors in Safari, Chrome or IE 10+ browsers</td>
<td>Less likely due to early QA using virtual machine images. Beta testers identified.</td>
<td>Assumes, 2 to 5 stories per browser.</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
A minimum set of assumptions should cover these aspects:

1. A measure of the ability to start delivery
2. A measure of initial scope and size.
3. A measure of expected rework and scope growth.
4. A measure of expected progress of scope delivery over time.
5. A measure of acceptable quality to be able to deliver to customers.
6. A measure of the ability to deliver to customers (environments, process, logistics).
Building a forecast from our assumptions
Monte Carlo Simulation
COMMON QUESTIONS…

HOW BIG?
Understanding the size of a feature or project with less effort

HOW LONG?
Understanding when a feature or project might be done and tracking progress

HOW MUCH?
It's too big, “what can I get by when…” Seeing options and making trade-off decisions earlier

@chrisshinkle
QUESTION: How an I estimate the size of a feature or project without analyzing every piece of work?

THEORY: The “size” patterns of randomly sample features, will persist through all other features. Analyze a few and compute for the many.
FORECASTING TOOLS: HOW BIG?

STORY COUNT
FORECASTER

bit.ly/storyforecaster

@chrisshinkle
QUESTION: How can I estimate the amount of time it will take to deliver a feature or project?

THEORY: Using a range estimate or actual team delivery rate, calculate how many time periods to complete delivery.
FORECASTING TOOLS: HOW LONG?

THROUGHPUT FORECASTER

bit.ly/throughputforecaster

@chrisshinkle
TOP 3 REASONS YOUR FORECASTS MAY FAIL

START DATE INCORRECT
Teams almost never fully available on day 1.

BACKLOG VS COMPLETION RATE
We may under-forecast if split-rate isn’t properly accounted.

IGNORING RISKS
Work that “might” need to be done but we don’t know yet

@chrisshinkle
TAKEAWAYS

• Use product outcomes for selecting what goes into the release plan
• Use forecasting to “guess” the backlog size and project duration
• Any statement about duration should include a range and probability

@chrisshinkle