Table of Contents

Adam Yuret - Waiting for Godot- Concrete approaches to busting delays and why your enterprise should care

Premise

Nobody likes waiting the glacially slow two-minutes it takes to microwave their lunch. Imagine how our customers feel waiting years for products only to discover the product doesn't actually solve their problem anymore. Or, worse yet, it solves a problem they needed solved years ago. According to David J Anderson or Troy Magennis, queues, handoffs and wait times represent at least 85% of delays in getting our product to markets. In this talk I'll share concrete approaches to busing those delays. We'll discuss common sources of delay and approaches to reducing or eliminating them using data-driven approaches such as Theory of Constraints and Systems Thinking. Agile is about shortening learning cycles by reducing the time we waste on non-essential activities such as doing work that doesn't create meaningful product, wasting energy on overly elaborate requirements documents that are outdated the minute they're created and working as individuals instead of high-performing teams. Come learn how to use Eli Goldratt’s Theory of Constraints to significantly reduce or eliminate huge delays in your value streams. We want to deliver value to our customers quickly in order to get critical feedback from them on how to evolve the next iteration of our products. While the customer waits, we're not learning from them. Let’s solve that problem together using Theory of Constraints and systems thinking. Learning Outcomes: Attendees will learn: What delays value from flowing through our systems (with some data and some anecdotal evidence) Why delays are the single greatest impediment to our ability to deliver value to our customers. How to identify and address bottlenecks within a system. The basics of Goldratt's Theory of Constraints (TOC) and why those principles and approaches are powerful. How TOC relates to complexity thinking (Cynefin) and using both amplifies their effectiveness at solving problems with managing your portfolio and programs. Introduction to Systems Thinking and how to manage the interconnected relationships within out organization effectively.

Summary

- Content rating (0-no new ideas, 5 - a new ideas/approach, 9-new ideas): 4
- Style rating (0-average presentstion, 5 - my level, 9-I learned something about presenting): 4

Not new to me. Choppy presentation

Action / Learning

- Good systems thinking metaphor - cynfin and theory of constraints (buffalo) - Who is the slowest buffalo; How do we make the slowest buffalo faster; How can we by buy faster buffalo Who is the new slowest buffalo
- Book - Rolling Rocks Downhill: The Fastest, Easiest, and Most Entertaining way to Learn Agile & Lean; The Agile Business Novel. Kindle Edition by Clarke Ching
- see Troy net flow chart if free resources
- Software Development Performance Index. This is work that came originally from Larry Maccherone while working with Rally. Ideas is to use performance, quality (release ability), predictability, and responsiveness. For example:
    - Release ability - if team worked on nothing else but close out bugs how long
    - Responsiveness - how long to get in a bug
    - Performance - thought put - closed work items
    - Predictability - how consistent is our delivery (versatility)
- Dave Allen - Teach Kids About Telling the Time to understand how slippery the english language is to drive requirements.

Presentation

Notes

Concrete approaches

Humanistic flow based systems

Whole people

Continuously improving deliver of customer value at a sustainable pace while responding to change.

Cynefin

Sense of belonging of community Meant some thing that what was hard to pronounce

Model understanding complexity

Obvious is domain Way we make toast is good enough Don't need to get better

Complicated Domain of experts Known Build a jet engine Working, with expert help, follow map / allow experts Will get jet

Complex Have hypothisis Don't know if it is going to work Domain of innovation Software moves from

complex to complicated (operationalize)

Chaos If you take from one domain and apply it to others then cliff to chaotic domain You will act Not a choice Act first Once Here hard to get out

Disorder Don't know

Linear system vs complex

Essential properties of a car not from the bits - essential attribute is ability to move people. Does not make sense as bits Russel ackoff Best parts of a car will not produce a car

Book - introduction to system - Donella meadows

Scrum - supercharge one part

Coach John McKay

Understand the entire system

Theory of Constraints

Where is the slowest buffalo Test? So we through the testers to the wolves

Techniques to remove the bottleneck Lots of wastage in test

Everything happening upstream is waste - can only go this fast

Don't have test mindset “Sprang from womb without capability?” I am genetically better at finding risk, than they are at creating it

Identify constraint Exploit constraint Subordinate everything else to constraint Elevate the constraint Return to step 1

For the developer / tester problem Prefer to help solve the problem, rather than sit and do nothing

First 3 steps does 30% improvement

Don't jump to elevate constraint This is get more people, invest in tools

Cnn choose a bottleneck - and then say just live with it

Dependencies / queues

Lead time - demand to delivery Cycle time - work starts to delivery

Need to look at lead time

Net flow chart More items started - red More items complete - green

Business unit all red Team cannot have a product owner because they are all too busy creating

thousands of requirements

Measure process cycle efficiency Actually doing something that adds value Not about velocity Not impactful Deal with handoffs and queues

Dependencies are bottlenecks

Don't get too attached to a specific set of metrics

Balanced metrics Software development performance index

Change one and other

Balanced scorecard Release ability - if team worked on nothing else but close out bugs how long Responsiveness - how long to get in a bug Performance - thought put - closed work items Predictability - how consistent is our delivery (versatility)

Limit ordering options Constrain flow of value Negative impact culture

Defining product value is hard Dave Allen value - teach kids about time

Local optimization are great if don't degrade whole system Measure flow of value Find delays to delivering customer value and remove them - make them visible

Cycle time Lead time Throughput Touch time Value (economic and oblique)

Who is the slowest buffalo How do we make the slowest buffalo faster How can we by buy faster buffalo Who is the new slowest buffalo

~~LINKBACK~~ ~~DISCUSSION~~

Forecast, Delays, Dependencies, Conference, Agile2016, Planning, Probability, Metrics