STOP
#NoProjects

The End of Projects and what happens next

Allan Kelly - @allankelly.net
Allan Kelly...

- Training for Agile
- Advice on
  - Agile, adopting Agile
  - Organizing teams

Author
- Xanpan: Team Centric Agile Software Development
- Business Patterns for Software Developers (2012)

allan@allankelly.net
http://www.allankelly.net
@allankellynet (Twitter)
Problem #1 - Benefit

Project Success Criteria
- On Schedule
- On Budget
- On Quality (Features)

Where's the value?
£ $ €
Project Model Assumptions

1. Value is knowable
   And is known before start
2. There is no value in flexibility
   i.e. Options are valueless
Pre-defined outcome?

Requirements change
The world changes

The observed rate of change in the US is about 2% per calendar month

Capers Jones, 2008

Compound to ~27% per annum
70% believe they are failing to identify and quantify the benefits adequately.
38% openly admit they overstate the benefits in order to obtain funding.
80% report that the review and evaluation of completed projects is also inadequate due to the focus on [achieving] cost, time and quality objectives and not on whether the intended benefits were realized.

Delivering value from IS and IT investments, John Ward, Cranfield School of Management, 2006 – survey of UK & Benelux managers
http://www.som.cranfield.ac.uk/som/dinamic-content/research/documents/deliveringvaluereport.pdf
The Project model leads to...

Conflict and...

Goal displacement

– Chasing date over benefit
– Chasing time over benefit
– Chasing cost over benefit
– Chasing features over benefit
Solution
Focus on Benefit not The End

Ask not, “When will the software be done?”

Do ask: “When will the software deliver value next?”

Think: Stream of Value
(which might stop one day)

Not: An end date
Change Governance

• Base Governance on actual delivered benefits
  – Not milestones completed
  – Not documents
  – Not budgets

What have you delivered for me lately?
Governance based on results

• Experiment friendly
  – Failure tolerant
  – Fail fast, fail cheap (salvage)

• Align with strategy

• Governance through *Iterative Portfolio Management*
Iterative Portfolio Management

• Fixed schedule for gate reviews
• Review what is ready on a date
  – Benefit delivered to date
  – Potential benefit if continued
Problem #2 – The End

Software isn’t temporary
Successful software doesn’t stop

Successful software continues to change
Only dead software has an end-date

Projects end
Successful software doesn’t
Successful software?

1) If they use it, it will change

2) Only Dead Software Stops changing

Data from SourceForge search for “WebBrowser” 19 Jan 2014

Moodle
Weekly downloads: 23,239
Last update: 3 days (16 Jan)

Web Torrent
Weekly downloads: 0
Last update: 17 April 2013 (9mths)

PerlLORD
Weekly downloads: 0
Last update: 25 May 2013 (8mths)
End dates damage quality

• Short term thinking leads to...
  – Corner cutting
  – Known & unfixed bugs
  – Residual technical debt
  – Knowledge lost
End Date considered harmful

Late requirements considered inferior
A Project is...

“A temporary organization that is needed to produce a unique and predefined outcome or result at a pre-specified time using predetermined resources.”
A project is....

"PMI defines a project by its two key characteristics:
• it is temporary and
• undertaken to create a product, service, or result that is unique."

Project Management Institute - http://pm4id.org/1/2/
A Match Made in Hell

Software Development

Project Management

Software is forever

Projects are TEMPORARY
Rational end dates?

In the US more than half of the large projects ... predetermined end date is selected, and it is forced on the project by arbitrary decree.

Capers Jones, 2008
Solution

• Work to the business deadline
  – Do what you can in the time
  – Do good work
Deadlines Good

End dates Bad
Temporary organizations

The most destructive idea known to software development
Temporary Organization?

- Storming
- Norming
- Forming
- Performing
- Destroying

Takes time & money!

Why destroy performing teams?
Why spend that money?
Why loose knowledge?
Temporary organizations

The most destructive idea known to software development

Disbanding teams destroys

– Knowledge
– Capability
– Performance
Corporate Psychopathy
Process by which corporations disband performing teams and release staff
Solution

Base work around stable
   – Teams

Aim for continuity
Continuous flow
Continuous improvement
Continuous delivery
Continuous benefit

Continuous not Temporary
Bring the work to the team
Organize by business stream & team

- Aim for stable teams & continuity
- Close to business
- Manage queues within capacity
What to do about it...

• Keep teams together
• Flow work to the teams
• Work in the small
• Work continually
• Demonstrate value
Problem #4 – Projects are Big

- Project model is optimized for big
- Used on small pieces of work it inefficient
- Projects push big decisions up
  - to big men
  - with big cheque books
  - top-down authority
Software development...

• Does NOT have economies of Scale
• Development has DISECONOMIES of scale
Milk is cheapest in BIG cartons

Software is cheapest in lots of small cartons

And small cartons of software reduce risk
Big increases risk
Solution
Software development...

- Does NOT have economies of Scale
- Development has DISECONOMIES of scale

Therefore

- Stop thinking **BIG**
- Start thinking *SMALL*
Consider a large project

Project A: Risk = 30% Value at risk = £1m
Therefore risk weighted value = £300,000

Prj B: Risk = 15%
Value @ risk = £½m
Therefore ... = £75,000

Prj C: Risk = 15%
Value @risk = £½m
Therefore ... = £75,000

E: Risk = 6%
@risk = £200k
Therefore = £12k

F: Risk = 6%
@risk = £200k
Therefore = £12k

G: Risk = 6%
@risk = £200k
Therefore = £12k

H: Risk = 6%
@risk = £200k
Therefore = £12k

I: Risk = 6%
@risk = £200k
Therefore = £12k

J: Risk = 6%
@risk = £200k
Therefore = £12k
Big Batch Size

Wait
Build this!

Wait
Deliver this!

Wait
Make lots of this!
Small Batch Size

... and keep going
The End of Projects

• Projects are accounting codes
• Finished Software is **Dead Software**
  – Living software changes
  – Dead software doesn’t
  – Living software doesn’t end
• Project thinking kills software
Wait a minute....

Not fair!
You misrepresent us!

Project
Manager

So what is “A project”?

Why does Prince 2 contain this definition?

And what does a “Project Manager” actually Manage?

We don’t
- Break up teams
- Stick doggedly to requirements

We do
- Allow change
- Consider value
- Work continually
Problem #3: Language
“False Projects”

When the language of projects is used to discuss work which is not a project

Confusion!

Stop confusing people
Stop misleading yourselves
Beyond Projects

A new model....
Waterfall 2.0  Continuous Flow

Jonathon’s Run Fall, Pennsylvania by Hubert Stoffels (http://flickr.com/photos/22195940@N00)
Creative Commons License
Continuous flow

• Work in the small
• Get good at doing small things
  – Deliver small increments of value
  – And evaluate results
• Go fast
• Value seeking
• Repeat, don’t stop
Organize for...

Work management optimized for small
  – Lots of small decisions
  – Devolved authority
Base work around stable
  – Teams
Aim for continuity
Could this ever work?

- Governance by results?
- Fail fast, fail cheap?

Seems to work well on Sand Hill Road ...
Please

Remember to rate this session

Thank you!
Thanks!
Beyond Projects

• It ain’t ever over
• BAU is not a dirty work

allan kelly
allan@allankelly.net

www.softwarestrategy.co.uk

Twitter: @allankellynet

Xanpan
Amazon or LeanPub
https://leanpub.com/xanpan
Print http://tinyurl.com/nnu7hom