Dr. Leigh Griffin
Engineering Manager
Red Hat

The Trials and Tribulations of Agile Adoption
Trials and Tribulations

Cambridge Dictionary Definition: *Troubles and events that cause suffering*

NIV Version of the Bible: James 1:2-3 Trials and Temptations: *Consider it pure joy, my brothers and sisters, whenever you face trials of many kinds, because you know that the testing of your faith produces perseverance.*

Mahatma Ghandi: *What is life worth without trials and tribulations which are the salt of life.*

Figure 2: Dilbert’s trials and tribulations
Red Hat Mobile: An Scrum Agile Conversion

- Red Hat Mobile Application Platform (RHMAP) formerly a start-up acquired by Red Hat
  - A “Scrum” BUT Team
  - Releasing every couple of days to customers
- Red Hat have a strong history of long running On Premise / Self Managed solutions
  - 6-24 month release cycles as a traditional Waterfall
  - Rigid security and compliance work
- First SaaS product in Red Hat, releasing monthly, one month after Engineering efforts wrap up
Misaligned way of working
The Great Game Begins

HOUSE AGILE

Sprint Finish is Coming

VS

HOUSE WATERFALL

PRD, DUTY, HONOR
My House: An Ivory Tower!
Beware of Success Stories
The trials we faced

• Going to cover primarily 4 trials we faced in Red Hat Mobile
  • I will bring in our path to success along the way!
  • We are an Agile team heavily influenced by Scrum (modified) and Kanban

• Going to bring in past experiences of having done this style of conversion in other companies
  • A green field Security company
  • A waterfall software house in the Insurance industry
  • A research organisation in the public domain
Trial 1: Peopleware
Trial 1: Peopleware

- Hardware and Software are easy, Peopleware is hard

- People working in a legacy way
  - Their System, Their Way, Their Job
  - Perception that you are taking their knowledge, power or job
  - They might be wary of risk and exposure
  - They might have built their career on this way of working

- The people want to move in a certain way, in our case Scrum, warts and all.
  - The team understood Scrum loosely
  - The management team wanted to move towards it more formally

I got a job :)
Owning Product Direction: Often a singular voice

- Historically 1 voice ruled them all
  - Most knowledge? Biggest ego? Loudest?
  - Reckless technical debt becomes a way of life

- Hard to open decisions to a team wide level when rigid command & control exists
  - More people wanted & needed a say in the direction
  - More so as Red Hat is a meritocracy with the best idea winning out

- Scrum, which we followed, demands a Product Owner role
Traditional Project & Product Management

- The overall program tracked individuals at a percentage capacity level
  - The classic fallacy of a 50:50 split on work

- Agreeing a project plan and outward time commitments months in advance
  - Often before the groundwork and analysis was done
  - Often with commitments already made!!
  - Held accountable for dates, traffic light system had us red most releases

- Sanctity of the Product Requirements Document (PRD)
  - The masterplan, thou shall not deviate from!
The need for balance

- Teams wanted more say and innovations
  - More room for technical debt paydown
  - More time for research and innovation
  - Unhappy being called a resource!

First Battle:
- 80% capacity with reserved 20% for Backlog Management

- Changed the perception of being 100% focused on Product / New Feature Needs
  - Raised awareness that all work goes to the betterment of the Product
Second Battle:

- We said no to all new product development for a period of 6 weeks
  - It caused arguments and headaches
  - We got escalated through various managers
  - Ultimately the right call was made as we had the right reasons

- We couldn’t stabilise and consolidate the platform with small time investments
  - We dramatically improved the quality of the platform and our tooling
  - We paid down a lot of pain and shortcuts
The need for balance

Third Battle:
• Eventually we stopped resourcing at an individual level and moved towards a team level (9 months in!)
• We stopped investing huge amounts of time in PRDs (12 months)
  • Moved all our work to User Stories and Epics
  • Changed the view from scope based to vision based

Fourth Battle:
• We broke the matrix and formed a true Scrum Team
  • QE became one with Engineering
  • Exceptionally difficult to do in a matrixed organisation
The need for balance

The Battles we lost:

• 60:20:20 reverted to 95:5:0 when customer pressure came on
  • We just couldn’t say no as a team as everything was a high priority

• Huge push back on depreciation of broken or no longer used features
  • We couldn’t make a “for the best” decision
  • We did not have a true neutral Product Owner

• We lost staff members
  • Comfort zone change
  • Exposed skillsets
Trial 2: The 7 deadly sins
7 Teams; 7 Streams; 7 Backlogs; 7 Silos = 7 headaches

- Historically we had well defined areas within the product that were disconnected
  - It felt like multiple products in one
  - All things to all people

- We rolled our Scrum processes out to 7 fully cross functional teams

- Each team was a well defined part of the product

- Each team had it’s own backlog to manage and it’s own vision of the world
Envy and Sloth

• Each teams backlog contained:
  • Technical initiatives
  • Bugs
  • Product features
  • Community features

• Managing Teams Expectation was a battle
  • Some teams were purely product facing
  • Some teams had more community facing
  • Some teams had more flexibility

• We inadvertently created an envy position
  • Some teams feeling exposed and under pressure
  • Some teams blogging and having fun and “appearing” lazy to others
Gluttony and Greed

• The liberty to “do what you want” because you are Agile slowly crept in
  • Teams taking too much liberty with personal initiatives
  • We tried to strike a healthy balance but struggled to manage it

• The product team saw 7 teams and a chance to drive 7 features
  • Our releases became overloaded with huge emphasis on new features
  • Our quality dipped because of the moving parts

• Roadmaps became more aggressive

• The value of features were rarely questioned as we had no Product Owner questioning value
• We found it very hard to say no
  • Being Agile is a badge of pride
  • We can move fast and give you features you crave

• We convinced ourselves it was working
  • Our stats showed huge improvements:
    • 6% to ~96% verification rates within a Sprint
    • All work now being done in the context of a sprint, no more dark days of hiding work!
  • Our team were for the most part happy
    • Being listened to
    • Masters of their own destiny
  • Management were happy
    • We became a standard bearer within Red Hat
    • We even got a few conferences out of it :)
Lust

- We wanted to keep improving

- Grand designs about Large Scale Scrum (LeSS) and even Small Scale Scrum
  - We knew practically how to Scale both ways

- Moving some teams to Kanban
  - We had a lot of unplanned work which Kanban is suited to

- We wanted to keep innovating

- The Lust to succeed and better yourself can blind you to imperfections
WRATH

“...in its purest form, presents with self-destructiveness, violence, and hate that may provoke feuds...”
The mistakes we made

- A lot can be traced to the lack of a Product Owner role
  - Goes back to peopleware, fear of change, political implications and more
  - We lost that neutral best for the product view, we became the PO!
  - This pivoted us from being a Scrum team to being an Agile team

- We created artificial work to satisfy and feed 7 teams
  - Value of what we created was questionable
  - The cost of maintaining code is high
  - Demotivating for staff to work on dead features

- It took 18 months to make a decision on sunsetting
  - Going forward 1 product not 7 in 1
Trial 3: Evolution Problems
The concept of Dynamic Teams emerge

- We were not comfortable with the pace of work of our static teams
  - Cross skilled teams but our output was not where it could be
  - Traced back to the 7 siloed areas of our product

- Skills based profiling of team members
  - Trying to pick the best skills for the job at hand

- Death to static teams where possible was on our agenda!
  - In the hope it would kill some of our siloed product areas
The PRD decision

• We put Dynamic Teams forward for defined and scoped work

• We had to revisit the fundamentals of what that meant

• We abandoned the full on 100+ page PRD document for good reasons (waterfall!)

• For several releases we…..
  • Didn’t really know whether we solved the problem we set out to solve completely
  • Over and under engineered some solutions as a result
We hired a Business Systems Analyst (BSA)
  - A very traditional Waterfall role
  - The voice of a customer

BSA was our front runner, meeting with stakeholders and helping gather requirements

Facilitated story generation and acted as a bridge
  - For business folks a BSA could distill down the technical needs
  - For technical folks a BSA could distill down the business needs
  - If you say it really quickly it sounds like a Product Owner!!
Headache 1: The need for a ScrumMaster

• If you pick your best people for a team you want them coding

• Our technical leads / principal engineers were our ScrumMasters
  • Management could not afford to ‘lose’ a developer optically
  • ScrumMaster role became a secondary role
  • Our pace became slower and our processes creaked

• Managers took the role
  • Not ideal, already overloaded
  • Agile enthusiasts and natural coaches to the team
  • Blurs the line between a recommendation and a mandate
Headache 2: The side bar conversations

- Traditional Agile / Scrum teams are command and control centered
Headache 2: The side bar conversations
The battles we lost

• Upskilling and research time became non existent
  • A static team naturally included that in their backlog

• Dynamic teams gravitated towards the A-Team mentality
  • The same names will solve any job (envy, gluttony, pride, wrath!)

• Controlling side bar conversations and getting the right info to the right people

• We lost the initial battle for dedicated Scrum Roles
  • We got a Product Owner in disguise after 12 months
  • We got part time ScrumMasters and it nearly sunk us
  • We got dedicated ScrumMasters after almost 2 years
Trial 4: What Agile Exposed
More Teams, Faster Work

• We grew the team to over 50 people all in
  • Almost doubled our Engineering team in a 9 month window
  • All 50 were contributing at several levels
    • JIRA tickets
    • PRDs and ideas
    • Code, code and more code!

• Our overall speed of delivery increased by an order of magnitude
  • 33 releases in 2017 Vs 11 in 2016
  • We were a well oiled machine
  • Teams churned through work
  • Quality did not suffer, in fact it improved
  • That’s outwardly a success story
Backlog Overload

- JIRA became a dumping ground
  - Ideas, Bugs, Enhancements
  - If it’s not logged don’t talk to us

- It was a full time job, but nobody owned it

- False expectations set
  - I opened that ticket 3 months ago!!
PR Nightmare

- All our code generates a Pull Request (PR) on Github

- PRs need to be reviewed
  - That can take a variable amount of time

- PRs need to be merged at a cost of ~2 hours per
The need for Automation

- Cumulatively we lost 363 developer days in a 9 month window exclusively on merging
  - If we were Lean we would be talking about waste

- Automation was invested in heavily to address this
  - Cost us 1000 developer days to solve
Battles we Lost

- Expectations on ticket fulfillment were never set
  - We tried aggressively closing tickets with no activity after 3 months, it failed
  - We tried educating people to communicate and not dump tickets, it failed
  - Still losing that war unfortunately

- We lost the battle of the PR
  - But won the war with automation
  - Came at a huge cost of time investment and a bigger battle to secure it
We got there, except, there moved

- We are a hugely successful Agile conversion
  - Now we are 5 Agile Teams with a mixture of Scrum and Kanban
  - Our speed of output is possibly the best in Red Hat
  - We united several tribes and are driving an Agile conversion project at a wider level

- When we got to the horizon, it moved
  - We constantly are resetting our expectations
  - We have made a lot of mistakes, some we invented for ourselves!
  - We haven’t solved them all, I’m not sure we ever will

- We learned to pick and choose our battles, the war is far more important
Final tips

• Evaluate yourselves as much as you evaluate your processes
  • We got blinded by our own success to an extent

• Invest in automation or similar processes
  • Predict the best case meteoric success you will have
  • It bit us hard and became an afterthought

• Find a support network
  • In Red Hat I am part of an Agile Coaches community
  • Attending conferences like this one :)

• Be happy you hit a wall
  • Be happier when you find your way past it