The 'A' Word
How we broke and fixed team autonomy

Elizabeth Ayer & Gareth Bragg

@ElizAyer  @GarethBragg
What we wanted

• Engaged, motivated teams
• Teams feeling effective
• Product Development fit to scale
Why “autonomous teams”? 
Autonomy
Mastery
Purpose
Giving control, not taking control
Pushing decisions to the information
Leader-leader model
We had the right conditions
Our Values

- We're reasonable
- We're ingeniously simple
- Transparency helps us continually improve
- We do our best work in teams
- When our customers succeed, so do we
- Don't be an asshole
Restructure

• Small teams:
  • Technical Team Lead
  • UX Designer
  • 5 Software Engineers
• Coaching:
  • Quality, Agile, Technical Communication
• Product Group Pairings:
  • Development Leads & Product Managers
Exercise 1:

How could increased autonomy help you in your role?

What challenges could it present?
Chapter 1: "Autonomy"
Not this.
WHAT'S THE TEAM AUTONOMY BARGAIN?

A DOCUMENT FRAMING TEAM AUTONOMY OUTLINING WHAT REDUCE EXPECTS OF AUTONOMOUS TEAMS, AND WHAT TEAMS CAN EXPECT IN RETURN

SUGGESTIONS FOR OUR TECHNICAL STRATEGY

- Code should be "safe" - avoid breaking
- Can we mergeƸ(merge) in enough of a window for very short
- Team can do work on their own without support
- Work in bands
- Are results reported? Is it new
- What does code mean? Can we explain? Can we change?
- Don't need to explain why we did it
- Is it "what?"
- Is it "why?"
- Is it "how?"
- Can we break down processes?
- Can we break down ownership?
- Code should be "safe" - avoid breaking
- Could we have a couple of examples?
- How many do we want? What happens when things go wrong?
- If you're thinking of your value:

PLEASE ADD YOUR COMMENTS:
Redgate will give teams:

1. **Freedom to decide** how to engineer products to solve customer problems

2. **Support** from specialists, coaches and experts in the organization

3. **Clear direction** and constraints from technical & company strategy, allowing teams to align with a greater purpose or vision

4. **Team stability**, aiming to keep teams together by default and move teams to the most important work, rather than breaking them up
We believe the following are key for teams building ingeniously simple software:

1. **Focus on your mission**
   Everyone on the team should know their mission and how their current work contributes to it.

2. **Be transparent**
   Everyone should be able to see your progress without having to ask what you're working on and why.

3. **Be agile**
   Ship valuable software frequently and incrementally, get rapid feedback and be able to respond quickly.

4. **Know your users, your product, your market**
   The freedom to make critical decisions, comes with the
Clear expectations
What we're going to do to improve things

We think there are three distinct tasks that we need to tackle as soon as possible:

- Reframe autonomy and move away from using it as a mechanism for change – starting with this blog post and reinforced by the following steps.

- Clarify our vision for great engineering teams – this will look a little like this, building on themes we've already talked about since the restructure. I'd like to gather input from across the Product Division too (look out for more on this soon).

- Help teams assess where they are and agree what support they need. We need to redo this too and then re-engage with the teams throughout.

Sorry we made this confusing and thanks for letting us know about it, but the proof of the pudding is in the eating - hopefully we'll all start to see the benefits.
Exercise 2:

Picture a development team you work with.

Write down 2 responsibilities you’d like them to take on.
   Grooming the backlog? Cadence for ceremonies?
   Setting or meeting quality standards? Noticing drops in KPIs/KRs?

For each responsibility, write down
   ▪ One skill the team would need to develop
   ▪ One change the org would need to make
Chapter 2: The Capability Gap
Building a set of principles for our teams
Development teams at Redgate enjoy delivering valuable, ingeniously simple solutions with the help of six Fantastic Beasts:

- With the power of the Porpoise, we stay focused on our purpose.
- The wise Owl focuses us on learning about our users and their world.
- The transparency of the Jellyfish keeps us open and honest.
- The collaborative Meerkat encourages us to listen and be mindful of others.
- The agile Chameleon reminds us to adapt as we frequently inspect feedback and learn from our situation.
- The proud Koala maintains a healthy balance between delivery and quality.
Owl

We learn about the users of our products

Team members engage with our users and strive to learn about them and their world.

Team members understand the domain in which their products exist.

Teams use this knowledge to make critical decisions and build software that our customers love.
Development teams at Redgate enjoy delivering valuable, ingeniously simple solutions with the help of six Fantastic Beasts.

**FANTASTIC BEASTS**

**FANTASTIC BEASTS AND WHERE TO IMPROVE THEM**

1. **We deliberately focus on our purpose**
2. **We are honest and transparent**
3. **We inspect and adapt**
4. **We learn about users**
5. **We collaborate and listen**
6. **We deliver sustainable quality**

The power of the Porpoise keeps us focused on our purpose.
The agile Chameleon reminds us to adapt as we frequently inspect feedback and learn from our situation.
The transparency of the Jellyfish keeps us open and honest.
The wise Owl focuses us on learning about our users and their world.
The collaborative Meerkat encourages us to listen and be mindful of others.
The proud Koala maintains a healthy balance between delivery and quality.
How do we engage with our users?
By engaging with them.
We know agile. Right?
Exercise 3:

How do you like to be led?

What traits are important in a leader, for you?
Chapter 3: Developing Leaders
What leadership means to us
Attributes of a Redgate Leader*

1. Growing others
2. Courageous
3. Accountable and responsible
4. Encourage clarity
5. Value collaboration

*These WILL be different for your org.
Tech Leads at 1 month
Tech Leads at 6 months
Giving control, not taking control
Pushing decisions to the information
Leader-leader model
Tech Leads at 6 months
Too much delegation to Tech Leads

AND

not enough delegation from them
Why not delegate?

- Not appropriate to
- Don't think about it
- Everyone else is busy
- Don't want to distract
- Lack of skills
- Not interested
- Quicker for me to do it
- Task is too important to delegate
- Accountability is on me/losing control
- Lack of knowledge/access
- Don't know how
- So they don't become the owner in the future

Logistical reasons
- Task is super interesting
- Data or opinions
- Lack of trust they will do it
- Task is pants
- Losing recognition
- Delegating yourself out of a job

Why we should delegate?

- Good learning exercise
- I like going home
- Hit by the bus rule
- Knowledge sharing
- Shared ownership
- Maintain balance of own role
- Other people can do it better
- Helping team performance
- Motivating and helps develop skills
- Builds trust
- Encourages retention - sense of team
- To avoid learnt helplessness from team

- Giving people chance of different roles
- Different view or task
- It is a skill
- Don't have time
- Proving valid task
- To look after yourself
- So you are in a better place to help the team
You Can’t Do It All

- You may have been the one who made projects succeed by solving the most difficult problems
- You may have got used to being the go-to expert on your codebases
- You used to throw the weight of your accumulated knowledge, skill and time at a problem.
- This also no longer works...

Superheroes Not Required
Where we are now?
What we wanted

• Engaged, motivated teams
• Teams feeling effective
• Product Development fit to scale
Inertia
Lessons

1. ‘Autonomy’ is an explosive word

2. Don’t introduce extra tension.

3. Align on Purpose. Autonomy and Mastery must support ‘why’.
Lessons

4. Engage teams in their own improvement.

5. Capability assessment, inside and out.

6. Leaders grow leaders who grow leaders.
Resources

• The Spotify Model
  • https://medium.com/the-ready/how-to-build-your-own-spotify-model-dce98025d32f

• Start with Why
  • https://startwithwhy.com/

• Fantastic Beasts

• Involving dev teams in user research
  • https://www.slideshare.net/ElizabethAyer/aitc2018-ux-impact-elizabeth-ayer

• Delegation
  • https://medium.com/ingeniouslysimple/how-to-get-more-done-by-doing-less-f047d2f53d47