



Cabinet Office

Julia Harrison

Head of Product

Government Digital Service

@JuliaFromIT

My name is Julia Harrison; I work at the Government Digital Service, or GDS as we call ourselves, which is part of the cabinet office.

I'm a head of product, and one of the areas I look after is Reliability Engineering.

Leading Smart People

This talk is called Leading Smart People.

And I'm going to start by briefly touching on something I hear sometimes, from product managers I work with, but also from managers of tech teams.

**I should
know all the
answers**

**I'll sound
stupid if I ask
questions**

Lies

**I'll only be
respected if I'm
the most technical**

**I'm an
impostor
here**

And these are some lies that people tell themselves. Maybe at some point in your career you told yourself lies like these, or maybe some people you work with do this right now.

Unless it's your first week, you can't be an impostor.

Worst case, you're an infiltrator.

GDS

And I have some good news...



Cabinet Office

Julia Harrison

Head of Product

Government Digital Service

@JuliaFromIT 🙌 *(totally an infiltrator)*

I started out in desktop support. I spent about ten years working on support and windows infrastructure projects, then I got into IT Service management, then service improvement, and from there I got into Product Management more or less by accident. And some years later, here I am the head of product looking after reliability engineering at GDS.

So I am totally an infiltrator.

But don't worry, I'm not a spy or anything, I'm security cleared.

Try never to be the smartest person in the room. And if you are, I suggest you invite smarter people... or find a different room.

Michael Dell

GDS

It's more than OK not to have all the answers or be the smartest - it's also desirable.

But if the leader isn't the smartest person in the room, what are they?



A leader, by definition, is someone people follow.

It's not the job of a leader to tell everyone what to do. It is the job of the leader to set direction, and inspire people to follow it. Not all leaders even have line management responsibilities.

A good leader is someone people follow not because they have to, but because they want to.

[Photo credit: [Not Canadian Geese!](#) by [Richard Giddins](#) is licensed under [CC BY 2.0](#)]

Where are we going?

If you want people to follow you, you should probably know where we're going

Vision

TechOps Vision

Transform government by making it simpler to build digital services and run them the right way.

GDS

At GDS, we have Reliability Engineering teams, and we have product managers with those teams.

And something product managers will do for a product, or even a team they work with, is set a vision.

A vision should be ambitious, inspiring, achievable. Here's the vision for one of the areas I look after.

Your vision should be something your stakeholders and your wider organisation can buy into. Get them excited about it. First, because it's helpful securing funding, but also because it'll come in handy getting their support in sticking to your strategy.

Strategy

RE Strategy

1. Bring GDS services together to a common TechOps platform of tools and services running on GOV.UK PaaS
2. Promote and use Continuous Deployment for GDS services
3. Look ahead to the latest generation of "serverless" Functions as a Service (FaaS) technology
4. Use a shared responsibility model between RE and service delivery teams
5. A standard approach to supporting GDS services

GDS

Strategy is how we achieve our vision.

Having a written strategy is the kind of thing you often mean to do, but keep putting off because it seems too difficult, but you might already have one, you just haven't written it down yet. If it's easy to make decisions about what to do and what not to do, because you know you're going in this direction rather than that direction, that hints at a strategy.

And strategy should exist at all the different layers of your organisation. Everyone should have an opportunity to contribute.

Minimum viable strategy:

- Communication tool
 - Can we, and others, others see where we're headed?
 - Is there enough detail for others to align with us?
 - Is there enough detail to challenge us if we're wrong?
- Decision-making tool
 - Would it help us decide what things to do, versus what

- not to do?

Here's our reliability engineering strategy, that will move us towards our vision. You can see it's a bit more specific. Hopefully just specific enough that people understand where we're going and how to align with us, or challenge us if they think we have it wrong.

Strategy should be stable, not rigid. If it feels like you're doing 'wrong things' because the strategy says so, that's a really good clue that you need to revisit your strategy. That's really important. Strategy should be serving you, not the other way around. And if everyone contributes, hopefully they they'll feel more able to challenge if it's not serving its purpose.

Autonomy

Mastery

Purpose ✓

GDS

From Dan Pink's book *Drive*. These are the conditions for intrinsic motivation. He mostly applies it to individuals, but it works just as well for teams.

To look at it another way, why would you hire smart people and NOT give them these things?

And if we're setting an inspiring, ambitious vision and strategy, we're giving people a sense of purpose. Awesome!

What next? What else do you need for people to *want* to follow you?



Even if you have line management authority, getting people to do things “because I’m your manager and I said so” should be a last resort. And you might not have line management authority over all the people you need to lead anyway.

Real leadership requires permission. You get that permission if the team respect you. They respect you when you add value.

When you work closely with a team, one really visible way to add value is by unlocking the knowledge of the team. Help them get to answers they didn’t know they knew.

[Photo credit: Aretha Franklin / Peri by Teresa Grau Ros is licensed under CC BY-SA 2.0]

Coaching teams

...and we do that by coaching

Asking **powerful** questions

GDS

The second most important thing to do when coaching is asking powerful questions. I'll come to the first in a bit.

First, some examples of powerful questions. You're probably using some of these already.

What do we know already?

What if we're wrong?

GDS

When a team feels stuck solving a difficult problem, or when you're in the middle of an incident and it feels like you're not getting anywhere, these are great questions to take stock of where you are, and flush out any assumptions that might need to be challenged.

Do we know what **good** looks like?

Do we know what **good enough** looks like?

GDS

These are great scoping questions. By stating up-front what we're aiming for, and writing it down, we make it much less tempting to continue tinkering or gold-plating, when really it's time to release something and move on.

Two days. Is that assuming everything works first time?

GDS

My favourite estimating question.

If the answer is Yes, the follow-up question is “...and when was the last time you had two uninterrupted days when everything worked first time?”

Can we solve part of the problem sooner,
then do more later?

GDS

This is a nice agile coaching question. Can we sequence work in such a way that we deliver some value sooner, rather than all at the end of a piece of work?

If we do that, will you feel proud of it?

GDS

It runs counter to the stereotype of product people, but I spend a lot of time asking teams what's the *smallest* thing we can do. And I've learnt by my mistakes that sometimes I push too hard.

If I ask this question and the team look kind of awkward, like they don't want to answer, then I've probably pushed too far. So I need to find out what's the smallest thing we can deliver that they'd feel proud of.

[...]

GDS

And I said that asking powerful questions is the second most important part of coaching. The first is listening. A powerful question is only powerful in the right context. Sometimes a team needs to be challenged or pushed, sometimes your job is to make them feel supported and reassured. Listening, “reading the room”, knowing when to push and probe, when to back off, and when the discussion is going just fine without your input, are all vital to successfully coaching teams.

Autonomy

Mastery ✓

Purpose ✓

GDS

And by doing all this with the team, they get better at asking the same kinds of questions, they start coaching each other. Which is great news for you and for them; it means the team gets better at solving hard problems. It's not so much about building individual mastery, but by learning from each other and learning to use each other's strengths, they build a kind of team mastery, which feels great to be a part of.

So that leaves just one of our three components of intrinsic motivation to cover. Autonomy.

Being a trusted partner

When you provide value to other parts of your organisation, there's a danger of becoming an order-taker. "Do the thing". Or worse "do the thing by next Friday". This isn't great for you, because it sucks to not feel like you have any control over your work. It also sucks for your organisation, because remember that strategy and vision we talked about? If you're taking orders from whichever senior manager shouts loudest that week, you don't get to work on that strategy and vision.

The alternative is to be a trusted partner. What comes to mind when you think about a successful partnership? Hopefully it's not one person directing the other all the time. Partnerships that really work, whether it's personal relationships, business partnerships, sporting partnerships, they're good at communicating just enough about what they need, they build a deep understanding, and they trust their partner to support them and to have their back.

And obviously, you can't get that just by saying it. So how do we get there and how do we keep it going?

Building partnerships is a big part of what good product managers do. So I'm going to share some tips from my time as a product manager...

1. Don't build things. **Solve problems.**

GDS

Well, of course we build things. But why?

I need a dashboard

GDS

Here's a typical thing a stakeholder might say to us.

And my response would be... "why?"

Or, more respectfully, "what will the dashboard tell you?"

Or, more specifically, "what action will you take in response to things you see on the dashboard?"

~~I need a dashboard~~

I need to know in advance when I should
add capacity to my platform

GDS

Already, by framing the requirement as the problem being solved, not the thing being built, you're free to find other ways to solve that problem. If the need is urgent, and for some reason you can't build a dashboard in the time they need, maybe a daily email would solve the most immediate problem?

And we talked a moment ago about coaching, asking powerful questions. It's fine to coach your users. It's fine to coach your managers, their managers, your senior stakeholders.

The more often you ask these kinds of questions, the better your stakeholders get at telling you what they need, not what the solution should be.

2. Everything is a trade-off.

GDS

Having informed conversations with your stakeholders is all about articulating trade-offs.

“You could have the exact thing you want, and we think it’ll take three months. But, because we understand the benefits you’re trying to achieve, we think we can get you 80% of the way there in two weeks. But it would add to the overall timeline. Would you like us to do that?”

“Yes, we *might* have this ready a week earlier if we resurrect the half-built thing in the legacy platform. But is pushing to release a week earlier really worth the risk to quality, supportability, and our reputation if it goes wrong?” And sometimes, maybe, if you’re working to a genuinely immovable deadline, it is.

Being as open about these things as we can means we make better choices *and* build trust and strengthen our partnerships.

3. Strategy helps you decide what to do, and what **not** to do.

GDS

Remember I said it was really important to get your wider organisation excited about your vision and bought into your strategy? Here's why it's *especially* important if you're trying to get out of the order-taker role.

Any time you take on new work that doesn't advance your strategy, it delays achieving your vision.

That doesn't mean you *never* do work that contradicts your strategy. It's a bit like tech debt; call it strategy debt if you like. Just like tech debt, if too much of it accumulates, you become overwhelmed by it, it becomes more and more difficult to move forward.

That doesn't mean you never do it, but you should acknowledge the costs and only *choose* to do it when the sacrifice is worth it.

4. Don't measure the thing. Measure the benefit.

GDS

So we're thinking about solving problems rather than building things. And if our measures of progress have, up until now, been based on "how much of the thing have we built", we could probably improve on that.

Let's run an example of how we'd do that, and why it's helpful.

Objective

*Migrate applications
to new Kubernetes
platform*

Key Results

- We migrated all of the things

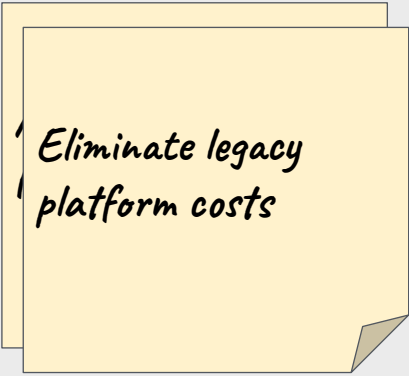
Objectives and Key Results (OKRs) - this is how our teams measure success at GDS. When we understand what our stakeholder needs, the team gets together to propose OKRs.

I'm taking a different example this time. We want to migrate applications off our old, creaky platform onto our new, shiny one.

Sounds exciting - yay!

But this could take months, or even years. How do we show that we're doing something valuable along the way? How do we choose which things to migrate first?

Objective



*Eliminate legacy
platform costs*

Key Results

- Migrate 100% of applications
- Reduce platform cost by £60,000/month
- What else...?

Again, we start by asking *Why*?

Why are we migrating the things? And by answering this, we can surface some more questions:

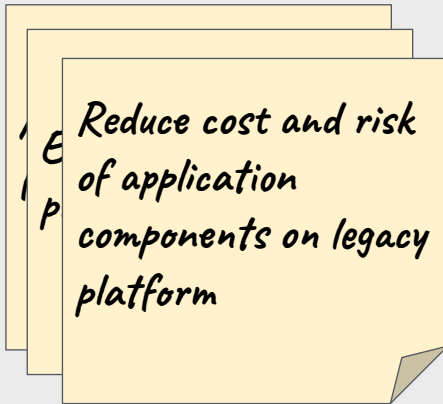
Is any value realised along the way, or does it only come at the end?

Does this help us choose which things to migrate first?

What else might we consider?

By using “100%” in the KRs, I’m reflecting that it’s not a binary “we succeeded or we failed, and if we failed we didn’t achieve anything” piece of work. (It’s also fair to say that shutting down the old platform and removing *all* the costs of supporting it is a binary. But it’s not the only value we deliver here.)

Objective



Key Results

- Migrate 100% of applications
- Reduce platform cost by £60,000/month
- Reduce risk score from critical to low
- Remove need for 24x7 support on legacy platform

That's better, but we can iterate a little further. Platform costs aren't the only costs here, there's also the cost of supporting the platform. And cost isn't the only reason we migrate away from old platforms; the older they get, often the riskier they become.

So now we can have informed conversations about which applications are the riskiest to leave on the old platform.

In our example (and in a recent real life example in one of my teams) it turns out not all the applications need 24x7 support. By prioritising the ones that do, we can get to a place where we don't need 24x7 support for the platform, before we've finished the migration work. No more on-call rota! That sounds like a win.

You might have your team develop the OKRs then propose them back to the stakeholders to make sure you've understood their needs, or you might work together on developing them. Either way, both the team and the people who benefit from the work need to be involved.

And this helps give us autonomy to figure out the best way to solve the problem. If you choose your OKRs well, that means *any* solution which achieves them is a good solution, and it might not be the thing you, or your stakeholders, first thought of.

Further reading on OKRs:

- *Measure What Matters* - John Doerr
- *Radical Focus* - Christina Wodtke

5. Get comfortable with uncertainty.

GDS

Another reason we set objectives this way is, life is uncertain. By focusing less on *the thing* we give ourselves flexibility to change direction, while still keeping the real objective in mind.

And we need to get comfortable with uncertainty. Things change. Things we thought we knew turn out not to be true.

Getting comfortable with uncertainty doesn't mean surrendering to it though. We don't have to live in a constant state of "dunno".

Accepting that we don't know the answers is a great place to start finding them out.

- **Feasibility risk:** can we build it?
- **Value risk:** will they buy it/choose it?
- **Usability risk:** will they be able to use it?
- **Viability risk:** does it meet our goals?

From *Inspired* - Marty Cagan

GDS

Here are some reasons your great idea, or your stakeholder's great idea, might not work.

At GDS, our SRE teams are cross-functional teams like any other product team. We have product managers, we get to work with interaction designers, content designers, performance analysts. The big advantage is we have access to people specialised in each of these types of risk.

You might not have those people in your teams, but if your organisation has a product management community, or a design community, for example, they might be happy to help out, even if it's just an informal chat about what you're trying to do. If these are challenges you care about and want to learn more about, think about getting a mentor from another community to help.

And a lot of technical PMs come from engineering as their background. If you really love these sorts of challenges, maybe a career change to PMing is for you.

1. What's our biggest unknown?
2. What's the cost of getting it wrong?
3. What's the cheapest/quickest way to learn more?

GDS

Getting your stakeholders comfortable with uncertainty is often easier than you'd think. We don't just say "we don't know", we talk candidly about the cost of getting it wrong, and set out the steps to reduce that risk by learning more

This is also the heart of agile.

Autonomy ✓

Mastery ✓

Purpose ✓

GDS

By having these more transparent conversations, by learning the real needs of your stakeholders, over time you get to a place where you understand each other better. You and your team stop being an order taker and become a trusted partner. You get to a place where other teams come to you with problems to solve, not things to build, and your teams can use their expertise to figure out how to solve them. They get more autonomy.

So you have your team of smart people, given the autonomy to use their expertise, the coaching techniques to get the best out of each other and build real mastery as a team, and inspired with a sense of direction and purpose. If you create those conditions, people will want to follow you. That's leadership.



So coming back to this image: it can feel like a daunting responsibility being the leader. But you don't have to know it all, you have the knowledge and skills of the team behind you, your job is to set the direction and give them what they need to get there.