Putting Your Platform on a Diet: Evolving Your TVP

Nicki Watt - CTO @ Trifork UK (Formerly OpenCredo)

OpenCredo

A TRIFORK COMPANY

Has now become ...

TRIFORK®

When last did you delete or deprecate anything in your platform?

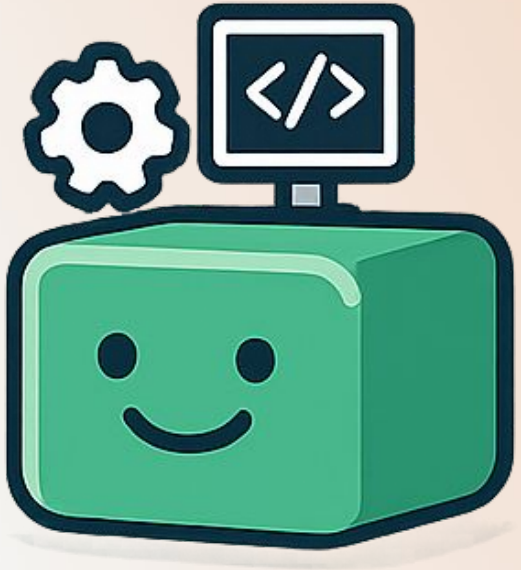
A. Bloaty McBloatface

What Is The Problem?

What’s the problem?

Thinnest Viable Platform (TVP)

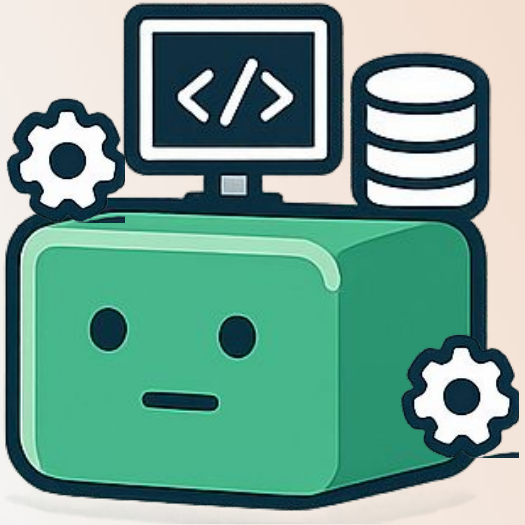
is all about giving teams just enough to move fast—without unnecessary complexity. And they often start out well!



What’s the problem?

But Platforms evolve over time …

Gathering with them, like moths to a flame, ever more well-intentioned additions, quick fixes, processes and tools.



What’s the problem?

Which if not actively planned & pruned

Can cause platforms to bloat!

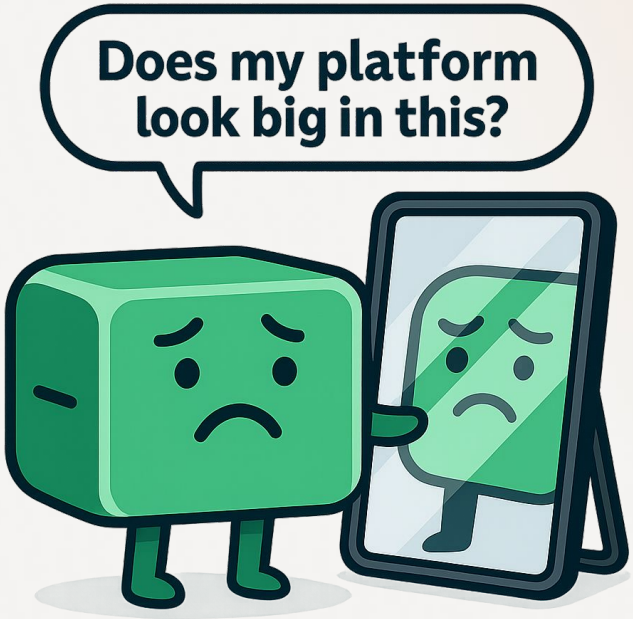
With every new addition having the potential to quietly nudge it from “helpful” to “hindrance.”

Where can bloat occur?

- Tools
- Services
- Processes

For both ***End Users & Platform Engineering teams*** alike!

The challenge ...



B. Signs of Bloat

What Does It Look Like?

SIGNS OF BLOAT

Bloat Indicators

1. Constant support requests to the platform team
2. Slow onboarding for new engineers
3. Teams bypass or work around the platform
4. Multiple tools solving the same problem
5. New platform features go unused

1. Constant support requests to platform team

1. Constant support requests to platform team

Is your platform accumulating "cognitive overhead": expecting more mental effort (bloat) from your users than it should?

2. Slow onboarding for new engineers

2. Slow onboarding for new engineers

Is your platform adding more friction, rather than reducing it for new teams & engineers?

3. Teams bypass or work around the platform

3. Teams bypass or work around the platform

Are you facing “platform abandonment” - when usage drops, but you’re still left with having to support and the surface area stays high?

4. Multiple tools solving the same problem

4. Multiple tools solving the same problem

Are your users facing “choice paralysis” - Providing too many options creating more confusion and less cohesion?

5. New platform features go unused

5. New platform features go unused

Are you accumulating “dead weight” - unused features adding a maintenance burden without delivering value?

Bloat Indicators

Platform bloat isn't just about how much code or tooling you have — it's also about how much additional friction you create.

C. Understanding Bloat

Why Do Platforms Bloat?

Understanding Bloat

1. Natural evolution of systems
2. Good intentions, never revisited
3. DIY / Commodity Dilema
4. Lack of planning and platform vision

Understanding Bloat

1. **Natural evolution of systems**
2. Good intentions, never revisited
3. DIY / Commodity Dilema
4. Lack of planning and platform vision

- Accumulation of tools, decisions & processes from different eras or teams
- Lost ownership and context

Understanding Bloat

1. **Natural evolution of systems**
2. Good intentions, never revisited
3. DIY / Commodity Dilema
4. Lack of planning and platform vision

Tips to address

- Good system for recording decisions (ADRs)
  a. LLMs - help search
- Revisit periodically - ensure reasoning still holds

Understanding Bloat

1. Natural evolution of systems
2. **Good intentions, never revisited**
3. DIY / Commodity Dilema
4. Lack of planning and platform vision

- One-off Quick fixes, POCs which live forever
- Over Engineering / Over Architecting

Understanding Bloat

1. Natural evolution of systems
2. **Good intentions, never revisited**
3. DIY / Commodity Dilema
4. Lack of planning and platform vision

Tips to address

- Label features & services (Quickfix / LTS Feature)
- Be explicit & realistic - Level of support offered

Understanding Bloat

1. Natural evolution of systems
2. Good intentions, never revisited
3. **DIY / Commodity Dilema**
4. Lack of planning and platform vision

- Build / Buy / Blend
- Inflexibility landing up on the extreme end of either:
  a. DIY instead of buy
  b. Buy instead of DIY

Understanding Bloat

1. Natural evolution of systems
2. Good intentions, never revisited
3. **DIY / Commodity Dilema**
4. Lack of planning and platform vision

Tips to address

- Blend
- Wardley Mapping
  → Help strategise
- Composable Modular Platform architecture

Understanding Bloat

1. Natural evolution of systems
2. Good intentions, never revisited
3. DIY / Commodity Dilema
4. **Lack of planning and platform vision**

Lack of:

- Platform coverage & scope
- Roadmapping
- EOL & Sunsetting policies
- Product Management
  → Local Optimisations

Understanding Bloat

1. Natural evolution of systems
2. Good intentions, never revisited
3. DIY / Commodity Dilema
4. **Lack of planning and platform vision**

Tips to address

Invest in:

- Platform coverage & scope
- Roadmapping
- EOL & Sunsetting policies
- Product Management
  → Global Applicability

Understanding Bloat

1. Natural evolution of systems
2. Good intentions, never revisited
3. DIY / Commodity Dilema
4. Lack of planning and platform vision
5. **BONUS: Hamstringing your platform team**

- Not protecting the team
- Not having the right kit
- Overly constrained processes & red tape

Understanding Bloat

1. Natural evolution of systems
2. Good intentions, never revisited
3. DIY / Commodity Dilema
4. Lack of planning and platform vision
5. **BONUS: Hamstringing your platform team**

Tips to address

- Platform Product Owner & Decision maker
- Invest in tooling which promotes fast feedback loops

Understanding Bloat

In the absence of vision, planning and pruning, all systems drift toward complexity.

D. Addressing Bloat

How can I fix or mitigate it?

Addressing Bloat

Prevention is better than cure!
Start early to avoid disappointment

Tips for Keeping Your Platform Lean

Mindset & Vision

- Lean & Platform As A Product Mindset
- Know your Target audience & boundaries
- Establish a Platform Charter you can refer back to

Tips for Keeping Your Platform Lean

Planning

- RoadMap
- Explicit EOL & Sunset Policies
  - https://endoflife.date/
- Don’t try and support everyone & every use case
- Only add features after real user feedback and clear use cases

Tips for Keeping Your Platform Lean

Architect for Evolution

- Composable Platform-as-a-Product capabilities.
- Plug-in-style architecture (where feasible)
- Support minimal platform modules, but allow for client extension

Tips for Keeping Your Platform Lean

Documentation

- Invest in good appropriate Docs
- Keep Track of Decisions (ADRs)

Conduct regular health checks

- Track feature usage & regularly review
- Monitor & Track Bloat Indicator Metrics

But for when you need to shed bloat …

Reduce The Support Surface

- Standardise: Only support a subset of versions & stacks
- Introduce & move users through the various EOL plans

Make as many DIY → Commodity shifts as possible

- Sheds complexity, maintenance & frictional bloat in one go

Addressing Bloat

Prevention is better than cure!
Start early to avoid disappointment

E. Conclusion

What do I need to remember?

Bloat Indicators

Platform bloat isn't just about how much code or tooling you have — it's also about how much friction you create.

Understanding Bloat

In the absence of vision, planning and pruning, all systems drift toward complexity

Addressing Bloat

Prevention is better than cure!
Start early to avoid disappointment

Stay Patient & Don’t Give Up!

There are generally no one off quick fixes, platforms needs continual and deliberate time & effort invested to keep them lean

THE END
Thanks!