Sustaining Fast Flow with Socio-Technical Thinking

Nick Tune || @ntcoding
Something has gone horribly wrong in this company.

All we're asking for is 2 text boxes to be put on a web page.... and it takes 3 months!!

Why does everything take so long here?

-- Head of Marketing
Why do tech companies turn from F1 cars into old bangers?

How it started....

How it's going...

time to put text box on web page: 1 hour

time to put text box on web page: 3 months

@ntcoding
When flow drops off, the whole company is affected

- Context: company has grown massively but flow has dropped off massively and product is becoming unreliable (downtime, bugs, etc)
- Heading to engineering off-site in Indianapolis in 2017 (~200 engineers)
- Purpose: reflect on engineering culture and practices, identify improvements to help become a modern engineering organization
You engineers need to do a better job. We've got a lot of angry customer and it's embarrassing us.

Software engineering is not a 9 - 5 job. You need to work harder and for as long as it takes [to get the work done on time and to a high standard].

-- CEO
Sustainable Fast Flow @7digital (2012)

- 25 deploys to production per-day across 6 teams
- First day: pair programming and TDD with a senior engineer, one-click deploy to production
- 2 years later: speed had improved - more teams, and more deploys per-team
Sustainable Fast Flow at Scale

Companies in a variety of industries are achieving sustainable fast flow

- Big UK supermarket
- Travel Company
- Advertising
- Restaurant bookings
- FinTech

2015
How to Achieve Sustainable Fast Flow

- Incentivising Sustainable Fast Flow
- Decoupling for Sustainable Fast Flow
- Platforming for Sustainable Fast Flow

Socio-Technical Thinking
Part 1: Incentivising Sustainable Fast Flow
People who go home on time don't become managers in this company.

-- Head of Ops
I have a reputation for shouting at people. Don't take it personally.

And besides, if I don't shout at you, the CEO will shout at you louder.
They will have to go!

You're not allowed to have whiteboards near your [dev teams] desks. It looks too messy.

No sticky notes on the walls either.

-- Head of HR
Why does nothing ever get delivered in this company?

We've had enough.

You [CEO] have 6 months to deliver {project} or you're fired.

-- Investors
Flow-Destructive Engineering Culture

- Always a rush to hit next arbitrary deadline
- Engineers constantly being "stolen" from one project to work on another
- Engineers called "resources", seen as numbers in a spreadsheet
- Single Jira workflow every team has to follow
- CTO enjoys regularly shouting at people

@ntcoding
We've got an extremely irate sales rep on the phone. His customer is complaining that the platform is malfunctioning, and he can't find the team that owns the relevant API...

...because nobody owns the API, it appears to have just been abandoned.

-- Engineer
These behaviours... incentivise..... which results in

Software Engineers

"rotated around different projects"

Forced to use dumb Jira workflow

Shouted at by CTO

"resources"

Messy, unreliable, unloved IT systems

Catastrophically blocked flow

@ntcoding
Our engineers have OCD.

They're always waffling to me about rewriting systems and fixing technical debt. They need to focus more on delivering results.

-- CEO
Always Looking for a Quick Fix

We don't have time to modernize our systems and change our engineering culture.... let's try this agile framework instead and hire some consultants.

The only quick fix is to travel back in time and incentivise sustainable socio-technical practices.
Incentivising Sustainable Fast Flow @7digital (2012)

Incentivised

**Technical**
- Deploying to prod every day for 2+ years
- Few production bugs
- One-click deployments
- High test coverage
- Well-designed software

**Social**
- Encouraged to build sustainable software by leadership
- Time during working hours to learn
- TDD, pair programming
- Go home on time every day
- You build it, you run it
- Teams own their process + continuous improvement
- 2 days innovation time per-month

Sustainable, company-wide fast flow

@ntcoding
Debunked: Avoid These Management Fallacies

Management fallacy: easy to move people to different teams if the process is the same

Management fallacy: easy to measure and compare teams if they use the same Jira workflow

Good teams own and continuously improve their ways of working and processes
Incentivising Sustainable Fast Flow

Good companies turn F1 cars into spaceships, not old bangers.

How it started....

Two years later....

@ntcoding
Part 2: Decoupling for Sustainable Fast Flow
Our teams are inefficient because they are organized in completely the wrong way.

We hired you to tell us what all our value streams are so we can re-org all of our teams.

Why is it taking so long?

-- CEO
Team boundaries do matter, but....

Team organization has a big impact on flow. Finding the right team topology matters.

You can't just re-org your whole company overnight. Software can't instantly be re-architected to support the new org-structure.

Identifying all your value streams is not a 2-day workshop. It takes a lot of time and effort and should be done gradually.
Teams and architecture relationship: It's complicated

Sustainable technical practices make your architecture more supple to support org changes
Align Socio-Technical Architecture with True Business Domains

A subject area in which an organization gains understanding of user needs, builds expertise, and develops capabilities.

**Examples:** Tax Calculation, Journey Planning, Treatment Discovery, Appointment Booking.
Industry Example: Hidden Domain, Experiences Industry

- **Problem**: multiple teams frequently involved when implementing new changes - high cost of coordination

- **Problem**: manual processes affecting 30+ people spread across nearly all teams - major time sink and source of frustration

- **Diagnosis**: hidden domain - different parts owned by different teams

- **Solution**: Decouple hidden domain, spin up a new team to own it, reduce operational complexity

@ntcoding
**Strategy:** Decouple for Fast Flow in Core Domains

- **Step 1:** Consolidate hidden core and reduce operational and technical complexity
- **Step 2:** Exploit big opportunities in core domains now that changes will be far easier (thanks to improved flow created by step 1)
**Domain-aligned Decoupling:** Socio-Technical Benefits

**Social**
- **Purpose:** team is responsible for discovering and delivering value in a domain. Motivated teams have faster flow.
- **Expertise:** team gains domain knowledge and work closer with domain experts, reducing time to implement new domain concepts.
- **Incentivised:** team will be responsible for code in the long-term, incentivising sustainability.

**Technical**
- **Domain-aligned code:** easier to understand code and incorporate new requirements - minimal translation costs.
- **Low coupling:** business concepts that change together live together in code, reducing technical and social dependencies.
Decoupled, Empowered Product Teams Create Better Products

bit.ly/3vNjgHw

@ntcoding
Decoupling: Techniques for Identifying True Domains

Event Storming

Domain Message Flow Modelling

Independent Service Heuristics

1. Sense-check: Could it make any logic?  
   - Is this thing independent enough?
   - Would consumers understand or use it?
   - Would it simplify execution?

2. Brand: Could you imagine this thing if...?  
   - Would it be a viable business (or not)?
   - Would it be a compelling offering?
   - Could a marketing campaign be a success?

3. Streams/Customers: Could this thing exist?
   - Would it be a viable business (or not)?
   - What would a subscription cost?
   - Is there a clearly defined customer?

4. Cost tracking: Could the organisation deliver on these things?  
   - Are the full costs of running this thing?
   - What are the data storage costs? Data transfer costs?

github.com/ddd-crew/eventstorming-glossary-cheat-sheet

github.com/ddd-crew/domain-message-flow-modelling

github.com/TeamTopologies/Independent-Service-Heuristics
Part 3: Platforming for Sustainable Fast Flow
Industry Example: Financial Services

- **Locked down**: 3 ops tickets to install docker on dev laptop
- **Banned**: Developers not allowed to download NPM packages at all
- **Forbidden**: Developers don't have access to production logs
- **Disaster**: 1 year - nothing delivered, lots of pressure
- **Cognitive load**: devs overloaded with infrastructure and bureaucracy
Industry Example: UK Gov, HMRC (2016)

- **Digital Platform (MDTP):** superb paved road and developer experience
- **Self-service:** new microservice or website all the way to production, with environments, deploy pipelines etc and great documentation
- **Continuous delivery:** metrics, monitoring, logging... out of the box support for daily deployments and You Build It, Your Run It

@ntcoding
Learn More: MDTP + Platform Playbook

Welcome to Her Majesty's Revenue and Customs (HMRC) on GitHub. Here you can learn about the HMRC Multichannel Digital Tax Platform (MDTP), the citizen and company digital tax services hosted on the MDTP, and the Agile principles and practices that under...

Disclaimer: I do not work for EE, I simply think this is good platform content which I can largely verify from personal experience
Platforms: Key to Sustainable Fast Flow (especially at scale)

"[Platforms] Enable stream-aligned teams to deliver work with substantial autonomy.

The stream-aligned team maintains full ownership of building, running, and fixing their application in production.

The platform team provides internal services to reduce the cognitive load [of stream-aligned] teams"

@ntcoding
You have to work on their [Platform Team's] terms.

Don't go to them with what you want, they decide on the solution.

If they don't want to talk, you go away and come back when they are ready.

-- Director, Travel Industry
They [ops people] won't accept that idea [platform teams are there to make software engineers successful].

They've always seen their job as delivering successful outcomes [and being superior to devs].

-- CIO, Financial Services
Successful Platforms Require Socio-Technical Thinking

- Mindset: help you be successful
- Reduce cognitive load
- Developer Experience
- Surveys
- Platform-as-a-product
- Joint retrospectives

Digital Platform Team

Stream-aligned Teams
Developer Experience (DX): Indicators of Sustainable Fast Flow

- **New service**: how long to get a new production-grade microservice all the way to production? 1 hour is good

- **Laptop to production**: how long between writing a line of code locally and putting it in production? 10 minutes is good

- **Time-to-productivity (new hire)**: deploying to production from own laptop on 1st day is good

- **Platform Ops tickets**: 0 is good
Don't tie your software developers up in red tape and bureaucracy.... roll out the red carpet if you want sustainable fast flow*

* this does not mean developers are special and should be treated better than anyone else
Wrap-up:
Socio-Technical Symbiosis
Symbiosis:
A close long-term association between two or more species in nature.

Example: Gut Microbiota living in human digestive system
Socio-Technical Symbiosis: Your Choice

**Mutualism**

*Mutually-beneficial*

Turn your F1 car into a spaceship

**Parasitism**

*One lives-off and harms another*

Turn your F1 car into an old banger
Socio-Technical Symbiosis

**Socio-Technical Mutualism: 3 Key Steps for Sustainable Fast Flow**

**Incentivise**
- go home on time every day
- time during work to learn and improve
- quality technical practices are essential

**Decouple**
- align socio-technical boundaries with true business domains
- map out your business collaboratively
- avoid big re-orgs

**Platform**
- reduce cognitive load for stream teams
- roll out the red DX carpet to eliminate blockers
- platform-as-a-product

@ntcoding
Socio-technical thinking leads to socio-technical mutualism, leads to sustainable fast flow, leads to happier employees and better products
What's stopping you introducing more sustainable, socio-technical thinking into your organization?

What could you start doing differently tomorrow?

What happens if you do nothing?

@ntcoding