TEAM TOPOLOGIES

Matthew Skelton, Manuel Pais

Foreword

"Team Topologies describes organisational patterns for team structure and modes of interaction, taking the force that the organisation exerts on the system as a driving design concern"

"Notably, it identifies four team patterns, describing their outcomes, form, and the forces they address and are shaped by"

Preface

"This book offers a practical, step-by-step, adaptive model for organisational design that we have used and seen work across businesses of varying levels of maturity: Team Topologies"

Teams as the Means of Delivery

"Systems thinking focuses on optimising for the whole, looking at the overall flow of work identifying what the largest bottleneck is today, and eliminating it. Then repeat"

"Team Topologies provides provides four fundamental team types - stream-aligned, platform, enabling and complicated-subsystem - and three core team interaction modes - collaboration, X-as-a-Service, and facilitating"

"Team structures must match the required software architecture or risk producing unintended designs"

".. and "Inverse Conway maneuver", whereby and organisation focuses on organising team structures to match the architecture they want the system to exhibit..."

"When we talk about cognitive load, it's easy to understand that any one person has a limit on how much information they can hold in their brains at any given moment"

"This requires restricting cognitive load on teams and explicitly designing the intercommunication between them to help produce the software - systems architecture that we need (based on Conway's law)"

Conway's Law and Why it Matters

"The way teams [traditionally] are set up and interact is often based on past projects and/or legacy technologies (reflecting the latest org-chart design, which might be years old, if not decades)"

"Organisation design using Conway's law becomes a key strategic activity that can greatly accelerate the discovering of effective software design and help avoid those less effective"

"Our research lend support to what is sometimes called the "inverse Conway maneuver", which states that organisations should evolve their team and organisational structure to achieve the desired architecture"

"Architecture thus becomes an enabler, not a hindrance, but only if we take a team-first approach informed by Conway's law"

"Conway's law suggests that this kind of many-to-many communication will tend to produce monolithic, tangled, coupled, interdependent systems that do not support fast flow. More communication is not necessarily a good thing"

"One key approach to achieving the software architecture (and associated benefits liske speed of delivery or time to recover from failure) is to apply tre reverse Conway maneuver: design teams to match the desired architecture"

Team-First Thinking

"Therefore, an organisation should never assign work to individuals; only to teams. In all aspects of software design, delivery, and operation, we start with the team"

"The limit on team size and Dunbar's number extends to groupings of teams, departments, streams of work, lines of business, and so on"

"The danger of allowing multiple teams to change the same system or subsystem is that no one owns either the changes made or the resulting mess"

"Teams may use shared services at runtime, but every running service, application, or subsystem is owned by only one team"

"Broadly speaking, for effective delivery and operations of modern software systems, organisations should attempt to minimise intrinsic cognitive load (through training, good choice of technologies, hiring, pair programming, etc.) and eliminate extraneous cognitive load altogether (boring and superfluous tasks and commands that add little value to retain in the working memory and can often be automated away), leaving more space for germane cognitive load (which is where the "value add" thinking lies)"

"Tip: Minimise cognitive load for others - is one of the most useful heuristics for good software development"

Static Team Topologies

"The first anti-pattern is ad hoc team design. The other common anti-pattern is shuffling team members"

"On the contrary, organisations that expose software development teams to the software running in the live environment tend to address user-visible and operational problems much more rapidly compared to their siloed competitors"

"The DevOps Topologies reflect two key ideas:

1. There is no one-size-fits-all approach to structuring teams for DevOps success
2. There are several topologies known to be detrimental (anti-patterns) to DevOps success. In short, there is no "right" topology, but several "bad" topologies for any one organisation"

"The Cloud team might still own the provisioning process - ensuring that necessary controls, policies, auditing are in place (especially in highly regulated industries)"

"However, without a clear mission and expiration for such a DevOps team, it's easy to cross the thin line between this pattern and the corresponding anti-pattern of yet another silo (DevOps team) with compartmentalised knowledge"

The Four Fundamental Team Topologies

"... four fundamental team topologies are: stream-aligned, enabling, complicated-subsystem, platform"

"A stream-aligned team is a team aligned to a single, valuable stream of work; this might be a single product or a service, a single set of features, a single user journey, or a single user persona"

"The stream-aligned team is the primary team type in an organisation, and the purpose of the other fundamental team topologies is to reduce the burden of the stream-aligned teams"

"An enabling team is composed of specialists in a given technical (or product) domain, and they help bridge this capability gap. Such teams cross-cut to the stream-aligned teams and have the required bandwidth to research, try out options, and make informed suggestions on adequate tooling, practices, frameworks, or any of the ecosystems choices around the application stack"

"A complicated-subsystem team is responsible for building and maintaining a part of the system that depends heavily on specialist knowledge, to the extent that most team members must be specialists in that area of knowledge in order to understand and make changes to the subsystem"

"The purpose of a platform team is to 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 cognitive load that would be required from stream-aligned teams to develop those underlying services"

"The change from infrastructure team to platform team in not simple or straightforward, because a platform team in managed as a product using proven software development techniques that might be quite unfamiliar to infrastructure people"

"Existing teams based on technology component should be either dissolved, with the work going into stream-aligned teams or converted into another team type"

"For example, DBA teams can often be converted to enabling teams if the stop work at the software-application level and focus on spreading awareness of database performance, monitoring, etc"

"Likewise, middleware teams can also be converted into platform teams if they make those parts of the system easier to use for stream-aligned teams"

"The most effective pattern for an architecture team is as a part-time enabling team (if one is needed at all)"

"A crucial role of a part-time, architecture-focused enabling team is to discover effective API's between teams and shape the team-to-team interactions with Conway's law in mind"

Choose Team-First Boundaries

"Typically, little thought is given to the visibility of boundaries for teams, resulting in a lack of ownership, disengagement, and a glacially slow rate of delivery"

"In summary, the business domain fracture plane aligns technology with business and reduces mismatches in terminology and "lost in translation" issues, improving the flow of changes and reducing rework"

"Splitting along the regulatory-compliance fracture plane simplifies auditing and compliance, as well as reduces the blast radius of regulatory oversight"

"Another natural fracture plane is where different parts of the system need to change at different frequencies. With a monolith, every piece moves at the speed of the slowest part"

"Splitting off subsystems with clearly different risk profile allows mapping the technology changes to business appetite for regulatory needs"

"There are situations where splitting off a subsystem based on technology can be effective, particularly for systems integration older or less automatable technology"

"We need to look for natural ways to break down the system (fracture planes) that allow the resulting parts to evolve as independently as possible. Consequently, teams assigned to those parts will experience more autonomy and ownership over them"

Team Interaction Modes

"... three essential ways in which teams can and should interact, taking into account team-first dynamics and Conway's law: collaboration, X-as-a-Service, facilitating. A combination of all three team interactions modes is likely needed for most medium-sized and large enterprises"

"Collaboration: working closely together with another team"

"The collaboration team mode is suitable where a high degree of adaptability of discovery is needed, particularly when exploring new technology or techniques"

"X-as-a-Service: consuming or providing something with minimal collaboration"

"The X-as-a-Service team interaction mode is suited to situations where there is a need for one or more teams to use a code library, component, API, or platform that "just works" without much effort, where a component or aspect of the system can be effectively provided "as a service" by a distinct team or group of teams"

"Facilitating: helping (or being helped by) another team to clear impediments"

"The facilitating team interaction mode is suited to situations where one or more teams would benefit from active help of another team facilitating (or coaching) some aspects of their work"

"To address the problem of poorly drawn service [X-as-a-Service] boundaries, the collaboration team mode can be used to help redraw service boundaries in a new place"

Evolve Team Structures With Organisational Sensing

"Deliberately changing the interaction mode of two teams to collaboration can be a powerful organisational enabler for rapid learning and adoption of new practices and approaches"

"Ultimately, be expecting and encouraging team interactions to move between collaboration and X-as-a-Service for specific reasons, organisations can achieve agility"

"The aim should be to try to establish a well-defined and capable platform that many teams can simply user as a service"

"A key aspect of this sensory feedback is the use of IT operations teams as high-fidelity sensory input for development teams, requiring joined-up communications between teams running systems (Ops) and teams building systems (Dev)"

"As we saw in Chapter 5, one of the simplest ways to ensure a continued flow of high-fidelity information from Ops to Dev is to have Ops and Dev on the sam team, or at least aligned to the same stream of change as a stream-aligned pair of teams, with swarming for operational incidents"