Conway's Law

Idea In Short

Conway's law is an observation that the design of any system is significantly affected by the communications structure of the organization that develops it. The law is commonly associated with software development but is considered applicable to systems and organizations of all types.

Origin

Many years ago, Melvin Conway observed that the way organizations are structured impacts any system they create. In his article 1, he wrote:

> Any organization that designs a system (defined more broadly here than just information systems) will inevitably produce a design whose structure is a copy of the organization's communication structure.

At the time, the Harvard Business Review rejected his original paper because he hadn't proved his hypothesis. The paper was published in April 1968 in Datamation, the leading IT magazine of the time. The law became well-known after Fred Brooks cited the article and the idea in his classic book The Mythical Man-Month, calling it Conway's Law. The name stuck and spread, especially among software developers.

> Conway's law was not intended as a joke or a Zen koan, but as a valid sociological observation. It is a consequence of the fact that two software modules A and B cannot interface correctly with each other unless the designer and implementer of A communicates with the designer and implementer of B. Thus the interface structure of a software system necessarily will show a congruence with the social structure of the organization that produced it.

Brooks recognized that the law has important corollaries in management theory:

> Because the design that occurs first is almost never the best possible, the prevailing system concept may need to change. Therefore, flexibility of organization is important to effective design.

Despite the author's remark - to apply this not just to information systems - a lot of research that validated this principle came from software developers2, including Satya Nadella's own hypertext and the transformation at Microsoft3:

> To truly get the best impact from our efforts, we will have to push ourselves to transcend Conway's law.

In many ways, the law asserts and validates that software is the product of an organization's communication structure.

HBS Study

Harvard Business School conducted a study4 of different codebases to see if it could prove Conway's original hypothesis as applied to software systems. In it, they took multiple examples of software created to solve the same purpose (for example word processing, financial management and database software), and compared the code bases created by loosely-coupled open source teams, and those created by tightly-coupled teams.

Their study found that the often co-located, focused product teams created software that tended more towards tightly-coupled, monolithic code bases. Whereas the open source projects resulted in more modular, decomposed code bases.

> A good example of Conway's Law was found in 1999 by UX expert Nigel Bevan as he studied corporate website designs. Bevan found that end-users often became frustrated with company websites because they were slow and difficult to use as the navigation, page design, and content structure often mirrored the internal concerns of an organization—instead of meeting the needs of the intended user first.

This has led many organizations to experiment with more distributed teams agile methodologies. But at the same time, there is still a consideration that a group of disconnected programmers could infuse their work with their individual biases.

Concepts

Homomorphism

Also called Isomorphism, this law underlies Conway's Law:

> Homomorphism is a structure-preserving map between 2 structures.

Homomorphism is the power of one system or structure to manifest itself in another system. The force creates the same structure. The structures copy each other. It preserves the structure. In simpler terms, a system replicates recognizable elements of the organization that developed it. In simpler terms, this is the Mirroring Effect, by which a system produces a mirror image of the organization that made it.

Reverse Conway's Law

The previous paradigm leads to what Allan Kelly defined as the Reverse of Conway's law:

> Organisations with long-lived systems will adopt a structure modelled on the system. Organisational design is system design. Architect the organization to architect the system.

For a few years, organizations have understood this link between organizational structure and software they create. Correspondingly, they have been embracing new structures to achieve the outcome they want.

> For example, Netflix and Amazon for example structure themselves around multiple small teams, each one with responsibility for a small part of the overall system. These independent teams can own the whole lifecycle of the services they create, affording them a greater degree of autonomy than is possible for larger teams with

> more monolithic code bases. These services with their independent concerns can change and evolve separately from one another, resulting in the ability to deliver changes to production faster. If these organizations had adopted larger team sizes, the larger monolithic systems that would have emerged would not have given them the same ability to experiment, adapt, and ultimately keep their customers happy.

For the moment, it suffices to say that multiple researchers have validated these principles in various fields, not just in software design.

Importance

Once we move away from software products into something broader, we can discover several relationships with concrete cases:

- We have seen already how a massive aeroplane accident has resulted from a faulty communication structure in an organization in theory designed to be entirely focused on safety and the importance of engineering.
- Several large scandals in organizations (think of Enron) have resulted from conscious organizational decisions that ultimately reflected the failure of an entire system.

Conway's law is becoming increasingly relevant today because the acceleration aspect of VUCA and incomprehensible of BANI makes all these aspects more visible.

Case Study: Retail

Physical stores and e-commerce sites deeply reflect how organizations are structured and their communication structure5. Think of apparel: few cases come immediately to mind.

> Apparel stores are mainly organised across gender (men/women) differentiations and age clusters (kids/adults). In most cases, this reflects a distinction that runs deep into the organization, from style to product development, to merchandising and sourcing. This distinction immediately creates issues the moment genderless fashion comes into play and limits blur: small sizes, another area grouped under kids, even if adults also buy them.

> So-called omnichannel services. Ever tried to buy online and return something in store? I'm sure you all have experienced at least a pair of rolled eyes by the chair when asking for this service. The issue is that most organizations are still organized by channel; thus, returning to a physical store means that something sold online will end up in the stock count of a physical store. In most cases, business control mechanisms, resource allocation, incentives all are still by channel, despite the promise of an omni-channel experience.

All these examples reflect how broad the implications of Conway's Law are, especially if we intend the concept of system in its most enormous possible sense, precisely as Conway himself intended.

Corollary to Conway's Law

To better understand Conway's law, we should appraise all the critical aspects and outputs that an organization produces and the experience it delivers to its customers. The quality of customer experience delivered by an organization will inevitably be the product of its organizational design.

A few years ago, in his book How Google Works, Eric Schmidt (Schmidt, 2014) casually referred to the outputs of Conway's Law:

> You should never be able to reverse engineer a company's organizational chart from the design of its product.

Common Root Causes

In reality, software is often a very complex subject, and producing software is an incredible challenge that requires an understanding of psychology, art, design, digital logic, data relationships, systems, tooling, leadership, teamwork, and much more. At least one person within a company tasked with building software must have a clear map of what this software is, what it will be, how it works, and how to make it work better. This person is generally the CEO, CTO, VP of Engineering, or some other high-level person of authority, and must do a clear job of communicating requirements and other important pieces of information

throughout the rest of the organization.Common issues resulting from poor communication between members of a software team include6:

- **Propagation delay:** Information takes far longer than expected to travel from those who have it to those who need it. Sometimes, a piece of information is no longer relevant by the time it reaches the intended recipient. This results in wasted effort and can drag out simple tasks and projects for indefinite amounts of time
- **Blind leadership:** When leaders are slow to listen and fast to reply, they miss out on critical feedback from team members, customers, and others with insight they don't have. There is no way one person to possibly see / hear / parse more information than an entire team of people, regardless of how intelligent this person may be. Not listening to feedback is arrogant and generally has a very negative impact on the team and therefore the final product (or products) as a whole
- **Elitism:** Not sharing information in a useful manner between members of a team can result in a very one-sided distribution of information across the team, meaning that one individual or group may have more information than they even need while another is doing without some critical piece of info that could easily be obtained from another part of the group. This can result in an us vs them scenario in which one group holds all the cards, and is obviously bad for everything from morale to overall product quality
- **Confusion:** Poor communication obviously leads to confusion, which often scales out into mass widespread lack of understanding, with different people having different ideas of how something is supposed to be done. It becomes much easier for people to blame or scapegoat each other, or for poor managers to hide their shortcomings behind team members (or blame the client / customer / pet dog / whoever else)
- **Wrong Target:** Hitting the bullseye on the wrong target is a common problem within the field of software development, and is one of the most devastating events one can experience in their career. Nothing is quite like riding the excitement of building a great product for weeks or months only to find out that you built the wrong thing altogether and have to start over

Cons of Conway's Law

Companies of all sizes and in all sectors have utilized and seen success from adopting Conway's Law and started utilizing small agile teams in their system designs. From Google

to the financial sector, companies in every industry have adopted Conway's Law as a way to spur innovation. Essentially, you devote a small team of closely knit individuals to a single project, let them iterate, and you'll get the most creative solutions. The notion has often been that Dunbar's Number applies to work relationships as well as personal ones: there's an upper limit of 150 people that can collaborate effectively in an organization. More than that and communication breaks down.

However, at the same time, this concept is also earning its own detractors.

A Forbes article7 from 2017 noted there are potential drawbacks to letting Conway's law guide the structure of your organization. The thinking goes that:

> Once you entrench small teams in this way, their respect and loyalty for that team often comes to outweigh their allegiance to the organization as a whole... Teams in disparate locations end up forming strong but exclusive identities as individual departments.

It's good for teams to form strong bonds, but those bonds should not become exclusionary and cliquish; otherwise, the organization loses its internal cohesion. As cited in the article:

> The result is sort of like the movie Mean Girls, where everyone is in their own clique and can look derisively at those who are not a part of it. McKenty called this blinkered mindset Not Invented Here Syndrome, and he offered three ways it appears in businesses: through geographic barriers, organizational boundaries, and with regard to domain expertise. Teams in disparate locations end up forming strong but exclusive identities as individual departments. And then, even on the micro-level within teams, you can have domain expertise keeping people apart, with people being unable to communicate effectively because one is a DBA and another is a developer.

As with most things, the answer to designing a team is not as clear cut as applying a popular observation made over 5 decades ago. There are businesses that have stretched the boundaries of what is possible by using small independent teams to create fast-paced changes, but even these smaller teams can still suffer from their own internal neurosis and

communication challenges.

Once things like insecurity, distrust, and fear start creeping into the mindset of a team, it's Conway's law that reminds us these same traits will seep into the systems these teams create. Once this happens we have to get these negative characteristics under control by embracing proper communication within a team or organization while also encouraging others to do so no matter how a team is scaled.

Summary

Conway's Law is a little-understood yet incredibly powerful concept that describes how deeply connected an organization's communication structure is with the final product or service it produces. Embrace proper communication within your team or organization, and encourage others to do so. If you're in a leadership position, start making changes wherever you find they are necessary, and understand that investing in the communication and documentation within your organization has a direct return on investment due to higher accuracy and reduced effort required, not to mention a reduction in frustration or burnout that can occur when people with information aren't being listened to. If you're not in a leadership position, start educating others on the importance of proper communication and you'll likely find yourself moving up in a world where idle chatter is the norm and focused, directed effort is the exception.