Thinking Distributed

Published on May 22, 2018

There's an incredible number of Lego pieces that we use today for building modern, distributed systems. Why do we do it this way and how do we manage the complexity that comes with it? Surprisingly, the toughest nuts to crack are often not the engineering problems.

I enjoy sitting on the balcony in the evening, listening to music. Recently I've started encountering issues where the playback keeps pausing randomly. Trying to solve the issue is somewhat like a murder mystery game - is it the Spotify app itself? The Samsung TV where the app runs? Bluetooth? The Wifi? Facebook Connect? Google Cloud which Spotify uses? Back in the days, the problem would have been either a spoilt CD, or the player had to be sent for repairs.

The world has changed

Modern systems are built in a modular way. Classic architecture designs don't achieve the **scale** we need and they're not **flexible** enough to keep up with the increasing pace of **innovation** and complexity around them. Building websites was a lot easier twenty years ago with Netscape Navigator and Internet Explorer on PCs being the only plausible options - but then think about some of today's incredible apps and services that would have been impossible back then. In 1998 we were able to categorize most of the web and add links into

a nicely organized portal, today we're consulting an omniscient 'oracle' called Google that becomes more intelligent *because* we're constantly using it.

So we know the world is not as it was before, and the solutions we build have many moving parts, each one at a different speed and over many of those building blocks we have no control at all. How do we make sure the music keeps playing?

entropy
/ˈɛntrəpi/

noun
noun: **entropy**; plural noun: **entropies**; symbol: **S**

1. PHYSICS
   a thermodynamic quantity representing the unavailability of a system's thermal energy for conversion into mechanical work, often interpreted as the degree of disorder or randomness in the system.
   "the second law of thermodynamics says that entropy always increases with time"
2. lack of order or predictability; gradual decline into disorder.
   "a marketplace where entropy reigns supreme"

The Modern Stack

Let's talk about the good things first - with a modular design we're breaking up the big problem into many small ones. In many cases, we can simply pick and choose modules & services that already exist and perform a particular task extremely well, patch them together in a programming language of our choice, add some secret sauce, run our mashup on the cloud and distribute it via a mobile app store. Almost like Gutenberg's moveable type and the letterpress. Today, anyone can publish a book - and on a zero budget if needed, since this can be done entirely online.

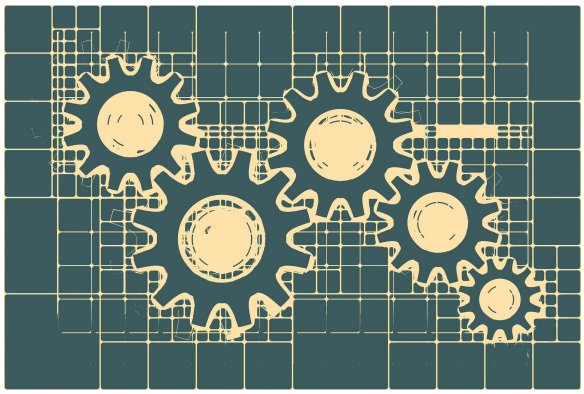
Just take Three

So what do we need in order to build something that gives us all those benefits, while keeping a reasonable level of control and predictability? We've already discussed the premise of a 'Divide & Conquer' approach - building an open, modular foundation with interchangeable Lego pieces, you choose the best one for each, and mix & match between Open Source, proprietary tools or components that you would like to or have to maintain yourself. Everything else can be grouped into the following categories from a macro perspective:

1. **Agile Infrastructure** - containers, microservices, Platform-as-a-Service, the cloud.
2. **Data & Intelligence** - APIs, open standards, monitoring, telemetry, real-time insights
3. **Automation** - orchestration, automated builds, testing, deployment, auto-scale.

For all those, you'll find comprehensive coverage in technical articles and blog posts all over LinkedIn and the web. Let's instead talk about the *4th element*.

Mindset & Collaboration



This all sounds just fabulous, and the individual technical concepts, platforms & tools are so exciting that often we simply focus on the geek aspects and mostly ignore the human factor in all this. How do we collaborate in the new world? How do we deal with uncertainty? Continuously improve - as a team and as individuals? How do we tell senior, experienced people that the classic approach no longer works and that the tools they've been using for the longest time are outdated and soon no longer relevant?

- Project requirements might still emerge - and change - after the project has already started
- System failures *will* happen, so it's better to anticipate them and finding ways to recover quickly & improve
- We'll push code into production and only know then whether it actually works
- Sending Excel sheets via email will no longer do the trick. Scheduled sit-down meetings or conference calls are becoming outdated mechanisms to achieve collaborative progress [1]
- We'll need to engage and collaborate with people outside the corporate firewall a lot more than previously
- Our personal skillset needs constant review and learning.

Conway's Law says that every system reflects the communication structure of the organization that builds it. In a distributed world, it's about the *attitude* of every individual and the need for each of us to appreciate that building it is a collaborative effort.

[1] Yet - if we change those, we're introducing a whole new set of governance issues and processes that need to be reviewed