PLATFORM STRATEGY

Innovation Through Harmonization

Gregor Hohpe

An Architect Elevator Guide

With contributions by Michele Danieli
and Jean-Francois Landreau

Platform Strategy

Innovation Through Harmonization

Gregor Hohpe

This book is available at http://leanpub.com/platformstrategy

This version was published on 2024-11-23

Platforms enable rapid and incremental software delivery. It only seems appropriate to embrace the same principles when writing a book about platforms. That's why this book started out as a Leanpub book with early releases that improved based on reader feedback. You are now reading the result of this iterative process.

Contents

About this Book

My prior role as chief architect of a global financial services organization placed me in the middle of several major transformation programs. A major goal of those programs was to accelerate software delivery without compromising quality or security—two aspects that were believed to be at odds with each other. Undeterred by existing beliefs, we set out to deploy something called *Agile Delivery Platform*, which made software delivery "fast and compliant" and placed the broader IT organization on a trajectory toward a modern operational model.

Reflecting back on our work, several important questions remain. Did we truly build a platform as the name suggested, or were we just naming it creatively? Assuming the former, which characteristics or capabilities qualified it as a platform? Could we have achieved the same results without a platform? Was building on a third-party product the key aspect of providing a platform? Would any "agile delivery mechanism" inherently be a platform? Were related initiatives like the "digital backbone" or the "digital core" also platforms but simply didn't include the term in their name?

Deep in our minds, we might have had plausible answers to those questions, but articulating them clearly would have been rather difficult. I freely admit that we chose the name without deep consideration; we mainly wanted to express both the project's purpose (to enable agile delivery) and the mechanism (a common platform for all software projects). Both "Agile" and "Platform" also sounded like noble and useful things to us.

A few years later, I benefited from two advancements: *Internal Developer Platforms* had become a "thing", and I helped build one for the Singapore Government's GovTech division. Making software delivery more efficient across government agencies, we were convinced that we were actually building a platform, perhaps helped by the inherent fuzziness of the term. However, diving deeper into the design trade-offs of developer platforms also raised a slew of new questions and design decisions that had to be made. Running our internal platform on top of commercial cloud platforms only added to the list of questions: what aspects should be part of our platform versus the

cloud platform? What if the cloud launches a feature that makes our function obsolete? Who should own the services that our platform provisions? Should our platform handle billing?

Eager to get to the root of it all, I joined a major cloud provider and spent several years working with customers to get the most out of that platform. Although one might expect that powerful cloud services would have made building in-house IT platforms superfluous, I observed exactly the opposite across large customers; they were building or deploying some sort of in-house platform, or often multiple platforms: developer platforms, data platforms, business platforms, digital platforms, multicloud platforms, and many more. And, unsurprisingly, they faced many of the same questions about their platforms.

After assisting many customers with their platform strategy, I took the Architect Elevator further down into the engine room to develop serverless products, which are widely considered the canonical example of a modern compute platform. So, once again I found myself deep in the mechanics of making platform decisions: what technical details can be safely hidden from the user, and which ones should be exposed? How can services be independent while also being constituent parts of something that's bigger than just the sum of its parts? Which aspect should be common across services? Is duplicate functionality across services a convenience or a design flaw? It seemed that the deeper I dug, the more questions arose.

Expecting a single book to answer all questions about platforms would be naive. Instead, I set out, aided by formidable contributors, to structure the questions and define decision models that can help you answer those questions for yourself. After all, you have more information about your undertaking than I would ever have, and thus you can make better decisions. That's why you won't find simplistic 1-2-3 recipes in this book. Instead, if you're new to the concept, you will find a smooth on-ramp, whereas if you're experienced, I may test and challenge your current thinking with novel viewpoints.

The Lure of Platforms

Over the past decade, platforms have fueled some of the most successful business models, yielding companies with never-before-seen market valuations. Although they come in various shapes, all platforms share an "amplifier effect" that makes them enormously successful but deceivingly difficult to replicate.

Marketplace platforms get their boost from the "flywheel" effect: more sellers provide a wider selection, thereby attracting more buyers, which in turn attract more sellers. That's the mechanism that propelled platform powerhouses like eBay, Uber, and Airbnb to success and astronomical valuations. But the magic of the platform business model doesn't end there. None of those companies keep any expensive inventory, because the platform's value proposition doesn't depend on directly delivering goods or services but on facilitating an exchange across an ecosystem of third parties.

IT also loves a multiplier effect for its investments and therefore values platforms as a shared foundation on top of which diverse solutions can be built. Operating systems and cloud platforms hide underlying complexity, allowing developers to focus on delivering business value instead of toiling deep down in the engine room. The need for higher levels of developer productivity in fast-moving environments has made in-house platforms a staple of modern IT strategies.

Outside of IT and online marketplaces, platforms already have a fairly long history. The automotive industry, for example, has long employed platforms to better amortize the heavy engineering investments required to build a safe and reliable car. Because few of the engineering elements like engine, transmission, or suspension are visible to buyers, they can be reused across models that differ in styling and interior options to attract different tastes and demographics. Yet, equating platforms with the act of combining all common elements and placing differentiators on top would be far too simplistic.

Building IT Platforms

Lured by the apparent benefits and successes of both platform business models and in-house platforms, many enterprise IT departments discover that building and financing your own platform is far more difficult than it might have initially appeared. Many platform initiatives consume multiyear investments just to be already obsolete or not accepted by users by the time they launched.

Do such misfortunes invalidate the platform concept and value proposition? Surely not. But they highlight the value of a clear strategy and a deep understanding of the constraints and assumptions that are baked into the platform. Instead of leaving implementation details for later or, worse yet, with a provider, platforms want to be built incrementally with a keen eye on value delivery.

This book lays out the elements of a platform strategy, highlighting critical decisions and trade-offs to help platform teams become innovation drivers in enterprises.

What Will I Learn?

This book condenses years of experience building and using platforms into a mental framework for platform development. Covering technical, organizational, and financial aspects, it is structured along the journey that an organization might take toward becoming a platform company.

Part I: Understanding Platforms

When people say "platform", they may mean different things. That's why it's wise to first look at the history of platforms, catalog different types of platforms, and highlight their benefits.

Part II: A Strategy for Platforms

Building platforms requires a sizable investment and a clear strategy. That strategy must turn the objectives into an actionable path defined by meaningful decisions.

Part III: In-House Platform

Most IT organizations experience platforms when they set out to build one. This part looks beneath the covers of such platform initiatives to highlight important characteristics.

Part IV: Designing Platforms

Platforms hide complexity, but building one isn't nearly as simple as it looks from the outside. This part employs metaphors to illustrate platform design decisions.

Part V: Implementing Platforms

Diving deeper into the engine room, this part investigates platform anatomies and proposes common platform blueprints.

Part VI: Growing Platforms

Platforms have to be rolled out across the organization. They also require delicate care and feeding over time so that they don't fall victim to excessive entropy or become a bottleneck. This part shows you how to do this successfully.

Part VII: Organizing for Platforms

If you are building platforms, you'll likely need a platform team, which is different from typical application delivery or operation teams. This part describes how to build and manage a platform team.

What About the Architect Elevator?

This book is the second title in a growing series of "Architect Elevator Guides", based on the novel role of architects defined in *The Software Architect Elevator*. Targeting architects and technical decision makers, these guides don't get hung up on buzzwords and don't try to provide simplistic one-size-fits-all answers. Rather, they combine decision models that exert architectural rigor with real-life anecdotes from the daily grind of corporate IT.

In line with this approach, the books carry two "warning labels"; first:

This book doesn't tell you what to do. It teaches you how to come up with the best answer yourself.

You know your situation best. Many things in IT can be copied from Stack Overflow, but strategy isn't one of them. So, instead of giving you a fish, I want to teach you how to fish, that is, how to analyze your unique situation and selectively compose learnings from other organizations into a suitable strategy.

And, second:

You are bound to leave with more questions than you came with. But you'll also be better prepared to answer them.

Asking the right questions is a necessary first step for any successful strategy. I hope you enjoy the read and appreciate having more questions!

Do I Need to Read All Chapters in Sequence?

Just like the internet, my books are interlinked with cross-references between chapters and titles. Each chapter stands on its own, allowing you to skip ahead or start with a chapter that's particularly relevant to your situation.

The *Architect Elevator* guides, despite their titles, address a much broader audience than just architects. IT executives and decision makers may skip the technical details in Parts IV and V but will hugely benefit from the remaining parts. Folks leading a platform team may focus on Parts III, VI, and VII, whereas platform developers may skim Parts I through III and dive deep into Parts IV through VI. Architects will enjoy taking the elevator from the top levels down into the engine room across all parts.

Do I Need to Read the Other Books First?

The books in the *Architect Elevator* series are self-contained so that you can be consume them independently. That's why it's totally fine to start with this book. Cloud services are platforms, so you also might want to read *Cloud Strategy*, but you can do so before or after you read this one. The books in the series refer back to the "umbrella" book, *The Software Architect Elevator*, which introduces the way of thinking that we use to dissect complex domains like cloud computing or IT platforms.

There is only minimal duplication across the books, so owning the complete collection is a great asset for any modern IT architect.

What's With the Jellyfish?

Each book in the *Architect Elevator* series for IT leaders and architects features aquatic animals on its cover. Sea and river creatures are incredibly diverse and beautiful, so I expect to run out of content before I run out of images. All photos are my own.

I felt that jellyfish make a suitable metaphor for platforms. Just like IT systems, they carry some complexity but function without a central nervous system (or a very rudimentary one). Unlike some IT landscapes, they are amazingly well-coordinated systems and beautiful to look at.

Getting Involved

My brain doesn't stop generating new ideas just because the book is published, so have a look at my blog to see what's new:

https://architectelevator.com/blog

Also, you can follow me on Twitter or LinkedIn to see what I am up to or to comment on my posts:

http://twitter.com/ghohpe

http://www.linkedin.com/in/ghohpe

To provide feedback and help make this a better book, please join our private discussion group at https://groups.google.com/d/forum/cloud-strategy-book

Acknowledgments

Like my prior books, *Platform Strategy* benefited from the involvement of a wide range of people. I won't be able to give due credit to everyone who provided valuable input, but I would like to call out Luca Acquaviva, Omar Khawaja, Manuel Pais, Johannes Seitz, and Clemens Utschig-Utschig for their insights and support. Special thanks go to HandyMaus for keeping everything working along the way.

One entity I don't intend to credit is AI. This book was entirely handwritten; no content was generated through GenAI technologies. Alas, I was tempted to make a bold statement here that the book is AI-free, but I realized that this doesn't entirely hold. For example, modern spell checkers are AI trained, rendering the book AI assisted, perhaps, but nevertheless 100% human authored.

Part I: Understanding Platforms

There must be something about platforms. According to a report compiled in 2020, 7 of the 10 most valuable companies in 2018 based their success on a platform business model. Ten years earlier, in 2008, that number was zero. Many technology companies provide technology platforms, such as cloud platforms, that enable other businesses with modern IT infrastructure. Either type of organization might use internal developer platforms to increase their development speed, and a data platform to make business decisions. And virtually all of them operate on the internet, which is perhaps the ultimate innovation platform.

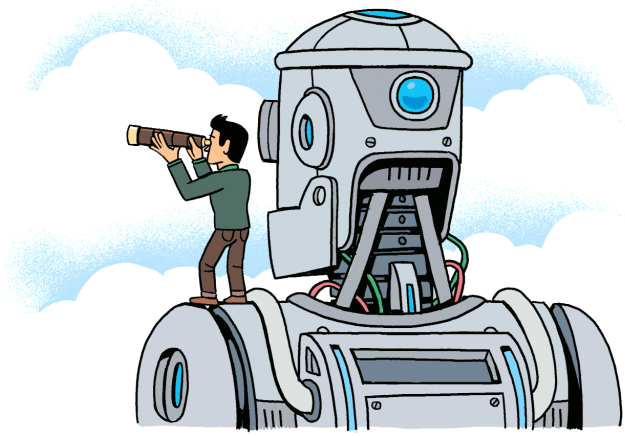
As we quickly discover, the term "platform" is overloaded and can mean different things in different contexts. It is therefore beneficial to look at what makes platforms different from prior approaches such as frameworks or common services, when something should actually be called a platform, and how platforms deliver benefits for your organization.

This part considers the platforms both in the business and technology space to set the context for diving deeper into designing successful platforms:

- Standing on the Shoulders of Giants examines the history and key properties of platforms.
- The Fab Four of Technology Platforms catalogs different types of platforms that you are likely to encounter.

1. Standing on the Shoulders of Giants

Do you really see further or is the air just thinner up there?



If I have seen further, it is because I used a telescope

Platforms are a common sight in daily life—they are normally a structure that you stand on; for example to catch a train or to get a better view. Generally defined as a raised level surface on which people or things can stand, technology platforms also appear to be raising things up a level or two. A platform model powers many successful companies, while cloud platforms have transformed modern IT operations by democratizing access to compute resources.

Platform effects are not new. In their like-titled book, Kim et al.[1] link the success of the Roman Empire to an open ecosystem strategy enabled by 80,000 kilometers of roads. They created a multisided market that spanned much of

[1] Kim, Song, Im: *Platform Strategy: a new paradigm for a changing world.* World Scientific Publishing; 2020.

Europe and reduced the friction for interaction between (not always voluntary) participants. The old saying that all roads lead to Rome seems to equally apply to large modern platform companies.

An eloquent—and widely cited—metaphor for using platforms are the words commonly attributed to Sir Issac Newton (although traced back to the 12th century):

> ***If I have seen further, it is by standing on the shoulders of giants.***

Many platform businesses have become giants, and architects have a keen desire to see further, so let's begin by taking a look at the concept of platforms and their various flavors.

Platforms Elevate

A raised platform elevates you. This applies to train platforms as much as to product or technology platforms: building on top of a technology platform means you don't need to start from scratch but can build on top of what others have created. Using a marketplace platform gives you instant access to a large set of merchants or customers. Publishing content to a social-media platform can elevate you to millions of participants consuming your content.

As powerful as platforms are, so elusive is their precise definition. Perhaps that's why Kim et al. resort to cataloging platform definitions spanning more than a decade. In his course on platform strategy for MIT Sloan's executive education, Zach Church aims for a definition of platform strategy:

> A platform strategy is an approach to entering a market which revolves around the task of allowing platform participants to benefit from the presence of others.[2]

[2] As you will discover throughout this book, entering a market is only part of a successful platform strategy, with many technology decisions and trade-offs to be made along the way.

Most definitions portray platforms as amplifiers: they create something bigger than just the sum of their parts. This characteristic helps explain the popularity of platforms, but also leads to another insight:

Platforms generate value through the interaction between their participants.

A platform without participants isn't providing value. It's like a farmers market that's lacking farmers and customers.

To offset your instant enthusiasm for such a powerful model, Church quickly reminds us that building a platform is "really, really hard". So, before you decide to jump on the platform bandwagon (there's a fundamental flaw in that mix of metaphors that I am intentionally overlooking), you should take a closer look at common types of platforms, their characteristics, challenges, and benefits.

Platforms: Faster, Better, Cheaper. Really?

The popularity of platforms in both business and IT domains is easy to grasp when you see the outstanding results that platforms deliver. Just like any major buzzword, platforms are also subject to hyperbole and re-labeling. My bank is now a financial platform, the local bookstore has become a content platform, and the supermarket has evolved into a marketplace platform. If you take the words from marketing departments at face value, you might even believe that without a platform your business is utterly doomed.

Let's put our feet firmly on the ground (or the platform, so to speak) and dissect the buzzword to understand the benefits platforms provide in different contexts:

Platforms enable

Platforms generate value by allowing participants to benefit from the presence of others. Those participants can be buyers and sellers in a marketplace who gain access to a wider range of goods or prospects, respectively, and can transact more easily and safely. They can also be users of common technology platforms like cloud computing, which encourages experimentation with a consumption-based charging model. Content creators and consumers participate on social-media platforms to share and interact globally.

Platforms democratize

Successful platforms make it easy for participants to join thanks to low barriers. For example, modern e-commerce platforms allow independent sellers to join with much less friction than trying to supply a major supermarket chain. Many influencers and content providers on social-media platforms started with a low initial investment, which would not have been possible in a traditional media model. Likewise, users can access powerful cloud platform services for pennies an hour.

Platforms self-perpetuate

Platforms that enable the exchange of (virtual or physical) goods between sellers and buyers appear to provide a form of *perpetuum mobile*: more buyers make it attractive for sellers to sign up for the platform, which in turn leads to a wider selection, which attracts more buyers. Walmart has known for a while that having one store carry everything is a promising business model. Modern platforms do so without the endless aisles and long checkout lines. Meanwhile, onboarding a new participant carries a near-zero cost: Airbnb doesn't need to build a hotel room to increase its inventory. Such *near-zero marginal cost business models* are a key tenet of fast-scaling digital business model success.

Platforms accelerate

IT platforms and marketplace platforms alike handle common but laborious tasks ("undifferentiated heavy lifting" in the parlance of a major cloud provider). By making this toil go away, platforms allow their users to focus on innovation and differentiation, whether it's an application that they are developing or social media content that they are uploading.

Platforms don't constrain

Many approaches that promise acceleration rely on a "my way or the highway" model: users benefit from lower friction if they abide by the rules of the framework. Platforms accelerate without constraining: Airbnb and other marketplace platforms offer more types of properties than traditional hotel chains, and cloud platforms allow users to build virtually anything on top.

Having been sold snake oil for decades, enterprise decision makers might look at these effects with a healthy dose of skepticism. Should we file platforms along with "model-driven architecture", "executable design models", and "seamless portability" in the large drawer of past technology ideas that not only sounded too good to be true, but actually were? Are platforms, in the words of a recent

Twitter user, just reinventing the wheel? Not so fast! Platforms are for real and are here to stay.

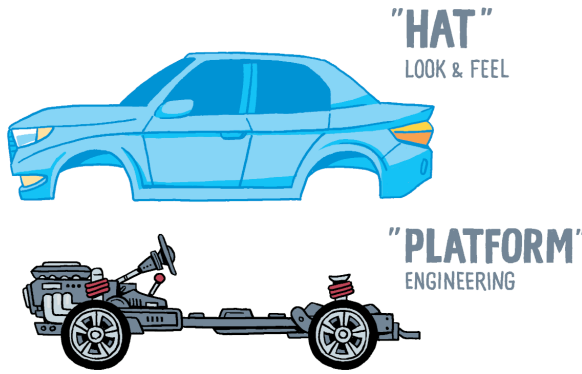
Established Platform Models

The success of platforms comes at a price: the label is used for numerous, quite different constructs. Although they share similarities, the mechanisms behind the different types of platforms are quite diverse. On the upside, platforms aren't an entirely new concept—it's just a common IT misbelief that we invented everything. Extraordinarily successful platform examples from automotive manufacturing, e-commerce, media, and IT will put the power of platforms in context.

Automotive Platforms

I am well known for car analogies, so I take great pleasure in reporting that automotive manufacturers have been using a platform concept for many decades (see Wikipedia). Those companies realized that a huge amount of engineering effort went into a vehicle's technical and safety components, such as engine, transmission, suspension, and anti-lock brakes. However, these engineering marvels are rarely visible to a prospective customer visiting the dealership. Even though the customer will consider safety ratings, power, and fuel consumption, which do result from the underlying components, purchasing decisions are often made based on the shape of the car, the interior finishing, the suppleness of the seats, or the badge on the trunk (boot).

It was a logical move, then, to reduce cost by conducting the base engineering effort once and reusing the chassis and its components across models. Following this strategy, car manufacturers could offer a variety of car models, targeted at different customer segments and often under different brands, without exploding engineering costs. Those cars exhibited a distinct look and feel despite sharing many components "under the hood".



Automotive platforms boost innovation both inside the platform and on top

These kinds of platforms are often referred to as "product platforms" or "inner platforms" to distinguish them from external platforms that involve customers or partners. The initial incarnations reused a complete chassis and many body parts, leaving relatively little room for customization. Some manufacturers described this approach as placing a "hat" (the car's body and interior) on top of the shared technical platform (the chassis).

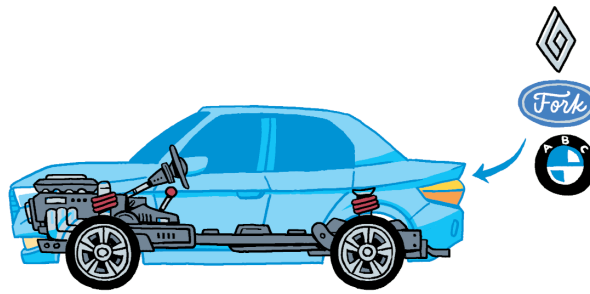
Volkswagen, which began using automotive product platforms in the 1970s, evolved the platform concept into several modular platforms, such as the Volkswagen Group MLB platform. Roughly translating into "modular longitudinal toolbox" (longitudinal referring to the engine orientation), it forms the basis for an amazing range of vehicles, from the Audi A4 all the way to the Bentley Bentayga SUV (using the so-called MLBevo matrix). Volkswagen's new MEB platform is looking to achieve the same for electric cars.

Against common belief, standardizing on shared platform elements didn't reduce choice or stifle automotive innovation. It achieved the exact opposite by boosting product diversity and technical innovation. BMW, for example, expanded its model range from a handful of series (3, 5, 7), to eight sedan series, eight SUV/crossover series, seven electric car series, several M series, plus a range of Mini vehicles. Such model diversity would not be economically viable without a platform strategy.

Automotive platforms amortize large engineering investments across a diverse range of models. Harmonizing those elements boosts diversity and innovation in others.

Common platforms provide better economies of scale for engineering innovations such as anti-lock brakes or autonomous driving. The standardization achieved via the platform (coupled with componentization) boosts innovation, one of magic properties of platforms.

With platforms, as with many other things, too much of a good thing can become an issue. The history of automotive platforms teaches a lesson about how much can be unified into the base platform and how much differentiation should remain on top of it. Based on the initial success, US car manufacturers took the platform idea a little too far in the 1980s by manufacturing essentially identical models that differed just in a few options or cosmetic elements. At its extreme, only the brand and model badges were different—a technique referred to as "badge engineering".



Relabeling is not platform engineering

Although tempting, this approach negated a key platform benefit: supporting diverse models and boosting innovation. Unsurprisingly, these attempts largely backfired, immortalized by the Cadillac Cimarron, which was positioned as a luxury car but was virtually indistinguishable from a fully equipped economy-class Chevrolet Cavalier. Its abysmal market response earned it a place on Forbes' list of Legendary Car Flops and its only contribution to the automotive world was lending its name to the "Cadillac Cimarron Effect".

Automotive history teaches us that both what's in the platforms and what's on top matter. Near replicas don't work.

A critical success factor for platforms is defining which aspects can be harmonized and which ones must be kept variable.

E-Commerce Platforms

When speaking about platforms these days, many people refer to digital ecosystems as embodied by companies like Amazon, eBay, Gumroad, Alibaba, and the like. These marketplace platforms connect buyers and sellers or people sharing a common interest.

A suitable definition that captures the intention of these business models can be found in Reillier's book, also titled *Platform Strategy*:[3]

> A business creating significant value through the acquisition, matching, and connection of two or more customer groups to enable them to transact.

Not all marketplaces are platforms. An analogy from the retail business helps find the delineation. A supermarket connects buyers with goods to be sold while holding its own inventory and being directly involved in the purchasing transaction. In contrast, a farmer's market connects buyers and sellers directly, letting them handle the transaction and maintain inventory. Whereas a farmer's market is a "multisided platform"[4], a supermarket is—strictly speaking—not. Farmer's markets democratize and enable transactions between farmers and consumers (the two sides), whereas supermarkets actively manage—and control—their supply chain.

Physical marketplaces and e-commerce platforms enjoy broad flexibility in their pricing models:

- They can charge buyers, for example by charging admission to a trade fair or farmer's market. Likewise, online platforms can charge membership fees to buyers, often pegged to premium services like fast or free delivery.
- They can charge sellers, for example by asking a fee for a farmer to have a stall. In the online world seller fees can be listing fees, transaction fees, membership fee, or any combination thereof.

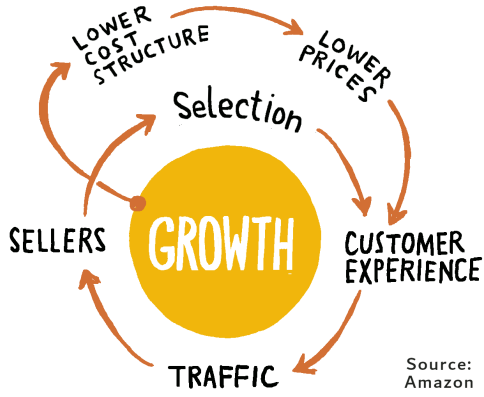
[3] Reillier B and L: *Platform Strategy: How to Unlock the Power of Communities and Networks to Grow Your Business.* Routledge; 2017.

[4] Hagiu A: *Multi-sided Platforms: From Microfoundations to Design and Expansion Strategies.* Harvard Business School; 2006, working paper 07-094.

- They can monetize through third parties, for example by inviting sponsors or selling advertising space, both in the physical world or the online counterparts. They can also monetize the data they collect by selling it to third parties.

This flexibility allows e-commerce platforms to offer services "for free" to select groups of the multisided market. Shifting pricing between participant groups or subsidizing specific types of transactions enables platforms to balance supply and demand, to drive growth, or, inversely, to moderate growth to uplift the quality of content or goods. Despite the risk of backlash, changes in pricing models are fairly frequent. Meetup provided a dramatic example by losing some 95% of their listings, but drastically improving quality, after starting to charge fee to organizers[5].

As the platform enables direct interaction between participants, it steps into the background—just like a real-life platform that's underneath the real action. Online platforms may be barely visible because they allow retailers to operate as white-label shops; for example, on Shopify. Likewise, when you visit the farmer's market you might notice the organizer's presence only by a banner at the entrance and a small organizer's booth. This is quite different from a supermarket where you are keenly aware of whose "ecosystem" you are in.



The Marketplace Flywheel

Major e-commerce platforms such as eBay, Amazon (which is careful to label itself as a marketplace), or Airbnb have harvested the perpetuating scale effects

[5]Parker, Van Alstyne, Choudary: *Platform Revolution: How Networked Markets Are Transforming the Economy.* W. W. Norton & Company; 2016.

cited at the beginning of the chapter. Amazon refers to this virtuous cycle as The Flywheel, inspired by the model presented by Jim Collins in *Good to Great*.[6]

This wheel includes two positive feedback loops. The first occurs over the number of buyers and sellers, which fuel each other's participation: more buyers attract more sellers, who carry a wider selection, which attracts yet more buyers. A secondary loop lowers the cost structure with increasing scale, which in turn allows lower prices, which then further fuel growth.

Such feedback loops are behind the tendency of e-commerce platforms' toward a "winner takes all" scenario in which a single platform dominates its respective market segment. That's the power of platforms and positive feedback loops.

Media Platforms

A close relative of e-commerce platforms are social and media/streaming platforms like Facebook, TikTok, Netflix, Twitch, and many others. Like their cousins, they operate a multisided market that connects providers and consumers of services, in this case, social data (photos, posts, events) or media such as audio and video streams. They enjoy a similar flywheel effect that attracts consumers to the platform with diverse content, which in turn attracts content providers because they'll get more eyeballs. They may lack the outer loop of lower prices (their cost of goods tends to be near zero) but that hasn't dampened their growth in the least bit.

The internet bubble of 2000 was driven by the obsession with "eyeballs", replacing traditional metrics like revenue or profit with the number of viewers. Two decades later, platforms have figured out how to scale and actually monetize those eyeballs.

The primary revenue streams for these platforms are advertising or subscription fees. Some media platforms play the role of both the platform operator and a platform participant; for example, producing their own video content ("Netflix originals") but also distributing third-party content. Platform companies enjoy a large amount of flexibility and tend to experiment all the time.

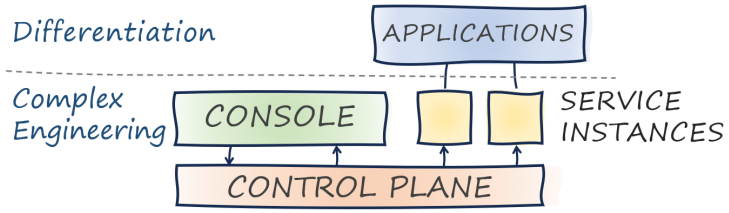
[6] Collins J: *Good to Great: Why Some Companies Make the Leap and Others Don't.* HarperBusiness; 2001.

Cloud Platforms

Cloud computing has perhaps been the most significant IT innovation of the past two decades, being challenged just recently by the rapid rise of AI and large language models (LLMs). The cloud also gave birth to a hugely successful technology business models, with just the top cloud service providers (CSPs) generating combined $200 billion in annual revenue. The collection of on-demand IT services provided by these companies are commonly labeled as platforms, with Google's cloud offering proudly carrying the label in its name: GCP—Google Cloud Platform.

The original motivation behind cloud platforms resembles that of automotive platforms. Just like cars, software requires a lot of heavy-duty engineering that isn't directly visible to the end user. Underneath a successful piece of software lies a vast array of compute infrastructure like data centers, global networks, servers and storage, software delivery pipelines, monitoring and failover mechanisms, synchronizing data stores, backup and disaster recovery, and regulatory compliance reporting. Those aspects can consume a large portion of a software project's timeline and budget before the first line of application code is ever delivered.

Combining those elements into common components that can be reused across many software projects allows developers to focus on providing customer value and differentiation; for example, through rapid feature delivery, unique functionality, or ease of use. They leave the heavy engineering work to the cloud platform providers, who have better economies of scale thanks to their broad customer base. IT departments consume those services in a low-friction way via a so-called cloud console (Web interfaces) or API calls that provision service instances to be used by custom applications.



Cloud platforms allow application teams to focus on differentiation

Although they follow similar approaches, cloud platforms also differ from

automotive platforms in several ways—after all, they are software. First, usage and application diversity are much broader. Whereas an automotive platform supports a handful of models for a specific automaker, cloud platforms support software development across a vast variety of use cases. It's the equivalent of using Volkswagen's chassis, engine, transmission, and suspension to put your custom interior and bodywork on top. Although this model exists as so-called "kit cars" (which generally modify a finished car to simplify registration), it's a small fringe business and large-scale automotive platforms lack such flexibility.[7] Bits and bytes are more malleable than steel.

Second, cloud platforms allow developers to use the common components easily without any welding or assembling. A platform's value isn't just defined by what's inside, but also by how easily its functions can be accessed. This is where cloud platforms shine. Even though they might look like traditional IT outsourcing from far away, the interaction between platform provider and platform user is notably different: month-long contract negotiations are replaced by an API call and near-instant provisioning.

Last, cloud platforms offer more fine-grained components than automotive platforms possibly can, offering several hundred individual services to customers. The only possible challenge remains an abundance of choice.

The success of cloud platforms is nothing short of phenomenal, creating a quarter-trillion-dollar market (according to Gartner) in just a decade-and-a-half. That's no surprise when you consider that cloud computing has fundamentally transformed IT—from months of infrastructure planning and provisioning to running an automation script and waiting a few minutes.

Business Platforms

Business platforms aim to elevate what cloud platforms achieved for IT infrastructure toward business applications. Starting off as powerful applications, these products added more capabilities for customization by splitting common functionality from specific needs, much like automotive manufacturers did. Prominent examples include SalesForce for Customer-Relationship Management (CRM) and SAP for Enterprise Resource Planning (ERP). The

[7] Volkswagen will, however, sell you a kit to convert your classic into an electric car: https://www.volkswagen.de/de/besitzer-und-service/magazin/elektromobilitaet/classic-cars-turn-into-electric-vehicles.html .

SalesForce application sits on top of the "Force.com" (now SalesForce Platform) development platform, whereas SAP features a Business Technology Platform (BTP), both allowing customers to build their own business applications and customizations.

The critical progression from a feature-rich business application to a platform is twofold. First, the vendors transitioned to a Software as a Service (SaaS) operational model that vastly lowers the friction of both user adoption and platform evolution. Second, configuration settings, which often constrained customers, gave way to custom applications, which utilize the application domain's data model but don't place any constraint on the code developed on top. The combination is indeed powerful.

Cloud computing platform vendors haven't been oblivious to the power of business platforms. Virtually all of them offer business capabilities as services, such as Amazon Connect for contact center management and the Microsoft Dynamics suite for accounting and related business applications.

The Platforms of Giants

There are valid reasons that platforms seem to be everywhere these days. Many platform providers are indeed giants, whether it's automotive giants, e-commerce giants, or internet giants. That doesn't mean, though, that there's already a platform for everything or that you have to be a giant to build one. Let's go on to catalog widely-used technology platforms, including those that are commonly built in-house.

2. The Fab Four of Technology Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Technology drives business models.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Technology Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Marketplaces

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Common Use Cases

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Interaction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Implementation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Considerations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Base Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Common Use Cases

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Interaction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Implementation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Considerations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Developer Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Common Use Cases

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Interaction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Implementation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Considerations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Business Capability Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Common Use Cases

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Interaction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Implementation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Considerations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Four in a Row

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Combinations Encouraged

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Primus Inter Pares?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Part II: A Strategy for Platforms

Platforms can play a pivotal role in IT transformation by reducing complexity, harmonizing software delivery, and stabilizing operations, all the while reducing cost. It's no surprise, then, that they have become a staple in today's IT and business strategies. But launching a multisided market business model, building an in-house developer platform, or even using a cloud base platform are major strategic decisions that require significant investments in both technology and organizational change.

That's why success with platforms rarely results from an IT department's effort alone. Bridging both IT and business strategy is a key aspect of a successful platform strategy. Any gaps in this area can become the source of expensive failures.

"Strategy" as a term sadly doesn't stand far behind "platform" when it comes to fuzziness and overuse. So, this chapter starts by recapping what it means to define a strategy before zooming into platform strategies. Strategy will never be a "paint by numbers" exercise (nor should it be), but organizations can benefit from common building blocks and recipes harvested from successful platform strategy definitions and executions.

The following key elements play a critical role in defining a platform strategy:

- Let's first describe what constitutes a sound business or IT strategy for large organizations.
- Becoming a platform company requires organizations to rethink their existing ways of working

- Platforms can overcome apparent opposites such as innovation and harmonization.
- Wardley Maps are excellent models to understand what drives platform adoption and how platforms drive innovation.
- Writing a strategy document can lead to writer's block, so it's good to have a mental framework.

3. Formulating a Strategy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Strategy is the difference between making a wish and making it come true.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Strategy Is Hard

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Strategy Is Hard to Argue Against

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Make Your Platform Wishes Come True

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

You Can't Copy-Paste Strategy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

IT and Business Strategy Form a Two-Way Street

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Think in the First Derivative

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Documenting a Strategy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Aim for Emphasis Over Completeness

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Use Conceptual Models

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Show the Path and the Terrain

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Credible Roadmaps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

From Strategy to Execution

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Strategy Is a Winding Road

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

4. Becoming a Platform Company

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Transformation can't be understood from the end product.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

More Than Meets the Eye

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Don't Burst the Boiler

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Classic IT Conflicts

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Opposites Attract

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Transforming by Seeing More Dimensions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Rethinking Old Ways

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

The Path to Platforms: On-Off-Line

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Business and Technology

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

5. The Platform Paradox

It's a kind of magic.

Come on guys, what's so difficult here?

Now that our view of platforms has become less fuzzy, let's dive deeper into the subtitle of this book: how platforms can break through age-old IT conflicts, specifically those between achieving standardization and boosting innovation. For that, we need to rethink the role that standards play and expect a few surprises.

The Magic of Platforms

Platforms play a unique role in overcoming perceived opposites. They are per definition harmonized—a platform's strength lies in its broad usage. A platform can't be customized for each user because the high onboarding friction would break one of the enabling factors for platforms (many in-house platforms fall

victim to this trap). At the same time, platforms clearly boost innovation: they reduce friction and give their users many degrees of freedom, as we already saw with automotive platforms.

Platforms break through the perceived dichotomy between harmonization and innovation.

For cloud platforms, I occasionally remind customers of this counterintuitive quality in somewhat blunt terms:

The cloud you are getting is the same one your competitors are getting.

Still, cloud platforms have been the biggest innovation drivers that IT has seen in the past decade-and-a-half. And, just as with automotive platforms, that's because they harmonize. The platform's constraints remove other constraints, like long lead times and heavy up-front investment in hardware.

Removing Constraints by Constraining

At first, the paradox of platforms—unifying to boost innovation and diversity—might appear like magic, especially to IT leaders who have collected many battle scars trying to find a balance between these opposing objectives. But the effect of removing constraints by placing others has been known for more than a century. A vivid example is the Baltimore Standard for fire hydrants:[1]

In 1904, when the city of Baltimore burned to the ground, firemen from surrounding towns came to assist but weren't able to do much: their hoses wouldn't fit on Baltimore's fire hydrants. Harmonizing those connections, in the Baltimore Standard of 1905, enabled new use cases such as fire crews assisting another town.

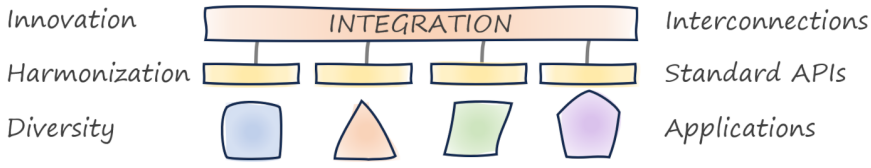
[1] For more insight on fire hydrants and standards, see the chapter "Governance Through Inception" in *The Software Architect Elevator*.

Information technology has seen similar effects—without burning anything down. HTTP is a standard: it is quite precise and universally adopted. By being a standardized protocol, it supported diversity between web browsers and web servers: any browser could visit any website, regardless of the respective technologies. It thus gave rise to the perhaps most innovative technology construct we have seen in the past decades, the internet.

HTTP is a standard that dramatically boosted innovation through harmonization.

Agreeing on *interface standards* is harmonization that becomes an innovation booster. Imagine a lack of harmonization, where one browser could work only with websites running on a server procured from the same vendor (the early browser wars almost drifted us into that direction). The internet surely would have never evolved the way it did.

HTTP is just one example of the power of interface standards. Common APIs achieve the same. By constraining—for example, by demanding common data formats and authentication mechanisms—diverse components can now interact and are no longer constrained in the choice of programming language or underlying run time.



Common interfaces harmonize and boost innovation

Common interfaces also accelerate the creation of new solutions and boost innovation by making more functionality instantly available for reuse.

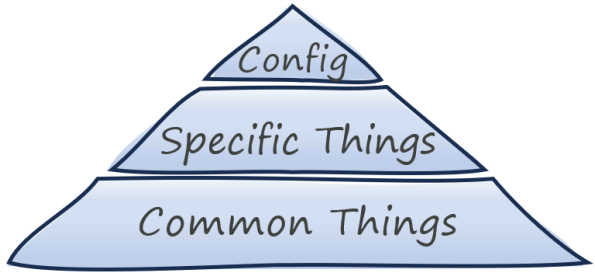
The IT Pyramid Fallacy

From far away, in-house platforms can resemble classic IT frameworks: all common elements that apply across business units/geographies/product lines are implemented in a base layer, on top of which resides a collection of more

specific, but still reusable elements. The cherry on top is that each business unit or geography merely has to configure a few settings, and, voilà, your business application is ready to serve customers without an additional line of code written. Although large software applications like CRM or ERP systems do unify many common processes, this approach looks much better in PowerPoint than in reality. The approach is flawed for two reasons:

1) You'd have to anticipate all users' needs. This is not only near-impossible, it'd also kill innovation.
2) Even if you could guess correctly, building the all-encompassing base layer requires a massive effort.

Such flawed visions routinely feature in IT strategy documents in the shape of a pyramid, notwithstanding the fact that people stopped building pyramids some 5,000 years ago, partly due to impracticality and partly due to horrible economics:[2]



We stopped building these things 5,000 years ago—except in IT

Platforms Aren't Pyramids

So, how are platforms different? Platforms enable users to construct the needed functionality without having anticipated each and every possibility:

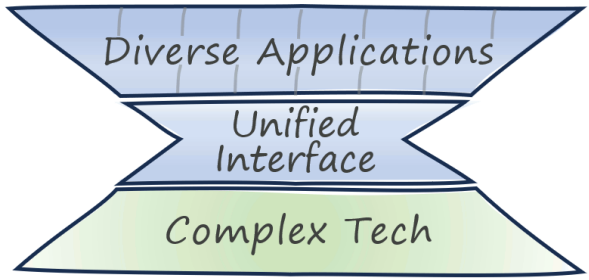
Platforms don't try to anticipate every use case.

[2] *The Software Architect Elevator* digs deeper into pyramids: "They don't build'em like that anymore."

This characteristic allows platforms to be leaner and more flexible at the same time. Users build on top of a platform; they don't just set configuration bits. Because it's all too easy for platform builders to fall into the trap of wanting to anticipate users' needs, I offer this simple test:

If your users haven't built something that surprised you, you probably didn't build a platform.

This observation might be the reason that Kim et al. cite *serendipity* as an essential element of a successful platform strategy: platforms leave room for users to innovate. So, the tip of a platform isn't narrow like a pyramid but supports broad innovation, yielding a double pyramid whose shape resembles an hourglass:



IT platforms hide complexity beneath and enable diversity above

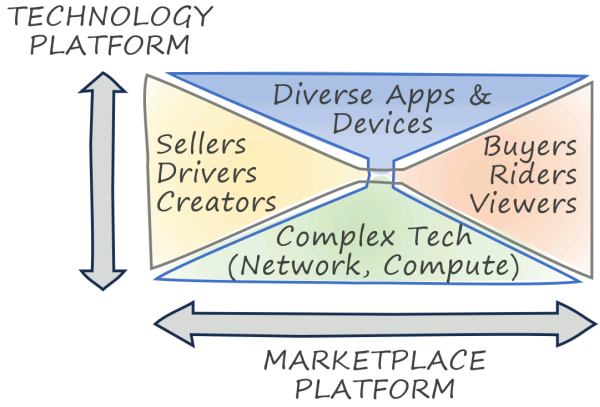
The narrow part of the hourglass (the neck or waist, depending on your preferred anatomical analogy) depicts the harmonization that platforms rely on: a wide diversity of base technologies or choices is simplified behind a much narrower interface. That interface in turn supports broad innovation on top.

Technology platforms hide complexity, such as networking, security, data center facilities, hardware provisioning, workload placement, failovers, and much more behind a simple interface, which in turn enables a variety of use cases. Automotive platforms demonstrate this effect by boosting innovation inside the chassis layer (thanks to better economies of scale) and in the top layer (thanks to better economies of speed). The failed concept of badge engineering occurred exactly when things slipped back into a pyramid-shape.

Folks with experience in enterprise integration, using tools like Enterprise Service Buses, are familiar with the hourglass picture: common data models and transports simplify an *n-squared* model, where each pair of applications has to talk individually, to an *O(n)* model that scales much more easily across many applications.

The Double Double Pyramid

Earlier chapters described both multi-sided marketplace platforms and technology platforms. Although they share common characteristics like low onboarding friction and democratization, they address different user groups and play at different levels of the organization. A multi-sided market is a business model that connects buyers and sellers (for e-commerce platforms), content creators and viewers (for social-media platforms), or drivers and riders (for ride-sharing platforms). Meanwhile, technology platforms harmonize and simplify a technology stack to boost developer productivity and application diversity. A technology platform can be an in-house developer platform, which shields development teams from the complexity of cloud run-time platforms. It can also be a mobile phone platform like Android or Apple's iPhone ecosystem, which (mostly) shields application developers from the diversity of mobile devices and network communications.



Many digital companies use technology platforms to offer marketplace platforms

Many successful digital businesses harvest the benefits of marketplace platforms and technology platforms.

For example, a ride-sharing company is a classic marketplace platform that connects riders with drivers. It's built on a mobile technology platform that shields the application from different mobile providers and networks around the globe. Internally, it uses a developer platform that boosts productivity by reducing the cloud base platform's cognitive load. Each of these platforms exhibits the hourglass shape that expresses diversity (the bulbs) enabled through harmonization (the neck).

How Platforms Break Barriers

No single aspect gives platforms these amazing properties; rather, it's the interplay of multiple factors:

Componentization
: Breaking something complex into standardized and recombinable components speeds up innovation through recomposition. To choose a historical example, standard-sized bricks have massively sped up construction without reducing creative possibilities.

Separating commodity from differentiators
: Platforms bake widely used functions into a common layer and make them easily consumable and composable into new solutions. Borrowed from the pyramid approach, finding this dividing line, especially as it keeps shifting, is no easy task. Successful platforms get this right.

Building Economies of Speed on Economies of Scale
: Building platforms is a scale business—they thrive on growth and require massive investments. However, platforms hide these scale effects from their customers by allowing frictionless incremental usage. They thus democratize access to resources, which boosts innovation.[3]

Centralizing decentralization
: Platforms are central elements that foster autonomy and independent decision making. They support decentralized organizations while providing a common safety net and necessary guardrails.

Let's look at how each of these effects contributes to platform magic:

[3] Platforms are great antidotes to black markets, which are well known to stifle innovation.

Componentization

Simon Wardley highlights the linkage between commoditization (something becoming commonly available and undifferentiated) and componentization (the ability to assemble something new from pre-made parts). Platforms achieve their potential by not being monoliths, instead exposing many recombinable elements that can then be commoditized.

Cloud platforms feature hundreds of individual services that can be combined to support a limitless variety of use cases.

Componentization is more than splitting something into parts. Chopping wood isn't componentization. Instead, the pieces need to have a clear relationship to one another, an aspect that we'll dive into a bit later.

Componentization generally requires an overarching architecture that defines boundaries and connecting elements. The automotive industry was successful with its platform strategy because cars have a well-understood architecture of individual components. Increasing componentization was the primary mechanism that allowed companies like Volkswagen to progress from the basic hat-platform model to a modular "toolbox" approach.

Separating Commodity From Differentiators

Product platforms, much like frameworks, look to draw the line between commonly needed components and those that differentiate: whatever can be widely used goes inside the base layer and the rest is built bespoke on top of it. Even though that sounds intuitive enough in principle, in reality, it's a delicate balancing act. Several factors make drawing that line difficult:

- **Needs vary by user group**: Some users might welcome items included in the platform, whereas others consider them restrictive. A "soft" platform edge can allow user contributions, but that doesn't make the problem go away entirely.
- **The boundary shifts**: IT evolves and so do the needs of platform users. Cloud platforms started out provisioning just virtual machines and storage but today provide higher-level services from serverless compute to data analytics and machine learning.

- **Cohesion over precision**: Identifying the exact dividing line between commodity and differentiator for each element doesn't necessarily yield a good platform. Users expect a uniform level of abstraction across platform services.
- **Interaction matters**: How users access the platform is as important as what's inside it. Valuable functionality that's difficult to access won't make a great platform. Vice versa, oversimplification will restrict usage, falling back into the pyramid model.

Drawing the line between what goes inside the platform and what is left to the application requires constant fine-tuning based on feedback cycles. Perhaps that's one reason that successful cloud platform providers are relentlessly customer centric.

Economies of Speed Built on Economies of Scale

Platforms thrive on scale. The most successful technology platforms, cloud services platforms, depend on enormous up-front investments, such as large-scale data centers and global fiber networks. The magnitude of these investments limits the market to just a handful of so-called hyperscalers.[4]

Perhaps the biggest innovation of cloud platforms has been to free users from these scale effects: they have instant access to resources and only pay for what they use. Those are the perfect properties to operate in *economies of speed* because they encourage experimentation and innovation.[5] By hiding the economies of scale, the cloud has democratized IT: whereas in the past, only large enterprises had access to fancy data centers and mainframes, today any start-up business, no matter how small, can use the same powerful technologies. The real magic trick of cloud platforms is as follows:

Cloud platforms provide scale-optimized technology as a speed-oriented product.

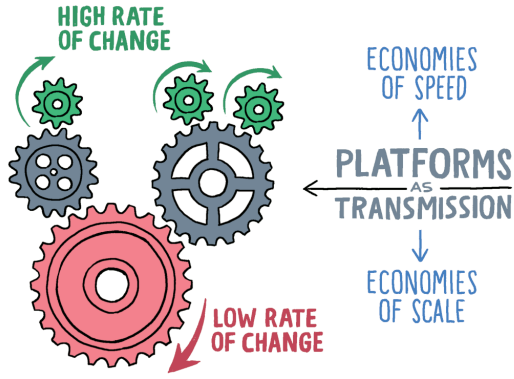
IT is frequently depicted as a "stack" of layers, ranging from an application to the middleware (like application servers or databases), operating systems,

[4]For a slightly dated but unusually transparent report on cloud economics, see https://mvdirona.com/jrh/TalksAndPapers/JamesHamilton_Mix2010.pdf

[5]Ironically these benefits can be undone by the long purchase cycles common with enterprise customers, which lead to multiyear contracts specifying minimum spending.

processor architectures, networks, and ultimately down to power supplies and physical server racks. Because items on top depend on the items lower down, lower layers generally have slower rates of change.

The prevalent processor architectures in use today (Intel x86 and ARM) date from 1978 and 1985, respectively, making it all but certain that they'll reach a popular lifetime of more than half a century. The Windows operating system is leading a happy adult life at 35 years and Linux is turning 30 this year. And it turns out that the 19-inch rack, the standard size for mounting computer hardware in data centers, dates back to 1922 (look it up on Wikipedia). Higher up, things move faster: developers now code mobile apps in Kotlin, whose 1.0 release dates back a mere five years, and deploy those apps on the latest Kubernetes release from a few months ago.



Enabling higher rates of change

Platforms are critical layers in this stack: they evolve comparatively slowly but enable high rates of change "above". They are like a transmission for the rate of change. Operating systems are a great example, and that's why they deserve the "platform" label.

Centralizing Decentralization

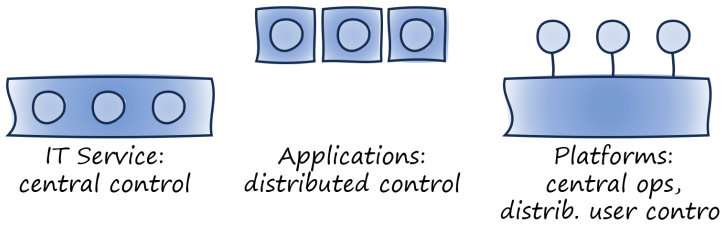
ThoughtWorks' Peter Gillard-Moss defines platforms as follows:

> Platforms are a means of centralizing expertise while decentralizing innovation to the customer or user. This happens through commoditization of services, but in a way which relinquishes an amount of control and is open to extension. This is a fundamental change to many organizations' core business models, which favor standards unification and work centralization.

Peter's wording aptly describes the platform paradox: we relinquish some control but do so in a centralized manner. This important nuance also explains why many so-called platform initiatives started by traditional infrastructure teams fail: they aren't looking to relinquish control but aim to strengthen it:

The most difficult step for organizations embarking on a platform journey is relinquishing control.

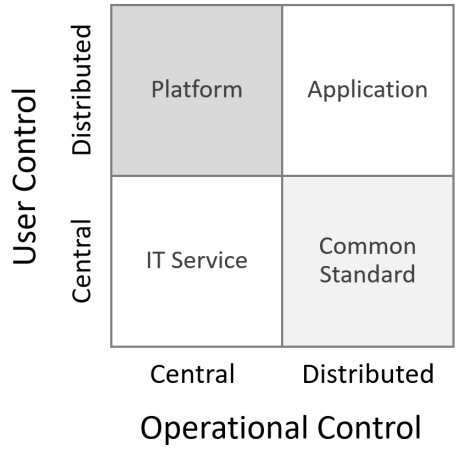
Traditional IT organizations link operational control with end-user control. A database administrator who assures the operational aspects of a database also exerts control over its use: developers need to file a ticket to change the schema. Although it's understandable that a person accountable for service uptime wants to restrict changes, the result is excessive friction. The typical alternative has each team managing its own database, ignoring central expertise and governance. Teams were forced to select either economies of speed or economies of scale.



Detaching user control from operational control

Platforms provide the best of both worlds: they provide central control and governance while decentralizing usage and innovation to the platform users. Again, we converted an issue into two independent dimensions.

Platform characteristics such as high levels of automation and a shared responsibility model make this apparent oxymoron work.



User control versus operational control

A common concern with relinquishing control is a perceived risk to governance. Experience shows, though, that carefully relinquishing control increases compliance. If IT infrastructure is rigid and cumbersome, teams are prone to going rogue and doing their own thing to make progress. An open platform gives users the ability to use parts of the common layer and perhaps even contribute to it.

A4 Paper Doesn't Stifle Creativity

Allow me to conclude a metaphor from *The Software Architect Elevator*. A4 paper is one of the most widely used standards in the world, but hardly anyone could claim that it stifles their creativity.

Good platforms should be like A4 paper: highly standardized with plenty of room for innovation and creativity.

6. Mapping Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

If you don't know where you are, a map won't help.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Maps as Visual Models

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Which Map Is Best?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Maps and Movement

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

2x2 Maps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Wardley Mapping

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Evolution

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

7. Addendum: “I ACED My Strategy”

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

A simple framework for writing IT strategy documents.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Alignment

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Clarity

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Evolution

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Decisions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

8. Talking with Platform Builders: SIMBAS

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

A Banking Platform for the Unbanked

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Pieter, what made you want to build a digital banking platform?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

What did you find to be different when you moved from an internal solution to building a platform to be used by many banks?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platforms harvest economies of scale to support economies of speed. How does that apply to SIMBAS?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

So, as we like to say, the platform “democratizes” access.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Does the platform help build a stronger financial ecosystem?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Along the cube model, in which direction are you looking to grow the platform?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

So, we learn again that scaling down is harder than scaling up?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

How do banks differentiate on top of your common platform that's built from standard components?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

A good platform is like a fruit salad, but may make it more difficult to swap out components. How does SIMBAS strike that balance?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Can a fruit ever go bad?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platforms drive innovation both on top and inside the platform. How does SIMBAS benefit providers?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Is building a banking platform different from other platforms?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Part III: In-House Platforms

Platforms not only play a major role as part of a business strategy, but also as enabler for IT delivery. IT organizations rolling out an internal developer platform typically experience the benefits but also the challenges of building an IT platform. Despite being built on top of commercial platforms or open-source components, platforms differ substantially from typical IT projects, requiring organizations to rethink their approach from traditional IT services to platforms.

At the same time, expectations for internal platforms are high: they are supposed to boost productivity, increase compliance, and reduce vendor lock-in. IT organizations must therefore be able to set appropriate expectations and understand decision trade-offs.

This part provides IT organizations with guidance on how to build and deploy an internal developer platform:

- In-house IT platforms come in several shapes and flavors.
- Existing service teams will find platforms to be entirely different animals altogether.
- Despite platforms having some magical properties, delivering benefits to the organization shouldn't be a magic trick.
- IT organizations may be puzzled why developers love opinionated platforms but despise restrictive ones.
- As with any architecture, platform teams must make conscious decisions and understand the trade-offs.
- Procuring a platform is easier than building one yourself, but that doesn't mean it's the easy way out.

9. In-House IT Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

My way is the highway.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

IT: Steering from the Top

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

IT Platform Benefits

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

IT Platform Classifications

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

IT Platforms Varieties

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Digital Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Engineering Productivity Platforms/Internal Developer Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Data Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Data Meshes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

API Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Abstraction Layers/Cross-Platform Platforms

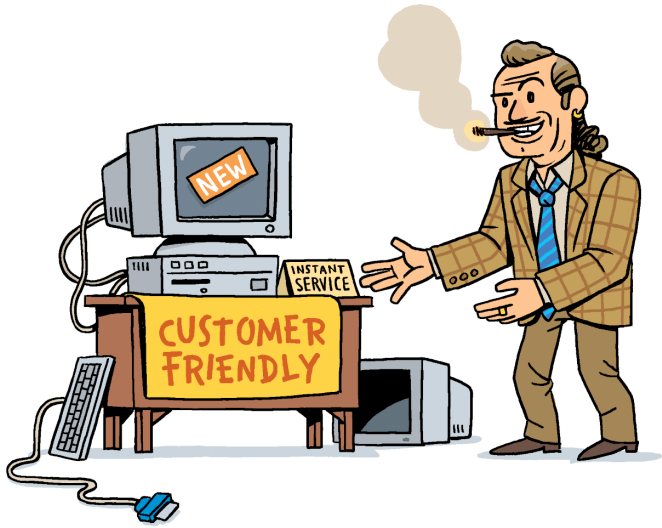
This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platforms and Software as a Service

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

10. IT Platform and IT Services Are Antonyms

After renaming all teams, still nothing improved…

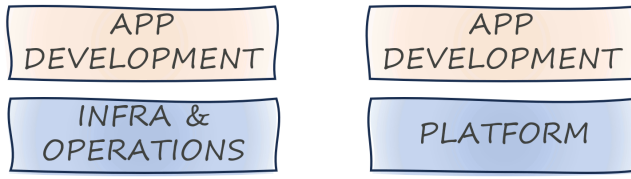


Teams, you won't believe the deal we got on your new development platform!

The examples of common IT platforms in the previous chapter make the concept of in-house platforms more tangible. However, enterprises might also consider their existing data center or IT services a platform, or at least face the temptation to apply that label. After all, they provision virtual machines on demand, which sounds a bit like a cloud, which in turn is a platform. As so often, things that might appear similar from far away reveal important differences upon closer inspection. Let's take a closer look at what makes a platform deserve the name and why traditional IT services are unlikely to pass that test.

Isn't It Just a Box on Top of Another Box?

When we draw high-level diagrams of platforms (including the diagrams in the Technology Platform Overview), they tend to look like a big box of "common things" with another box (or multiple boxes) of diverse things on top of it. Such a picture looks oddly familiar to anyone who has spent time in large-scale IT:



A structural model cannot show the differences

Much of IT is structured into a common infrastructure and operations layer, on top of which diverse applications are deployed. So, where's the difference between that model and all the excitement about platforms?

A Static Model Can't Show Dynamic Differences

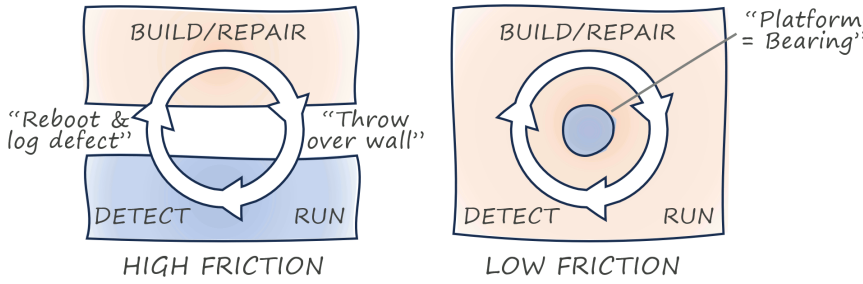
There are huge differences, but this structural model cannot show them. That's because this model is static—it shows only the pieces but not the interaction between them. Platforms are anything but static, so explaining the difference requires a new mental model.

The issues with the traditional "Dev and Ops" model are well known. Application developers ("dev") push for higher levels of flexibility and independence to release new functionality and meet customer expectations. They are also keen to try out new technologies to make sure that they use the best tooling possible, and that their resume remains up to date. Infrastructure and Operations, on the other hand, are more conservative and perhaps more resistant to change given that they are tasked with maintaining stable and secure operations.

Putting the two together results in a classic conflict of interest: developers throw half-tested software over the wall because the operations team has to wake up when the pagers go off. In return, operations can't do much to address outages besides rebooting a server and logging a defect for the development team to fix. This negative cycle is decorated with ample finger-pointing in both directions.

A Dynamic Model for a Dynamic World

A dynamic model makes the issue clear:



A dynamic model makes the differences clear

As the left side of the diagram shows, the loop across deploying software, detecting its operational characteristics (or usage metrics), and correcting them, spans two organizational units whose opposing incentives pit them against each other. Often referred to as the "outer loop" of software delivery (the "inner" comprising the code/test/debug), modern software approaches like continuous integration place higher demands on making the outer loop spin faster and with less friction. In a traditional organizational model, the outer loop crosses organizational boundaries and inhibits these new ways of working.

Placing operational responsibilities in the development team, thus avoiding the organizational boundary, is one way to reduce this friction. This "you build it, you run it" model does indeed align the incentives into a single team, but it also burdens that team with a broad set of tasks and responsibilities. This means that cognitive load for teams increases (every developer now must also be a cloud and ops specialist) or team members will specialize (some folks are ops-heavy and others dev-heavy), which essentially reverts the state of affairs to the starting point, just under a common team label.

Placing operational responsibilities within the development team is the correct setup, but those teams must have the matching tools to perform these tasks as efficiently as possible. That's where platforms come in. Developer/engineering productivity platforms can be depicted as the axle that helps the outer loop spin faster.

A platform team builds the axle that makes the outer loop spin faster. But it's not part of that loop to avoid becoming a bottleneck.

The platform team is part of another loop, though: the loop that takes input from development teams to evolve the platform. This loop spins much more slowly and does not directly affect the delivery teams' velocity.

Platform Characteristics

Understanding that developer platforms are a stark departure from the traditional operational model, it's easy to see how existing teams might believe they're building a platform when in reality they're not.[1]

Platform teams telling me that they provision or operate resources for the development teams is a warning sign that they might be stuck in the old model, just with a modern label.

That's why a set of key characteristics that qualify an in-house project as a genuine platform is needed. Consider these like a checklist that helps you determine whether something you're building is a platform or not:

Speed First, Efficiency Second

Large organizations traditionally view common elements (including platforms) as an opportunity to avoid duplication and achieve efficiency by doing things just once instead of multiple times. Such *Economies of Scale* helped traditional enterprises lower their unit costs, but software platforms are different. The traditional focus on efficiency through reuse leads to a dangerous side effect called out by Professor Jan Bosch his article "Platforms should focus on speed, not efficiency":

When companies focus on efficiency, the consequence tends to be that everything slows down.

Slowing things down is deadly in *Economies of Speed*, but that's what happens with reuse because it requires coordination. Platforms, in contrast, speed things up.

[1] A telling example can be found in Evan Botcher's article *What I Talk About When I Talk About Platforms*: https://martinfowler.com/articles/talk-about-platforms.html

Provides Value Indirectly

Platforms deliver value indirectly via other projects. Value is realized by the platform users; for example, by reducing projects' development effort. Platforms can also deliver value centrally, as well; for example, by providing better transparency across an IT organization's project portfolio, which in turn allows better decisions or more effective resource allocation. Just like multisided markets, IT platforms also serve multiple user groups:

- Project developers who can speed up software delivery
- Component developers who can find more users for their functional blocks
- Management who gains more transparency into workloads and resource utilization
- HR who can attract talent who are interested in working in a modern technology environment

Delivering value indirectly implies that a platform can deliver value only in combination with other projects. Running the risk of stating the obvious, this means that platforms without users generate no value. Platforms are indirect value enablers, not direct value creators.

Thrives on Scale

Something created as a one-off to support another project isn't a platform. Platforms are built to host a wide variety of other projects, reducing duplication of common components while enabling diversity in project implementation. Platforms thrive on scale—the more users are on the platform, the more attractive it becomes to be on the platform. In comparison, popular in-house IT systems (to the extent they exist) become victims of their own success, resulting in a bottleneck. They end up slowing the organization down, which is exactly the opposite of what a platform should do.

Minimizes Marginal Cost

Successful platforms grow because new customers can sign up with minimal effort for both the user and the platform, meaning the platform's marginal cost for an additional customer is near zero or low.

Successful IT platforms inherit this property from e-commerce platforms like Airbnb. Airbnb doesn't have to build any new rooms to sign up a new host, allowing it to scale almost infinitely. For in-house IT platforms, automation, self-service APIs, and building on an elastic infrastructure are common mechanisms to achieve the same effect. Platforms that ignore this aspect are bound to become victims of their own success.

Reduces Friction

Low friction extends beyond user sign up. Traditional IT processes require would-be users to submit complex ticket requests that trigger a time-consuming and often manual provisioning process. Such cumbersome interactions stem from local optimizations within the team that shift the burden to users.

High onboarding friction all but guarantees the quick demise of any in-house platform.

Low friction doesn't always equate to a low barrier. In-house platforms and base (cloud) platforms can have low technical friction but require users to adopt a different mental model. Users who are already familiar with the new way of working—for example, using declarative scripts for infrastructure provisioning—will find the friction to be low. Others will encounter a steep learning curve at first.

Embraces Self-Service

Self-service is the default mechanism through which IT platforms assure low friction. Instead of filing a ticket, teams directly provision a virtual machine. Transparency forms the other half of making a team self-sufficient: they need to have a good read on what's going on, so that they can pick the appropriate course of action through self-service. This way of working leads to a shared responsibility model (see below).

Run as a Product, Not a Project

A platform can't be built by gathering requirements, implementing them, and calling it a day. We have seen how platforms can evolve user behavior and

thus unlock additional opportunities, leading to a fruitful cycle of continuous improvement. A product must target a well-understood market, meet specific customer needs, and evolve alongside those needs. That's how platform teams must operate.

Evolves Continuously

Successful IT platforms evolve both in the depth of the services they offer and in the scope of services they provide. Users benefit from standing on a platform that continually grows and lifts them up.

When rolling out the *Agile Delivery Platform* inside a large enterprise, we decided to regularly update the underlying software product (an on-premises Platform as a Service [PaaS]) to the latest version, which ran counter to the IT operations team's preference of postponing updates until after all application owners agreed. A shared responsibility model was instrumental in allowing us to work this way.

Puts Customers ahead of Processes

So-called common services restrict users to a given set of software libraries, third-party products, or specific processes. While the motivation is easy to understand—organizations are looking to harmonize, reduce complexity, and boost compliance—these goals can't stand in the way of platforms enabling their users. Recall that "platforms enable" was the very first benefit described in Chapter 1.

Platforms therefore must find a way to serve their customers ahead of the platform stakeholders. After all, without customers the platform will provide no value whatsoever. Companies that are extremely successful with the platform model tend to be customer centric. Amazon even adopted "customer obsession" as one of its leadership principles and has managed to build not one but two successful platform businesses.

Is Centrally Built and Operated

Platform users don't need to concern themselves with the operations of the platform, because it is operated by a dedicated team. Platforms are available as an always-on production service that pools resources for standardized and automated management. This property is key to reducing the friction of teams operating on top of the platform.

Shares Responsibility

Platforms have a shared responsibility agreement with their client projects. Whereas the platform takes care of certain operational qualities like security or compliance, the client projects carry other aspects of the same qualities, such as availability, security, and compliance. You can't just deploy any old crappy application to a platform and expect miracles.

Self-service can be an important part of the shared responsibility, allowing client teams to perform operational tasks like monitoring resource usage or performing restarts. An apt analogy is cars: a manufacturer can equip the car with a seatbelt and an airbag, but you still need to be a responsible driver. AWS articulates their expectations clearly in their Shared Responsibility Model for security and compliance.

Users Extend

IT platforms should be extensible by platform users; for example, to share functionality they have developed. This approach stands in contrast to traditional IT-services organizations that are looking to maintain central control and give users very limited influence over the services that they provide. Platforms are open, whereas IT services are generally closed.

Honorable Mentions

A few additional characteristics are considered essential by some but debated by others, so they aren't part of the previous list but instead get an "honorable mention" here.

Voluntary Adoption

Voluntary platform usage is generally the preferred way because it builds more engagement and also provides better feedback to the platform teams. Mandated usage conflicts with customer centricity—only a tax authority might claim to be both, but that's outside the scope of this book. At the same time, a large organization isn't a democracy—whether you like it or not, hierarchies and decision powers exist for a reason.

As a pragmatist, I caution teams wanting to make their platform mandatory that it's nearly impossible to enforce things in large, federated organizations. There are one-thousand-and-one ways to work around "required" tools and even if things come to a head, you won't have sufficient political capital to fight and win every battle. This isn't a Hollywood movie where the heroes prevail. It's enterprise IT and the hallways are littered with skeletons of those who fought the noble battle.

What you can do, though, is make people's lives easier if they use the platform and harder if they don't. If that sounds slightly mischievous, just think of it as a core property of multisided markets. Successful platform businesses commonly subsidize early users or charge a third party to provide a valuable product at near-zero cost—until they reach scale and flip the model. If it works for the "giants", why shouldn't you get a slice of it?

Managed by a Dedicated Team

Platforms are typically managed by a dedicated team that acts like a small business of its own. This approach makes good sense, but I am reluctant to call it a defining characteristic. If your platform exhibits all the aforementioned characteristics but is managed in a community model, more power to you! You may also find the opposite, a platform team that doesn't actually build a platform.

Platform ≠ IT Service

As elaborated at the beginning, IT platforms often evolve (or are occasionally re-labeled) from more traditional IT Services, but have entirely different character-

istics. A side-by-side comparison summarizes the contrast between developer platforms and traditional IT Service Management (ITSM):

Characteristic	Platform	IT Service
Main Driver	Speed	Reuse
Value Proposition	Direct	Indirect
Scale Effect	Thrives	Bottleneck
Marginal cost	Low	Medium/High
Friction	Low	High
Interaction	Self Service	Ticket-based
Run as	Product	Project
Evolution	Continuous	Sporadic
Orientation	Customer Centric	Process Centric
Responsibility	Shared	Separated
Extensibility	Open or semi-open	Closed
Adoption	Voluntary	Mandated

This list can help debunk lipstick-on-a-pig maneuvers that re-label existing processes and operational systems as platforms. The one common property (and therefore not included in the list) is that both IT services and platforms are centrally operated. Platforms depart from the traditional model by separating operational control from user control.

The table also highlights how drastic the change from traditional IT services to a platform model is, which explains why so many IT teams struggle to build successful platforms.

The Platform Gestalt

A collection of characteristics is a great start but ignores that they support one another. As architects, we don't see characteristics as a simple checklist but also want to understand how they are connected—again the lines are as interesting

as the boxes (see "Drawing the Line" in *The Software Architect Elevator*[2]).

- A shared responsibility model enables continuous evolution, as described in the earlier anecdote.
- Self-service is a key contributor to low marginal cost, the ability to thrive with scale, and low friction.
- Making a platform user extensible is the ultimate form of being customer centric.

As a side note, a similar effect applies to good pattern languages: individual patterns are useful, but a cohesive language that shows how the patterns relate is much stronger. The best description of designing pattern languages I have seen is in the fifth volume of the Pattern-Oriented Software Architecture (POSA) series.[3]

Non-Technical Aspects

The platform model extends far beyond the technical aspects to also include organizational and operational aspects. That's why this book contains an entire part on Organizing for Platforms.

[2] Hohpe: *The Software Architect Elevator.* O'Reilly Media; 2020.

[3] Buschmann, Henney, Schmidt: *Pattern-Oriented Software Architecture Volume 5: On Patterns and Pattern Languages.* Wiley; 2007.

11. Mechanisms, Not Magic

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Making things work is not an implementation detail.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Making Things Simpler Isn't Simple

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Marchitecture

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Mechanisms Provide Linkage

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Welcome to the Twilight Zone of Architecture

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Cognitive Load

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Mechanisms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Restricted Choice

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Meaningful Defaults

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Assumptions/Scope

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Aggregation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Abstractions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Automation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Functional Addition

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Mapping Mechanisms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Non-Technical Mechanisms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

12. Do You Have an Opinion? A Mind of Your Own?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Why we love opinionated platforms but despise restrictive ones.[1]

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Being Opinionated

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Opinions Have a Shape

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Those Are My Opinions. If You Don't Like Them, I Have Others.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

[1] Some developers may be inclined to sing along when they encounter an in-house platform: "I thought you were special."

Open Source Can Afford To Be Opinionated

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Unnatural Selection

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Cohesion

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Freeways Are Opinionated

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Are the Streets Really Paved With Gold?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Employer Lock-In

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

13. Making Platform Decisions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

You want a quick decision? Give me a coin...

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

The Most Important Decisions Might Be the Ones You Didn't Know You Made

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Truth versus Comfort

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Decision Catalog

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Open or Closed?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Free or Charged?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Mandated or Voluntary?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Immortality?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Can the Platform Shrink?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Rate and Cadence of Change

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Preconditions and Assumptions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

A Platform Design Canvas

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

14. Procuring a Platform

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Why buy it when you can build it?[1]

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Do You Need a Platform?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Should You Build a Platform?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Selecting a Platform

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Utilizing a Platform

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

[1] An ill-founded approach used by too many engineering teams

Respecting Platform Opinions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Leaving a Platform

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

15. Talking with Platform Builders: Singapore GovTech

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

A Developer Platform for the Singapore Government

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

What role does GovTech play in the Singapore Government?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Your developer platform initiative started in 2016. Were you ahead of the trend?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

So, you built a product for yourself first rather than ponder over what developers might need?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

How did you decide to bootstrap your platform with a CI/CD product?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Back in 2018, the platform engineering buzz hadn't much taken off. Where did you find guidance?

Today we would call that "reducing cognitive load"...

It sounds like you divided your platform into slices!

How different is building a developer platform for the public sector from a commercial one?

NECTAR, APEX, SHIP, and HATS started off as separate products. Does that make your platform a fruit basket?

When rolling out your platform, were some things harder than expected?

Could you share a bit about your pricing models?

Did your users build something that surprised you?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Are you planning to build additional platforms or expand the existing ones (following the cube)?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Do you have any final piece of advice for platform teams?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Part IV: Designing Platforms

With the widespread success and potential of platforms, it's tempting to want to build one in your organization, perhaps to streamline application delivery or as the foundation of your organization's business. That's generally a sound idea, but, as so often, there's a lot more behind other companies' platform success than might appear from the outside.

Designing and building a successful platform requires more consideration and engineering effort than many teams anticipate. Some organizations spent considerable time and effort to build elaborate platforms that, upon launch, didn't find much adoption. This part highlights critical design decisions that guide the shape of an in-house platform. As platform design is a complex topic, the chapters in this part suggest real-life metaphors to highlight the decision trade-offs and nuances. Those mental models can also be helpful when assessing third-party platforms.

- Architecture is often defined by quality attributes, and it's no different for platforms.
- A good platform is more than the sum of its parts, so choose carefully whether you are designing a Fruit Salad or Fruit Basket.
- Platforms are generally seen as horizontal elements, but they often sit on vertical pillars.
- In-house platforms that build on top of external base platforms need to decide whether to float or sink.
- Platforms can make great abstraction layers, but it's all too easy to fall victim to the Grim Wrapper.
- Platforms reduce cognitive load by hiding complexity. But hiding too much can lead to dangerous illusions.
- Even the most awesome abstraction crumbles when failure rears its head and reminds you that it doesn't respect abstraction.

16. The 7 “C”s of Platform Quality

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platforms might not be forever, but they do have more Cs than diamonds.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Architecture Decisions and Trade-Offs

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Delineating “C”s

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Car Metaphors for the Win

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

One C to Rule Them All

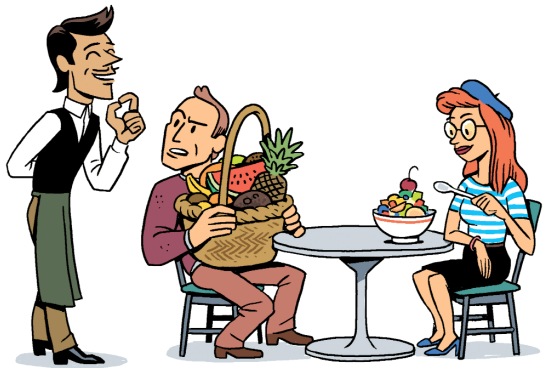
This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Architecture Strategy à la “C”

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

17. Fruit Salad or Fruit Basket?

Good platforms are more than a collection of services.



Sire, this is the market-fresh fruit, as you requested

Most platforms comprise a collection of individual pieces such as components or services. However, the value of the platform doesn't just derive from its scope—that is, how many components it contains—but also from how these pieces come together to form a meaningful whole. Given my knack for applying real-life metaphors to IT architecture, we find design guidance in a rather unexpected location: the produce section of a supermarket.

Summing Up the Parts

Platforms combine elements into a meaningful whole. Those elements can be purpose-built components or, as is likely for in-house platforms, existing elements from a cloud or data platform. For example, a modern application delivery platform might include a CI/CD build and deployment pipeline, a serverless container runtime, observability services, and a service mesh for communication.

It's tempting to describe such a platform by the list of services or projects that it comprises, in this example perhaps Bamboo, Jenkins, Nexus, Spinnaker, Kubernetes, Prometheus, and Istio. But equating a solution to the list of its components would be a vast oversimplification and miss the core purpose of architecture. It's not only important that your platform includes Jenkins and Prometheus, but also how the platform presents the combination of these packages. After all, the platform should be more than just an installation script.

A non-car analogy highlights the importance of how systems are put together:

Architects are like chefs. Good ingredients help, but a great meal comes from how they're put together.

Simply listing ingredients, whether it's for dinner or for building a platform, isn't a useful expression of the final system. You could prepare many dishes from flour, cheese, and tomatoes, ranging from pizza to lasagna to a grilled cheese sandwich. You'd want the restaurant to state which one you're ordering.

The same holds true for IT platforms. If you tell folks that your platform contains a CI/CD pipeline, they will have a pretty decent idea about what this component does. But they won't know how they will interact with the component or whether the platform centrally tracks all builds or collects key metrics like successful deploys. To make matters worse, platforms built from open-source projects suffer from the "creative" name choices that tend to confuse even the most well-meaning audience.

Thinking about a platform as a bill of materials also ignores the primary source of value for building it in the first place: if the platform is simply a collection of readily available tools, why wouldn't application teams just use their own Bamboo and Jenkins instances instead of adopting your platform? Hosted services like GitLab CI/CD make this approach even more appealing because no installation is required. Teams might even feel that using individual services gives them more freedom than your perhaps overly prescriptive platform.

Your platform therefore must equate to more than an installation script or a catalog of already existing services. Another food metaphor helps you express this design aspect vividly.

Fruit Baskets

Fruit baskets are really just baskets with fruit

Platforms that are compilations of individual pieces can be compared to fruit baskets: they collate several items into a convenient collection. However, in the end, all the customer gets is fruit. A fruit basket makes for a nice decoration, but it's largely a convenience. People aren't much worse off if they just buy the fruit itself, and they may actually be better off because the basket might lack their favorite fruit or, vice versa, includes fruit that they dislike. That pattern is well known from discounted bundle sets, which invariably include items that you wouldn't buy individually (luckily the lid of the yet-to-be-used 6-liter stock pot from my cooking set fits one of my saucepans).

Because purchasing bulk fruit at the supermarket is a vital alternative, fruit baskets can't command much of a margin over the ingredients, aside from affording a nicer presentation for special occasions. Japanese fruit baskets have taken that particular aspect to whole new levels of sophistication (and pricing), but it will prove difficult to replicate with IT services.

Serving Fruit Salad

Platform designers should therefore aim beyond fruit baskets. There's a much more platform-esque way of consuming fruit: the fruit salad.

A fruit salad opens up new opportunities

Just like fruit baskets, fruit salads are collections of fruit; however, the fruit is cut up and packaged into a ready-to-eat meal. What might appear like a minor convenience allows the fruit salad to deliver much more value than just the sum of its parts by:

Supporting use cases that fruit baskets or fruit cannot

They're easy to carry and perfect for a picnic, for eating on the go, or for having lunch at your desk. Fruit baskets, by contrast, are great for decoration but not for eating on the go.

Offering an enhanced user experience

They balance texture and sweetness independent of the size of the fruits. In contrast, if you buy one pear and one watermelon, you'll struggle to achieve much of a balance in your basket. You could say, fruit salads are more cohesive—they create a balanced whole.

Scaling down

They're available in small portions, regardless of the fruit size, in unlike a fruit basket containing watermelons. Their fine-grained, usage-based pricing model matches the customer's needs. Because scaling down is more difficult than scaling up, the compact format is a distinct advantage.

Reducing toil

Customers don't need to peel or cut the fruit, and the ingredients don't crush in a bag or your luggage. Not needing to cut up the fruit can be very handy, for example, when boarding an airplane where knives aren't allowed.

So, although a fruit salad might appear as not much more than cut-up fruit, it's actually a new product with a different value proposition. That's why it also fetches a significantly higher margin: many fruit salads clock in at more than 10 Euros per kilogram, whereas apples and oranges, the main ingredients, go for 2.49 (pre-inflation).

IT Fruit Salads

Platform builders should be inspired by fruit salads and not just define their platform by the tools and products it comprises.

When someone tells you that they built a GitLab/Spinnaker or Bitbucket/Bamboo platform, that's a fruit basket. All the customer gets are fruits, and increasingly those fruits can also be had directly from the vendor.

To make your platform more than just the sum of the pieces, the pieces need to be well integrated or automated, so that it becomes noticeably easier for platform customers to use the whole platform instead of the individual pieces. Sane default settings can get people going more quickly, as can better integration between the tools; for example, through shared account settings or by defining consistent version numbering across tools. Such integrations may seem trivial but can be surprisingly powerful in large systems. For example, relating changes in metrics back to a software release and the associated source changes can save endless hours of debugging. As with any product, the measure isn't how difficult it was to build the feature, but how valuable it is for users.

Every good thing can also be overdone. Platform teams may be tempted to overlay a complete abstraction over the individual tools, perhaps in the pursuit of a vendor-independent, universally pluggable software delivery engine. Sadly, such attempts tend to face the same fate as the Grim Wrapper. Let the fruit salad be a salad of fruits—don't try to make it a "nutrition-optimized breakfast supply with interchangeable Rutaceae".

Having Opinions

Naturally, a fruit salad–style platform will be more opinionated, meaning it doesn't intend to be all things to all people. Customers will be more likely to accept the constraints that the salad imposes since they benefit from a smooth integration and ease of use. In the worst case, customers can still pick out individual pieces of fruit and thereby get value by consuming only part of the "platform".

In comparison, opinionated fruit baskets are a tougher sell, as the benefit won't outweigh the perceived lack of choice. A beautiful wrapping might get customers' attention, but that initial attraction will wear off rather quickly.

Making Fruit Salad From Fruit Baskets

When I shared the fruit basket metaphor with a friend of mine who's running a platform tool company, he quickly concluded:

> Cloud platforms are more like fruit baskets. We refine them into in-house fruit salads.

Cloud platforms exhibit a good degree of cohesiveness; for example, through unified provisioning APIs, consoles, monitoring, or security. But a cloud vendor needs to build for the whole world to make the economies of scale work. That means, it's harder for the provider to offer opinionated fruit salads tailored to your specific needs and tastes. That's where in-house platforms shine as they can make more assumptions and can be more opinionated.

Making fruit salad (in-house platforms) from fruit baskets (cloud platforms) is a common enough situation that some cloud platforms come with a knife and chopping board, so to speak. For example, in 2021 AWS introduced Proton, described as a "deployment workflow tool for modern applications", which roughly translates into a fruit salad builder service.

Being more than the sum of the parts is particularly important when a platform lives on top of another platform, such as a cloud platform. Cloud service providers and other vendors continuously adding and integrating new tools into

the base platform challenges the value-add of "fruit basket"–style platforms. Platform teams should therefore decide at the outset whether their platform should be floating or sinking.

Remixing Salads

In the case of in-house IT fruit salads, you face the unusual situation that new fruits come along fairly regularly in the form of new services released by the base platform providers. This poses a challenge as customers will demand those fruits to be swiftly included in your (or rather, their) salad. However, the more balanced your current salad is—in other words, the more cohesive and well-integrated the services are—the more difficult it becomes to seamlessly integrate new services. Fruit salads face this classic architecture trade-off of optimizing for today versus optimizing for the future.

Although I am not fond of military analogies, one product that is frequently labeled as a near-perfect combination of desirable properties is the Japanese "Zero" fighter plane. The combination of excellent maneuverability, long range, low stall speed for use on aircraft carriers, and significant firepower made it extremely effective. However, modifying this design, for example, to achieve higher speed or add armor, proved challenging because any addition in weight would negate the other properties. The best fruit salads are perhaps seasonal.

Learning From the Real World

Knowing that Starbucks isn't the only place to learn about IT architecture and strategy, the next time you visit the supermarket, spend extra time pondering the produce section! It isn't just healthy but also insightful.

18. Cantilevered Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Horizontal platforms sit on vertical pillars.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Fractal Platform Architectures

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Do Control Planes Really Have Control?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Shipping the Org Chart Without Shipping the Org Chart

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Jenga: Stacking Planes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Nesting UI Layers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Nesting API Layers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Developer Portals as Micro-Frontends

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Rotating 90 Degrees: Making Horizontals

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

19. Will Your Platform Float or Sink?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Most people want to swim—until they realize their cost is sunk.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Do Rising Tides Lift All Boats?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Floating or Sinking

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Deciding Is Easy—Until You Get There

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Buoyancy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Underwater Maps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Rising Levels

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

You Pay in Opportunity Cost

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Breaking Changes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

It's Never a Static World

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

20. Beware the Grim Wrapper!

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

What starts well doesn't always end well.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Tailoring the Base Layer

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Configuration

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Intercepting

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Tracking

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Wrapping

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

What Control?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Control Is a Two-Way Street

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Wrapping Considered Harmful

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Keeping up with the rate of change

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Simpler doesn't mean easier

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Unable to benefit from external knowledge

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

You might build the lowest common denominator

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

You need to operate your abstraction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Not All Wrappers Are Grim

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

21. Build Abstractions Not Illusions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Sometimes less is actually less.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Making Things Simpler, but Not Too Simple

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Abstraction: One More Layer Can't Hurt, Can It?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Good Abstractions Are Obvious but Difficult To Find

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Composition Isn't Abstraction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Illusions Illustrated

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Good Abstractions Don't Come Bottom Up

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Domain Models Provide Actual Abstraction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Stringly Typed Domain Models

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Serverless: Distributed System's Essential Complexity

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

What's "Essential"? It Depends, and ChatGPT Won't Tell You

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

The "Happy Path" Illusion

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

22. Failure Doesn't Respect Abstraction

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Time to enjoy a good stack trace!

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Flying on Empty

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Searching for Alpha Particles

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Stack Traces Provide Feedback

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Abstraction Is a Tool, Not a Replacement

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

You Can't Manage What You Don't Understand

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Part V: Implementing Platforms

As with any software-intensive system, implementation isn't just a detail. Platform users will see the implementation but might not see your well-thought-out design considerations. They might not see the control plane of your platform, but those implementation aspects also determine a platform's user experience, its performance, and its ability to evolve.

This part dives into the technical design details of internal developer platforms:

- A platform isn't a single piece but has a distinct anatomy
- Application teams interact with platforms in more ways than with regular applications, including templates and automation languages.
- Platforms provision resources on behalf of application team behalf. Resource ownership and tenancy are therefore critical design decisions for platform builders.

23. Platform Anatomy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platforms also hide their own complexity.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

By Michele Danieli

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Main Components

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Management Plane

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Visual Interfaces

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Portal

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Dashboards

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Programmatic Interfaces

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Command-Line Interface

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Application Programming Interface

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Automation Language

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Control Plane

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

User Management

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Service Catalog

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Service Orchestration

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Services Plane

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Base Services

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Third-Party Services

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Custom Services

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

User-Contributed Services

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Example: Kubernetes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Example: Internal Developer Platform

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Control Plane

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Service Plane

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

24. Platform Orchestration

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

From text processor to cloud compiler.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Templates: Paint by Numbers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Service Orchestration

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Translate

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Deploy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Tracing Back

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Application Code

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Application Architecture as Code

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Next Step: Cloud Compilers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

25. Ownership and Tenancy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Are you selling, leasing, or providing serviced apartments?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Ownership Drives Speed

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Tenancy

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Pushing Tenancy Down the Stack

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Logical Separation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Namespace Separation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Resource-Level Separation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Shared Resources and Fairness

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Ownership

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platforms Are Wholesale Customers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Tenancy Hierarchies

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Bringing Down the Control Plane

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Part VI: Growing Platforms

Platforms thrive on scale, but platform users don't just appear overnight. Instead, they have to be identified, recruited, and carefully nurtured. If your platform is too basic, it will turn off early adopters and fizzle out before it ever gains traction. Vice versa, over-investing into features before you have real user feedback is risky because it could lead you down a wrong path.

This part discusses how you can grow your platform over time to expand its user base.

- Platforms live in three dimensions, so it's natural to model platform evolution as a cube.
- It's easy to assume that each user is "all in" on a platform, but that's not how things start. Plotting users' experience with your platform over time can be a valuable design technique.
- Explaining what your platform delivers versus what users are expected to handle can be surprisingly difficult. Perhaps a picture says 1000 words?
- Maintaining a platform roadmap is a delicate balancing act between accepting user input and sticking to the product strategy.
- You might not be able to serve a wide range of customers with a single offering. Instead, you might need to tier and slice your platform.

26. Platform Evolution Is a Cube

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platforms may be flat, but their path isn't.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

The "Cube"

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Market Reach

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Breadth

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Depth

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

The Three "X's" of Product Lifecycle

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Common Pitfalls

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Perfection Before Actual Use

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Spreading Too Thin

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Depth versus Complexity

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Balancing Breadth Against Depth

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Principles Guide

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Transparency Buys Goodwill

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

User Perspective

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

27. The Shape of Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform adoption is all but linear.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

It's All About the Ramp

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

An Experience Model

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Experience Curves

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Theory

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Ideal

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Cliff

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Hockey Stick

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Gear Shift

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Reality

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Home-Grown Platforms Have Invisible Cliffs

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Reshaping Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

28. Visualizing Platforms

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Seeing is believing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Maps Come First

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Visualizations

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Onboarding Timelines

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Capability Maps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Lifecycle Maps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Operational Maps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Connectivity / Data Flow Maps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Extensibility Maps

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Expressive Visuals

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Architecture Before Product

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Documenting the Platform

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

29. Charting a Platform Roadmap

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Staying on track while laying it.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

By Michele Danieli

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Balancing Completeness With Cohesion

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Customer Feedback Is Biased

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Selecting Metrics

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Sensing the Future

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Panta Rhei

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Retrospectives

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

The House of Quality

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Building Platforms Outside-In

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

30. Tiering and Slicing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Many sizes can fit all.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

By Jean-Francois Landreau

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Scaling Down

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Tiering

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Performance Tiers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Frequency versus Response Time Tiers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Resilience Tiers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Security Tiers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Support Tiers

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Slicing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Feature Flags

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Composable Services

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Stack of Services

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Product Lines

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Part VII: Organizing for Platforms

Platform technology can be complex, and building one requires significant technical expertise. However, addressing sociotechnical aspects, handling the dynamics between people and untangling the organizational knot, are equally relevant. Whether you are setting up a platform team without a platform or building one, leaving these aspects unaddressed spells doom for even the most technically brilliant platform.

Rather than work off service tickets, as would be the case in traditional IT organizations, platform teams enable development teams without injecting themselves into the inner loop of feature delivery. But this doesn't mean that platform teams work in isolation; rather, it's the opposite: they must be connected to parts of the organization as diverse as product engineering, operations, IT and business leadership, HR, and finance. That's why staffing and structuring a platform team isn't easy.

This part provides guidance on how to structure platform teams, the skills they need, and how they interact with other teams:

- Running a platform team can be like running a small company.
- A platform needs well-defined interfaces and align along two axes.
- Platform teams are well advised to place the customer at the center.
- Some platform teams can achieve benefits without actually building a platform.

31. Platform, Inc.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Running a platform team is much like running a company.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

A Product Organization

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

CEO

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

CTO

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

VP Product

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

VP Engineering

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

VP Marketing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Support and Professional Services

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Roles Aren't People

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Fitting It Together

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

From Infrastructure Team to Platform Team

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Cloud Service Teams

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Keeping a Platform Team

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

People wanting to steal your engineers is a high-class problem.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

If they do leave, you had a good deal.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Play your assets.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Personal relationships outlast employment relationships.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

32. Multi-sided Platform Teams

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform teams interface in two dimensions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

By Michele Danieli

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Shifting Gears: Platform Teams Are the Clutch

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Aligning the axes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

East-West: Breaking Down Technology Silos

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Can an extra layer abstract the org chart?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Focus on outcomes to improve collaboration quality.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

North-South: Adoption Is a Two-Way Street

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Internal marketing or the art of dialog.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Listen to your users but don't expect print-ready requirements.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Avoid user churn by listening to feedback.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Maintain a balanced roadmap.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Define What a Platform Means to You

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

33. The Customer-Centric Platform Team

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Self-service doesn't have to be anonymous.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Customer Engagement Models

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Self-Service

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Setup

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Consulting

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Community

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Co-Creation

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Mixing and Matching

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

The Power of Community

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Centers of Excellence

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Which Customers?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

34. Platform Teams Without Platform

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Pyramids last 5,000 years, but diamonds are forever.

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

By Jean-Francois Landreau

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Do Two Pyramids Make a Diamond?

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Keeping the Diamond in Shape

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform-Enabling Teams

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Mental Models Impact Cognitive Load

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Platform Teams Without Platform

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

AI Assist

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Human Giants

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Spikes

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Tailoring

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Pairing

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Training

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Blueprints

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Forcing Functions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Common Transitions

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Cloud Adoption

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

DevOps Transition

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

AI Technologies

This content is not available in the sample book. The book can be purchased on Leanpub at http://leanpub.com/platformstrategy.

Author Biography

Gregor Hohpe is a senior principal evangelist for serverless. Riding the Architect Elevator from the engine room to the penthouse, he connects corporate strategy to technical implementations, and vice versa.

Gregor served as Smart Nation Fellow to the Singapore government, as technical director at Google Cloud, and as chief architect at Allianz SE, where he deployed the first software delivery platform. He has experienced the technology business from most every angle, ranging from start-up to professional services, internet-scale engineering, corporate IT, and cloud services.

Other Titles by This Author

Cloud Strategy, Leanpub, 2020

The Software Architect Elevator, O'Reilly, 2020

Enterprise Integration Patterns, Addison-Wesley, 2003 (with Bobby Woolf)

Michele Danieli is Architecture and Technology Director at Objectway for Financial Advisory in Italy. After starting his career in a garage, writing planning software for telcos, he held technical and managerial roles at Allianz Technology, UnipolSai, and IBM and worked as senior advisor for FIS markets at Imola Informatica. He still loves to build things in the engine room and counters architecture dogmatism with his tag line, Architecture must be Useful not only True.

Jean-François Landreau is a senior solutions architect with AWS. After a decade developing software for financial markets, he followed the shift of collective excitement to SRE and DevOps and never looked back. He is a strong believer that you can't take enlightened enterprise decisions if you are too far away from the engine room.