Deloitte.

Your Business as a Platform

Saul Caganoff - Principal, Deloitte Platform Engineering

@scaganoff

What is a Platform?

Platforms

Platforms Everywhere

Product Platform: *Common design, formula, or a versatile **product**, based on which a family (line) of **products** is built over time.*

Software Platform: *A computing **platform** is, in the most general sense, whatever pre-existing environment a piece of computer **software** or code object is designed to run within, obeying its constraints, and making use of its facilities.*

Business Platform: A **platform** is a business based on enabling value-creating interactions between external producers and consumers. The platform provides an open, participative infrastructure for these interactions and sets governance conditions for them.

Platform Strategy: *A **platform** is typically one that allows others to build freely and openly on top of itself. The more open and free it is, the more rapidly it is likely to grow.*

Platform Thinking: *In construction, a **platform** is something that lifts you up and on which others can stand. The same is true in business. By building a digital platform, other businesses can easily connect their business with yours, build products and services on top of it, and **co-create** value. This ability to "plug-and-play" is a defining characteristic of Platform Thinking.*

Platforms

Diversity, yet Unity

Platforms exist in three main forms:

- within firms as “Product Platforms”
- across firms as multi-product systems
- and in the form of multi-sided markets.

But there is unity in the architecture of these platforms:

- **modularization** of complex systems
- some components remain stable (the platform itself)
- other components are **encouraged** to vary (the complements)
- the most stable elements are the **interfaces** between the platform and the complements.

“The combination of stability and variety makes it possible to create novelty without developing a whole new system from scratch. Thus platform systems are evolvable.” – The Architecture of Platforms: A Unified View, HBR, 2008

Some Example Platforms

Shopping Malls

Manufacturing

Technology

Gaming

Social

Logistics

Mobile Devices

Software as a Service

Platforms
The Platform Ecosystem

Element	Definition	Example
Platform	The extensible system that provides core functionality shared by apps that interoperate with it and the interfaces through which they operate.	iOS, Android, Dropbox, Twitter, AWS, Firefox, Chrome
Apps	An add-on software subsystem or service that connects to the platform to add functionality to it. Also referred to as a module, extension, plug-in or add-on.	Apple Apps, Android Apps, Chrome Plugins.
Ecosystem	The collection of the platform and the apps specific to it. *(I would also add the developers of the Apps as part of the ecosystem).*	Apple iOS, Windows, Linux
Interfaces	*Specifications that describe how the platform and apps interact and exchange information.*	APIs Protocols
Architecture	*A conceptual blueprint that describes how the ecosystem is partitioned into a relatively stable platform and a complementary set of apps that are encouraged to vary, and the design rules binding on both.*	
End-users	*The collection of existing and prospective adopters of the platform.*	

- "Platform Ecosystems", Amrit Tiwana, 2014

Platform Architecture

Architecture is a tool for managing structural complexity.

Complexity

Complexity is managed by **partitioning** (black boxes)

- App developers are partitioned from platform developers
- Partitioning also operates within the platform – e.g. domains
- **Bezo's big mandate** – reduce complexity by reducing interactions to manageable interfaces

Integration is the coordination of development among app developers and platform owner:

- Reduce integration costs by reducing the need for explicit communication - requires modularization and standardization

Architectural Requirements

1. **Simple** – understandable at a high level of abstraction. Interactions between platform and apps should be well-defined & explicit.
2. **Resilient** – one defective app doesn't crash the whole system. Requires weak coupling through stable interfaces.
3. **Maintainable** – changes can be made cost-effectively. Again, requires modularisation and weak-linking through stable interfaces.
4. **Evolvable** – extensible in the near-term and emergent behaviour in the long term.

"Architecture is a tool for balancing the need for autonomy among app developers without compromising the capacity to integrate their work into a cohesive ecosystem."

Platforms

Benefits

For the Platform

Massively distributed innovation:

- aka co-creation, crowd-sourcing
- Partitioning of innovation activities & their integration
- Need to be able to scale to large groups of app developers

Risk Transfer:

- App developers bear the risk of innovation

Competitive Sustainability:

- Network effects catalyze a virtuous cycle
- …once you overcome the "chicken & egg" problem

For App Developers

Technology Foundations:

- Cost of undifferentiated infrastructure is amortized across all developers
- Concentrate only on unique benefits
- Faster time to value
- Economical to tackle niche functions or customization

For End-Users

Mix and Match Customization

- Products can be bundled and customised

Faster Innovation & Lower Cost

- Darwinian competition among app developers can deliver faster innovation
- Cost savings can be passed down the chain

Platform
Challenges

Downsides to Modularization
- Substantial up-front planning and cost.
- Technical performance overhead.
- Architectural innovation is more difficult vs monoliths.
- You need the “right level” of modularization.

The Chicken and Egg Problem
- App developers need the platform before they can build.
- The platform takes time to build.
- How can we co-evolve the platform and apps?

Attracting & Engaging Developers
- The platform needs to be simple to use but powerful in application
- We may think enterprises have a captive audience, but SOA experience is that they can choose to ignore the platform

Your Business as a “Technology Platform”

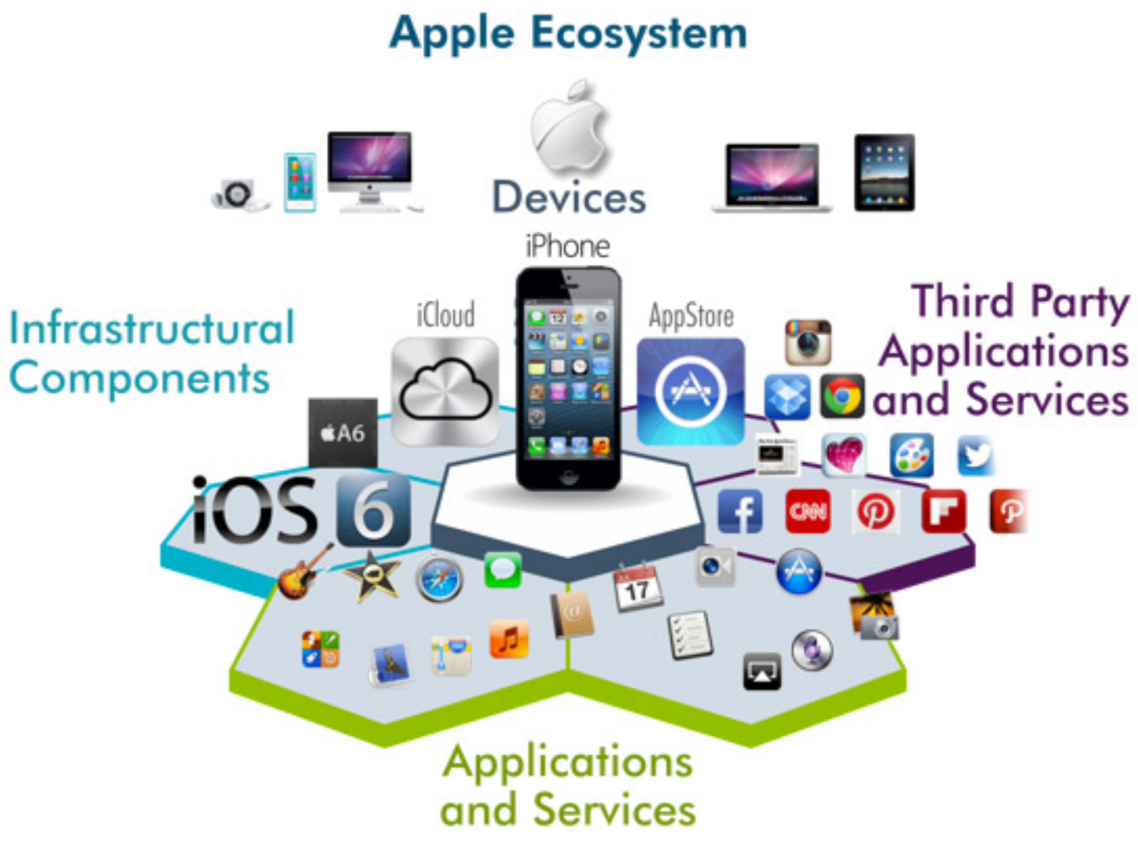
Architecture

Example: The Apple iOS Platform

Critical features of the Apple iOS Platform

The Apple iOS Platform provides the following features:

- **Hardware** – the iPhone, iWatch, iPod, iPad, MacBook etc
- **Software** – iOS operating system, APIs, SDKs, Programming Languages
- **APIs** – the interfaces that bind Software to Hardware via iOS
- **Shared Services** – Security, Notifications, Location, Accelerometer, Communications, etc
- **Education** – Manuals, conferences, workshops, books, evangelization
- **App Management** – registration, review, publication, discovery, delivery


Apple Ecosystem
Devices
iPhone
iCloud
AppStore
Infrastructural Components
Third Party Applications and Services
A6
iOS 6
Applications and Services

The Programmable Enterprise

Critical features provided by an Enterprise Platform

Imagine your enterprise as an operating system...as a platform like Apple iOS. What do you provide?

- **Core functionality** – both generic and specific to your business
- **Core data** – both generic and specific to your business
- **APIs** – Stable interfaces to core functionality and data
- **Shared services** – Security, Communications, Logging/Monitoring, Notifications (Events)
- **Education** – training, documentation, hackathons, SDKs, evangelization
- **App Management** – registration, review, publication, discovery, delivery

The key value is in what is "*specific to your business*" because that is the area of differentiation and it is built on top of "undifferentiated heavy lifting".

The Programmable Enterprise

How do Platform Elements map to an Enterprise Platform?

Element	Whom
Platform	IT Department (**Platform Engineering**) • Developers, operations, DevOps teams provide the platform
Apps	IT Department (**Solutions Engineering**) • Developers use the platform to deliver solutions for the business & customers
Ecosystem	The enterprise! External stakeholders, partners, suppliers, customers.
Interfaces	APIs Notifications
Architecture	Platform Architecture Development standards (interfaces) Governance and QA
End-users	Business stakeholders Customers

App developers and Platform providers:

- both part of a traditional IT department
- become standalone entities

Platform teams (operations and DevOps) now become a "profit center" rather than a "cost center" for the business.

- a16z Podcast on Microservices, Sept 2016

Platform Enablers
The Value Platform - APIs

Developer **engagement** is critical

Treat your API as a **product**

- product management
- with a managed lifecycle

API **Catalogue** – make APIs easy to find & use

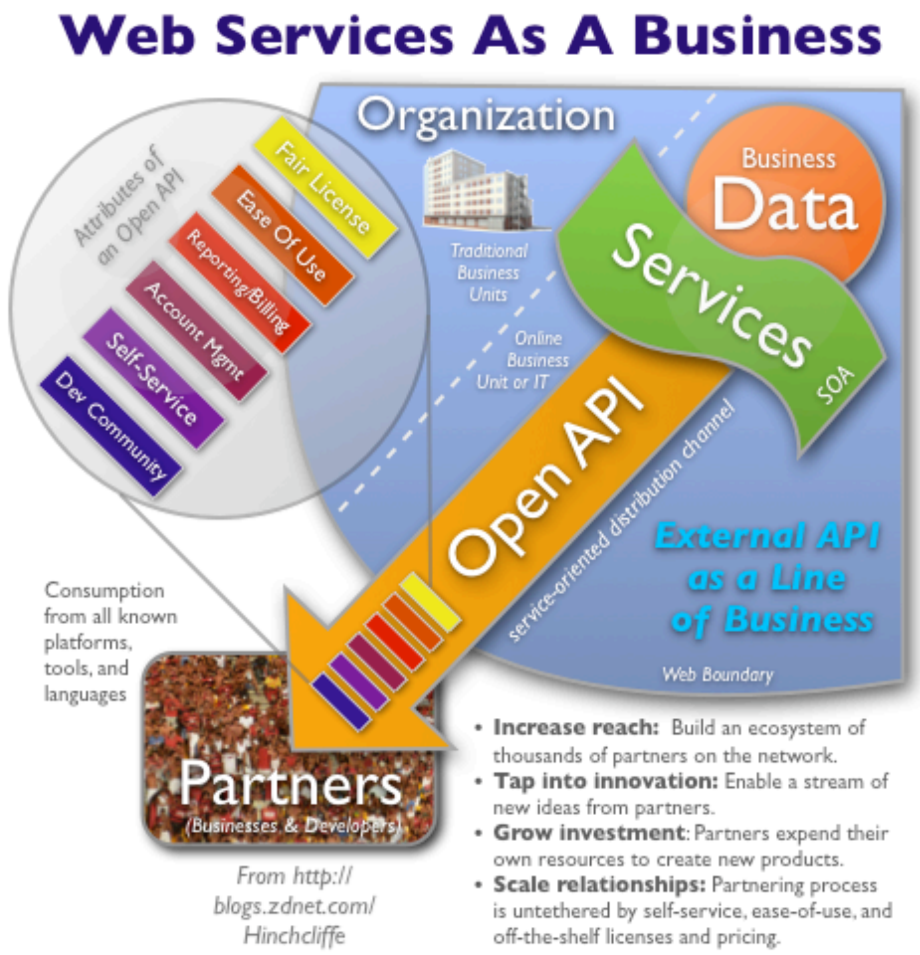
Metrics & **Analytics**

Self serve governance

API first design

Strict **separation** of providers & consumers

Security & Policy **Management**


Web Services As A Business
Organization
Attributes of an Open API
Fair License
Ease Of Use
Reporting/Billing
Account Mgmt
Self-Service
Dev Community
Traditional Business Units
Online Business Unit or IT
Business Data
Services
SOA
Open API
service-oriented distribution channel
External API as a Line of Business
Web Boundary
Consumption from all known platforms, tools, and languages
Partners
(Businesses & Developers)
From http://blogs.zdnet.com/Hinchcliffe
• Increase reach: Build an ecosystem of thousands of partners on the network.
• Tap into innovation: Enable a stream of new ideas from partners.
• Grow investment: Partners expend their own resources to create new products.
• Scale relationships: Partnering process is untethered by self-service, ease-of-use, and off-the-shelf licenses and pricing.

Dionne Hinchcliffe, ZDNet, 2009

Platform Enablers
The Delivery Platform - Platform as a Service (PaaS)

PaaS abstracted above infrastructure & people

- Enable rapid delivery without compromising enterprise maturity

DevOps & Microservices

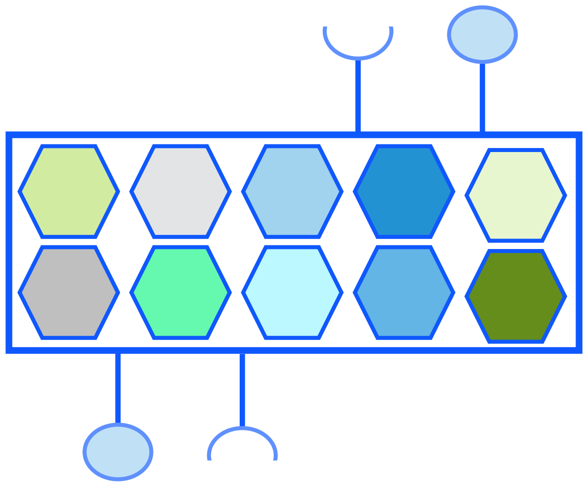
- Development team autonomy
- Applications, customizations, API providers, data management etc.

Hybrid Ready

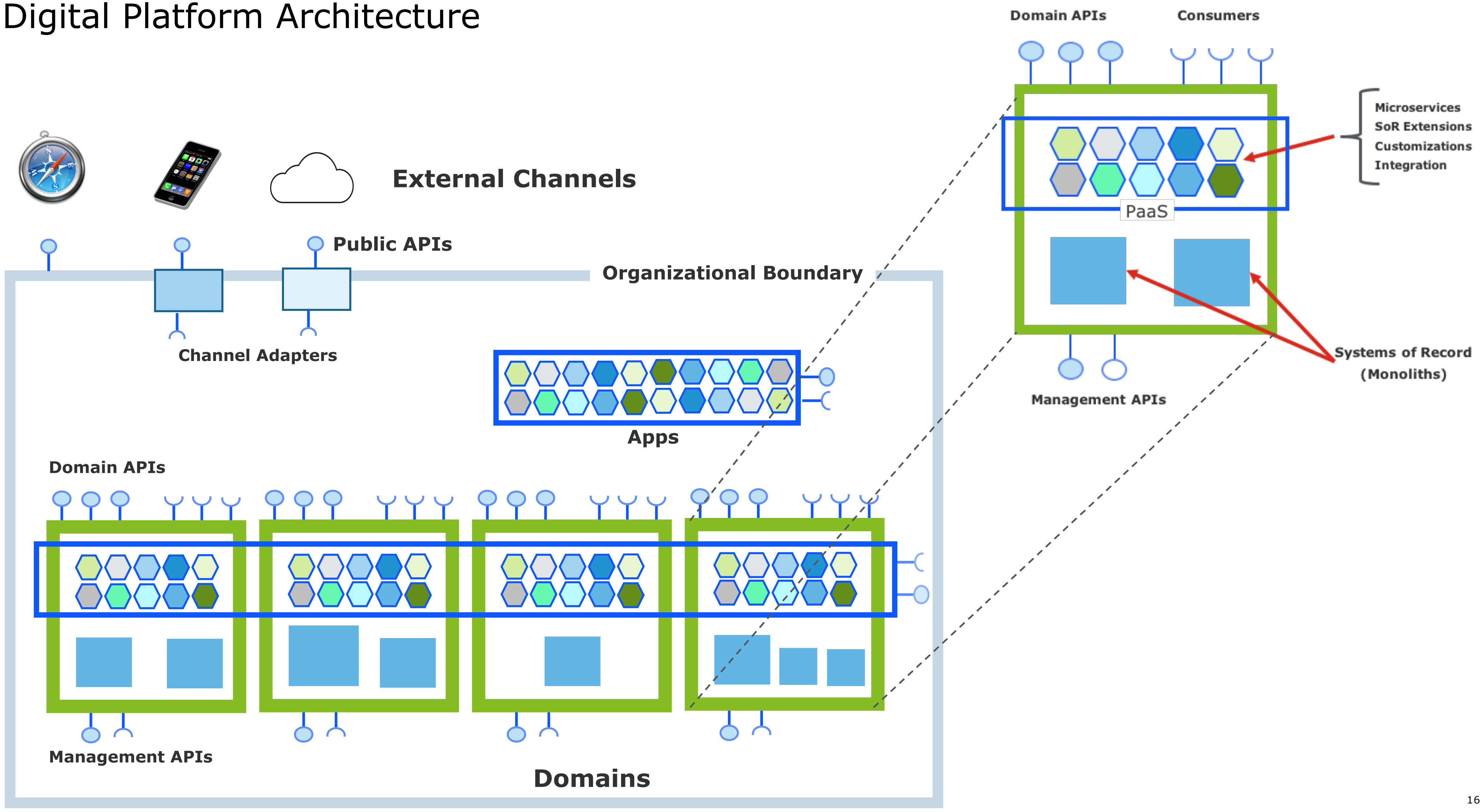
- Stretch across cloud and on-premise

PaaS Services

- Monitoring, logging, security, auto-scale, load-balancing
- Multi-tenant data store (e.g. Cassandra)
- Multi-tenant messaging
- API Management



Digital Platform Architecture



Your Business as a “Business Platform”

Governance & Economics

Business Platforms
Multi-Sided Markets

Most Business Platforms are **multi-sided markets** involving external producers and consumers:

- **Pull**: the platform needs to attract producers and consumers
- **Facilitate**: the platform facilitates interactions the tools, rules and environment.
- **Match**: the platform matches compatible providers and consumers.
- **Chicken & Egg Problem**: (again) how do we attract providers without consumers and vice versa?

Network Effects

- **Positive** Network Effects – value increases with the number of participants on the platform.
  - e.g. "winner takes all" markets
- **Negative** Network Effects – value decreases with the number of participants:
  - matching inefficiency
  - proliferation of "bad actors"

Business Platforms

Governance is a tool for managing behavioural complexity.

Governance is best applied at the interfaces rather than on internal implementation details.

- Ensures compatibility, interoperability and ability to orchestrate.
- Providers & consumers are well-behaved "citizens".

Culture shifts from building everything to assembling and orchestrating services.

People concentrate on high value activities as "undifferentiated" tasks are automated and absorbed by the platform.

Your Business as a Platform
Key Takeaways

Platforms come in many forms.

Common Platform Characteristics:

- Modularization
- Stable core
- Complements are encouraged to change
- Interfaces (APIs) are critical for stability, control & evolution

Platform concepts can benefit your business.

Technology platform concepts:

- Leverage core capabilities
- Allow & encourage change & innovation
- Are enabled by APIs and PaaS technologies

Business platform concepts:

- Demonstrate the right level of governance
- Recognize the importance of:
  - People
  - Culture
  - Economics

  for platform success.

References

[1] “Platform Revolution”, Parker, Van Alstyne & Choudary, WW Norton & Co 2016

[2] “Platform Ecosystems”, Amrit Tiwana, Morgan Kaufman, 2014

[3] “Product Strategy for High Technology Companies”, M.E. McGrath, 2001

[4] “Invisible Engines”, Evans, Hagiu, Schmalensee, MIT Press, 2006

[5] “Three Elements of a Successful Platform", HBR, 2013

[6] “Platform Strategy” Blog, Sanjeet Choudary

[7] “The Architecture of Platforms: A Unified View”, Baldwin & Woodward, HBR, 2008