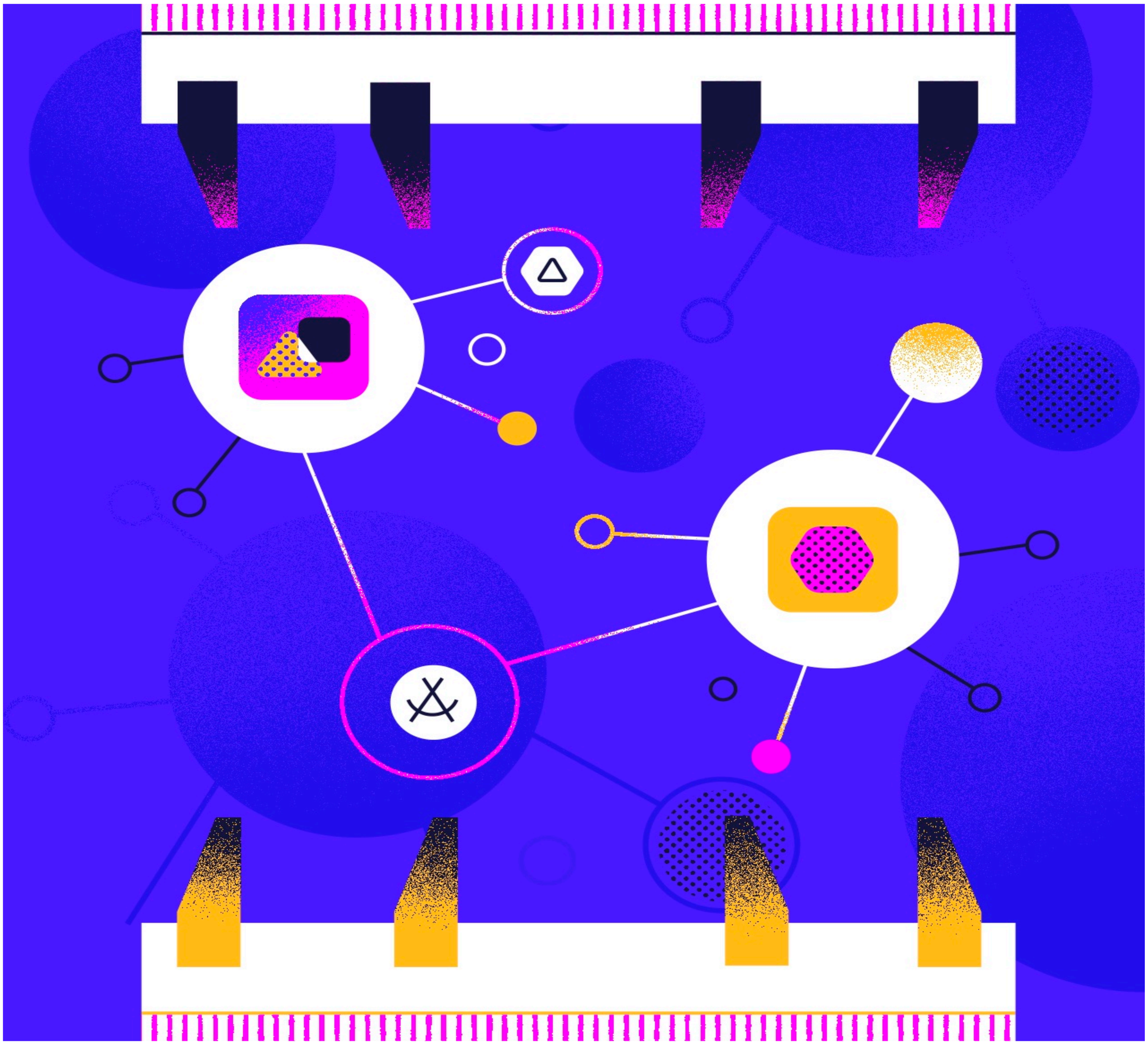
BASIS SET VENTURES

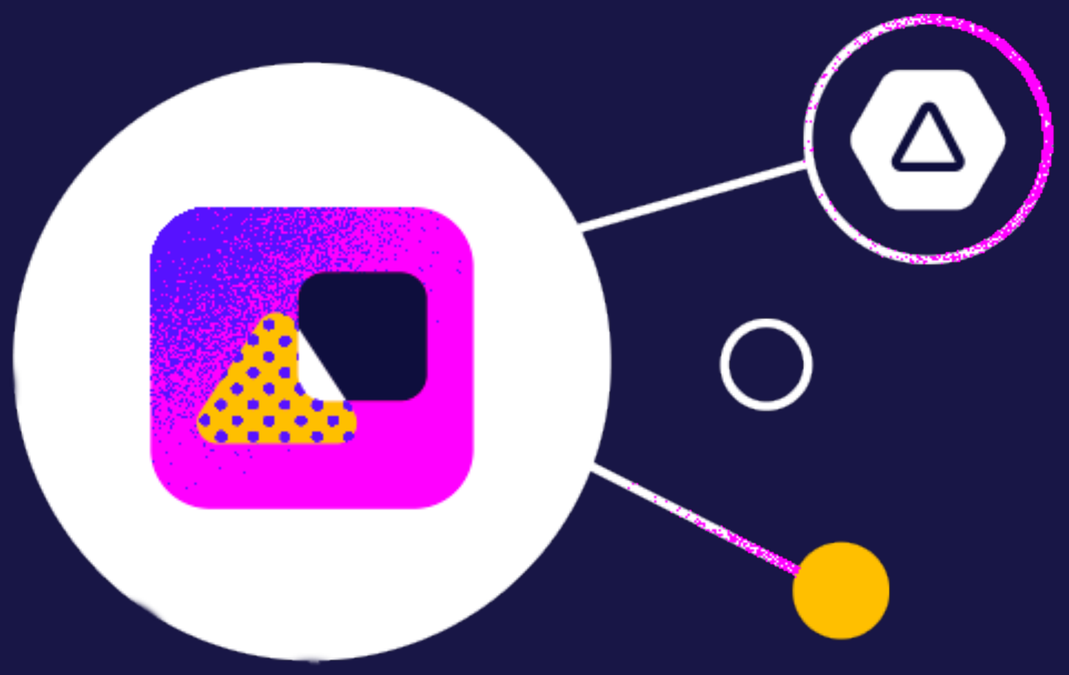
Vertical APIs.

Exploring opportunities to enable vertical APIs with machine intelligence.



basisset.ventures

Table of contents.



VERTICAL API Definitions

API

Code that governs the access point of a server

Vertical API

For the purposes of this research, I'll be limiting this discussion to "process" APIs (as opposed to data APIs for example). Process APIs further break down into "transactional" APIs and "service" APIs. These are my terms — no formal taxonomy, from a business model perspective, exists within the API ecosystem.

We will be further limiting discussion to APIs that automate specific business functions or automate tasks within a specific industry vertical. These APIs enable a full process to be outsourced and served via API.

Process API

Facilitate the complete outsourcing of an end-to-end business function. A request is put out via API and an output is returned, either in the form of a service or a transaction.

Transactional API

The familiar API calling model where pricing matches the number of times an access point is tapped. Marketplace dynamics. High velocity preferred. Often low pricing per transaction. Value is rooted in the network being accessed via API abstraction.

Service API

SaaS-like API business model whereby a function is outsourced but pricing is recurring and pre-fixed. Lower velocity. Can reach into enterprise-grade pricing independent of volume. Value is driven by automation and ultimately the service being delivered via API.

Service API

Example Business Functions

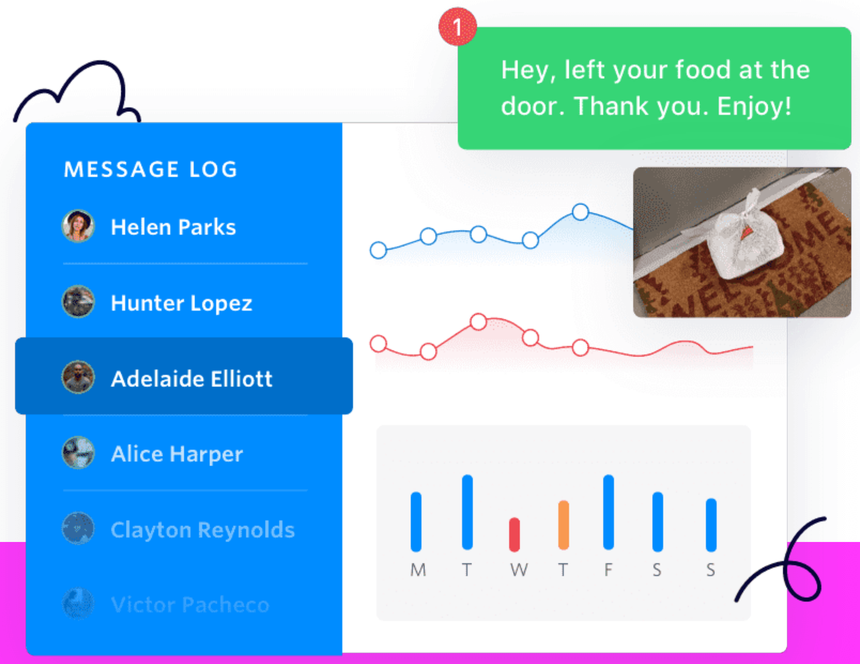
Example Business Functions

- Background checks
- Shipping functions
- Payments
- Bank transactions

Example Industry Verticals

- Mortgage origination
- Banking/finance KYC

Anatomy of a “transactional process” API business



twilio

Twilio was one of the first VC “home runs” in the API space. Referencing the above taxonomy, it is a transactional process API. Twilio is a communications API enabling developers to easily add text messaging, calling and other communications functions directly into applications.

If you’ve called an Uber, you’ve received a text message enabled by Twilio. There are many lessons to be learned but the main one is that APIs can monetize at multiple levels of abstraction.

Images from twilio.com

API ANATOMY

1

Network layer

Manual painstaking integration into global network

Twilio Example

Direct integrations into global telecommunications infrastructure

2

Primary service layer

Customizable functions directly leveraging network layer integrations

Twilio Example

Voice, messaging and video APIs

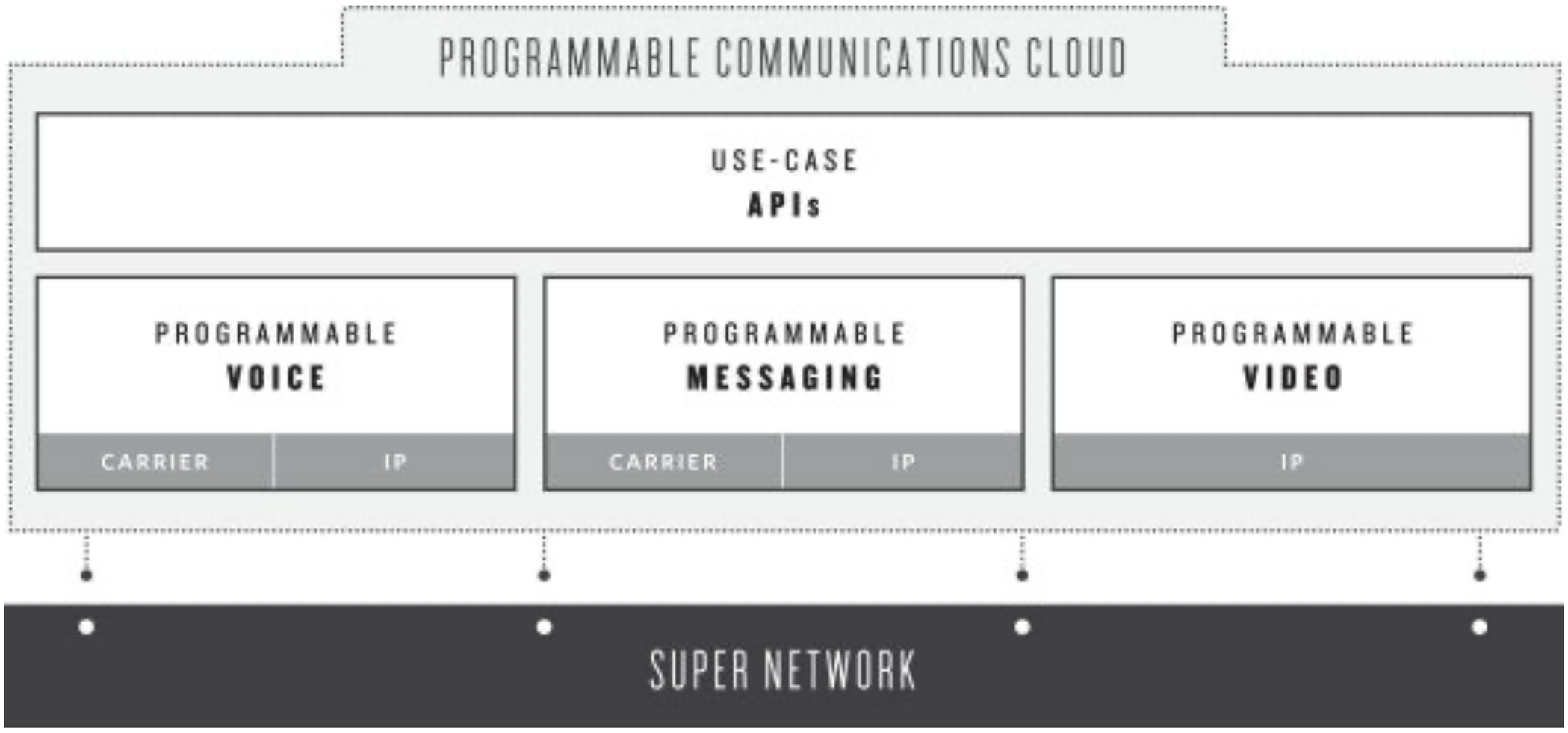
3

Secondary service layer and abstracted layers

Use-case specific functions bundled together and sold to developers

Twilio Example

2FA, account notifications



Taken from Twilio S1

Ingredients for a “transactional process” API business

While all APIs are different, the “network layer” for all modern process APIs that have achieved scale possess a common set of properties. These properties will help us identify markets that are low hanging fruit for future APIs.

Legacy network must have many disparate nodes

Sufficient friction must exist such that developers have a reason to outsource to an API. Twilio notoriously got hit when Uber reduced usage after attempting to build Twilio in-house. 15% of Twilio revenue was wiped off their books post-IPO in dramatic fashion.

Network must have the right balance of concentration, often consolidating

A consolidating market answers the “why now” question. Telecom for example is still fragmented but significantly consolidated from even a decade ago.

Power lies with developers so most companies go bottom-up

Developers are the kingmakers in the API world. Documentation matters. Usability matters. Twilio won over competition because developers loved it and it was really easy to use with lots of pre-built modules at the higher layers of abstraction.

Networks must be very large, frequently global

It must be difficult enough to build integrations that integrations become defensible. There must not be so much fragmentation as to render it impossible to digitally consolidate nodes in a timely fashion.

There must be a high velocity of user actions

Similar paradigm to marketplace transaction velocity but with an API, two software systems are interacting

“Take rate” here is analogous to per API call, where “pay-as-you-go” pricing is a similar proxy for value proposition. Pricing can range from a few pennies (especially with up front volume commitments) to $10s of dollars in the case of lower velocity products like background checks.

Lesson: APIs are getting more complex over time

Traditional API
“Transactional Process”

- Low pricing
- High velocity

Example

twilio

2008

Primary
Service Layer

- Medium pricing
- Medium velocity

Example

spruce

2016

SaaS API
“Service Process”

- High pricing
- Low velocity

Example

Skyflow

2019

COMPLEXITY

Framework for Thinking About API Complexity Over Time

Automation is becoming a modular part of code that can be dropped in by developers in enterprise. Sophistication of buyers in financial services for example has bred an entirely new class of hybrid and “service process” APIs that do not require high transaction velocities to scale.

The value proposition of these APIs lies in their ability to automate processes at the infrastructure layer, as opposed to the traditional SaaS platform layer. This is fundamentally different from the network value propositions of traditional “transactional process” APIs.

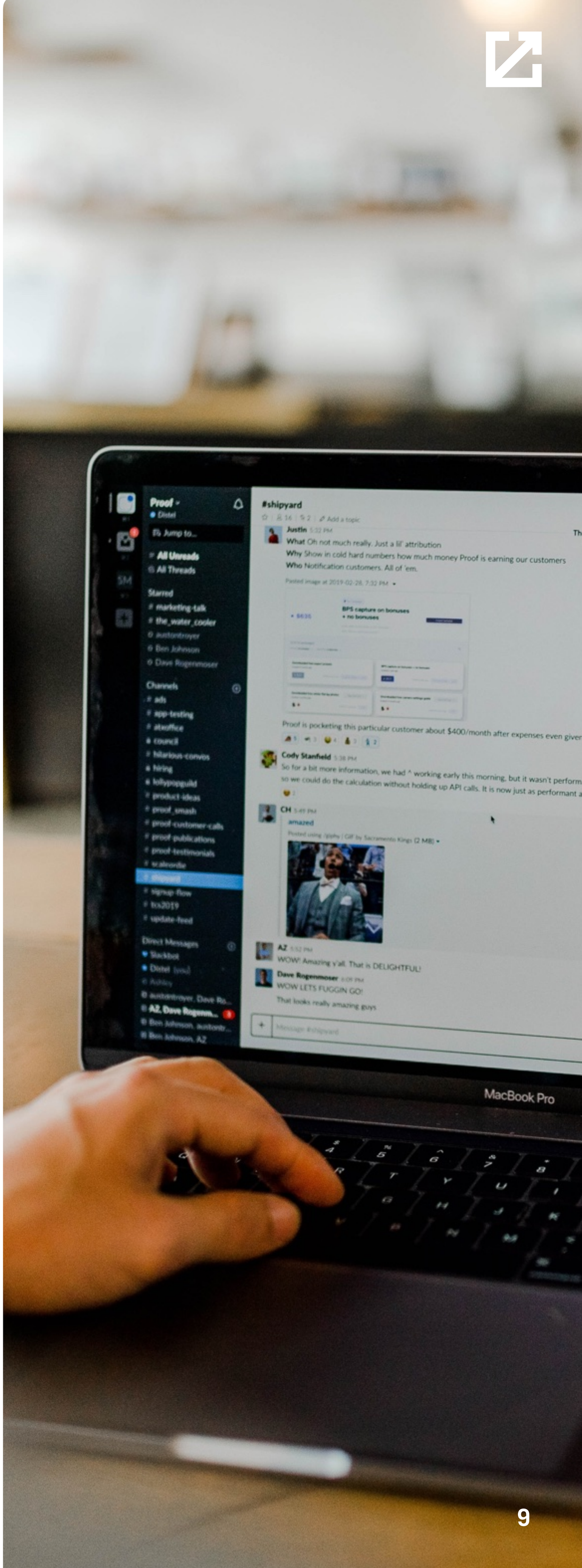
The timing is perfect

The proliferation of SaaS is at a tipping point, particularly within traditional industry. The more the world digitizes, the more feasible it becomes for APIs to connect independent systems across enterprise lines.

Fragmented ERP systems are difficult to integrate with at scale, but SaaS is driving software consolidation across industries.

The data connector from inside enterprises naturally abstracts into the physical world. Much of the world's physical infrastructure remains offline and inaccessible to developers. APIs today make it possible for business HR platforms to talk to government databases of personal background data. APIs in the future might be able to make it possible for transportation management systems (TMS) to talk to warehouse management systems (WMS) at storage facilities across the country. The recurring theme is communication and data — the highest velocity of activities.

New players with the foresight to find and build valuable digital roads connecting software with physical infrastructure are prospectors, profiteering on digital tolls. Roads are expensive to build. Most don't have sufficient traffic to justify tolls. But every once in a while, the right person at the right time owns the Oakland Bay Bridge, where travelers have little choice but to pony up or buy a raft.



Emerging areas

HIGH VELOCITY B2B TRANSACTIONS

Scheduling

Scheduling, in nearly every industry, requires an understanding of labor and physical capacity constraints. Scheduling within enterprises is hard enough — most can relate to the challenges of wrangling availability even when preferences are known. But scheduling across enterprises, particularly for operations purposes is even more difficult.

Escrow

Spruce did escrow for real estate but clearing house functions are a critical part of many other businesses. Escrow as a service delivered via API is worth a second look, especially for international payments.

Licensing

Picking up where Checkr left off with specific licensing solutions for regulated industries like healthcare, insurance and construction contracting.

Procurement

Inventory is something that is not easily accessible today. If a founder could create a network of inventory, even starting in a single space, this could be valuable for procurement teams.

END-TO-END AUTOMATED PROCESSES

An alternate take on full stack businesses where an end to end automated process is delivered via API. Privacy is the hot vertical now, it’s worth thinking about what could come next here and benefit from a SaaS pricing model.

NEW NETWORKS BOOTSTRAPPED BY CUSTOMERS

Radar is a pretty cool case study on an API bootstrapping its own network through partnerships and sales. The company has geolocation data and labeled retail data for thousands of stores. The leg work was done by customers, limiting the manual integration work. Worth digging more here for other potential variations of this model.

Challenges

Customer satisfaction

Many of the API-first companies discussed in this report struggled with customer concentration in the early days. Checkr and Twilio were both effectively made on the back of Uber's growth (and the growth of the gig economy). It does seem that timing is critically important in launching APIs, building developer excitement and overcoming customer concentration challenges.

Mitigating factor

Believe in the growth of the curve

Narrow timing

Unlike most SaaS products, APIs rely on emerging verticals for success. As a result they're especially dependent on inflection points. Builders need to be building with new tools to justify their existence and startups and tech companies are almost always early adopters.

Mitigating factor

Look for an emerging class of startup "first-movers"

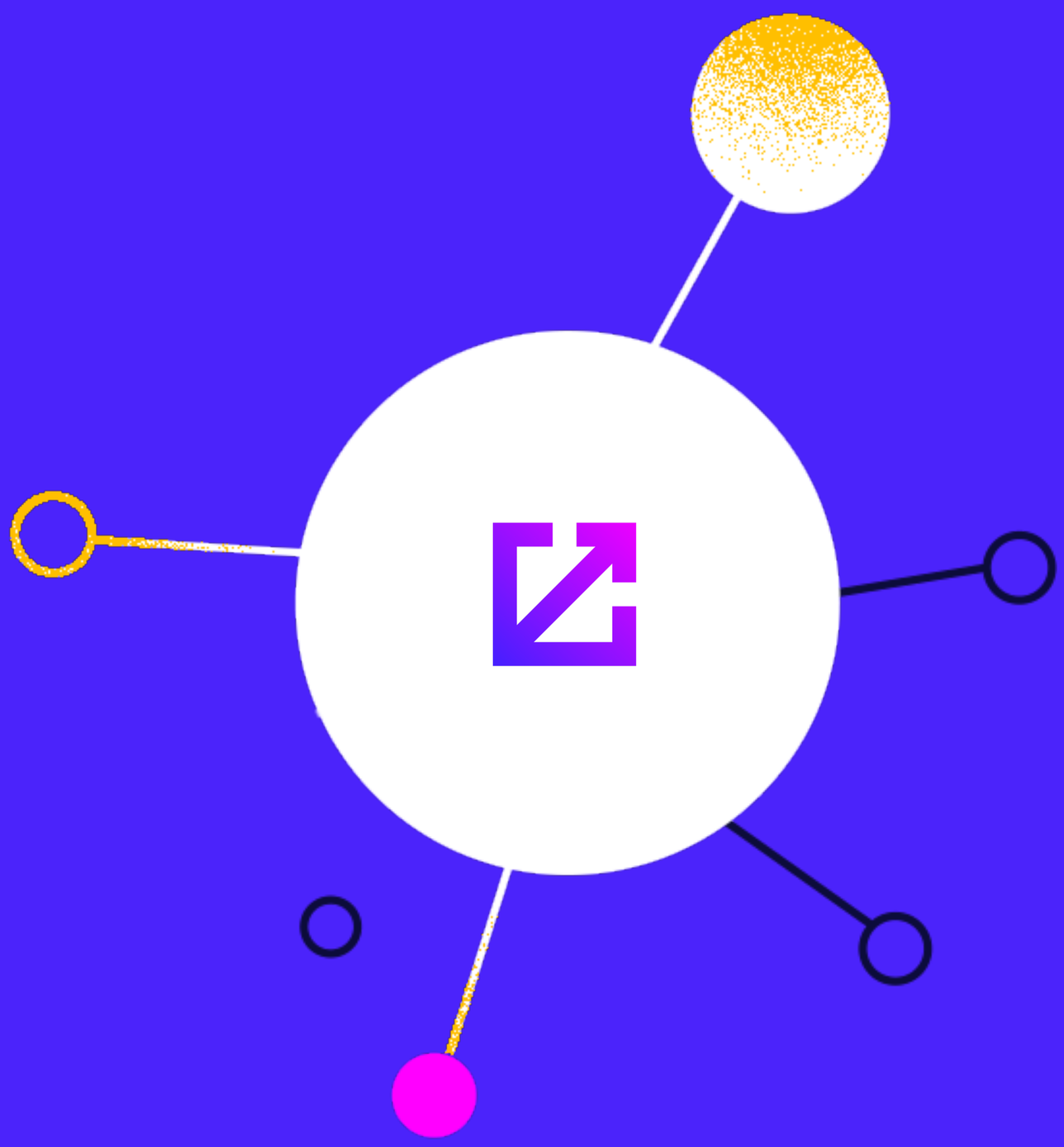
Lack of market sophistication

Many companies in traditional industry do not have the resources to integrate APIs into their core infrastructure. Some work is done by outsourced development consultancies but we don't yet have an example of an API-first company getting started this way. This is the primary reason why B2B API use cases lag their B2C counterparts.

Mitigating factor

Business process APIs have an advantage over entirely vertical APIs because adoption can first spread in a wider array of places.

BASIS SET VENTURES



basisset.ventures