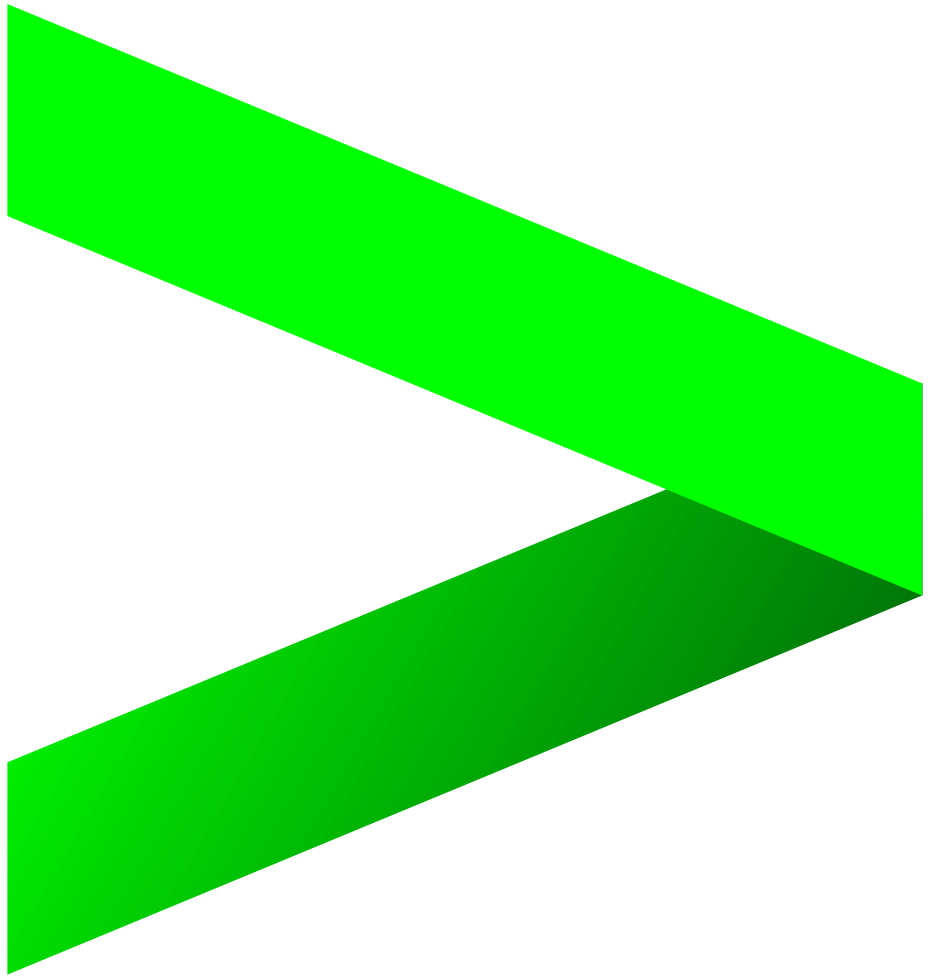
DIGITAL DECOUPLING



OPENMUNICH 2017

Tobias L. Maier

accenture technology

HI!

Tobias L. Maier

Technology Architect

Emerging Technology Innovation

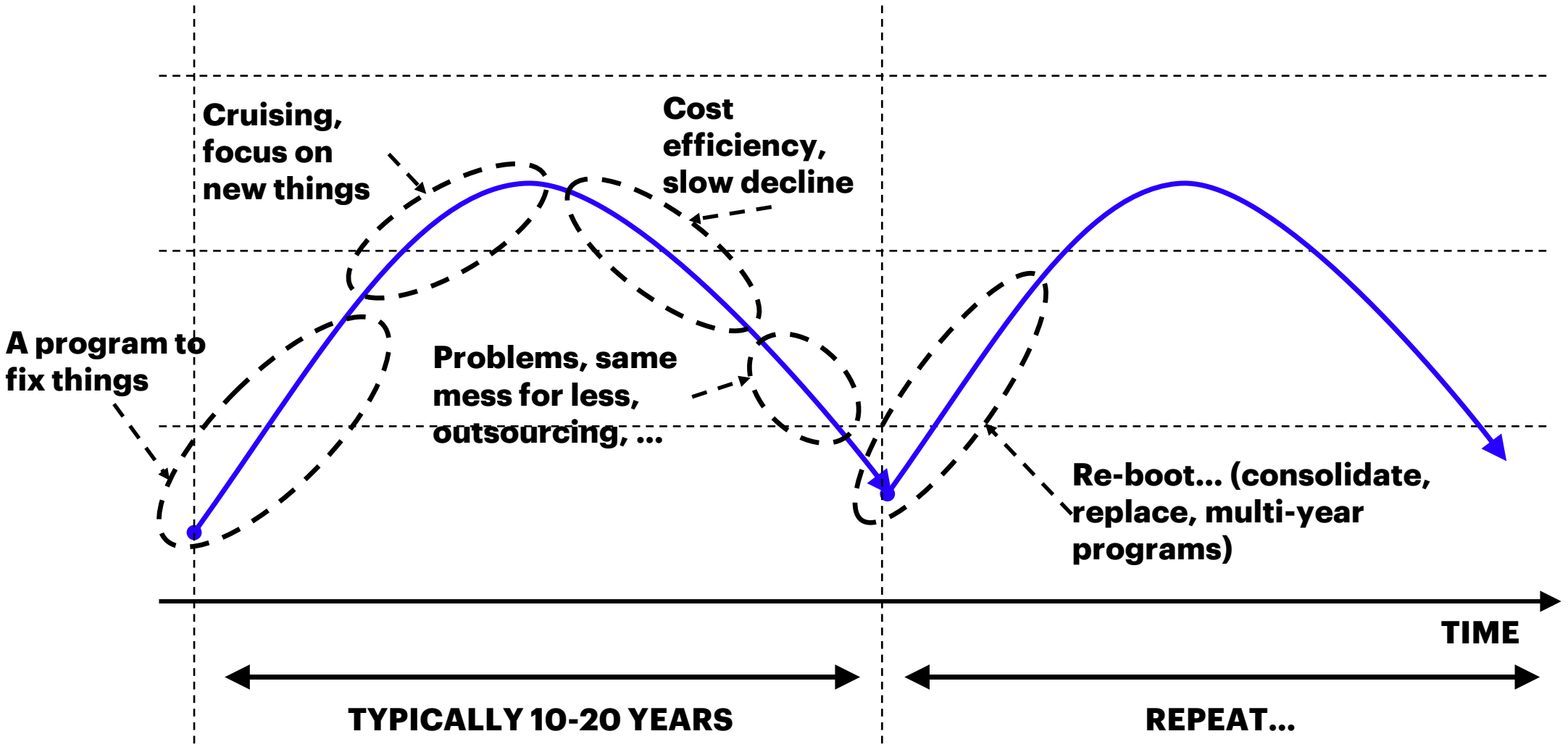
accenture technology

@tobias_maier

Photo © Niklas Altekamp – www.niklas-altekamp.de

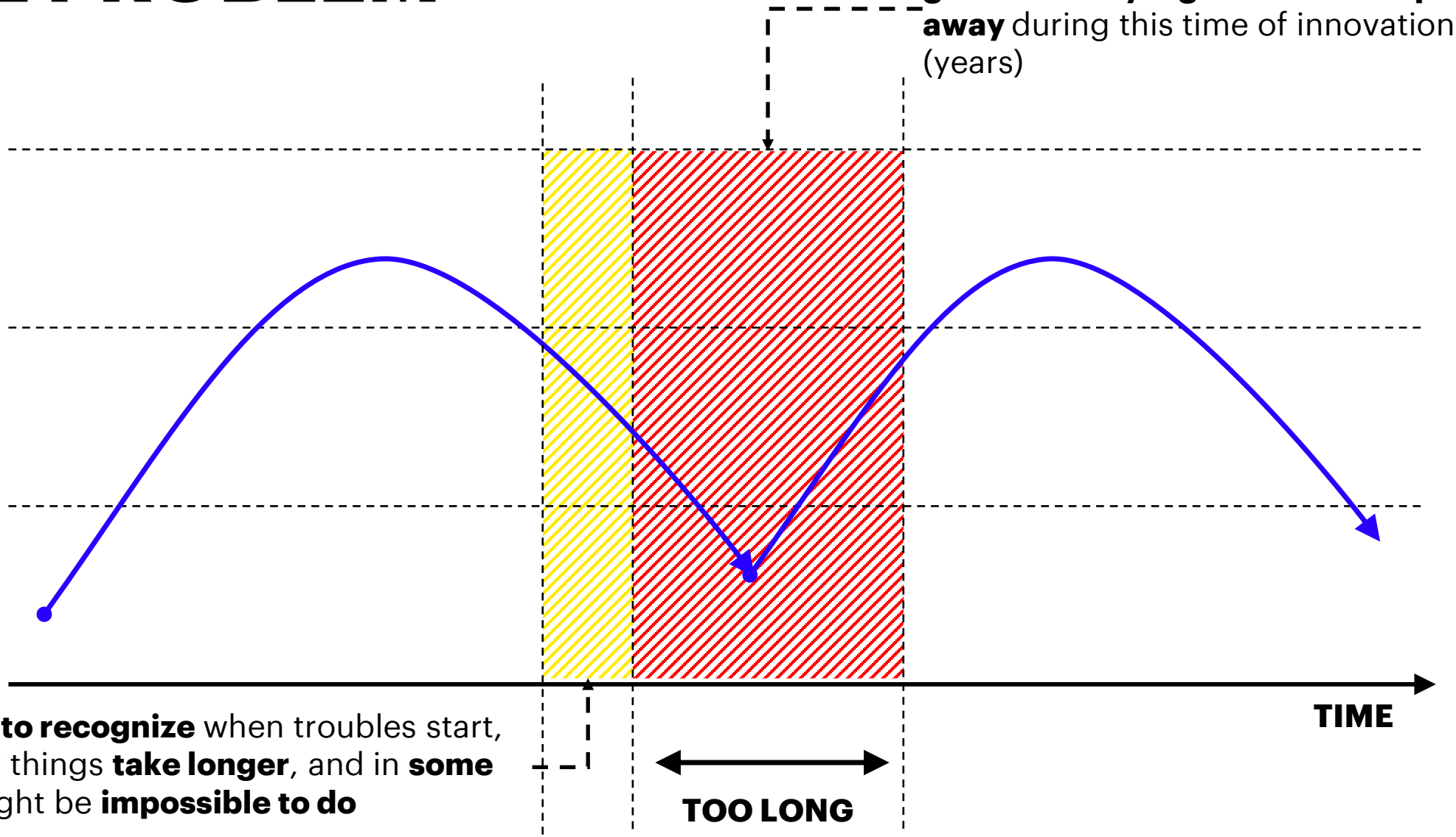
THE DISCONTINUITY MODEL

THE BASIC MOVEMENT PATTERN



THE PROBLEM

It **takes too long to get out of the hole**. The **cost to get out is very high** and the **competition can run away** during this time of innovation inactivity (years)



Not able to recognize when troubles start, suddenly things **take longer**, and in **some** cases might be **impossible to do**

THE 7 ANTI-PATTERNS ACCELERATING THE MOVEMENT

7 ANTI-PATTERNS

1. **Consolidation as the only approach, driven by**
2. **A product alone solves a business problem**
3. **A perfect architecture and governance**
4. Functional end state is reachable
5. Adapt IT systems to meet all business processes requirements
6. Co-dependent relationship between business and IT
7. All old = bad

CONSOLIDATION AS THE ONLY APPROACH

Driven by

- Need to standardize processes across the organization
- Need to create more real-time solutions
- Need to reduce costs in a spaghetti landscape

7 ANTI-PATTERNS

A PRODUCT ALONE SOLVES A BUSINESS PROBLEM

Relying on someone else having the hammer

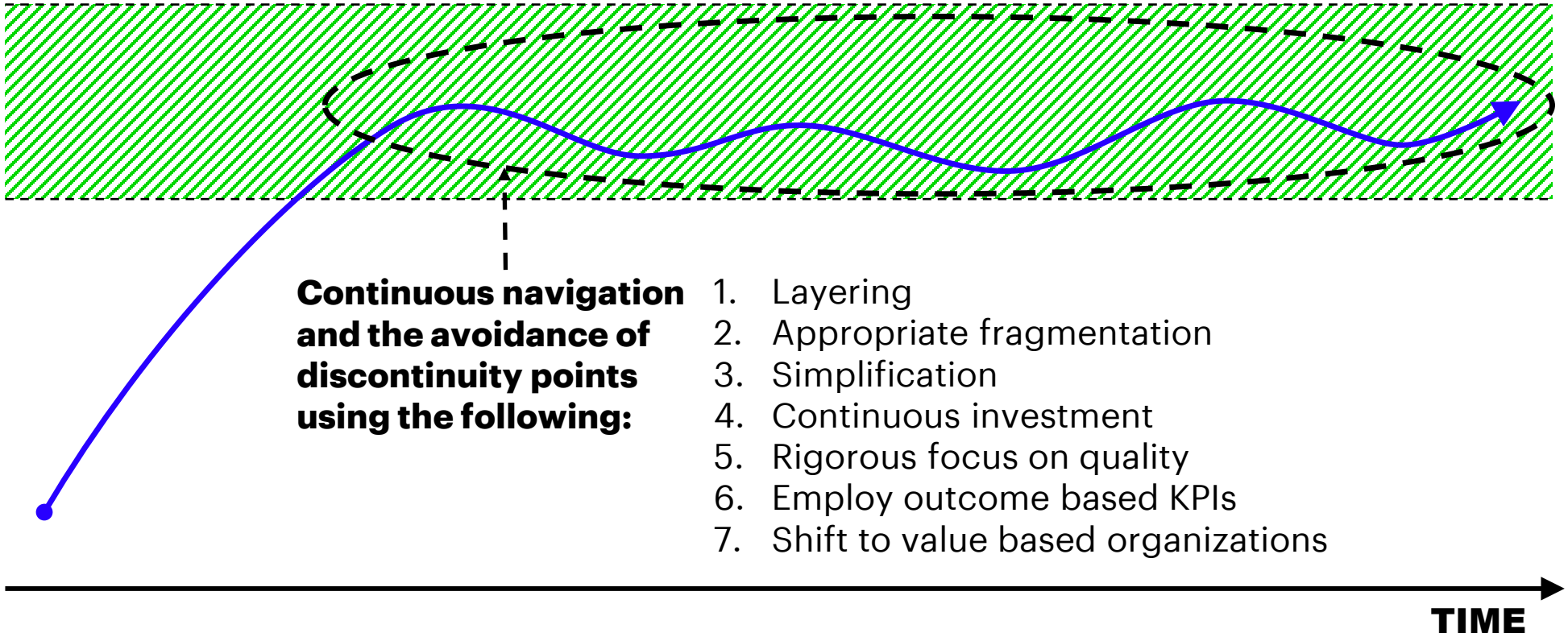
7 ANTI-PATTERNS

A PERFECT ARCHITECTURE AND GOVERNANCE

- Ivory tower architecture
- Focus on standards that don’t drive structure – e.g. canonical data models

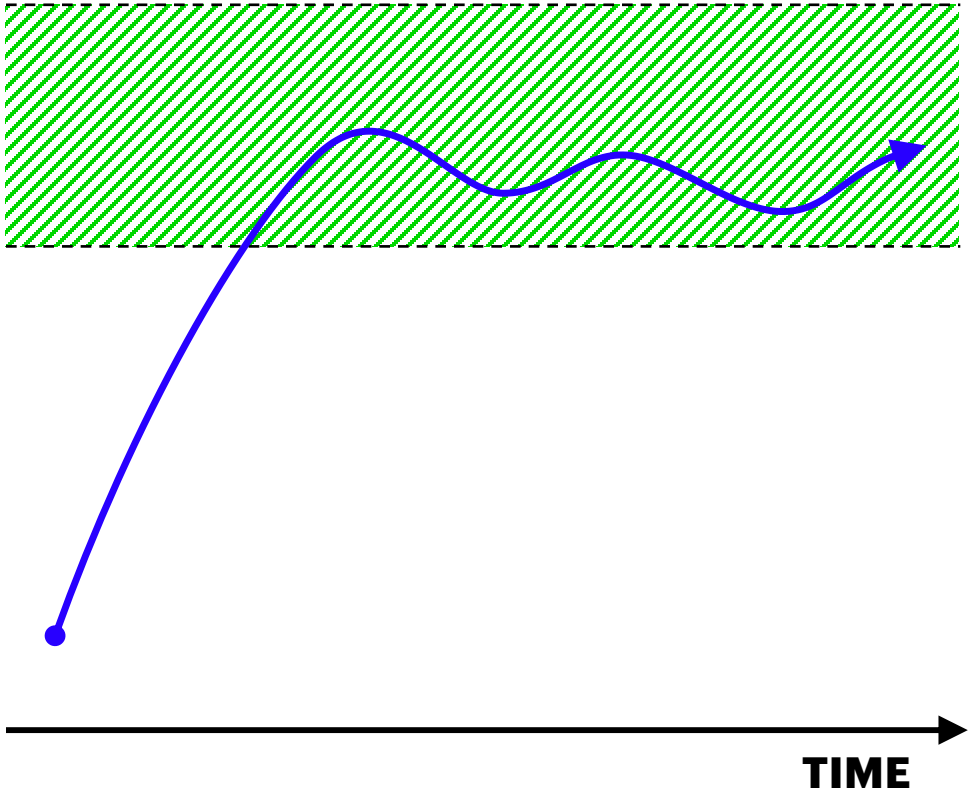
7 ANTI-PATTERNS

THE 7 HABITS OF SUCCESSFUL IT

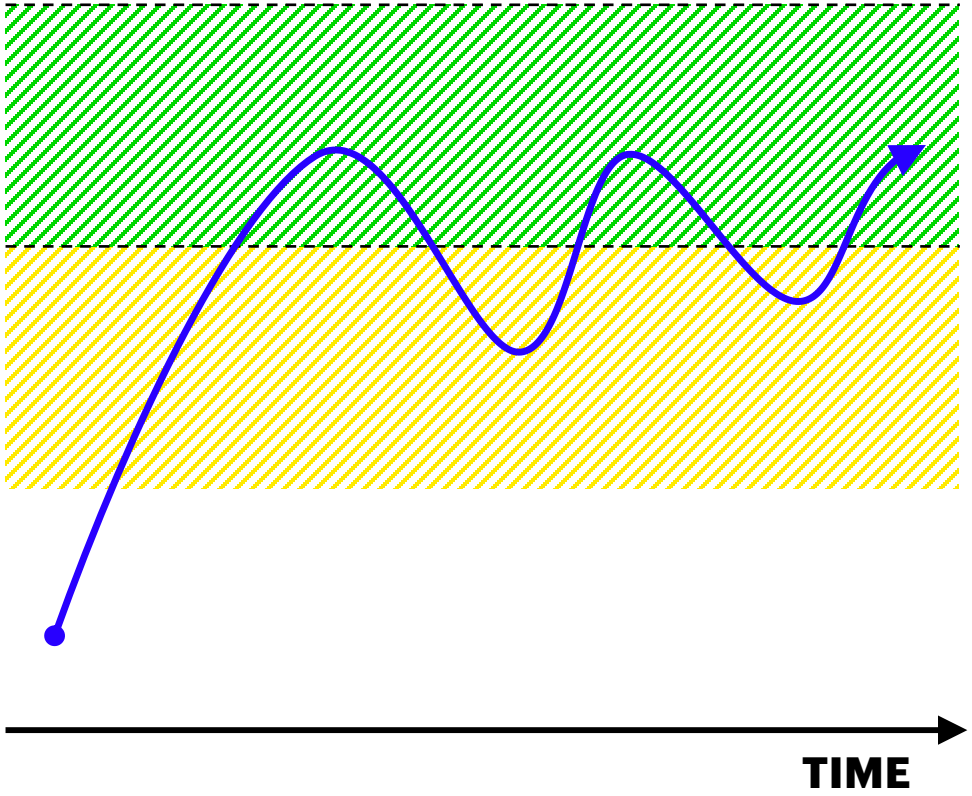


REALITY CHECK

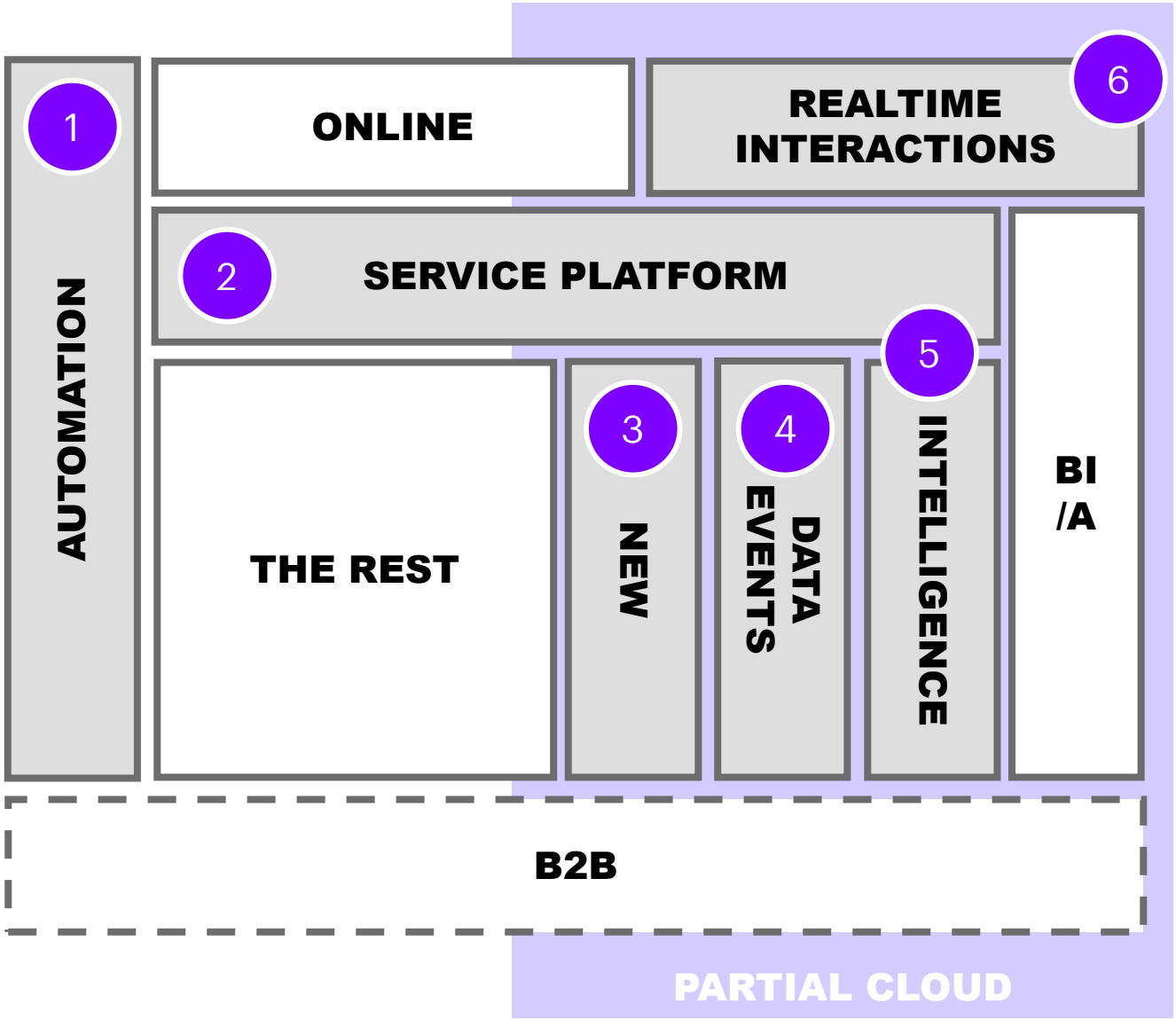
DO YOU NEED FULL TECHNICAL PROTECTION FROM DISRUPTION?



OR CAN YOU TAKE A BIT LESS DEMANDING APPROACH, MIXING EXPONENTIAL IT WITH CLASSICAL MODELS?



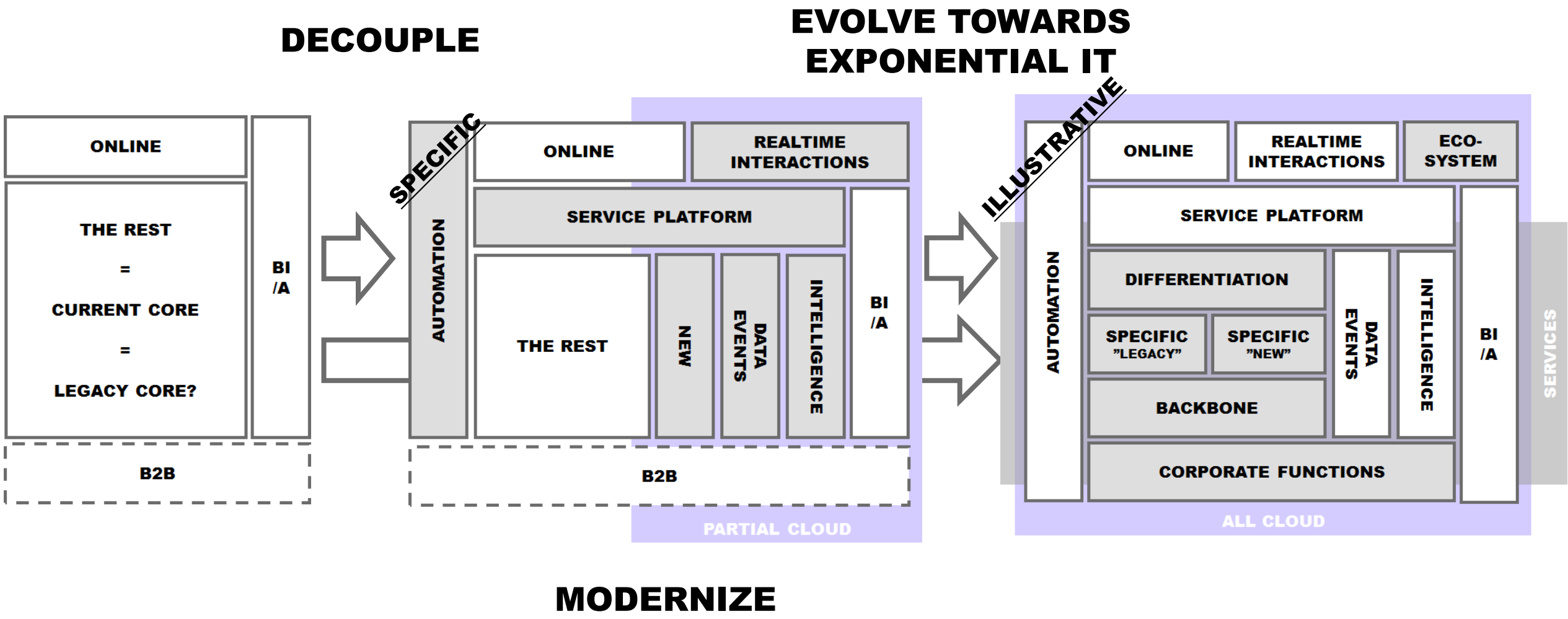
DIGITALLY DECOUPLING THE CURRENT IT



1. Automate using RPA
2. Build out differentiated services using Microservices, DevOps and the cloud and provide API access to the common core
3. Utilize Cloud to quickly build new core services
4. Build out a data lake with real-time eventing capabilities
5. Build systems of intelligence
6. move ”beyond the shopping cart”, build for example voice and chat based online services

+ Seek combinatorial effects
+ Maximize cloud for new

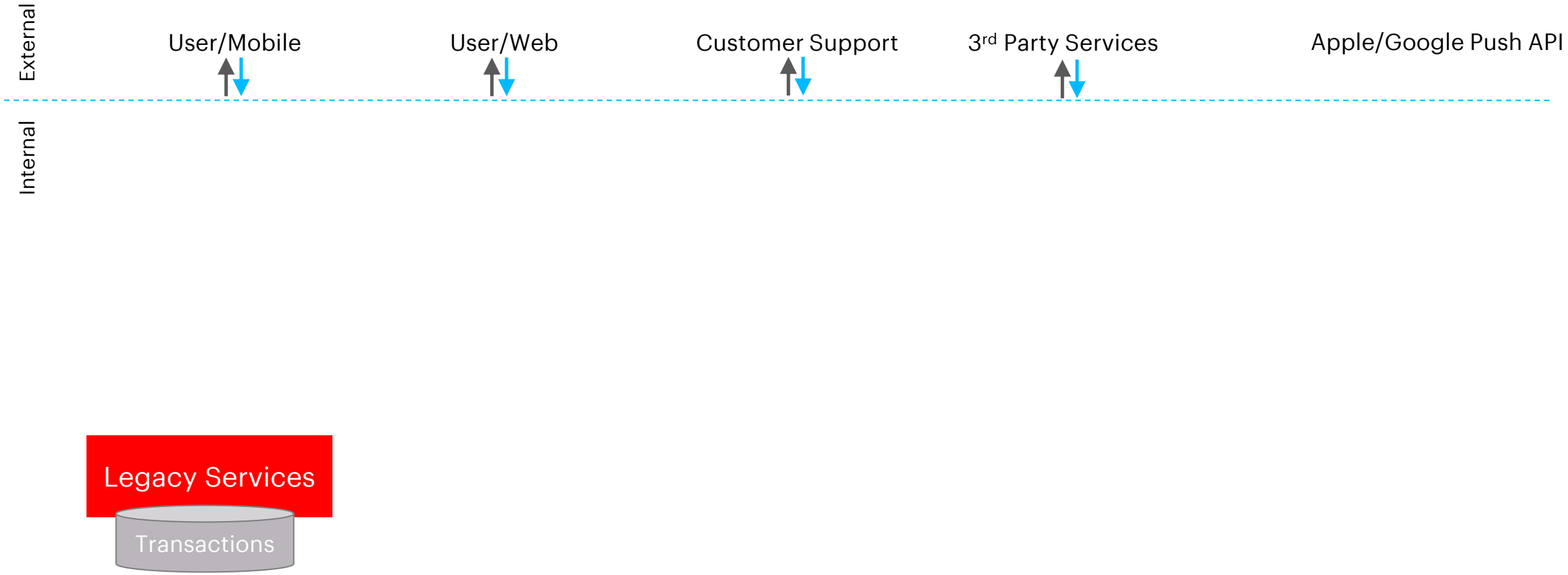
EVOLVE TOWARDS EXPONENTIAL IT THROUGH DIGITAL DECOUPLING AND MODERNIZATION



SOMETHING ELSE VS. DECOUPLING

Vision and aim of a perfect architecture, the „big shot“	Always evolving architecture allowing to pivot anytime
Strict 3/4/5 tiers á la Data | Integration | Processes | UI | …	Purposeful fragmentation with high independence of systems
Central integration solutions, e.g. huge ESB | Workflow Engines | …	Each part follows simple, open integration standards
Consolidation: One canonic data model, one data owner, one *	Pragmatic data duplication when needed with simple principles

ARCHITECTURE EXAMPLE



ARCHITECTURE STRATEGIES TO DIGITALLY DECOUPLE

#1: Wrap the Monolith(s)

#2: Make data accessible

#3: Real-time interactivity

#4: Be cloud native

#5: Be relevant

#1 WRAP THE MONOLITH
LOVE IT | CHANGE IT | KILL IT

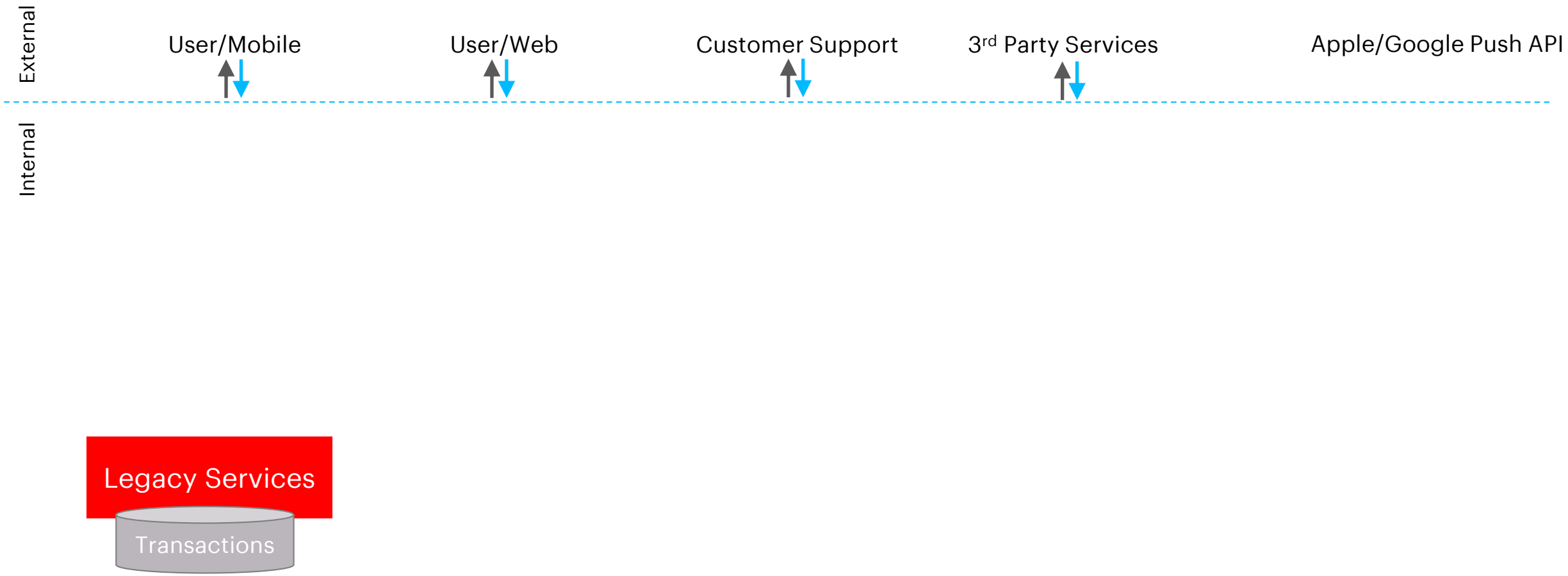
What?

- API Wrapper around Monolith, making it a “normal” component
- Encapsulate with API’s & clear contracts
- Standard JSON/XML, but not a beauty contest
- Products (integration, RPA, …) help a lot

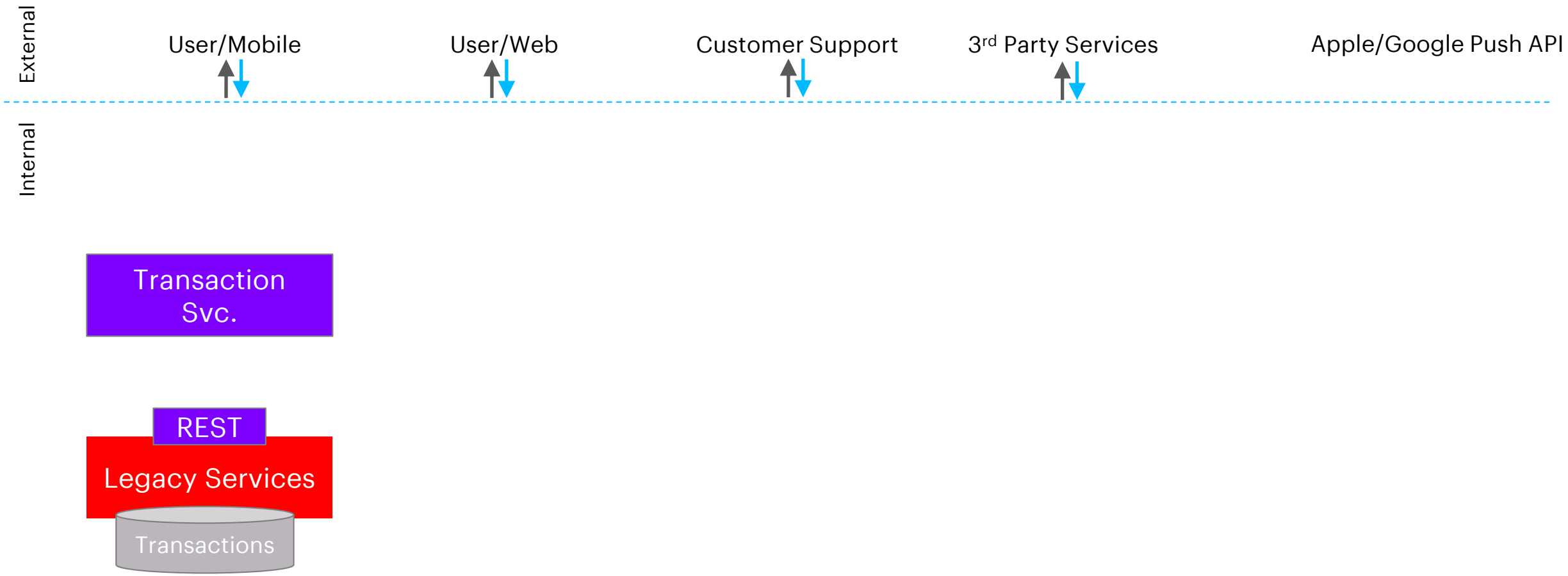
Why?

- Regain control & visibility
- Opens for a range of future tactics: slicing | right-sizing | replacing | migrating

#1 WRAP THE MONOLITH
LOVE IT | CHANGE IT | KILL IT



#1 WRAP THE MONOLITH
LOVE IT | CHANGE IT | KILL IT



#2 MAKE DATA ACCESSIBLE
THE BLOOD IN TODAYS BUSINESS

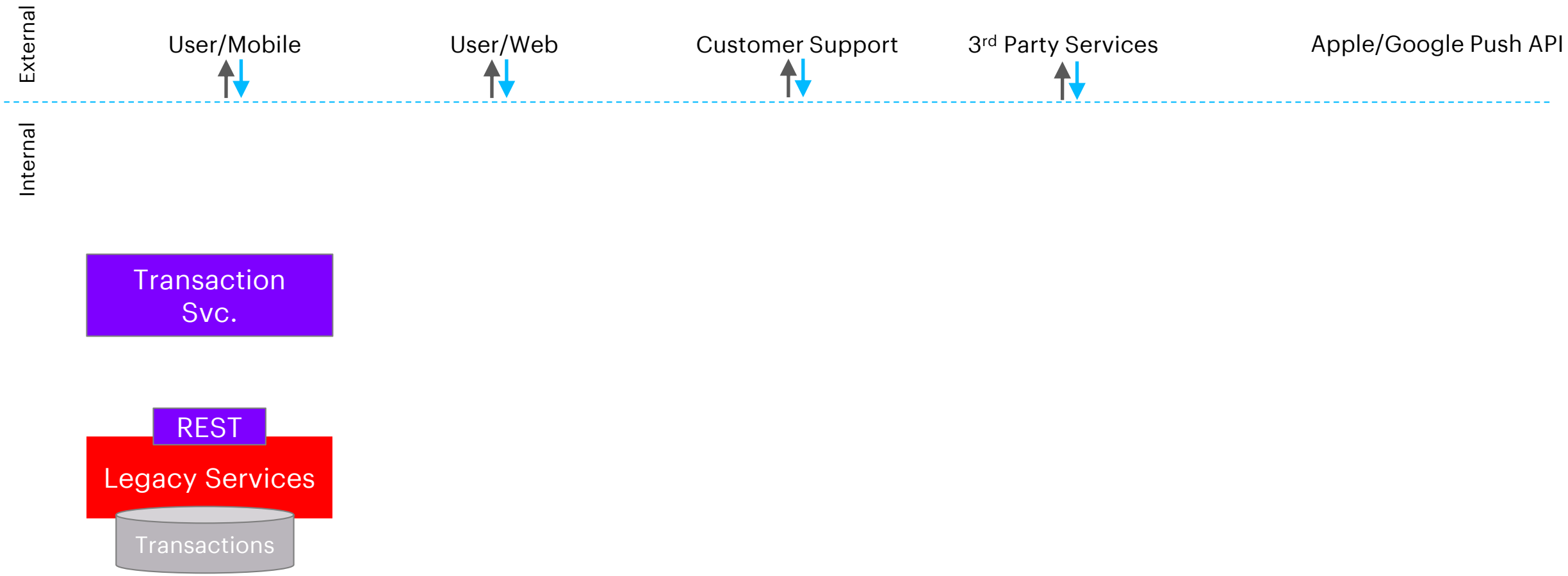
What?

- Build a *working real-time* ODS (often an operational data lake)
- Get all data there, use case by use case
- Get original data formats, transform within lake

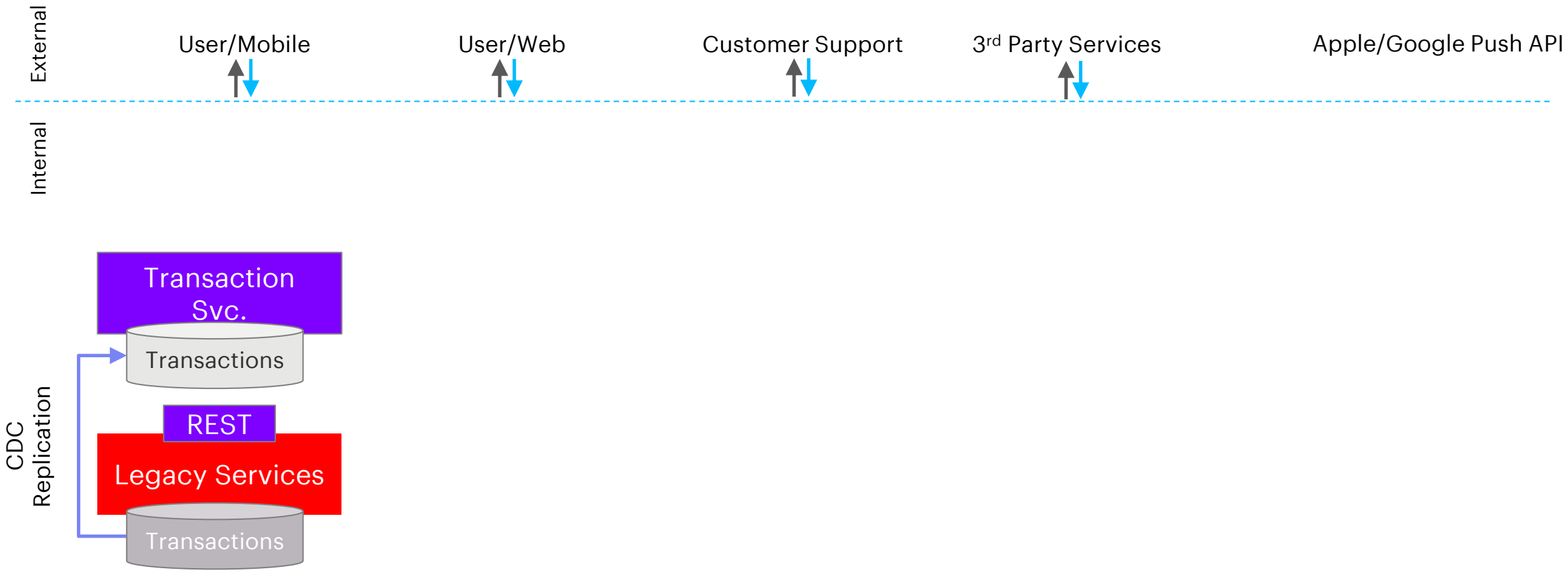
Why?

- Quick & cheap (read) access to data, easy to add services
- Analytics and AI-driven use cases need data. Lots of up-to-date data.

#2 MAKE DATA ACCESSIBLE
THE BLOOD IN TODAYS BUSINESS



#2 MAKE DATA ACCESSIBLE
THE BLOOD IN TODAYS BUSINESS



#3 REAL-TIME INTERACTIVITY
THE BACKBONE OF CUSTOMER EXPERIENCE

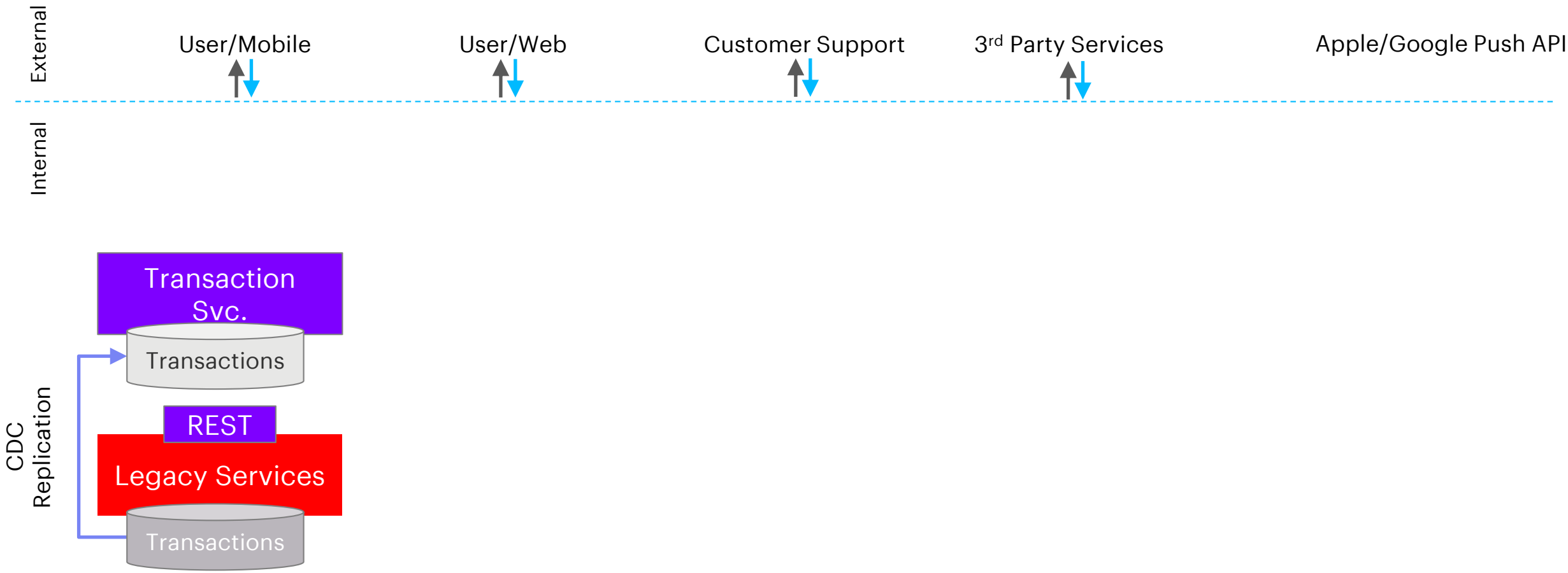
What?

- Establish a *scalable* and *agile* event processing capability
- Be able to plug into a data feed to generate new representations and ACT!

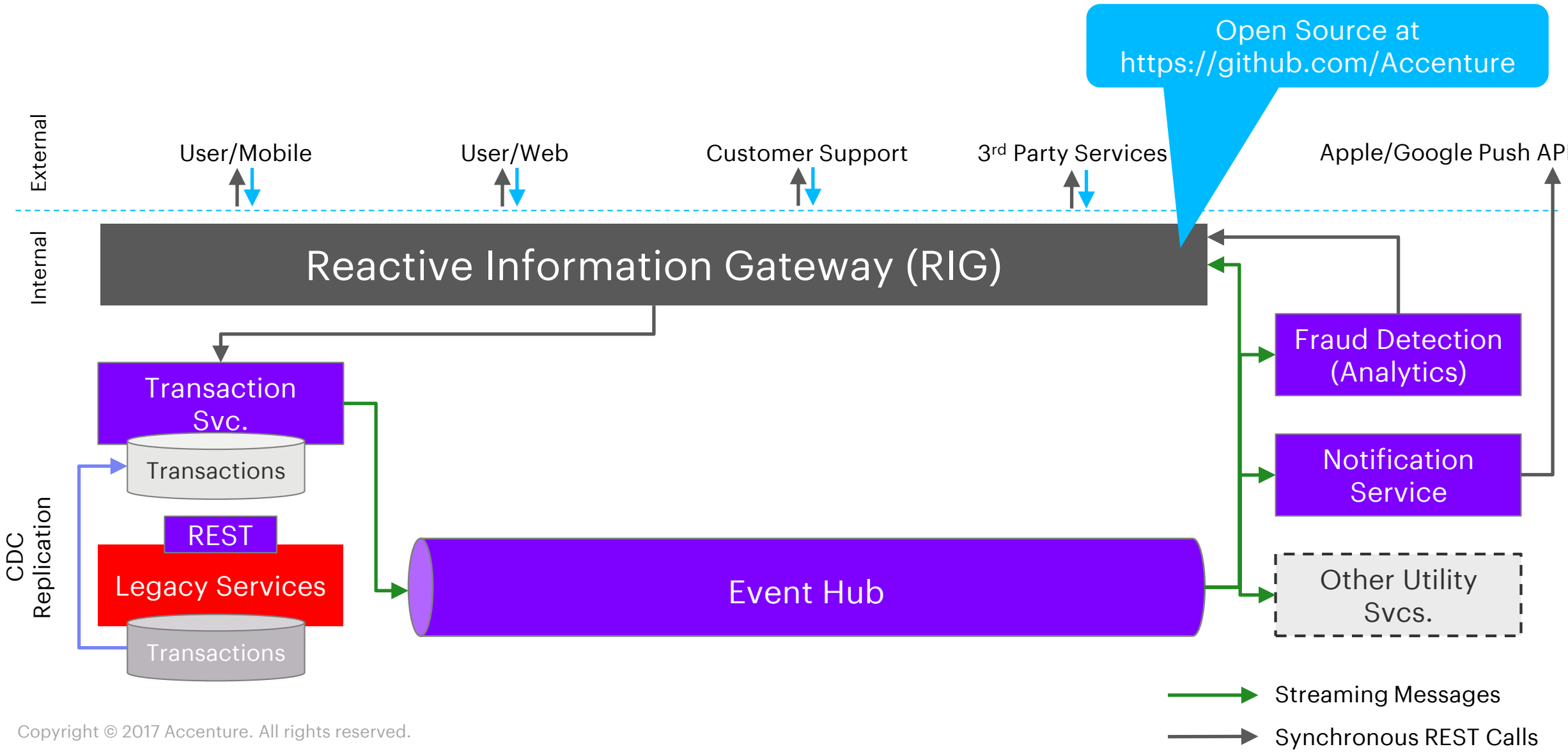
Why?

- Drive out AI in real-time; next-best-action, smart pro-active service,…
- Create real-time mashups of data, new representations (e.g. full-time search), drive automation

#3 REAL-TIME INTERACTIVITY
THE BACKBONE OF CUSTOMER EXPERIENCE



#3 REAL-TIME INTERACTIVITY
THE BACKBONE OF CUSTOMER EXPERIENCE



#4 BE CLOUD NATIVE
ITS ALL ABOUT SPEED TO MARKET

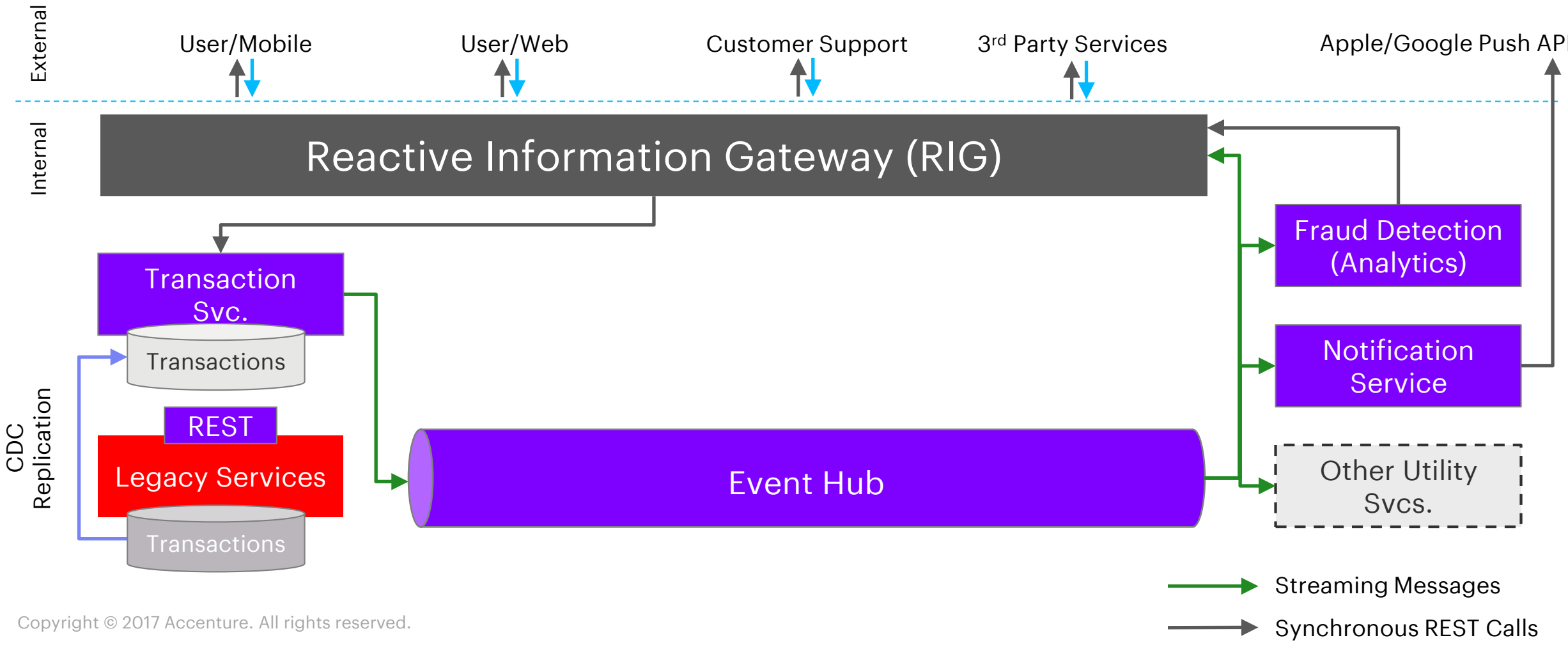
What?

- Embrace Microservices with clear contracts (e.g. OpenAPI) and simple powerful rules
- Embed in state-of-the art DevOps procedures, heavily leverage containers

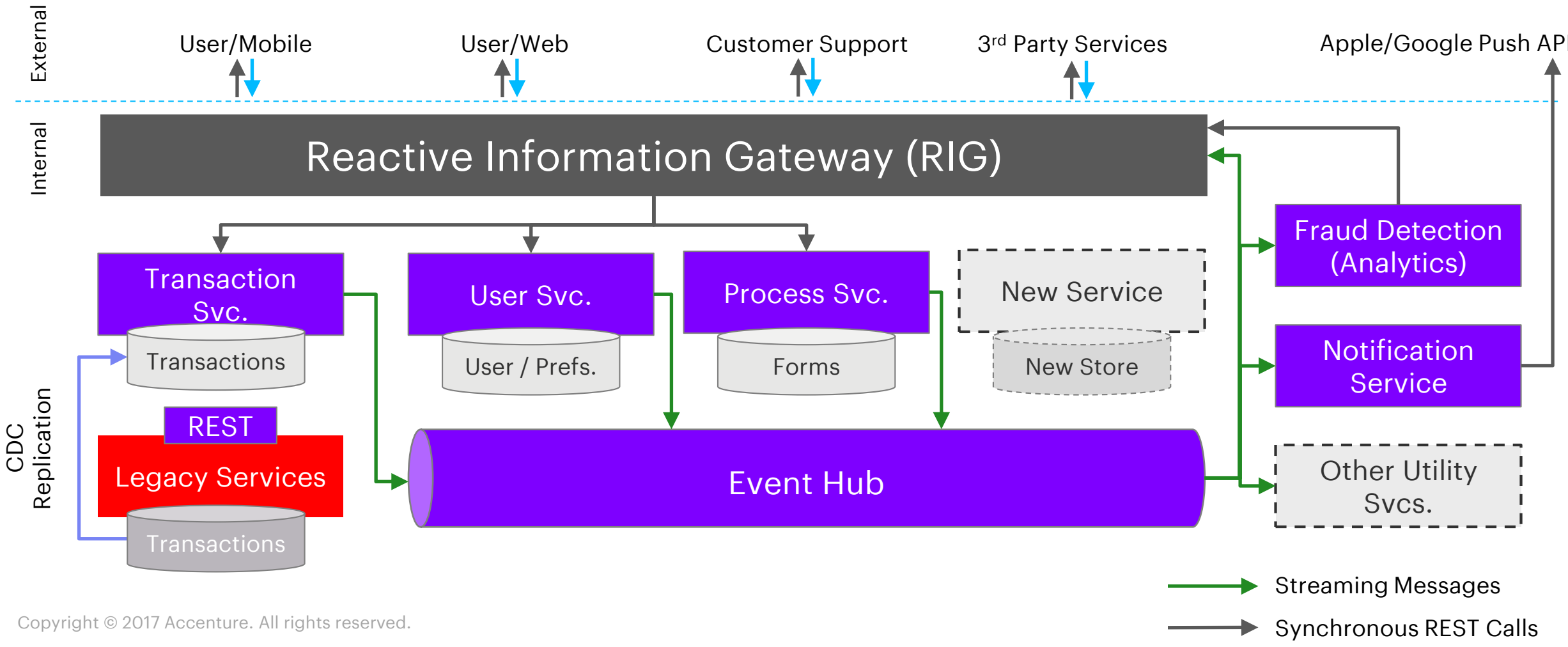
Why?

- Time to market. Teams don’t step on each other toes and enjoy controlled freedom
- Risk-control. A microservice is always replaceable, change-able, upgradeable

#4 BE CLOUD NATIVE
ITS ALL ABOUT SPEED TO MARKET



#4 BE CLOUD NATIVE
ITS ALL ABOUT SPEED TO MARKET



#5 BE RELEVANT
YOU GOT THE LOOKS

What?

- Focus on tools that get you forward quickly*
- Embrace the latest UI/UX
- Embrace JavaScript
- Follow the cool-kids (Facebook & co); this is a fast moving field, not architecture perfection

Why?

- Market expectation. Users don't read manuals and want their stuff on all gadgets
- Taking risks is Ok, lifetime is 2-3 years, key principles and security are ensured on API level

* Examples: Portals are often a massive slowdown; React has a nicer learning curve than Angular,...


NEW
APPLIED
NOW