ANDREESSEN HOROWITZ

TABLE OF CONTENTS

Emerging Architectures for Modern Data Infrastructure

Matt Bornstein, Jennifer Li, and Martin Casado

SHARE

Posted October 15, 2020

*This is an updated version of a post we originally published in 2020. You can read the original version **here**.*

The growth of the data infrastructure industry has continued unabated since we published a set of reference architectures in late 2020. Nearly all key industry metrics hit record highs during the past year, and new product categories appeared faster than most data teams could reasonably keep track. Even the benchmark wars and billboard battles returned.

To help data teams stay on top of the changes happening in the industry, we're publishing in this post an updated set of data infrastructure architectures. They show the current best-in-class stack across both analytic and operational systems, as gathered from numerous operators we spoke with over the last year. Each architectural blueprint includes a summary of what's changed since the prior version.

We'll also attempt to explain *why* these changes are taking place. We argue that core data processing systems have remained relatively stable over the past year, while supporting tools and applications have proliferated rapidly. We explore the hypothesis that *platforms* are

beginning to emerge in the data ecosystem, and that this helps explain the particular patterns we're seeing in the evolution of the data stack.

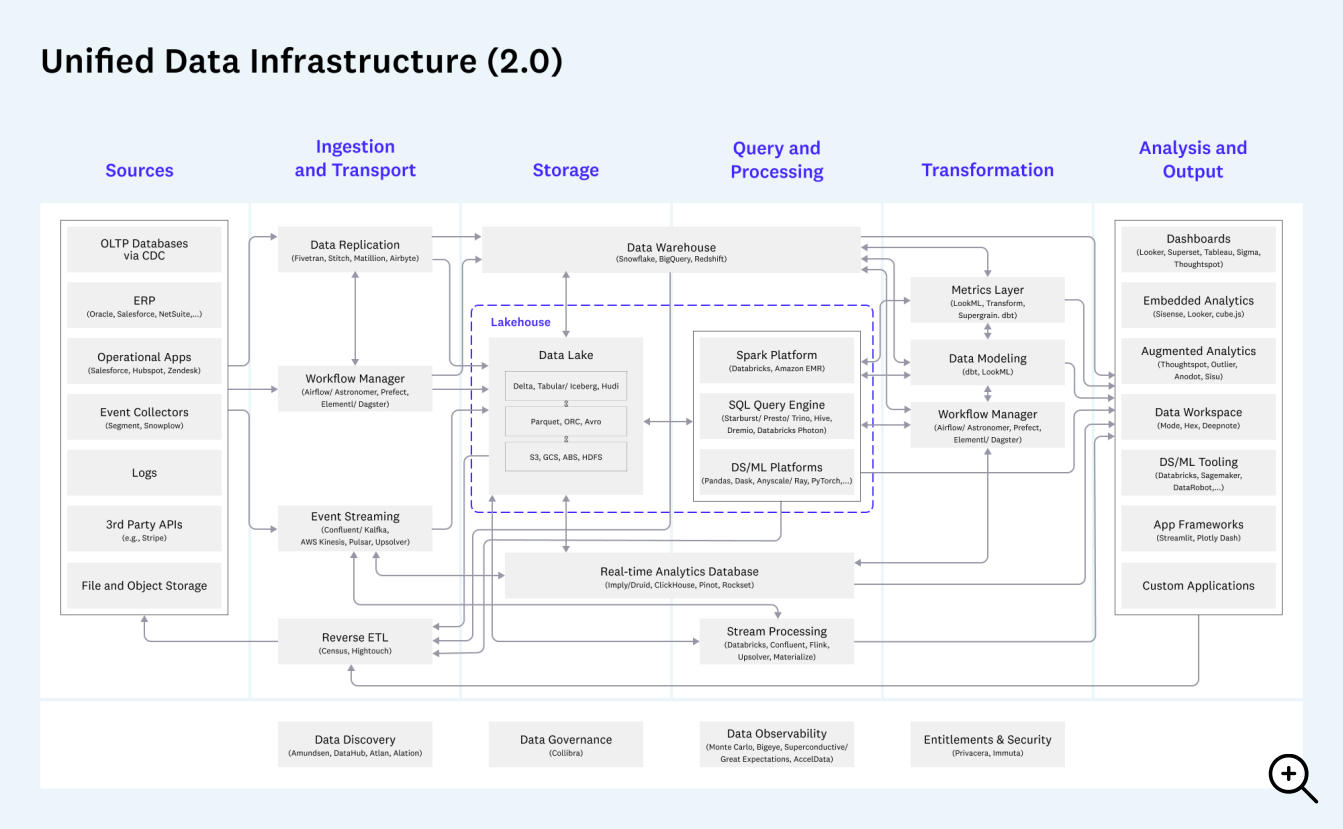
TABLE OF CONTENTS

To compile this work, we relied again on input from dozens of data experts, who are listed at the end of this post. This simply wouldn't exist without them, so thank you!

Updated reference architectures

Before we get too deep in the details, here are the latest architecture diagrams. These were compiled with the help of leading data practitioners, based on what they run internally and what they recommend for new deployments.

The first view shows a unified overview across all data infrastructure use cases:



Notes: Excludes OLTP, log analysis, and SaaS analytics apps.

Unified Data Infrastructure (2.0): Definitions

TABLE OF CONTENTS

Sources	Ingestion and Transport	Storage	Query and Processing	Transformation	Analysis and Output
Generate relevant business and operational data	Extract data from operational systems (E) Deliver to storage, aligning schemas between source and destination (L) Transport analyzed data back to operational systems as needed	Store data in a format accessible to query & processing systems Optimize for consistency, performance, cost, and scale	Translate high-level code (usually written in SQL, Python, or Java/ Scala) into low-level data processing jobs Execute queries and data models against stored data, often using distributed compute Includes both historical analysis - describing what happened - and predictive analysis - describing expectations for the future	Transform data into a structure ready for analysis (T) Orchestrate processing resources for this purpose	Provide an interface for analysts and data scientists to derive insights and collaborate Present results of analysis to internal and external users Embed data models into user-facing applications

The second view zooms in on machine learning, which is a complex and increasingly independent tool chain:

Machine Learning Infrastructure (2.0)

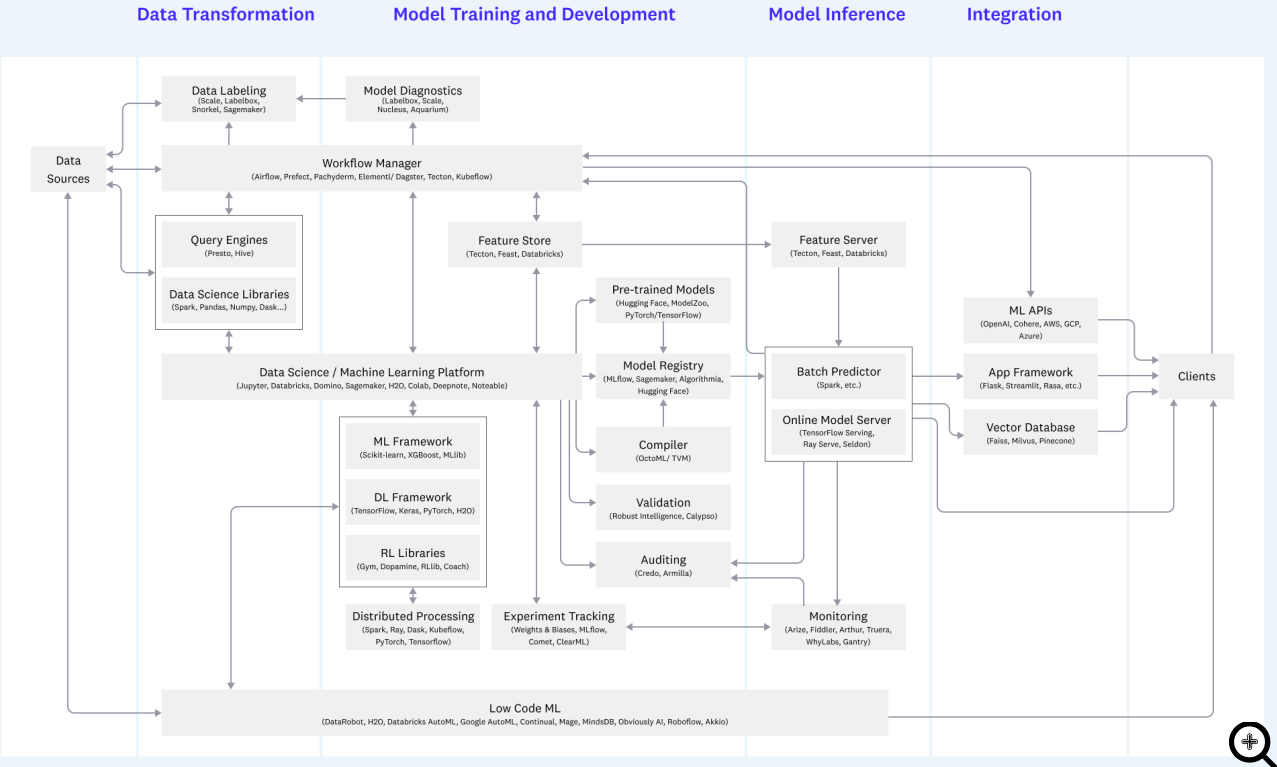


TABLE OF CONTENTS

Machine Learning Infrastructure (2.0): Definitions

Data Transformation	Model Training and Development	Model Inference	Integration
Convert raw data into a form ready for model training, including annotation for supervised learning	Train models against processed data - often building on top of a model pre-trained on a corpus of public data Track the experimentation and model training process, including input data, hyperparemeters used and resulting model performance Analyze, validate, and audit model performance as part of an iterative loop, often leading to retraining and/or additional data collection and processing Prepare trained models for deployment by compiling to the relevant hardware targets and storing for access at the inference stage	Execute trained models against input data, either in real time (online) or in batches (offline) Monitor production models for data drift, harmful predictions, performance drops, etc.	Integrate model outputs into user-facing applications in a structured and repeatable way

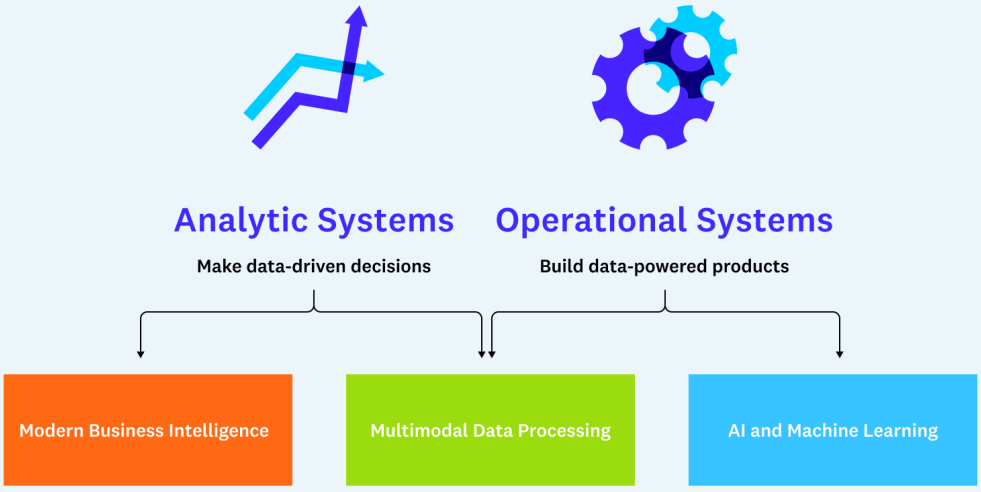
In the rest of this post, we'll comment on what's changed since v1 of the data stack and explore the underlying root causes.

Changelog

What hasn't changed: Stability in the core

Despite the frenzy of data infrastructure activity over the past year, it's surprising to see — in some ways — how little has changed.

In our first post, we drew a distinction between *analytic* systems that support data-driven decision-making and *operational* systems that power data-driven products. We then mapped these categories to three patterns, or blueprints, often implemented by leading data teams.



One of the key questions was whether these architectural patterns would converge. A year later, that doesn't seem to have taken place.

In particular, the analytic and operational ecosystems both continue to thrive. Cloud data warehouses like Snowflake have grown rapidly, focused largely on SQL users and business intelligence use cases. But adoption of other technologies has also accelerated — data lakehouses like Databricks, for instance, are adding customers faster than ever. Many data teams we spoke with confirmed that heterogeneity is likely here to stay in the data stack.

Other core data systems — namely, ingestion and transformation — have proven similarly durable. This is especially visible in the modern business intelligence pattern, where the combination of Fivetran and dbt (or similar technologies) has become nearly ubiquitous. But it's also true to an extent in operational systems, where *de facto* standards like Databricks/Spark, Confluent/Kafka, and Astronomer/Airflow have emerged.

What's new: Cambrian explosion

Around the stable core, the data stack has evolved rapidly over the past year. Broadly speaking, we've seen the most activity in two areas:

New *tools* designed to support key data processes and workflows, like data discovery, observability, or ML model auditing

New *applications* that allow data teams and business users to generate value from data in new, more powerful ways, like data workspaces, reverse ETL, and ML application frameworks

We're also seeing the introduction of some new technologies designed to enhance core data-processing systems. Notably, there has been active debate around the metrics layer in the analytical ecosystem and the lakehouse pattern for operational systems — both of which are converging toward useful definitions and architectures.

Updated blueprints

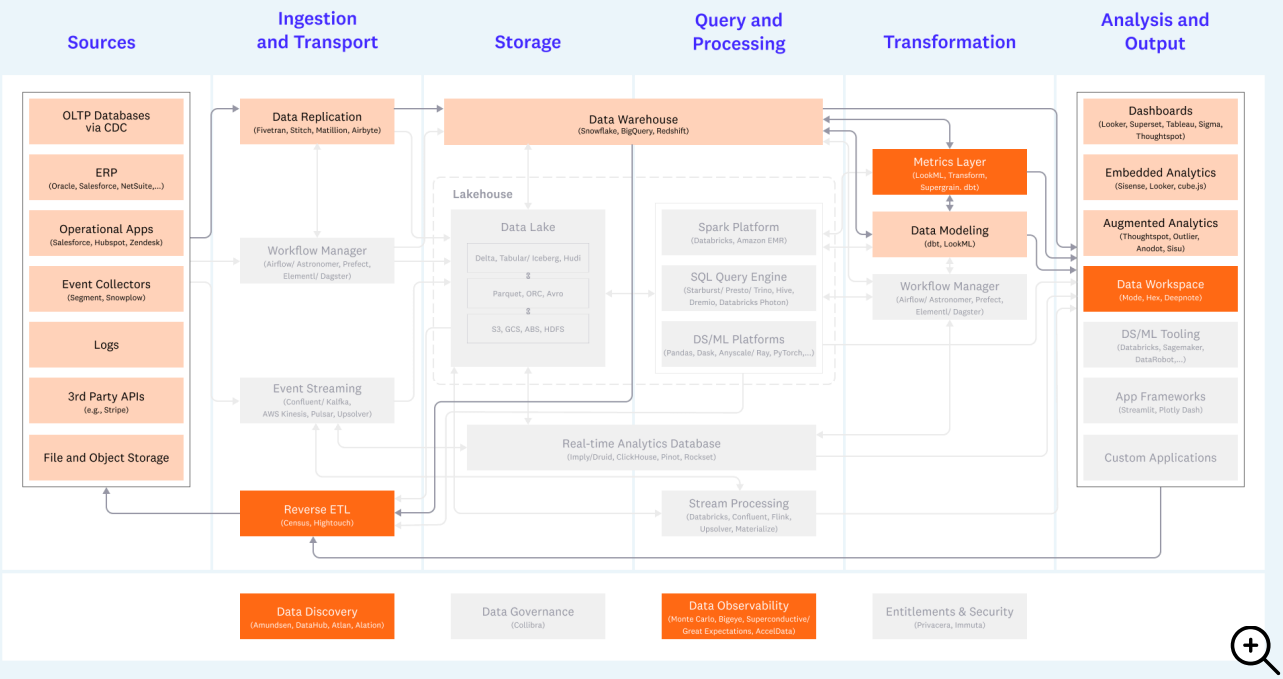
With that context, we'll go into detail on each of the major data infrastructure blueprints. Each section below shows an updated diagram (diff'd against v1 of the stack) and an analysis of key changes. These sections are intended primarily as reference for data teams implementing these stacks, and reading them isn't necessary to follow the rest of the post.

Blueprint 1: Modern Business Intelligence

Cloud-native business intelligence for companies of all sizes

TABLE OF CONTENTS

Blueprint 1: Modern Business Intelligence



Darker boxes are new or meaningfully changed since v1 of the architecture in 2020; lighter colored boxes have remained largely the same. Gray boxes are considered less relevant to this blueprint.

What hasn't changed:

The combination of data replication (like Fivetran), cloud data warehouses (like Snowflake), and SQL-based data modeling (with dbt) continues to form the core of this pattern. Adoption for these technologies has grown meaningfully, prompting the funding and early growth of new competitors (e.g. Airbyte and Firebolt).

Dashboards continue to be the most common application used in the output layer, including Looker, Tableau, PowerBI, and newer entrants like Superset.

What's new:

There has been a surge of interest in the **metrics layer**, a system providing a standard set of definitions on top of the data warehouse. This has been hotly debated, including what capabilities it should have, which vendor(s) should own it, and what spec it should follow. So far, we've seen several credible pure-play products (like Transform and Supergrain), plus expansion into this category by dbt.

Reverse ETL vendors have grown meaningfully, particularly Hightouch and Census. The purpose of these products is to update operational systems, like CRM or ERP, with outputs and insights derived from the data warehouse.

Data teams are showing stronger interest in **new applications** to augment their standard dashboards, especially data workspaces (like Hex). Broadly speaking, new apps are likely

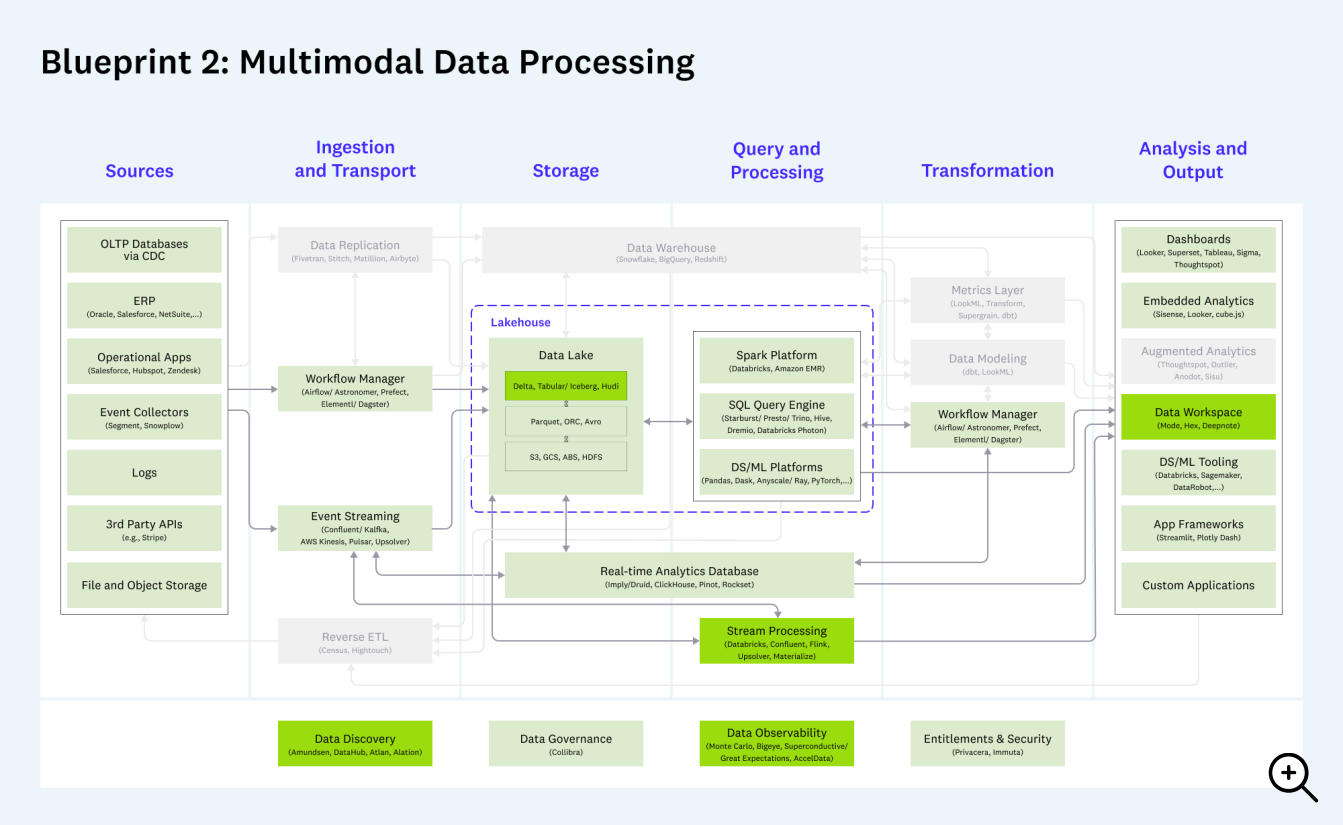
the result of increasing standardization in cloud data warehouses — once data is cleanly structured and easy to access, data teams naturally want to do more with it.

TABLE OF CONTENTS

Data discovery and observability companies have proliferated and raised substantial amounts of capital (especially Monte Carlo and Bigeye). While the benefits of these products are clear — i.e. more reliable data pipelines and better collaboration — adoption is still relatively early, as customers discover relevant use cases and budgets. (Technical note: although there are several credible new vendors in data discovery — e.g. Select Star, Metaphor, Stemma, Secoda, Castor — we have excluded seed-stage companies from the diagram in general.)

Blueprint 2: Multimodal Data Processing

Evolved data lakes supporting both analytic and operational use cases – also known as modern infrastructure for Hadoop refugees



Note: Darker boxes are new or meaningfully changed since v1 of the architecture in 2020; lighter colored boxes have remained largely the same. Gray boxes are considered less relevant to this blueprint.

What hasn't changed:

Core systems in data processing (e.g. Databricks, Starburst, and Dremio), transport (e.g. Confluent and Airflow), and storage (AWS) continue to grow rapidly and form the backbone of this blueprint.

TABLE OF CONTENTS

Multimodal data processing remains diverse by design, allowing companies to adopt the systems best suited to their particular needs across both analytics and operational data applications.

What's new:

There is growing recognition and clarity for the **lakehouse** architecture. We've seen this approach supported by a wide range of vendors (including AWS, Databricks, Google Cloud, Starburst, and Dremio) and data warehouse pioneers. The fundamental value of the lakehouse is to pair a robust storage layer with an array of powerful data processing engines like Spark, Presto, Druid/Clickhouse, Python libraries, etc.

The **storage layer** itself is getting an upgrade. While technologies like Delta, Iceberg, and Hudi are not new, they are seeing accelerated adoption and are being built into commercial products. Some of these technologies (particularly Iceberg) also interoperate with cloud data warehouses like Snowflake. If heterogeneity is here to stay, this is likely to become a key part of the multimodal data stack.

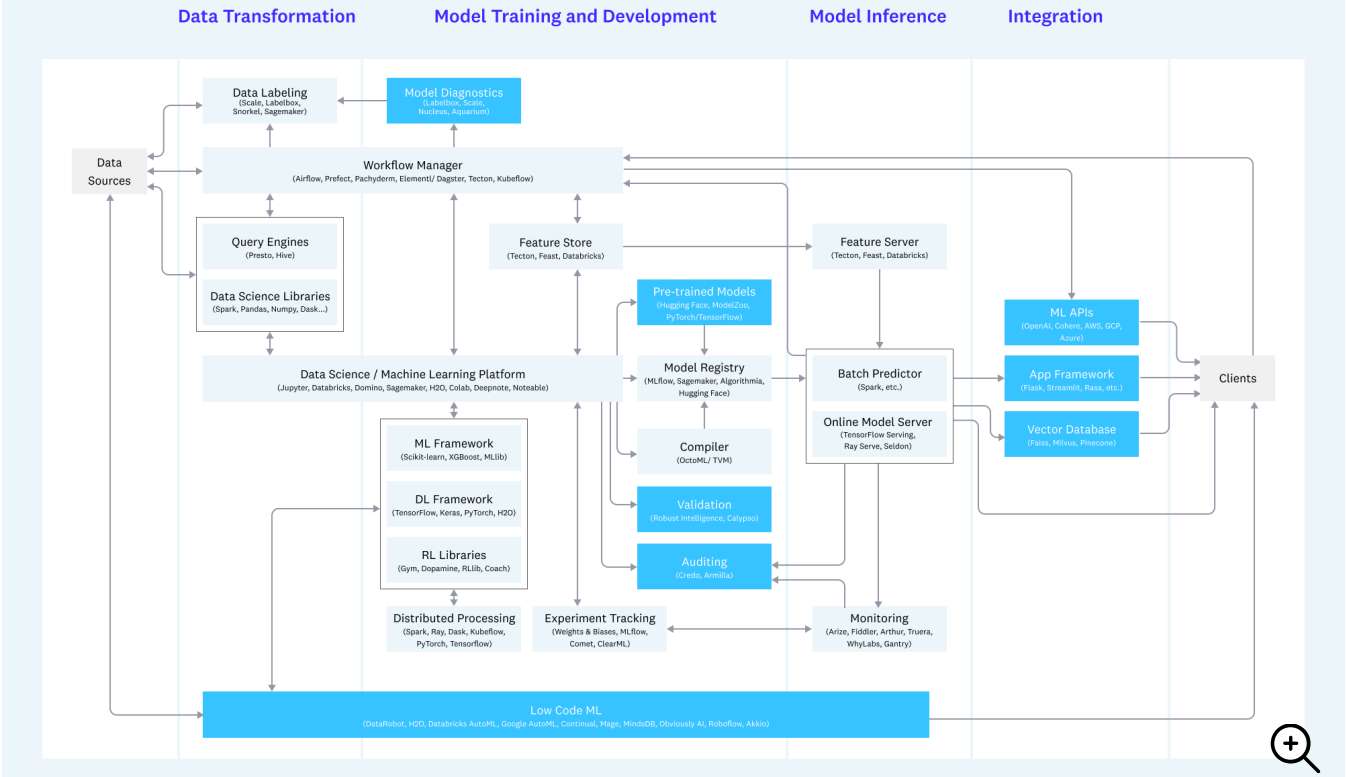
There may be an uptick in adoption taking place for **stream processing** (i.e., real-time analytical data processing). While first-generation technologies like Flink still haven't gone mainstream, new entrants with simpler programming models (like Materialize and Upsolver) are gaining early adoption, and, anecdotally, usage of stream processing products from incumbents Databricks and Confluent has also started to accelerate.

Blueprint 3: Artificial Intelligence and Machine Learning

Stack for robust development, testing, and operation of machine learning models

TABLE OF CONTENTS

Blueprint 3: Artificial Intelligence and Machine Learning



Note: Darker boxes are new or meaningfully changed since v1 of the architecture in 2020; lighter colored boxes have remained largely the same. Gray boxes are considered less relevant to this blueprint.

What hasn't changed:

Tooling for model development is largely similar today compared to 2020, including the major cloud vendors (e.g. Databricks and AWS), ML frameworks (e.g. XGBoost and PyTorch), and experiment management tools (e.g. Weights & Biases and Comet)

Experiment management has effectively subsumed model visualization and tuning as independent categories.

Building and operating a machine learning stack is complicated and requires specialized expertise. This blueprint is not for the faint of heart — and productionizing AI is still challenging for many data teams.

What's new:

The ML industry is consolidating around a **data-centric approach**, emphasizing sophisticated data management over incremental modeling improvements. This has several implications:

Rapid growth for **data labeling** (e.g. Scale and Labelbox) and growing interest in **closed-loop data engines,** largely modeled on Tesla's Autopilot data pipelines.

TABLE OF CONTENTS

Increased adoption for **feature stores** (e.g. Tecton), for both batch and real-time use cases, as a means to develop production-grade ML data in a collaborative way.

Revived interest in **low-code ML** solutions (like Continual and MindsDB) that at least partially automate the ML modeling process. These newer solutions focus on bringing new users (i.e. analysts and software developers) into the ML market.

Use of **pre-trained models** is becoming the default, especially in NLP, and providing tailwinds to companies like OpenAI and Hugging Face. There are still meaningful problems to solve here around fine-tuning, cost, and scaling.

Operations tools for ML (sometimes called MLops) are becoming more mature, built around **ML monitoring** as the most in-demand use case and immediate budget. Meanwhile, a raft of new operational tools — including, notably, **validation** and **auditing** — are appearing, with the ultimate market still to be determined.

There is increased focus on how developers can seamlessly integrate ML models into applications, including through **pre-built APIs** (e.g. OpenAI), **vector databases** (e.g. Pinecone), and more opinionated frameworks.

The data platform hypothesis

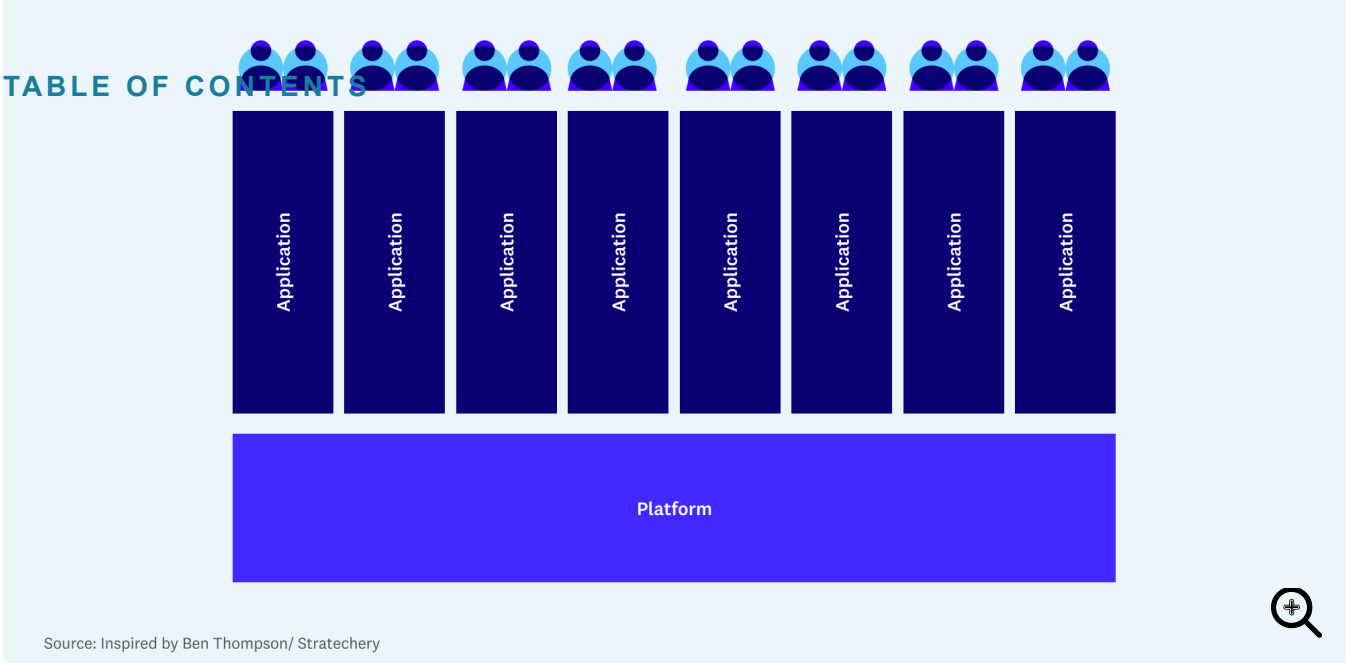
To recap: Over the past year, the data infrastructure stack has seen substantial stability in core systems and rapid proliferation of supporting tools and applications.To help explain why this might be happening, we introduce here the idea of *data platforms*.

What is a platform?

The word “platform” is overloaded in the data ecosystem, often used by internal teams to describe their whole tech stacks or by **vendors** to sell loosely connected product suites.

In software more broadly, a platform is something other developers can build on top of. Platforms generally provide limited value on their own — most users have no interest, for instance, in accessing the guts of Windows or iOS. But they provide an array of benefits, like a common programming interface and a large installed base, that allow developers to build and distribute the applications users ultimately care about.

The defining trait of a platform, from an industry standpoint, is mutual dependence — both technically and economically — between an influential platform provider and a large pool of 3rd-party developers.



What is a *data* platform?

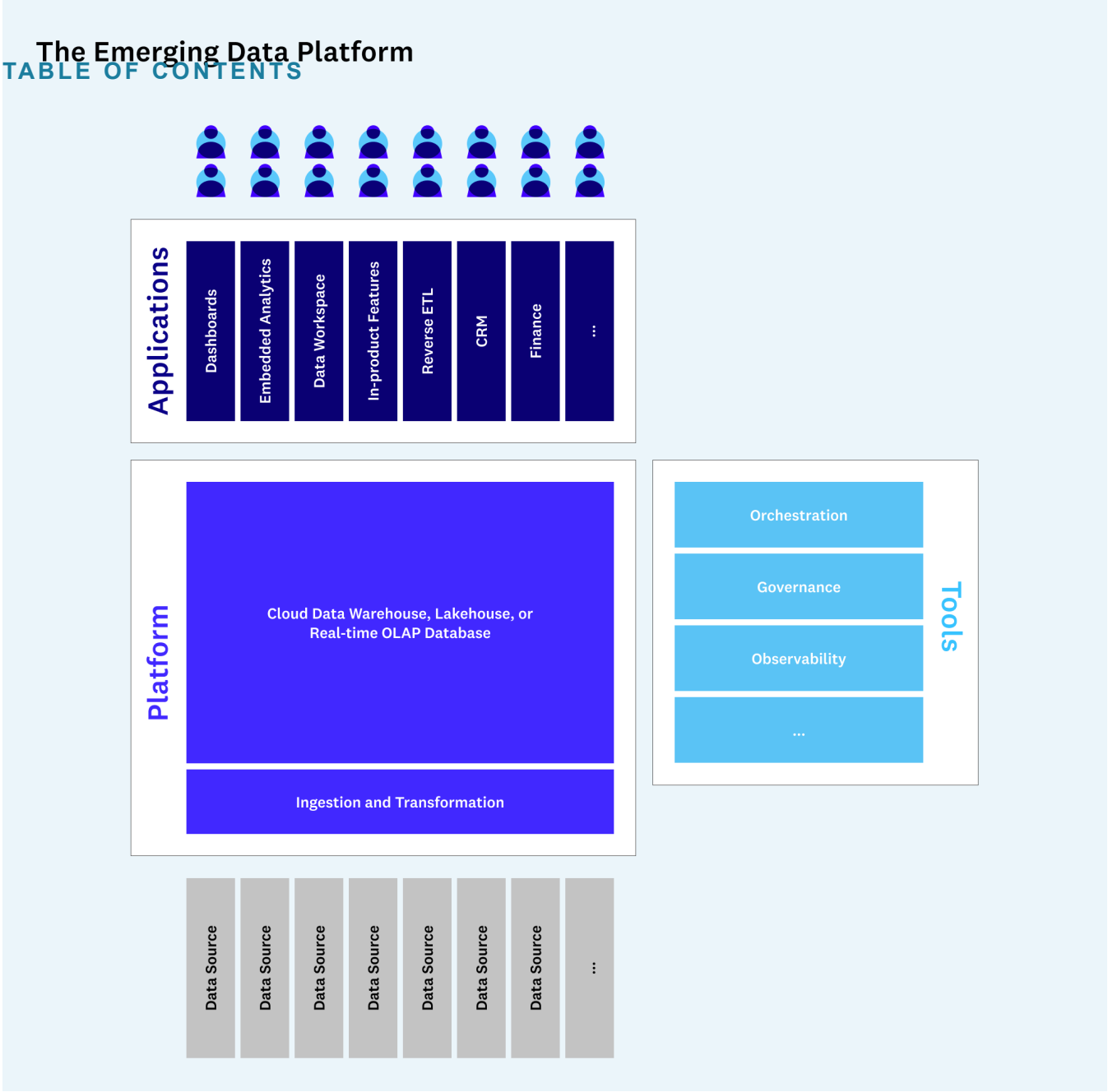
Historically, the data stack has not been an obvious fit for the definition of a platform. Mutual dependence existed — among ETL, data warehouse, and reporting vendors, for instance — but the integration model tended to be one-to-one, rather than one-to-many, and was supplemented heavily by professional services.

According to a number of data experts we spoke with, this may be starting to change.

The platform hypothesis argues that the “backend” of the data stack — roughly defined as data ingestion, storage, processing, and transformation — has started to consolidate around a relatively small set of cloud-based vendors. As a result, customer data is being collected in a standard set of systems, and vendors are investing heavily to make this data easily accessible to other developers — as a fundamental design principle in systems like Databricks, and via SQL standards plus **custom compute APIs** in systems like Snowflake.

“Frontend” developers, in turn, have taken advantage of this single point of integration to build out a range of new applications. They rely on clean, joined data in the data warehouse/lakehouse, without worrying about the underlying details of how it got there. A single customer may buy and build many applications on top of one core data system. We’re even starting to see traditional enterprise systems, like financial or product analytics, being rebuilt with a “warehouse-native” architecture.

The picture might look like this:



To be clear, this doesn't mean that OLTP databases or other important backend technologies will disappear in the near future. But native integration with OLAP systems may become a critical component of application development. And over time, more and more business logic and application functionality could transition to this model. We may see a whole class of new products built on this data platform.

The emergence of data apps?

The data platform hypothesis is still very much open to debate. However, we *are* seeing an increase in sophisticated vertical SaaS solutions implemented as horizontal layers on top of the data platforms. And so, while early, we'd argue that the changes taking place in the data stack are at least *consistent* with the idea that platforms are taking hold.

TABLE OF CONTENTS

There are many reasons, for example, that companies like Snowflake and Databricks have become stable pieces of the data stack, including great products, capable sales teams, and low-friction deployment models. But there's also a case to be made that their stickiness is reinforced by platform dynamics — once a customer has built and/or integrated a range of data applications with one of these systems, it often doesn't make sense to transition off.

A similar argument can be made for the surge of new data infrastructure products in recent years. The typical explanations for this trend have to do with vast troves of data, increasing corporate budgets, and a glut of VC funding. But those things have arguably been true for decades. The reason we're seeing so many new products appear now may have to do with platforms — namely, that it's never been easier to get a new data application adopted, and it's never been more important to properly maintain the platform.

Finally, the platform hypothesis provides some predictive power in terms of competitive dynamics. At scale, platforms can be extremely valuable. Core data systems vendors may be competing aggressively today not just for current budgets, but for a long-term platform position. Eye-popping valuations for data ingestion and transformation companies — or especially heated debates over new categories like the metrics layer or reverse ETL — also make more sense if you believe they are a core part of the emerging data platform.

Looking ahead

We're still in the early stages of defining the analytical and operational data platform, and the pieces of the platform are in flux. As such, it's probably more useful as an analogy than as a strict definition. But it may be a useful tool to filter signal from noise, and to help develop a sense of why the market is moving the way it is. Data teams now have more tools, resources, and organizational momentum behind them than at any point (likely) since the invention of the database. And we're very excited to see how the app layer evolves on top of the emerging platforms.

List of contributors to Emerging Data Architectures (all versions): Peter Bailis, Mike del Balso, Max Beauchemin, Scott Clark, Jamie Davidson, George Fraser, Krishna Gade, Ali Ghodsi, Abe Gong, Nick Handel, Tristan Handy, Shinji Kim, Mars Lan, Xiangrui Meng, Clemens Mewald, Bob Muglia, Jad Naous, Robert Nishihara, Diego Oppenheimer, Amit Prakash, Ori Rafael, Praveen Rangnath, Nick Schrock, Benn Stancil, Carl Steinbach, Ion Stoica, Kevin Stumpf, Arsalan Tavakoli, Venkat Venkataramani, Don Vu, Reynold Xin, FJ Yang, Matei Zaharia.

team

TABLE OF CONTENTS

Sign up for our a16z newsletter to get analysis and news covering the latest trends reshaping AI and infrastructure.

a16z Infra

My personal Substack

Type your email... Subscribe

substack

By subscribing you agree to Substack's Terms of Use, our

SEE ALL NEWSLETTERS

Matt Bornstein is a partner at Andreessen Horowitz focused on AI, data systems, and infrastructure.

FOLLOW

X

Linkedin

Jennifer Li is a General Partner at Andreessen Horowitz, where she focuses on enterprise and infrastructure investments in data systems, developer tools, and AI.

FOLLOW

X

Linkedin

Martin Casado is a general partner at Andreessen Horowitz, where he leads the firm's $1.25 billion infrastructure practice.

FOLLOW

X

Linkedin

MORE FROM THESE CONTRIBUTORS

Investing in Reducto

Jennifer Li and Yoko Li

Investing in Relace
TABLE OF CONTENTS
Yoko Li, Guido Appenzeller, and Martin Casado

America Cannot Lose the Robotics Race

Martin Casado and Anne Neuberger

Investing in Phota Labs

Yoko Li, Martin Casado, and Jennifer Li

The Rise of Computer Use and Agentic Coworkers

Eric Zhou, Yoko Li, Seema Amble, and Jennifer Li

are not representative of all investments in vehicles managed by a16z and there can be no assurance that the investments will be profitable or that other investments made in the future will have similar characteristics or results. A list of investments made by funds managed by a16z is available here: https://a16z.com/investments/. Past results of a16z's investments, pooled investment vehicles, or investment strategies are not necessarily indicative of future results. Excluded from this list are investments (and certain publicly traded cryptocurrencies/ digital assets) for which the issuer has not provided permission for a16z to disclose publicly. As for its investments in any cryptocurrency or token project, a16z is acting in its own financial interest, not necessarily in the interests of other token holders. a16z has no special role in any of these projects or power over their management. a16z does not undertake to continue to have any involvement in these projects other than as an investor and token holder, and other token holders should not expect that it will or rely on it to have any particular involvement.

TABLE OF CONTENTS

With respect to funds managed by a16z that are registered in Japan, a16z will provide to any member of the Japanese public a copy of such documents as are required to be made publicly available pursuant to Article 63 of the Financial Instruments and Exchange Act of Japan. Please contact compliance@a16z.com to request such documents.

For other site terms of use, please go here. Additional important information about a16z, including our Form ADV Part 2A Brochure, is available at the SEC's website: http://www.adviserinfo.sec.gov.