Chapter 11. The Future of Data Engineering

This book grew out of the authors' recognition that warp speed changes in the field have created a significant knowledge gap for existing data engineers, people interested in moving into a career in data engineering, technology managers, and executives who want to better understand how data engineering fits into their companies. When we started thinking about how to organize this book, we got quite a bit of pushback from friends who'd ask, "How dare you write about a field that is changing so quickly?!" In many ways, they're right. It certainly feels like the field of data engineering—and, really, all things data—is changing daily. Sifting through the noise and finding the signal of *what's unlikely to change* was among the most challenging parts of organizing and writing this book.

In this book, we focus on big ideas that we feel will be useful for the next several years—hence the continuum of the data engineering lifecycle and its undercurrents. The order of operations and names of best practices and technologies might change, but the primary stages of the lifecycle will likely remain intact for many years to come. We're keenly aware that technology continues to change at an exhausting pace; working in the technology sector in our present era can feel like a rollercoaster ride or perhaps a hall of mirrors.

Several years ago, data engineering didn't even exist as a field or job title. Now you're reading a book called *Fundamentals of Data Engineering*! You've learned all about the fundamentals of data engineering—its lifecycle, undercurrents, technologies, and best practices. You might be asking yourself, what's next in data engineering? While nobody can predict the future, we have a good perspective on the past, the present, and current trends. We've been fortunate to watch the genesis and evolution of data engineering from a front-row seat. This final chapter presents our thoughts on the future, including observations of ongoing developments and wild future speculation.

The Data Engineering Lifecycle Isn't Going Away

While data science has received the bulk of the attention in recent years, data engineering is rapidly maturing into a distinct and visible field. It's one of the fastest-growing careers in technology, with no signs of losing momentum. As companies realize they first need to build a data foundation before moving to "sexier" things like AI and ML, data engineering will continue growing in popularity and importance. This progress centers around the data engineering lifecycle.

Some question whether increasingly simple tools and practices will lead to the disappearance of data engineers. This thinking is shallow, lazy, and shortsighted. As organizations leverage data in new ways, new foundations, systems, and workflows will be needed to address these needs. Data engineers sit at the center of designing, architecting, building, and maintaining these systems. If tooling becomes easier to use, data engineers will move up the value chain to focus on higher-level work. The data engineering lifecycle isn't going away anytime soon.

The Decline of Complexity and the Rise of Easy-to-Use Data Tools

Simplified, easy-to-use tools continue to lower the barrier to entry for data engineering. This is a great thing, especially given the shortage of data engineers we've discussed. The trend toward simplicity will continue. Data engineering isn't dependent on a particular technology or data size. It's also not just for large companies. In the 2000s, deploying "big data" technologies required a large team and deep pockets. The ascendance of SaaS-managed services has largely removed the complexity of

understanding the guts of various “big data” systems. Data engineering is now something that *all* companies can do.

Big data is a victim of its extraordinary success. For example, Google BigQuery, a descendant of GFS and MapReduce, can query petabytes of data. Once reserved for internal use at Google, this insanely powerful technology is now available to anybody with a GCP account. Users simply pay for the data they store and query rather than having to build a massive infrastructure stack. Snowflake, Amazon EMR, and many other hyper-scalable cloud data solutions compete in the space and offer similar capabilities.

The cloud is responsible for a significant shift in the usage of open source tools. Even in the early 2010s, using open source typically entailed downloading the code and configuring it yourself. Nowadays, many open source data tools are available as managed cloud services that compete directly with proprietary services. Linux is available preconfigured and installed on server instances on all major clouds. Serverless platforms like AWS Lambda and Google Cloud Functions allow you to deploy event-driven applications in minutes, using mainstream languages such as Python, Java, and Go running atop Linux behind the scenes. Engineers wishing to use Apache Airflow can adopt Google’s Cloud Composer or AWS’s managed Airflow service. Managed Kubernetes allows us to build highly scalable microservice architectures. And so on.

This fundamentally changes the conversation around open source code. In many cases, managed open source is just as easy to use as its proprietary service competitors. Companies with highly specialized needs can also deploy managed open source, then move to self-managed open source later if they need to customize the underlying code.

Another significant trend is the growth in popularity of off-the-shelf data connectors (at the time of this writing, popular ones include Fivetran and Airbyte). Data engineers have traditionally spent a lot of time and resources building and maintaining plumbing to connect to external data sources. The new generation of managed connectors is highly compelling, even for highly technical engineers, as they begin to recognize the value of recapturing time and mental bandwidth for other projects. API connectors will be an outsourced problem so that data engineers can focus on the unique issues that drive their businesses.

The intersection of red-hot competition in the data-tooling space with a growing number of data engineers means data tools will continue decreasing in complexity while adding even more functionality and features. This simplification will only grow the practice of data engineering, as more and more companies find opportunities to discover value in data.

The Cloud-Scale Data OS and Improved Interoperability

Let’s briefly review some of the inner workings of (single-device) operating systems, then tie this back to data and the cloud. Whether you’re utilizing a smartphone, a laptop, an application server, or a smart thermostat, these devices rely on an operating system to provide essential services and orchestrate tasks and processes. For example, I can see roughly 300 processes running on the MacBook Pro that I’m typing on. Among other things, I see services such as WindowServer (responsible for providing windows in a graphical interface) and CoreAudio (tasked with providing low-level audio capabilities).

When I run an application on this machine, it doesn’t directly access sound and graphics hardware. Instead, it sends commands to operating system services to draw windows and play sound. These commands are issued to standard APIs; a specification tells software developers how to communicate with operating system services. The operating system *orchestrates* a boot process to provide these services, starting each service in the correct order based on dependencies among them; it also maintains services by monitoring them and restarting them in the correct order in case of a failure.

Now let's return to data in the cloud. The simplified data services that we've mentioned throughout this book (e.g., Google Cloud BigQuery, Azure Blob Storage, Snowflake, and AWS Lambda) resemble operating system services, but at a much larger scale, running across many machines rather than a single server.

Now that these simplified services are available, the next frontier of evolution for this notion of a cloud data operating system will happen at a higher level of abstraction. Benn Stancil called for the emergence of standardized data APIs for building data pipelines and data applications.[1] We predict that data engineering will gradually coalesce around a handful of data interoperability standards. Object storage in the cloud will grow in importance as a batch interface layer between various data services. New generation file formats (such as Parquet and Avro) are already taking over for the purposes of cloud data interchange, significantly improving on the dreadful interoperability of CSV and the poor performance of raw JSON.

Another critical ingredient of a data API ecosystem is a metadata catalog that describes schemas and data hierarchies. Currently, this role is largely filled by the legacy Hive Metastore. We expect that new entrants will emerge to take its place. Metadata will play a crucial role in data interoperability, both across applications and systems and across clouds and networks, driving automation and simplification.

We will also see significant improvements in the scaffolding that manages cloud data services. Apache Airflow has emerged as the first truly cloud-oriented data orchestration platform, but we are on the cusp of significant enhancement. Airflow will grow in capabilities, building on its massive mindshare. New entrants such as Dagster and Prefect will compete by rebuilding orchestration architecture from the ground up.

This next generation of data orchestration platforms will feature enhanced data integration and data awareness. Orchestration platforms will integrate with data cataloging and lineage, becoming significantly more data-aware in the process. In addition, orchestration platforms will build IaC capabilities (similar to Terraform) and code deployment features (like GitHub Actions and Jenkins). This will allow engineers to code a pipeline and then pass it to the orchestration platform to automatically build, test, deploy, and monitor. Engineers will be able to write infrastructure specifications directly into their pipelines; missing infrastructure and services (e.g., Snowflake databases, Databricks clusters, and Amazon Kinesis streams) will be deployed the first time the pipeline runs.

We will also see significant enhancements in the domain of *live data*—e.g., streaming pipelines and databases capable of ingesting and querying streaming data. In the past, building a streaming DAG was an extremely complex process with a high ongoing operational burden (see Chapter 8). Tools like Apache Pulsar point the way toward a future in which streaming DAGs can be deployed with complex transformations using relatively simple code. We have already seen the emergence of managed stream processors (such as Amazon Kinesis Data Analytics and Google Cloud Dataflow), but we will see a new generation of orchestration tools for managing these services, stitching them together, and monitoring them. We discuss live data in "The Live Data Stack".

What does this enhanced abstraction mean for data engineers? As we've already argued in this chapter, the role of the data engineer won't go away, but it will evolve significantly. By comparison, more sophisticated mobile operating systems and frameworks have not eliminated mobile app developers. Instead, mobile app developers can now focus on building better-quality, more sophisticated applications. We expect similar developments for data engineering as the cloud-scale data OS paradigm increases interoperability and simplicity across various applications and systems.

"Enterprisey" Data Engineering

The increasing simplification of data tools and the emergence and documentation of best practices means data engineering will become more "enterprisey."[2] This will make many readers violently

cringe. The term *enterprise*, for some, conjures Kafkaesque nightmares of faceless committees dressed in overly starched blue shirts and khakis, endless red tape, and waterfall-managed development projects with constantly slipping schedules and ballooning budgets. In short, some of you read "enterprise" and imagine a soulless place where innovation goes to die.

Fortunately, this is not what we're talking about; we're referring to some of the *good* things that larger companies do with data—management, operations, governance, and other "boring" stuff. We're presently living through the golden age of "enterprisey" data management tools. Technologies and practices once reserved for giant organizations are trickling downstream. The once hard parts of big data and streaming data have now largely been abstracted away, with the focus shifting to ease of use, interoperability, and other refinements.

This allows data engineers working on new tooling to find opportunities in the abstractions of data management, DataOps, and all the other undercurrents of data engineering. Data engineers will become "enterprisey." Speaking of which…

Titles and Responsibilities Will Morph...

While the data engineering lifecycle isn't going anywhere anytime soon, the boundaries between software engineering, data engineering, data science, and ML engineering are increasingly fuzzy. In fact, like the authors, many data scientists are transformed into data engineers through an organic process; tasked with doing "data science" but lacking the tools to do their jobs, they take on the job of designing and building systems to serve the data engineering lifecycle.

As simplicity moves up the stack, data scientists will spend a smaller slice of their time gathering and munging data. But this trend will extend beyond data scientists. Simplification also means data engineers will spend less time on low-level tasks in the data engineering lifecycle (managing servers, configuration, etc.), and "enterprisey" data engineering will become more prevalent.

As data becomes more tightly embedded in every business's processes, new roles will emerge in the realm of data and algorithms. One possibility is a role that sits between ML engineering and data engineering. As ML toolsets become easier to use and managed cloud ML services grow in capabilities, ML is shifting away from ad hoc exploration and model development to become an operational discipline.

This new ML-focused engineer who straddles this divide will know algorithms, ML techniques, model optimization, model monitoring, and data monitoring. However, their primary role will be to create or utilize the systems that automatically train models, monitor performance, and operationalize the full ML process for model types that are well understood. They will also monitor data pipelines and quality, overlapping into the current realm of data engineering. ML engineers will become more specialized to work on model types that are closer to research and less well understood.

Another area in which titles may morph is at the intersection of software engineering and data engineering. Data applications, which blend traditional software applications with analytics, will drive this trend. Software engineers will need to have a much deeper understanding of data engineering. They will develop expertise in things like streaming, data pipelines, data modeling, and data quality. We will move beyond the "throw it over the wall" approach that is now pervasive. Data engineers will be integrated into application development teams, and software developers will acquire data engineering skills. The boundaries that exist between application backend systems and data engineering tools will be lowered as well, with deep integration through streaming and event-driven architectures.

Moving Beyond the Modern Data Stack,

Toward the Live Data Stack

We'll be frank: the modern data stack (MDS) isn't so modern. We applaud the MDS for bringing a great selection of powerful data tools to the masses, lowering prices, and empowering data analysts to take control of their data stack. The rise of ELT, cloud data warehouses, and the abstraction of SaaS data pipelines certainly changed the game for many companies, opening up new powers for BI, analytics, and data science.

Having said that, the MDS is basically a repackaging of old data warehouse practices using modern cloud and SaaS technologies; because the MDS is built around the cloud data warehouse paradigm, it has some serious limitations when compared to the potential of next-generation real-time data applications. From our point of view, the world is moving beyond the use of data-warehouse-based internal-facing analytics and data science, toward powering entire businesses and applications in real time with next-generation real-time databases.

What's driving this evolution? In many cases, analytics (BI and operational analytics) will be replaced by automation. Presently, most dashboards and reports answer questions concerning *what* and *when*. Ask yourself, "If I'm asking a *what* or *when* question, what action do I take next?" If the action is repetitive, it is a candidate for automation. Why look at a report to determine whether to take action when you can instead automate the action based on events as they occur?

And it goes much further than this. Why does using a product like TikTok, Uber, Google, or DoorDash feel like magic? While it seems to you like a click of a button to watch a short video, order a ride or a meal, or find a search result, a lot is happening under the hood. These products are examples of true real-time data applications, delivering the actions you need at the click of a button while performing extremely sophisticated data processing and ML behind the scenes with miniscule latency. Presently, this level of sophistication is locked away behind custom-built technologies at large technology companies, but this sophistication and power are becoming democratized, similar to the way the MDS brought cloud-scale data warehouses and pipelines to the masses. The data world will soon go "live."

The Live Data Stack

This democratization of real-time technologies will lead us to the successor to the MDS: the *live data stack* will soon be accessible and pervasive. The live data stack, depicted in Figure 11-1, will fuse real-time analytics and ML into applications by using streaming technologies, covering the full data lifecycle from application source systems to data processing to ML, and back.

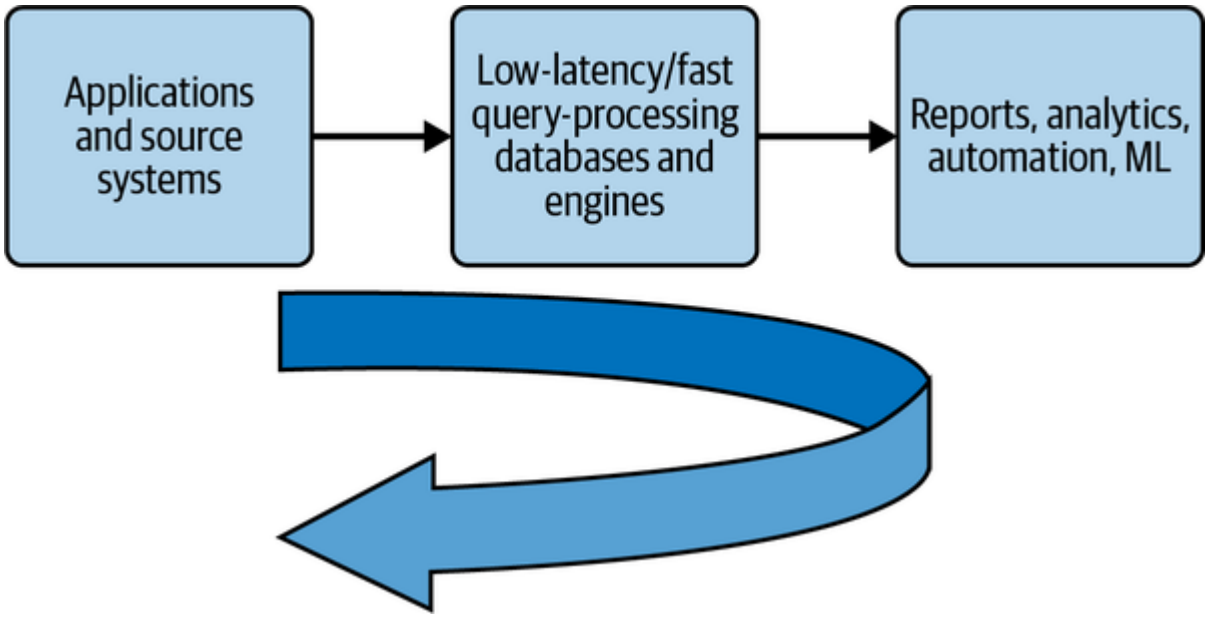

Figure 11-1. In the live data stack, data and intelligence moves in real time between the application and supporting systems

Just as the MDS took advantage of the cloud and brought on-premises data warehouse and pipeline technologies to the masses, the live data stack takes real-time data application technologies used at elite tech companies and makes them available to companies of all sizes as easy-to-use cloud-based offerings. This will open up a new world of possibilities for creating even better user experiences and business value.

Streaming Pipelines and Real-Time Analytical Databases

The MDS limits itself to batch techniques that treat data as bounded. In contrast, real-time data applications treat data as an unbounded, continuous stream. Streaming pipelines and real-time analytical databases are the two core technologies that will facilitate the move from the MDS to the live data stack. While these technologies have been around for some time, rapidly maturing managed cloud services will see them be deployed much more widely.

Streaming technologies will continue to see extreme growth for the foreseeable future. This will happen in conjunction with a clearer focus on the business utility of streaming data. Up to the present, streaming systems have frequently been treated like an expensive novelty or a dumb pipe for getting data from A to B. In the future, streaming will radically transform organizational technology and business processes; data architects and engineers will take the lead in these fundamental changes.

Real-time analytical databases enable both fast ingestion and subsecond queries on this data. This data can be enriched or combined with historical datasets. When combined with a streaming pipeline and automation, or dashboard that is capable of real-time analytics, a whole new level of possibilities opens up. No longer are you constrained by slow-running ELT processes, 15-minute updates, or other slow-moving parts. Data moves in a continuous flow. As streaming ingestion becomes more prevalent, batch ingestion will be less and less common. Why create a batch bottleneck at the head of your data pipeline? We'll eventually look at batch ingestion the same way we now look at dial-up modems.

In conjunction with the rise of streams, we expect a back-to-the-future moment for data transformations. We'll shift away from ELT—in database transformations—to something that looks more like ETL. We provisionally refer to this as *stream, transform, and load* (STL). In a streaming context, extraction is an ongoing, continuous process. Of course, batch transformations won't entirely go away. Batch will still be very useful for model training, quarterly reporting, and more. But streaming transformation will become the norm.

While the data warehouse and data lake are great for housing large amounts of data and performing ad hoc queries, they are not so well optimized for low-latency data ingestion or queries on rapidly moving data. The live data stack will be powered by OLAP databases that are purpose-built for streaming. Today, databases like Druid, ClickHouse, Rockset, and Firebolt are leading the way in powering the backend of the next generation of data applications. We expect that streaming technologies will continue to evolve rapidly and that new technologies will proliferate.

Another area we think is ripe for disruption is data modeling, where there hasn't been serious innovation since the early 2000s. The traditional batch-oriented data modeling techniques you learned about in Chapter 8 aren't suited for streaming data. New data-modeling techniques will occur not within the data warehouse but in the systems that generate the data. We expect data modeling will involve some notion of an upstream definitions layer—including semantics, metrics, lineage, and data definitions (see Chapter 9)—beginning where data is generated in the application. Modeling will also happen at every stage as data flows and evolves through the full lifecycle.

The Fusion of Data with Applications

We expect the next revolution will be the fusion of the application and data layers. Right now, applications sit in one area, and the MDS sits in another. To make matters worse, data is created with no regard for how it will be used for analytics. Consequently, lots of duct tape is needed to make systems talk with one another. This patchwork, siloed setup is awkward and ungainly.

Soon, application stacks will be data stacks, and vice versa. Applications will integrate real-time automation and decision making, powered by the streaming pipelines and ML. The data engineering lifecycle won't necessarily change, but the time between stages of the lifecycle will drastically shorten. A lot of innovation will occur in new technologies and practices that will improve the experience of engineering the live data stack. Pay attention to emerging database technologies designed to address the mix of OLTP and OLAP use cases; feature stores may also play a similar role for ML use cases.

The Tight Feedback Between Applications and ML

Another area we're excited about is the fusion of applications and ML. Today, applications and ML are disjointed systems, like applications and analytics. Software engineers do their thing over here, data scientists and ML engineers do their thing over there.

ML is well-suited for scenarios where data is generated at such a high rate and volume that humans cannot feasibly process it by hand. As data sizes and velocity grow, this applies to every scenario. High volumes of fast-moving data, coupled with sophisticated workflows and actions, are candidates for ML. As data feedback loops become shorter, we expect most applications to integrate ML. As data moves more quickly, the feedback loop between applications and ML will tighten. The applications in the live data stack are intelligent and able to adapt in real time to changes in the data. This creates a cycle of ever-smarter applications and increasing business value.

Dark Matter Data and the Rise of...Spreadsheets?!

We've talked about fast-moving data and how feedback loops will shrink as applications, data, and ML work more closely together. This section might seem odd, but we need to address something that's widely ignored in today's data world, especially by engineers.

What's the most widely used data platform? It's the humble spreadsheet. Depending on the estimates you read, the user base of spreadsheets is between 700 million and 2 billion people. Spreadsheets are the dark matter of the data world. A good deal of data analytics runs in spreadsheets and never makes its way into the sophisticated data systems that we describe in this book. In many organizations, spreadsheets handle financial reporting, supply-chain analytics, and even CRM.

At heart, what is a spreadsheet? A *spreadsheet* is an interactive data application that supports complex analytics. Unlike purely code-based tools such as pandas (Python Data Analysis Library), spreadsheets are accessible to a whole spectrum of users, ranging from those who just know how to open files and look at reports to power users who can script sophisticated procedural data processing. So far, BI tools have failed to bring comparable interactivity to databases. Users who interact with the UI are typically limited to slicing and dicing data within certain guardrails, not general-purpose programmable analytics.

We predict that a new class of tools will emerge that combines the interactive analytics capabilities of a spreadsheet with the backend power of cloud OLAP systems. Indeed, some candidates are already in the running. The ultimate winner in this product category may continue to use spreadsheet paradigms, or may define entirely new interface idioms for interacting with data.

Conclusion

Thank you for joining us on this journey through data engineering! We traversed good architecture, the stages of the data engineering lifecycle, and security best practices. We've discussed strategies for choosing technologies at a time when our field continues to change at an extraordinary pace. In this chapter, we laid out our wild speculation about the near and intermediate future.

Some aspects of our prognostication sit on a relatively secure footing. The simplification of managed tooling and the rise of "enterprisey" data engineering have proceeded day by day as we've written this book. Other predictions are much more speculative in nature; we see hints of an emerging *live data stack*, but this entails a significant paradigm shift for both individual engineers and the organizations that employ them. Perhaps the trend toward real-time data will stall once again, with most companies continuing to focus on basic batch processing. Surely, other trends exist that we have completely failed to identify. The evolution of technology involves complex interactions of technology and culture. Both are unpredictable.

Data engineering is a vast topic; while we could not go into any technical depth in individual areas, we hope that we have succeeded in creating a kind of travel guide that will help current data engineers, future data engineers, and those who work adjacent to the field to find their way in a domain that is in flux. We advise you to continue exploration on your own. As you discover interesting topics and ideas in this book, continue the conversation as part of a community. Identify domain experts who can help you to uncover the strengths and pitfalls of trendy technologies and practices. Read extensively from the latest books, blog posts, and papers. Participate in meetups and listen to talks. Ask questions and share your own expertise. Keep an eye on vendor announcements to stay abreast of the latest developments, taking all claims with a healthy grain of salt.

Through this process, you can choose technology. Next, you will need to adopt technology and develop expertise, perhaps as an individual contributor, perhaps within your team as a lead, perhaps across an entire technology organization. As you do this, don't lose sight of the larger goals of data engineering. Focus on the lifecycle, on serving your customers—internal and external—on your business, on serving and on your larger goals.

Regarding the future, many of you will play a role in determining what comes next. Technology trends are defined not only by those who create the underlying technology but also by those who adopt it and put it to good use. Successful tool *use* is as critical as tool *creation*. Find opportunities to apply real-time technology that will improve the user experience, create value, and define entirely new types of applications. It is this kind of practical application that will materialize the *live data stack* as a new industry standard; or perhaps some other new technology trend that we failed to identify will win the day.

Finally, we wish you an exciting career! We chose to work in data engineering, to consult, and to write this book not simply because it was trendy but because it was fascinating. We hope that we've managed to convey to you a bit of the joy we've found working in this field.

[1] Benn Stancil, "The Data OS," *benn.substack*, September 3, 2021, *https://oreil.ly/HetE9*.

[2] Ben Rogojan, "Three Data Engineering Experts Share Their Thoughts on Where Data Is Headed," *Better Programming*, May 27, 2021, *https://oreil.ly/IsY4W*.