Data Journey First DataOps

Third Edition: 2023

THE DATA OPS COOKBOOK

Methodologies and Tools That Reduce Analytics Cycle Time While Improving Quality

Christopher Bergh, Gil Benghiat and Eran Strod

The DataOps Cookbook

Methodologies and Tools That Reduce Analytics Cycle Time While Improving Quality

Data Journey First DataOps
Third Edition

by

Christopher Bergh, Gil Benghiat, and Eran Strod

The DataOps Cookbook

To order additional copies of this book:
info@datakitchen.io

105 Waltham Street #101
Lexington, MA 02421

Printed in the United States of America
Cover design and layout by Ariel Plotkin-Gould

Table of Contents

Preface to the Third Edition

Since the first edition of the DataOps Cookbook in 2019, we have talked with thousands of companies about their struggles to deliver data-driven insight to their customers. In many ways, they all have the same problems. They have built data and analytic systems with great hope of success. Still, they are burdened with too many errors, are overwhelmed with custom requests, and know they fail to succeed in their goal of leading their organizations to be more data-driven.

And on top of that, they are all unhappy, stressed, and wondering what went wrong. As an industry, we have a conceptual hole in how we think about data analytic systems. We build them and put them into production, but then we hope all the steps data goes through from source to customer value work out correctly. We all know that our customers frequently find data and dashboard problems. Teams are shamed and blamed for problems they didn't cause.

We had the same problem starting in 2005 when we left software development and started to lead data teams. Over the years, we have put together these principles of DataOps, built multiple generations of software to solve the problem, and helped many customers succeed. We talk with data teams every few days with the same “morning dread” I had lending data teams 15 years ago. That feeling that something is going to go wrong, you’ll have no idea how to find it, and fixing it will put off all the other tasks that need to get done.

Enter 'Data Journey First DataOps.' The new idea showcased in the third edition of the DataOps Cookbook is to focus first on understanding and observing the journey that data takes through your production environment - from ingestion to processing to delivering actionable insights. This monitoring process identifies data errors, tool problems, and timing issues, enabling a quick win for your DataOps implementation by driving immediate improvements. Lowering production errors increases the reliability of your data and gives your team more time to focus on automation.

We’ve included many new chapters about Data Journeys, Observability, and the benefits of focusing on production errors as the first step in DataOps. These include

- Data Journey Manifesto
- Why the Data Journey Manifesto?
- Five Pillars of Data Journeys
- Data Journey First DataOps
- The Terms and Conditions of a Data Contract are Data Tests
- “You Complete Me,” said Data Lineage to Data Journeys.
- The Only Success Metrics That Matter For Your Data & Analytics Team
- DataOps Observability: Taming the Chaos:
- DataOps TestGen: 'Mystery Box Full Of Data Errors'
- Tomorrow’s Forecast: Cloudy with a Chance of Data Errors

Thanks for reading!

Introduction To The First Edition

In the early 2000s, Chris and Gil worked at a company that specialized in analytics for the pharmaceutical industry. It was a small company that offered a full suite of services related to analytics — data engineering, data integration, visualization and what is now called “data science.” Their customers were marketing and sales executives who tend to be challenging because they are busy, need fast answers and don’t understand or care about the underlying mechanics of analytics. They are business people, not technologists.

When a request from a customer came in, Chris and Gil would gather their team of engineers, data scientists and consultants to plan out the how to get the project done. After days of planning, they would propose their project plan to the customer. “It will take two weeks.” The customer would shoot back, “I need it in two hours!”
Walking back to their office, tail between their legs, they would pick up the phone. It was a customer boiling over with anger. There was a data error. If it wasn’t fixed immediately the customer would find a different vendor.

The company had hired a bunch of smart people to deliver these services. “ I want to innovate — Can I try out this new open source tool,” the team members would ask. “No,” the managers would have to answer. “We can’t afford to introduce technical risk.”

They lived this life for many years. How do you create innovative data analytics? How do you not have embarrassing errors? How do you let your team easily try new ideas? There had to be a better way.

They found their answer by studying the software and manufacturing industries which had been struggling with these same issues for decades. They discovered that data-analytics cycle time and quality can be optimized with a combination of tools and methodologies that they now call ***DataOps***. They decided to start a new company. The new organization adopted the kitchen metaphor for data analytics. After all, cooking up charts and graphs requires the right *ingredients* and *recipes*.

The DataKitchen founders (Chris, Gil and Eric) built their own DataOps Platform. What happened next was remarkable. When analytics go faster, users embrace analytics and innovate. Rapid, high-quality analytics can unlock an organization's creative potential.

Having experienced this transformation, the DataKitchen founders sought a way to help other data professionals. There are so many talented people stuck in no-win situations. This book is for data professionals who are living the nightmare of slow, buggy analytics and frustrated users. It will explain why working weekends isn't the answer. It provides you with practical steps that you can take tomorrow to improve your analytics cycle time.

DataKitchen markets a DataOps Platform that will help analytics organizations implement DataOps. However, this book isn't really about us and our product. It is about you, your challenges, your potential and getting your analytics team back on track.

The values and principles that are central to DataOps are listed in the DataOps Manifesto which you can read below. If you agree with it, please join the thousands of others who share these beliefs by signing the manifesto. There may be aspects of the manifesto that require further explanation. Please read on. By the end of this book, it should all make sense. You'll also notice that we've included some real recipes in this book. These are some of our favorites. We hope you enjoy them!
Please reach out to us at info@datakitchen.io with any comments or questions.

Chris, Gil and Eran

The DataOps Manifesto

Background

Through firsthand experience working with data across organizations, tools, and industries we have uncovered a better way to develop and deliver analytics that we call DataOps. Whether referred to as data science, data engineering, data management, big data, business intelligence, or the like, through our work we have come to value in analytics:

• Individuals and interactions over processes and tools

• Working analytics over comprehensive documentation

• Customer collaboration over contract negotiation

• Experimentation, iteration, and feedback over extensive upfront design

• Cross-functional ownership of operations over siloed responsibilities

DataOps Principles

1. CONTINUALLY SATISFY YOUR CUSTOMER

Our highest priority is to satisfy the customer through the early and continuous delivery of valuable analytic insights from a couple of minutes to weeks.

2. VALUE WORKING ANALYTICS

We believe the primary measure of data analytics performance is the degree to which insightful analytics are delivered, incorporating accurate data, atop robust frameworks and systems.

3. EMBRACE CHANGE

We welcome evolving customer needs, and in fact, we embrace them to generate competi- tive advantage. We believe that the most efficient, effective, and agile method of communi- cation with customers is face-to-face conversation.

4. IT'S A TEAM SPORT

Analytic teams will always have a variety of roles, skills, favorite tools, and titles.

5. DAILY INTERACTIONS

Customers, analytic teams, and operations must work together daily throughout the project.

6. SELF-ORGANIZE

We believe that the best analytic insight, algorithms, architectures, requirements, and de- signs emerge from self-organizing teams.

7. REDUCE HEROISM

As the pace and breadth of need for analytic insights ever increases, we believe analytic teams should strive to reduce heroism and create sustainable and scalable data analytic teams and processes.

8. REFLECT

Analytic teams should fine-tune their operational performance by self-reflecting, at regular intervals, on feedback provided by their customers, themselves, and operational statistics

9. ANALYTICS IS CODE

Analytic teams use a variety of individual tools to access, integrate, model, and visualize data. Fundamentally, each of these tools generates code and configuration which describes the actions taken upon data to deliver insight.

10. ORCHESTRATE

The beginning-to-end orchestration of data, tools, code, environments, and the analytic team's work is a key driver of analytic success.

11. MAKE IT REPRODUCIBLE

Reproducible results are required and therefore we version everything: data, low-level hardware and software configurations, and the code and configuration specific to each tool in the toolchain.

12. DISPOSABLE ENVIRONMENTS

We believe it is important to minimize the cost for analytic team members to experiment by giving them easy to create, isolated, safe, and disposable technical environments that reflect their production environment.

13. SIMPLICITY

We believe that continuous attention to technical excellence and good design enhances agility; likewise simplicity—the art of maximizing the amount of work not done—is essential.

14. ANALYTICS IS MANUFACTURING

Analytic pipelines are analogous to lean manufacturing lines. We believe a fundamental con- cept of DataOps is a focus on process-thinking aimed at achieving continuous efficiencies in the manufacture of analytic insight.

15. QUALITY IS PARAMOUNT

Analytic pipelines should be built with a foundation capable of automated detection of ab- normalities (jidoka) and security issues in code, configuration, and data, and should provide continuous feedback to operators for error avoidance (poka yoke).

16. MONITOR QUALITY AND PERFORMANCE

Our goal is to have performance, security and quality measures that are monitored continu- ously to detect unexpected variation and generate operational statistics.

17. REUSE

We believe a foundational aspect of analytic insight manufacturing efficiency is to avoid the repetition of previous work by the individual or team.

18. IMPROVE CYCLE TIMES

We should strive to minimize the time and effort to turn a customer need into an analytic idea, create it in development, release it as a repeatable production process, and finally refactor and reuse that product.

Join the Thousands of People Who Have Already Signed The Manifesto

19. Start With Your Data Journey:

None of your customers ever said, “I want more errors in my data.” Trust in data analytics is key to adoption. Make your first step in DataOps to understand and observe the journey that data takes through your production environment – from ingestion to processing to delivering actionable insights.

The Data Journey Manifesto

After working for years with many data analytic teams using different technologies, we've come to believe that reducing errors and defects in the insight production process is the key to success. We've been shamed and blamed by our customers for problems with the data we did not cause, trapped with existing data processes we don't understand, and sat in dread every morning, waiting for something to break in our data, reports, models, or other customer deliverables. We are tired of the stress and wasted productivity needed to find problems deep within the system. We want a method that enables us to observe our data's complicated paths to avoid problems and errors, and customer frustration.

Principles

Define and Know What Should Be:

At any time, in data analytic systems, know what should be, what is, and the exact difference between the two.

Make Hope Infrequent:

Hoping your data systems work in production is not a strategy. It's a recipe for failure.

Customers Finding Problems Is Not OK:

Customers finding problems in your data analytics is unacceptable. Find problems before your customer does.

Don't Trust Your Data Providers:

Some data providers are on top of their game, while others barely fulfill an unwanted task. Either way, providers make mistakes. Get used to it. Protect against it. Use it as an opportunity for improvement.

Don't Assume What Worked Last Week Will Work Today:

Your team is constantly changing the code and configuration in your data estate. Make sure it is still working.

Find The Problem Fast:

Finding the exact source of the problem – whether in raw data, integrated data, models, reports, servers, software, and/or code – is half the battle.

Perfect Data Quality Is Not a Cure-all:

Even with perfect initial data quality, many other things can still go wrong

Avoid Manual Quality Testing Like The Plague:

Completely automate the testing of your data and tools.

Your Data Production Is A Factory:

Heed the lessons of Toyota, Lean, and Deming. Every tool in the system – data ingestion, transformation, database, predictive model, and visualization – is a workstation on that assembly line.

Complicated Data Architectures Need NASA's Mission Control:

Our modern data architectures are built on performative complexity. Your data architecture has many 'little boxes,' each of which can fail.

We Need A New Idea: The Data Journey

Is The Expectation Layer:

Data Journeys represent the expectations of all the myriad paths data takes from source to the insight value you deliver to your customer.

Observes, But Does Not "Run":

Data Journeys track and monitor all levels of the data stack, from data quality validation to servers, software, code, costs, and utilization. It's the 'digital twin' of complicated batch and streaming data architectures. Data Journeys hold expectations and don't 'run' anything.

Alerts In Real Time:

A Data Journey supplies real-time statuses and alerts. With this information, you can know if everything ran on time and without errors and immediately identify the parts that didn't.

Goes Across And Down:

Data Journeys define the process lineage for the many complex elements that deliver insight. It covers components '*across*' your toolchain and 'down' your technology stack, including logs, messages, run status, metrics, data validation tests, and other information from your data estate.

Groups Components:

A Data Journey has many components. Paraphrasing Anna Karenina: all happy, error-free Data Journeys are alike; each unhappy Data Journey is broken in its unique way. Data Journeys find the 'unhappy' component quickly.

Trusts But Verifies:

"Trust, but verify" is an old Russian proverb. Trust comes from monitoring every component in your Data Journey, then verifying the data that touches it. Test, validate, and look for anomalies at every step.

Shares Schedules:

Your production schedule is public property – share widely. Use it to eliminate silos between the engineers who built the components of the Data Journeys, the operators who run them, the customers who use them, and the managers who get yelled at if there is a problem.

Learns From Production History:

Each instance of a Data Journey provides history and evidence to find the root cause of a defect, help your team improve, and share evidence of improvement in production errors and unmet SLAs.

It Can Be A Business Workflow, Too:

An instance of a Data Journey usually represents the batch or streaming technical steps used to create value from data. However, some Data Journey instances represent a business workflow in 'the real world. A particular customer may use that Data Journey to check on the status of that process.

Lowers Deployment Risk:

Use the Data Journey to find the impact of regressions during development. You can't ship code to production based on manual or static analysis. Use the Data Journey to help automatically regression test your code in development to find the impact of changes.

Reduces Errors And Drives Productivity:

Your data analytics team productivity drops when they spend time finding and fixing problems in production. Unidentified errors in your Data Journeys cause costly business mistakes, erode the trust of your customers, and may have a compliance risk.

DataOps Ribeye

by Christopher Bergh

INGREDIENTS AND TOOLS

- Rib eye steaks 1 ½-2" thick
- Large baking potatoes
- Corn on the husk
- Olive oil
- Ground sea salt & Fresh coarse ground pepper or Penzey's Chicago Steak Seasoning
- Green Egg or another charcoal grill
- Instapen thermometer

PREPARE CORN AND POTATOES

1. Season/marinade steaks to your liking, but I now like the 'TRex way of prepping/grilling my steaks (see below): coat the steaks with olive oil, liberally apply salt and pepper (or Penzey's Chicago Steak Seasoning) and rub the mixture into the steaks on both sides.
2. Soak corn in water for at least 30 minutes or more.
3. Brush potatoes with olive oil and salt on both sides. Heat Green Egg to 400 degrees with a few chunks of Hickory wood and put potatoes directly on grill grid, direct heat, turning once after 30 minutes. About 20-25 minutes before the potatoes are done, put the corn on the grill grid. Turn the corn about every 8-10 minutes or just enough to keep the corn from getting burned (a little charring of the corn is okay, though).
4. Remove the potatoes and corn and wrap in aluminum foil and place in a warming oven or drawer at 170 degrees.

PREPARE THE STEAKS

1. TRex method of grilling/cooking steaks: Crank the draft door and daisy wheel open all the way on the Egg for maximum temperature. When you get the Green Egg up to at least over 600 degrees (be careful to "burp" the Egg at this high temp), put the steaks on and sear for 90 seconds per side.
2. Take the steaks off the Egg and let sit/rest for 20 minutes. Shut down the lower vent and Daisy Wheel to get the Green Egg back down to 425 degrees.
3. Throw some more chunks of Hickory wood on the fire and try and maintain a temp around 400 degrees. After 10 minutes of letting the steaks sit/rest put back on the Green Egg and cook for approximately 4-5 minutes per side for a medium-rare/medium result. Check internal temperature of steak so that it is 120 degrees

“What Is DataOps”

You can view DataOps in the context of a century-long evolution of ideas that improve how people manage complex systems. It started with pioneers like W. Edwards Deming and statistical process control - gradually these ideas crossed into the technology space in the form of Agile, DevOps and now, DataOps. In the next section we will examine how these methodologies impact productivity, quality and reliability in data analytics.

Delivering Analytics at Amazon Speed

The world changed in February 2005 when Amazon Prime brought flat-rate, unlimited, two-day shipping into a world where people expected to pay extra to receive packages in four to six business days. Since its launch, Amazon Prime has completely transformed the retail market, making low-cost, predictable shipping an integral part of consumer expectations. This business model, which some have called the “on-demand economy,” is popping up in many industries and markets across the globe.

For example, some may remember video stores where movies were rented for laterviewing. Today, 65 percent of global respondents to a recent Nielsen survey watch video on demand (VOD), many of them daily. With VOD, a person’s desire to watch a movie is fulfilled within seconds. Amazon participates in the VOD market with their Amazon Prime Video service. Instant fulfillment of customer orders seems to be part of Amazon’s business model. They have even brought that capability to IT. About 10 years ago, Amazon Web Services (AWS) began offering computing, storage, and other IT infrastructure on an as-needed basis. Whether the need is for one server or thousands and whether for hours, days, or months, you only pay for what you use, and the resources are available in just a few minutes.

To successfully compete in today’s on-demand economy, companies need to deliver their products and services just as Amazon has done—in other words, at *Amazon speed*. What might be surprising to many is how the expectations of instant fulfillment are crossing over

into data analytics, which, along with everything else in the digital economy, is now expected to happen at Amazon speed and with Amazon predictability.

A typical example: the VP of sales enters the office of the chief data officer (CDO). She'd like to cross-reference the customer database with some third-party consumer data. The CDO asks for time to study the problem and, days later, has planned the project. Resources will be allocated and configured, schemas will be updated, reports will be elegantly designed, and the delivery pipeline will be thoroughly tested. The changes will take several weeks. "Not acceptable," the VP of sales fires back. The new analytics are needed for a meeting with the board later in the week. "The competition is ahead of us; we can't wait weeks." This scenario is playing out in one form or another in corporations around the globe.

ANALYTICS IN THE ON-DEMAND ECONOMY

Analytics must be delivered rapidly in order to meet user expectations in the on-demand economy. This is simply not possible with an approach that depends upon "hope and heroism" or "caution."

In order to deliver value consistently, quickly and accurately, data-analytics teams must learn to create and publish analytics in a new way. We call this new approach *DataOps*. DataOps is a combination of tools and methods, which streamline the development of new analytics while ensuring impeccable data quality. DataOps helps shorten the cycle time for producing analytic value and innovation, while avoiding the trap of "hope, heroism and caution."

Data Analytics Can Learn from Agile

If you were managing 100 software developers, you would have to choose the best way to maximize their productivity. Since the dawn of the computer era many software project management approaches have been tried. The waterfall model dominated software project management up until the 1990's. In the early days of computing, project management was adapted from the manufacturing and construction industries, which required detailed planning and a great degree of structure. Projects were organized into phases (conception, initiation, analysis, design, construction, testing, production/implementation and maintenance) and progressed through these phases sequentially. Once a phase was done, the team moved forward to the next phase.

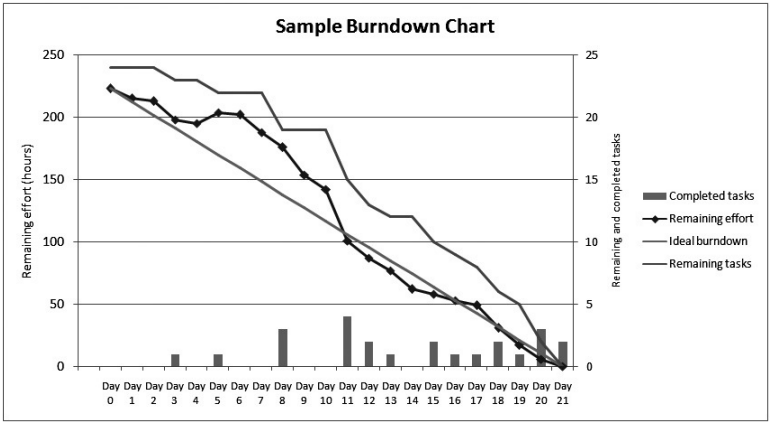


Figure 1: In Agile development, a burndown chart shows work remaining over time.

The waterfall model is better suited to situations where the requirements are fixed and well understood up front. This is nothing like the technology industry where the competitive environment evolves rapidly. In the 1980's a typical software project required about 12 calendar months. In technology-driven businesses (i.e. nearly everyone these days) custom- ers demand new features and services, and competitive pressures change priorities on a seemingly daily basis. The waterfall model has no mechanism to respond to these changes. In waterfall, changes trigger a seemingly endless cycle of replanning causing delays and resulting in project budget overruns.

In the early 2000's, the software industry embraced a new approach to code production called Agile Development. Agile is an umbrella term for several different iterative and incremental software development methodologies.

In Agile Software Development, the team and its processes and tools are organized around the goal of publishing releases to the users every few weeks (or at most every few months). A development cycle is called a sprint or an iteration. At the beginning of an iteration, the team commits to completing working and (the most) valuable changes to the code base. Features are associated with user stories, which help the development team

understand the context behind requirements. User stories include descriptions of features and acceptance criteria. The Agile methodology is particularly good for non-sequential product development where market requirements are quickly evolving. This is similar to the data-analytics environment where each new analysis and report of the data inspires requests for additional queries.

Agile is widely credited with boosting software productivity. One study sponsored by the Central Ohio Agile Association and Columbus Executive Agile Special Interest Group found that Agile projects were completed 31 percent faster and with a 75 percent lower defect rate than the industry norm. The vast majority of companies are getting on-board. In a survey of 400 IT professionals by TechBeacon, two-thirds described their company as either “pure agile” or “leaning towards agile. Among the remaining one third of companies, most use a hybrid approach, leaving only nine percent using a pure waterfall approach.

In an increasingly competitive marketplace, Agile methods allow companies to become more responsive to customer requirements and accelerate time to market. Agile also improves ROI because features delivered in each iteration can be immediately monetized instead of waiting months for a big release. Agile is the major reason that release frequency improved from around 3 months in the 1990’s to about 3 weeks in the 2000’s. However, improvements didn’t stop there. Today releases are occurring every few seconds using an updated approach
which builds upon Agile.

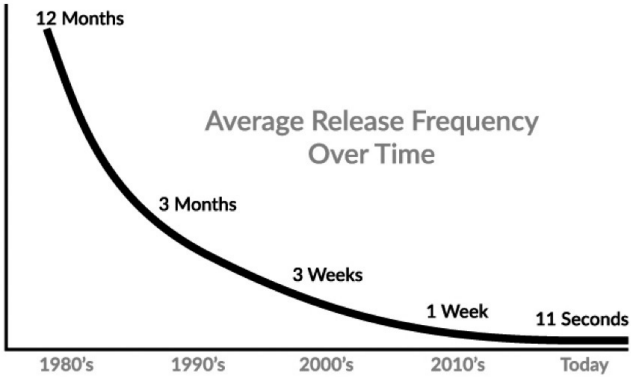


Figure 2: The decrease in release cycle time as software development evolved from waterfall to Agile to DevOps.

Data Analytics Can Learn from DevOps

Before the advent of on-demand cloud services, the various groups in software development (design, development, test, quality, support, …) had to set-up their own infrastructure. Whatever components were needed (physical servers, networks, storage, software, …) had to be ordered, installed, configured and managed by the IT department. Servers would be ordered at different times and from different vendors, each slightly different from the other.

Depending on the task at hand, different machines could have a different array of software applications with revisions of each app being continuously updated. Target devices could range from embedded IoT (Internet of Things) to the largest, most powerful servers. With all of this variability, it was quite common for individuals within the company to be running code in different environments. Outside the four walls of the company, customers could be running in yet another environment. This situation presented challenges.

The software development pipeline can be organized as follows: planning, resourcing, development, testing, quality assurance, and deployment. Continuous delivery requires automation from planning all the way through to delivery/deployment. Previously, software development, testing, quality assurance, and customers could each be running in different environments. This hampered their ability to communicate and led to misunderstandings and delays.

If, for example, the customer reported a problem, it might not be replicable in the support, test or development groups due to differences in the hardware and software environments being run. This lack of alignment fostered misunderstandings and delays and often led to a lack of trust and communication between the various stakeholders.

In a complex world requiring the physical provisioning of servers, installation of stacks and frameworks, and numerous target devices, the standardization and control of the run-time environment has been difficult and slow. It became necessary to break down barriers between the respective teams in the software development pipeline. This merging of development and IT/Operations is widely known as DevOps, which also has had enormous impact on the world of software development. DevOps improves collaboration between employees from the planning through the deployment phases of software. It seeks to reduce time to deployment, decrease time to market, minimize defects, and shorten the time required to fix problems.

About a decade ago, Amazon Web Services (AWS) and other cloud providers, began offering computing, storage and other IT resources as an on-demand service. No more waiting weeks or months for the IT department to fulfill a request for servers. Cloud providers now allow you to order computing services, paying only for what you use, whether that is one processor for an hour or thousands of processors for months. These on-demand cloud services have enabled developers to write code that provisions processing resources with strictly specified environments, on-demand, in just a few minutes. This capability has been called Infrastructure as Code (IaC). IaC has made it possible for everyone in the software development pipeline, all the different groups mentioned above, to use an identical environment tailored to application requirements. With IaC, design, test, QA and support

can easily get on the same page. This leads to much better collaboration between the groups and breaks down barriers that prevented open communication. In other words, no more finger pointing.With IT infrastructure being defined by code, the hard divisions between IT operations and software development are able to blur. The merger of development and operations is how the term DevOps originated.

With the automated provisioning of resources, DevOps paved the way for a fully automated test and release process. The process of deploying code that used to take weeks, could now be completed in minutes. Major organizations including Amazon, Facebook and Netflix are now operating this way. At a recent conference, Amazon disclosed that their AWS team performs 50,000,000 code releases per year. This is more than one per second! This methodology of rapid releases is called continuous delivery or alternatively, continuous deployment, when new features (and fixes) are not only delivered internally but fully deployed to customers.

DevOps starts with continuous delivery and Agile development and adds automated provisioning of resources (infrastructure as code) and cloud services (platform as a service) to ensure that the same environment is being utilized at every stage of the software development pipeline. The cloud provides a natural platform that allows individuals to create and define identical run-time environments. DevOps is beginning to achieve critical mass in terms of its adoption within the world of software development.
DevOps improves collaboration between employees from the planning through the deployment phases of software. It seeks to reduce time to deployment, decrease time to market, minimize defects, and shorten the time required to fix problems.

The impact of DevOps on development organizations was shown in a 2014 survey, "The 2014 State of DevOps Report" by Puppet Labs, IT Revolution Press and ThoughtWorks, based on 9,200 survey responses from technical professionals. The survey found that IT organizations implementing DevOps were deploying code 30 times more frequently and with 50 percent fewer failures. Further, companies with these higher-performing IT organizations tended to have stronger business performance, greater productivity, higher profitability and larger market share. In other words, DevOps is not just something that engineers are doing off in a dark corner. It is a core competency that helps good companies become better.

The Data analytics team transforms raw data into actionable information that improves decision making and provides market insight. Imagine an organization with the best data analytics in the industry. That organization would have a tremendous advantage over competitors. That could be you.

Data Analytics Can Learn from Manufacturing

What could data analytics professionals possibly learn from industrial manufacturers? It turns out, a lot. Automotive giant Toyota pioneered a set of methods, later folded into a discipline called *lean manufacturing,* in which employees focus relentlessly on improving quality and reducing non-value-add activities. This culture enabled Toyota to grow into the one of the world's leading car companies. The Agile and DevOps methods that have led to stellar improvements in coding velocity are really just an example of lean manufacturing principles applied to software development.

Conceptually, manufacturing is a pipeline process. Raw materials enter the manufacturing floor through the stock room, flow to different work stations as work-in-progress and exit as finished goods. In data-analytics, data progresses through a series of steps and exits in the form of reports, models and visualizations. Each step takes an input from the previous step, executes a complex procedure or set of instructions and creates output for the subsequent step. At an abstract level, the data-analytics pipeline is analogous to a manufacturing process. Like manufacturing, data analytics executes a set of operations and attempts to produce a consistent output at a high level of quality. In addition to lean-manufacturing-inspired methods like Agile and DevOps, there is one more useful tool that can be taken from manufacturing and applied to data-analytics process improvement.

W. Edwards Deming championed statistical process control (SPC) as a method to improve manufacturing quality. SPC uses real-time product or process measurements to monitor and control quality during manufacturing processes. If the process measurements are maintained within specific limits, then the manufacturing process is deemed to be functioning properly. When SPC is applied to the data-analytics pipeline, it leads to remarkable improvements in efficiency and quality. For example, Google executes over one hundred million automated test scripts per day to validate any new code released by software developers. In the Google consumer surveys group, code is deployed to customers eight minutes after a software engineer finishes writing and testing it.

In data analytics, tests should verify that the results of each intermediate step in the production of analytics matches expectations. Even very simple tests can be useful. For example, a simple row-count test could catch an error in a join that inadvertently produces a Cartesian product. Tests can also detect unexpected trends in data, which might be flagged as warnings. Imagine that the number of customer transactions exceeds its historical average by 50%. Perhaps that is an anomaly that upon investigation would lead to insight about business seasonality.Tests in data analytics can be applied to data or models either at the input or output of a phase in the analytics pipeline. Tests can also verify business logic.

Business logic tests validate assumptions about the data. For example:

• Customer Validation – Each customer should exist in a dimension table

• Data Validation – At least 90 percent of data should match entries in a dimension table

Input tests check data prior to each stage in the analytics pipeline. For example:

• Count Verification – Check that row counts are in the right range, ...

• Conformity – US Zip5 codes are five digits, US phone numbers are 10 digits, ...

• History – The number of prospects always increases, ...

• Balance – Week over week, sales should not vary by more than 10%, ...

• Temporal Consistency – Transaction dates are in the past, end dates are later than start dates, ...

• Application Consistency – Body temperature is within a range around 98.6F/37C, ...

• Field Validation – All required fields are present, correctly entered, ...

Output tests check the results of an operation, like a Cartesian join. For example:

• Completeness – Number of customer prospects should increase with time

• Range Verification – Number of physicians in the US is less than 1.5 million

The data analytics pipeline is a complex process with steps often too numerous to be monitored manually. SPC allows the data analytics team to monitor the pipeline end-to-end from a big-picture perspective, ensuring that everything is operating as expected. As an automated test suite grows and matures, the quality of the analytics is assured without adding cost. This makes it possible for the data analytics team to move quickly — enhancing analytics to address new challenges and queries — without sacrificing quality.

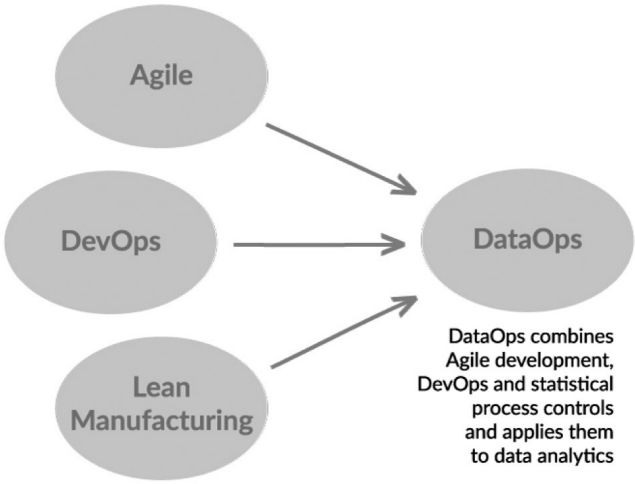


Figure 3: DataOps has evolved from lean manufacturing and software methodologies.

DataOps for Data Analytics

The speed and flexibility achieved by Agile and DevOps, and the quality control attained by SPC, can be applied to data analytics. Leading edge proponents of this approach are calling it DataOps. DataOps, simply stated, is Agile development and DevOps with statistical process control, for data analytics. DataOps applies Agile methods, DevOps, and manufacturing quality principles, methodologies and tools, to the data-analytics pipeline. The result is a rapid-response, flexible and robust data-analytics capability, which is able to keep up with the creativity of internal stakeholders and users.

DataOps is an analytic development method that emphasizes communication, collaboration, integration, automation, measurement and cooperation between data scientists, analysts, data/ETL (extract, transform, load) engineers, information technology (IT), and quality assurance/governance. The method acknowledges the interdependence of the entire end-to-end analytic process. It aims to help organizations rapidly produce insight, turn that insight into operational tools, and continuously improve analytic operations and performance. It enables the whole analytic team involved in the analytic process to follow the values laid out in the DataOps Manifesto.

When DataOps is implemented correctly, it addresses many of the issues discussed earlier that have plagued data-analytics teams.

Challenge	DataOps Approach
Changing Requirements	The team delivers something of value to users at each iteration. If the requirements change, it is simple to put new requirements on the request list for a future iteration.
Slipped schedules	Iterations occur in rapid succession allowing greater visibility to the progress that is being made. As a team gains experience with DataOps, their forecasting improves.
Disappointed users	Users receive new features quickly and give feedback to the development team about how to keep improving the data analytics with even more new features.
Inflexibility	DataOps enables teams to respond quickly to change. The team can pivot at the beginning of the next iteration, which by definition is always relatively soon.
Poor quality	Extensive, automated testing, in the form of statistical process control, is a key element in DataOps.
Low ROI	With features being delivered in short increments, the monetization of the data-analytics investment begins much earlier, improving ROI.
Irrelevant features	The data-analytics team is churning out management's highest priority features in quick succession.

Table 1: Challenge/DataOps Approach

DataOps views the data-analytics pipeline as a process and as such focuses on how to make the entire process run more rapidly and with higher quality, rather than optimizing the productivity of any single individual or tool by itself.

Key Benefits of DataOps

DataOps can accelerate the ability of data-analytics teams to create and publish new analytics to users. It requires an Agile mindset and must also be supported by an automated platform which incorporates existing tools into a DataOps development pipeline. DataOps spans the entire analytic process, from data acquisition to insight delivery. Its goal is to achieve more insight and better analysis, while still being faster, cheaper and higher quality.

The key business benefits of adopting DataOps are:

• Reduce time to insight

• Improve analytic quality

• Lower the marginal cost to ask the next business question

• Improve analytic team morale by going beyond hope, heroism and caution

• Promote team efficiency through agile process, reuse and refactoring

DataKitchen markets an automated DataOps platform that helps companies accelerate their DataOps implementation, but this book is about DataOps not us. This book is not trying to sell you anything. You can implement DataOps all by yourself, using your existing tools, by implementing the seven steps described in the next section. If you desire assistance, there is an ecosystem of DataOps vendors who offer a variety of innovative solutions and services.

The Seven Steps to Implement DataOps

Data analytics has become business critical, but requirements quickly evolve and data-analytics teams that respond to these challenges in the traditional ways often end up facing disappointed users. DataOps offers a more effective approach that optimizes the productivity of the data analytics pipeline by an order of magnitude.

Imagine the next time that the Vice President of Marketing requests a new customer segmentation, by tomorrow. With DataOps, the data-analytics team can respond 'yes' with complete confidence that the changes can be accomplished quickly, efficiently and robustly. How then does an organization implement DataOps? You may be surprised to learn that an analytics team can migrate to DataOps in seven simple steps.

STEP 1 - ADD DATA AND LOGIC TESTS

If you make a change to an analytic pipeline, how do you know that you did not break anything? Automated testing insures that a feature release is of high quality without requiring time-consuming, manual testing. The idea in DataOps is that every time a data-analytics team member makes a change, he or she adds a test for that change. Testing is added incrementally, with the addition of each feature, so testing gradually improves and quality is literally built in. In a big run, there could be hundreds of tests at each stage in the pipeline.

Adding tests in data analytics is analogous to the statistical process control that is implemented in a manufacturing operations flow. Tests insure the integrity of the final output by verifying that work-in-progress (the results of intermediate steps in the pipeline) matches expectations. Testing can be applied to data, models and logic. The figure below shows examples of tests in the data-analytics pipeline.

For every step in the data-analytics pipeline, there should be at least one test. The philosophy is to start with simple tests and grow over time. Even a simple test will eventually catch an error before it is released out to the users. For example, just making sure that row counts are consistent throughout the process can be a very powerful test. One could easily make a mistake on a *join* and make a *cross product* which fails to execute correctly. A simple row-count test would quickly catch that.

Tests can detect warnings in addition to errors. A warning might be triggered if data exceeds certain boundaries. For example, the number of customer transactions in a week may be OK if it is within 90% of its historical average. If the transaction level exceeds that, then a warning could be flagged. This might not be an error. It could be a seasonal occurrence for example, but the reason would require investigation. Once recognized and understood, the users of the data could be alerted.

DataOps is not about being perfect. In fact, it acknowledges that code is imperfect. It's natural that a data-analytics team will make a best effort, yet still miss something. If so, they can determine the cause of the issue and add a test so that it never happens again. In a rapid release environment, a fix can quickly propagate out to the users.

With a suite of tests in place, DataOps allows you to move fast because you can make changes and quickly rerun the test suite. If the changes pass the tests, then the data-analytics team member can be confident and release it. The knowledge is built into the system and the process stays under control. Tests catch potential errors and warnings before they are released so the quality remains high.

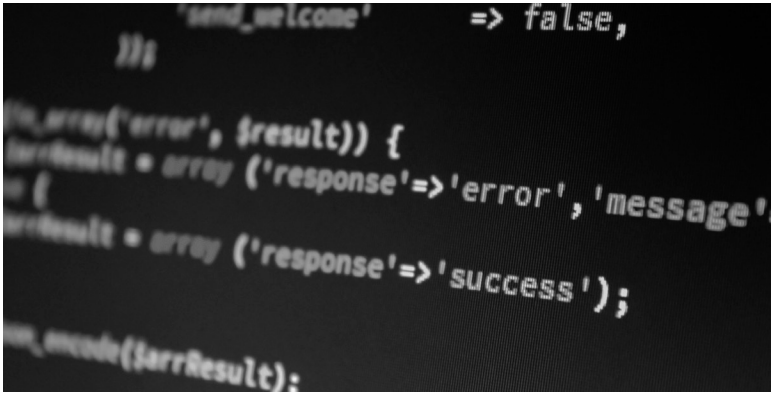
Automated tests continuously monitor the data pipeline for errors and anomalies. They work nights, weekends and holidays without taking a break. If you build a DataOps dashboard, you can view the high-level state of your data operations at any time. If warning and failure alerts are automated, you don't have to constantly check your dashboard. Automated testing frees the data-analytics team from the drudgery of manual testing, so they can focus on higher value-add activities.

Figure 4: Tests enable the data professional to apply statistical process controls to the data pipeline

STEP 2 - USE A VERSION CONTROL SYSTEM

There are many processing steps that turn raw data into useful information for stakeholders. To be valuable, data must progress through these steps, linked together in some way, with the ultimate goal of producing a data-analytics output. Data may be preprocessed, cleaned, checked, transformed, combined, analyzed, and reported. Conceptually, the data-analysis pipeline is a set of stages implemented using a variety of tools including ETL tools, data science tools, self-service data prep tools, reporting tools, visualization tools and more. The stages may be executed serially, but many stages can be parallelized. The pipeline is deterministic because the pipeline stages are defined by scripts, source code, algorithms, html, configuration files, parameter files, containers and other files. All of these items are *essentially* just code. Code controls the entire data-analytics pipeline from end to end in a reproducible fashion.

The artifacts (files) that make this reproducibility possible are usually subject to continuous improvement. Like other software projects, the source files associated with the data pipeline should be maintained in a version control (source control) system such as Git. A version control tool helps teams of individuals organize and manage the changes and revisions to code. It also keeps code in a known repository and facilitates disaster recovery. However, the most important benefit of version control relates to a process change that it facilitates. It allows data-analytics team members to *branch and merge*.



STEP 3 - BRANCH AND MERGE

In a typical software project, developers are continuously updating various code source files. If a developer wants to work on a feature, he or she pulls a copy of all relevant code from the version control tool and starts to develop changes on a local copy. This local copy is called a *branch*. This approach can help data-analytics teams maintain many coding changes to the data-analytics pipeline in parallel. When the changes to a branch are complete and tested, the code from the branch is *merged* back into the trunk, where the code came from.

Branching and merging can be a major productivity boost for data analytics because it allows teams to make changes to the same source code files in parallel without slowing each other down. Each individual team member has control of his or her work environment. They can run their own tests, make changes, take risks and experiment. If they wish, they can discard their changes and start over. Another key to allowing team members to work well in parallel relates to providing them with an isolated machine environment.

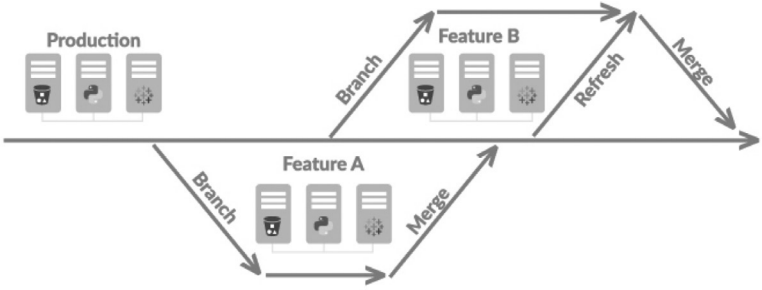
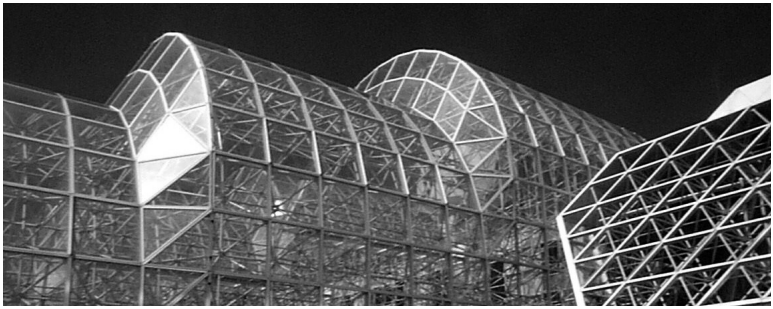


Figure 5: Branching and merging enables parallel development in data analytics.

STEP 4 - USE MULTIPLE ENVIRONMENTS

Every data-analytics team member has their own development tools on their own laptop. Version control tools allow team members to work on their own private copy of the source code while still staying coordinated with the rest of the team. In data analytics, a team member can't be productive unless they also have a copy of the data that they need. Most use cases can be covered in less than a Terabyte (TB). Historically, disk space has been prohibitively expensive, but today, at less than $25 per TB per month (cloud storage), costs are now less significant than the opportunity cost of a team member's time. If the data set is still too large, then a team member can take only the subset of data that is needed. Often the team member only needs a representative copy of the data for testing or developing one set of features.

When many team members work on the production database, it can lead to conflicts. A database engineer changing a schema may break reports. A data scientist developing a new model might get confused as new data flows in. Giving team members their own *Environment* isolates the rest of the organization from being impacted by their work.



STEP 5 - REUSE & CONTAINERIZE

Another productivity boosting method for teams is the ability to reuse and containerize code. Each middle step in the data-analytics pipeline receives output from a prior stage and provides input to the next stage. It is cumbersome to work with an entire data-analytics pipeline as one monolith, so it is common to break it down into smaller components. It's easiest for other team members to reuse smaller components if they can be segmented or containerized. One popular container technology is Docker.

Some steps in the data-analytics pipeline are messy and complicated. For example, one operation might call a custom tool, run a python script, use FTP and other specialized logic. This operation might be hard to set up (because it requires a specific set of tools) and difficult to create (because it requires a specific skill set). This scenario is another common use case for creating a container. Once the code is placed in a container, it is much easier to use by other programmers who aren't familiar with the custom tools inside the container but know how to use the container's external interfaces. It is also easier to deploy that code to each environment.

STEP 6 - PARAMETERIZE YOUR PROCESSING

There are cases when the data-analytic pipeline needs to be flexible enough to incorporate different run-time conditions. Which version of the raw data should be used? Is the data directed to production or testing? Should records be filtered according to some criterion (such as private health care data)? Should a specific set of processing steps in the workflow be included or not? To increase development velocity, these options need to be built into the pipeline. A robust pipeline design will allow the engineer or analyst to invoke or specify these options using parameters. In software development, a parameter is some information (e.g. a name, a number, an option) that is passed to a program that affects the way that it executes. If the data-analytic pipeline is designed with the right flexibility, it will be ready to accommodate different run-time circumstances.

For example, imagine a pharmaceutical company that obtains prescription data from a 3rd party company. The data is incomplete, so the data producer uses algorithms to fill in those gaps. In the course of improving their product, the data producer develops a different algorithm to the fill in the gaps. The data has the same shape (rows and columns), but certain fields are modified using the new algorithm. With the correct built-in parameters, an engineer or analyst can easily build a parallel data mart with the new algorithm and have both the old and new versions accessible through a parameter change.

STEP 7: WORK WITHOUT FEAR OR HEROISM

Many data analytics professionals live in fear. In data analytics there are two common ways to be professionally embarrassed (or get fired):

• Allow poor quality data to reach users

• Deploy changes that break production systems

Data engineers, scientists and analysts spend an excessive amount of time and energy working to avoid these disastrous scenarios. They attempt "heroism" — working weekends. They do a lot of hoping and praying. They devise creative ways to avoid overcommitting. The problem is that heroic efforts are eventually overcome by circumstances. Without the right controls in place, a problem will slip through and bring the company's critical analytics to a halt.

The DataOps enterprise puts the right set of tools and processes in place to enable data and new analytics to be deployed with a high level of quality. When an organization implements DataOps, engineers, scientists and analysts can relax because quality is assured. They can *Work Without Fear or Heroism*. DataOps accomplishes this by optimizing two key workflows.

The Value Pipeline

Data analytics seeks to extract value from data. We call this the Value Pipeline. The diagram below shows the Value Pipeline progressing horizontally from left to right. Data enters the pipeline and moves into production processing. Production is generally a series of stages: access, transforms, models, visualizations, and reports. When data exits the pipeline, in the form of useful analytics, value is created for the organization. DataOps utilizes toolchain workflow automation to optimize operational efficiency in the Value Pipeline. Data in the Value Pipeline is updated on a continuous basis, but code is kept constant. Step 2 in the seven steps of implementing DataOps — using version control — serves as the foundation for controlling the code deployed.

As mentioned above, the worst possible outcome is for poor quality data to enter the Value Pipeline. DataOps prevents this by implementing data tests (step 1). Inspired by the statistical process control in a manufacturing workflow, data tests ensure that data values lay within an acceptable statistical range. Data tests validate data values at the inputs and outputs of each processing stage in the pipeline. For example, a US phone number should be ten digits. Any other value is incorrect or requires normalization.

Once data tests are in place, they work 24x7 to guarantee the integrity of the Value Pipeline. Quality becomes *literally* built in. If anomalous data flows through the pipeline, the data tests catch it and take action — in most cases this means firing off an alert to the data analytics team who can then investigate. The tests can even, in the spirit of auto manufacturing, "stop the line." Statistical process control eliminates the need to worry about what might happen. With the right data tests in place, the data analytics team can *Work Without Fear or Heroism*. This frees DataOps engineers to focus on their other major responsibility — the Innovation Pipeline.

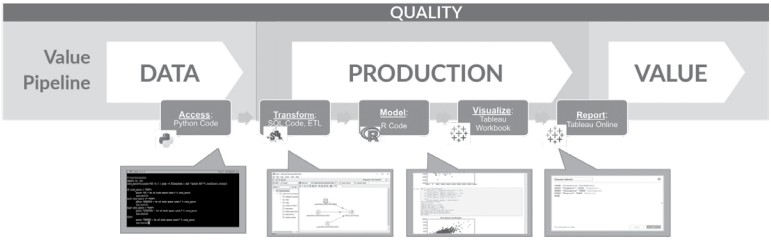


Figure 6: The value pipeline

The Innovation Pipeline

The Innovation Pipeline seeks to improve analytics by implementing new ideas that yield analytic insights. As the diagram illustrates, a new feature undergoes development before it can be deployed to production systems. The Innovation Pipeline creates a feedback loop. Innovation spurs new questions and ideas for enhanced analytics. This requires more development leading to additional insight and innovation. During the development of new features, code changes, but data is kept constant. Keeping data static prevents changes in data from being falsely attributed to the impact of the new algorithm. A fixed data set can be set-up when creating a development environment — step 4 in the seven steps of implementing DataOps.

DataOps implements continuous deployment of new ideas by automating the workflow for building and deploying new analytics. It reduces the overall cycle time of turning ideas into innovation. While doing this, the development team must avoid introducing new analytics that break production. The DataOps enterprise uses logic tests (step 1) to validate new code before it is deployed. Logic tests ensure that data matches business assumptions. For example, a field that identifies a customer should match an existing entry in a customer dimension table. A mismatch should trigger some type of follow-up.

With logic tests in place, the development pipeline can be automated for continuous deployment, simplifying the release of new enhancements and enabling the data analytics team to focus on the next valuable feature. With DataOps the dev team can deploy without worrying about breaking the production systems — they can Work Without Fear or Heroism. This is a key characteristic of a fulfilled, productive team.

The Value-Innovation Pipeline

In real world data analytics, the Value Pipeline and Innovation Pipeline are not separate. The same team is often responsible for both. The same assets are leveraged. Events in one affect the other.
The two workflows are shown combined into the Value-Innovation Pipeline in the figure below.
The Value-Innovation Pipeline captures the inter- play between development and production and between data and code. DataOps breaks down this

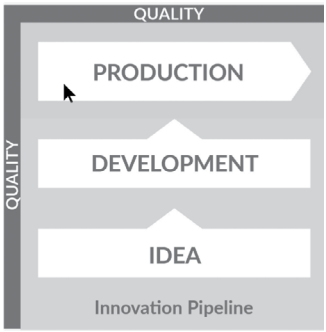


Figure 7: The innovation pipeline

barrier so that cycle time, quality and creativity can all be improved. The DataOps enterprise masters the orchestration of data to production and the deployment of new features both while maintaining impeccable quality. Reduced cycle time enables DataOps engineers to impact the organization in highly visible ways. Improved quality enables the team to move forward with confidence. DataOps speeds the extraction of value from data and improves the velocity of new development while ensuring the quality of data and code in production systems. With confidence in the Value-Innovation pipeline that stems from DataOps, the data analytics team avoids the anxiety and over-caution that characterizes a non-DataOps enterprise. Work Without Fear or Heroism!

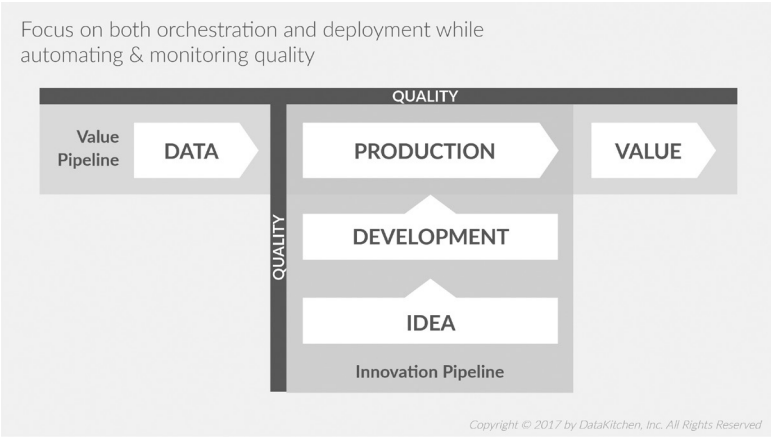


Figure 8: The Value and Innovation Pipelines illustrate how new analytics are introduced into data operations.

DataOps is NOT Just DevOps for Data

One common misconception about DataOps is that it is just DevOps *applied to* data analytics. While a little semantically misleading, the name "DataOps" has one positive attribute. It communicates that data analytics can achieve what software development attained with DevOps. That is to say, DataOps can yield an order of magnitude improvement in quality and cycle time when data teams utilize new tools and methodologies. The specific ways that DataOps achieves these gains reflect the unique people, processes and tools characteristic of data teams (versus software development teams using DevOps). Here's our in-depth take on both the pronounced and subtle differences between DataOps and DevOps.

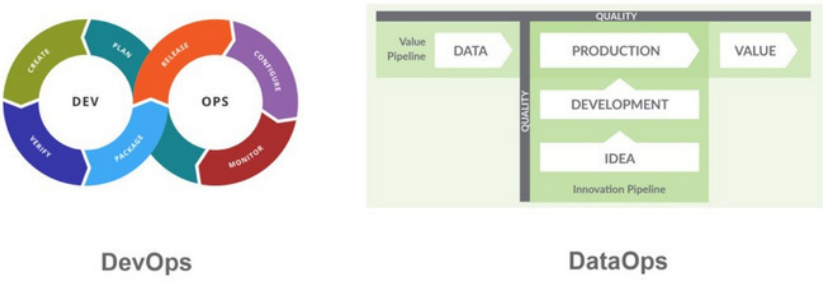


Figure 1: DevOps is often depicted as an infinite loop, while DataOps is illustrated as intersecting Value and Innovation Pipelines

The Intellectual Heritage of DataOps

DevOps is an approach to software development that accelerates the build lifecycle (formerly known as release engineering) using automation. DevOps focuses on continuous integration and continuous delivery of software by leveraging on-demand IT resources (infrastructure as code) and by automating integration, test and deployment of code. This merging of software development and IT operations ("DEVelopment" and "OPerationS") reduces time to deployment, decreases time to market, minimizes defects, and shortens the time required to resolve issues.

Using DevOps, leading companies have been able to reduce their software release cycle time from months to (literally) seconds. This has enabled them to grow and lead in fast-paced, emerging markets. Companies like Google, Amazon and many others now release software many times per day. By improving the quality and cycle time of code releases, DevOps deserves a lot of credit for these companies' success.

Optimizing code builds and delivery is only one piece of the larger puzzle for data analytics. DataOps seeks to reduce the end-to-end cycle time of data analytics, from the origin of ideas to the literal creation of charts, graphs and models that create value. The data lifecycle relies upon people in addition to tools. For DataOps to be effective, it must manage collaboration and innovation. To this end, DataOps introduces Agile Development into data analytics so that data teams and users work together more efficiently and effectively.

In Agile Development, the data team publishes new or updated analytics in short increments called “sprints.” With innovation occurring in rapid intervals, the team can continuously re-assess its priorities and more easily adapt to evolving requirements. This type of responsiveness is impossible using a Waterfall project management methodology which locks a team into a long development cycle with one “big-bang” deliverable at the end.

Studies show that Agile software development projects complete faster and with fewer defects when Agile Development replaces the traditional Waterfall sequential methodology. The Agile methodology is particularly effective in environments where requirements are quickly evolving — a situation well known to data analytics professionals. In a DataOps setting, Agile methods enable organizations to respond quickly to customer requirements and accelerate time to value.

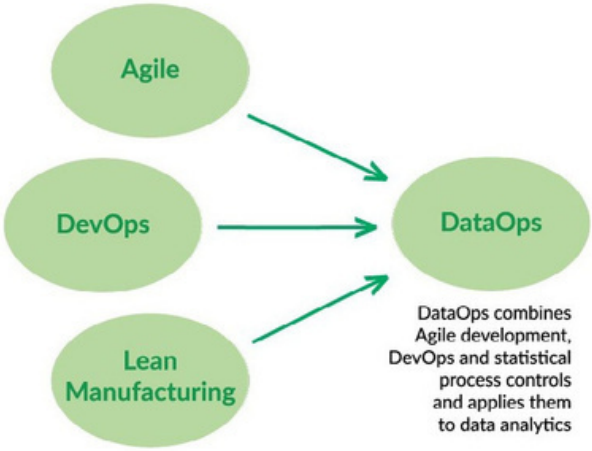


Figure 2: The intellectual heritage of DataOps.

Agile development and DevOps add significant value to data analytics, but there is one more major component to DataOps. Whereas Agile and DevOps relate to analytics development and deployment, data analytics also manages and orchestrates a data pipeline. Data continuously enters on one side of the pipeline, progresses through a series of steps and exits in the form of reports, models and views. The data pipeline is the “operations” side of data analytics. It is helpful to conceptualize the data pipeline as a manufacturing line where quality, efficiency, constraints and uptime must be managed. To fully embrace this manufacturing mindset, we call this pipeline the *“data factory.”*

In DataOps, the flow of data through operations is an important area of focus. DataOps orchestrates, monitors and manages the data factory. One particularly powerful lean-manufacturing tool is statistical process control (SPC). SPC measures and monitors data and

operational characteristics of the data pipeline, ensuring that statistics remain within acceptable ranges. When SPC is applied to data analytics, it leads to remarkable improvements in efficiency, quality and transparency. With SPC in place, the data flowing through the operational system is verified to be working. If an anomaly occurs, the data analytics team will be the first to know, through an automated alert.

While the name “DataOps” implies that it borrows most heavily from DevOps, it is all three of these methodologies - Agile, DevOps and statistical process control — that comprise the intellectual heritage of DataOps. Agile governs analytics development, DevOps optimizes code verification, builds and delivery of new analytics and SPC orchestrates and monitors the data factory. Figure 3 illustrates how Agile, DevOps and statistical process control flow into DataOps.

You can view DataOps in the context of a century-long evolution of ideas that improve how people manage complex systems. It started with pioneers like Deming and statistical process control — gradually these ideas crossed into the technology ~~space in~~ the form of Agile, DevOps and now, DataOps.

DevOps vs. DataOps — The Human Factor

As mentioned above, DataOps is as much about managing people as it is about tools. One subtle difference between DataOps and DevOps relates to the needs and preferences of stakeholders.

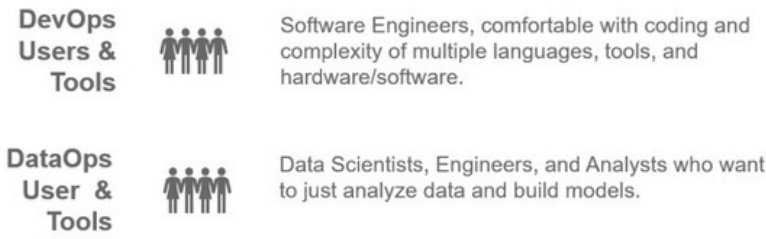


Figure 3: DataOps and DevOps users have different mindsets

DevOps was created to serve the needs of software developers. Dev engineers love coding and embrace technology. The requirement to learn a new language or deploy a new tool is an opportunity, not a hassle. They take a professional interest in all the minute details of code creation, integration and deployment. DevOps embraces complexity.

DataOps users are often the opposite of that. They are data scientists or analysts who are focused on building and deploying models and visualizations. Scientists and analysts are typically not as technically savvy as engineers. They focus on domain expertise. They are interested in getting models to be more predictive or deciding how to best visually render data. The technology used to create these models and visualizations is just a means to an end. Data professionals are happiest using one or two tools — anything beyond that adds unwelcome complexity. In extreme cases, the complexity grows beyond their ability to manage it. DataOps accepts that data professionals live in a multi-tool, heterogeneous world and it seeks to make that world more manageable for them.

DevOps vs. DataOps - Process Differences

We can begin to understand the unique complexity facing data professionals by looking at data analytics development and lifecycle processes. We find that data analytics professionals face challenges both similar and unique relative to software developers.

The DevOps lifecycle is commonly illustrated using a diagram in the shape of an infinite symbol — See Figure 4. The end of the cycle ("plan") feeds back to the beginning ("create"), and the process iterates indefinitely.

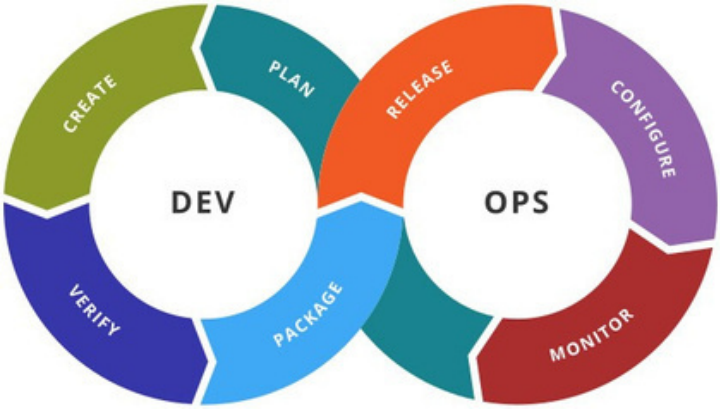


Figure 4: The DevOps lifecycle is often depicted as an infinite loop

The DataOps lifecycle shares these iterative properties, but an important difference is that DataOps consists of two active and intersecting pipelines (Figure 5). The data factory, described above, is one pipeline. The other pipeline governs how the data factory is updated — the creation and deployment of new analytics into the data pipeline.

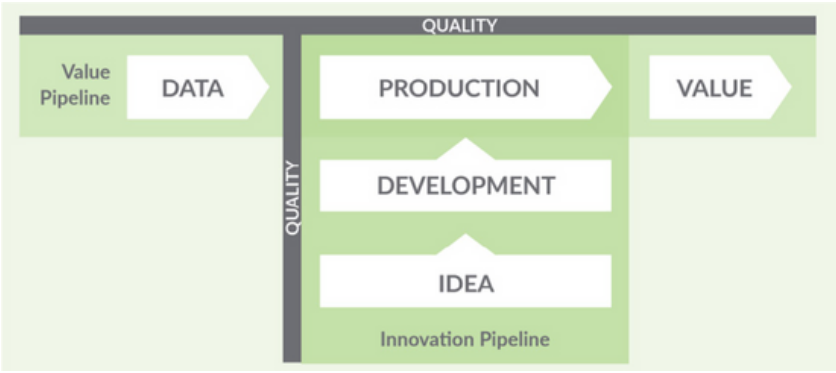


Figure 5: The DataOps lifecycle — the Value and Innovation Pipelines

DevOps vs. DataOps — Development and Deployment Processes

DataOps builds upon the DevOps development model. As shown in Figure 14, the DevOps process flow includes a series of steps that are common to software development projects:

- **Develop** – create/modify an application
- **Build** – assemble application components
- **Test** – verify the application in a test environment
- **Deploy** – transition code into production
- **Run** – execute the application

DevOps introduces two foundational concepts: Continuous Integration (CI) and Continuous Deployment (CD). CI continuously builds, integrates and tests new code in a development environment. Build and test are automated so they can occur rapidly and repeatedly. This allows issues to be identified and resolved quickly. Figure 14 illustrates how CI encompasses
the build and test process stages of DevOps.

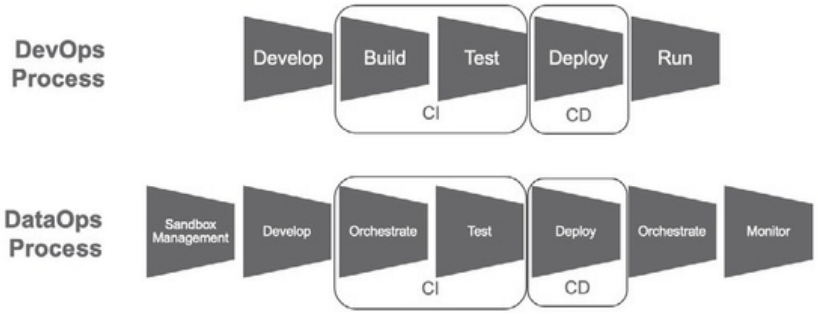


Figure 6: Comparing the DataOps and DevOps processes

CD is an automated approach to deploying or delivering software. Once an application passes all qualification tests, DevOps deploys it into production. Together CI and CD resolve the main constraint hampering Agile development. Before DevOps, Agile created a rapid succession of updates and innovations that would stall in a manual integration and deploy- ment process. With automated CI and CD, DevOps has enabled companies to update their software many times per day.

The Data Journey Is First In DataOps

When your data team is in crisis from errors in production, complaining customers, and uncaring data providers, we all wish we could be unshaken as the Buddha. Our recent survey showed that 97% of data engineers report experiencing burnout in their day-to-day jobs. Perhaps we could just chill out in those stressful situations and “let go,” as the Buddha suggests. The spiritual benefits of letting go may be profound, but finding and fixing the problem at its root is, as Samuel Florman writes, “existential joy.”

Finding problems before your customers know they exist helps your team's happiness, productivity, customer trust, and customer data success. Given the complicated distributed systems we use to get value from data and the diversity of data, we need a simplifying framework. That idea is the Data Journey. In an era where data drives innovation and growth, it's paramount that data leaders and engineers understand and monitor the five pillars of a Data Journey. The key to success is the capability to understand and monitor the health, status, and performance of your data, data tools, pipeline, and infrastructure, both at a macro and micro level. Failures on the Data Journey cost organizations millions of dollars.

Figure 7: Observing your Data Journey is DataOps first step. It is a problem to find an issue across all your complex, multi-tool data toolchain, then down into the raw data, integrated data, and software/hardware that runs it all.

Data Journey First DataOps requires a deep and continuous understanding of your production data estate. It provides a dynamic understanding of how your data flows, transforms, gets enriched, and is consumed. It allows you to trust through active verification. By observing Data Journeys that make up your Value Pipeline, you can detect problems early, streamline your processes, and lower embarrassing errors in production.

Data quality validation testing of data at rest and data in use is critical. Checking data at rest involves looking at syntactic attributes such as freshness, distribution, volume, schema, and lineage. You also need robust testing and evaluation processes throughout the last mile of the Data Journey when data is integrated and used in tools such as predictive models of visualization tools.

Figure 8: Problems can occur anywhere in the Journey data takes form source to value. Testing in key.

The Duality of Testing in DataOps

Tests in DataOps have a role in both the Value and Innovation Pipelines. In the Value Pipeline, tests monitor the data values flowing through the data factory to catch anomalies or flag data values outside statistical norms. In the Innovation Pipeline, tests validate new analytics before deploying them.

In DataOps, tests target either data or code. Figure 17 below illustrates this concept. Data that flows through the Value Pipeline is variable and subject to statistical process control and monitoring. Tests target the data which is continuously changing. Analytics in the Value Pipeline, on the other hand, are fixed and change only using a formal release process. In the Value Pipeline, analytics are revision controlled to minimize any disruptions in service that could affect the data factory.

In the Innovation Pipeline code is variable and data is fixed. The analytics are revised and updated until complete. Once the sandbox (analytics development environment) is set-up, the data doesn't usually change. In the Innovation Pipeline, tests target the code (analytics), not the data. All tests must pass before promoting (merging) new code into production. A good test suite serves as an automated form of impact analysis that runs on any and every code change before deployment.

Some tests are aimed at both data and code. For example, a test that makes sure that a database has the right number of rows helps your data and code work together. Ultimately both data tests and code tests need to come together in an integrated pipeline as shown in Figure 9. DataOps enables code and data tests to work together so all around quality remains high.

Figure 9: In DataOps, analytics quality is a function of data and code testing

DataOps Complexity — Sandbox Management

When an engineer joins a software development team, one of their first steps is to create a "sandbox." A sandbox is an isolated development environment where the engineer can write and test new application features, without impacting teammates who are developing other features in parallel. Sandbox creation in software development is typically straightforward the engineer usually receives a bunch of scripts from teammates and can configure a sand- box in a day or two. This is the typical mindset of a team using DevOps.

Sandboxes in data analytics are often more challenging from a tools and data perspective. First of all, data teams collectively tend to use many more tools than typical software dev teams. There are literally thousands of tools, languages and vendors for data engineering, data science, BI, data visualization, and governance. Without the centralization that is characteristic of most software development teams, data teams tend to naturally diverge with different tools and data islands scattered across the enterprise.

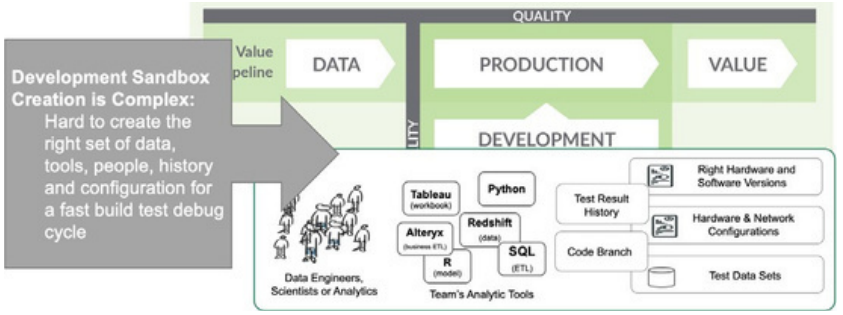


Figure 10: A "sandbox" is an isolated development environment where the data professional can write and test new analytics without impacting teammates.

DataOps Complexity - Test Data Management

In order to create a dev environment for analytics, you have to create a copy of the data factory. This requires the data professional to replicate data which may have security, governance or licensing restrictions. It may be impractical or expensive to copy the entire data set, so some thought and care is required to construct a representative data set. Once a multi-terabyte data set is sampled or filtered, it may have to be cleaned or redacted (have sensitive information removed). The data also requires infrastructure which may not be easy to replicate due to technical obstacles or license restrictions.

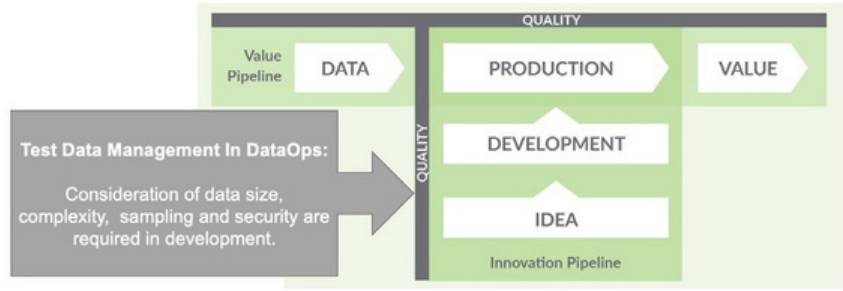


Figure 11: The concept of test data management is a first order problem in DataOps.

The concept of test data management is a first order problem in DataOps whereas in most DevOps environments, it is an afterthought. To accelerate analytics development, DataOps has to automate the creation of development environments with the needed data, software, hardware and libraries so innovation keeps pace with Agile iterations.

DataOps Connects Organizations

DevOps strives to help development and operations (information technology) teams work together in an integrated fashion. In DataOps, this concept is depicted in Figure 20. The development team are the analysts, scientists, engineers, architects and others who create data warehouses and analytics.

In data analytics, the operations team supports and monitors the data pipeline. This can be IT, but it also includes customers — the users who create and consume analytics. DataOps brings these groups together so they can work together more closely.

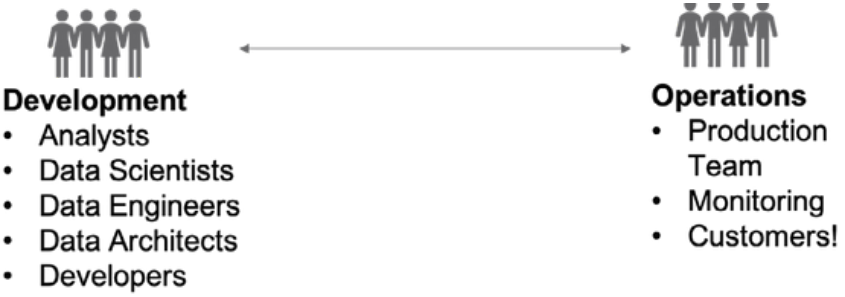


Figure 12: DataOps combines data analytics development and data operations

Freedom vs. Centralization

DataOps also brings the organization together across another dimension. A great deal of data analytics development occurs in remote corners of the enterprise, close to business units, using self-service tools like Tableau, Alteryx, or Excel. These local teams, engaged in decentralized, distributed analytics creation play an essential role in delivering innovation to users. Empowering these pockets of creativity maintains the enterprise's competitiveness, but frankly, a lack of top-down control can lead to unmanaged chaos.

Centralizing analytics development under the control of one group, such as IT, enables the organization to standardize metrics, control data quality, enforce security and governance, and eliminate islands of data. The issue is that too much centralization chokes creativity.

A critical benefit of DataOps is its ability to harmonize the back-and-forth between the decentralized and centralized development of data analytics — the tension between centralization and freedom. In a DataOps enterprise, new analytics originate and undergo refinement in the local pockets of innovation. When an idea proves helpful or worthy of wider distribution, it is promoted to a centralized development group that can more efficiently and robustly implement it at scale.

DataOps brings localized and centralized development together, enabling organizations to reap the efficiencies of centralization while preserving localized development — the tip of the innovation spear. DataOps brings the enterprise together across four dimensions, as shown in Figure 13.

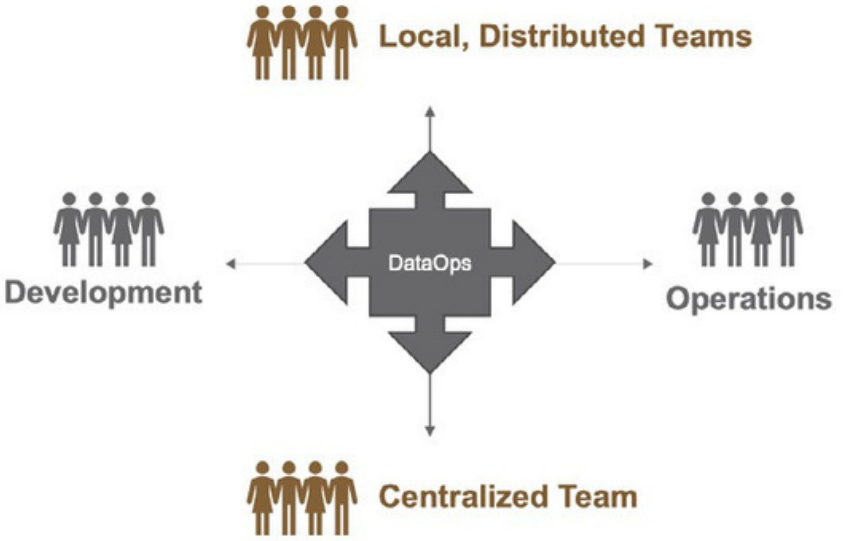
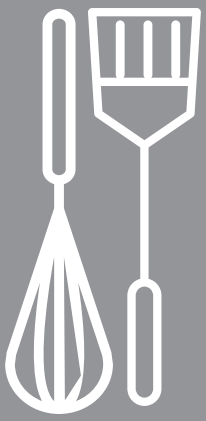


Figure 13: Hub vs Spoke Teams



DataOps Chili Mole

by Gil Benghiat

INGREDIENTS AND TOOLS

- 1 pound ground beef (90% lean or leaner)
- 1 diced onion
- 2 diced Jalapeño or 2 Serrano or 1 habanero pepper • 2 pounds frozen sweet corn
- 28 oz can crushed tomatoes
- 15.5 oz can of black beans, NOT drained
- 15.5 oz can of black beans, drained
- 15.5 oz can of red beans, drained
- 15.5 oz can of white beans, drained
- 7 cloves of crushed garlic
- ½ cup sugar
- 3 ½ table spoons of unsweetened cocoa powder
- 3 table spoons of chili powder
- 1 teaspoon of salt

INSTRUCTIONS

1. Crock Pot: Combine all ingredients and cook on high for 5-8 hours. Stir occasionally.
2. Stove Top: Combine ground beef, onion, and pepper. Cook on medium high until beef is cooked through. Add the remaining ingredients and cook on low-simmer for 1-2 hours. Stir occasionally.
3. For vegan chili: Substitute 5 tablespoons of canola oil for the ground beef.
4. Serve with rice.

Data Journey First DataOps

Putting Problems in Your Data Estate at the Forefront

Welcome to the high-octane world of DataOps, a powerhouse that turbocharges data analytics development and management. This innovative approach merges the agility of Agile Development, the stability of DevOps, and the meticulousness of Statistical Process Controls to enable a dynamic, enriched, and nimble data ecosystem that is truly remarkable.

Historically, automation has taken center stage in the theater of DataOps. But as we navigate ever-evolving seas of data, it's high time we reset our course. We must adopt a pioneering and exceptionally effective strategy—where we prioritize the intricacies and nuances of the 'Data Journey' even before we approach automation.

Welcome to the dawn of a transformative approach—which we proudly call 'Data Journey First DataOps.' This is not just an approach; it's a revolution in which every data, tool, server, and step becomes part of a meaningful story, enhancing our data initiatives' overall value, impact, and trust.

The Why and How of 'Data Journey First DataOps'

Let's start with the 'why.' Businesses today are under more pressure than ever. With growing data demands and shrinking timelines, data analytic teams need value delivered quickly without radically altering their established systems. The initial heavy lifting often involved in full DataOps automation might be too time-consuming and disruptive.

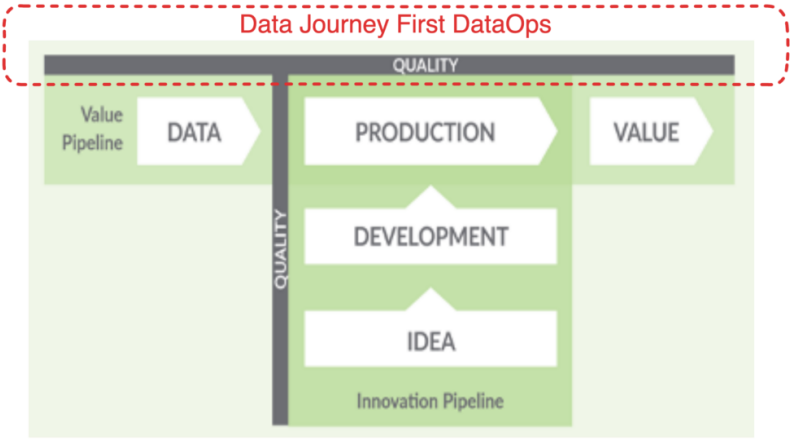
Lots Of Blame And Shame	Teams are panicked when customers find problems and spend time, without a shared context, trying to determine who is responsible.
Data Teams Already Have A Ton of Things To Get Done.	Teams are already busy and stressed and need to meet customer expectations.
Teams Did Not Build Current Architecture For Rapid And Low-Risk Changes Those Systems	Teams have complicated in-place data architectures and tools and fear changes to what is already running. Any change to production takes time because a lack of automation is hazardous.
Constant Data And Tool Errors In Production	Teams cannot see across all tools, pipelines, jobs, processes, datasets, and people. As a result, their customers tell them when there is a problem.
No Time For Data Validation Testing	Teams must learn what, where, and how to check raw, integrated, or 'data in use' to ensure the correct outputs.

Obstacles to Automation in Your Data Operations

Enter ‘Data Journey First DataOps.’ The idea here is to focus first on understanding and observing the journey that data takes through your production environment – from ingestion to processing to delivering actionable insights. This monitoring process identifies data errors, tool problems, and timing issues, enabling a quick win for your DataOps implementation by driving immediate improvements. Lowering production errors increases the reliability of your data and gives your team more time to focus on automation. And all this has to happen in days, not months. There are only five pillars that define what teams need to get value quickly, with little work, and without significantly changing what they already have working in production.

Data Journey First DataOps: An Overview

The first step in DataOps isn’t implementing DataOps automation techniques or deploying cutting-edge data management platforms. Instead, it focuses on a humble yet pivotal element – lowering errors in your production Data Journey. The term ‘Data Journey First DataOps’ encapsulates this philosophy.



DataOps Concept: Data Journey at the Forefront

Observing Production Data Journeys:

This step involves a deep and continuous understanding of your production data estate. Actively monitoring the entire Data Journey goes beyond the static Data Lineage of where your data originates and where it lands. It adds the dynamic understanding of how your data flows, transforms, gets enriched, and is consumed. It allows you to trust through active verification. By observing these Data Journeys comprehensively and actively, you can detect problems early, streamline your processes, and make informed decisions.

Lower Errors in Production:

A keen focus on the Data Journey helps to pinpoint errors, inconsistencies, and delays in your production data, tools, and deliverables. Data engineers can lower error rates in production by identifying these issues early and resolving them swiftly, improving the overall quality of data and, thereby, the reliability of insights derived from the data.

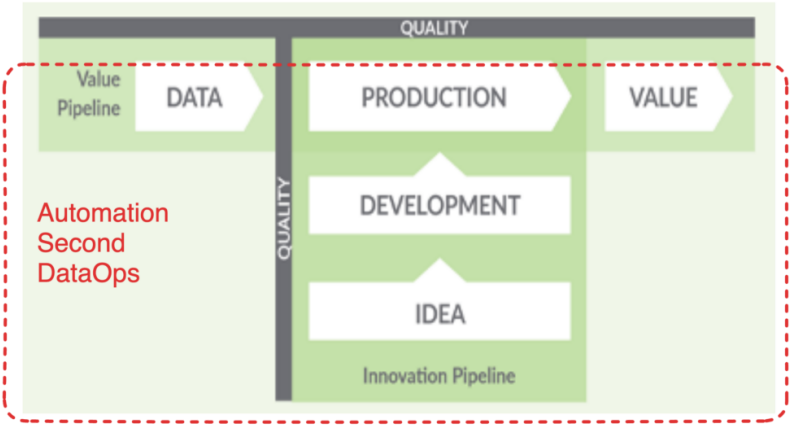
Creating Time for Automation in Step Two:

By prioritizing your Data Journey, you allow your team to address the fundamental issues affecting your data estate before moving to the second step – automation. This sequence ensures your team can leverage automation more effectively when the time comes.

Automation Second DataOps: The Next Step

Once we've established stable, fully observed Data Journeys, we can move onto the second stage: 'Automating DataOps.' The first, second, and third pipelines of DataOps automation come into play, designed to increase productivity and reduce cycle times.

1. Meta-Orchestrate Your Production Pipelines to organize your entire data analytics toolchain.
2. Automate Your Deployment Pipelines for fast and fearless development.
3. Automate Your Environment Pipelines for reusability, security, and developer speed.



DataOps Automation Second Concept: Environments, Deployment, and Production.

Automated DataOps significantly enhance cycle times. It accelerates your ability to make small, low-risk changes to production data systems, enables quicker decision-making, and boosts productivity by freeing data teams to focus on learning through rapid iterations. But remember, this step can be practical only when it builds upon the groundwork laid by 'Data Journey First DataOps.'

Benefits Of Data Journey First DataOps

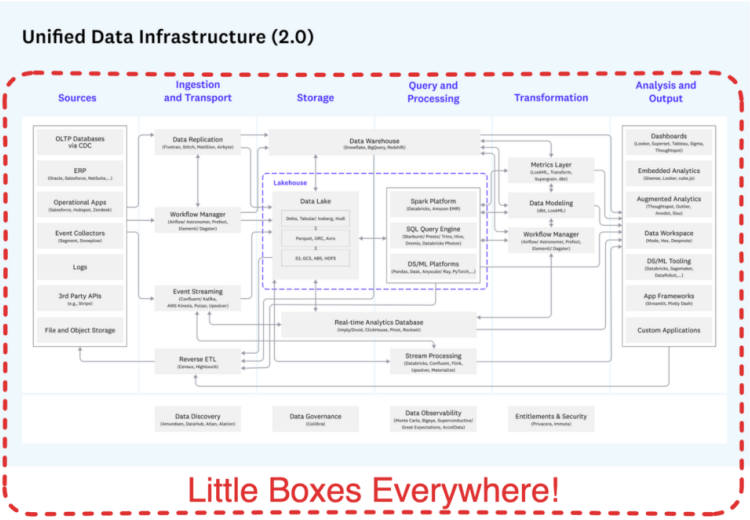
Choosing a Data Journey first approach to DataOps is essential because:

It's easy and fast to get done.

Most data teams are often overwhelmed with tasks and have pressing customer needs. They need valuable, quickly implemented DataOps solutions without drastically changing what they already have. Implementing a Data Journey is the first step.

It minimizes disruption of your existing 'as-built' data estate.

As-built data and analytic systems have many steps to prepare for full-fledged DataOps automation. Teams have rushed to get a production solution in place without adequate investment in deployment pipelines, environment management, and orchestration of their production pipelines. These gaps result in the need for solutions that address the primary pain points without causing significant disruption.



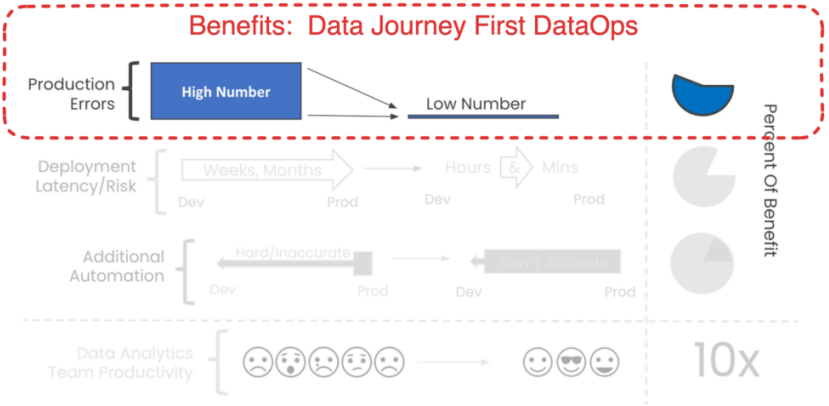
Aren't you tired of seeing architectural diagrams composed of little boxes everywhere? (credit a16z)

No analytic customer ever said, 'I want more data errors' in my Data Journey!

22% of data engineers' time is spent on innovation, but 78% on errors and manual execution (Gartner 2022). An Eckerson survey found that over 79% of projects have too many errors.

Addressing this challenge delivers a big chunk of the value of adopting DataOps.

Most organizations want their first DataOps projects to deliver significant value fast with a small effort and minimal changes to existing operations. Data Journeys are fast and easy to implement.



Reducing production errors delivers a large portion of the overall value of DataOps

It lowers your team's stress to make more fundamental changes.

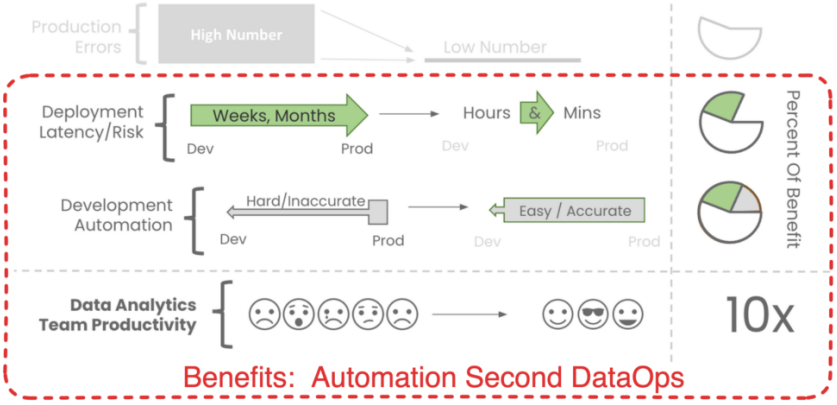
A recent survey of 700 data engineers by DataKitchen & Data.world in 2022 found that 52% of Data Engineers said errors are a significant source of burnout. Data engineers deal with hundreds of data sets and diverse customer needs. They have backlogs on their daily task lists. So they don't have time or energy to learn about each data set or customer to create robust production data validation tests. The entire Data Journey is invisible to them. They need help creating data tests and observing the entire Data Journey for success.

It lays the data validation and testing groundwork to improve deployment cycle time.

In a world of complexity, failure, and frustration, data and analytics teams need to deliver insight to their customers with no errors and a high rate of change. They must deploy small changes, new data sets, new tools, and updated code to production quickly and with low risk. From initiating a data analytics task to its completion and deployment, cycle time is a critical metric affecting an organization's ability to generate insights and make data-driven decisions. A shorter cycle time enables a more agile decision-making process and allows organizations to capitalize on emerging trends or opportunities. Testing, observing, and monitoring production Data Journeys provides testing and exception monitoring capabilities that can be used to improve deployment.

In Conclusion

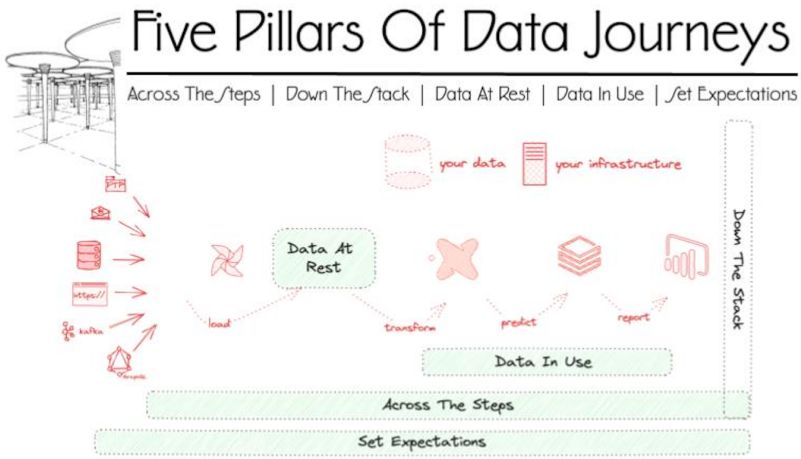
Data Teams often need more. They desire quick, valuable assistance without significantly changing their existing systems. The 'Data Journey First DataOps' approach aligns perfectly with these needs.



Automation gives a portion of the overall benefits of DataOps.

'Data Journey First DataOps' is a philosophy that acknowledges the current constraints of data and business teams and proposes a solution that offers rapid, substantial value. Whether you're 'Leading with Data Journey,' 'Prioritizing Data Journey,' or simply 'Putting Data Journey First in DataOps,' remember that the goal is to deliver quick wins today while laying the foundation for a more robust, automated data operations landscape tomorrow. The first successful DataOps implementation is all about quick wins to lower errors and creating foundational elements for the next improvement phase, automating DataOps.

Introducing The Five Pillars Of Data Journeys



> *"There are those who discover they can leave behind destructive reactions and become patient as the earth, unmoved by fires of anger or fear, unshaken as a pillar, unperturbed as a clear and quiet pool."*
> *– Gautama Buddha*

When your data team is in crisis from errors in production, complaining customers, and uncaring data providers, we all wish we could be unshaken as the Buddha. Our recent survey showed that 97% of data engineers report experiencing burnout in their day-to-day jobs. Perhaps we could just chill out in those stressful situations and "let go," as the Buddha suggests. The spiritual benefits of letting go may be profound, but finding and fixing the problem at its root is, as Samuel Florman writes, "existential joy."

Finding problems before your customers know they exist helps your team's happiness, productivity, customer trust, and customer data success. Given the complicated distributed systems we use to get value from data and the diversity of data, we need a simplifying framework. That idea is the Data Journey. In an era where data drives innovation and growth, it's paramount that data leaders and engineers understand and monitor their Data Journey's various facets. The key to success is the capability to understand and monitor the health, status, and performance of your data, data tools, pipeline, and infrastructure, both at a macro and micro level. Failures on the Data Journey cost organizations millions of dollars. Putting the Data Journey idea into five pillars is a great way to organize and share the concept. The table below gives an overview:

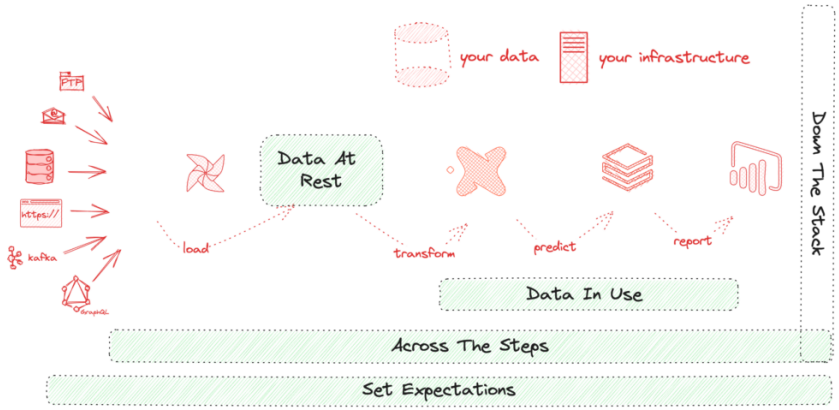
Five Pillars Of Data Journeys

Across The Steps | Down The Stack | Data At Rest | Data In Use | Set Expectations

	Across The Steps	Down The Stack	Data At Rest	Data In Use	Set Expectations
What is Checked and Observed?	Check Runs, Order Of Operations, Schedule	Monitor Metrics, Logs, And Cost	Validate Data Quality Automatically With Business Domain Tests	Test The Results Of Models, Visualization, Delivery, And Utilization	Compare Expected Against Reality, Alert, Analyze
Examples	Orchestrator Schedule, Overlapping Jobs, Delays	Error Message, Server Cpu, Disk Size, Run Cost	Schema, Freshness, Volume; Percentage Regional Sales Growth	Root Means Square Error From Model; PowerBI Dashboard Values	Order of steps, timing, data test results, costs, metrics
Value	Process Reliability	Technology Status	Data Quality	End User Experience	Incident Alerting

Table Summarizing the Five Data Journey Pillars.

Another way to look at the five pillars is to see them in the context of a typical complex data estate. You may have four steps your data takes from its source to customer use, or twenty. However, every Data Journey spans many 'little boxes' like the diagram below.



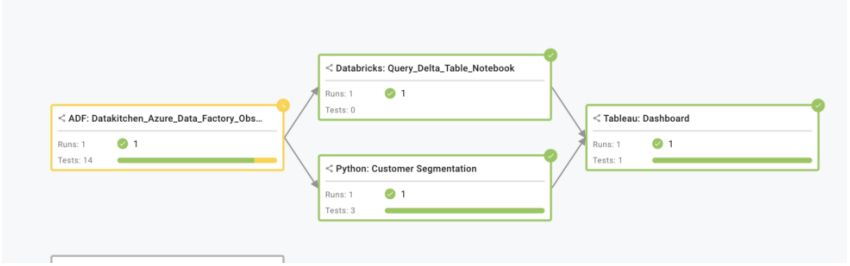
Five Pillars of Data Journeys in Operational Context.

Pillar 1. Across The Steps

> *"All happy, error-free Data Journeys are alike; each unhappy Data Journey is broken in its unique way.*
> *– Anna Karenina Author Leo Tolstoy (sort of).*

Things will break along your Data Journey. The question is, where did it happen? In our experience, the locus of those problems changes over time. Initially, the infrastructure is unstable, but then we look at our source data and find many problems. Our customers start looking at the data in dashboards and models and then find many issues. Putting the data together with other data sets is another source of errors. After data systems start to get used, changes will introduce more problems.

The critical question is where the problem is. This pillar emphasizes the need to continually monitor the execution of each process within every step data takes on its journey to your customer to ensure that the order of operations is correct, tasks execute according to schedule, and the data itself is correct. The Data Journey, in this sense, provides transparency about the status and outcomes of individual tasks, offers insights into potential bottlenecks or inefficiencies in the sequence of operations, and helps ensure that scheduled tasks are executed as planned. Consider a data pipeline orchestrated by Airflow.



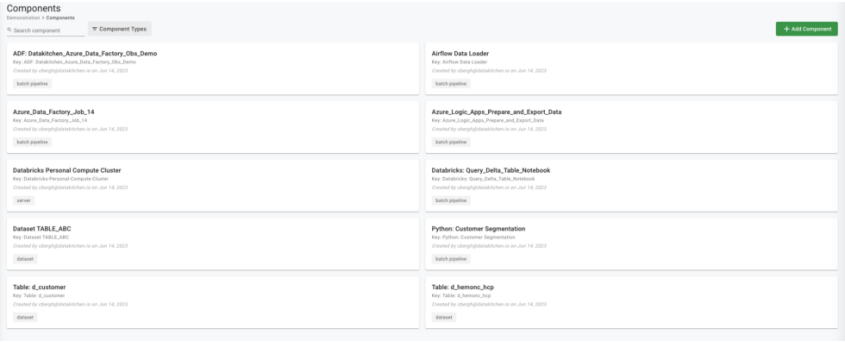
The example above shows a Data Journey across multiple tools.

Observability in this context involves monitoring the orchestrator's schedule and identifying potential issues like overlapping jobs that could cause bottlenecks or delays due to resource contention. Did the Airflow job complete before the dashboard was loaded? Was it on time? The value here is increased process reliability. With such observability, you can quickly pinpoint process issues, minimize downtime, notify downstream, and ensure a smoother, more reliable end-to-end Data Journey.

Pillar 2. Down The Stack

> *“Shrek: Data Journeys are like onions.*
> *Donkey: They stink?*
> *Shrek: No. Layers. Onions have layers. Data Journeys have layers. Do you get it?*
> *Donkey: Oh, they have layers. Oh. You know, not everybody like onions.”*
> *– Shrek Movie (?)*

Monitoring is another pillar of Data Journeys, extending down the stack. It involves tracking key metrics such as system health indicators, performance measures, and error rates and closely scrutinizing system logs to identify anomalies or errors. Moreover, cost monitoring ensures that your data operations stay within budget and that resources are used efficiently. These elements contribute to a fuller understanding of the operational landscape, enabling proactive management and issue mitigation. Going down the stack could involve checking error messages to identify faulty processes, monitoring server CPU usage to spot potential performance issues, assessing disk sizes to ensure sufficient storage capacity, and tracking run costs to ensure your operations stay within budget.



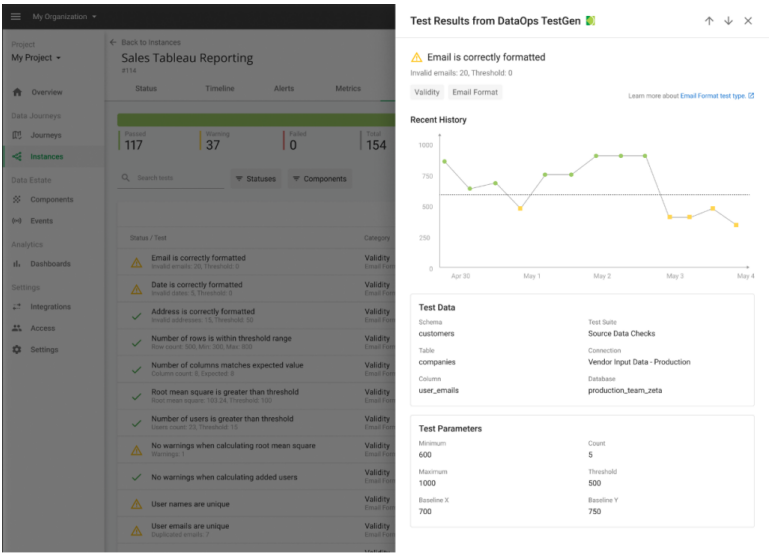
Data Journeys run on software, on servers, and with code. They can break.

The major value here is a clear and comprehensive understanding of your technology’s status. You can proactively spot and address issues before they escalate and ensure your technology stack runs smoothly and cost-effectively.

Pillar 3. Data At Rest

> *“Data don’t tell me you’re sorry ’cause you’re not / Bad data I know you’re only sorry you got caught.”*
> *– Take A Bow, Rihanna (I may have heard it wrong)*

Validating data quality at rest is critical to the overall success of any Data Journey. Using automated data validation tests, you can ensure that the data stored within your systems is accurate, complete, consistent, and relevant to the problem at hand. This pillar emphasizes the importance of implementing thorough data validation tests to mitigate the risks of erroneous analysis or decision-making based on faulty data. Checking data at rest involves looking at syntactic attributes such as freshness, distribution, volume, schema, and lineage. Start checking data at rest with a strong data profile. Then the ingestion-focused data tests can look for validations by checking incoming data schema, assessing data row counts, load data, evaluating data volume, or specific column values for anomalies.



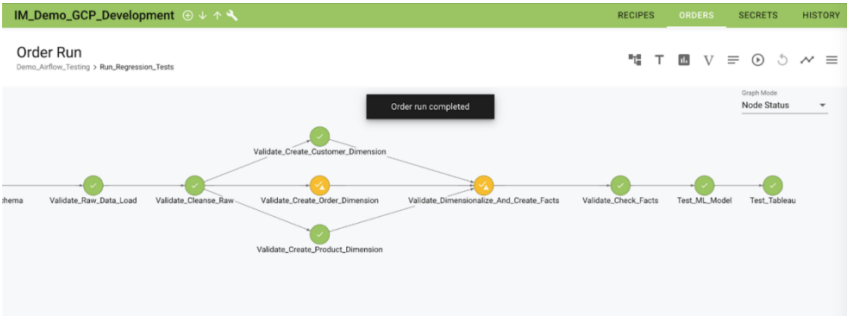
The image above shows an example ‘’data at rest’ test result.

Checking data at rest also involves looking beyond data syntax. Teams need data validation tests that are based on domain-specific or business rules that are meaningful to their organizations. These tests can rely upon historical values to determine whether data values are reasonable (or within the reasonable range). For example, a test can check the top fifty customers or suppliers. Did their values unexpectedly or unreasonably go up or down relative to historical values? What is the acceptable range? 10% 50%? Data engineers are unable to make these business judgments. They must rely on data stewarts or their business customers to ‘fill in the blank’ on various data testing rules.
The central value here is ensuring trust through data quality. By conducting these checks, you can catch data issues early, ensuring that your downstream analyses and decisions are based on high-quality data.

Pillar 4. Data In Use

> *"Truth about your data tools hurts. Maybe not as much as jumping on a bicycle with a seat missing, but it hurts." —Lt. Frank Drebin/Leslie Nielsen (maybe)*

Monitoring and testing the data to ensure its reliability continually is crucial. This action involves testing the results of data models for accuracy and relevance, evaluating the effectiveness of data visualizations, ensuring that data delivery mechanisms are operating optimally, and checking the data utilization to ensure it meets its intended purpose. This pillar underscores the need for robust testing and evaluation processes throughout the 'last mile' of the Data Journey.



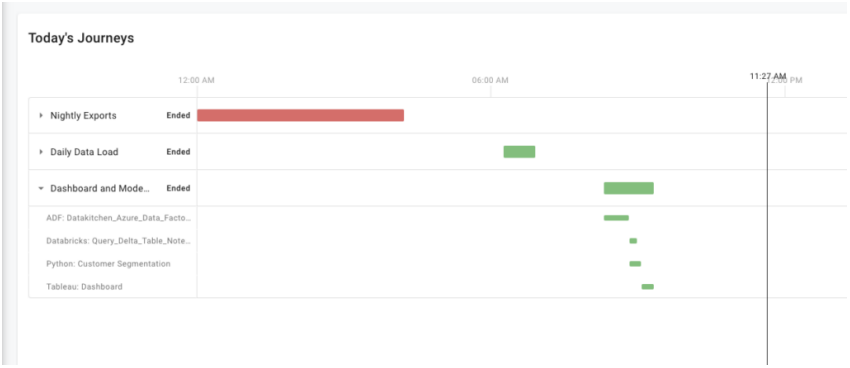
The above image shows an example custom 'data in use' test of a predictive model and API.

The value here is improved end-user experience. Conducting these tests ensures that your data products (like predictive models or visualizations) are accurate, relevant, and valuable to your end users. After all the hard work and multiple systems data took to get to your customer, isn't value the key to judging success?

Pillar 5. Set Expectations

> *"High Data Journey expectations are the key to everything."*
> *– Sam Walton (slightly modified)*

The final pillar of Data Journeys involves setting and managing expectations. A Data Journey is a collection of expectations of how your data world should be. Of course, the world never meets our expectations.
A Data Journey allows you to compare anticipated outcomes against reality, set up alert mechanisms to notify stakeholders when discrepancies arise, and analyze results to understand what led to the outcome. It emphasizes the need for a systematic approach to understanding and managing deviations from expected outcomes. Data problems often come with a 'blast radius.' For example, what reports, models, and exports are affected if an ingested file is too small? Data Journeys are the 'process lineage' that can help you find the full extent and impact of a problem and notify those who may be impacted.



A dashboard allows you to share the progress of the latest instance of your Data Journeys.

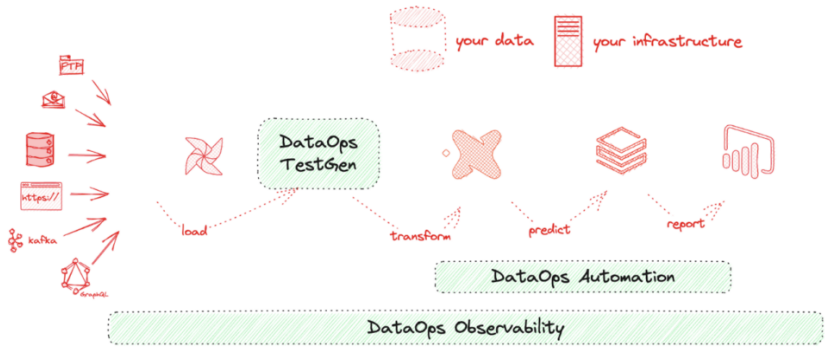
Trust building between the data team and their customer is vital. The more your data team knows about problems before they occur, the more trust your customers will have in your team. Data Journeys with incident alerting provided the bridge to build that trust.

Conclusion

The “Five Pillars of Data Journeys” outline a comprehensive approach to tracking and monitoring data across its lifecycle. Firstly, it highlights the importance of understanding the sequence and results of data operations, including regular checks, maintaining the correct order of operations, and adhering to schedules. Secondly, it underscores the necessity to monitor metrics, logs, and associated costs down the stack, ensuring the efficiency and cost-effectiveness of data operations. Thirdly, it suggests automatic data quality validation at rest through business domain tests, enhancing data integrity and reliability. Fourthly, the pillars advocate for testing the results of models, visualizations, and data utilization to validate data in use, assuring the effectiveness of data applications. Lastly, it encourages setting and comparing expectations against reality, alert systems, and in-depth analysis to maintain a robust and accurate data environment.

DataKitchen Data Journey Products

DataKitchen’s products implement the five pillars of Data Journeys. DataKitchen’s products track and monitor all levels of the data stack, from data to tools to servers to code to tests across all critical dimensions. They supply real-time statuses and alerts on start times, processing durations, test results, and infrastructure events, among other metrics. With this information, you can know if everything ran on time and without errors and immediately detect the parts that didn’t.

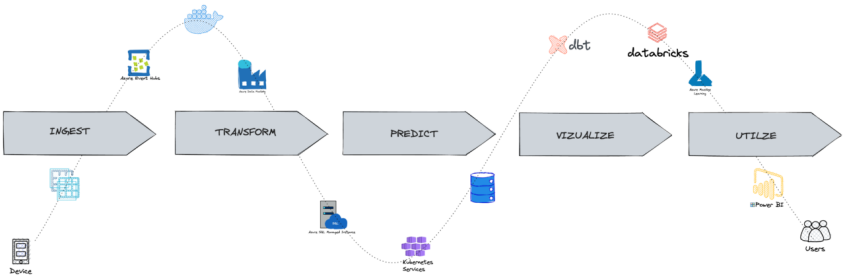


DataKitchen's Products Implement The Five Pillars Of Data Journeys

DataOps Observability provides the Data Journey abstraction, expectations, alerts, and analysis. DataOps TestGen provides in-database data testing results shared with DataOps Observability. Finally, DataOps Automation provides tool, model, and API-level testing shared with DataOps Observability.

Why the Data Journey Manifesto?

So why another manifesto in the world? Really? Why should I care?



About seven years ago, we wrote the DataOps Manifesto. We wrote the first version because, after talking with hundreds of people at the 2016 Strata Hadoop World Conference, very few easily understood what we discussed at our booth and conference session. We had been talking about “Agile Analytic Operations,” “DevOps for Data Teams,” and “Lean Manufacturing For Data,” but the concept was hard to get across and communicate. I spent much time de-categorizing DataOps: we are not discussing ETL, Data Lake, or Data Science.

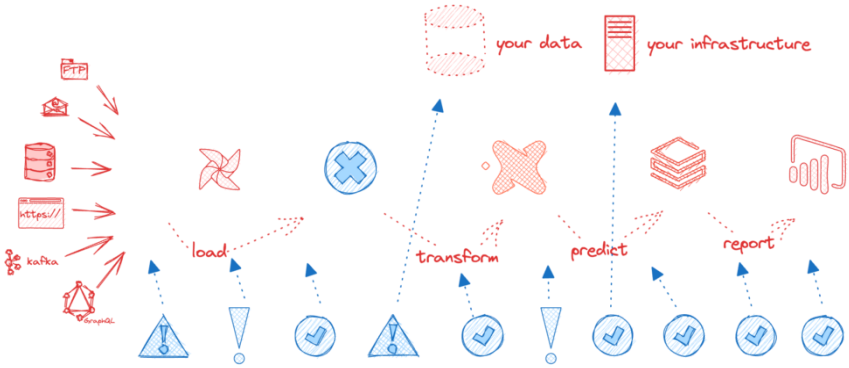
So on the flight black back, **I wrote the draft DataOps Manifesto** and sent it to some people, got some comments, and we put up a webpage. Today we have had over **20,000 signatures**, millions of page views, and copycat clones, and it is frequently used as a reference guide. For example, just a few weeks ago, Microsoft announced data fabric, and John Kerski used it to frame up the discussion of how Microsoft data fabric supports DataOps principles. The DataOps Manifesto is a useful set of principles to guide your understanding of these powerful, grounded, industry-spanning ideas on improving technical team productivity, delivery quality, and cycle time in data analytics.

So today, another fundamental idea needs to be defined and given the manifesto treatment: the Data Journey.

As an industry, we have a conceptual hole in how we think about data analytic systems. We put them into production but then hope all the steps that data goes through from source to customer value work out correctly. We all know that our customers frequently find data and dashboard problems. Teams are shamed and blamed for problems they didn’t cause. They have problems with the data trapped in existing complicated multi-step data processes they need help understanding, often fail, and output insights that no one trusts. I talk with data teams every few days that have the same “morning dread” I had leading data teams 15 years ago. That feeling that something is going to go wrong, you’ll have no idea how to find it, and fixing it will put off all the other tasks that need to get done.

Marketers and Customer Experience leaders have had a similar experience of dread. They need to learn customers' interactions with their brand and marketing touchpoints. There is a crazy complexity in the path each person takes via a website, customer service team, and various other brand channels. So how could they give a consistent message? How could they improve service? The idea of a Customer Journey is very simple: a diagram that illustrates the steps your customer(s) go through in engaging with your company, whether it be a product, an online experience, a retail experience, a service, or any combination. The Customer Journey visually represents the total sum of experiences any given customer has with a brand. Automatically tracking the Customer Journey helps to inform marketing strategy and personalization efforts, improves onboarding, giving a clearer understanding of who your customers are and how the company serves them best. Software engineers have adopted a similar idea and called it a User Journey.

So as Customer Journeys have helped tame the complexity, lack of understanding, finger-pointing, and frustrated end customers in the marketing and CX world, the idea of a Data Journey is a method to watch over our data's complicated paths, avoid problems and errors, stop customer frustration, and increase your productivity and your team's happiness. It's Customer Journey for data analytic systems.



"Data Journey" refers to the various stages of data moving from collection to use in data analysis tools and systems. Much like the goal of a customer journey, a Data Journey should give you a better understanding of how, when, where, and what data flows through your data analytic systems. Data Journeys track and monitor all levels of the data estate, from data to tools to code to tests across all critical dimensions.

monitor all levels of the data estate, from data to tools to code to tests across all critical dimensions.

In the data world, we focus a lot on the data. If the data in your database needs to be corrected, has been transformed with errors, or is not clearly understood, then using that data will be a problem. However, you have many data tools in front of that database and behind that database: Talend, Azure Data Factory, DataBricks, Custom Tools, Custom Testing Tools, ETL Tools, Orchestrators, Data Science Tools, Dashboard Tools, bucket stores, servers, etc. Those tools work together to take data from its source and deliver it to your customers. That set of multi-tool set of expectations is a 'Data Journey. The Data Journey concept is about observing, not changing, your existing data estate. Data Journeys track and monitor all levels of the data stack, from data to tools to servers to code to tests across all critical dimensions. It supplies real-time statuses and alerts on start times, processing durations, test results, costs, and infrastructure events, among other metrics. With this information, you can know if everything ran on time and without errors and immediately detect the parts that didn't.

We continue to over-invest, as an industry, in the tools that run within our data estate. There are dozens of orchestrators, ETL Tools, databases, data science tools, data visualization tools, and data governance tools. Yet we lack a simple concept to understand, analyze, and predict all the myriad paths data takes within those tools. So we are blind to obvious errors and can't see the effects of errors downstream on our customers. No wonder, according to a study by HFS Research, 75 percent of business executives do not trust their data, and 70 percent do not consider their data architecture "world-class."

The Terms and Conditions of a Data Contract are Data Tests

Data contracts are a new idea for data and analytic team development to ensure that data is transmitted accurately and consistently between different systems or teams. The Terms and Conditions of a Data Contract are Automated Production Data Tests.

A data contract is a formal agreement between two parties that defines the structure and format of data that will be exchanged between them. Data contracts are a new idea for data and analytic team development to ensure that data is transmitted accurately and consistently between different systems or teams.

One of the primary benefits of using data contracts is that they help to ensure data integrity and compatibility. By defining the structure and format of the data upfront, both parties can be confident that the data will be transmitted and received expectedly. This can help prevent data loss or corruption, which can seriously affect businesses or organizations.

Another benefit of data contracts is that they allow for easier maintenance and updates. Because the structure and format of the data are defined in the contract, it is easier for developers to make changes or updates to the data without breaking compatibility with other systems. This can save time and effort, as developers do not need to spend as much time troubleshooting issues related to data transmission.

Data contracts can also improve communication and collaboration between different teams or departments. By defining the data that will be exchanged, both parties can better understand the needs and requirements of the other, leading to more effective communication and collaboration.

Several data contracts can be used in data and analytic team development. One common type is the schema, which defines the structure and format of data using a specific markup language such as XML or JSON. Another type of data contract is the service contract, which defines the operations performed by a particular service, such as a SQL query or API.

The best data contract is an automated production data test. It defines the terms and conditions of the data contract. Data testing plays a critical role in the process of implementing data contracts. Data contracts define the structure and format of data exchanged between two parties. Data testing ensures that the data is transmitted and received accurately and consistently.

Several different types of quality control checks can be used to ensure the accuracy and reliability of data in a production environment. Some common types of quality control checks for production data include:

- **Data validation:** This check is used to verify that the data being processed or stored in a production system is accurate and conforms to the required standards and specifications. Data validation can involve checking the data for errors, inconsistencies, or missing values and can be performed using automated tools or manual processes.
- **Data integrity checks** ensure that the data being processed or stored in a production system is complete and accurate. Data integrity checks can verify that the data is consistent with other sources or systems and that it has not been tampered with or corrupted in any way.
- **Data quality checks:** These checks ensure that the data being processed or stored in a production system is high quality and meets the required standards. Data quality checks can involve verifying the data's accuracy, completeness, and consistency and checking for errors or issues that may affect the data's usability or value.
- **Statistical process control (SPC) checks** are quality control methods that use statistical analysis to monitor and control processes to ensure that they operate within specified limits. In the context of data systems, SPC checks can be used to monitor and control the quality of data being processed or stored in the system. SPC checks involve collecting data from the process being monitored and using statistical techniques to analyze the data to identify trends, patterns, or deviations from expected values. This can help identify any issues or problems with the process that may need to be addressed to improve the data quality.

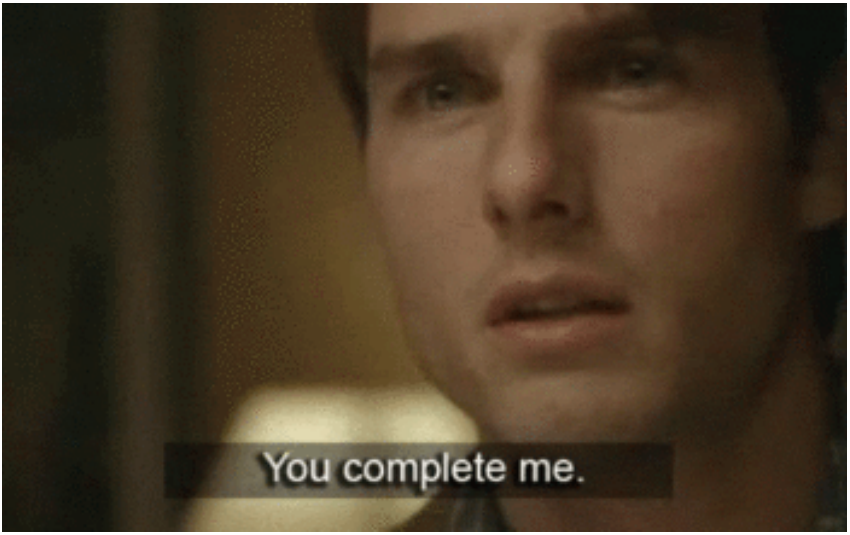
Overall, the role of data testing in data contracts is to ensure that the data is transmitted and received accurately and consistently according to the structure and format defined in the contract. By performing thorough data testing, organizations can be confident that their data exchange processes are reliable and efficient and comply with the data contract terms.

Data contracts are a valuable new tool in data and analytic team development. They help ensure data integrity and compatibility, facilitate easier maintenance and updates, and improve communication and collaboration between different teams or departments. By defining the structure and format of the data upfront, both parties can be confident that the data will be transmitted and received in the expected manner, which can help to prevent issues and improve the overall efficiency of the data exchange process.

Data tests are enforcement of the terms and conditions in your data contracts!

“You Complete Me,” said Data Lineage to Data Journe

The benefits of a long-lasting relationship between Data Journeys and data lineage can help give a complete view of your data operations. Both tools help deliver more trusted insight to your customer.



What is data lineage?

Data lineage traces data’s origin, history, and movement through various processing, storage, and analysis stages. It is used to understand the provenance of data and how it is transformed and to identify potential errors or issues. Data lineage can also be used for compliance, auditing, and data governance purposes. Data lineage has a long history, starting as a tool for compliance and auditing in mainframe systems, evolving to address the challenges of understanding the origin, history, and movement of data across multiple systems and processes, and becoming a critical aspect of data governance, compliance, and data management in recent years, with the growing complexity of data and the increasing importance of data privacy and security. Data lineage and a data catalog are better together because they provide a more complete and accurate view of the data.

What about DataOps Observability? How does it compare?

A five-on-five comparison of data lineage vs. DataOps Observability
Five on data lineage:

- Data lineage traces data’s origin, history, and movement through various processing, storage, and analysis stages.

- Data lineage helps to understand the provenance of data and how it is transformed and identify potential errors or issues.
- Data lineage is important for data governance, compliance, and auditing purposes.
- Data lineage can be used to create a data dictionary or data catalog, which can be used to understand the data better.
- Data lineage is typically stored in separate systems from the data itself and can be difficult to keep up to date.
-

Five on DataOps Observability:

1. DataOps Observability is the ability to understand the state and behavior of data and the software and hardware that carries and transforms it as it flows through systems. It allows organizations to see how data is being used, where it is coming from, its quality, and how it is being transformed.
2. DataOps Observability includes monitoring and testing the data pipeline, data quality, data testing, and alerting.
3. Data testing is an essential aspect of DataOps Observability; it helps to ensure that data is accurate, complete, and consistent with its specifications, documentation, and end-user requirements.
4. Data testing can be done through various methods, such as data profiling, Statistical Process Control, and quality checks.
5. DataOps Observability is crucial for identifying and addressing issues with the data pipeline, including quality and timeliness, improving data governance, end-user satisfaction, and ensuring compliance with regulations.

Your house is on fire: look at blueprints or listen to smoke alarms? What is missing in data lineage?

Data lineage answers the question, “Where is this data coming from, and where is it going?” It is a way to describe the data assets in an organization. It is a description used to help data users understand where data came from and, with a data catalog, the content of specific data tables or files. Data Lineage does not answer other key questions, however. For example, data lineage cannot answer questions like: “Can I trust this data used in this specific pipeline in this way?” “And if not, what happened during the data journey that caused the problem?” “Has this data been updated with the most recent files?”

Understanding a complete data journey and building a system to monitor that complex series of steps is an active, action-oriented way of improving the results you deliver to your customers. DataOps Observability solutions create and describe Data Journeys representing this run-time lineage.

Think of it this way: **if your house is on fire, you don’t want to go to town hall and get the blueprints** of your home to understand better how the fire could spread. You want smoke detectors in every room so you can be alerted quickly to avoid damage. Data Lineage is the blueprint of the house; DataOps Observability is the set of fire detectors sending you signals in real-time. Ideally, you want both run time and data lineage to provide the complete picture.

Data Lineage's trust gap.

While data lineage can provide a clear understanding of the origin, history, and movement of data, it may not always fully capture the impact of a data error. Data lineage primarily focuses on tracking the movement of data and how it is transformed. Still, it may not always provide a complete picture of how data is used or how it affects other systems or processes.

Data errors can have a wide range of impacts depending on the nature of the error and how it propagates through the data pipeline. This impact can be hard to predict and understand without additional context. For example, a data error may only be apparent when combined with other data or used in a specific analysis or report. Additionally, data lineage may not capture the impact of data errors on downstream systems or processes. For example, if an error in the data causes a downstream system to fail, data lineage may not capture this information.

In summary, data lineage can provide a clear understanding of the origin, history, and movement of data, but it may not always fully capture the impact of a data error. It is important to have additional tools and processes in place to understand the impact of data errors and to minimize their effect on the data pipeline and downstream systems.

Data lineage vs. the run time operations on data

Runtime operations, such as those captured and monitored by DataOps Observability solutions, refer to the actions performed on data while it is being processed. These operations can include data movement, validation, cleaning, transformation, aggregation, analysis, and more. They are performed on the data during production runtime when it is actively processed.

Data lineage can provide information about how data is transformed and moved, but it does not provide information about the specific runtime operations performed on the data. Conversely, runtime operations can impact the data but not provide information about its history, movement, and origin.

Data lineage and runtime operations on data are related but distinct concepts. Data lineage provides information about the origin, history, and movement of data, while runtime operations provide information about the actions performed on data while it is being processed. Both concepts are important for understanding and managing data effectively.

Data observability and data lineage are complementary concepts.

Data lineage refers to tracing data's origin, history, and movement through various processing, storage, and analysis stages. It is used to understand the provenance of data and how it is transformed and to identify any potential errors or issues.

On the other hand, DataOps Observability refers to understanding the state and behavior of data as it flows through systems. It allows organizations to see how data is being used, where it is coming from, and how it is being transformed. This includes other information such as data quality metrics, processing steps, timing, data test results, and more.

As such, data lineage and DataOps Observability have their own separate capabilities and goals, which, when used together, deliver higher-quality systems and products.

Trust comes from verification first, then root cause analysis

Trust in data, which is critical for the successful use of data, comes from verification first, then understanding the root cause of any issues second. Verification is checking that data is accurate, complete, and consistent with its specifications or documentation. This includes checking for errors, inconsistencies, or missing values and can be done through various methods such as data profiling, data validation, and data quality assessments.

Once the data has been verified, it is important to understand the root cause of any identified issues. This includes identifying where the data came from, how it was collected, and how it was transformed before it was stored in the database. Understanding the root cause of issues can help prevent them from happening again in the future and improve data governance. Data Lineage can be one of many helpful tools in diagnosing root-cause errors.

Data lineage is what's in your database – which is not everything.

Data lineage primarily focuses on tracking the movement and transformation of data within the database or data storage systems. It does not capture the full details of all the steps and tools used to process and analyze the data before storing it in the database. DataOps Observability handles that.

Data lineage in a database typically includes information about where the data came from, when it was created or last modified, and who created or modified it. However, it does not typically include information about the tools or processes used to collect, clean, or transform the data before it was stored in the database.

To capture a more complete picture of the data's journey, it is important to have a data pipeline management and monitoring system that can track the data flow and all the steps and tools it went through before it reaches the database. This additional information can help to understand the data better, troubleshoot issues, and improve data governance. To capture a more complete picture of the data's journey, it is important to have a DataOps Observability system in place.

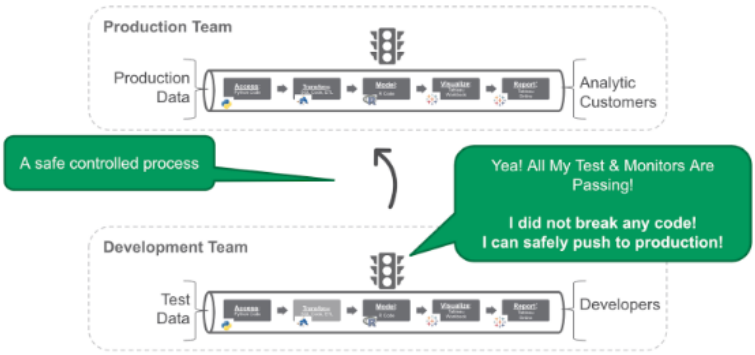
Data lineage is static and often lags by weeks or months.

Data lineage is often considered static because it is typically based on snapshots of data and metadata taken at a specific time. This means that the data lineage information may not reflect the current state of the data or the most recent changes that have been made. Additionally, data lineage information is often stored in separate systems from the data itself, making it difficult to keep it up to date and in sync with the data.

These factors can contribute to outdated data lineage lagging by weeks or months. This can make it difficult to understand the current state of the data and how it is being used, which can be problematic for data governance, compliance, and auditing purposes. Additionally, in situations where data is frequently updated or transformed, this lag in data lineage can make it difficult to identify issues or errors with the data. However, a DataOps Observability capability, by monitoring data processes in real-time, can create and feed a lineage system with the latest details, ensuring the information is up to date.

Data lineage fails at impact analysis.

You are in bed, smell smoke, touch the door nob, and it's hot. Is it time to get the blueprints of your apartment out to check which room will burn next? Like an apartment blueprint, Data lineage provides a written document that is only marginally useful during a crisis. This is especially true in the case of the one-to-many, producer-to-consumer relationships we have on our data architecture. Which report tab is wrong? Which production job filled that report? When did it last run? Did it fail? Are problems with data tests? These questions are much more salient and actionable than pulling out the dusty old data lineage blueprint. Data Observability trumps lineage in a production crisis.
But what about during development? Doesn't lineage help me tell the impact of my code change? Yes, it does – to a point. Data lineage does not provide detailed regression, functional, or performance test results. Testing data and analytic systems require a development system with accurate test data, tools, and relevant tool code. All of those things come together with an overlay of automated tests. Only then can you tell the true impact of a column name change on the data transformations, the models, and the visualization you give to your customers. No software engineer trusts an abstract metamodel of their code to prove that it works in production without running their software code through a series of rigorous automated tests before it gets to production. We must do the same as data analytic teams. DataOps Observability enables this.



Code and tools can cause production problems — data lineage does not help.

One view of data and analytic systems is that it is a grouping of tools, driven by code, acting in a specified order upon data. Data lineage alone can not fully capture the issues caused by code and tools acting upon data. Data lineage primarily focuses on tracking the movement and transformation of data, but it does not provide information about the code and tools used to process and analyze the data.

Code and tools can introduce errors or issues with the data, such as bugs, incorrect logic, or unexpected behavior, that data lineage may not capture. Additionally, if the code or tools are not properly tested, monitored, or maintained, they may introduce issues that are not immediately apparent over time.

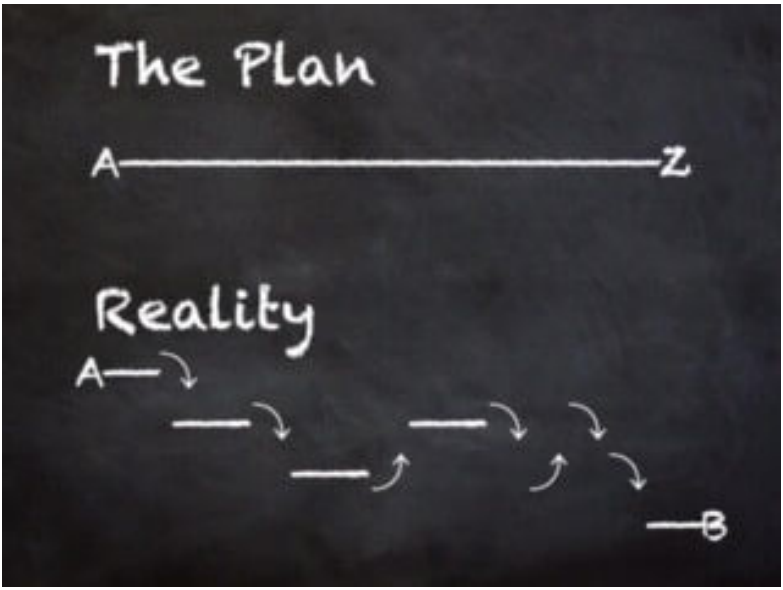
To address these issues, it is important to have additional processes and tools in place, such as code review, testing, and monitoring, to ensure that the code and tools are functioning correctly and to detect any issues as early as possible. Data lineage can still provide valuable information about the data. Still, it should be used with DataOps Observability tools to understand and address issues caused by code and tools fully.

Data lineage does not directly improve data quality.

Though valuable, Data Quality scores are largely static. They measure data sets at a point in time. DataOps Observability is dynamic; it is the testing of data, integrated data, and tools acting upon data — as it is processed — that produces details on data timeliness, flow rates, and errors. A financial analogy: Data Quality is your Balance Sheet, and Data Observability is your Cash Flow Statement – any successful business requires both. Crafting your data observations into a singular Data Journey that integrates all tools, tech, data, and results in a single view .. that is DataOps Observability.

Data reality vs. the data lineage specification.

In reality, data often differs from its specifications or stated lineage due to a variety of factors. One common reason is that data is often collected and entered into systems by humans, and human error can lead to inaccuracies or inconsistencies. Additionally, data may be transformed or cleaned during processing, resulting in changes to the data that are not reflected in the original lineage or documentation. Furthermore, data may not be kept up to date or stored in multiple locations, making it difficult to reconcile the data with its original lineage.



Another reason for the discrepancy between reality and specifications is that data is often collected from various sources, which may not have the same accuracy or completeness. Some data may also be missing or incomplete, leading to missing values, outliers, or other issues. Furthermore, data may be updated over time, but the original specifications may not be updated accordingly. In addition, the context of data usage may change over time, leading to the data being used in ways that were not originally intended, leading to misinterpretation and errors.

Reality and specifications for data often differ due to various factors such as human error, the transformation of data during processing, lack of up-to-date documentation, data collected from various sources, data's context change over time, and more. This can make it difficult to understand the data's current, correct lineage and use it effectively.

One way to ensure that data meets the specifications and, more importantly, the business requirements is to track and monitor all data processes with a DataOps Observability solution.

Managing data team stress: A brain scan vs. cognitive behavior therapy

In a recent DataKitchen/data.world survey, 97% of data engineers report experiencing burnout in their day-to-day jobs. 70% say they will likely leave their current company for another data engineering job in the next 12 months. 79% have considered leaving the industry entirely.

What is the best way to solve this stress? Let's try another analogy: a brain scan vs. cognitive behavior therapy. Brain scans, such as functional magnetic resonance imaging (fMRI) or positron emission tomography (PET) scans, create detailed brain images to understand how it functions. While brain scans can provide valuable information about brain activity, they are not typically used as a standalone treatment for anxiety. Cognitive-behavioral therapy (CBT) is a type of talk therapy that focuses on changing how a person thinks and behaves to reduce anxiety. It involves identifying and changing negative thought patterns and behaviors contributing to anxiety. CBT is an effective treatment for anxiety and is recommended by the National Institute of Mental Health (NIMH) as a first-line treatment.

Data Lineage is like a brain scan of your data. In contrast, DataOps Observability uses a series of data tests and monitors to find those 'negative thought patterns' in your data and notify you immediately. Like CBT, DataOps Observability effectively treats data team anxiety, while data lineage is used to create detailed images of the state of your data tables to understand how it functions.

Data lineage can provide valuable information, but it fails as a standalone treatment for data team anxiety. However, it can be used with other treatments to understand the underlying mechanisms of data anxiety and to help tailor treatment!

Conclusion

In the 1990's RomCom Jerry Maguire, the main characters have this iconic exchange:

> *Jerry Maguire: [babbling and struggling] I love you. You… you complete me. And I just…*
> *Dorothy: Shut up,*
> *[pause]*
> *Dorothy: Just shut up.*
> *[Pause]*
> *Dorothy: You had me at "hello." You had me at "hello."*

We've tried to show the benefits of a long-lasting relationship between DataOps Observability and data lineage. Both can help give a complete view of your data and operations. Together, both tools help deliver more trusted insight to your customer.

Now, go ahead and "Show Me The Money" (with your data)!

Two Downs Make Two Ups: The Only Success Metrics That Matter For Your Data & Analytics Team:

You spend all day helping your customers leverage analytics for improved business performance, so why are you so un-analytic about how you run your data analytics teams? Where is your data and analytic team metrics report?

Introduction. How to measure your data analytics team?

So it's Monday, and you lead a data analytics team of perhaps 30 people. You've got a new boss. And she is numbers driven – great! But wait, she asks you for your team metrics. Like most leaders of data analytic teams, you have been doing very little to quantify your team's success. All you had to do was write a few bullet points every week for your last boss. It's your first leadership meeting, and she gives you this look – where are my numbers?

At DataKitchen, we have talked with many CDOs, data leaders, and other data team managers, and they have, ironically, been very un-analytic about how they run their teams. Bullet points and bravado seem to be the norm. You spend all day helping your customers leverage analytics for improved business performance, so why are you so un-analytic about how you run your data analytics teams? Where is your metrics report?

What should be in that report about your data team? What should I track? What are the metrics that matter?

Gartner attempted to list every metric under the sun in their recent report, "Toolkit: Delivery Metrics for DataOps, Self-Service Analytics, ModelOps, and MLOps," published February 7, 2023. It lists dozens of metrics to track across their operational categories: DataOps, Self-Service, ModelOps, and MLOps. Forty-five metrics! For example, Gartner's DataOps metrics can be categorized into Velocity, Efficiency, and Quality. Under Velocity, the Mean Time to Deliver Data metric measures the time it takes to deliver data. The Active Data Ratio metric determines the percentage of datasets that deliver value. The Data Change Request Ratio metric measures the rate of business demand for data. The Data Reuse Ratio metric calculates how many new requests can be fulfilled from existing components. The Mean Time to Recovery metric measures how quickly defects can be resolved. Under Efficiency, the Number of Data Product Owners metric measures the value of the business's data products. Under Quality, the Data Quality Incidents metric measures the average data quality of datasets, while the Active Daily Users metric measures user activity across data platforms.

DataOps Metrics

Category	Metric Name	Metric Definition	Purpose
Velocity	Mean Time to Deliver Data	Time from data request to usable dataset/number of requests	Measure how long data takes to be delivered
Velocity	Active Data Ratio	Number of datasets produced/Number of datasets in use	Measure the amount of data that is delivering value
Efficiency	Data Change Request Ratio	Number of change requests/Number of data sources	Measure the rate of business need for data
Velocity	Data Reuse Ratio	Number of new requests/Number of reused data assets	Measure how many new requests can be fulfilled from existing components
Velocity	Mean Time to Recovery	Total time to recover data/Number of data incidents	Measure how quickly defects can be resolved
Quality	Data Quality Incidents	Number of data quality issues/Number of data assets in use	Measure the average data quality of datasets
Quality	Active Daily Users	Number of user login/Total number of users	Measure user activity across data platforms
Efficiency	Number of Data Product Owners	Number of products that are delivered with a PO role	Measure value of products that are used by the business

Other data leaders have tried to adapt the IT/software incident metrics world to the data world. IT incident metrics are used to evaluate the reliability and efficiency of a system or process. MTBF (Mean Time Between Failures) is the average time between system or process failures. MTTR (Mean Time to Repair) is the average time to repair a failed system or process. MTTA (Mean Time to Acknowledge) is the average time it takes for a team to acknowledge an incident once it has occurred. MTTF (Mean Time to Failure) is the average time until a failure occurs, assuming that the failure is non-repairable. These metrics help organizations identify areas for improvement in their software systems and processes and can be used to set goals for reducing downtime and increasing system availability. The challenge with these metrics is that software failure is often more discrete – your application or website is up or down. But what about a Data Journey constructed from multiple data engineering tools, servers, data sets, and dashboards? And is a small data error affecting one data sales region a failure? To that specific user, it is. Just because the infrastructure appears to be working in data systems does not mean users will not see problems.

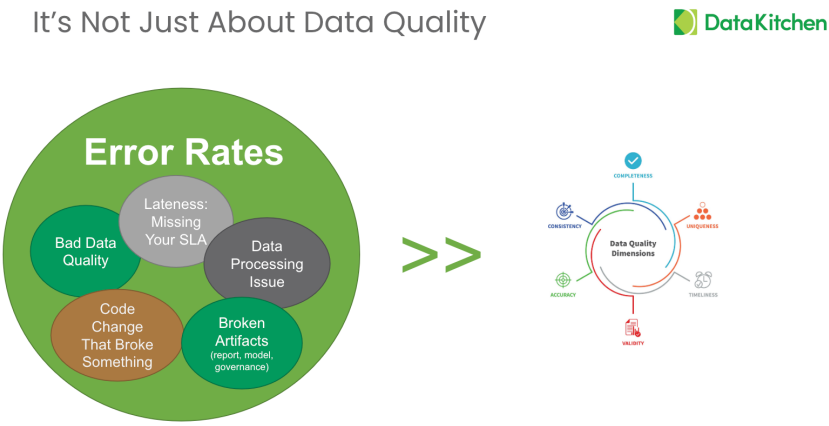
> *Data systems require trust. And trust comes from having low errors. And your team getting things done.*

Just Track Four Things.

Here are a set of simple, general key performance indicators (KPIs) that can be used to evaluate the performance of a data analytics team. Organizations can concentrate on two critical “down” metrics (errors and cycle time) and two “up” metrics (productivity and customer satisfaction) to effectively measure the performance of their data analytics teams.

Metric Number One: Errors

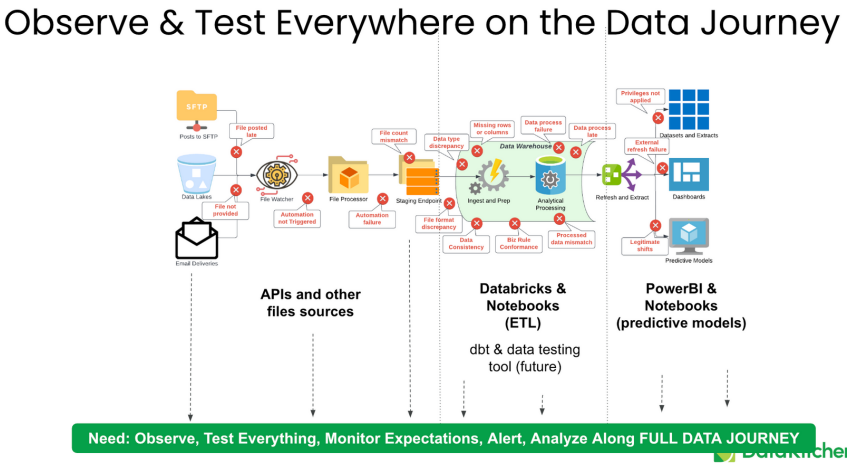
Reducing errors in data analytics is crucial for ensuring the accuracy and reliability of the insights generated by the team. Errors can originate from various sources, including data collection, integration, models, visualization, governance, and security. Organizations will make better-informed, data-driven decisions by minimizing errors and avoiding the negative consequences of relying on inaccurate information. Automated techniques can track errors. Automated data test tools can help identify data entry errors or processing errors.



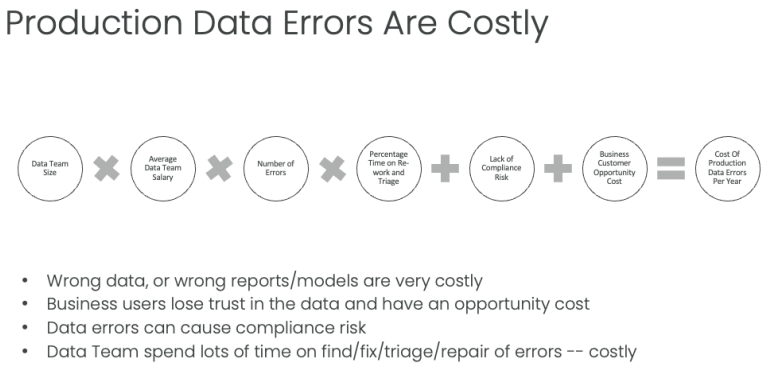
But what are errors? Look at the world from your customer perspective – the dashboard is wrong, the data is late, or the numbers ‘look funny.’ All of these items are errors in their eyes. Those errors can come from many sources

1. Bad raw data in the analytic process
2. Some processing errors during the data production process
3. Some new code/config acting upon data produces a problem
4. Being late and missing an SLA (or performance/speed)
5. Broken Artifacts – the data is right, but the tools acting upon the data (such as the model or report) show incorrect amounts.

Where do you find errors? Look for errors everywhere in your Data Journey. Find the problems fast, notify the right person, and drive the person responsible for fixing the problem to the source of the error quickly. Use tools like DataKithchen DataOps Observability to build and watch your Data Journeys.



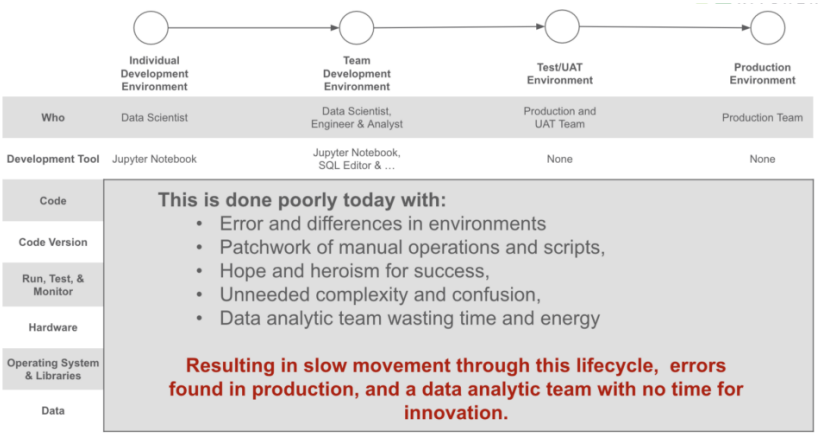
There are four main benefits of reducing errors. More Innovation – Reducing errors eliminates unplanned work that pulls data team members from their high-priority analytics development tasks. An enterprise cannot derive value from its data unless data scientists can stay focused on innovation. More Trust – Errors undermine trust in data and the data team. Less confidence means less data-driven decision-making; in other words, emotionally-biased decision making. Less Stress – Errors can occur at any moment. The feeling of continuous anxiety is unhealthy for team members and reduces their ability to relax and enjoy their work. Happy and relaxed people think more clearly and creatively. Less Embarrassment – Errors in data analytics tend to be very public. When data errors corrupt reports and dashboards, it can be extremely uncomfortable for the manager of the data team. As embarrassing as having your mistakes highlighted in public, it's much worse if the errors go unnoticed. Imagine if the business makes a costly and critical decision based on erroneous data.



Metric Number Two: Cycle Time

In today's fast-paced business environment, the ability to quickly deploy new data and analytics solutions has become increasingly important for organizations seeking to maintain a competitive edge. Cycle time, or the duration from initiating a data analytics task to its completion and deployment, is a critical metric affecting an organization's ability to generate insights and make data-driven decisions. A shorter cycle time enables a more agile decision-making process and allows organizations to capitalize on emerging trends or opportunities.

What is the significance of cycle time in data analytics deployment? A shorter cycle time enables organizations to quickly identify and act upon opportunities or challenges, granting them a competitive advantage in their respective industries. Rapid deployment of data and analytics solutions ensures businesses can stay ahead of their competitors and capitalize on emerging trends. As a result, businesses can adapt their strategies and operations more effectively, ultimately improving their overall performance. By reducing wasted time through minimizing cycle time, organizations can optimize their resource allocation, ensuring that data analysts, data scientists, and other stakeholders can focus on high-impact projects and maximize their contributions to the organization's success.

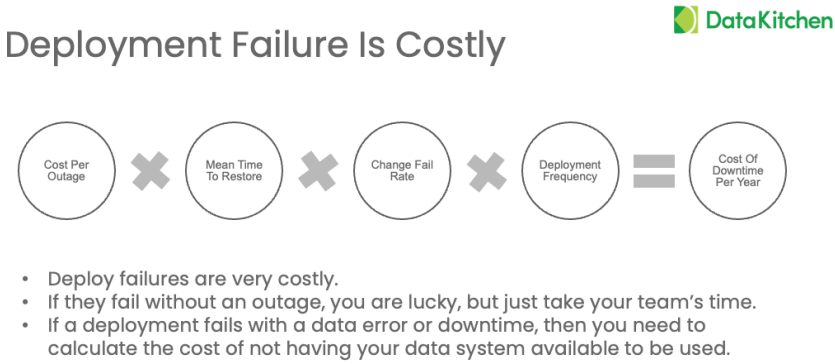


The first part of cycle time is the speed of deployment from development into production and use by your customer. Follow the first several principles of the DataOps Manifesto: get something in your customer's hands and accept feedback to maximize your team's learning. To do this, your team must create a high-quality train track' to deploy your new code/config/data between environments. Deployment methods include an automated process like 'CI/CD' or a more functional "Blue-Green" test approach. Either way, part of your cycle time measurement is speed to deploy.

However, it is not just the speed at which you can deploy some new SQL, a new data set, a new model, or another asset from development into production. It is the enabling of deployment with low risk as well. You would not run a train through a busy city without signals at every intersection. Likewise, automated testing based on good test data in a representative technical environment with source code taken from a versioned repository is key. You want to enable your young data developers to make a small change in a big, complex system and automatically, without any manual intervention, have the system tell them that it will have the expected impact on production.

Organizations can adopt Agile methodologies, such as Scrum or Kanban, to reduce cycle time, which promotes iterative development, continuous feedback, and flexible planning. Additionally, investing in data testing and observability and implementing DataOps principles can help streamline data workflows and reduce the time it takes to complete data analytics projects.

Continuous time to deploy new data and analytics is a critical metric impacting an organization's ability to generate insights and make data-driven decisions. By understanding the factors affecting cycle time and implementing strategies to reduce it, organizations can accelerate the delivery of valuable insights, gain a competitive advantage, and optimize resource waste.



Metric Number Three: Team Productivity

Increasing the productivity of a data analytics team allows organizations to generate more valuable insights with the same or fewer resources. A highly productive team can efficiently prioritize and execute high-impact projects, ultimately contributing to the organization's overall success. Tracking and monitoring productivity is essential for a data analytics team. It helps optimize team performance and ensures that the team's work aligns with the organization's strategic goals.
Unfortunately, many teams do not track the amount of work they do. In one recent survey, a CDO asked his team how they spent their time. Why were they not getting anything new done? He found that over 60% of their time is not doing value-added work. They are waiting, blocked, re-doing work already done, fixing errors, etc.

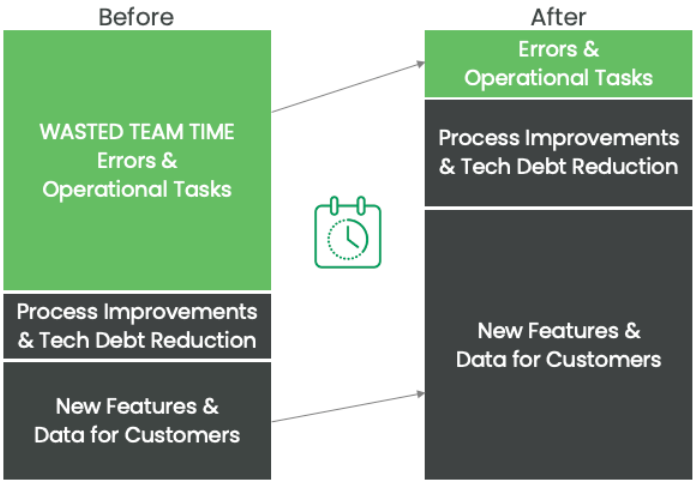
Waste	Bug fixing & re-work	5%
	Incident response	5%
	Waiting/Blocked/Handoffs	2%
	Over-engineering	2%
	Extra effort for Sub-optimal solutions or workarounds	2%
	Relearning (e.g. lack of documentation or SME support)	3%
	Un-implemented work	2%
	Unnecessary meetings	2%
Unplanned Activity	Unplanned chats	2%
	Unplanned but value-add	11%
Core Management	1:1s	10%
	Team meetings	5%
	Company meetings	3%
	Other Admin tasks	10%

- From a customer survey of their data team
- Over 60% of their time is not doing value added work
- Look at the cost of all the overhead your team is doing!

How does one track a team's productivity? Productivity can be measured using metrics such as the number of projects completed per team member, the percentage of projects completed on time, and the estimated ROI of each project. However, more agile ways of tracking productivity are better. A simple way is the idea of a 'story point.' Story points measure the complexity and effort required to complete a particular task or user story in agile development. While they are often used to estimate the amount of work needed for a sprint or release, they are great for tracking productivity.

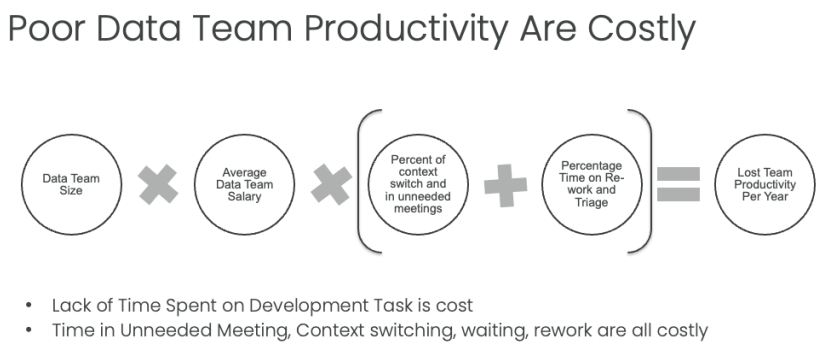
To use story points to track productivity, you first need to establish a baseline velocity, which is the average number of story points your team completes in a given sprint. You can use this baseline velocity as a reference point to measure how much work your team completes over time. For example, if your team's baseline velocity is 20 story points per sprint, and they complete 25 story points in the next sprint, you can infer that they were more productive than usual. Similarly, if they completed only 15 story points in the following sprint, you can infer that they were less productive than usual. By tracking your team's velocity over time, you can gain insights into their productivity and identify trends or patterns that may help you improve your processes or identify areas for improvement.

It's important to note that story points are not a perfect measure of productivity. They are subjective and can vary depending on the team's experience and understanding of the task. However, they can still be useful tools for tracking and improving agile data analytics development productivity.



Where is productivity lost? A typical view is that my team is just slow to do their job. It takes them too long to write SQL, python, or make a dashboard. Surveys from Gartner, Eckerson, and others show that waste is the biggest impact on productivity. Reducing waste is a core principle of lean manufacturing, which aims to create more efficient and effective processes while minimizing costs and resources. In lean manufacturing, waste refers to any activity or process that does not add value to the final product. There are many more sources of time waste in a data team, for example, excess team meetings. All sources of waste can be rolled up into the error and cycle time metric. Data teams are full of wasted time and rework.

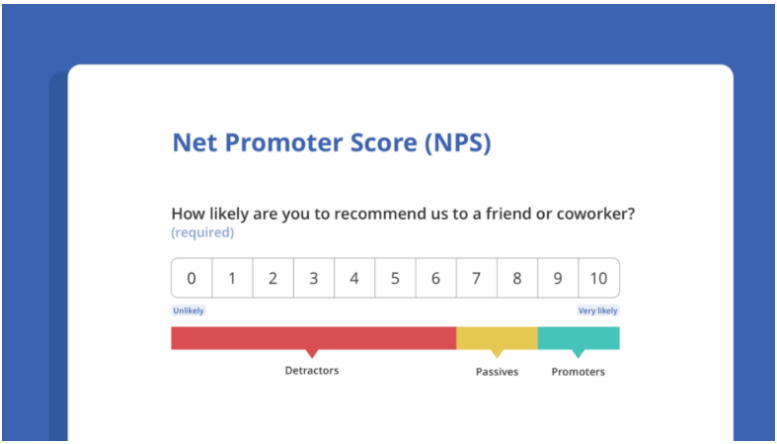
Given today's economy, a key question that data teams leaders are asking is how to increase the total amount of data analytics insight generated without continually adding more staff (and cost). The answer lies in tracking productivity and reducing the amount of wasted time. Data leaders are learning that their teams are not very efficient or effective. Where is the problem? Is it in lots of re-work, delays, excess meetings, and context switching? Is it in the fact that teams 'throw their work over the wall and hope it works in production? Or is it in the loss of time for the team to rush to fix things when they break?



Metric Number Four: Customer Satisfaction

Ensuring high levels of customer satisfaction is essential for demonstrating the value of a data analytics team to internal and external stakeholders. Satisfied customers are more likely to continue using the team's services, provide positive feedback, and advocate for the team. Consistently delivering high-quality work on time and within budget can increase stakeholder trust and satisfaction, ultimately contributing to the organization's success. Data trust is imperative.
As we noted in a recent blog, the real problem in data analytics is that teams need to deliver insight to their customers without error, put new ideas into production rapidly, and minimize their 'insight manufacturing' expenses … all at the same time. Customer expectations are high for your team. And don't be surprised if you find your customer is not happy with the state of data analytics in your organization. For example, in a recent CDO survey, less than half of the organizations – 40.8% — report successfully using analytics. A meager 23.9% of companies report creating a data-driven organization. A dismal 20.6% of organizations state they have established a data culture.

Measuring customer satisfaction for a data team is vital for several reasons. Firstly, it helps the team understand how well they are meeting the needs and expectations of their customers, which can inform their priorities and strategies going forward. Secondly, it can help identify areas for improvement and potential pain points in the data team's processes or services. Finally, it can also build customer relationships and demonstrate the value of the data team's work.



The data team can use various methods to measure customer satisfaction, including surveys, feedback forms, interviews, and user testing. These methods can help the team gather quantitative and qualitative customer satisfaction data. For example, a survey could ask customers to rate their satisfaction with the data team's services on a scale of 1 to 10 and provide space for open-ended feedback on what the team is doing well and what could be improved. An interview could involve asking customers about their experiences working with the data team and exploring areas where they have had difficulties or frustrations. Once the data team has gathered customer feedback, they can use it to identify areas for improvement and make changes to their processes or services as needed. They can also share the results with their customers and other stakeholders, demonstrating that they listen and respond to feedback.

Measuring customer satisfaction is an integral part of building a successful data team. It can improve the team's processes and services, build relationships with customers, and demonstrate the value of the team's work.

Which metrics should I use, exactly?

The world of team metric measurement is confusing. As we previously discussed, Gartner recently published dozens of metrics to track for data teams, and software engineering has its metric world. DORA Metrics, Flow Metrics, and IT/Software Incident Metrics are different metrics used to measure various aspects of software development, deployment, and operations.

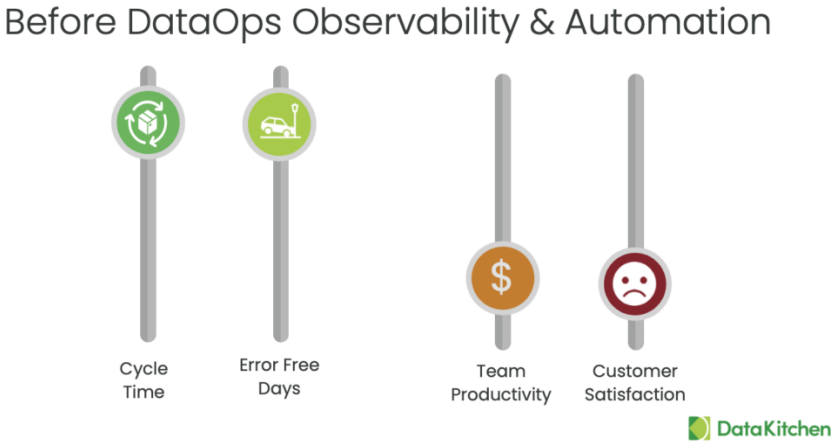
- **Gartner Data/Model/MLOps Metrics:** Gartner states that effective metrics for data and analytics delivery teams should focus on reducing problems and delays caused by code, data, and model defects and drift. To achieve this, teams should map their entire value stream, from product initiation to delivery, to identify areas for improvement and reduce lead times while improving quality. This value stream map can also help to reduce the cost of handovers between DataOps, ModelOps, and MLOps processes and functions.
- **DORA Metrics,** which stands for "DevOps Research and Assessment," were developed by a team of researchers to measure the performance of DevOps teams. These metrics focus on four key areas: delivery throughput, deployment frequency, change failure rate, and mean time to restore (MTTR). DORA Metrics are designed to provide a holistic view of the performance of a DevOps team and can be used to identify areas for improvement and track progress over time.
- **Flow Metrics** are a set of metrics used to measure workflow through a software development process. These metrics help teams identify bottlenecks and inefficiencies in their workflows and optimize their processes to improve throughput and quality. Flow Metrics typically include lead time, cycle time, and work in progress (WIP).
- **IT/Software Incident Metrics** are used to track and measure incidents that occur in software development and operations. These metrics can include the number of incidents, severity, time to resolution, and the mean time between failures (MTBF). IT/Software Incident Metrics are designed to help teams identify trends and patterns in incidents and take action to prevent them from occurring in the future.

While these metrics are all related to software or data development and operations, they each focus on different aspects of performance and can be used to answer different questions. DORA Metrics focus on the overall performance of a DevOps team, Flow Metrics focuses on the flow of work through a process, and IT/Software Incident Metrics focus on the occurrence and resolution of incidents. Gartner focuses on the unique characteristics of data and data production. Each metric type is useful in its own right, and teams may use a combination of these metrics to gain a more comprehensive understanding of their performance. However, these various metrics can be seen as rolling into our four categories in the table below. They can be used as subcomponents to help quantify the four main metrics described in this blog post.

ERRORS	CYCLE TIME	PRODUCTIVITY	CUSTOMER SATISFACTION
Data Quality Incidents	Mean Time To Deliver Data	Story Points Per Month	Active Daily Users
Number Of Decisions Monitored	Mean Time To Recover (DORA)	Backlog Size Over Time	Mean Number Of Active Users
MTTF (Mean Time To Failure)	Mean Time To Deliver Analytical Model	Tasks Completed	Number Of Decisions Impacted Per Model
MTTA (Mean Time To Acknowledge)	MTTR (Mean Time To Repair)	Time Usage	Mean Value Per Decision
MTBF (Mean Time Between Failures)	Mean Time To Retrain A Model	Team Satisfaction	Total Value Across All Business KPIs
Number Of Data Incidents	Data Delivery Time	Cloud Costs	Net Promoter Score (NPS)
Number Of Data Quality Issues	Time To Restore Data	Return on Investment	Monthly Active Users
Change Failure Rate (DORA)	Deployment Frequency (DORA)	Number Of Datasets Produced	Number Of Datasets In Use
Late, Missed SLA	Lead Time For Changes (DORA)	Git Commits	Customer Satisfaction (CSAT)
Security Incidents	Model Deployment Failure Rate	Work in Progress (WIP)	
Bad Raw Data, Integrated Data			
Broken Reports, Models, Catalogs			

Two Downs Make Two Ups: the relationship between errors and cycle time to productivity and customer satisfaction.

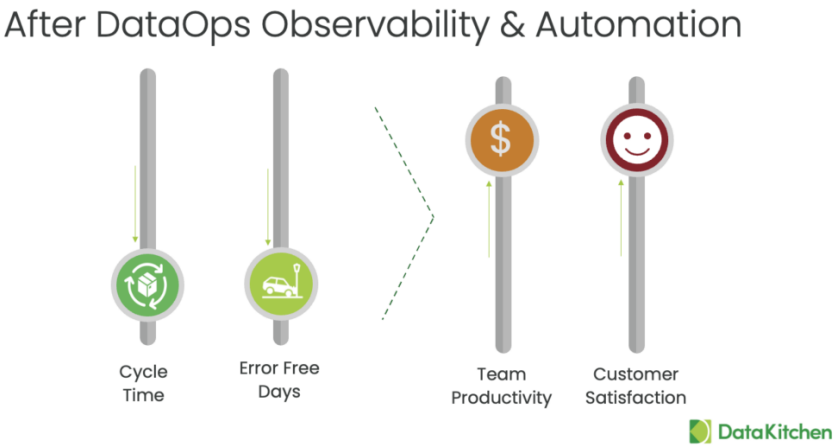
The concept of “Two Downs Make Two Ups” means that there is a relationship between errors and cycle time and their impact on productivity and customer satisfaction. The idea is that when errors and cycle time are lowered, there is an increase in productivity and customer satisfaction.



Errors can be anything from mistakes made in data entry to issues with code to a server going down and can significantly impact productivity and customer satisfaction. When errors occur, it can lead to delays in completing work and reduced quality of work produced. This can, in turn, lead to decreased customer satisfaction, as customers need to receive the level of service they expect.

Cycle time is when it takes for a deployment process to be completed, from start to finish. The longer the cycle time, the more time it takes to complete work, leading to decreased productivity and increased costs. By reducing cycle time, teams can complete work more quickly and efficiently, increasing productivity and cost savings.

By reducing errors and cycle time, teams can improve productivity and customer satisfaction. For example, by implementing DataOps Observability, quality testing measures, and using automation tools, teams can reduce the likelihood of errors occurring. Complete the new work correctly the first time, reducing the need for rework and delays.



Data teams are full of wasted time and rework. Customers are more likely to be satisfied when they receive their products or services promptly and feel that they can trust the results.

In summary, “Two Downs Make Two Ups” highlights the importance of reducing errors and cycle time to increase productivity and customer satisfaction. By focusing on these critical areas, teams can improve the quality of their work, complete tasks more efficiently, and deliver better results to their data customers.

Are you looking for help in tracking and improving these metrics? The principles of DataOps enabled via DataOps Observability and Automation are keys to unlocking this challenge. DataKitchen’s scalable software accelerates your path to measuring and accelerating your way to improving these metrics.

DataOps Observability: Taming the Chaos

Jason is the Chief Data Officer for Company X, and he is responsible for six teams of data engineers, scientists, and analysts across three geographic locations. The breadth of his teams' work and the technologies they use present a significant challenge to his main goal: to deliver new and useful analytics solutions to the business. The problem? He already has 1000 data pipelines running with no way to know if they are working properly - or producing what is expected - with 10,000 users relying on the data insights they produce. And, his teams are so overworked maintaining these systems and firefighting that they cannot deliver on the many new requests coming from the business.

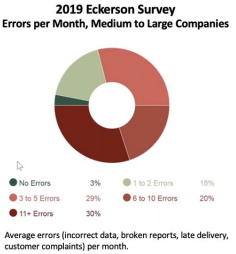
Like Jason, you have a significant investment in your data and the infrastructure and tools your teams use to create value. Do you know for sure that it's all working properly, or do you hope and pray that a source data change, code fix, or new integration won't break things? If something does break, can you find and fix the problem quickly, or does your team spend days just diagnosing the issue? Do you fear that phone call or email from an angry customer who finds the problem before you do? How can you be confident that nothing will go wrong, and your customers will continue to trust your deliverables?

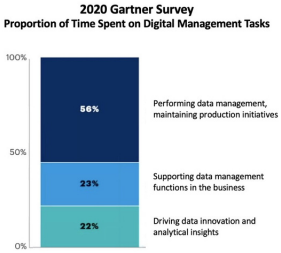
DataOps Observability can help you ensure that your complex data pipelines and processes are accurate and that they deliver what they were designed to do. Observability will also validate that your data science models, reports, and other parts of the data value chain are performing as expected. This solution sits on top of your existing infrastructure, without replacing staff or systems, to monitor your data operations. And it can alert you to problems before anyone else sees them. DataOps Observability is a logical first step in implementing DataOps in your organization. It tackles the problems you are facing right now, and prepares you for a future of full DataOps automation.

ERRORS HAPPEN; DO YOU REACT OR PREVENT?

Sure enough, Jason gets the call he's been dreading. There will be a problem, customers will be angry, he'll have to work late. But it's worse than that! This call is from the CEO, his boss's boss, and it's only 7:30 a.m. He breaks into a sweat and answers the phone. Through the yelling, he learns that a compliance report, sent to the board, government regulators, and critical business partners, was empty! The pipeline process finished with no problems, on time. But no data. He makes promises, ends the call, and cancels the rest of his day. He gets 26 of his best and brightest engineers on a video call to begin troubleshooting. At 1:00 p.m., the ad hoc team has found the root cause: a change to a source file resulted in passing a blank field through the pipeline. At 2:45, they have a fix implemented on one engineer's machine, but it will take six weeks to get that fix deployed to the production pipeline. At 3:00, he takes a deep breath and calls the CEO to break the news.

Sound familiar? When was the last time this happened to you? Today's data and analytic systems are complex, made up of any number of disparate tools and data stores. Crisis happens, errors are inevitable, and they are not only a huge embarrassment but also can affect business performance. Jason's experience is not a unique one. A 2019 DataKitchen/Eckerson survey found that 79% of companies have more than three data-related errors in their pipelines per month. A 2020 Gartner survey revealed that 56% of engineer time is spent on operational tasks and addressing errors, not on innovations and customer requests.





Many companies are stuck in a culture of "hope and pray" that their changes and integrations won't break anything and a practice of "firefighting" when things inevitably go wrong. They wait for customers to find problems. They blindly trust their providers to deliver good data on time without changing data structures. They interrupt the daily work of their best minds to chase and fix a single error in a specific pipeline, all the while not knowing if any of the other thousands of pipelines are failing.

It's a culture of productivity drains, which results in customers losing trust in the data. Bad data reaches the customer because companies haven't invested enough, or at all, in testing, automation, and monitoring. They have no way to catch errors early. And with engineers jumping around to put out the fires, their already demanding workloads are deprioritized. Customer delivery dates are missed, new features cannot be integrated, and innovations stop coming. As customer confidence in the data and analytics decreases, so does customer satisfaction.

Another result is rampant frustration within data teams. A survey of data engineers conducted by DataKitchen in 2022 revealed some shocking statistics. Data engineers are the people building pipelines, as well as doing data processing and data production. And they are suffering: 78% feel they need to see a therapist, and a similar number have considered quitting or switching careers. The industry is experiencing a shortage of people who do data work due to a supply problem as well as the exodus of people leaving the field because it's so stressful.leaving the field because it's so stressful.

Data Teams are Suffering

- 52% hope and pray that things don't break
- 78% of Data Engineers are so stressed they need a therapist.
- 70% expect to change jobs within a year.
- 79% have considered switching careers entirely.

Source: DataKitchen & data.world survey of data engineers, 2022

WHY HAVEN'T THEY SOLVED IT?

Jason follows the progress on the production fix very closely and updates the CEO at every stage. Naturally, the CEO wants to know why it takes so long to resolve the issue. "It's just code; it's what you guys do every day! Can't you patch it or something?" When the fix is fully tested and deployed to the production pipeline, Jason has time to reflect. He'll conduct a post mortem with his team on the specific issue, but he wants to find a solution to the bigger problem. Why has the company never addressed this kind of risk? What can he introduce into his data infrastructure and development cycle to ensure these things don't happen again?

You're just too busy. You'll get to it next year...right? The situation is so bad, your team is stressed, and things are breaking left and right. So, why hasn't your company, and many others, solved these problems? There are several reasons.

Teams are obviously very busy. They have backpacks full of customer requests, and they go to work every day knowing that they won't and can't meet customer expectations. The work that they are able to complete may sit for a while because any change to the complicated architecture or fragile pipeline may break something. When a change finally deploys, the teams have no way to see the whole landscape to ensure everything continues to run smoothly. Even if they could look at everything across their complex data operations, they don't know what and where to check for dependencies, failures, and data inconsistencies.

Obstacles to Fixing the Problems in Your Data Operations

No time and resources Teams are already busy and stressed and are not meeting customer expectations.Low appetite for change Teams have complicated in-place data architectures and tools, and they fear changes to what is already running.No single pane of glass Teams have no ability to see across all tools, pipelines, jobs, processes, datasets, and people.No insight on testing Teams don't know what, where, and how to check operations to ensure that the outputs are right. Lots of blame and shame Teams are panicked when customers find problems and spend time, without a shared context, trying to find out who is responsible.

And then there are the considerable numbers of teams and data processes involved. When something goes wrong, there is a lot of finger pointing and blame to go around. Each team knows its small part of the landscape, but there is no shared context among them of the larger picture, its infrastructure and problem areas, to adequately and efficiently address the issues.

It's a big risk if your teams can't answer basic questions about production processes or if the answers are found in pockets of your organization with teams who have no way to communicate broadly. Do they know if the data is going to arrive on time? Do they know if integrations with other datasets are right? Without a single source of truth about what's running, how can you pinpoint the failures? How can you find problems before your customers do?

Teams Can't Answer Basic Questions

- Did the data management job finish successfully and on time?
- Is my output data correct?
- Are the dashboard, report, dataset being used?
- What resources did the process consume?
- Did my source data arrive on time?
- Is my source data current, and is the data quality as expected?
- Did job X run after all the jobs in group A were completed?
- How many jobs ran yesterday; how long did they take?
- How many jobs will run today; how long will they take?
- Is there a pipeline with frequent or intermittent errors?

There's a similar risk when teams do not understand the status of development processes and deployments to production. Do your teams know how and when deployments are happening? Do they know what processes and environments are affected by a deployment? Do they know what changes are going into production? Not knowing these answers makes it very hard to ensure that everything is going well and even harder to find problems when they occur.

The industry is replete with data tools, automation tools, and IT monitoring tools. But none of these tools fully addresses the problem. You have to monitor your entire data estate and all the reasons why pipelines may succeed or fail, not just server performance, load testing, and tool and transaction testing. A complete Observability solution is required.

Tool/Component Options	Azure	AWS	GCP	Third Party/others
ETL/Orchestration	Data Factory	Glue, DataPipeline	DataFlow, Composer (Airflow)	Talend, Informatica, IBM DataStage
Big Data ETL/Orchestration	DataBricks	Glue, Kinesis, EMR	PubSub, Composer (Airflow), Cloud DataFusion	Airflow, Streamsets, FiveTran
Data Science	Python, Tensorflow, DataBricks	Python, Sagemaker	Python, AI Platform, Tensorflow	DataRobot, IBM Watson
DevOps	Azure Devops	Code Deploy, Code Pipeline	CloudBuild	DBT, Delphix, Puppet
Version Control	GitHub, Azure Repos (git)	CodeCommit (git)	Cloud Source Repositories (git)	GitHub, GitLab, Bitbucket
Secret Store	Azure Key Vault	Secrets Manager	Google Secret Manager	Hashicorp
Storage	ADLS	S3	GCS	IBM Cloud Object Storage
Databases	Cosmos, SQL Server, Synapse	Redshift, Aurora, DocumentDB	Big Query, Postgres, Big Table	Snowflake, Teradata, Vertica
Analytic Tools	PowerBI	QuickSight	Looker, DataLab, DataStudio	Qlik, Tableau, Thoughtspot, Cognos
Infra automation	Terraform	Cloud Formation, Chef, Puppet	Terraform, Chef, Procurement Manager	Chef, Ansible
Data catalog	Azure Data Catalog	LakeFormation, Glue	Google Data Catalog	Collibra, Watson Catalog

This is where DataOps Observability fills a gap in the industry.

DATAOPS OBSERVABILITY TO THE RESCUE

Jason knows he needs to implement changes, but what and where? The more he reads about DataOps, the more he knows his company needs to make the transition. He thinks he can sell his boss and the CEO on this idea, but his pitch won't go over well when they still have more than six major data errors every month. He wonders what it would take to create a dashboard that could monitor his pipelines and alert him to potential problems or track negative trends before he gets the next dreaded call.

When considering how organizations handle serious risk, you could look to NASA. The space agency created and still uses something called "mission control" with many screens sharing detailed data about all aspects of a space flight. That shared information is the basis for monitoring mission status, making decisions and changes, and then communicating to all people involved. It is the context for people to understand what's going on in the moment and to review later for improvements and root cause analysis.

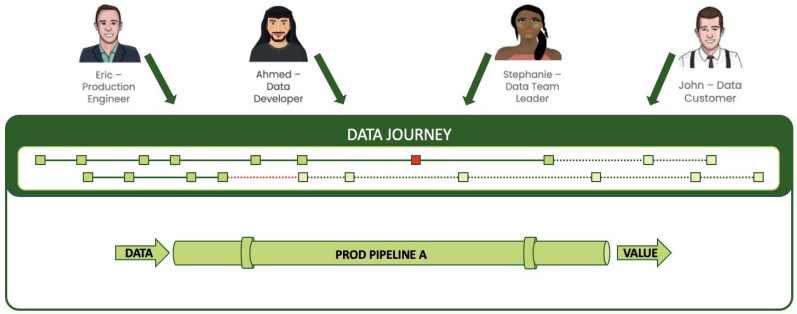
Any data operation, regardless of size, complexity, or degree of risk, can benefit from DataOps Observability. Its goal is to provide visibility of every journey that data takes from source to customer value across every tool, environment, data store, data and analytic team, and customer so that problems are detected, localized, and raised immediately. DataOps Observability does this by monitoring and testing every step of every data and

The goal of DataOps Observability is to provide visibility of every journey that data takes from source to customer value across every tool, environment, data store, data and analytic team, and customer so that problems are detected, localized, and raised immediately.

DataOps Observability does this by monitoring and testing every step of every data and analytic pipeline in an organization, in development and production, so that teams can deliver insight to their customers with no errors and a high rate of innovation. It relies on a hierarchy of data journeys, or representations of actual pipelines, that observe and track the processes within the end-to-end value chain of the data. Data journey observability is the first of two steps in implementing DataOps. It tackles the immediate challenges in your data operations by providing detailed information about what's going on right now. You need to sort out the current state of your data enterprise before you focus on the iterative development, automation, and customer value goals of full DataOps transformation.

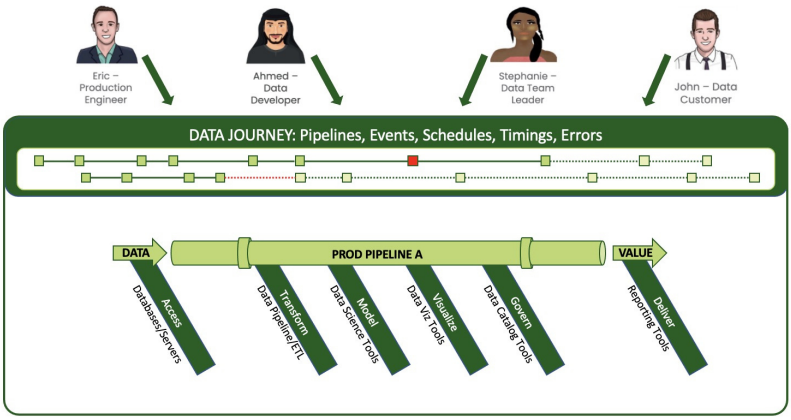
Jason considers his dashboard idea but quickly realizes the complexity of building such a system. It's not just a fear of change. It's because it's a hard thing to accomplish when there are so many teams, locales, data sources, pipelines, dependencies, data transformations, models, visualizations, tests, internal customers, and external customers. Too many moving parts, he thinks, but there has to be some way to prove things work before my customers see them!

DataOps Observability works because it visualizes data journeys that span all of these moving parts, beginning with the people. Take four main constituents in your operations: production engineers, data developers, data team managers, and customers. They all have different roles and different relationships with the data. Data journeys give them all a shared context about data operations, what pipelines are running or will run, the status and quality of those pipelines, what development is in progress, and where it will be deployed.



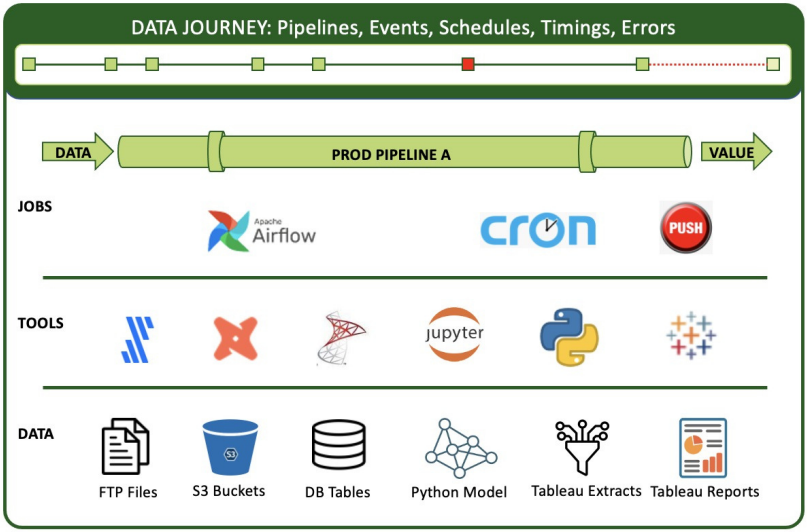
All constituents have a shared view of your data operation.

And when these stakeholders can see a shared view of a single pipeline, they know everything that's happening during a run of that pipeline. The journey reveals all of the complex steps, toolchains, and actions within the data workflow, most importantly, where things go wrong.



A journey spans complex steps and toolchains in every pipeline and signals failures.

For example, in a single pipeline you might have some FTP file sources that you ingest into S3 buckets. That data then fills several database tables. A Python model runs, and you deliver some Tableau extracts that publish to Tableau reports. Your "simple" pipeline involves a toolchain that features Fivetran, DBT, SQL, a Jupyter notebook, and Tableau. And to run everything, you use a wrapper like Airflow or a cron job or a manual procedure. With DataOps Observability, you can integrate the full context of all of these elements into a data journey that monitors the stack.



Journeys provide a context for the many complex elements of a pipeline.

A JOURNEY MAPS AND OBSERVES EVERYTHING

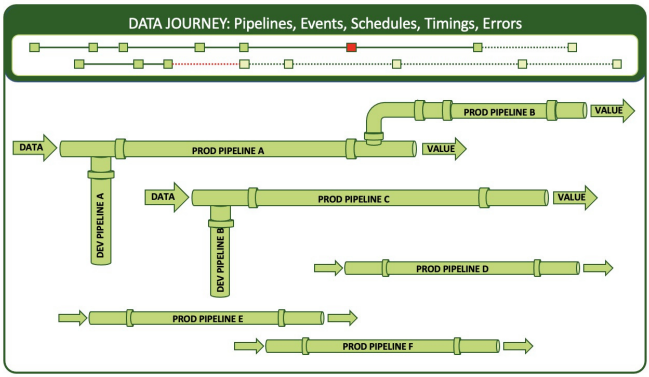
The journey tracks all levels of the stack from data to tools to code to tests across all critical dimensions. It supplies real-time statuses and alerts on start times, processing durations, test results, and infrastructure events, among other metrics. And if you're armed with this information, you can know if everything ran on time and without errors and immediately identify the specific parts that didn't .As valuable as this visibility is to this single pipeline and its users, most organizations have large numbers of pipelines that need this level of observation.

Jason realizes that he's developed something similar before. In his role in Marketing as Customer Experience Manager, he built customer journey maps to track customer touchpoints with their brand and products. That effort guided improvements in their sales and marketing programs and produced a huge increase in customer engagement. So why can't he do the same thing for the many journeys that his data takes from source to customer value?

DataOps Observability journeys are current and future representations of data stores, processes, pipelines, or groups of pipelines and their upstream and downstream dependencies.

Every data operation is a kind of factory, some with hundreds of pipelines, each made up of a combination of tools, technologies, and data stores. As a result, there are thousands of tools, and each tool has code that's acting on the data in some way. Even more challenging is the fact that these pipelines are anything but uniform. Some are batched, some are streaming, some are scheduled or triggered from a dependency, while some are entirely manual. And they all may have different deployment methods and environments. Further, the pipelines are owned and operated by very different teams in different departments within your company.

The good news is that a single data journey can span multiple, related pipelines to track all of the data and analytic infrastructure within them.



A data journey spans and tracks multiple pipelines.

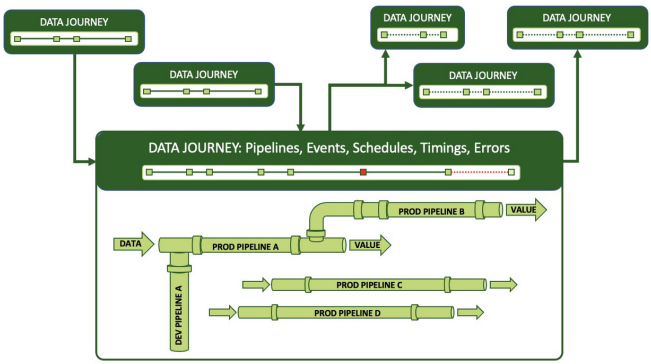
NO JOURNEY EXISTS IN A VACUUM

A data journey can “see” all of this because it is a meta-structure that goes beyond typical test automation, application performance monitoring, and IT infrastructure monitoring software. While quite valuable, these solutions all produce lagging indicators. With these tools you may know that you are approaching limits on disk s pace, but you can’t know if the data on that disk is correct. You may know that a particular process has completed but you can’t know if it completed on time or with the correct output. You can’t quality-control your data integrations or reports with only some of the details.

Since data errors happen with more frequency than resource failures, data journeys provide crucial additional context for pipeline jobs and tools and the products they produce. They observe and collect information, then synthesize it into coherent views, alerts, and analytics for people to predict, prevent, and react to problems.

Jason takes a well-deserved break and meets up with his colleague Maria, Manager of Data Science, at the local coffee shop. He decides to run his data journey map idea by his friend. Maria listens carefully, nodding in agreement, then asks a question. “Will this help me know when my scientists are going to get new feeds of sales data from your engineers?” Jason frowns. What data feed? How did he not know about this downstream use?

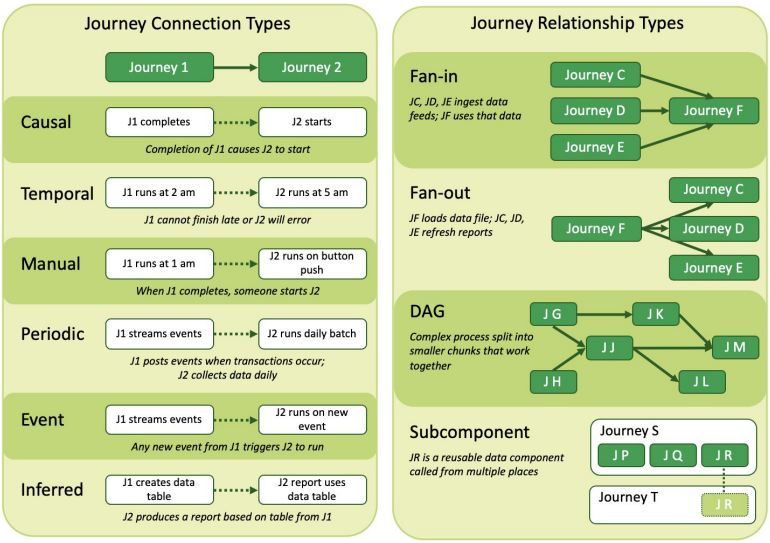
This lack of visibility is why data journeys track and collect information across all levels of your organization. Starting small, they simplify the complex steps within pipelines. You can define journeys to represent the process “chunks” you truly care about, ignoring the noise of more granular details. Then expanding their scope, journeys also represent and track complex relationships among all your pipelines where no connections are currently coded. To accomplish this, you can set up relationships among the journeys themselves. Imagine being able to pull together contexts from across the organization where data is shared or processes are dependent on inputs and transformations.



Links defined among journeys can organize and observe typically undocumented relationships.

So often these relationships among data pipelines are tribal knowledge, rather than concrete and actionable connections. But when there's a problem, you need to know how these relationships could amplify it.
By identifying how pipelines actually connect via data dependencies and relate through shared data use across your data estate, you can build relationships among the representational journeys to reflect them.

- Build in causal connections, where one pipeline starts after another completes.
- Build in temporal connections, where two pipelines are scheduled to run separately but have dependencies. If the first is late finishing there are problems.
- Build in event-driven connections where streaming pipelines get data and run based on specific events.
- Build in a fan-in relationship where those same event-driven pipelines finish at the end of the day, fill up S3 buckets and build tables in Snowflake. Then another process takes those tables and builds a star schema, producing some reports.
- Build in a sub-component relationship where you have a pipeline containing reusable sub-pipelines.



Journeys can map links among your pipelines that are not recorded elsewhere.

CONSTRUCTING EFFECTIVE JOURNEYS

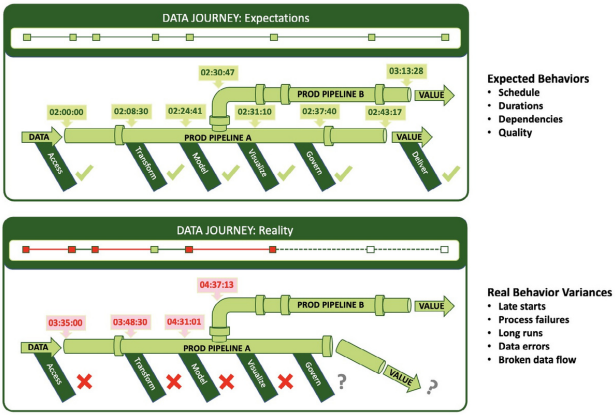
As he thinks through the various journeys that data take in his company, Jason sees that his dashboard idea would require extracting or testing for events along the way. So, the only way for a data journey to truly observe what's actually happening is to get his tools and pipelines to auto-report events.

An effective DataOps observability solution requires supporting infrastructure for the journeys to observe and report what's happening across your data estate. Observability includes the following components.

- Functionality to set pipeline expectations
- Logs and storage for problem diagnosis and visualization of historical trends
- An event or rules engine
- Alerting paths
- Methods for technology/tool integrations
- Data and tool tests
- An interface for both business and technical users

SETTING EXPECTATIONS

In addition to the tracking of relationships and quality metrics, DataOps Observability journeys allow users to establish baselines and concrete expectations for run schedules, run durations, data quality, and upstream and downstream dependencies. Observability users can then see and measure the variance between expectations and reality during and after each run. This aspect of data journeys gives users real-time information about what is happening now or later today and if data delivery will be on time or late. It can feed a report or dashboard that lists all datasets and when they arrived, as well as the workflows and builds that are running.

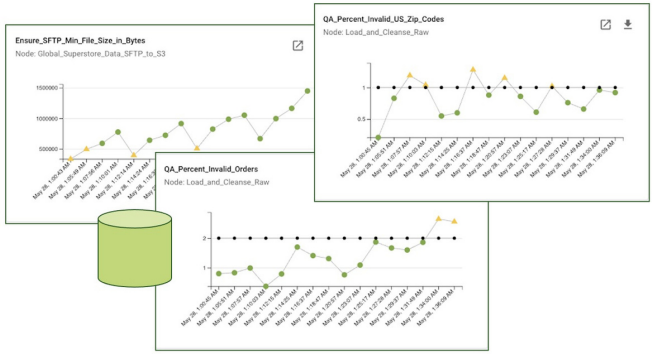


By setting process expectations, journeys can identify variances in each pipeline run.

With this information in a shared context, your analyst working on a data lake will know if the 15 datasets she is viewing are accurate, the most recent, or of the same date range. And she'll know when newer data will arrive. She can work more efficiently knowing when to conduct her analyses and what delivery date to communicate to her customers.

STORING RUN DATA FOR ANALYSIS

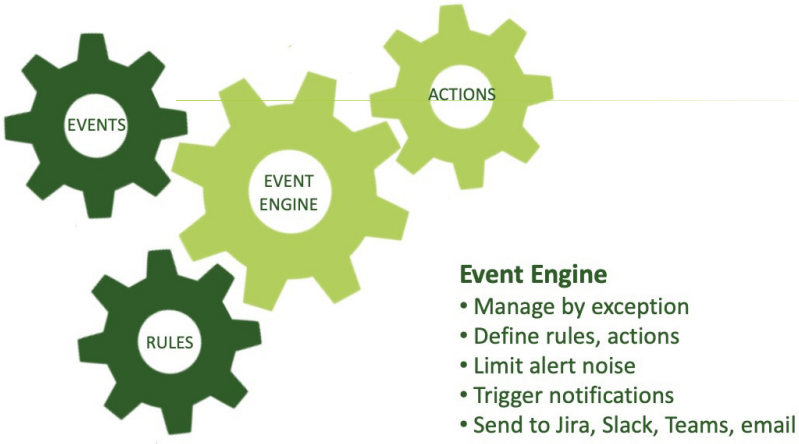
Real-time details are not the only purpose for setting and measuring against expectations. DataOps Observability must also provide a store of run data over time for root cause diagnosis and statistical process control analysis. It's not just about what is happening during the operation; it's also about what happens after the operation. By storing run data over time, Observability allows your analysts to look at variances from an historic perspective and make adjustments. Finding trends and patterns can help your team optimize your data pipelines and even predict and prevent future problems.



Storing data about your journeys allows you to analyze, learn, and predict.

IDENTIFYING EVENTS AND ACTIONS

So DataOps Observability captures all this information and stores it in a database, but that ends up being a lot of data that someone has to monitor and sift through. Fortunately, Observability includes an event engine that can react to what's happening in the journeys and cut down the signal-to-noise ratio. Given a set of rules, the event engine can trigger notifications through email or to Slack and other services, send alerts to your operations people, issue commands to start or end pipeline runs, and prompt other actions when the difference between expected results and actual runs exceeds your thresholds of tolerance.



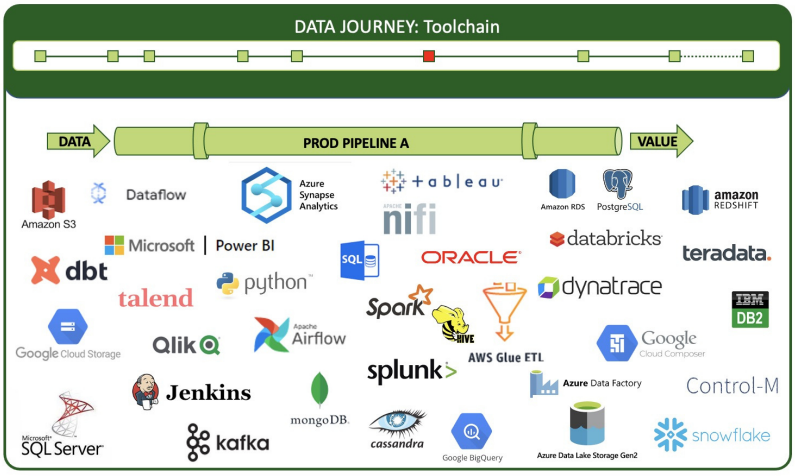
With this actionable data combined with an understanding of the bigger context of your data estate, you can empower your employees to stop production pipelines before problems escalate.

ALERTING ON ALL CRITICAL DIMENSIONS

While alerts on run schedules, durations, dependencies, and data tests are valuable, these metrics alone do not tell the whole story of every data journey. After all, pipelines execute on infrastructure, and that infrastructure has to be managed. To that end, DataOps Observability must also accept logs and events from IT monitoring solutions like DataDog, Splunk, Azure Monitor, and others. Observability can consolidate the information into critical alerts on the health of data pipelines, providing clear visibility into the state of your operations.

INTEGRATING YOUR TOOLS

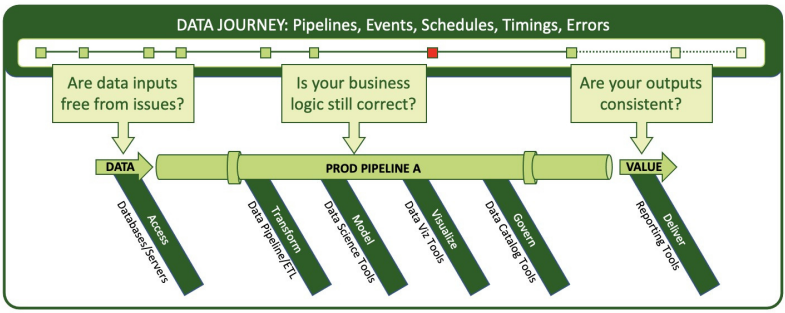
DataOps Observability has to support any number and combination of tools because your toolchain is potentially monumental and complex. With an OpenAPI specification, DataOps Observability can offer easy connections. Even better, pre-built integrations with your favorite tools make it simple to start transmitting events to and receiving commands from Observability right away.



DataOps Observability integrates with all technologies in your entire toolchain.

TESTING AT EVERY STEP

You want to track more than the schedule and duration of your runs. You want to check that the contents of your pipelines; the data, the models, the integrations, the reports, the outputs that your customers see-are as accurate, complete, and up to date as expected. You want to find errors early. To do that, your journeys need to include automated tests. Tests that evaluate your data and its artifacts work by checking things like data inputs, transformation results, model predictions, and report consistency. They run the gamut from typical software development tests (unit, functional, regression tests, etc.) to custom data tests (location balance, historical balance, data conformity, data consistency, business logic tests, and statistical process control). Read more about DataOps testing in A Guide to DataOps Tests or Add DataOps Tests for Error-Free Analytics.



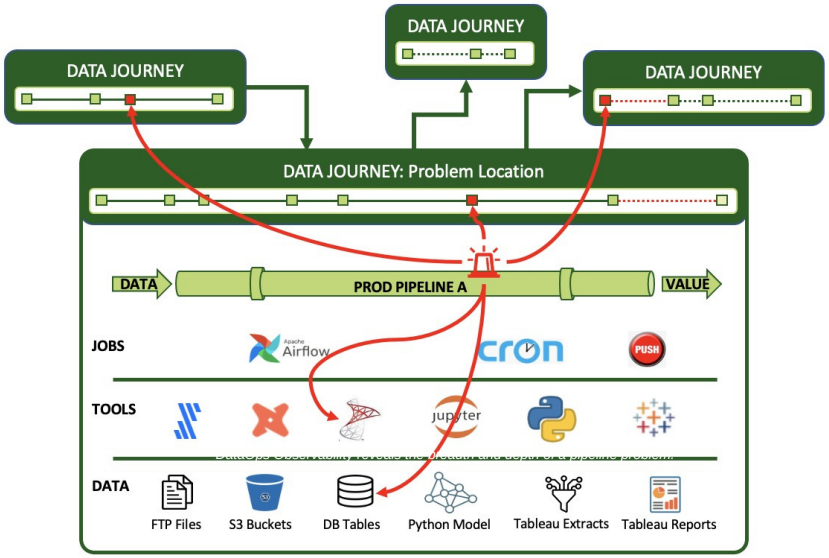
Add automated tests at every step in a pipeline and transmit results to the event engine.

Your organization already has many sources of tests, which can transmit vital information to data journeys with rules for taking action. You may have a database that has its own test framework, your data engineers may have written some SQL tests, your data scientists may have written Python tests, you may have test engineers writing Selenium test suites, you may be using DataKitchen DataOps TestGen or Automation for testing. All usable information.

PACKAGING IT IN A FRIENDLY, FLEXIBLE UI

Finally, DataOps Observability must make it easy for multiple types of constituents to monitor what affects their jobs. Your production engineer needs to check on operations. Your data developers need to verify that the changes they make are working. Your data team manager needs to measure her team's progress. Your business customer needs to know when the analytics will be available for his projects. They all require different views of the same data.

DataOps Observability has to provide interface views based on differing roles. The greatest benefit of the Observability UI is that it can reveal the breadth of a data estate with the elements and relationships that are vital to any given stakeholder. And, at the same time, it offers a way to plumb the depths of the complex layers of any pipeline to locate problems. It serves as a springboard for investigating and fixing issues with insight as to where those issues may reverberate across the enterprise.



Together, the journeys, the expectations, the data store, the event engine, the rules and alerts, the integrated tools, the tests, and a robust interface all enable complete observability. All of these elements come together to inform you of where the problems are, minimize the risk to customer value, provide a shared context for all constituents to know what's happening and what's going to happen, and alert the right people when something goes wrong.

DATAOPS OBSERVABILITY BENEFITS SUMMARY

Jason formed a small team and implemented DataKitchen DataOps Observability. The team built a data journey and monitored a subset of pipelines as a proof of concept. When he presented some initial data to a group of stakeholders, his audience was surprised at the errors revealed, given that only the critical, customer-facing errors had garnered their attention to date. Jason showed them how DataOps Observability could monitor, alert him to problems, prompt early fixes of these issues, and ultimately prevent many of these errors.

DataOps Observability is your mission control for every journey from data source to customer value, from any development environment into production, across every tool, every team, every environment, and every customer so that problems are detected, localized, and understood immediately.
In a world of complexity, failure, and frustration, data and analytics teams need to deliver insight to their customers with no errors and a high rate of change. As in Jason's experience, the stress and embarrassment of breaking things crowd out the ability to create new insights. In a 2021 survey of 600 data professionals, responses suggested an overwhelming majority are calling for relief. In the survey, 97% reported experiencing burnout, 91% reported frequent requests for analytics with unrealistic or unreasonable expectations, and 87% reported getting blamed when things go wrong.

You don't have to live with these problems. DataOps Observability provides critical features and benefits.

- Data journeys allow monitoring of every data process from source to customer value.
- Production expectations including data and tool testing reduce embarrassing errors to zero.

- Development data and tool testing increase the delivery rate and lower the risk of
- deploying new analytic insights.
- Historical dashboards enable you to find the root causes of issues.
- An intuitive, role-based user interface allows stakeholders, IT engineers, managers, data engineers, data scientists, analysts, an d your business customers,
- to be on the same page.
- Simple connections and an open API enable fast integrations without replacing
- your existing tools.

When you aim to produce rapid, trusted customer insight, you need to start by reducing your team's hassles and embarrassment while increasing the time it has to develop and deliver. By monitoring all data journeys, you can lower your error rates and achieve your goals. Do not replace your tools or infrastructure; use the DataKitchen DataOps Observability product on top of those tools. When you find a problem, use the DataKitchen DataOps Automation product to fix it permanently by automating the testing, deployment, and orchestration of all your chosen technologies.

Spend less time worrying about what may go wrong and gain more time to create by observing the entire data journey and taking early action to stay on track.

DataOps Chicken Wings

by Nick Bracy

INGREDIENTS AND TOOLS

- 1/4 cup of soy sauce
- 3 tbsp of sesame oil
- 1 1/2 tbsp siracha sauce
- Freshly grated ginger and garlic
(I use about 2-3 tsp of minced garlic — 2 cloves)
- Ginger *(can be overpowering, a grated piece of ginger that is about a square inch is good, about the size of the top part of your thumb—from knuckle to the top of your thumb)*
- 1 tbsp of rice wine vinegar
- A generous tbsp of organic honey or agave
- Black pepper to taste
- 1/2 cup of ketchup (optional)

INSTRUCTIONS

1. Place chicken drums and wings in a large zip-lock back, add marinade, seal zip-lock bag, mix contents of the bag around gently (you don't want to accidentally open the bag and marinate your kitchen floor or counter), make sure your chicken is well coated inside the bag.
2. Refrigerate your chicken in the marinade for 8-24 hours *(You can also just cook them right away if you don't have the time)*
3. Best slow cooked for 5-6 hours in a crockpot or on 225 degrees in a conventional oven—use all the contents in the bag. *(If you don't have that kind of time, bake at 400 degrees Fahrenheit.)* 3.5 lbs. of chicken should bake for 55-60 minutes; 4.5 lbs. of chicken requires 60-65 minutes.

Recipe inspired by Joe DeFran

Second DataOps For EVERYONE, EVERYWHERE

DataOps for the Chief Data Officer

Warring Tribes into Winning Teams: Improving Teamwork in Your Data Organization

If the groups in your data-analytics organization don't work together, it can impact analytics-cycle time, data quality, governance, employee retention and more. A variety of factors contribute to poor teamwork. Sometimes geographical, cultural and language barriers hinder communication and trust. Technology-driven companies face additional barriers related to tools, technology integrations and workflows which tend to drive people into isolated silos.

THE WARRING TRIBES OF THE TYPICAL DATA ORGANIZATION

The data organization shares a common objective; to create analytics for the (internal or external) customer. Execution of this mission requires the contribution of several groups shown in Figure 30. These groups might report to different management chains, compete for limited resources or reside in different locations. Sometimes they behave more like warring tribes than members of the same team.

Figure 30: Delivery of analytics (the value chain) to customers requires contributions from several groups in the data organization

Let's explore some of the factors that isolate the tribes from one another. For starters, the groups are often set apart from each other by the tools that they use. Figure 31 is the same value chain as above but reconstructed from the perspective of tools.

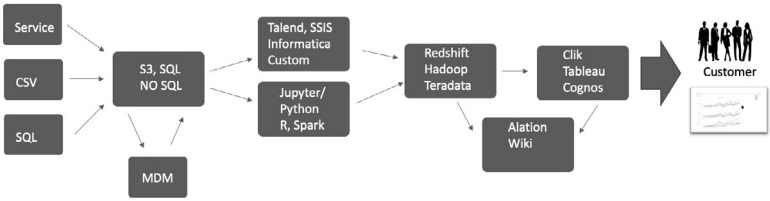


Figure 31: The value chain shown from a tools perspective

To be more specific, each of the roles mentioned above (figure 30) view the world through a preferred set of tools (figure 31):

- Data Center/IT — Servers, storage, software
- Data Science Workflow — Kubeflow, Python, R
- Data Engineering Workflow — Airflow, ETL
- Data visualization, Preparation — Self Service tools, Tableau, Alteryx
- Data Governance/Catalog (Metadata management) Workflow — Alation, Collibra, Wikis

The day-to-day existence of a data engineer working on a master data management (MDM) platform is quite different than a data analyst working in Tableau. Tools influence their optimal iteration cycle time, e.g., months/weeks/days. Tools determine their approach to solving problems. Tools affect their risk tolerance. In short, they view the world through the lens of the tools that they use.

The division of each function into a tools silo creates a sense of isolation which prevents the tribes from contemplating their role in the end-to-end data pipeline. The less they understand about each other, the less compelling the need to communicate about actions taken which impact others. Communication between teams (people in roles) is critical to the organization's success. Most analytics requires contributions from all the teams. The work output of one team may be an input to another team. In the figure below, the data (and metadata) build as the work products compound through the value chain.

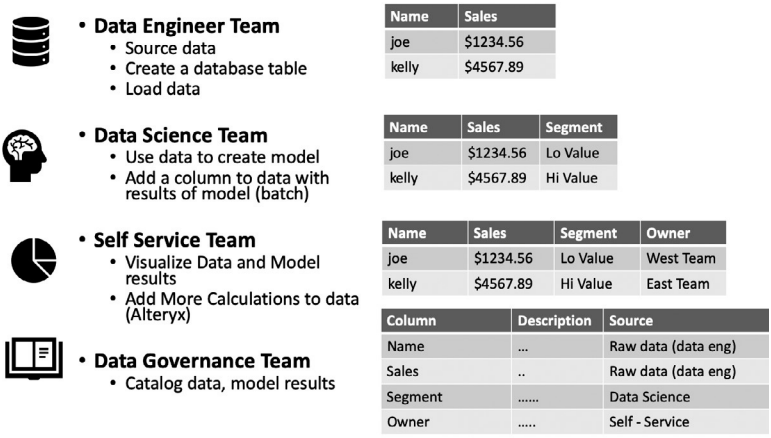


Figure 32: Each group adds unique value to analytics. In most cases, the work of one group is an input to the next group.

In many enterprises, there is a natural tendency for the groups to retreat into the complexity of their local workflow. In figure 33, we represent the local workflow of each tribe with a directed-acyclic graph (DAG).

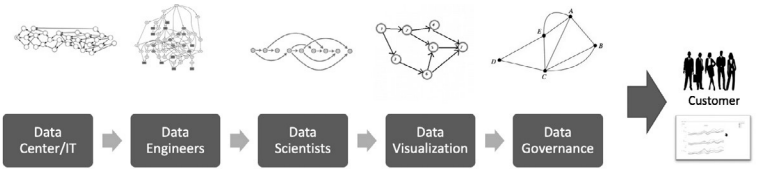


Figure 33: Work groups tend to focus on the complexity of their local workflow

It is too easy to overlook the fact that the shared purpose of these local workflows is to work together to publish analytics for end-customers.

OTHER FACTORS THAT INCREASE GROUP ISOLATION

Group isolation is also induced by platforms, release cadence and geographic locations. The example below shows a multi-cloud or multi-data center pipeline with integration challenges.

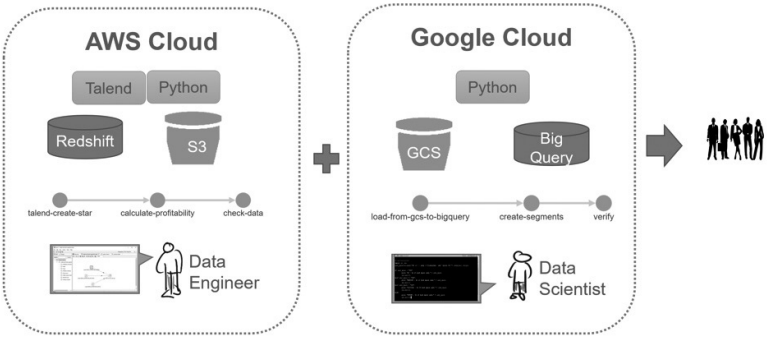


Figure 34: Multi-cloud or multi-data center pipelines with integration challenges

The two groups managing the two halves of the solution have difficulty maintaining quality, coordinating their processes and maintaining independence (modularity). Group one tests part one of the system (figure 35). Group two validates part two. Do the part one and two tests deliver a unified set of results (and alerts) to all stakeholders? Can tests one and two evolve independently without breaking each other? These issues repeatedly surface in data organizations.

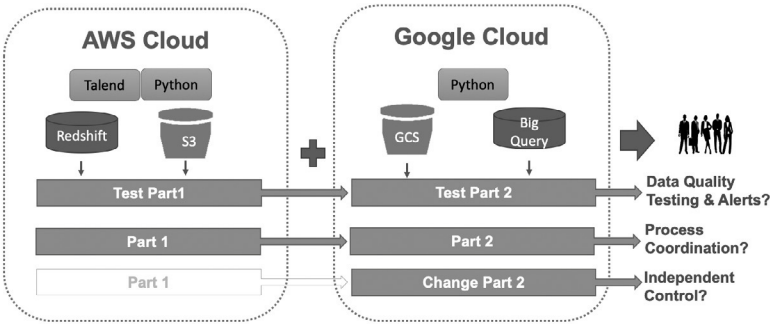


Figure 35: Integration challenges of multi-cloud or multi-data center solutions

In another example, assume that two groups are required to work together to deliver analytics to the VP of marketing. The home office in Boston handles data engineering and creates data marts. Their iteration period is weekly. The local team in New Jersey uses the data marts to create analytics for the VP of Marketing. Their iteration is daily (or hourly).

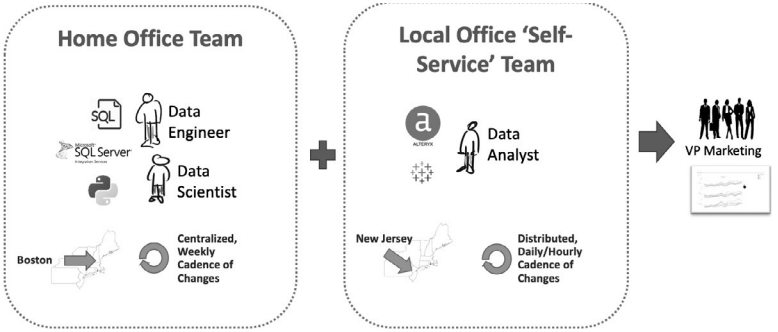


Figure 36: Issues related to multi-team workflows

One day, the VP of Marketing requests new analytics (deadline ASAP) from the data analysts for a meeting later that day. The analysts jump into action, but face obstacles when they try to add a new data set. They contact data engineering in Boston. Boston has its own pressures and priorities and their workflow, organized around a weekly cadence, can't respond to these requests on an "*ASAP*" basis.

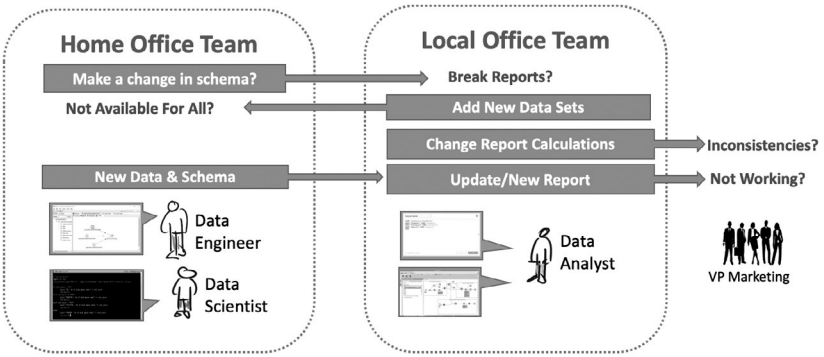


Figure 37: Challenges with multi-team coordination

The home office team in Boston finally makes the needed changes, but they inadvertently break other critical reports (figure 37). Meanwhile, out of desperation, the New Jersey team adds the required data sets and updates their analytics. The new data sets are only available to New Jersey, so other sites are now a revision behind. New Jersey's reports are inconsistent with everyone else's. Misunderstandings ensue. It's not hard to imagine why the relationship between these groups could be strained.

These challenges may seem specific to data organizations, but at a high level, everything that we have discussed boils down to poor communication and lack of coordination between individuals and groups. As such, we can turn to management science to better understand the problem and explore solutions.

RELATIONAL COORDINATION

Strip away the technological artifacts from the situations described above and you are left with an organization that cannot foster strong role relationships and communication between employees. These challenges are not unique to technology-driven organizations. Many enterprises across a wide variety of industries face similar issues.

For those who don't remember, the airline business in the 1980s and 1990s was brutally competitive, but during this same period, Southwest Airlines revolutionized air travel. By the early 2000s, they had experienced 31 straight years of profitability and had a market capitalization greater than all the other major US airlines combined. Brandeis management professor Jody Hoffer Gittell investigated the factors in Southwest Airlines' performance and, back in 2003, published a quantitative, data-driven analysis shedding light on Southwest's success.

Dr. Gittell surveyed the major players in the airline industry and found a correlation between key performance parameters (KPP) and something that she termed Relational Coordination (RC), the way that relationships influence task coordination, for better or worse. "Relational coordination is communicating and relating for the purpose of task integration — a powerful driver of performance when work is interdependent, uncertain and time constrained."

In her study, higher RC levels correlated with better performance on KPPs, even when comparing two sites within the same company. Since that time RC has been applied in industries ranging from healthcare to manufacturing across 22 countries.

One common misconception is that RC focuses on personal relationships. While personal relationships are important, RC is more concerned with the relationship of roles and workflows within the organization. RC studies how people interact and exchange information in executing their role-based relationships.

Relational Coordination can be expressed as characteristics of relationships and communication:

Dimensions of RC	Low RC	High RC
Relationships	Functional Goals Exclusive Knowledge Lack of Respect	Shared Goals Shared Knowledge Mutual Respect
Communication	Infrequent Delayed Inaccurate Finger-pointing	Frequent Timely Accurate Problem-solving

Members of the "Low-RC" organization express their goals solely in terms of their own function. They keep knowledge to themselves and there may be a tendency for one group to look down upon another group. Inter-group communication is inadequate, inaccurate and might be more concerned with finding blame than finding solutions. As expected, the "High-RC" organization embodies the exact opposite end of this spectrum.

"High-RC" team members understand the organization's collective goal. They not only know what to do but why, based on a shared knowledge of the overall workflow. Everyone's contribution is valued, and no one is taken for granted. There is constant communication, especially when a problem arises.

At this point you may be thinking: "OK fine, this is all *touchy-feely* stuff. I'll try to smile more and I'll organize a pizza party so everyone can get to know each other." Maybe you should (smiling will make you feel good and parties are fun after all), but our experience is that the good feeling wears off once the last cupcake is gone and the mission-critical analytics are offline.

How do you keep people working *independently* and *efficiently* when their work product is a *dependency* for another team? How can one team *reuse* the data or artifacts or code that another team produces?

For most enterprises, improving RC requires foundational change. You need to examine your end-to-end data operations and analytics-creation workflow. Is it building up or tearing down the communication and relationships that are critical to your mission? Instead of allowing technology to be a barrier to Relational Coordination, how about utilizing automation and designing processes to improve and facilitate communication and coordination between the groups? In other words, you need to restructure your data analytics pipelines as services (or microservices) that create a robust, transparent, efficient, repeatable analytics process that unifies all your workflows.

BUILDING A HIGH-RC ENTERPRISE USING DATAOPS

DataOps is a new approach to data analytics that applies lean manufacturing, DevOps and Agile development methods to data analytics. DataOps unifies your data operations pipeline with the publication of new analytics under one orchestrated workflow.

Robust – Statistical process control (lean manufacturing) calls for tests at the inputs and outputs of each stage of the data operations pipeline. Tests also vet analytics deployments, like an impact review board, so new analytics don't disrupt critical operations.

Transparent – Dashboards display the status of new analytics development and the operational status of the data operations pipeline. Automated alerts communicate issues immediately to appropriate response teams. Team members can see a birds-eye-view of the end-to-end workflow as well as local workflows.

Efficient – Automated orchestration of the end-to-end data pipeline (from data sources to published analytics) minimizes manual steps that tie up resources and introduce human error. Balance is maintained between centralization and decentralization; the need for fast-moving innovation, while supporting standardization of metrics, quality and governance.
Repeatable – Revision control with built-in error detection and fault resilience is applied to the data operations pipeline.
Sharable and Chunkable – Encourage reuse, by creating a services oriented architecture (SOA) for your team to use together.

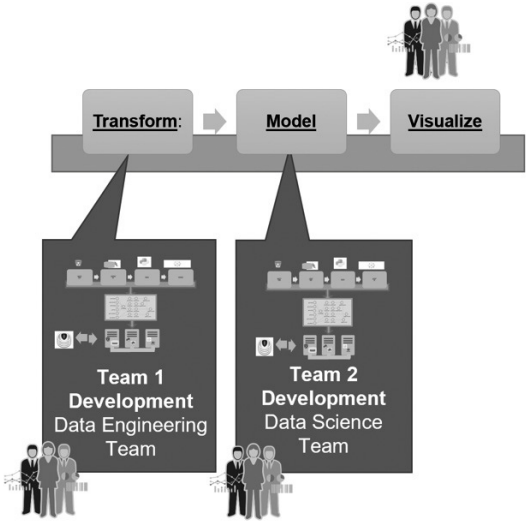


Figure 38: DataOps is a task coordination and communication framework that uses technology to break down the barriers between the groups in the data organization.

It may help to provide further concrete examples of a DataOps implementation and how it impacts productivity. Some of these points are further explained in our blog DataOps in Seven Steps:

• **Data Sharing** – data sources flow into a data lake which is used to create data warehouses and data marts. Bringing data under the control of the data organization decouples it from IT operations and enables it to be shared more easily.

• **Deployment** of code into an existing system – continuous integration and continuous delivery of new analytics, leveraging on-demand IT resources and automated orchestration of integration, test and deployment.

• **Environment** startup, shutdown – With computing and storage on-demand from cloud services (infrastructure as code), large data sets and applications (test environments) can be quickly and inexpensively copied or provisioned to reduce conflicts and dependencies.

• **Testing** of data and other artifacts – Testing of inputs, outputs, and business logic are applied at each stage of the data analytics pipeline. Tests catch potential errors and warnings before they are released so the quality remains high. Test alerts immediately

inform team members of errors. Dashboards show the status of tests across the data pipeline. Manual testing is time-consuming and laborious so it can't be done in a timely way. A robust, automated test suite is a key element in continuous delivery.

• **Reuse** of a set of steps across multiple pipelines – Analytics reuse is a vast topic, but the basic idea is to componentize functionalities as services in ways that can be shared. Complex functions, with lots of individual parts, can be containerized using a container technology (like Docker).

We have seen marked improvements in analytics cycle time and quality with DataOps. It unlocks an organization's creativity by forging trust and close working relationships between data engineers, scientists, analysts and most importantly, users. DataOps is a task coordination and communication framework that uses technology to break down the barriers between the groups in the data organization. Let's look at the DataOps enterprise from the perspective of Relational Coordination.

Dimension	Warring Tribes (Weak RC)	The DataOps Enterprise (Strong RC)
Shared Goals	-Separation into isolated tools silos with little regard for or understanding of others	-Visibility into how analytics builds in stages until delivery to the customers or users
Shared Knowledge	-System knowledge concentrated in the Impact Review Board -Little reuse of analytics components -Bureaucratic processes govern change -Little visibility into the end-to-end pipeline	-System knowledge implemented in tests that anyone can view -Reusable analytics components are maintained in source control -Rapid cycle time for new analytics -Complete visibility into the global and local workflows of the data pipeline
Mutual Respect	-Each tribe in the data-analytics organization thinks it is better than the others	-No one is taken for granted. The workflow shows how everyone's contribution is important
Frequent and Timely Communication	-Communication is limited and definitely not a priority during frequent high-severity outages	-Dashboards and automated alerts keep everyone informed 24x7
Problem-solving Communication	-When something goes wrong, the tribes focus on finding someone to blame	-Discussion centers on tests that caught or will catch problems

CONCLUSION

Technology companies face unique challenges in fostering positive interaction and communication due to tools and workflows which tend to promote isolation. This natural distance and differentiation can lead the groups in a data organization to act more like warring tribes than partners. These challenges can be understood through the lens of Relational Coordination; a management theory that has helped explain how some organizations achieve extraordinary levels of performance as measured by KPPs. DataOps is a tools and methodological approach to data analytics which raises the Relational Coordination between teams. It breaks down the barriers between the warring tribes of data organizations. With faster cycle time, automated orchestration, higher quality and better end-to-end data pipeline visibility, DataOps enables data analytics groups to better communicate and coordinate their activities, transforming warring tribes into winning teams.

DataOps Resolves the Struggle Between Centralization and Freedom in Analytics

Freedom and employee empowerment are essential to innovation, but a lack of top-down control leads to chaos. Self-service tools enable data analysts to create new analytics very quickly, but they can drift in different directions. Imagine a team of analysts building reports that tally sales figures and each come up with a different result. One approach includes drop shipments and sales from distributors/subsidiaries. Another report might consist of product sales, but not services. These different approaches each have their use case, but from a manager's perspective inconsistency creates the appearance of inaccuracy. You can't establish a shared reality when everyone has different numbers.

Some managers respond to this challenge by centralizing analytics. With data and analytics under the control of one group, such as IT, you can standardize metrics, control data quality, enforce security and governance, and eliminate islands of data. All worthy endeavors, however forcing analytic updates through a heavyweight IT development process is a sure way to stifle innovation. It is one of the reasons that some companies take three months to deploy ten lines of SQL into production. Analytics have to be able to evolve and iterate quickly to keep up with user demands and fast-paced markets. Managers instinctively understand that data analytics teams must be free to innovate. The fast-growing self-service tools market (Tableau, Looker, etc.) addresses this market.

Centralizing analytics brings it under control but granting analysts free reign is necessary to stay competitive. How do you balance the need for centralization and freedom? How do you empower your analysts to be innovative without drowning in the chaos and inconsistency that a lack of centralized control inevitably produces? Visit any modern enterprise, and you will find this challenge playing out repeatedly in budget discussions and hiring decisions. You might say, it is a *struggle between centralization and freedom*.

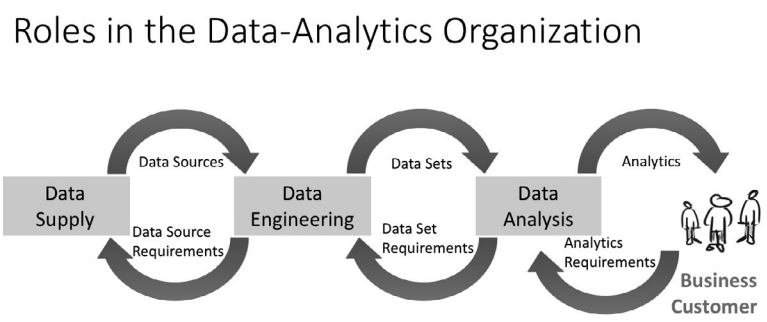


Figure 25: Data Supply, Data Engineering and Data Analysis work together in a supply chain to fulfill the analytics requirements of customers (business users).

The Analytics Supply Chain

DataOps processes and tools offer you a way to harmonize these opposing forces, empowering data analysts, while exerting a measured amount of centralization and control on your end-to-end process. To explore these ideas further, we need to review the structure of the analytics supply chain, and how the various roles relate to each other. The analytics organization in a DataOps enterprise consists of three essential job functions: data analysts, data engineers and data suppliers. You can think of the three roles as groups forming a supply chain. Data suppliers extract data for data engineers who create targeted data sets. Data analysts consume these data sets and generate analytics for business use cases. As figure 25 shows, the three functions work together in an interlinked fashion with data and analytics flowing to the right and feedback and requirements flowing to the left. Each group focuses on its immediate customer (its right neighbor), but together they share the mission of delivering analytic insights to business users. While they share a common underlying mission, the three groups operate in different business contexts.

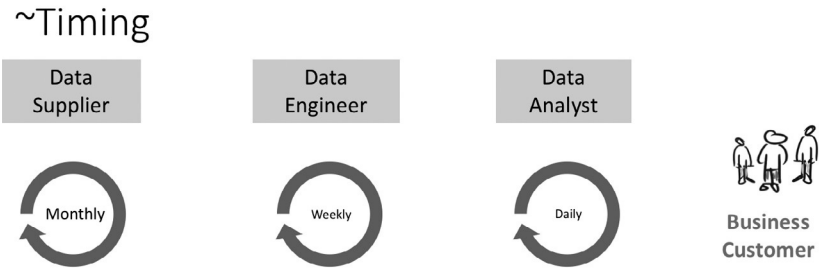


Figure 26: Data suppliers, engineers and analysts use different cycle times driven mainly by their tools, methods and proximity to demanding users.

Data Analysts

Data analysts directly support business users who work in fast-paced markets, which continuously evolve. A successful analyst finds ways to respond quickly to user requests. If new analytics are needed, the analyst pulls that together. New charts and graphs, updates, changes to calculated fields, integrations of new data sets — top-performing analysts do whatever it takes to address user requirements.

Analysts choose tools and processes oriented toward this business context. They use powerful, self-service tools, such as Tableau, Alteryx, and Excel, to quickly create or iterate on charts, graphs, and dashboards. They organize their work into daily sprints (figure 26), so they can deliver value regularly and receive feedback from users immediately. Agile tools like Jira are an excellent way to manage the productivity of analyst daily sprints.

The data analyst is the tip of the innovation spear. Organizations must give data analysts maximum freedom to experiment. There are a lot more data in the world than companies can analyze. Not everything can be placed in data warehouses. Not all data *should* be operationalized. Companies need data analysts to play around with different data sets to establish what is predictive and relevant.

When Freedom is Not Free

As their body of analytics grows, data analysts can get bogged down in non-value-add tasks. Self-service tools do not include mechanisms that promote and enable reuse. The data analyst may end up copying a calculated field into many reports and then have to manage changes to that algorithm manually. It becomes a revision control nightmare. New data sets are often integrated using labor-intensive and error-prone manual steps. This becomes a heavy burden on data analysts, consuming more than 75% of their time. Help and support from data engineering addresses these challenges for the team.

Some companies mistakenly ask data engineering to create data sets for every idea. It is best to let analysts lead on implementing new analytic ideas and proving them out before considering how data engineering can help. For example, consider the following:

1. Have the new analytics proven to be useful to many users?
2. Have calculations been reused in many reports/charts/dashboards?
3. Can automation reduce duplication of effort or manual integration errors?
4. Does data quality need improvement?

By this standard, the organization focuses its data engineering resources on those items that give the most *bang for the buck*. Keep in mind that when analytics are moved into a data warehouse, some of the benefits of centralization come at the expense of reduced freedom — it is slower to update a data warehouse than a Tableau worksheet. It's important to wait until analytics have *earned the right* to make this transition. The value created by centralizing must outweigh the restriction of freedom.

Data Engineers

Data engineers choose tools and processes which facilitate their objectives — to produce quality-checked data sets like data lakes, data warehouses and data marts for Data Analysts. These data sets include field calculations that analysts can leverage, promoting reuse. Data engineers can also automate data integration and other processes, minimizing manual steps for the data analyst. With the added centralization offered by data engineering, analysts can mitigate non-value add tasks and keep innovating rapidly.

Data engineers utilize programmable platforms such as AWS, S3, EC2 and Redshift. These tools require programming in a high-level language and offer greater potential functionality than the tools used by analysts. The relative complexity of the tools and scope of projects in data engineering fit best in weekly Agile iterations (figure 26). DataOps platforms like DataKitchen enable the data engineer to streamline the quality control, orchestration and data operations aspects of their duties. With automated support for agile development, impact analysis, and data quality, the data engineer can stay focused on creating and improving data sets for analysts.

After data sets have proven their value, it's worth considering whether the benefits of further centralization outweigh the cost of a further reduction in freedom. Data suppliers fulfill the function of greater centralization by providing data sources or data extracts for data engineering.

There are several reasons that a project may have earned the right to transition to data suppliers. Analytics may provide functionality that executives wish to make available to the entire corporation, not just one business unit. It could also be a case of standardization — for example, the company wants to standardize on an algorithm for calculating market share. In another example, perhaps data engineering has implemented quality control on a data set and wishes to achieve efficiencies by pushing this functionality upstream to the data supplier. A data supplier may be an external third party or an internal group, such as an IT master data management (MDM) team.

Master Data Management

In MDM, an enterprise links all of its critical data to a common reference. For example, in the pharmaceutical industry, there are 20-30 public data sets that describe physicians, payers and products. The initial merging and mastery of the data sets may, and in some cases should, be performed by the data engineering team, for example, for the purpose of business analytics.

After the usefulness of the mastered data is established, the company might decide that the data has broader uses. They may want the customer or partner list to be available for a portal or tied into a billing system. This use case requires a higher standard of accuracy for the mastered data than was necessary for the analytic data warehouse. It's appropriate at this point to consider moving the MDM to a data supplier, such as a corporate IT team, who are adept at tackling more extensive, development initiatives. Put another way, initial data mastery may have been good enough for analytic insights, but data must be perfect when it is being used in a billing system. The data supplier takes the MDM to the next level.

Function	Data Supply	Data Engineering	Data Analysis
Iteration Cycle Time	Monthly	Weekly	Daily
Deliverables	Data Extracts	Organized, quality-checked data sets	Produce insights via charts, graphs, dashboards
Customer	Data Engineers	Data Analysts	Business users
Technical tools	RDBS, MDM, Salesforce, etc.	AWS, DataKitchen, GIT	Self-service tools
Process management tools	MS Office and others	Jira	Jira

Table 2: Data Supply, Data Engineering and Data Analysis prefer different tools and methods which influence their optimal cycle times

Data Suppliers

Projects transitioned to data suppliers tend to incorporate more process and tool complexity than those in data engineering, leading to a more extended iteration period of one or more months (figure 26). These projects use tools such as RDBS, MDM, Salesforce, Excel, sFTP, etc., and rely upon waterfall project management and MS Project tracking. Table 2 summarizes tools and processes preferred by data suppliers as contrasted with engineers and analysts.

The Centralization-Freedom Spectrum

Data analysts, data engineers and data suppliers sit on a centralization-innovation (freedom) spectrum with data suppliers offering the most centralization capabilities, data analysts producing the fastest innovation and data engineering serving as a transition space in the center. The characteristic strengths and weaknesses of each of these groups are strongly influenced by their daily, weekly and monthly iteration periods. By organizing the various groups in this way, the company has access to the full spectrum of development; fast innovation, medium-scope development and longer-term, complex projects. For every need, the project has a home. The principle of granting analysts as much freedom as possible ensures that the innovation engine continues to turn. Waiting for analytics to stabilize and *earn the right* to be transitioned ensures that centralization adds value where it should and doesn't infringe on freedom and creativity.

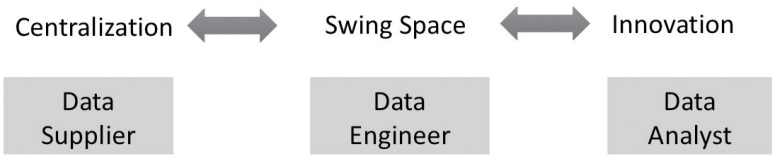


Figure 27: Data Suppliers, Data Engineers and Data Analysts sit on a spectrum of centralization and innovation/freedom.

The DataOps Framework for Innovation Management

The supply chain model that we have discussed illustrates how DataOps processes and tools help enterprises empower data analysts while exerting a measured amount of centralization and control on the end-to-end data pipeline. The special sauce behind DataOps is auto-
mated orchestration, continuous deployment and testing/monitoring of the data pipeline. DataOps reduces manual effort, enforces data quality and streamlines the orchestration of the data pipeline.

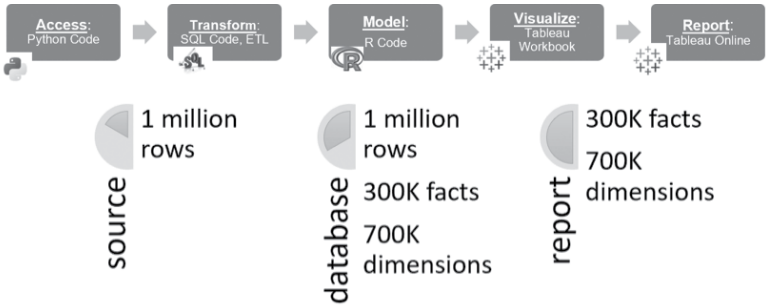


Figure 28: Tests verify that data rows, facts and dimensions match business logic throughout the data pipeline

For example, Figure 28 shows how the DataOps platform orchestrates, tests and monitors every step of the data operations pipeline, freeing up the team from significant manual effort. The test verifies that the quantity of data matches business logic at each stage of the data pipeline. If a problem occurs at any point in the pipeline, the analytics team is alerted and can resolve the issue before it develops into an emergency. With 24x7 monitoring of the data pipeline, the team can rest easy and focus on customer requirements for new/updated analytics.

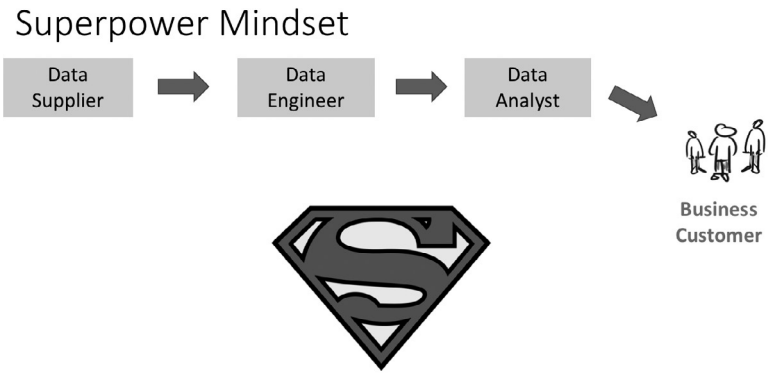


Figure 29: DataOps enables a superpower mindset — make your customer awesome

Giving Your Team Superpowers

The goal of DataOps is to minimize overhead and free data engineers and analysts to focus on delivering analytics to customers. With automation, DataOps enables data professionals to improve their productivity by an order of magnitude. A single data engineer supports ten data analysts, who in turn support 100 business professionals. It's like gaining *superpowers*. Data suppliers, engineers and analysts then concentrate their energy on their primary objec- tive — *making customers awesome*.

Improving Teamwork in Data Analytics with DataOps

Previously, we wrote about how members of large data organizations sometimes behave more like warring tribes than members of the same team. We discussed how DataOps facilitates commu- nication and task coordination between groups. Today we move from the macro to the micro-lev-
el. We look at how DataOps operates, within a team, to ease the flow of work from one team member to the next.

Whether celebrating a team's success or contemplating its failure, people tend to focus on team leadership as the most crucial factor in team performance. Richard Hackman, a pioneer in the field of organizational behavior who studied teams for more than 40 years, called this the "leader attribution error." People generally pay more attention to factors that they can see (like leaders) than to the background structural and contextual factors that actually determine team performance. Hackman's groundbreaking insight was to look beyond personalities, attitudes, or behavioral styles. (Put down your Meyers-Briggs assessments.) What matters most to high-performance teams is the presence of "enabling conditions."
As W. Edwards Deming famously said, "A bad system will beat a good person every time." DataOps applies this point of view to data analytics by taking a process-oriented approach to improving analytics quality and reducing cycle-time. It seeks to uncover the specific factors that best contribute to team success.
We live in a world where the average tenure of a CDO or CAO is about 2.5 years. A couple of years ago, Gartner predicted that 85 percent of AI projects would not deliver for CIOs. Forrester affirmed this unacceptable situation by stating that 75% of AI projects underwhelm. Clearly, data-analytics teams need a "tune-up."
After conducting 300 interviews and 4,200 surveys over 15 years, Haas and Mortensen (HBR, June 2016) built upon Richard Hackman's work by identifying four specific conditions most critical for team success:

1. Compelling direction – explicit and consequential goals that the team is working toward

2. **Strong structure** – includes optimally designed tasks and processes, and norms that promote positive dynamics.
3. Supportive context – includes an information system that provides access to the data needed for the work, and the material resources required to do the job
4. Shared mindset – collective identity and a shared reality

Per Haas and Mortensen, teams are more diverse, dispersed, digital, and dynamic than ever before. Modern organizations suffer from two corrosive problems — "*us versus them*" *thinking* and *incomplete information*. The four critical enabling conditions above help teams overcome these two pervasive problems and can raise overall team productivity while improving the quality of their work product.

INSIGHTS UNLIMITED

When enterprises invite us in to talk to them about DataOps, we generally encounter dedicated and competent people struggling with conflicting goals/priorities, weak process design, insufficient resources, clashing mindsets and differing views of reality. We would love to bring you inside these meetings, behind the closed doors, so you could see and experience this for yourself.

Imagine a typical enterprise. We'll call them…. **"Insights Unlimited."** Let's peek inside the data team's weekly staff meeting:

Manager: Good morning, everyone. As you know, our new Chief Data Officer has been asking questions about the large and growing list of work items on Jira. The backlog has grown steadily and…

Eric (Production Engineer): You're kidding me right. I lost most of last week chasing down data errors that originated upstream from one of our data sources. And the new reports that the development team gave me last week took 20 hours to install and then broke the weekly sales report. I thought Bill's (VP of Sales) head was going to explode.

Padma (Data Engineer): Hey, if you had let me test the changes in the "real" environment, I could have caught those problems upfront.

Eric (Production Engineer): As I have said before, the operational systems are not a sandbox. Plus, we have to control access to private HIPAA data.

A typical data analytics team has many key players, with distinct skill sets and tool preferences. There may be production engineers, data engineers, data analysts, data scientists, BI analysts, QA engineers, test engineers, ETL engineers, DBAs, governance and more. In our little anecdote, we could have filled a room full of grumpy and frustrated data professionals and business colleagues. For the sake of simplicity, we pared the team down to two members, Eric and Padma, who could each represent many people. To further explore the teamwork issues at Insights Unlimited, let's get to know Eric and Padma a little better. Note, that we'll be meeting a third key player on our data team as the exploration of Insights Unlimited continues.

ERIC, PRODUCTION ENGINEER AT INSIGHTS UNLIMITED

Role: Production Perfectionist

Goals: Protect and perfect the daily grind of delivering data; *minimize* errors and chaos

Skills: Taskmaster, *little-bit-of-everything* tech skill

Eric – Production Engineer

Eric is the backbone of data operations and keeps the operational systems running. This is a mission-critical role, and every error has visibility. Eric regularly takes calls at odd hours. We have to forgive Eric for being a little blunt and insensitive to the needs of the data science team. He has learned the hard way never to let anyone in the development team anywhere near the production systems. He gets a little impatient debugging other people's analytics. He wastes a lot of time responding to high-severity

alerts generated by poor data quality. He has the unenviable job of standing in front of directors and VPs and explaining what broke their charts and graphs. Every organization needs an Eric, and sometimes he is an oversubscribed resource (a bottleneck). If deploying new analytics into production requires Eric's time, then his backlog of tasks will grow, increasing cycle time. It's a multi-language, multi-tool, multi-platform world and Eric has to manage all of that. Eric is also a gatekeeper. Complex, risky changes are only allowed when there is scheduled downtime, like a long weekend.

PADMA, DATA ENGINEER AT INSIGHTS UNLIMITED

Role: Data Doer (or perhaps data scientist, BI/Analysts, etc.)

Goals: Create cool features for customers, flexible data architecture

Skills: SQL, ETL Tools, databases, machine learning

Padma – Data Engineer

Padma is the star that turns ideas into analytics that serve the business. She's an expert in analytics and machine learning tools. What motivates Padma is creating exciting new analytics. Whereas Eric wants to control change to reduce errors, Padma values a flexible data architecture that can be adapted quickly to new requirements. She wants to add new data sources and update schemas easily. Padma is a thought leader in AI and data science.

She's less interested in the IT infrastructure that powers the data pipeline. When a new project is assigned, Padma sometimes has to wait months for the IT department to order and configure a new development system or give her access to new data. She also waits weeks or months for the production team to deploy her new analytics. With the company's inefficient processes, Padma has trouble keeping up with user demands for new analytics, and colleagues sometimes think "she" is the bottleneck. Padma puts a lot of effort into quality, but because her test environment is different from production, there are always issues that surface during the production integration. Sometimes erroneous data flows into Padma's analytics, distorting results. That isn't something she can easily anticipate or address because operations lie outside her domain.

WITHOUT DATAOPS, A BAD "SYSTEM" OVERWHELMS GOOD PEOPLE

The situation that we see playing out with Eric and Padma repeats in many organizations we encounter. Padma and Eric have different pressures and priorities. Inadequate workflow processes prevent them from doing their best work. The team lacks the structural and contextual support necessary to enable successful teamwork.

Imagine that the Vice President of Marketing makes an urgent request to the data analytics team: "I need new data on profitability ASAP." At *Insights Unlimited*, the process for creating and deploying these new analytics would go something like this:

1. The new requirement falls outside the scope of the development “plan of record” for the analytics team. Changing the plan requires departmental meetings and the approval of a new budget and schedule. Meetings ensue.

2. Padma requests access to new data. The request goes on the IT backlog. IT grants access after several weeks.
3. Padma writes a functional specification and submits the proposed change to the Impact Review Board (IRB), which meets monthly. A key-person is on vacation, so the proposed feature waits another month.

4. Padma begins implementation. The change that she is making is similar to another recently developed report. Not knowing that, she writes the new analytics from scratch. Padma’s test environment does not match “production.” so her testing misses some corner cases.

5. Testing on the target environment begins. High-severity errors pull Eric into an “all-hands-on-deck” situation, putting testing temporarily on hold.
6. Once the fires are extinguished, Eric returns to testing on the target and uncovers some issues in the analytics. Eric feeds error reports back to Padma. She can’t easily reproduce the issues because the code doesn’t fail in the “dev” environment. She spends significant effort replicating the errors so she can address them. The cycle is repeated a few times until the analytics are debugged.

7. Analytics are finally ready for deployment. Production schedules the update. The next deployment window available is in three weeks.

8. After several months have elapsed (total cycle time), the VP of Marketing receives the new analytics, wondering why it took so long. This information could have boosted sales for the current quarter if it had been delivered when she had initially asked.

Every organization faces unique challenges, but the issues above are ubiquitous. The situation we described is not meeting anyone’s needs. Data engineers went to school to learn how to create analytic insights. They didn’t expect that it would take six months to deploy twenty lines of SQL. The process is a complete hassle for IT. They have to worry about governance and access control and their backlog is entirely unmanageable. Users are frustrated
because they wait far too long for new analytics. We could go on and on. No one here is enjoying themselves.

The frustration sometimes expresses itself as conflict and stress. From the outside, it looks like a teamwork problem. No one gets along. People are rowing the boat in different directions. If managers want to blame someone, they will point at the team leader.

At this point, a manager might try beer, donuts and trust exercises (hopefully not in that order) to solve the “teamwork issues” in the group. Another common mistake is to coach the group to work more slowly and carefully. This thinking stems from the fallacy that you have to *choose between quality and cycle time*. In reality, you can have both.

This team lacks the critical enabling conditions of success that Haas and Mortensen identified. We recommend a process-oriented solution that addresses everyone's goals and priorities, coordinates tasks, provisions resources and creates a shared reality. Our method for turning *a band of squabbling data professionals* into a high-performance team is called **DataOps**. (By the way, if you are new to the term, DataOps is the hottest thing in data science.)

DATAOPS IMPROVES TEAMWORK

DataOps shortens the cycle time and improves the quality of data analytics. Data teams that do not use DataOps may try to reduce the number of errors by being more cautious and careful. In other words, *slowing down*. DataOps helps organizations improve data quality while going faster. This might seem impossible until you learn more about how DataOps approaches analytics development and deployment.

DataOps is a set of methodologies supported by tools and automation. To say it in one breath; think Agile development, DevOps and lean manufacturing (i.e., statistical process controls) applied to data analytics. DataOps comprehends that enterprises live in a multi-language, multi-tool, heterogeneous environment with complex workflows. To implement DataOps, extend your existing environment to align with DataOps principles. As we have written extensively, you can implement DataOps by yourself in seven steps, or you can adopt a DataOps Platform from a third party vendor. In our example case, we are going to assume that *Insights Unlimited* begins using the DataOps Platform from DataKitchen. First, we'll describe how DataOps addresses the process, resource and information sharing challenges at *Insights Unlimited*. Next, we'll describe an analytics development project at *Insights Unlimited* using DataOps.

DATAOPS JOB #1: ABSTRACTING, SEPARATING AND ALIGNING RELEASE ENVIRONMENTS

Enterprises that collocate development and production on the same system face a number of issues. Analytics developers sometimes make changes that create side effects or break analytics. Development can also be processor intensive, impacting production performance and query response time.

DataOps provides production and development with dedicated system environments. Some enterprises take this step but fail to align these environments. Development uses cloud platforms while production uses on-prem. Development uses clean data while production uses raw data. The list of opportunities for misalignment are endless. DataOps requires that system environments be aligned. In other words, as close as possible to identical. The more similar, the easier it will be to migrate code and replicate errors. Some divergence is necessary. For example, data given to developers may have to be sampled or masked for practical or governance reasons.

Figure 39 below shows a simplified production environment. The system transfers files securely using SFTP. It stores files in S3 and utilizes a Redshift cluster. It also uses Docker containers and runs some Python. Production alerts are forwarded to a Slack channel in real-time. Note that we chose an example based on Amazon Web Services, but we could have selected any other tools. Our example applies whether the technology is Azure, GCP, on-prem or anything else.

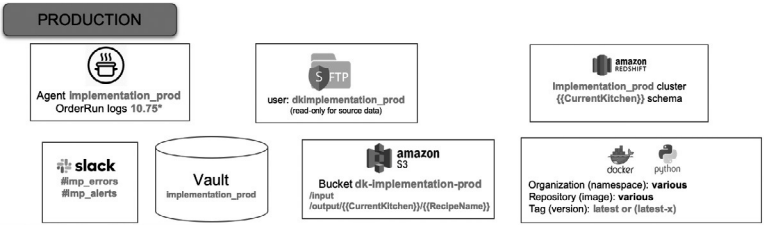


Figure 39: Simplified Production Technical Environment

DataOps segments production and development into separate release environments — see Figure 40. In our parlance, a release environment includes a set of hardware resources, a software toolchain, data, and a security Vault which stores, encrypted, sensitive access control information like usernames and passwords for tools. Our production engineer, Eric, manages the production release environment. Production has dedicated hardware and software resources so Eric can control performance, quality, governance and manage change. The production release environment is secure — the developers do not have access to it.

The development team receives its own separate but equivalent release environment, managed by the third important member of our team; Chris, Insight Unlimited's DataOps Engineer. Chris also implements the infrastructure that abstracts the release environments so that analytics move easily between dev and production. We'll describe this further down below. Any existing team member, with DataOps skills, can perform the DataOps engineering function, but in our simplified case study, adding a person will better illustrate how the roles fit together.

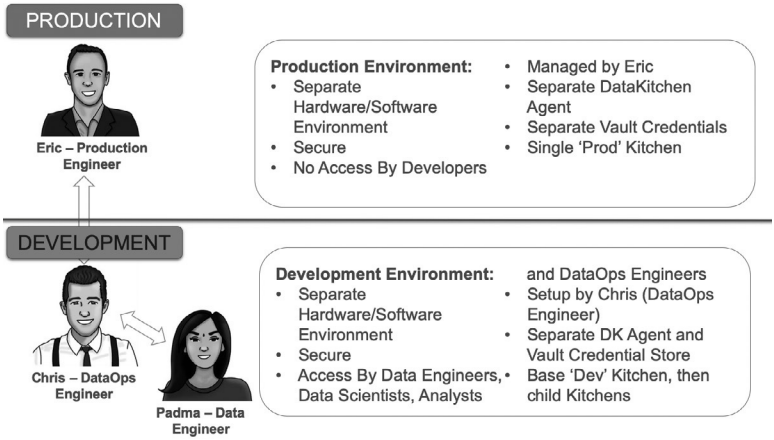


Figure 40: Production and development maintain separate but equivalent environments. The production engineer manages the production release environment and the DataOps Engineer manages the development release environment.

Chris creates a development release environment that matches the production release environment. This alignment reduces issues when migrating analytics from development to production. Per Figure 40, the development environment has an associated security Vault, just like the production environment. When a developer logs into a development workspace, the security Vault provides credentials for the tools in the development release environment. When the code seamlessly moves to production, the production Vault supplies credentials for the production release environment.

Figure 41 below illustrates the separate but equivalent production and development release environments. If you aren't familiar with "environments," think of these as discrete software and hardware systems with equivalent configuration, tools and data.

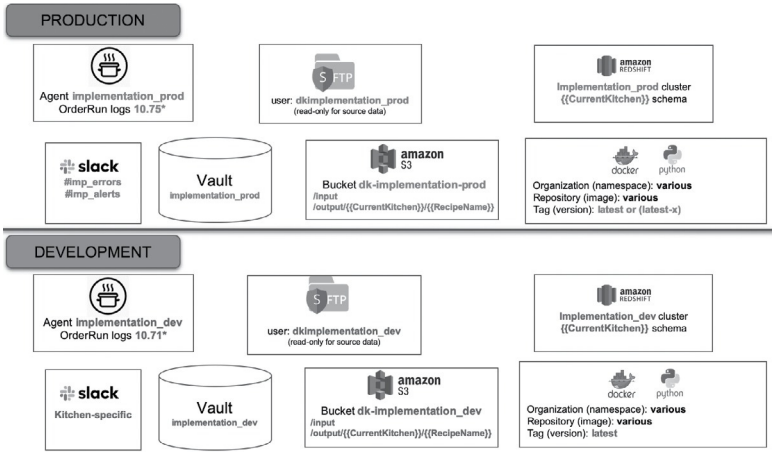


Figure 41: DataOps segments the production and development workspaces into separate but equivalent release environments.

Before we continue any further, let's formally add Chris to the team.

CHRIS, DATAOPS ENGINEER

Role: Cycle Time and Quality Optimizer

Goals: Setup and maintain development environments; accelerate and ease deployment

Skills: DevOps, cloud/IT, DataKitchen, security

Chris – DataOps Engineer

Chris uses DataOps to create and implement the processes that enable successful teamwork. This activity puts him right at the nexus between data-analytics development and operations. At *Insights Unlimited*, Chris is one of the most important and respected members of the data team. He creates the mechanisms that enable work to flow seamlessly from development to production. Chris makes sure that environments are aligned and that everyone has the hardware, software, data, network and other resources that they need. He also makes available

software components, created by team members, to promote reuse — a considerable multiplier of productivity. In our simple example, Chris manages the tasks that comprise the pre-release process. Padma appreciates having Chris on the team because now she has everything that she needs to create analytics efficiently on a self-service basis. Eric is happy because DataOps has streamlined deployment, and expanded testing has raised both data and analytics quality. Additionally, there is much greater visibility into the artifacts and logs related to analytics, whether in development, pre-release or in production. It's clear that Chris is a key player in implementing DataOps. Let's dive deeper into how it really works.

A DATAOPS "KITCHEN": A RELEASE ENVIRONMENT, WORKSPACE AND PIPE- LINE BRANCH

Our development team in Figure 40 consists of Chris and Padma. In a real-world enterprise, there could be dozens or hundreds of developers. DataOps helps everyone work as a team by minimizing the amount of rekeying required so that analytics move seamlessly from developer to developer and into production. DataOps also organizes activities so that tasks remain coordinated and team members stay aligned. The foundation of these synchronized activities is a virtual workspace called a "Kitchen."

A Kitchen is a development workspace with everything that an analytics developer requires. It contains hardware, software, tools, code (with version control) and data. A Kitchen points to a release environment which gives it access to all of the resources associated with that environment. A Kitchen also enforces workflow and coordinates tasks.
The processing pipelines for analytics consist of a series of steps that operate on data and produce a result. We use the term "Pipeline" to encompass all of these tasks. A DataOps Pipeline encapsulates all the complexity of these sequences, performs the orchestration work, and tests the results. The Idea is that any analytic tool that is invokable under software control can be orchestrated by a DataOps Pipeline. Kitchens enable team members to access, modify and execute workflow Pipelines. A simple Pipeline is shown in Figure 42. Pipelines, and the components that comprise them, are made visible within a Kitchen. This encourages reuse of previously developed analytics or services. Code reuse can be a significant factor in reducing cycle time.

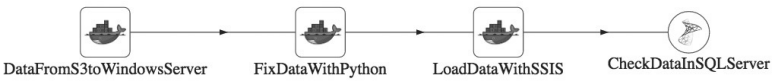


Figure 42: A simple DataOps Pipeline is represented by a directed acyclic graph (DAG). Each node in the graph is a sequence of orchestrated operations.

Kitchens should also tightly couple to version control (*Insights Unlimited* uses Git). When the development team wants to start work on a new feature, they instantiate a new child Kitchen which creates a corresponding Git branch. When the feature is complete, the Kitchen is merged back into its parent Kitchen, initiating a Git merge. The Kitchen hierarchy aligns with the source control branch tree. Figure 43 shows how Kitchen creation/deletion corresponds to a version control branch and merge.

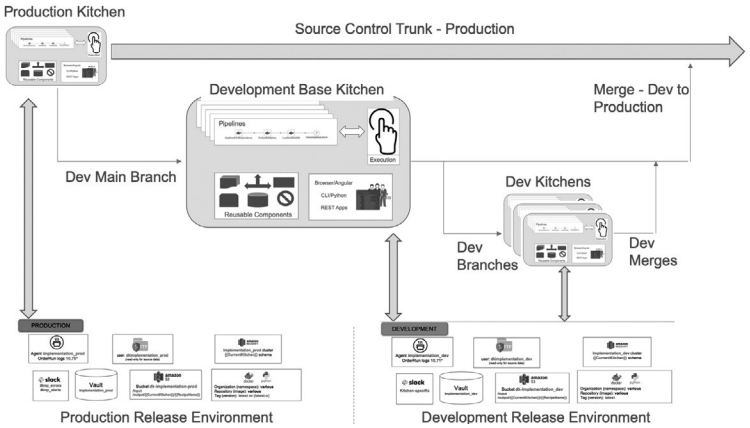


Figure 43: Kitchens point to a release environment. They represent source control branches and merges, and also serve as development, test and release workspaces.

Kitchens may be persistent or temporary; they may be private or shared, depending on the needs of a project. Access to a Kitchen is limited to a designated set of users or "Kitchen staff." The Vault in a release environment supplies a Kitchen with the set of usernames and passwords needed to access the environment toolchain.

DataOps empowers an enterprise to provide people access to data, eliminating gatekeepers. As mentioned above, developers access test data from within a Kitchen. In another example, a Pipeline could extract data from a data lake and create a data mart or flat file that serves Alteryx, Tableau and Excel users in the business units. DataOps promotes and enables *data democratization*, providing everyone access to the data relevant to their job. When "*self-service*" replaces "gatekeepers," more work gets done in parallel and analytics development cycle-time accelerates significantly.

Figure 44: Eric, Chris and Padma each have personal Kitchens, organized in a hierarchy that aligns with their workflow.

Figure 44 above shows a Kitchen hierarchy. The base Kitchen is “demo_production,” which points to the production release environment described earlier. This Kitchen is Eric’s workspace, and it enables him to coordinate his interactions with the development team. There is only one Kitchen corresponding to Eric’s production release environment. No iterative work takes place in production. Instead, think of “demo_production” as a manufacturing flow where assembly lines run on a tight schedule.

Chris’ workspace is a Kitchen called “demo_dev.” The “demo_dev” Kitchen is the baseline development workspace, and it points to the development release environment introduced above, at the bottom of Figure 41. In our example, Chris’ Kitchen serves as a pre-release staging area where merges from numerous child development Kitchens consolidate and integrate before being deployed to production. With release environments aligned, Kitchens don’t have to do anything different or special for merges across release environments versus merges within a release environment.

Every developer needs a workspace so they may work productively without impacting or being impacted by others. A Kitchen can be persistent, like a personal workspace, or temporary, tied to a specific project. Once Kitchen creation is set-up, team members create workspaces as needed. This “self-service” aspect of DataOps eliminates the time that developers used to wait for systems, data, or approvals. DataOps empowers developers to *hit the ground running*. In Figure 44, Padma has created the Kitchen “dev_kitchen.” Padma’s Kitchen can leverage Pipelines and other services created by the dev team.

DATAOPS SEGREGATES USER ACTIVITY

With multiple developers sharing a release environment, the DataOps Platform segregates developer activity. For example, all of the developer Kitchens share the Redshift cluster shown in Figure 41. Note the notation “{{CurrentKitchen}}” associated with Redshift in Figure 41. Each developer has a Redshift schema within the cluster identified by their Kitchen name. For example, an access by Padma would target a schema identified by her unique Kitchen name “dev_kitchen.” The DataOps Platform uses Kitchen names and other identifiers to segregate user activity within a shared release environment. Segregation helps keep everyone’s work isolated while sharing development resources.

Slack messages are similarly segregated by Kitchen, in our Figure 41 example. Note how production alerts are directed to the Slack channels “#imp_errors” and “#imp_alerts,” while dev alerts are sent to a kitchen-specific Slack channel. This prevents production from seeing any dev-related slack messages. It also prevents the developers from receiving each other’s Slack messages. Alerts could easily be managed on a much finer-grained level if required.

DATAOPS AT INSIGHTS UNLIMITED

Let’s look at how *Insights Unlimited* is now utilizing a DataOps Platform to develop and deliv- er analytics with minimal cycle time and unsurpassed quality. We’ll walk through an example of how DataOps helps team members work together to deploy analytics into production.

Think back to the earlier request by the VP of Marketing for “new analytics.” DataOps coordinates this multi-step, multi-person and multi-environment workflow and manages it from

inception to deployment. The *Insights Unlimited* DataOps process addresses this requirement in six steps.

STEP 1 — STARTING FROM A TICKET

The Agile Sprint meeting commits to the new feature for the VP of Marketing in the upcom- ing iteration. The project manager creates a JIRA ticket.

STEP 2 — CREATION OF THE DEVELOPMENT KITCHEN

In a few minutes, Padma creates a development Kitchen for herself and gets to work. Chris has automated the creation of Kitchens to provide developers with the test data, resources and Git branch that they need. Padma's Kitchen is called "dev_Kitchen" (see Figure 44).

If Padma takes a technical risk that doesn't work out, she can abandon this Kitchen and start over with a new one. That effectively deletes the first Git branch and starts again with a new one.

STEP 3 — IMPLEMENTATION

Padma's Kitchen provides her with Pipelines that serve as a significant head start on the new profitability analytics. Padma procures the test data she needs (de-identified) and configures toolchain access (SFTP, S3, Redshift, …) for her Kitchen. Padma implements the new analytics by modifying an existing Pipeline. She adds additional tests to the existing suite, checking that incoming data is clean and valid. She writes tests for each stage of ETL/processing to ensure that the analytics are working from end to end. The tests verify her work and will also run as part of the production flow. Her new Pipelines include orchestration of the data and analytics as well as all tests. The tests direct messages and alerts to her Kitchen-specific Slack channel. With the extensive testing, Padma knows that her work will migrate seamlessly into production with minimal effort on Eric's part. Now that release environments have been aligned, she's confident that her analytics work in the target environment.

Before she hands off her code for pre-production staging, Padma first has to merge down from "demo_dev" Kitchen so that she can integrate any relevant changes her coworkers have made since her branch. She reruns all her tests to ensure a clean merge. If there is a conflict in the code merge, the DataOps Platform will pop-up a three panel UI to enable further investigation and resolution. When Padma is ready, she updates and reassigns the JIRA ticket. If the data team were larger, the new analytics could be handed off from person to person, in a line, with each person adding their piece or performing their step in the process.

STEP 4 — PRE-RELEASE

In our simple example, Chris serves as the pre-release engineer. With a few clicks, Chris merges Padma's Kitchen "dev_Kitchen" back into the main development Kitchen "demo_dev," initiating a Git merge. After the merge, the Pipelines that Padma updated are visible in Chris' Kitchen. If Chris is hands-on, he can review Padma's work, check artifacts, rerun her tests, or even add a few tests of his own, providing one last step of QA or governance. Chris creates a schedule that, once enabled, will automatically run the new Pipeline every Monday at 6 am. When Chris is satisfied, he updates and reassigns the JIRA ticket, letting Eric know that the feature is ready for deployment.

STEP 5 — PRODUCTION DEPLOYMENT

Eric easily merges the main development Kitchen "demo_dev" into the production Kitchen, "demo_production," corresponding to a Git merge. Eric can now see the new Pipelines that Padma created. He inspects the test logs and reruns the new analytics and tests to be 100% sure. The release environments match so the new Pipelines work perfectly. He's also happy to see tests verifying the input data using DataOps statistical process control. Tests will detect erroneous data, before it enters the production pipeline. When he's ready, Eric enables the schedule that Chris created, integrating the new analytics into the operations pipeline. DataOps redirects any Slack messages generated by the new analytics to the production Slack channels.

STEP 6 — CUSTOMER SEES RESULTS

The VP of Marketing sees the new customer segmentation and she's delighted. She then has an epiphany. If she could see this new data combined with a report that Padma delivered last week, it could open up a whole new approach to marketing for *Insights Unlimited* — something that she is sure the competitors haven't discovered. She calls the analytics team and… back to Step 1.

DATAOPS BENEFITS

As our short example demonstrated, the DataOps Teamwork Process delivers these benefits:

- **Ease movement between team members with many tools and environments** — Kitchens align the production and development environment(s) and abstract the machine, tools, security and networking resources underlying analytics. Analytics easily migrate from one team member to another or from dev to production. Kitchens also bind changes to source control.
- **Collaborate and coordinate work** — DataOps provides teams with the compelling direction, strong structure, supportive context and shared mindset that are necessary for effective teamwork.
- **Automate Work and Reduce errors** — Automated orchestration reduces process variability and errors resulting from manual steps. Input, output and business logic tests at each stage of the workflow ensure that analytics are working correctly, and that data is within statistical limits. DataOps runs tests both in development and production, continuously monitoring quality. Warnings and errors are forwarded to the right person/channel for follow up.
- **Maintain security** — Kitchens are secured with access control. Kitchens then access a release environment toolchain using a security Vault which stores unique usernames/passwords.
- **Leverage best practices and re-use** — Kitchens include Pipelines and other reusable components which data engineers can leverage when developing new features.
- **Self-service** — Data professionals can move forward without waiting for resources or committee approval.
- **Data democratization** — Data can be made available to more people, even users outside the data team, who bring contextual knowledge and domain expertise to data-analytics initiatives. "Self-service" replaces "gatekeepers" and everyone can have access to the data that they need.
- **Transparency** — Pipeline status and statistics are available in messages, reports and dashboards.

- **Data democratization –** Data can be made available to more people, even users outside the data team, who bring contextual knowledge and domain expertise to data-analytics initiatives. “Self-service” replaces “gatekeepers” and everyone can have access to the data that they need.
- **Transparency –** Pipeline status and statistics are available in messages, reports and dashboards.

SMOOTH TEAMWORK WITH DATAOPS

DataOps addressed several technical and process-oriented bottlenecks at *Insights Unlimited* that previously delayed the creation of new analytics for months. Their processes can im- prove further, but they are now an order of magnitude faster and more reliable. At the next staff meeting, the mood of the team is considerably improved:

Manager: *Good morning, everyone. I’m pleased to report that the VP of Marketing called the CDO thanking him for a great job on the analytics last week.*
Padma *(Data Engineer): Fortunately, I was able to leverage a Pipeline developed a few months ago by the MDM team. We were even able to reuse most of their tests.*
Chris *(DataOps Engineer): Once I set-up Kitchen creation, Padma was able to start being productive immediately. With matching release environments, we quickly migrated the new analytics from dev to production.*

Eric *(Production Engineer): The tests are showing that all data remains within statistical limits. The dashboard indicators are all green.*

DataOps helps our band of frustrated and squabbling data professionals achieve a much higher level of overall team productivity by establishing processes and providing resources that support teamwork. With DataOps, two key performance parameters improve dramati- cally — the development cycle time of new analytics and quality of data and analytics code. We’ve seen it happen time and time again.

What’s even more exciting is the business impact of DataOps. When users request new analytics and receive them in a timely fashion, it initiates new ideas and uncharted areas of exploration. This tight feedback loop can help analytics achieve its true aim, stimulating creative solutions to an enterprise’s greatest challenges. Now that’s teamwork!

Eliminate Your Analytics Development Bottlenecks

APPLYING THE THEORY OF CONSTRAINTS TO DATA ANALYTICS

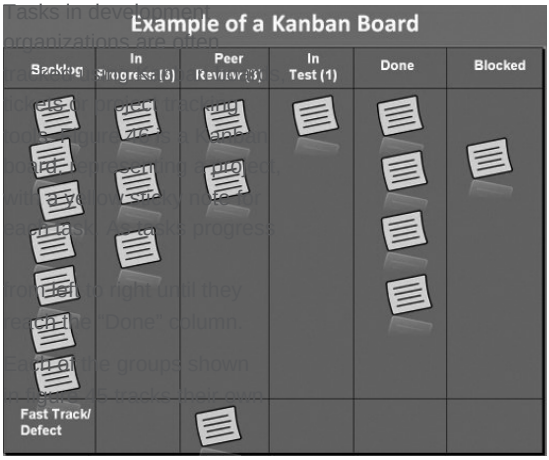
Business users often have no concept of what it takes to design and deploy robust data analytics. The gap between expectations and execution is one of the main obstacles holding the analytics team back from delighting its users. Managers may ask for a simple change to a report. They don't expect it to take weeks or months.

Analytics teams need to move faster, but cutting corners invites problems in quality and governance. How can you reduce cycle time to create and deploy new data analytics (data, models, transformation, visualizations, etc.) without introducing errors? The answer relates to finding and eliminating the bottlenecks that slow down analytics development.

Figure 45: The creation of analytics in a large data organization requires the contribution of many groups.

YOUR DEPLOYMENT PIPELINE

Analytics development in a large data organization typically involves the contribution of several groups. Figure 45 shows how multiple teams work together to produce analytics for the internal or external customer.



through milestones, they move

Figure 46: Example Kanban Board

projects. Figure 47 shows the data-analytics groups again, but each with their own Kanban boards to track the progress of work items. To serve the end goal of creating analytics for users, the data teams are desperately trying to move work items from the backlog (left column) to the done column at the right, and then pass it off to the next group in line.

Data professionals are smart and talented. They work hard. Why does it take so long to move work tickets to the right? Why does the system become overloaded with so many unfinished work items forcing the team to waste cycles context switching?

Figure 47: The development pipeline with Kanban boards

To address these questions, we need to think about the creation and deployment of ana- lytics like a manufacturing process. The collective workflows of all of the data teams are a linked sequence of steps, not unlike what you would see in a manufacturing operation. When we conceptualize the development of new analytics in this way, it offers the possi- bility of applying manufacturing management tools that uncover and implement process improvements.

THE THEORY OF CONSTRAINTS

One of the most influential methodologies for ongoing improvement in manufacturing operations is the Theory of Constraints (ToC), introduced by Dr. Eliyahu Goldratt in a business novel called “The Goal,” in 1984. The book chronicles the adventures of the fictional plant manager Alex Rogo who has 90 days to turn around his failing production facility. The plant can’t seem to ship anything on time, even after installing robots and investing in other improvements dictated by conventional wisdom. As the story progresses, our hero learns why none of his improvements have made any difference.

THE BOTTLENECK

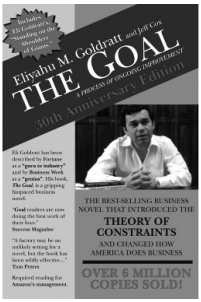


Figure 48

The plant’s complex manufacturing process, with its long sequence of interdependent stages, was throughput limited by one particular operation — a certain machine with limited capacity. This machine was the “*constraint*” or bottleneck. The Theory of Constraints views every process as a series of linked activities, one of which acts as a constraint on the overall throughput of the entire system. The constraint could be a human resource, a process, or a tool/ technology.

In “The Goal,” Alex learned that *“an improvement at any point in the system, not at the constraint, is an illusion.”* An improvement made at a stage that feeds work to the bottleneck just increases the queue

of work waiting for the bottleneck. Improvements after the bottleneck will always remain starved. Every loss of productivity at the bottleneck is a loss in the throughput of the entire system. Losses in productivity in any other step in the process don't matter as long as that step still produces faster than the bottleneck.

Even though Alex's robots improved efficiency at one stage of his manufacturing process, they didn't alleviate the true system constraint. When Alex's team focused improvement efforts on raising the throughput of the bottleneck, they were finally able to increase the throughput of the overall manufacturing process. True, some of their metrics looked worse (the robot station efficiency declined), but they were able to reduce cycle time, ship product on time and make a lot more money for the company. That is, after all, the real "goal" of a manufacturing facility.

FINDING YOUR BOTTLENECK

To improve the speed (and minimize the cycle time) of analytics development, you need to find and alleviate the bottleneck. This bottleneck is what is holding back your people from producing analytics at a peak level of performance. The bottleneck can often be identified using these simple indications:

• **Work in Progress (WIP)** – In a manufacturing flow, work-in-progress usually accumulates before a constraint. In data analytics, you may notice a growing list of requests for a scarce resource. For example, if it takes 40 weeks to provision a development system, your list of requests for them is likely to be long.
• **Expedite** – Look for areas where you are regularly being asked to divert resources to ensure that critical analytics reach users. In data analytics, data errors are a common source of unplanned work.

• **Cycle Time** – Pay attention to the steps in your process with the longest cycle time. For example, some organizations take 6 months to shepherd 20 lines of SQL through the impact review board. Naturally, if a step is starved or blocked by a dependency, the bottleneck is the external factor.

• **Demand** – Note steps in your pipeline or process that are simply not keeping up with demand. For example, often less time is required to create new analytics than to test and validate them in preparation for deployment.

EXAMPLE BOTTLENECKS IN DATA ANALYTICS

You may notice a common theme in each of the example bottlenecks above. A bottleneck is especially problematic because it prevents people on the analytics team (analysts, scientists, engineers, …) from fulfilling their primary function — creating new analytics. Bottlenecks distract them from high priority work. Bottlenecks redirect their energy to non-value add activities. Bottlenecks prevent them from implementing new ideas quickly.

When managers talk to data analysts, scientists and engineers, they can quickly discover the issues that slow them down. Figure 49 shows some common constraints. For example, data errors in analytics cause unplanned work that upsets a carefully crafted Kanban board. Work-in-progress (WIP) is placed on hold and key personnel context switch to address

the high-severity outages. Data errors cause the Kanban boards to be flooded with new tasks which can overwhelm the system. Formerly high priority tasks are put on hold, and management is burdened, having to manage the complexity of many more work items. Data errors also affect the culture of the organization. After a series of interruptions from data errors, the team becomes accustomed to moving more slowly and cautiously. From a Theory of Constraints perspective, data errors severely impact the overall throughput of the data organization.

A related problem, also shown in figure 49, occurs when deployment of new analytics breaks something unexpectedly. Unsuccessful deployments can be another cause of unplanned work which can lead to excessive caution, and burdensome manual operations and testing.

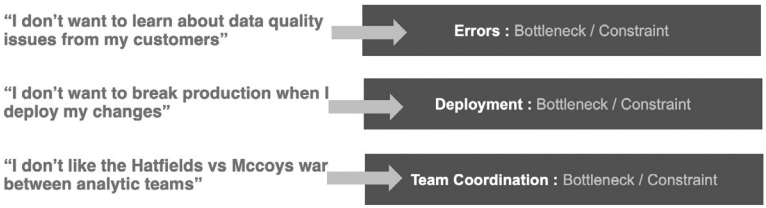


Figure 49: Translating problems to constraints

Another common constraint is team coordination. The teams may all be furiously rowing the boat, but perhaps not in the same direction. In a large organization, each team's work is usually dependent on each other. The result can be a serialized pipeline. Tasks could be parallelized if the teams collaborated better. New analytics wouldn't break existing data operations with proper coordination between and among teams.

A wide variety of constraints potentially slow down analytics development cycle time. In development organizations, there are sometimes multiple constraints in effect. There is also variation in the way that constraints impact different projects. The following are some potential rate-limiting bottlenecks to rapidly deploying analytics:

- Dependency on IT to make schema changes or to integrate new data sets
- Impact Review Board
- Provisioning of development systems and environments
- Long test cycles
- Data errors causing unplanned work
- Manual orchestration
- Fear of breaking existing analytics
- Lack of teamwork among data engineers, scientists, analysts, and users
- Long project cycles — deferred value

When you have identified a bottleneck, the Theory of Constraints offers a methodology called the Process Of On-Going Improvement (POOGI) to address it. If you have many active bottlenecks that all need to be addressed, it may be more effective to focus on them one at a time. Below, we will suggest a method that we have found particularly effective in prioritizing projects.

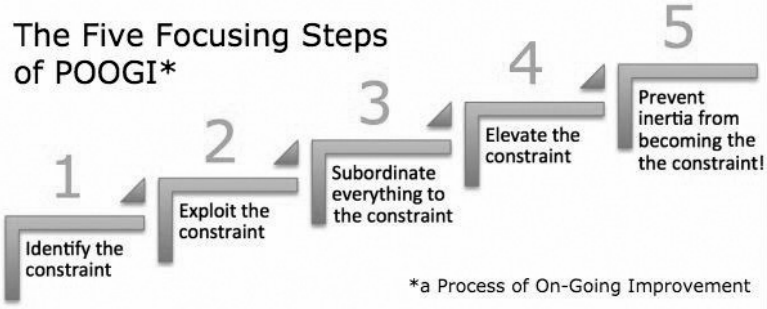


Figure 50: Source: Theory of Constraints Institute, Process of On-Going Improvement (POOGI)

ALLEVIATING THE BOTTLENECK

Once identified, the Theory of Constraints recommends a five-step methodology to address the constraint:

1. Identify the constraint

2. Exploit the constraint – Make improvements to the throughput of the constraint using existing resources

3. Subordinate everything to the constraint – Review all activities and make sure that they benefit (or do not negatively impact) the constraint. Remember, any loss in productivity at the constraint is a loss in throughput for the entire system.

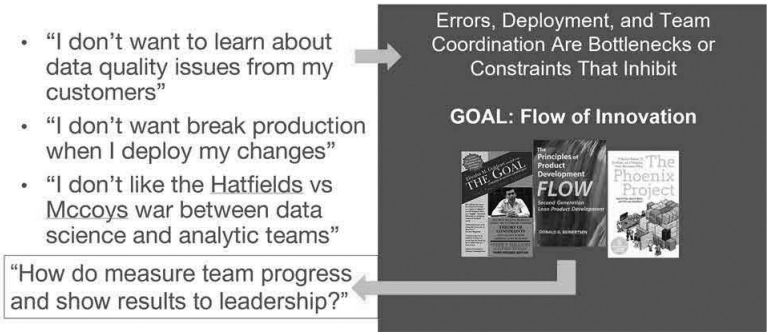


Figure 51: Errors, deployment and team coordination are bottlenecks that inhibit the flow of analytics innovation

4. Elevate the constraint – If after steps 2–3, the constraint remains in the same place, consider what other steps, such as investing resources, will help alleviate this step as a bottleneck
5. Prevent inertia from becoming a constraint by returning to step 1.

THE THEORY OF CONSTRAINTS APPLIED TO IT

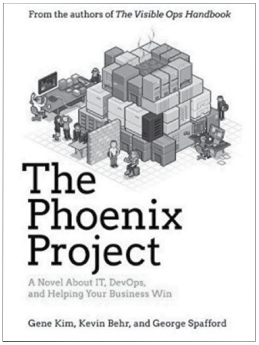


Figure 52

A leading book on DevOps, called "The Phoenix Project," was explained by author Gene Kim to be essentially an adaptation of "The Goal" to IT operations. To alleviate their bottleneck, the team in the book implements Agile development (small lot sizes) and DevOps (automation). One important bottleneck was a bright programmer named Brent who was needed for every system enhancement and was constantly being pulled into unplanned work. When the team got better at relieving and managing their constraints, the output of the whole department dramatically improved.

PRIORITIZING DATAOPS PROJECTS BASED ON DESIRED OUTCOMES

If you have identified multiple bottlenecks in your development process, it may be difficult to decide which one to tackle first. DataOps is a methodology that applies Agile, DevOps and lean manufacturing to data analytics. That's a lot of ground to cover. One way to approach this question is to think like a product or services company.

Figure 53: Many data professionals can relate to the experience of working diligently to deliver what customers say they want only to receive a lukewarm response.

The data organization creates analytics for its consumers (users, colleagues, business units, managers, …). Think of analytics as your product and data consumers as your customers. Like any product or service organization, perhaps you should simply ask your customers what they want? The problem is that customers don't actually know what products or services they want. What customer would have asked for Velcro or Post-It notes or Twitter? Many data professionals can relate to the experience of working diligently to deliver what customers say they want only to receive a lukewarm response.

There is much debate about how to listen to the voice of the customer (Dorothy Leonard, Harvard Business School, The Limitations of Listening). Customer preferences are reliable when you ask them to make selections within a familiar product category.

If you venture outside of the customer's experience, you tend to encounter two blocks. People fixate on the way that products are normally used, preventing them from thinking *outside the box*. Second, customers have seemingly contradictory needs. Your data-analytics customers want analytics to be error-free, which requires a lot of testing, but they dislike waiting for lengthy QA activities to complete. Data professionals might feel like they are in a no-win situation.

Management consultant Anthony Ulwick contends (Harvard Business Review) that you should not expect your customers to recommend solutions to their problems. They aren't expert enough for that. Instead, ask about desired *outcomes*. What do they want analytics to do for them? The customers might say that they want changes to analytics to be completed very fast so they can play with ideas. They won't tell you to implement automated orchestration or a data warehouse which can both contribute to that outcome.

The outcome-based methodology for gathering customer input breaks down into five steps.

STEP 1 — PLAN OUTCOME-BASED CUSTOMERS INTERVIEWS

Deconstruct, step by step, the underlying processes behind your delivery of data analytics. It may make sense to interview users like data analysts who leverage data to create analytics for business colleagues.

STEP 2 — CONDUCT INTERVIEWS

Pay attention to desired outcomes not recommended solutions. Translate solutions to outcomes by asking what benefit the suggested feature/solution provides. Participants should consider every aspect of the process or activity they go through when creating or consuming analytics. A good way to phrase desired outcomes is in terms of the type (minimize, increase) and quantity (time, number, frequency) of improvement required. Experts in this method report that 75% of the customers' desired feedback is usually captured in the first two-hour session.

STEP 3 — ORGANIZE THE DATA

Collect a master list of outcomes, removing duplicates and categorize outcomes into groups that correspond to each step in the process

STEP 4 — RATE THE OUTCOMES

Conduct a quantitative survey to determine the importance of each desired outcome and the degree to which the outcome is satisfied by the current solution. Ask customers to rate, on a scale of 1–10, the importance of each desired outcome (Importance) and the degree to which it is currently satisfied (Satisfaction). These factors are input into the opportunity algorithm below which helps rate outcomes based on potential.

The opportunity algorithm makes use of a simple mathematical formula to estimate the potential opportunity associated with a particular outcome:

Opportunity = Importance + (Importance - Satisfaction)

When you are done, you should have produced something like the below example.

Desired Outcome	Importance	Satisfaction	Opportunity
Minimize data errors	9.5	3.2	15.8
Release new analytics (iterations) weekly	8.3	4.2	12.4
Provision development environments within 2 days	9.5	7.5	11.5
Test new analytics and approve deployment within 8 hours	9.1	8.4	9.8
Minimize time of impact review of new analytics	5.1	1.0	9.2
Minimize time to make schema changes	7.7	6.6	8.8

Table 3: Desired outcomes ranked by opportunity strength

STEP 5 — GUIDE INNOVATION

The table above reveals which outcomes are important to users and deprecates those outcomes that are already well served by the existing analytics development process. The outcomes which are both important and unsatisfied will rise to the top of the priority list. This data can be used as a guide to prioritize process improvements in the data analytics development pipeline and process.

THE PATH FORWARD FOR DATAOPS

DataOps applies manufacturing management methods to data analytics. One leading method, the Theory of Constraints, focuses on identifying and alleviating bottlenecks. Data analytics can apply this method to address the constraints that prevent the data analytics organization from achieving its peak levels of productivity. Bottlenecks lengthen the cycle time of developing new analytics and prevent the team from responding quickly to requests for new analytics. If these bottlenecks can be improved or eliminated, the team can move faster, developing and deploying with a high level of quality in record time.

If you have multiple bottlenecks, you can't address them all at once. The opportunity algorithm enables the data organization to prioritize process improvements that produce outcomes that are recognized as valued by users. It avoids the requirement for users to understand the technology, tools, and processes behind the data analytics pipeline. For DataOps proponents, it can provide a clear path forward for analytics projects that are both important and appreciated by users.

Prove Your Awesomeness with Data: The CDO DataOps Dashboard

Do you deserve a promotion? You may think to yourself that your work is exceptional. Could you prove it?

As a Chief Data Officer (CDO) or Chief Analytics Officer (CAO), you serve as an advocate for the benefits of data-driven decision making. Yet, many CDO's are surprisingly unanalytical about the activities relating to their own department. Why not use analytics to shine a light on yourself?

Internal analytics could help you pinpoint areas of concern or provide a big-picture assessment of the state of the analytics team. We call this set of analytics the *CDO Dashboard*. If you are as good as you ~~think you are,~~ the CDO Dashboard will show how simply awesome you are at what you do. You might find it helpful to share this information with your boss when discussing the data analytics department and your plans to take it to the next level. Below are some reports that you might consider including in your *CDO dashboard*:

BURNDOWN CHART

The burndown chart graphically represents the completion of backlog tasks over time. It shows whether a team is on schedule and sheds light on the productivity achieved in each development iteration. It can also show a team's accuracy in forecasting its own schedule.

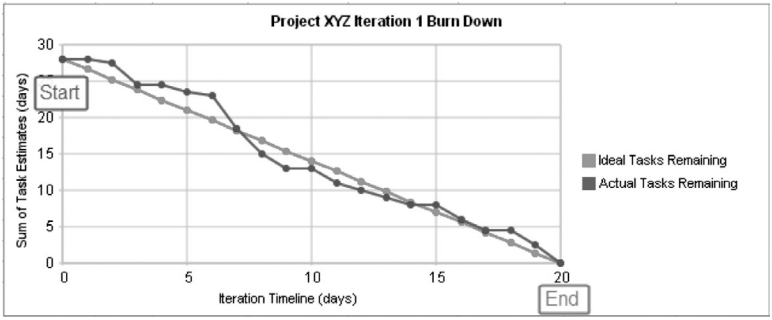


Figure 54: Sample Burndown chart

VELOCITY CHART

The velocity chart shows the amount of work completed during each sprint — it displays how much work the team is doing week in and week out. This chart can illustrate how improved processes and indirect investments (training, tools, process improvements, …) increase velocity over time.

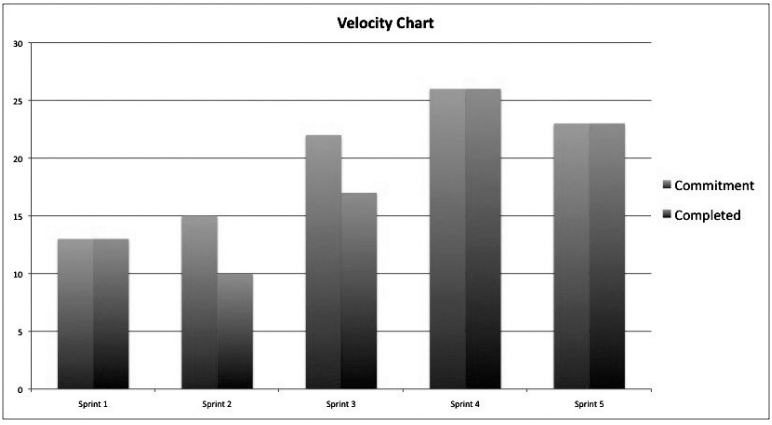


Figure 55: Sample Velocity chart

TORNADO REPORT

The Tornado Report is a stacked bar chart that displays a weekly representation of the oper- ational impact of production issues and the time required to resolve them. The Tornado Re- port provides an easy way to see how issues impacted projects and development resources.

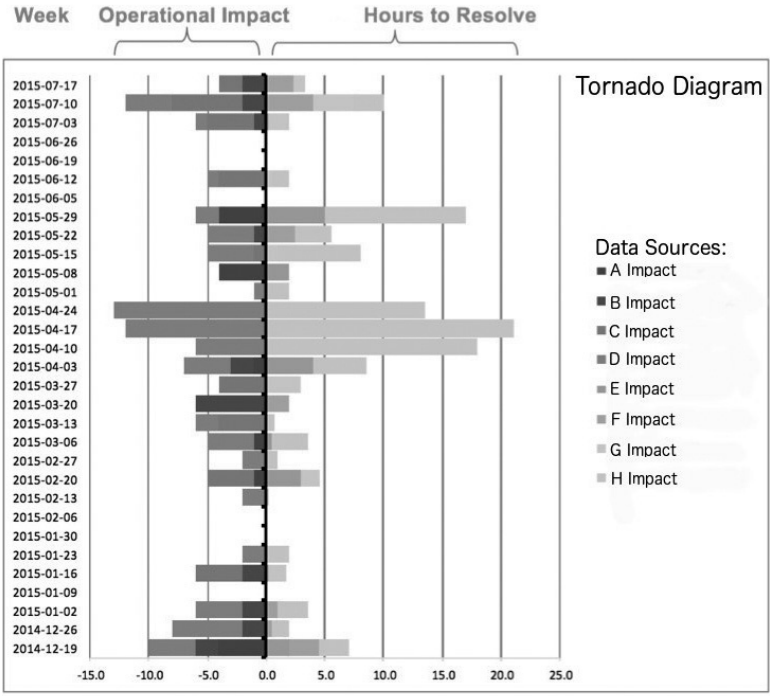


Figure 56: Sample Tornado report

DATA ARRIVAL REPORT

A large organization might receive hundreds of data sets from suppliers and each one could represent dozens of files. All of the data has to arrive error-free in order to, for example, build the critical Friday afternoon report. The Data Arrival report tracks how vendors per- form relative to their respective service level agreements (SLA).

The Data Arrival report enables you to track data suppliers and quickly spot delivery issues. Any partner that causes repeated delays can be targeted for coaching and management. The Tornado Report mentioned above can help quantify how much time is spent managing these issues in order to articulate impact. These numbers are quite useful when coaching a peer organization or vendor to improve its quality.

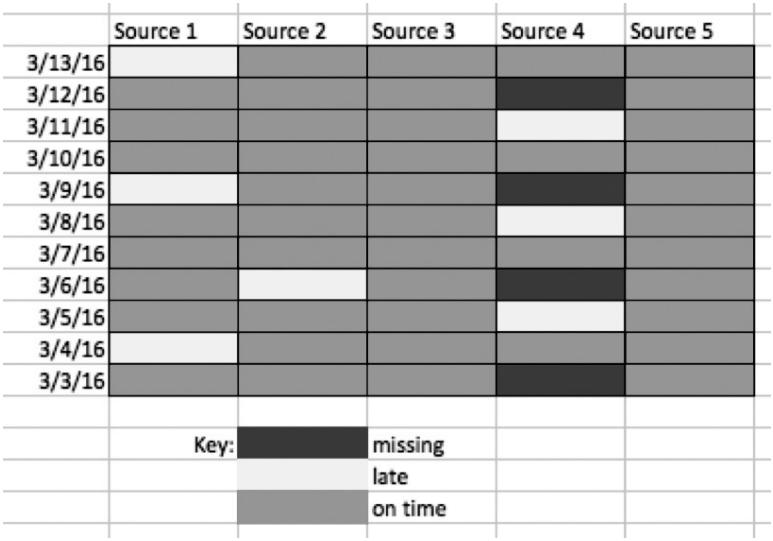


Figure 57: Sample Data Arrival Report

TEST COVERAGE AND INVENTORY

The Test Coverage and Inventory Reports show the degree of test coverage of the data analytics pipeline. It shows the percent of tables and data covered by tests and how test coverage improves over time. The report can also provide details on each test. In a DataOps enterprise, results from tests run on the production pipeline are linked to real-time alerts. If a process fails with an error, the analytics team can troubleshoot the problem by examining test coverage before or after the point of interest.

STATISTICAL PROCESS CONTROLS

The data analytics pipeline is a complex process with steps often too numerous to be monitored manually. Statistical Process Control (SPC) allows the data analytics team to monitor the pipeline end-to-end from a big-picture perspective, ensuring that everything is operating as expected.

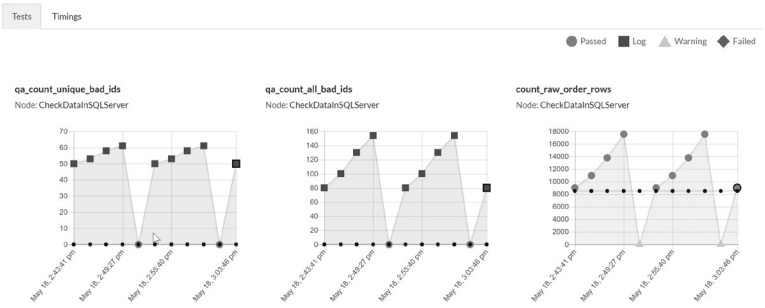


Figure 58: Sample Statistical Process Controls

NET PROMOTER SCORE

A Net Promoter Score is a customer satisfaction metric that gauges a team's effectiveness. For a data team, this is often a survey of internal users who are served by analytics. The Net Promoter Score can show that the data analytics team is effective at meeting the needs of its internal customer constituency or that satisfaction is improving.

One of the main goals of analytics is to improve decision-making. The CDO Dashboard puts information at the fingertips of executives so they have a complete picture of what is happening in the data analytics domain. When it's time to review performance, the CDO Dashboard can help you show others that the analytics department is a well-oiled machine. *Now, about that promotion…*

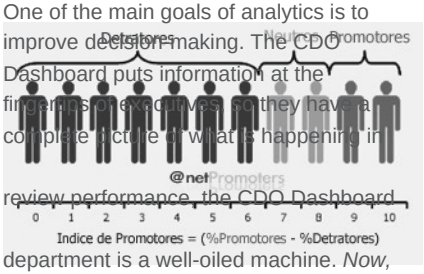


Figure 59: Net Promoter Score

Surviving Your Second Year as CDO

As the Chief Data Officer (or Chief Analytics Officer) of your company, you manage a team, oversee a budget and a hold a mandate to set priorities and lead organizational change. The bad news is that everything that could possibly go wrong from a security, governance and risk perspective is your responsibility. If you do a perfect job, then no one on the management team ever hears your name.

The average tenure of a CDO or CAO is about 2.5 years. In our conversations with data and analytics executives, we find that CDOs and CAOs often fall short of expectations because they fail to add sufficient value in an acceptable time frame. If you are a CDO looking to survive well beyond year two, we recommend avoiding three common *traps* that we have seen ensnare even the best and brightest.

1) THE TRAP OF DATA DEFENSE

Babson College professor Tom Davenport classifies data and analytics projects as either defense or offense. Data defense seeks to resolve issues, improve efficiency or mitigate risks. Data quality, security, privacy, governance, compliance — these are all critically important endeavors, but they are in essence, just enabling activities. You could think of data defense as providing *indirect* value.

Data offense expands top-line revenue, builds the brand, grows the company and in general puts *points on the board*. Using data analytics to help marketing and sales is data offense. Companies may acknowledge the importance of defense, but they care passionately about offense and focus on it daily. Data offense provides the organization with *direct* value and it is *what gets CDOs and CAOs promoted*.

The challenge for a CDO is that data defense is hard. A company's shortcomings in governance, security, privacy, or compliance may be glaringly obvious. In some cases, new regulations like GDPR (General Data Protection Regulation, EU 2016/679) demand immediate

action. Data defense has a way of consuming more than its fair share of attention and staff. If not put in perspective, data defense is a trap that can divert the CDO's attention and resources away from offensive activities that create value for the organization.

2) THE TRAP OF DEFERRED VALUE

Projects that implement new platforms and solutions can require months, if not years, of integration and oversight. If conceived as a waterfall project, with a *big-bang* deliverable at the end, these projects produce little to no value until they are complete. We call this the trap of *deferred value*, and it is possibly the main reason that many CDOs never make it past year three of their tenure.

In a fast-paced, competitive environment, an 18-month integration project can seem like the remote future. Also, success is uncertain until you deliver. Your C-level peers know that big software integration projects fail half the time. Projects frequently turn out to be more complex than anticipated, and they often miss the mark. For example, you may have thought you needed ten new capabilities, but your internal customers only really require seven, and two of them were not on your original list. The issue is that you won't know which seven features are critical until around the time of your second annual performance review and by then it might be too late to right the ship.

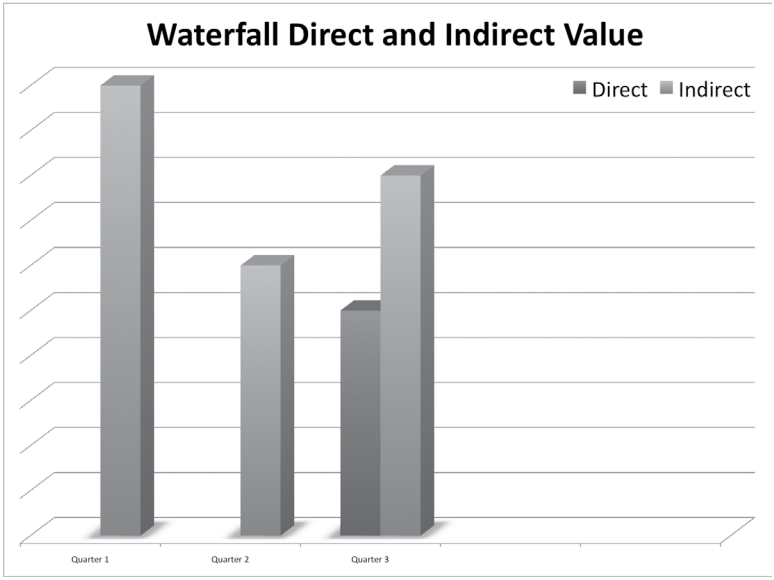


Figure 60: CDO's often make the dual mistake of (1) focusing too much on delivering indirect value (governance, security, privacy, or compliance, ...) and (2) using a waterfall project methodology which defers the delivery of value to the end of a long project cycle. In the case shown, it takes several months to deliver direct value

3) THE TRAP OF DATA VALUATION

Industry analysts and the media have long touted the strategic value of data. Following the advice of analysts, a CDO may decide to embark on a project to quantify the monetary value of the company's data. This seems like a worthy endeavor that some say should attain a high level of visibility.

A data valuation project can take months of effort and consumes the attention of the CDO and her staff on what is essentially an internally-focused, intellectual exercise. In the end, you have a beautiful PowerPoint presentation with detailed spreadsheets to back it up. Your data has tremendous value that can and should be carried on the balance sheet. You tell everyone all about it — why don't they care?

Don't confuse data valuation with data offense. Knowing the theoretical value of data is not data offense. While data valuation may be useful and important in certain cases, it is often a distraction. All of the time and resources devoted to creating and populating the valuation model could have been spent on higher value-add activities.

DIRECT VS. INDIRECT VALUE

Investments in data analytics can create value either directly or indirectly. Sales growth is an example of direct value. Indirect value lays the foundation for future growth and productivity. In both cases, value is delivered either quickly or in a longer time frame. One common mistake is to focus too heavily on indirect-value, long-term projects.

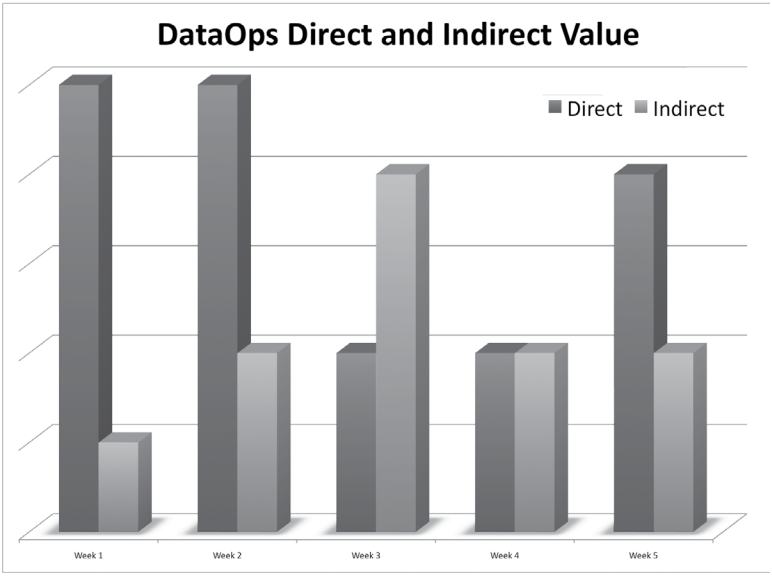


Figure 61: DataOps uses an iterative product management methodology (Agile development) that enables the CDO to rapidly deliver direct value (growing the top line).

That's not to say indirect or long-term projects don't have their place. They can be important and worthwhile. For example, a CDO may wisely invest in employee training or building technical infrastructure. It's essential to create the right mix, investing in enough indirect value-creators to support long-term growth and enough direct-value and short-term projects to maintain a high level of visibility.

DATAOPS ACCELERATES VALUE CREATION

The trick is to reorganize the data and analytics teams to be responsive and adaptable to the needs of internal customers and users. DataOps can help here. DataOps subscribes to an Agile, iterative approach. Deliver something of value in a few weeks and build on that in successive intervals. DataOps combines Agile with DevOps and lean manufacturing methods to provide a data and analytics team with the processes and tools needed to accelerate the value-creation cycle. *Raise a glass to year two!*

CAOs and CDOs: Earn the Trust of your CEO

One of the greatest challenges in analytics is earning the trust of your organization's CEO and management team. A lot of people in the business world make decisions with their gut. They rely on experience and intuition, but many companies would prefer to depend upon data. You cannot walk through an airport these days and not see some version of an adver- tisement saying we are the company who is going to help you to be more data-driven.

People do not always trust data. Imagine you are an executive and an employee walks into your office and shows you charts and graphs that contradict strongly held assumptions about your business. A lot of managers in this situation favor their own instincts. Data-analytics professionals, who tend to be doers, not talkers, are sometimes unable to convince an organization to trust its data.

BUILDING TRUST IN THE DATA

We've discovered that the best way for data-analytics professionals to build trust with their management team is to deliver value consistently, quickly and accurately. To accomplish this, you need to create and publish analytics in a new way. We call this new approach DataOps. DataOps is a combination of tools and methods, which streamline the development of new analytics while ensuring impeccable data quality. DataOps helps shorten the cycle time for producing analytic value and innovation, addressing some of the fundamental challenges that prevent organizations from trusting their data.

EARN TRUST BY DELIVERING A JOURNEY OF VALUE

DataOps uses the Agile Development methodology to create valuable new analytics for the business or organization. Agile accepts that people don't necessarily know what they want until they see it. The data-analytics team delivers new analytics in a short time frame and receives immediate feedback from users. This tight feedback loop steers the development of new analytics to the features that are most valuable to the business. Users aren't expected to take a *leap of faith*. They are gradually introduced to their own data and direct the journey

of each future improvement in analytics through their feedback. This feedback both improves the analytics and draws them into the process as an active and invested stakeholder. When users grow to appreciate the value provided, it is time to operationalize the analytics and deliver them on a continuous basis.

EARN TRUST BY DELIVERING QUICKLY

In DataOps automated tests enable the data-analytics team to deploy new analytics quickly and with confidence. Minimizing the cycle time of new analytics is critical to earning the trust of users. The relevance of a question asked, at any given moment, decays rapidly as the situation facing the organization quickly evolves. Customers and prospects do not stand still. If analytics take too long, frustration builds, and the answer to a question might be delivered long after the question has ceased to be relevant.

DataOps relies upon the data lake design pattern, which enables data analytics teams to update schemas and transforms quickly to create new data warehouses that promptly address pressing business questions. DataOps incorporates the continuous-deployment methodology that is characteristic of DevOps. This reduces the cycle time of new analytics by an order of magnitude. When users get used to quick answers, it builds trust in the data-analytics team, and stimulates the type of creativity and teamwork that leads to breakthroughs.

EARN TRUST BY DELIVERING ACCURATELY

Trust is tough to earn and easy to squander. If bad data makes its way through your data pipeline, the users might not ever again trust the data. DataOps tests and monitors business logic and data validity by testing data at each stage of the data-analytics pipeline. We liken this testing to the statistical process control used in lean manufacturing. The tests can start simple, but over time they are expanded in breadth until they become a formidable check
on quality. If a problem occurs with an internal or external dataset, or at any processing stage, the data-analytics team will be alerted immediately by the automated tests. DataOps protects the integrity of the data, so the data itself is worthy of user trust.

THE BENEFIT OF TRUSTED ANALYTICS

Earning your organization's trust makes the job of the data-analytics team a lot easier, but much more is at stake. Companies that don't trust their data will be outcompeted in the marketplace. Managers will make decisions based on instinct, past experience or preconceived notions. In cases like this, managers sometimes develop different versions of reality and can't agree on the facts, let alone strategic plans.

A company that trusts its data develops a unified view of reality and can formulate a shared vision of how to achieve its goals. Data-driven companies deliver higher growth and ultimately higher valuations than their peers. As a CAO or CDO, leading the organization to become more data-driven is your mission. DataOps makes that easier by helping the data-analytics team deliver quickly and robustly, creating value that is recognized and trusted by the organization.

The Four Stage Journey to Analytics Excellence

First, we walk, then we run. The same is true in data analytics. In our many discussions, we have encountered companies that are just starting out with data analytics and others with substantial organizations handling petabytes of data. Everyone that we meet is somewhere along this spectrum of maturity. We've found that just because an enterprise's data analytics organization is large does not mean that it is *excellent*. In fact, the flaws in a process or methodology become particularly noticeable when a team grows beyond the initial stages.

We view every company as being somewhere on a journey towards achieving excellence. In our experience, the journey is divided into four stages. That said, some get there faster by taking a shortcut. We'll discuss the four-stage journey and the shortcut to excellence below.

STAGE 1 - DATA DESERT

Companies generate data from a variety of enterprise applications. This data can help organizations gain a better understanding of customers, products, and markets. If your company is not reaping value from your data, then you live in a data desert. In a data desert, the data is underutilized or lays dormant. Like a mineral resource that remains in the ground, the data could have enormous potential, but without data analytics that potential goes unrealized.

This situation could have implications for the company's future. What if competitors have devised a way to use data analytics to garner a competitive advantage? Without a comprehensive data strategy, a company risks missing the market.

STAGE 2- BOUTIQUE ANALYTICS

Some organizations are engaged in analytics but do so in a decentralized fashion or on a small scale. Some enterprises are just getting started in analytics. Whether or not a person has programming skills, it is possible to do a fair amount of analytics using everyday tools like spreadsheets. One can accomplish even more using data visualization software. We call this Boutique Analytics. In a boutique shop, data analytics professionals are akin to artisans.

Boutique Analytics tend to be ad hoc or create one-off reports that answer questions posed by a manager. For example, a global enterprise may wish to know how much of its revenue it derives from one customer. Data is exported from CRMs or operations systems and pulled into a spreadsheet for analysis. The term Boutique Analytics may make it sound small in scale, but some large enterprises are known to rely solely upon this approach. A large enterprise might run weekly reports exporting sales data into a flat file. The global sales and marketing team can then easily manipulate the data in a spreadsheet. The sharing of data using flat files can be used to complement an enterprise's operational analytics.
There is nothing inherently wrong with Boutique Analytics. It is a great way to explore the best ways to deliver value based on data. The eventual goal should be to operationalize the data and deliver that value on a regular basis. This can be time-consuming and error-prone if executed manually.

STAGE 3- WATERFALL ANALYTICS

If an analytics initiative is successful and the team grows, a company will eventually begin to manage analytics more formally. Companies usually have a deeply entrenched project management culture based upon the methodology used by their research and development teams. Often project management is based on the Waterfall method so it is natural for these organizations to implement Waterfall Analytics.

In the Waterfall world, development cycles are long and rigidly controlled. Projects pass through a set of sequential phases: architecture, design, test, deployment, and maintenance. Changes in the project plan at any stage cause modifications to the scope, schedule or budget of the project. As a result, Waterfall projects are resistant to change. This is wholly appropriate when you are building a bridge or bringing a new drug to market, but in the field of data analytics, changes in requirements occur on a continuous basis. Teams that use Waterfall analytics often struggle with development cycle times that are much longer than their users expect and demand. Waterfall analytics also tends to be labor intensive, which makes every aspect of the process slow and susceptible to error. Most data-analytics teams today are in the Waterfall analytics stage and are often unaware that there is a better way.

Stage	Name	Model	Data Pipeline	Cycle Time
1	**Data Desert**	N/A	None	N/A
2	**Boutique Analytics**	Individual Artisan	One-Off or Ad Hoc	Custom
3	**Waterfall Analytics**	Waterfall Project Management	Manual Process	Long
4	**DataOps Analytics**	Agile Development, DevOps, Lean Manufacturing	Automated Process	Short

Table 4: Four Stages of Analytics

STAGE 4 - DATAOPS ANALYTICS

DataOps is a new approach to data analytics, which is superior to Waterfall Analytics in terms of flexibility, quality, and development cycle time. DataOps adopts key concepts from lean manufacturing. It views data analytics as a continuously operating pipeline, which can be automated, monitored and controlled. New analytics are created using Agile Development, a methodology created in the software engineering field. Agile manages the development of new analytics by delivering valuable features in short increments. This allows an organization to quickly adapt to new requirements or change course based on the demands of the marketplace. Analytics are deployed using the continuous deployment methodology pioneered by DevOps. Automated orchestration replaces labor-intensive manual processes. This means that new analytics can be published continuously, on-demand with minimal human intervention. Data quality flowing through the data analytics pipeline is monitored using automated data and logic tests executed as part of the continuous deployment automation. These tests are inspired by the statistical process control widely used in modern manufactur- ing operations.

TAKE THE SHORTCUT

The mistake that many companies make is that they languish in stage 3. The better approach is to take a shortcut, skip stage 3 entirely, and move directly to stage 4. If your organization
is already in stage 3, then it's advantageous to advance as quickly as possible.

THE JOURNEY TO EXCELLENCE

DataOps provides the foundation for data analytics excellence. It streamlines the development of new analytics, shortens cycle time, and automates the data-analytics pipeline, freeing the team to focus on value-adding activities. It also controls the quality of data flowing through the pipeline so users can trust their data. With DataOps in place, the team is productive, responsive and efficient. They will race far ahead of competitors whose analytics are less nimble and less impactful. DataOps shortens your journey to analytics excellence.

Pitching a DataOps Project That Matters

DataOps addresses a broad set of use cases because it applies workflow process automation to the end-to-end data-analytics lifecycle. DataOps reduces errors, shortens cycle time, eliminates unplanned work, increases innovation, improves teamwork, and more. Each of these improvements can be measured and iterated upon.

These benefits are hugely important for data professionals, but if you made a pitch like this to a typical executive, you probably wouldn't generate much enthusiasm. Your data consumers are focused on business objectives. They need to grow sales, pursue new business opportunities, or reduce costs. They have very little understanding of what it means to create development environments in a day versus several weeks. How does that help them "evaluate a new M&A opportunity by Friday?"

If you pitch DataOps in terms of its technical benefits, an executive or co-worker might not understand its full potential value. Instead, explain how agile and error-free analytics serves the organization's mission. What would it mean to monetize data more effectively than competitors? Data is the modern business decision apparatus (just ask Google, Target, Amazon, or Facebook). DataOps enables companies to rapidly assess and pursue opportunities, avoiding strategic mistakes, and shrinking time-to-market. What would it mean for a company to lead its industry in savvy and business agility? When discussing a DataOps initiative with an executive or colleague, focus on his/her top business objective and find a project related to it. Impactful DataOps projects are those that help colleagues and executives pursue their objectives. Below we suggest some additional unconventional approaches to finding high-visibility DataOps projects.

Find Unhappy Analytics Users

A strained relationship between the data team and users can point to a potential DataOps pilot project. A data team with unhappy users is ripe for transformational change. You may instinctively wish to turn away from grumbling users. You should be thankful for them. The more vocal and unhappy the customers are, the bigger the opportunity to turn the situation around and bring high-impact improvements to the broadest possible group. A large community of dissatisfied customers is also likely to be a higher priority for managers and executives. Ask your unhappy customers or colleagues what concerns them most about the data-analytics team. User discontent may be expressed in feelings and observations. User surveys can organize and quantify user anecdotes into actionable priorities. The list of possible issues is long, but you might hear feedback that includes:

- Data science/engineering/analytic teams do not deliver the insight that the business customers need.
- The data team takes too long to deliver analytics.
- Users mistrust the data itself or the team working on the data.
- Stakeholders have hired consultants or shadow teams to do data work.

Be Grateful for Negative Feedback

Negative feedback often stems from deep, underlying issues. The data team may not deliver relevant analytics because business users and data analysts are isolated from each other. Users may mistrust data and analytics because of errors. When business units hire their own data analysts, it's a sign that they are underserved. They may feel like the data organization is not addressing their priorities.

User feedback may feel concrete to users, but as a data professional, you will have to translate these requirements into metrics. For example, users may not trust the data. That may seem abstract and not directly actionable. Try measuring your errors per week. If you can show users that you are lowering that number, you can build trust. A test coverage dashboard can illustrate progress in quality controls. Demonstrating your success with data can help gradually win over detractors. What other problems have eroded trust? You may need to look for more than one contributing factor.

In many organizations, analytics follows a complex path from raw data to processed analytics that create value. Your data crosses organizational boundaries, data centers, teams, and organizations. Errors can creep in anywhere along this path. What are the historical drivers of issues/errors? Which teams own each part of the process? A lack of responsiveness sometimes squanders trust. Measure how fast teams can respond to errors and requests.

Another common user complaint is that data-analytics teams take too long to deliver requested features. The length of time required to deliver analytics can be expressed in a metric called cycle time. Benchmark how fast you can deploy new ideas or requests into production. To reduce cycle time, examine the data science/engineering/analytic development process. For example, how long does it take to create a development environment? How up-to-date are development environments? How well-governed are development environments?

Creating a Feedback Loop of Trust

As DataOps improves trust in data and data-team responsiveness, business users will naturally begin to work more closely with the data team. As the data team becomes more agile, interaction with users increases in importance. DataOps focuses on delivering value to customers in short, frequent iterations. The value that business users receive after interacting with the data team reinforces the value of working together. DataOps enterprises frequently observe greater and more frequent communication and collaboration between users and the data team. The positive feedback loop of collaboration and value creation encourages users and data professionals to invest in working closely together. In the end, the quality of collaboration that DataOps fosters becomes the engine that takes an organization to new heights.

The “Right to Repair” Data Architecture with DataOps

We’ve been attending data conferences for over 20 years. It has been common to see presenters display a data architecture diagram like the (simplified) one below (figure 69). A data architecture diagram shows how raw data turns into insights. As the Eckerson Group writes, “a data architecture defines the processes to capture, transform, and deliver usable data to business users.”

In our canonical data architecture diagram, data sources flow in from the left and pass through transformations to generate reports and analytics for users or customers on the right. In the middle, live all of the tools of the trade: raw data, refined data, data lakes/warehouses/marts, data engineering, data science, models, visualization, governance and more. Tools and platforms can exist in the cloud or on premises. Most large enterprise data architectures have evolved to use a mix of both.

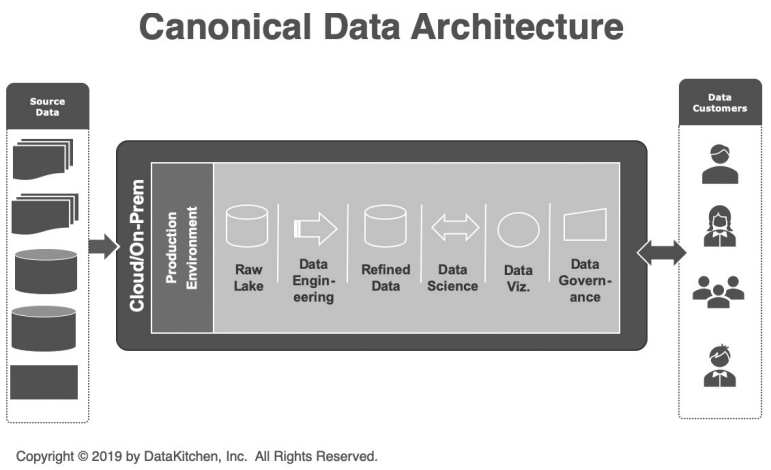


Figure 69: A typical Production-only view of Data Architecture

When data professionals define data architectures, the focus is usually on production requirements: performance, latency, load, etc. Engineers and data professionals do a great job executing on these requirements. The problem is that the specifications don’t include architecting for rapid change.

They only think about production, not the process to make changes to production. **A DataOps Data Architecture makes the steps to change what is in production a “central idea.”** Thinking first about changes over time to your code, your servers, your tools, and monitoring for errors are first class citizens in the design.

Take this example. Mobile phone designs increasingly locate batteries in fixed locations underneath sensitive electronics. In many cases, batteries can no longer be easily accessed and replaced by a consumer. The “Right to Repair” movement advocates for policies that enable customers to fix the things that they own instead of throwing them away.

When managers and architects fail to think about architecting the production data pipeline for rapid change and efficient development, it is a little like designing a mobile phone with a fixed battery. You can end up with processes characterized by unplanned work, manual deployment, errors, and bureaucracy. It can take months to deploy a minor 20 line SQL change. Building a data architecture without planning for change is much worse than building a mobile phone with a fixed battery. While mobile phone batteries are swapped every few years, your analytics users are going to want changes every day or sometimes every hour. That may be impossible with your existing data architecture, but you can meet this requirement if you architect for it. If data architectures are designed with these goals in mind, they can be more flexible, responsive, and robust. Legacy data pipelines can be upgraded to achieve these aims by enhancing the architecture with modern tools and processes.

A DATAOPS DATA ARCHITECTURE

Imagine if your data architects were given these requirements up front. In addition to the standard items, the user story or functional specification could include requirements like these:

1. Update and publish changes to analytics within an hour without disrupting operations
2. Discover data errors before they reach published analytics
3. Create and publish schema changes in a day

If you are a data architect yourself (or perhaps you play one), you may already have creative ideas about how you might address these types of requirements. You would have to maintain separate but identical development, test, and production environments. You would have to orchestrate and automate test, monitoring, and deployment of new analytics to production. When you architect for flexibility, quality, rapid deployment, and real-time monitoring of data (in addition to your production requirements), you are moving towards a DataOps data architecture as shown in figure 70.

The DataOps data architecture expands the traditional operations-oriented data architecture by including support for Agile iterative development, DevOps, and statistical process control. We call these tools and processes collectively a DataOps Platform. The DataOps elements in

our new data architecture are shown in figure 70.

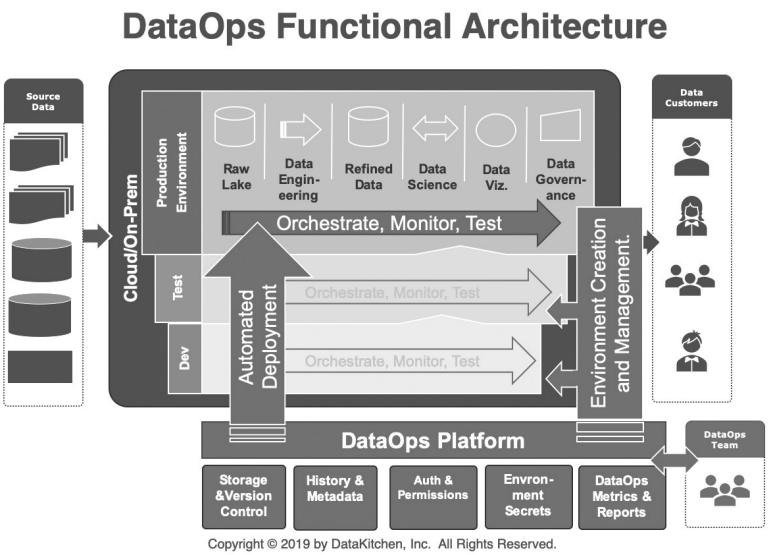


Figure 70: DataOps Functional Data Architecture

BREAKDOWN OF THE DATAOPS ARCHITECTURE

A DataOps architecture contains support for environment creation and management. This enables separate development, test, and production environments, which in turn support orchestration, monitoring, and test automation. The software automates impact review and new-analytics deployment so that changes can be vetted and published continuously. Agents in each environment operate on behalf of the DataOps Platform to manage code and configuration, execute tasks, and return test results, logs, and runtime information. This enables the architecture to work across heterogeneous tools and systems. The DataOps Platform also integrates several other functions which support the goal of rapid deployment and high quality with governance:

- **Storage /Revision Control —** Version control manages changes in artifacts; essential for governance and iterative development. (example: git, docker hub)
- **History and Metadata** — Manage system and activity logs (example, MongoDB)
- **Authorization and Permissions** — Control access to environments (example: Auth0)

- **Environment Secrets** — Role-based access to tools and resources within environments (example: Vault)
- **DataOps Metrics and Reports** — Internal analytics provide a big-picture assessment of the state of the analytics and data team. We call this the CDO Dashboard. (example: Tableau)
- **Automated Deployment** — This involves moving the code/configuration from one environment (e.g., a test environment) to a production environment. (Examples: Jenkins, CircleCI)
- **Environment Creation and Management** — treat your infrastructure as code be able to create places for your team to do work with all the required hardware, software, and test data sets they need. (example: chef, puppet, etc.)
- **Orchestrate, Test, Monitor** — As your pipelines are running, orchestrate all the tools involved, test and monitor, and alert if something goes wrong. (examples, Airflow, Great Expectations, Grafana, etc.)

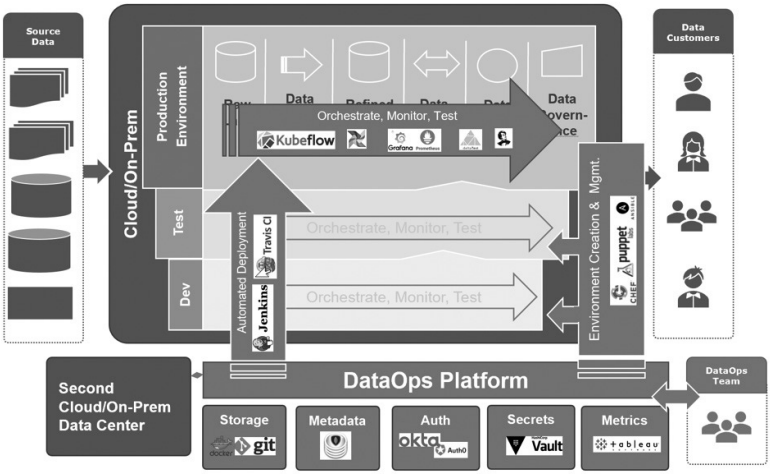


Figure 71: DataOps Data Architecture with Example Tools

MULTI-LOCATION DATAOPS DATA ARCHITECTURE

Companies are increasingly moving their work from on-premises to the cloud. Enterprises are choosing to have multiple cloud providers, as well. As a result, your data analytics workloads can span multiple physical locations and multiple teams. Your customers only see the result of that coordination. How can you do DataOps across those locations and teams and not end up with a "*Data Oooooops*"? Think of a "hub and spoke" model for your DataOps Data Architecture. As shown in figure 72, the DataOps Platform is the hub for your distributed sites engaging in development and operations. Testing is also coordinated between the sites.

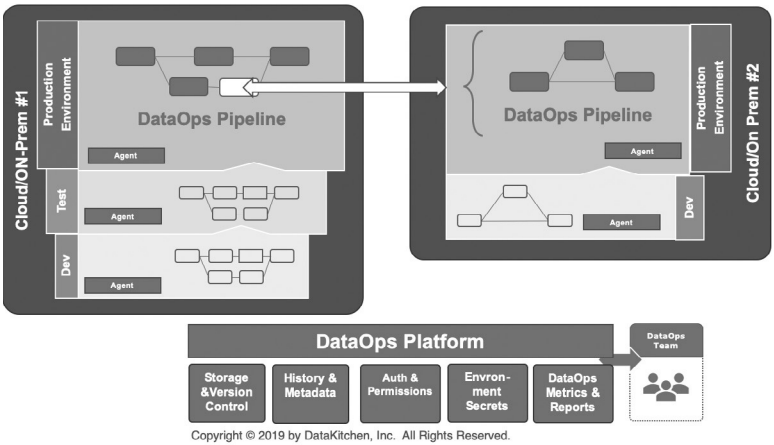


Figure 72: Multi-location DataOps Data Architecture

BUILDING DATAOPS INTO AN EXISTING DATA ARCHITECTURE

Whether your current data architecture is on-prem or in the cloud or a mix of both; whether you have a standard environment or live in a multi-tool world, you can evolve your system to incorporate DataOps functionalities. You can build a DataOps Platform yourself or leverage solutions from the vibrant and growing DataOps ecosystem. DataOps can help you architect your data operations pipeline to support rapid development and deployment of new analytics, robust quality, and high levels of staff productivity.

You have the "Right to Repair" your data architecture — design for it!

Enabling Design Thinking in Data Analytics with DataOps

Want to boost data-analytics innovation? Try "**Design Thinking.**"

Design Thinking is a solution-based design methodology which organizations use to address ill-defined or tricky problems that defy conventional approaches. It uniquely marries design with customer empathy to produce solutions that address latent customer needs. The De-sign Thinking methodology re-frames problems in human-centric ways, creates many ideas in brainstorming sessions, and then adopts a hands-on approach to prototyping and testing. The Design Thinking process boils down to three steps:

- **Empathy** — Gain an understanding of the problem you are trying to solve by consulting experts, observing, and empathizing. The discovery process enables designers to think beyond their own assumptions, about the end user's needs, and the problem space.
- **Ideation** — Generate lots of ideas — as many as possible
- **Experimentation** — Create prototype solutions, test, learn and repeat

Design Thinking has grown beyond its physical design roots to guide innovation in education, business and computer science. As you would expect, data professionals are now applying design thinking to ~~data science~~. Design thinking can serve as a major boost to corporate innovation. Unfortunately, most data organizations are not set-up for a rapid feedback loop of Ideation and Experimentation, so creativity never shifts into high gear.

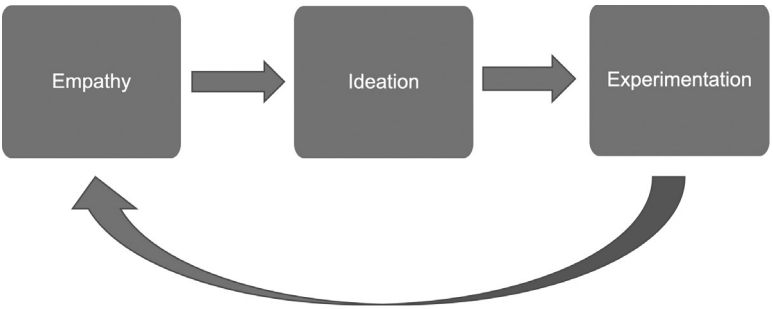


Figure 73: Design Thinking consists of three stages: Empathy, Ideation and Experimentation

Figure 73 shows the stages of Empathy, Ideation and Experimentation in series. As any experienced "Design Thinker" will tell you, the stages do not necessarily happen in sequence. Experimentation can lead to deeper Empathy, which fuels Ideation. The process could have

many feedback loops. One issue that frustrates Design Thinking in data analytics is that Experimentation can take much longer than Empathy and Ideation. It can take weeks or months for the data team to implement a relatively minor change in analytics. Nothing interrupts the creative juices of innovation from flowing like waiting and waiting and waiting some more. When users can't see immediate feedback on their ideas, they may lose interest.

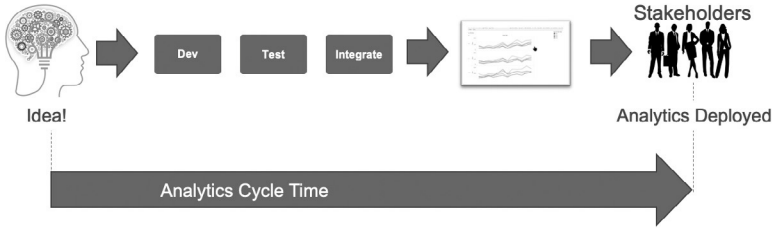


Figure 74: Cycle time is the period of time required to turn a new idea into de- ployed analytics. In many organizations, cycle time is unacceptably long.

FACTORS THAT LENGTHEN CYCLE TIME

We'd like to say that data teams work hand-in-hand with their users like a well-oiled machine, fielding new idea proposals, implementing them rapidly and quickly iterating toward higher-quality models and analytics. Unfortunately, our experience is the opposite. Data teams are constantly interrupted by data and analytics errors. Data scientists spend 75% of their time massaging data and executing manual steps.

Productive Design Thinking depends on a quick turn-around between ideation and experimentation. The problem is that the Experimentation phase can be unacceptably slow. Lengthy analytics cycle time occurs for a variety of reasons:

- Poor teamwork within the data team
- Lack of collaboration between groups within the data organization
- Waiting for IT to disposition or configure system resources
- Waiting for access to data
- Moving slowly and cautiously to avoid poor quality
- Requiring approvals, such as from an Impact Review Board
- Inflexible data architectures
- Process bottlenecks
- Technical debt from previous deployments
- Poor quality creating unplanned work

As daunting as some of these challenges are, some data organizations have proven that it is possible to achieve rapid cycle time. They do this using a methodology called DataOps.

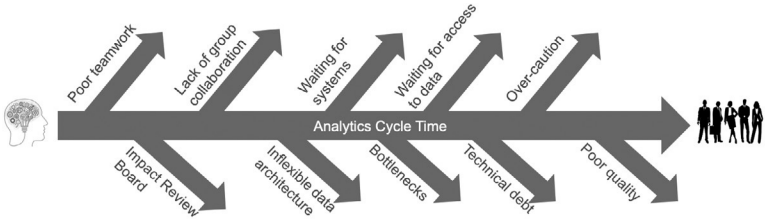


Figure 75: Factors that derail the dev team and lengthen analytics cycle time

HOW DATAOPS MINIMIZES CYCLE TIME

Data analytics can leverage the same processes and methodologies that have boosted the productivity of software engineering 100x in the last decades. We call these techniques (collectively) DataOps. It includes three important methodologies: Agile Software Development, DevOps and statistical process controls (SPC). DataOps requires data teams to rethink how they manage projects, create and deploy new features, publish analytics and control quality:

- **Managing projects** — Features are delivered iteratively using Agile Development. Agile is perfect for the "Ideate and Experiment" cycle characteristic of Design Thinking
- **Creation and deployment of analytics** — Cycle time is minimized when development and production environments are aligned and when tools support seamless teamwork among members of the dev team and between groups within the data organization. DataOps also orchestrates the quality assurance and continuous deployment of code.
- **Operations** — Automated orchestration cleans raw data, executes ETL and publishes analytics as part of the operational workflow.
- **Quality** — Tests validate raw data as well as inputs, outputs and business logic in every stage of operations and new-analytics deployment. Dashboards and real-time alerts provide transparency related to issues.

Quality is an important aspect of cycle time. A team can't reduce cycle time if they are being constantly interrupted by quality problems and high-severity alerts. DataOps applies automated testing to the data operations pipeline as well as the release pipeline for new analytics.

DataOps comprehends that enterprises live in a multi-language, multi-tool, heterogeneous environment with complex workflows. To implement DataOps, extend your existing environment to align with DataOps principles. As we have written extensively, you can implement DataOps by yourself in seven steps, or you can adopt a DataOps Platform from a third party vendor.

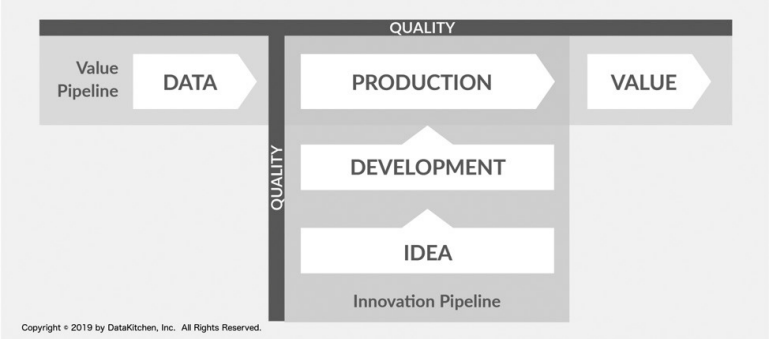


Figure 76: DataOps manages the creation and deployment of analytics ("Innova- tion Pipeline) and orchestrates data operations ("Value Pipeline").

DATAOPS ENABLES DESIGN THINKING

With automated orchestration, end-to-end testing and seamless transitions from dev to production, DataOps minimizes analytics cycle time, enabling the close coupling of Ideation and Experimentation required for Design Thinking. Analytics teams publish changes to analytics quickly and confidently. When users propose ideas, new analytics can be created rapidly, providing immediate feedback. When users see immediate results, it triggers their brain to further Ideate. When implemented at high velocity, Design Thinking can make short work out of an organization's most formidable challenges.

DataOps Puts Agility into Agile Data Warehousing

Data analytics professionals get used to being in no-win situations. Internal customers make a simple request; for example, add a new file to the database. Users expect requests like these to take days, yet, in many large organizations, they require months to complete. At DataKitchen, we repeatedly hear from companies that they need to improve their cycle time for new analytics. One approach, *Agile Data Warehousing*, applies Agile principles to data warehouse projects in an attempt to speed innovation. However, many companies quickly discover that simply implementing Scrum is not sufficient to attain results.

Imagine that you oversee a fifty-person team managing numerous large integrated databases (DB) for a big insurance or financial services company. You have 300 terabytes (TB) of data which you manage using a proprietary database. Between software, licensing, maintenance, support and associated hardware, you pay $10M per year in annual fees. Even putting another single CPU into production could cost hundreds of thousands of dollars.

Someday these large databases will move to the cloud at a fraction of the cost. New databases will be turned on and off like light bulbs with the enterprise only paying for the resources they consume. That's a long-term goal.

In the short term, the team has to produce results using the existing platform. You can't afford the separate instant stations of the entire data set for development, quality assurance (QA), performance testing and production so non-production machines are given subsets of the data. The Necessity of provisioning physically separate hardware instantiations is one barrier to greater Agility. The machine environments are different and have to be managed and maintained separately. New analytics are tested on each machine in turn first in dev, then QA and finally production. You may not catch every problem in dev and QA since they aren't using the same data and environment as production.

Running regression tests manually is time-consuming so it can't be done often. This creates risk whenever new code is deployed. Also, when changes are made on one machine they have to be manually installed on the others. The steps in this procedure are detailed in a 30-page text document, which is updated by a committee through a cumbersome series of reviews and meetings. It is a very siloed and fractured process, not to mention inefficient; during upgrades, the DB is offline, so new work is temporarily on hold.

In our hypothetical company, the organization of the workforce is also a factor in slowing the team's velocity. Everyone is assigned a fixed role. Adding a table to a database involves several discrete functions: a Data Quality person who analyzes the problem, a Schema/Architect who designs the schema, an ETL engineer who writes the ETL, a Test Engineer that writes tests and a Release Engineer who handles deployment. Each of these functions is performed sequentially and requires considerable documentation and committee review before any action is taken. Hand-off meetings mark the transition from one stage to the next.

The team wants to move faster but is prevented from doing so due to heavyweight processes, serialization of tasks, overhead, difficulty in coordination and lack of automation. They need a way to increase collaboration and streamline the many inefficiencies of their current process without having to abandon their existing tools.

HOW DATAOPS HELPS

DataOps is a new approach to data analytics that automates the orchestration of data to production and the deployment of new features, both while maintaining impeccable quality. DataOps does not mandate the use of any particular tool or technology, but support in the following areas can be critical to Agile Data Warehousing in large teams, such as the one described:

Shared Workspace – DataOps creates a shared workspace so team members have visibility into each other's work. This enables the team to work more collaboratively and seamlessly outside the formal structure of the hand-off meeting. DataOps also streamlines documentation and reduces the need for formal meetings as a communication forum.
Orchestration – DataOps deploys code updates to each machine instantiation and automates the execution of tests along each stage of the data analytics pipeline. This includes data and logic tests that validate both the production and feature deployment pipelines. Tests are parameterized so they can run in the subset database of each particular machine environment equally well. As the test suite improves, it grows to reflect the full breadth of the production environment. Automated tests are run repeatedly so you can be confident that new features have not broken old ones.

These tools and process changes together break down the organizational and technology barriers that prevent the team from implementing Agile methods in data analytics. DataOps unburdens the team from non-value-add tasks and empowers them to self-organize around new creative initiatives. When the team is free to innovate, the continuous improvement culture built into DataOps will begin working to reduce the cycle time of new analytics from months to days (and less). This ultimately puts the Agility back into Agile Data Warehousing by delivering high-quality analytics to users in a timely fashion.

Speed Up Innovation with DataOps

LEVERAGING DATA LAKES, DATA WAREHOUSES AND SCHEMAS FOR FASTER ANALYTICS

Analytics professionals often strain to make one change to their analytic pipeline per month. DataOps increases their productivity by an order of magnitude. DataOps accelerates innovation by automating and orchestrating the data analytics pipeline and speeding ideas to production. It does this by applying Agile Development, DevOps and statistical process controls to data analytics. This enables the DataOps Engineer to quickly respond to requests for new analytics while guaranteeing a high level of quality. In order to understand this, it is helpful to know a little about the role of data lakes, schemas and data warehouses in DataOps.

DATAOPS REQUIRES EASY ACCESS TO DATA

When data is moved from disparate silos into a common repository, it is much easier for a data analytics team to work with it. The common store is called a data lake. To optimize DataOps, it is often best to move data into a data lake using on-demand simple storage.

People often speak about data lakes as a repository for raw data. It can also be helpful to move processed data into the data lake. There are several important advantages to using data lakes. First and foremost, the data analytics team controls access to it. Nothing can frustrate progress more than having to wait for access to an operational system (ERP, CRM, MRP, …). Additionally, a data lake brings data together in one place. This makes it much easier to process. Imagine buying items at garage sales all over town and placing them in your backyard. When you need the items, it is much easier to retrieve them from the backyard rather than visiting each of the garage sale sites. A data lake serves as a common store for all of the organization's critical data. Easy, unrestricted access to data eliminates restrictions on productivity that slow down the development of new analytics.

Note that if you put public company financial data in a data lake, everyone who has access to the data lake is an "insider." If you have confidential data, HIPAA data (Health Insurance Portability and Accountability Act of 1996) or Personally identifiable information (PII) — these must be man- aged in line with government regulations, which vary by country.

The structure of a data lake is designed to support efficient data access. This relates to how data is organized and how software accesses it. A database schema establishes the relationship between the entities of data.

UNDERSTANDING SCHEMAS

A database schema is a collection of tables. It dictates how the database is structured and organized and how the various data relate to each other. Below is a schema that might be used in a pharmaceutical-sales analytics use case. There are tables for products, payers, period, prescribers and patients with an integer ID number for each row in each table. Each sale recorded has been entered in the fact table with the corresponding IDs that identify the product, payer, period, and prescriber respectively. Conceptually, the IDs are pointers into the other tables.

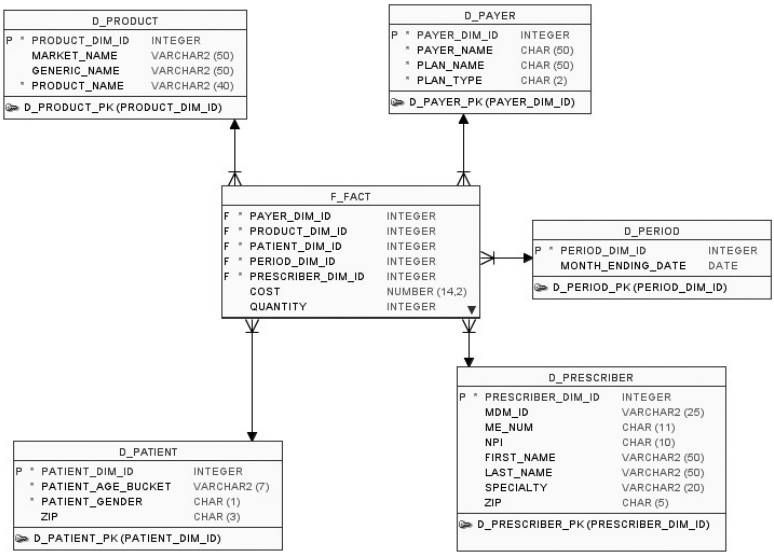


Figure 77: The Schema of a Pharmaceutical-Sales Analytics System

The schema establishes the basic relationships between the data tables. A schema for an operational system is optimized for inserts and updates. The schema for an analytics system, like the star schema shown here, is optimized for reads, aggregations, and is easily under- stood by people.

Suppose that you want to do analysis of patients based on their MSA (metropolitan service area). An MSA is a metropolitan region usually clustered near a large city. For example, Cambridge, Massachusetts is in the Greater Boston MSA. The prescriber table has a zip-code field. You could create a zip-code-to-MSA lookup table or just add MSA as an attribute to the patient table. Both of these are schema changes. In one case you add a table and in the other case you add a column.

TRANSFORMS CREATE DATA WAREHOUSES

The data lake provides easier access, but lacks the optimizations needed for visualizations or modeling. For example, data often enters the data lake in the format of the source system and not using an optimized schema that facilitates analysis. Data warehouses better address analytic-specific requirements. For example, the data warehouse could have a schema that supports specific visualization, modeling or other features.

You might hear the term *data mart* in relation to data analytics. Data marts are a streamlined form of data warehouses. The two are conceptually very similar.
Data transforms (scripts, source code, algorithms, …) create data warehouses from data lakes. In DataOps this process is optimized by keeping transform code in source control and by automating the deployment of data warehouses. An automated deployment process is significantly faster, more robust and more productive than a manual deployment process.

THE DATAOPS PIPELINE

The automation of the pipeline that transforms the schemas of data lakes, creating data warehouses and data marts, is a key reason that DataOps is able to improve the speed and quality of the data analytic pipeline. Without using a data lake, data is highly dispersed, and difficult to access. Schemas of operational systems are difficult to navigate and most likely not optimized for analytics.

DataOps moves the enterprise beyond slow, inflexible, disorganized and error-prone manual processes. The DataOps pipeline leverages data lakes and transforms them into well-crafted data warehouses using continuous deployment techniques. This speeds the creation and deployment of new analytics by an order of magnitude. Additionally, the DataOps pipeline is constantly monitored using statistical process control so the analytics team can be confident of the quality of data flowing through the pipeline. Work Without Fear or Heroism. With these tools and process improvements, DataOps compresses the cycle time of innovation while ensuring the robustness of the analytic pipeline. Faster and higher quality analytics ultimately lead to better insights that enable an enterprise to thrive in a dynamic environment.

How to Inspire Code Reuse in Data Analytics

In DataOps, the data analytics team moves at lightning speed using highly optimized tools and pro- cesses. One of the most important productivity tools is the ability to reuse and containerize code.

When we talk about reusing code, we mean reusing data analytics components. All of the files that comprise the data analytics pipeline scripts, source code, algorithms, html, configuration files, parameter files we think of these as code. Like other software development, code reuse can significantly boost coding velocity.

Code reuse saves time and resources by leveraging existing tools, libraries or other code in the extension or development of new code. If a software component has taken several months to develop, it effectively saves the organization several months of development time when another project reuses that component. This practice can be used to decrease projects budgets. In other cases, code reuse makes it possible to complete projects that would have been impossible if the team were forced to start from scratch.

Containers make code reuse much simpler. A container packages everything needed to run a piece of software code, run times, tools, libraries, configuration files into a stand-alone executable. Containers are somewhat like virtual machines but use fewer resources because they do not include full operating systems. A given hardware server can run many more containers than virtual machines.

A container eliminates the problem in which code runs on one machine, but not on another, because of slight differences in the set-up and configuration of the two servers or software environments. A container enables code to run the same way on every machine by automating the task of setting up and configuring a machine environment. This is one DataOps techniques that facilitates moving code from development to production — the run-time environment is the same for both. One popular open-source container technology is Docker.

Each step in the data-analytics pipeline is the output of the prior stage and the input to the next stage. It is cumbersome to work with an entire data-analytics pipeline as one monolith, so it is common to break it down into smaller components. On a practical level, smaller components are much easier to reuse by other team members.

Some steps in the data-analytics pipeline are messy and complicated. For example, one operation might call a custom tool, run a python script, use FTP and other specialized logic. This operation might be both hard to set up, because it requires a specific set of tools, and difficult to create, because it requires a specific skill set. This scenario is another common use case for creating a container. Once the code is placed in a container, it is much easier to use by other programmers who aren't familiar with the custom tools inside the container but know how to use the container's external interfaces. All of the complexity is embedded in- side the container. It is also easier to deploy that code to different environments. Containers make code reuse much more turnkey and allow developers much greater flexibility in sharing their work with each other.

Plumbing Wisdom for Data Pipelines

While admiring the latest cloud tech, don't forget that humans have been debugging pipelines since the Romans built the aqueducts. Any good plumber can give you some hard-won tips on managing data pipelines effectively, insights that might save your career from going down the drain.

While you're admiring the latest cloud tech, don't forget that humans have been debugging pipelines, at least since the Romans built the aqueducts. Any good plumber can give you some hard-won tips on managing data pipelines effectively, insights that might save your career from going down the drain.

Your overalls are cleaner when you work on new construction.

Green fields smell sweet, and a nice long design phase can give you a rosy view of your project. The real world is a bit more pungent. That's why a key goal of DataOps is to expose your best-laid plans to real-life urgencies faster and initiate that crucial feedback loop sooner. With DataOps, you define a framework and a set of best practices to deal with real-world changes and crises. It's about tapping into that feedback loop earlier to keep your process reliable — and relevant.

It's not done until you've run the water through it.

Good testing, like exercise and veganism, is the subject of fervent talk and half-hearted action. There are lots of reasons good people test inadequately. Deadlines are high-profile, and details go unnoticed. The stakes may be high, but the reckoning seems distant. It's possible, and often irresistible, to move on to the next emergency before the last ingenious solution has been proven out. But water is inexorable, and plumbers quickly face the consequences of their mistakes. Testing is intrinsic to the job.

So you've checked that it works — are you done? Of course not. Plumbers don't just check the water level; they may install a low-water cutoff switch or a relief valve so the system responds to changing inputs. They might install a leak detector to monitor for problems automatically, and send alerts, while the house is unattended. By automating your tests, then running them with each refresh, you build in safety valves for your data pipeline.

Don’t over-tighten your fittings.
Some lessons you learn the hard way. Materials expand over time, and today’s perfect fit can become tomorrow’s stuck coupling or busted pipe. In data engineering, too, systems fail when tolerances are too tight. Your schedule between the end of one process and the kick-off of another needs to account for delays that inevitably crop up. Your data types must strike a balance between tight checks and resilient operations. Every process has dimensions of variability you can’t control. Expect perfection, and you’ll guarantee it won’t happen.

Don’t hang junk off your pipes.
Every plumber has a story about the folks who found it convenient to hang their wardrobe from the water pipes — until they flooded the basement. Data Engineers take similar risks when they over-complicate their code or tack on extraneous processes at the wrong points in a critical path build. Organize your data flows by function and criticality. You want a simple structure so people can immediately jump in and orient themselves. Your goal isn’t to intimidate people or squeeze the most complexity into the smallest space. Even if it looks simple now, it’s going to get more complicated over time — don’t create an unwieldy mess.

Read the plans, but believe the piping.
When maintaining a home, blueprints and schematics can give you lots of information, some of which are even accurate. Plumbing “as-built” can vary greatly from “as-designed,” and the only reliable way to know what you’re working with is to go to the pipes. As data engineers, we strive to divide our grumbling equally between the awful documentation we must decipher and the hassle of documentation we have to produce. It is even easier to skimp on documentation than testing. The gap between specs and reality starts to grow from the day the specs are defined.
There are two solutions to this. One is to set clear rules that documentation must be thorough and up-to-date. These rules will be carefully filed away with your documentation. The alternative is to acknowledge that the primary source of your docs will always be your data structures and code. The strategy: lower the friction to document your team’s work. If you encourage your team to comment consistently, you can rely on tools to automate daily refreshes of your knowledge base for everybody else. Details will be more up-to-date. Gaps will become readily apparent. Your documentation efforts, like your business, will be data-driven.

Tell the residents before you shut off the water on Saturday night.
Water and mediocrity flow downhill. Good plumbers realize how much people depend on their work. You don’t have to tell a plumber that sh*t happens. But transparency and good communication will get you out of many jams. People appreciate knowing what’s happening and how it might impact their work. Once you realize that part of your work is managing downstream expectations, you’ll get more of the credit you deserve.

Don’t bite your nails on the job.

Data engineers are human, and human frailties are part of the systems you build. Any pipeline that expects people to get it right every time is not just a source of stress – it’s structurally deficient. If you build out systemic solutions to human error, you will have better pipelines and a more engaged, effective team.
There’s a different issue here for plumbers. I won’t get into it. Either way, if you follow the rules above, you will be flushed with success!

Infographic – Data Engineers are Burned Out and Calling for DataOps

A survey commissioned by data.world and DataKitchen reveals a disturbing state of affairs among data engineering professionals. The study of 600 data engineers, conducted by Wakefield Research, suggests an overwhelming majority are burned out and calling for relief. This infographic highlights the results.

Data engineers aren't the only ones suffering – the struggle is organization wide. Only...

1/3
of companies confirm that the CDO (Chief Data Officer) role is successful and established

24%
of companies have created a data-driven organization

29%
are achieving transformational business outcomes with data

How do we make it right?

DataOps: A shining light

89%
agree technology alone is ineffective without processes that deploy, monitor, and manage analytics throughout the lifecycle

78%
of data engineers say it's essential or very important for their company to incorporate DataOps in order to successfully manage its data processes

What is DataOps?

DataOps is a collection of technical practices, workflows, cultural norms, and architectural patterns that enable:

- Rapid innovation and experimentation that delivers new insights
- Extremely high quality and very low error rates
- Collaboration across people, technology, and environments
- Clear and precise measurement and monitoring

Read the white paper to learn how DataOps can improve the lives of your data engineers

DOWNLOAD NOW

Sources:
data.world and DataKitchen, Data Engineer Survey 2021
New Vantage Partners, Big Data and AI Executive Survey 2021 Executive Summary of Findings

data.world DataKitchen

10 DataOps Principles for Overcoming Data Engineer Burnout

For several years now, the elephant in the room has been that data and analytics projects are failing. Gartner estimated that 85% of big data projects fail. Data from New Vantage partners showed that the number of data-driven organizations has actually declined to 24% from 37% several years ago and that only 29% of organizations are achieving transformational outcomes from their data.

In addition, only one-third of companies have an established CDO role, and the average tenure of the CDO is only 2.5 years. Add all these facts together, and it paints a picture that something is amiss in the data world.

Yet, among all this, one area that hasn't been studied is the data engineering role. We thought it would be interesting to look at how data engineers are doing under these circumstances. Are they thriving or feeling the impact of failed projects? We surveyed 600 data engineers, including 100 managers, to understand how they are faring and feeling about the work that they are doing. The top-line result was that 97% of data engineers are feeling burnout.

If you have worked in the big data industry, you will likely resonate with the survey participants. Data engineering resembles software engineering in certain respects, but data engineers have not adopted the best practices that software engineering has been perfecting for decades. When there is a problem in data pipelines, data engineers are expected to fix it using ad hoc processes that simply will not scale.

Data engineering burnout stems from never succeeding in getting systems under control. There's an unending wave of problems: customer requests, broken systems, and errors. One problem is that data engineers are seen as a cost minimization role instead of a generator of value. People think of them as firefighters, and when there is an outage, they should be working day and night to address it. Data engineers end up fixing the same problem over and over.

Enterprises must empower data engineers to fix processes instead of just bugs. Imagine a data pipeline error or data problem that impacts critical analytics. Most organizations find out about these errors from their customers, such as a VP of Sales who notices that the bookings report is millions of dollars off. This oversight triggers an all-hands-on-deck emergency response. If data pipelines span teams, then there is an unpleasant (and often political) discovery phase where people may point fingers at each other. Under tremendous pressure and scrutiny, the data team works the weekend to rush a fix into the production pipeline. When the problem is addressed, everyone heaves a sigh of relief. The heroes who fixed the problem are sometimes praised, or perhaps they are scorned for letting the problem occur in the first place.

One widespread problem is a reliance on the culture of heroism. People jump whenever there is a problem, but heroism is not a strategy. It doesn't scale. It leads to burnout and turnover. When people leave the company, they take tribal knowledge with them, so burnout has a tangible business impact. In our survey, data engineers cited the following as causes of burnout:

The relentless flow of errors

Manual processes crowd out innovation

Steady Stream of Half-Baked Requests

Blaming and finger-pointing

Restrictive data governance Policies

For see the entire results of the data engineering survey, please visit "2021 Data Engineering Survey: Burned-Out Data Engineers are Calling for DataOps."

Methods to Avoid Burnout

The data engineers in our survey have many challenges. They wake up the day after an analytics deployment and worry about the little things they didn't check. They get yelled at for missing milestones. Users roll their eyes when the analytics team can't turn on a dime. Everyone in the industry faces these problems. The important thing to realize is that these problems are not the fault of the people working in the data organization. They are process problems. The data analytics lifecycle is a factory, and like other factories, it can be optimized with techniques borrowed from methods like lean manufacturing.
Instead of directing negativity at the enterprise or others or oneself, data engineers can use workplace challenges as fuel to transform suffering into system building. The problems of burnout and lack of data project success are not an indication of personal failure or that the field is broken. It points to work that needs to be done to build systems around automation and data toolchains. To this end, we offer ten tips to avoid data engineering burnout.

1. Don’t suffer; build a system

Many data engineers view their job as servicing a queue of tasks. The faster they can take tasks off the queue and check them off, the more they feel they have accomplished. When faced with a task, many data engineers try to address it, thinking only about the task as defined. If the problem is a bug, you fix the bug and move on.

What if that same problem recurs over and over? Take a broad, holistic view of the situation. Don’t just fix the bug. Improve the system so that the problem does not recur ever again. Instead of just answering an analytics question, think about how to answer the following ten similar questions. Instead of just solving a problem manually, build a robot that will solve the problem for you every time it occurs in the future.

Write tests that catch data errors. Automate manual processes. Implement methods. Build observability and transparency into your end-to-end data pipelines. Automate governance with governance-as-code . Instead of fixing bugs and creating analytics, design the system that enables rapid design of robust, observable and governed analytics. Build a system that eliminates the causes of data engineering suffering.

The critical question that data engineers need to ask is whether their customers (users, colleagues, consumers) are getting value out of data. Forget how hard you work. Consider how to shrink the task queue through automation and expend the least possible manual effort to transform data to value for customers.

2. Don’t agonize — automate that sh

Leader of data teams need to make time for the team to automate tasks. Celebrate automation as an activity that generates tremendous value. This transformation has already taken place in software engineering.

Twenty years ago, a software dev team could have one release engineer for a 35 person dev team. In those days, the release engineer was seen as a lesser role and compensated less than the developers. Today the role of release engineer is called DevOps engineer, and most organizations devote 25–28% of their staff to automating software development processes. Why? Because automation improves velocity and generates value. Data teams need to follow the lead of software engineering teams.

When we say automation, we don’t mean finding ways for the pair of hands on the keyboard to generate SQL faster. The world is full of languages and tools, and each has its strengths and weaknesses. Automation is the system around the tools. The system creates on-demand development environments, performs automated impact reviews, tests/validates new analytics, deploys with a click, automates orchestrations, and monitors data pipelines 24×7 for errors and drift. These forms of automation will improve analytics productivity and quality by 10x.

We talk about systemic change, and it certainly helps to have the support of management, but data engineers should not underestimate the power of the keyboard. The person sitting in front of the keyboard with the technical skills to create data integrations, transformations and analytics can make change one test and one automated orchestration at a time. It is just as heroic to develop a robust automated pipeline as it is to work the weekend fixing a broken piece of analytics. Data engineers should feel empowered to reimagine the environment in which they work with robust systems and designs that scale.

3. Run toward errors

Errors are embarrassing. They make us look bad. A person that is blamed for errors might lose their position of respect or authority in an organization. People instinctively turn away from mistakes.

Data has errors, and it will always have errors. Don't hide them. Don't forget about them. Errors are opportunities to automate. Errors are opportunities for improvement. Run towards errors. Embrace your errors and use them to build a system that prevents the same errors from ever recurring.

4. Don't be afraid to make a change

One of the cardinal sins of data engineering is overcaution. In an environment with a continuous flow of errors, where shame and blame are the norms, data engineers can be scared to deploy code or architecture changes. One of the common responses to fear is to proceed more slowly and carefully. To avoid errors and outages, fearful engineers give each analytics project a more extended development and test schedule. Effectively, this is a decision to deliver higher quality but fewer features to users. Overcaution is a double-edged sword. The slow and methodical approach often makes users unhappy because analytics are delivered more slowly than their stated delivery requirements. As requests pile up, the data analytics team risks being viewed as bureaucratic and unresponsive.
Data engineers do not have to choose between agility and quality. An automated system can test and deploy new analytics in a staging environment aligned with production. If data pipelines have thousands of tests verifying quality, the data team can be sure that pipelines are operating correctly and that data and analytics are sound. Automation and observability enable you to proceed with confidence. With the proper methods and systems in place, you don't have to fear change.

5. 'Done' means live in your customer's hands

When is a project ready to be pushed to production? When can you declare it done?
When schedules are tight, it's tempting to address a task as narrowly as possible, throw it over the wall and declare victory. If there are side effects, it is someone else's problem.
Data engineers better serve their mission when they define "done" in terms of creating value for customers. Focusing on data as a whole product, instead of narrowly defining projects, helps to tear down the walls between groups within the enterprise. Viewing data as a product puts the customer's success at the top of the priority list.

In concrete terms, a project is done when a system has been built around it. You can orchestrate tools, team environments and processes in one single pipeline. Code is version controlled and parameterized for reuse across different environments. Data pipelines have enough automated tests to catch errors, and error events are tied to end-to-end observability frameworks. Data engineers would do well to take a holistic view of projects and think about how changes fit into the overall system.

6. Don't be a hero; make heroism a rare even

Sometimes heroism saves the day. Sometimes circumstances force you to put in that long night, do the thing that doesn't scale, and get the job done for the customer. The trick is to circle back the next day and work on the robust fix that prevents that problem from recurring. With discipline, a team can slowly wean off its dependency on heroism. Heroism doesn't ever go to zero, but if it happens all the time, the team will be exhausted and burned out. Exhausted engineers tend to change jobs, leaving the rest of the data team holding the bag.

7. Seek out opportunities for reuse and sharing

The work that engineers perform in data and analytics is code. A sure way to create more work is to copy someone else's code, tweak it a little bit and create a variant that has to be supported independently of the original code. Organizations that operate in this way are soon buried under the crush of technical debt.

Data engineers are in the " complexity " business, so anything that reduces complexity is a "win." If data teams can build reusable components , they gain scale by sharing those components across numerous pipelines. Data teams have large distributed systems, numerous tools and many data sources. Abstracting and grouping related features into functional units, making them reusable and giving them APIs (application programming interfaces), helps create leverage that scales while minimizing technical debt. Design scalability and reuse are essential features that data engineers shouldn't dismiss in pursuit of delivering data value for our customers.

8. Practice DataGovOps

Data governance is a necessity, not a luxury. Data governance needs to control critical data, but governance needs to serve the higher mission of helping create value from data. DataGovOps applies DataOps principles to governance in the same way that Agile and DevOps apply to product development.

Organizations need to be agile in defining who accesses data, what kind of access is appropriate and how to utilize data to maximize its value. At the same time, data governance can't compromise on its mission to prevent data from being used in unethical ways, contrary to regulations or misaligned strategically.

DataGovOps seeks to strike the right balance between centralized control and decentralized freedom. When governance is enforced through manual processes, policies and enforcement interfere with freedom and creativity. With DataOps automation, control and creativity can coexist. DataGovOps uniquely addresses the DataOps needs of data governance teams who strive to implement robust governance without creating innovation-killing bureaucracy. If you are a governance professional, DataGovOps will not put you out of a job. Instead, you'll focus on managing change in governance policies and implementing the automated systems that enforce, measure, and report governance. In other words, governance-as-code .

9. Measure your processes (and improve them)

Data engineers can apply their numerical and analytical skills to measure the organizations in which they work. Process metrics for data engineering and data science teams help an organization understand itself and how it can improve. If you start to measure activities, you may be surprised to discover the amount of time people spend in meetings, how many errors reach production, and how long it takes to make changes to analytics. These critical measurements of error rates, cycle times and productivity shed light on how people work as a team and how the data team creates value for customers.

10. Don't put your head in the sand — focus on value delivery

DevOps has become so important in the software engineering space, yet some people still don't understand the need for DataOps in the data space. There's a mistaken notion out there that adding more robust automated processes will slow down data teams.

One way to think about DataOps is like the brakes on a car. Why does a car have brakes? Is it to go slowly? No, brakes help you go *fast safely* . The engineering processes and automation that we have described are intended to help data engineers go faster. They take a little more time to set up initially, but they save huge amounts of time downstream.

Forging a Path to Agile and DataOps

The ideas discussed above stem from methods developed 30–40 years ago in industrial factories where teams worked together on complex projects. These ideas evolved into the Toyota production system and Lean Manufacturing.

People encountered similar problems in how software is built, and the DevOps and Agile methodologies were born. Now we are conducting the same discussion in data and analytics. Data organizations have large teams that must work together amid an enormous amount of complexity.

We know from the auto industry that cars that once drove only 50,000 miles can now last 150,000 miles. Perhaps electric vehicles will be able to go 500,000 miles. These improvements are the result of quality methods that focus on cycle time and eliminating waste. The same thing can happen in the big data industry. We can go faster and farther using fewer resources. DataOps methods can help data organizations follow the path of continuous improvement forged by other industries and prevent data team burnout in the process.

For more information, DataKitchen has published the "The DataOps Cookbook," which helps guide you through everything you need to know about DataOps. For the survey results, see DataOps Cookbook 2021 Data Engineering Survey Results.

The Ten Standard Tools To Develop Data Pipelines In Microsoft Azure

The Ten Standard Tools To Develop Data Pipelines In Microsoft Azure. Is it overkill? Paradox of choice? Or the right tool for the right job? We discuss.

The Ten Standard Tools To Develop Data Pipelines In Microsoft Azure.

While working in Azure with our customers, we have noticed several standard Azure tools people use to develop data pipelines and ETL or ELT processes. We counted ten 'standard' ways to transform and set up batch data pipelines in Microsoft Azure. Is it overkill? Don't they all do the same thing? Is this the paradox of choice? Or the right tool for the right job? Let's go through the ten Azure data pipeline tools

Azure Data Factory: This cloud-based data integration service allows you to create data-driven workflows for orchestrating and automating data movement and transformation.

Azure Databricks Workflows: An Apache Spark-based analytics platform optimized for the Microsoft Azure cloud services platform. You can use it for big data analytics and machine learning workloads. Workflows is a DAG runner embedded in Databricks. Databricks Notebooks are often used in conjunction with Workflows.

Azure Databricks Delta Live Tables: These provide a more straightforward way to build and manage Data Pipelines for the latest, high-quality data in Delta Lake.

SQL Server Integration Services (SSIS): You know it; your father used it. It does the job.

Azure Synapse Analytics Pipelines: Azure Synapse Analytics (formerly SQL Data Warehouse) provides data exploration, data preparation, data management, and data warehousing capabilities. It provides data prep, management, and enterprise data warehousing tools. It has a data pipeline tool, as well. We just learned of this too.

Azure Logic Apps: This service helps you schedule, automate, and orchestrate tasks, business processes, and workflows when integrating apps, data, systems, and services across enterprises or organizations.
Azure Functions: You can write small pieces of code (functions) that will do the transformations for you.
Oozie on HDInsight. Azure HDInsight: A fully managed cloud service that makes processing massive amounts of data easy, fast, and cost-effective. It can use open-source frameworks like Hadoop, Apache Spark, etc. Oozie is an open-source DAG runner.
Power dataflows: Power BI dataflows are a self-service data preparation tool. They allow users to create data preparation logic and store the output data in tables known as entities.
Azure Data Factory Managed Airflow: The super popular DAG runner managed by Azure
Extra Bonus: choose from one of the 100+ ETL and data pipeline tool options in the Azure Marketplace: DBT Cloud, Matillion, Informatica, Talend, yadda-yadda.

Indeed, designing patterns involving data pipelines often involves using multiple tools in conjunction, each with its strengths. Here are a few examples that we have seen of how this can be done:
Batch ETL with Azure Data Factory and Azure Databricks: In this pattern, Azure Data Factory is used to orchestrate and schedule batch ETL processes. Azure Blob Storage serves as the data lake to store raw data. Azure Databricks, a big data analytics platform built on Apache Spark, performs the actual data transformations. The cleaned and transformed data can then be stored in Azure Blob Storage or moved to Azure Synapse Analytics for further analysis and reporting.
Data warehousing with Azure Synapse Analytics and Power BI: Data from various sources can be ingested into Azure Synapse Analytics, where it can be transformed and modeled to suit the needs of your analysis. The transformed data is then served to Power BI, where it can be transformed, visualized, and further explored.
Machine learning workflows with Azure Machine Learning and Azure Databricks: Azure Databricks can be used to preprocess and clean data, then the transformed data can be stored in Azure Blob Storage. Azure Machine Learning can then use this data to train, test, and deploy machine learning models. The models can be served as a web service through Azure Functions or Azure Kubernetes Service**.**
Serverless data processing with Azure Logic Apps and Azure Functions: Azure Logic Apps can orchestrate complex workflows involving many different services. As part of these workflows, Azure Functions can be used to perform small pieces of data transformation logic. This pattern is suitable for scenarios where the volume of data is not too large and transformations can be performed statelessly.

THERE WILL NEVER BE ONE “BEST” DATA TRANSFORMATION PIPELINE TOOL.

Data transformation is a complex field featuring varying needs and requirements depending on data volume, velocity, variety, and veracity. This complexity leads to diverse tools, each catering to a different need. There are several reasons why there will never be one single “best” data transformation pipeline tool:

- Different use cases: Different tools are optimized for different use cases. Some tools are excellent for batch processing (e.g., Azure Data Factory), some are built for real-time streaming (e.g., Azure Stream Analytics), and others might be more suited for machine learning workflows (e.g., Azure Machine Learning). Depending on your use case, one tool (or a mix of tools) might be more suitable.
- Diverse Data Sources: In the modern world, data comes from various sources, including traditional databases, IoT devices, cloud services, APIs, and more. Each source might require a different tool to extract and manage data effectively.
- Processing Needs: The computational requirements can vastly differ based on the nature of the transformation. A lightweight, serverless function might adequately handle simple transformations, while complex, data-intensive tasks might need a robust big data solution like Azure Databricks.
- Scalability: Not all tools are designed to scale in the same way. Some might be excellent at handling small to medium data volumes but need help with petabyte-scale data.
- Expertise and Learning Curve: Different teams have different skill sets. A tool that is easy to use for a team experienced in Python might not be the best choice for a team specializing in SQL.
- Cost: Different tools have different pricing structures. One tool might be preferred over another, depending on the budget and cost-effectiveness.

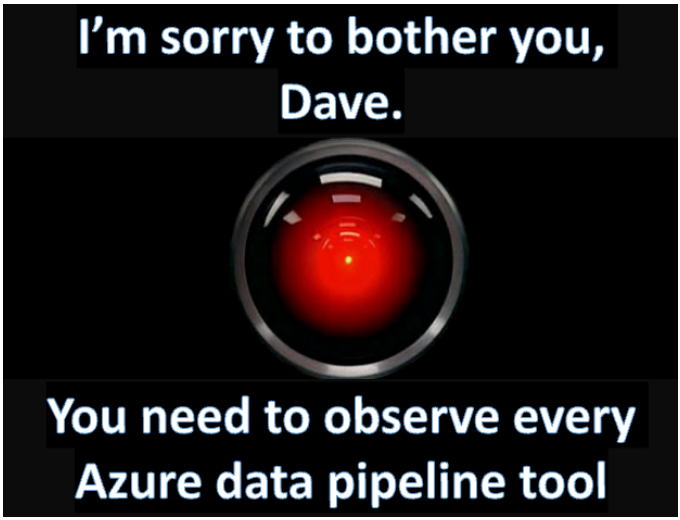
The "toolbox approach" in Microsoft Azure and other similar platforms is beneficial. Azure provides a "shopping" experience, where based on the project's needs, you can pick and choose the tools that best fit your requirements. This flexible and modular approach allows you to build a data pipeline tailored to your needs. In other words, it's not about finding the best tool but the right tool (or tools) for the job.

Data Pipeline Tools And The Paradox of Choice

Psychologist Barry Schwartz studied the effects of more choices and noticed that having more choices doesn't always help us choose better but can make us feel worse about what we chose, even if it was great. Reduced happiness arises from expectation escalation, lost opportunity cost, regret, and self-blame. When we continuously chase the cool new tool, are we focused on delivering value to our customers?
Schwartz may have been right, but simplicity may not be possible in your organization. Maybe you have a history of legacy data pipelines, your organization structure allows for freedom of tool choice, or your engineering philosophy is the 'best tool for the job.' More and more companies are using multiple tools with functional overlap. How can we operate successfully in this multi-tool world?

How To Deal With The Risk Of Multiple Tools

When considering how organizations handle serious risk, you could look to NASA. The space agency created and still uses "mission control" with many screens sharing detailed data about all aspects of a space flight. That shared information is the basis for monitoring mission status, making decisions and changes, and communicating with everyone involved. It is the context for people to understand what's happening now and to review later for improvements and root cause analysis.



Any data operation, regardless of size, complexity, or degree of risk, can benefit from the same level of visibility as that enabled by DataKitchen DataOps Observability. Its goal is to provide visibility across every journey data takes from source to customer value across every tool, environment, data store, analytic team, and customer so that problems are detected, localized, and raised immediately. DataOps Observability does this by monitoring and testing every step of every data and analytic pipeline in an organization, in development and production, so that teams can deliver insight to their customers with no errors and a high rate of innovation.

D*ataOps Observability aims to provide visibility across every journey that data takes from source to customer value across every tool, environment, data store, data, analytic team, and customer so that problems are detected, localized, and raised immediately.*

We call that multi-tool data assembly line a 'Data Journey.' And just like data in your database, the data across the Data Journey, and the technologies that make up the Data Journey, all have to be observed, tested, and validated to ensure success. Successful Data Journeys track and monitor all levels of the data stack, from data to tools to servers to code across all critical dimensions. A Data Journey supplies real-time statuses and alerts on start times, processing durations, test results, technology and tool state, and infrastructure conditions, among other metrics. With this information, data teams will know if everything is running on time and without errors and immediately detect the parts that didn't.

DataOps for the Data Scientist

A Great Model is Not Enough: Deploying AI Without Technical Debt

DATAOPS IN DATA SCIENCE AND MACHINE LEARNING

AI and Data Science are *all the rage*, but there is a problem that no one talks about. Machine learning tools are evolving to make it faster and less costly to develop AI systems. But deploying and maintaining these systems over time is getting exponentially more complex and expensive. Data science teams are incurring enormous technical debt by deploying systems without the processes and tools to maintain, monitor and update them. Further, poor quality data sources create unplanned work and cause errors that invalidate results.

A couple of years ago, Gartner predicted that *85 percent of AI projects would not deliver for CIOs*. Forrester affirmed this unacceptable situation by stating that *75% of AI projects underwhelm*. We can't claim that AI projects fail only for the reasons we listed. We can say, from our experience working with data scientists on a daily basis, that these issues are real and pervasive. Fortunately, data science teams can address these challenges by applying lessons learned in the software industry.

TRADITIONAL MODEL DEVELOPMENT VERSUS AI

AI is most frequently implemented using machine learning (ML) techniques. Building a model is different than traditional software development. In traditional programming, data and code are input to the computer which, in turn, generates an output. This is also true of traditional modeling where a hand-coded application (the model) is input to a computer, along with data, to generate results.

In machine learning, the code can learn. The ML application trains the model using data and target results. An ML model developer feeds training data into the ML application, along with correct or expected answers. Errors are then fed back into the learning algorithm to boost

the model's precision. This procedure continues until the model reaches the target level of accuracy. When complete, this process generates a set of parameters that are described by a set of files (code and configuration). In production, the model evaluates input data and gen- erates results. Figure 62 shows the different processes governing traditional and ML model development.

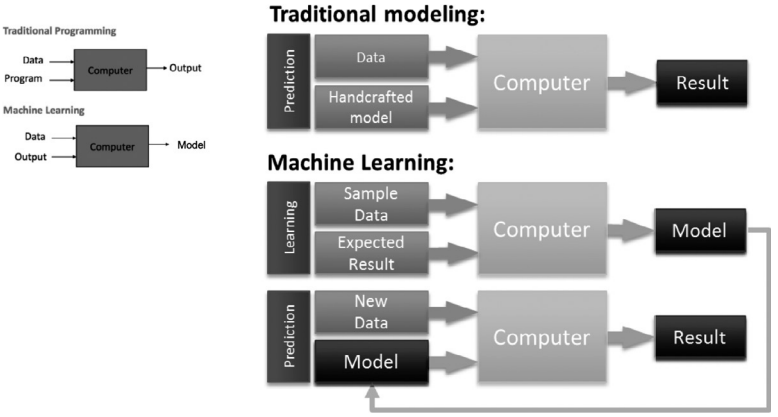


Figure 62: Traditional model programming versus machine learning model development

Below, Figure 63 further elaborates on the complex set of steps that are involved in model building. Naturally, AI projects begin with a business objective. Data is often imperfect so the team has to clean, prepare, mask, normalize and otherwise manipulate data so that it can be used effectively. Feature extraction identifies metrics (measured values) that are informative and facilitate training. After the building and evaluation phases (see Figure 63), the model
is deployed, and its performance is monitored. When business conditions or requirements change, the team heads back to the lab for additional training and improvements. This process continues for as long as the model is in use.

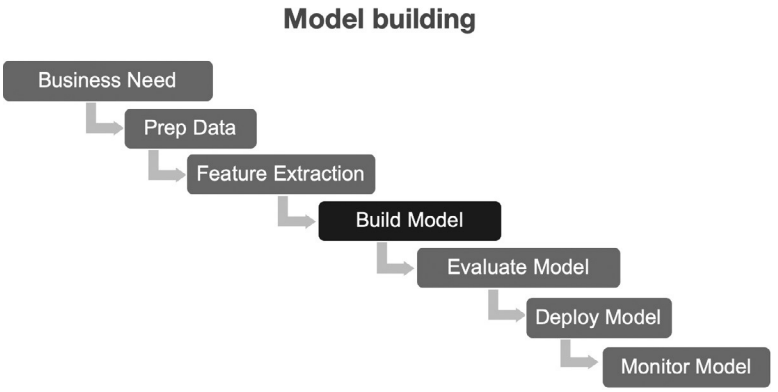
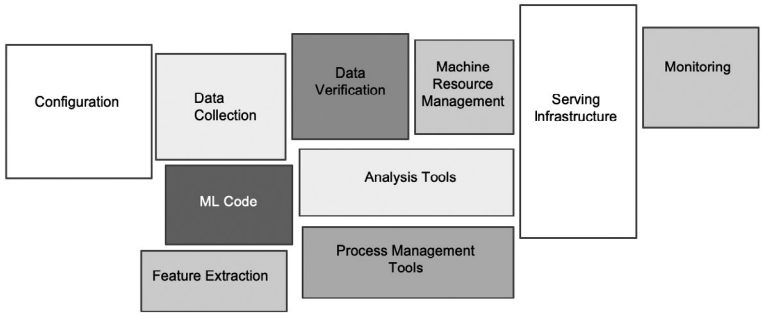


Figure 63: The complex sequence of steps in building an ML model

AI development and deployment is a complex workflow. If executed manually, it is slow, error-prone and inflexible. The actual output of the model development process (a set of files, scripts, parameters, source code, …) is only a small fraction of what it takes to deploy and maintain a model successfully. Figure 64 below shows the *Machine Learning Code* in a system context. Notice that the ML code is only a small part of the overall system.



Source: *Hidden Technical Debt in Machine Learning Systems*, **Advances in Neural Information Processing Systems 28 (NIPS 2015)**

Figure 64: Machine Learning Code is only a small part of the overall system.

Model creation and deployment commonly use the tools shown in Figure 65. Note that this is only a portion of what is required by the system in Figure 64. If the responsibility for these processes and toolchains falls on the data science team, they can end up spending the majority of their time on data cleaning and data engineering. Unfortunately, this is all too common in contemporary enterprises. Addressing this situation requires us to take a holistic view of the value pipeline and analytics creation.

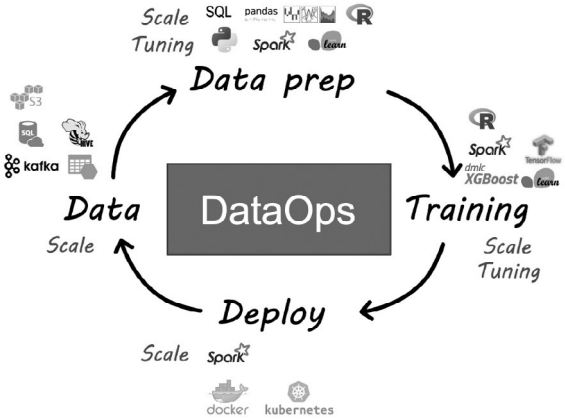


Figure 65: Model development and deployment requires a complex toolchain

TWO JOURNEYS / TWO PIPELINES

We conceptualize AI (and all data analytics) as two intersecting pipelines. In the first pipeline, data is fed into AI and ML models producing analytics that deliver value to business stakeholders. For example, an ML model reviews credit card purchases and identifies potential fraud. We call this the *"*Value Pipeline*."*

The second pipeline is the process for new model creation — see Figure 62 and Figure 63. In the development pipeline, new AI and ML models are designed, tested and deployed into the Value Pipeline. We call this the *"*Innovation Pipeline*."* Figure 66 depicts the Value and Innovation Pipelines intersecting in production.

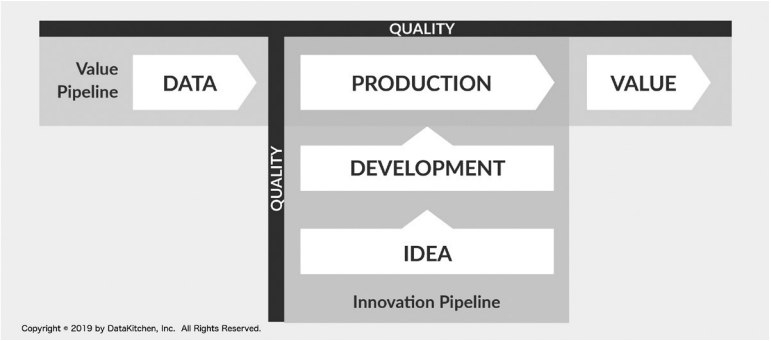


Figure 66: The Value and Innovation Pipelines

Conceptually, each pipeline is a set of stages implemented using a range of tools. The stages may be executed serially, parallelized or contain feedback loops. In terms of artifacts, the pipeline stages are defined by files: scripts, source code, algorithms, html, configuration files, parameter files, containers and other files. From a process perspective, all of these artifacts are essentially just *source code*. Code controls the entire data-analytics pipeline from end to end: ideation, design, training, deployment, operations, and maintenance.

When discussing code and coding, data scientists who create AI and ML models, often think *"this has nothing to do with me."* I am a data analyst/scientist, not a coder. I am an ML tool expert. In process terms, what I do is just a sophisticated form of configuration. This is a common misconception and it leads to technical debt. When it is time for that debt to be paid, the speed of new analytics development (cycle time) will slow to a crawl.

AI / ML / DATA SCIENCE WORK IS JUST CODE

Tools vendors have a business interest in perpetuating the myth that if you stay within the well-defined boundaries of their tool, you are protected from the complexity of software development. This is ill-considered albeit well-meaning.

Don't get us wrong. We love our tools, but don't buy into this falsehood. The $50+ billion AI market is divided into two segments: "tools that create code" and "tools that run code." The point is — AI is code. The data scientist creates code and must own, embrace and manage the complexity that comes along with it.

LESSONS LEARNED — DATAOPS

The good news is that when AI is viewed through a different lens, it can leverage the same processes and methodologies that have boosted the productivity of software engineering 100x in the last decades. We call these techniques (collectively) DataOps. It includes three important methodologies: Agile Software Development, DevOps and statistical process controls (SPC).

• Studies show that software development projects complete significantly faster and with far fewer defects when **Agile Development**, an iterative project management methodology, replaces the traditional Waterfall sequential methodology. The Agile methodology is particularly effective in environments where requirements are quickly evolving — a situation well known to data science professionals. Some enterprises understand that they need to be more Agile and that's great. *(Here's your chance to learn from the mistakes of many others.)* You won't receive much benefit from Agile if your quality is poor or your deployment and monitoring processes involve laborious manual steps. "Agile development" alone will not make your team more "agile."

• **DevOps**, which inspired the name DataOps, focuses on continuous delivery by leveraging on-demand IT resources and by automating test and deployment of code. *Imagine clicking a button in order to test and publish new ML analytics into production.* This merging of software development and IT operations reduces time to deployment, decreases time to market, minimizes defects, and shortens the time required to resolve issues. Borrowing methods from DevOps, DataOps brings these same improvements to data science.

• Like lean manufacturing, DataOps utilizes **statistical process control** (SPC) to monitor and control the Value Pipeline. When SPC is applied to data science, it leads to remarkable improvements in efficiency and quality. With SPC in place, the data flowing through the operational system is verified to be working. If an anomaly occurs, the data team will be the first to know, through an automated alert. Dashboards make the state of the pipeline transparent from end to end.
• DataOps eliminates technical debt and improves quality by orchestrating the Value and Innovation Pipelines. It catches problems early in the data life cycle by implementing tests at each pipeline stage. Further, it greatly accelerates the development of new AI, enabling the data science team to respond much more flexibly to changing business conditions.

DATAOPS FOR YOUR AI AND ML PROJECT

DataOps is an automated, process-oriented methodology, used by analytic and data teams, to improve the quality and reduce the cycle time of data analytics. DataOps is not any specific vendor's solutions. It leverages automation, tools and other best practices.

You can implement DataOps for your AI and ML project yourself by following these seven steps:

Step 1 — Add Data and Logic Tests

To be certain that the data analytics pipeline is functioning properly, it must be tested. Testing of inputs, outputs, and business logic must be applied at each stage of the data analytics pipeline. Tests catch potential errors and warnings before they are released so the quality remains high. Manual testing is time-consuming and laborious. A robust, automated test suite is a key element in achieving continuous delivery, essential for companies in fast-paced markets.

Step 2 — Use a Version Control System

All of the files and processing steps that turn raw data into useful information are *source code*. All of the artifacts that data scientists create during ML development are just source code. These files control the entire data-analytics pipeline from end to end in an automated and reproducible fashion. When these file artifacts are viewed as code, they can be managed liked code.

In so many cases, the files associated with analytics are distributed in various places within an organization without any governing control. A revision control tool, such as Git, helps to store and manage all of the changes to code. It also keeps code organized, in a known repository and provides for disaster recovery. Revision control also helps software teams parallelize their efforts by allowing them to *branch and merge*.

Step 3 — Branch and Merge

When an analytics professional wants to make updates, he or she checks a copy of all of the relevant code out of the revision control system. He or she then can make changes to a local, private copy of the code. These local changes are called a branch. Revision control systems boost team productivity by allowing many developers to work on branches concurrently. When changes to the branch are complete, tested and known to be working, the code can
be checked back into revision control, thus merging back into the trunk or main code base.

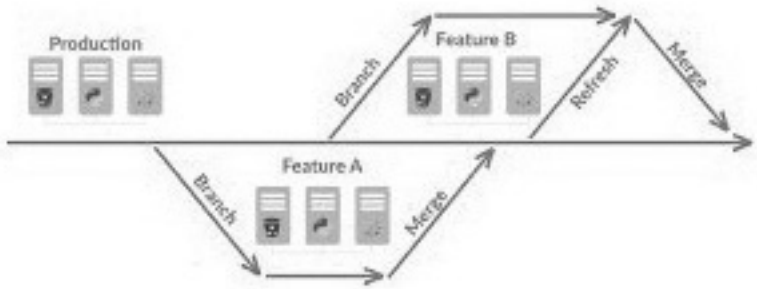


**Figure 67: With "Branch and Merge" team members can work on features
independently without impacting each other or value pipeline.**

Branching and merging allow the data science team to run their own tests, make changes, take risks and experiment. If a set of changes proves to be unfruitful, the branch can be discarded, and the team member can start over.

Step 4 — Use Multiple Environments

Infrastructure-as-a-service (IaaS) (or alternatively, platform-as-a-service or virtualization) has evolved to the point where new virtual machines, operating systems, stacks, applications and copies of data can be provisioned, with a click or command, under software control. DataOps calls for development and test environments that are separate from operations. The last thing that anyone would want is for a data scientist making changes to crash enterprise-critical analytics. When the team creating new analytics is given their own environments, they can iterate quickly without worrying about impacting operations. IaaS makes it easy to set-up development and test system environments that exactly match a target operations environment. This helps prevent finger-pointing between development, quality assurance and operations/IT.

It's worth reemphasizing that data scientists need a copy of the data. In the past, creating copies of databases was expensive. With storage on-demand from cloud services, a Terabyte data set can be quickly and inexpensively copied to reduce conflicts and dependencies. If the data is still too large to copy, it can be sampled.

Step 5 — Reuse & Containerize

Data science team members typically have a difficult time leveraging each other's work. Code reuse is a vast topic, but the basic idea is to componentize functionalities in ways that can be shared. Complex functions, with lots of individual parts, can be containerized using a container technology like Docker and Kubernetes. Containers are ideal for highly customized functions that require a skill set that isn't widely shared among the team.

Step 6 — Parameterize Your Processing

The data analytics pipeline should be designed with run-time flexibility. Which dataset should be used? Is a new data warehouse used for production or testing? Should data be filtered? Should specific workflow steps be included or not? These types of conditions are coded in different phases of the data analytics pipeline using parameters. In software development, a parameter is some information (e.g., a name, a number, an option) that is passed to a program that affects the way that it operates. With the right parameters in place, accommodating the day-to-day needs of the users and data science professionals becomes a routine matter.

Step 7 — Orchestrate Two Journeys

Many data science professionals dread the prospect of deploying changes that break production systems or allowing poor quality data to reach users. Addressing this requires optimization of both the Value and Innovation Pipelines. In the Value Pipeline (production), data flows into production and creates value for the organization. In the Innovation Pipeline, ideas, in the form of new analytics and AI, undergo development and are added to the Value Pipeline. The two pipelines intersect as new analytics deploy into operations. The DataOps enterprise masters the orchestration of data to production and the deployment of new features both while maintaining impeccable quality. With tests (statistical process control) controlling and monitoring both the data and new development pipelines, the dev team can deploy without worrying about breaking the production systems. With Agile Development and DevOps, the velocity of new analytics is maximized.

Figure 68 below shows how the seven steps of DataOps tie directly into the steps for model development shown in Figure 63. For example, orchestration automates data preparation, feature extraction, model training and model deployment. Version control, branch and merge, environments, reuse & containers and parameterization all apply to these same phases. Tests apply to all of the phases of the model life cycle.

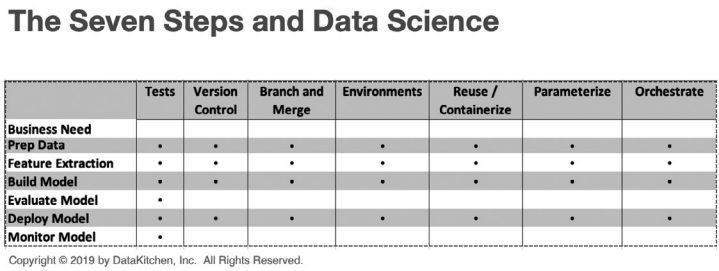


Figure 68: How the 7 steps of DataOps related to model development

CONCLUSION

While AI and data science tools improve the productivity of model development, the actual ML code is a small part of the overall system solution. Data science teams that don't apply modern software development principles to the data lifecycle can end up with poor quality and technical debt that causes unplanned work.

DataOps offers a new approach to creating and operationalizing AI that minimizes technical debt, reduces cycle time and improves code and data quality. It is a methodology that enables data science teams to thrive despite increasing levels of complexity required to deploy and maintain AI in the field. It decouples operations from new analytics creation and rejoins them under the automated framework of continuous delivery. The orchestration of the development, deployment, operations and monitoring toolchains dramatically simplifies the daily workflows of the data science team. Without the burden of technical debt and unplanned work, they can focus on their area of expertise; creating new models that help the enterprise realize its mission.

What Data Scientists Really Need

Kurt Cagle's perceptive analysis of data science titled "Why You Don't Need Data Scientists" explains the many reasons that data science falls short of the high expectations usually placed on it:

• Fancy dashboards are pretty but are only as valuable as the data behind them. Data quality often….stinks.

• Data sets are quirky and difficult to work with.

• Users/stakeholders know their business domain but little about what data can do for them.

• A multimillion-dollar initiative to rebuild the data pipeline from the ground up is generally *off the table*.

• The people who own the databases won't give data scientists access.

• Everyone agrees that integrating data from disparate databases is really, really hard, but in reality, it's much harder than people think.

These are all excellent points and often the conversation ends here — in exasperation. We can tell you that we have been there and have the PTSD to prove it. Fortunately, a few years ago, we found a way out of what may seem at times like a no-win situation. We believe that the secret to successful data science is a little about *tools* and a lot about *people and processes*.

DON'T BOIL THE OCEAN

Use Agile methods to create new analytics. Leverage infrastructure that enables teams to work together in an Agile way. Start small and simple. Create something quickly that adds value. Get feedback from your stakeholders. Repeat iteratively.

TESTS ARE BEST

Implement automated process controls that monitor data at every stage of your data analyt- ics pipeline. Think of your data analytics as a lean manufacturing pipeline where the quality of data cannot drift outside statistical and logical bounds. Let your tools work 24x7 so data scientists can stay focused on creating analytics that add value.

AUTOMATE AND ORCHESTRATE

Data Scientists spend 75% of their time doing data engineering. It's about time that data professionals took a page from DevOps. Automate workflow and the deployment of new analytics. Orchestrate the end-to-end data pipeline so we stop sucking the life out of data scientists. A single data engineer should be able to support ten data analysts and scientists, who in turn should be supporting 100 business professionals. An automated pipeline can get you there.

DataOps Healthy Hearty Banana Oat- meal Bread

by Andrew Sadoway

INGREDIENTS AND TOOLS

- 1 1/4 cups whole wheat flour
- 1 1/4 old-fashioned oats
- 1 teaspoon baking powder
- 1/2 teaspoon baking soda
- 1/4 teaspoon salt
- 1 teaspoon cinnamon
- Dash of nutmeg (approx. 1/8 teaspoon)
- 3 medium-*largeish* bananas, mashed (defrosted from freezer OK) • 1/2 cup low fat plain yogurt + 1 teaspoon vanilla
- 2 tablespoons honey, 1/3 cup brown sugar
- 1 large egg
- Splash of milk, as needed
- 3/4 cup dried cranberries
- 3/4 cup chopped walnuts + 1/4 cup chopped walnuts

INSTRUCTIONS

Preheat oven to 350. Combine dry ingredients (flour through nutmeg) in a small bowl. In a separate bowl, mix together yogurt, vanilla, brown sugar and honey. Add egg. Add mashed up bananas. Slowly fold dry ingredients into wet. Stir in cran- berries and 3/4 cup walnuts gently. Pour mixture into buttered loaf pan. Sprinkle remaining walnuts on top of loaf. Bake about 45 minutes, or until lightly browned and knife comes out clean.

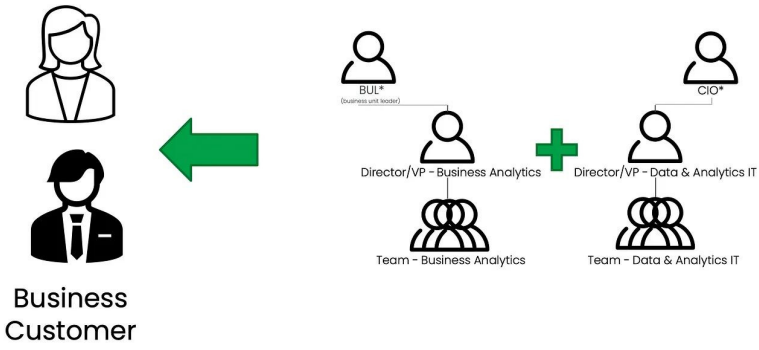
DATAOPS FOR DATA ANALYSTS

DataOps For Business Analytics Teams

Business analysts often find themselves in a no-win situation with constraints imposed from all sides. Their business unit colleagues ask an endless stream of urgent questions that require analytic insights. Business analysts must rapidly deliver value and simultaneously manage fragile and error-prone analytics production pipelines. Data tables from IT and other data sources require a large amount of repetitive, manual work to be used in analytics. In business analytics, fire-fighting and stress are common. Instead of throwing people and budgets at problems, DataOps offers a way to utilize automation to systematize analytics workflows. DataOps consolidates processes and workflows into a process hub that curates and manages the workflows that drive the creation of analytics. A DataOps process hub offers a way for business analytics teams to cope with fast-paced requirements without expanding staff or sacrificing quality.

Analytics Hub and Spoke

The data analytics function in large enterprises is generally distributed across departments and roles . Teams under the CDO and CAO are sometimes separate from the CIO. For example, teams working under the VP/Directors of Data Analytics may be tasked with accessing data, building databases, integrating data, and producing reports. Data scientists derive insights from data while business analysts work closely with and tend to the data needs of business units. Business analysts sometimes perform data science, but usually, they integrate and visualize data and create reports and dashboards from data supplied by other groups. Figure 1 below shows how contributions from two teams — an IT team on the right and a business analytics (BA) team on the left — must combine to create the analytics that serve business unit users. Some organizations use the metaphor of a “hub and spoke.” A centralized data or IT team maintains data platforms utilized by self-service users spread across the organization. The effort of both of these teams has to combine seamlessly to serve business customers.



If you work in the data industry, you've likely encountered this pattern of self-service , business unit teams and centralized data and analytic IT teams. In this post, we will focus on the business analytic teams, working directly for lines of business, and their unique challenges. The business analyst's goal is to create original insight for their customer, but they spend far too much time engaging in repetitive manual tasks. The business analyst is embedded in the business unit. They are collocated with business customers and identify with their objectives and challenges. The business analyst is on the front lines of the business unit's data usage. If anything goes wrong in data analytics, the business analyst is the first to know. They see the data errors , the production errors, the broken reports and the inaccurate dashboards. As a result, they deal with stress, firefighting and first hand.

Business Analytic Challenges

Business analytic teams have ongoing deliverables — a dashboard, a PowerPoint, or a model that they refresh and renew. Business analysts live under constant pressure to deliver. Business analysts must quickly update their analytics production deliverables as business conditions change. Customers and market forces drive deadlines and timeframes for analytics deliverables regardless of the level of effort required. It may take six weeks to add a new schema, but the VP may say she needs it for this Friday's strategy summit. If the IT or data engineering team can't respond with an enabling data platform in the required time frame, the business analyst does the necessary data work themselves. This ad hoc data engineering work often means coping with numerous data tables and diverse data sets using Alteryx, SQL, Excel or similar tools.

BA teams often do not receive a data platform tailored to their needs. They are given a bunch of raw tables, flat files and other data. Out of perhaps hundreds of tables, they have to pull together the ten tables that matter to a particular question. Tools like Alteryx help a little, but business analysts still perform an enormous amount of manual work. When the project scope and schedule are fixed, managers have to add more staff to keep up with the workload.

Sometimes BA teams turn to IT, which may have its drawbacks. Some IT teams are fantastic. They focus on making their customer, the business analyst teams, successful. Another kind of IT team is " *less helpful."* These IT partners are unfortunately all-too-common in data-driven organizations. They prepare artifacts and throw them over the wall to the business analytics team. They are less oriented toward delivering customer value and more focused on servicing their internal process or internal software development lifecycle. These groups tend to be more insulated from their customer's actual needs, so they take many months to deliver, and when a project does get done, it usually requires rework. A business analyst team receiving deliverables from an IT organization like that doesn't what they receive and ends up having to QA it thoroughly, which also involves a lot of effort.

We've talked to business analytic teams extensively about their challenges during our years working in the data analytics industry. Business analyst teams want to serve their internal customers better and faster but often live with a perpetual feeling of failure. An analyst may work hard to bring original insights to their customer, but it's common for a business colleague to react to analytics with a dozen follow-up questions. A successful analytics deliverable just generates more questions and more work. Imagine working in a field where a " *job well done* " generates an exponential amount of additional work. It's hard to feel good about yourself when you leave a meeting with a list of action items, having shown off a new deliverable. Many business analysts feel like failures.

Many business analytic teams feel that the responsibility for delivering insights to the business rests on their shoulders and they are not adequately supported by IT, the gatekeepers of data. Analysts feel like IT is cutting them off at the knees. As a result, analytic teams are always behind the curve and very reactive with business customers. Business analytic teams field an endless stream of questions from marketing and salespeople and they can't get ahead. They need high-quality data in an answer-ready format to address many scenarios with minimal keyboarding. What they are getting from IT and other data sources is, in reality, poor-quality data in a format that requires manual customization. They're spending 80% of their time performing tasks that are necessary but far removed from the insight generation and innovation that their business customers need.

DataOps Process Hub

DataOps provides a methodology and a paradigm to understand and address these challenges. There's a recent trend toward people creating data lake or data warehouse patterns and calling it data enablement or a . DataOps expands upon this approach by focusing on the processes and workflows that create data enablement and business analytics. is effectively a " *process hub* " that creates, manages and monitors the data hub. The BA team needs to take thousands of files or tables and synthesize them into the ten tables that matter the most for a particular business question. How do you get that to happen as quickly and robustly as possible? How do you enable the analytics team to update existing analytics with a schema change or additional data in a day or an hour instead of weeks or months?

The key to enabling rapid, iterative development is through a process hub , which complements and automates the data hub. A process hub, in essence, automates every possible workflow related to the data hub. A robust process hub is expressly designed to enable the data team to minimize the time spent maintaining working analytics and improve and update existing analytics with the least possible effort. Many large enterprises allow consultants and employees to keep tribal knowledge about the data architecture in their heads. The process hub captures all informal workflows as code executed on demand. that verify and validate data flowing through the data pipelines are executed continuously. An impact review test suite executes before new analytics are deployed. Production analytics are revision controlled , and new analytics are tested and deployed seamlessly. When people rotate in and out of roles in the data organization, all of the intellectual property related to business analytics is retained (in Domino models, SQL, reports, tests, sc

FEATURE	BENEFIT
Data designed for **Business Analytics**	• Significantly **faster time-to-insight** for ad hoc requests
Automating (& reusing) processes with a button push	• Significantly fewer dollars spent on technical mechanics & re-work • Reallocate Business Analytics consultants to more valuable tasks
Enablement for **Business Analytics, NOT IT infrastructure**	• Extend Data Lake/Warehouse investment, not replace

Table 1: Process hub features and benefits

We can summarize the benefits of the process hub as follows:

- Significantly faster time-to-insight for ad hoc requests — The business analytics team gets more immediate insight because the process hub provides them with a data representation that they both control and can easily modify for ad hoc requests.
- Less rework and maintenance — Automation enables the team to reallocate time previously spent on technical mechanics and rework to more valuable tasks, such as responding to customer inquiries.
- Enablement for business analytics — The focus on the process hub is to extend, not replace a data hub (i.e., IT-created infrastructure such as a data lake/warehouse). The process hub tailors the generic data hub to the custom, evolving needs of the business analytics team.

The process hub effectively frees your organization's team of highly trained analysts from performing repetitive, manual tasks. The process hub changes the operating model of how business analytics teams work. Sometimes it helps to speak in concrete terms, so let's briefly discuss the roles required on a business analytics team that uses a process hub.

Process Hub Roles and Responsibilities

Business Analyst / Data Scientist — Whatever the title, this person works with data to create innovative insights for customers in the business unit. This person is chartered with presenting data in a way that makes sense to their non-technical, non-analytic business colleagues. The work product could be a chart, graph, model or dashboard. This role typically already exists, but the process hub enables this person to stay productively focused on the value-added aspects of working with data. Two other roles make the business analyst successful.

Analytics Engineer — This "data doer" bridges the gap between what IT delivers and what business analytics requires. They take the data in an IT data hub and build data representations that business analysts need to do their work. If the Analytics Engineer performs this task manually, productivity has only advanced incrementally. However, with the help of a process hub and the DataOps Engineer, the work of the Analytics Engineer spurs an order of magnitude improvement in business analyst productivity.

DataOps Engineer — Knowing that the business analyst will always have follow-up questions and keep iterating on analytics, we need data representations that analysts can quickly and robustly modify. The DataOps Engineer creates the technical environment, including automated workflows and testing, so that the Analytics Engineer and Business Analyst stay focused on their customers. The DataOps Engineer creates and manages the DataOps process hub that makes the invisible . Processes and workflows that used to be stored in people's heads are instantiated as code. DataOps Engineering Is also about collaboration through shared abstraction. In practice, this means taking valuable 'nuggets' of code that everyone creates (i.e., ETL, SQL, Python, XML, tableau workbooks, etc.) and integrating them into a shared system: inserting nuggets into pipelines, creating tests, running the data factory, automating deployments, working across people/teams, measuring success and enabling self-service.

The DataOps process hub is a shared abstraction that enables scalable, fast, iterative work to develop. Another way to think about it is simply as automation of end-to-end lifecycle processes and workflows: production orchestration, production data monitoring/testing, self-service environments, development regression and functional tests, test data automation, deployment automation, shared components, and process measurement.

Stop Firefighting

The DataOps process hub helps make workflows and processes visible and tangible. An analytics engineer, a visualization analyst, a data scientist all have a place where they see their work come together. The process hub helps them resolve issues, automate processes and improve workflows. DataOps is fundamentally about automation, and analytics engineering focuses on creating valuable datasets that make analysts successful. A process hub helps the team stop moving from one emergency to the next. A lot of business analytic teams are constantly firefighting. DataOps is about automating tasks, so you aren't repeating manual processes. DataOps also focuses on abstracting and generalizing operations, so you aren't managing multiple versions of the same code or artifacts.

DataOps Engineering is about designing the system that creates analytics more robustly and efficiently. DataOps advances these aim by:

- Easing the path to production and ensuring scalability.
- Streamlining processes for automation, efficiency and reuse.
- Moving assumptions, business rules and code upstream.
- Improving reliability, consistency and efficiency
- Building out a flexible framework.
- Curating and managing production processes.
- Building out a production toolkit.
- Creating shortcuts for future tasks.
- Collaborating instead of blindly adhering to policies and procedures

It's prudent to assume that you will never completely avoid data emergencies. DataOps scales your processes to make the next urgent deadline easier to meet.

Managing a DataOps Team

A lot of industries talk about bringing data together in data hubs. DataOps adds a process hub to consolidate data workflows together. The process hub augments the data hub to centralize the people that act upon data into one place. Managing and curating the processes acting upon data is as important as the data itself. The process hub enables process scalability, productivity and project success. From a macro perspective, the process hub enables the business analytic teams to gain leverage to be more successful.

From the manager's unique perspective, the process hub helps team and organization leaders manage data development and operations in a disciplined, consistent and repeatable manner. Are processes working? Are business customers happy with what the team is producing? Is the team fully productive? Can people move seamlessly between teams and projects? Can decentralized staff, consultants and vendors work together effectively? How can the teams gain consistency? How can the teams ensure collaboration? How does the team take a long-term view and not create technical debt? How do the teams curate reusable components (or shortcuts) that span projects? How can the organization simplify, abstract and generalize data patterns? The process hub helps answer these questions by collecting metrics on all the processes instantiated as code.

Managers are frequently asked to accomplish " " deliverables with a " " budget. This zero-sum dilemma comes down to increasing productivity so that a business analytics team can accomplish more with the same level of resources. The only reasonable response to this challenge is to focus on automation, quality and process efficiency. Put your processes in one place. Improve them. Curate, manage and automate them. We've seen organizations attain incredible productivity gains using a process hub — on the order of *ten times more productivity* . These impressive gains stem from spending more time focused on customer requirements and working in a more Agile way.

Conclusion

DataOps is fundamentally a philosophy and methodology of “ *business-processes-as-code* .” A process hub keeps you from continuously throwing additional staff at problems. It utilizes automation to make BA teams more agile and responsive to the rapid-fire requests from business units. With a DataOps process hub, managers of BA teams can oversee a “ *well-oiled machine* “ of analysts iterating efficiently on analytics to address evolving business challenges.

Eight Challenges of Data Analytics

Companies increasingly look to analytics to drive growth strategies. As the leader of the data-analytics team, you manage a group responsible for supplying business partners with the analytic insights that can create a competitive edge. Customer and market opportunities evolve quickly and drive a relentless series of questions. Analytics, by contrast, move slowly, constrained by development cycles, limited resources and brittle IT systems. The gap be- tween what users need and what IT can provide can be a source of conflict and frustration. Inevitably this mismatch between expectations and capabilities can cause dissatisfaction, leaving the data-analytics team in an unfortunate position and preventing a company from fully realizing the strategic benefit of its data.

As a manager overseeing analytics, it's your job to understand and address the factors that prevent the data-analytics team from achieving peak levels of performance. If you talk to your team, they will tell you exactly what is slowing them down. You'll likely hear variations of the following eight challenges:

1 – The Goalposts Keep Moving

Users are demanding customers for a data-analytics team. Their requirements change constantly. They require immediate responses, and no matter how much the analytics team delivers, users keep generating new requests. It's enough to overwhelm any data-analytics team.

They don't know what they want. Users are not data experts. They don't know what insights are possible until someone from your team shows them. Sometimes they don't know what they want until after they see it in production (and maybe not even then). Often, business stakeholders do not know what they will need next week, let alone next quarter or next year. It's not their fault. It's the nature of pursuing opportunities in a fast-paced marketplace.

They need everything ASAP. Business is a competitive endeavor. When an opportunity opens, the company needs to move on it faster than the competition. When users bring a question to the data-analytics team, they expect an immediate response. They can't wait weeks or months — the opportunity will close as the market seeks alternative solutions.

The questions never end. Sometimes providing business stakeholders with analytics generates more questions than answers. Analytic insights enable users to understand the business in new ways. This spurs creativity, which leads to requests for more analytics. A healthy relationship between the analytics and users will foster a continuous series of questions that drive demand for new analytics. However, this relationship can sour quickly if the delivery of new analytics can't meet the required time frames.

2 – Data Lives in Silos

In pursuit of business objectives, companies collect an enormous amount of data: orders, deliveries, returns, website page views, mobile app navigations, downloads, clicks, metrics, audio logs, social media and more. Further, this data can be combined with demographic,

psychographic or other third-party market data. All of this data is collected in separate databases which typically do not talk to each other. They utilize numerous platforms, APIs and technologies. Accessing all of this data is a daunting task requiring such a wide range of skills that it is rare to find a single person who can do it all. Integrating data from these myriad sources becomes a major undertaking.

Business stakeholders want fast answers. Meanwhile, the data-analytics team has to work with IT to gain access to operational systems, plan and implement architectural changes, and develop/test/deploy new analytics. This process is complex, lengthy and subject to numerous bottlenecks and blockages.

3 – Data Formats are not Optimized

Data in operational systems is usually *not* structured in a way that lends itself to the efficient creation of analytics. For example, an ERP system might have a schema that is optimized for inserts, updates, and for display in a web user interface. For operational systems, these are the actions that need to happen in real time.

A database optimized for data analytics is structured to optimize reads and aggregations. It's also important for the schema of an analytics database to be easily understood by humans. For example, the field names would be descriptive of their contents and data tables would be linked in ways that make intuitive sense.

4 – Data Errors

Whether your data sources are internal or from external third parties, data will eventually contain errors. Data errors can prevent your data pipeline from flowing correctly. Errors may also be subtle, such as duplicate records or individual fields that contain erroneous data. Data errors could be caused by a new algorithm that doesn't work as expected, a database schema change that broke one of your feeds, an IT failure or one of many other possibilities. Data errors can be difficult to trace and resolve quickly.

5 – Bad Data Ruins Good Reports

When data errors work their way through the data pipeline into published analytics, internal stakeholders can become dissatisfied. This causes unplanned work, which diverts your key contributors from the highest priority projects. Bad data also harms the hard-won credibil-
ity of the data-analytics team. If business colleagues repeatedly see bad data in analytics reports, they might learn not to trust or value the work product of the data-analytics team.

6 – Data Pipeline Maintenance Never Ends

Data-analytics is a pipeline process that executes a set of operations and attempts to produce a consistent output at a high level of quality. Every new or updated data source, schema enhancement, analytics improvement or other change triggers an update to the pipeline. The data-analytics team is continuously making changes and improvements to the data pipeline. Each one of these changes must be made carefully so that it doesn't break operational analytics. The effort required to validate and verify changes often takes longer than the time required to create the changes in the first place. You may not realize it, but your analysts, data scientists and engineers may be spending 80% of their time updating, maintaining and assuring the quality of the data pipeline. This is necessary work, but much of it is behind the scenes and unappreciated when viewed against the growing backlog of new requests from business customers.

7 – Manual Process Fatigue

Data integration, cleansing, transformation, quality assurance and deployment of new analytics must be performed flawlessly day in and day out. The data-analytics team may have automated a portion of these tasks, but some teams perform numerous manual processes on a regular basis. These rote procedures are error prone, time consuming and tedious.

Further, manual processes can also lead to high employee turnover. Many managers have watched high-performing data-analytics team members burn out due to having to repeatedly execute manual data procedures. Manual processes strain the productivity of the data team in numerous ways.

8 – The Trap of "Hope and Heroism"

The landscape is littered with projects and initiatives cancelled or deferred due to changing requirements, slipped schedules, disappointed users, inflexibility, poor quality, low ROI, and irrelevant features.

According to the research firm Gartner, Inc., half of all chief data officers (CDO) in large organizations will not be deemed a *success* in their role. Per Forrester Research, 60% of the data and analytics decision-makers surveyed said they *are not very confident* in their analytics insights. Only ten percent responded that their organizations sufficiently *manage the quality of data and analytics.* Just sixteen percent believe they perform well in producing *accurate models.*

Many CDO's and data-analytics professionals respond to these challenges in one of three ways:

Heroism - Data-analytics teams work long hours to compensate for the gap between performance and expectations. When a deliverable is met, the data-analytics team is considered heroes. However, yesterday's heroes are quickly forgotten when there is a new deliverable to meet. Also, this strategy is difficult to sustain over a long period of time, and it, ultimately, just resets expectations at a higher level without providing additional resources. The heroism approach is also difficult to scale up as an organization grows.

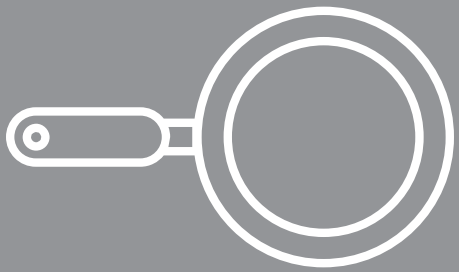
Hope - When a deadline must be met, it is tempting to just quickly produce a solution with minimal testing, push it out to the users and *hope* it does not break. This approach has inherent risks. Eventually, a deliverable will contain data errors, upsetting the users and harming the hard-won credibility of the data-analytics team.

Caution - The team decides to give each data-analytics project a longer development and test schedule. Effectively, this is a decision to deliver higher quality, but fewer features to users. One difficulty with this approach is that users often don't know what they want until they see it, so a detailed specification might change considerably by the end of a project. The slow and methodical approach might also make the users unhappy because the analytics are delivered more slowly than their stated delivery requirements and as requests pile up, the data-analytics team risks being viewed as bureaucratic and inefficient.

None of these approaches adequately serve the needs of both users and data-analytics professionals, but there is a way out of this bind. The challenges above are not unique to analytics, and in fact, are shared by other organizations.

Overcoming the Challenges

Some say that an analytics team can overcome these challenges by buying a new tool. While it is true that new tools are helpful, they are not enough by themselves. You cannot truly transform your staff into a high-performance team without an overhaul of the methodologies and processes that guide your workflows. In this book, we will discuss how to combine tools and new processes in a way that improves the productivity of your data analytics team by orders of magnitude.



DataOps No-Knead Bread

by Gil Benghiat

INGREDIENTS AND TOOLS

- 3 cups bread flour
- 1/4 teaspoons instant yeast
- ½ teaspoon salt
- 1 ½ cups very warm, almost hot water
- Oil for work surface (canola)

INSTRUCTIONS

1. Combine flour, yeast and salt in a large bowl and stir with your DataKitchen spoon. Add water and stir until blended; dough will be shaggy. You may need an extra ¼ cup of water to get all the flour to blend in. Cover bowl with plastic wrap. Let dough rest at least 4 hours (12-18 hours is good too) at warm room temperature, about 70 degrees.
2. Lightly oil a work surface and place dough on it; fold it over on itself once or twice. Cover loosely with plastic wrap and let rest 30 minutes more. This is a good time to turn the oven on to 425°F.
3. Put a 6-to-8-quart heavy covered pot (cast iron, enamel, Pyrex or ceramic) in the oven as it heats. When dough is ready, carefully remove pot from oven. Slide your hand under dough and put it into pot, seam side up. Shake pan once or twice if dough is unevenly distributed; it will straighten out as it bakes.
4. Cover with lid and bake 30 minutes, then remove lid and bake another 15 to 30 minutes, until loaf is beautifully browned. Cool on a rack.

NOTES

In a convection oven, cook 23 minutes with the lid on, and then 5 minutes with the lid off.

You don't need to pre-heat the pot. You can put the dough on a cookie sheet. The only difference is the crust will not be as crunchy or as beautifully browned. You can experiment with a round shape or Italian or French loaf shapes. The longer shapes will take less time to cook.

You can also cook at a lower temperature (e.g. 350°F). In all cases, take the bread out when the internal temperature reaches 190°F - 200°F. Use a meat thermometer to check.

Derived from New York Times Speedy No-Knead Bread

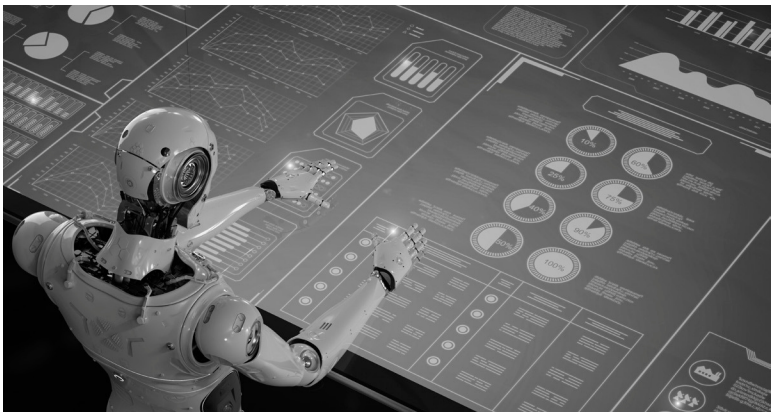
DataOps for Data Quality

Disband Your Impact Review Board: Automate Analytics Testing

Some companies take six months to write 20 lines of SQL and move it into production.

The last thing that an analytics professional wants to do is introduce a change that breaks the system. Nobody wants to be the object of scorn, the butt of jokes, or a cautionary tale. If that 20-line SQL change is misapplied, it can be a “career-limiting move” for an analytics professional.

Analytics systems grow so large and complex that no single person in the company understands them from end to end. A large company often institutes slow, bureaucratic procedures for introducing new analytics in order to reduce fear and uncertainty. They create a waterfall process with specific milestones. There is a lot of documentation, checks and balances, and meetings — lots of meetings.



IMPACT ANALYSIS

One of the bottlenecks in an analytics release process is called “impact analysis.” Impact analysis gathers experts on all of the various subsystems (data feeds, databases, transforms, data lakes/warehouses, tools, reports, …) so they can review the proverbial 20 lines of SQL and try to anticipate if/how it will adversely impact data operations.

Imagine you are building technical systems that integrate data and do models and visualizations. How does a change in one area affect other areas? In a traditional established company, that information is locked in various people’s heads. The company may think it has no choice but to gather these experts together in one room to discuss and analyze proposed changes. This is called an “impact analysis meeting.” The process includes the company’s most senior technical contributors; the backbone of data operations. Naturally, these individuals are extremely busy and subject to high-priority interruptions. Sometimes it takes weeks to gather them in one room. It can take additional weeks or months for them to approve a change.

The impact analysis team is a critical bottleneck that slows down updates to analytics. A DataOps approach to improving analytics cycle time adopts process optimization techniques from the manufacturing field. In a factory environment, a small number of bottlenecks often limit throughput. This is called the Theory of Constraints. Optimize the throughput of bottlenecks and your end-to-end cycle time improves (check out *“The Goal”* by Eliyahu M. Goldratt).

GET OUT OF YOUR HEAD

The Impact Analysis Meeting is a bottleneck because it relies upon your top technical experts — one of the most oversubscribed resources in the company. What if you could extract all the knowledge and experience trapped in the brains of your company’s experts and code
it into a series of tests that would perform the impact analysis for you? This would give you a quick way to test out changes to analytics without requiring bureaucratic procedures and meetings. If the tests pass, you could deploy with confidence. No more waiting on the impact review team. With a comprehensive test suite, you reduce reliance on the impact analysis bottleneck and move a lot faster.

AUTOMATING IMPACT ANALYSIS

Manual testing moves the bottleneck from impact review to the testing team. Manual testing is performed step-by-step, by a person. This tends to be expensive as it requires someone to create an environment and run tests one at a time. It can also be prone to human error. DataOps automates testing. Environments are spun up under machine control and test scripts, written in advance, are executed in batch. Automated testing is much more cost-effective and reliable than manual testing, but the effectiveness of automated testing depends on the quality and breadth of the tests. In a DataOps enterprise, members of the analytics team spend 20% of their time writing tests. Whenever a problem is encountered, a new test is added. New tests accompany every analytics update. The breadth and depth of the test suite continuously grow.

One advantage of automated testing is that it's easier to run so it's executed repeatedly and regularly. Manual testing is often too expensive and slow to run on a regular basis. To ensure high quality, you have to be able to consistently and regularly test your data and code.

These concepts are new to many data teams, but they are well established in the software industry. As figure 78 shows, the cycle time of software development releases has been (and continues to be) reduced by orders of magnitude through automation and process improvements. The automation of impact analysis can have a similar positive effect on your organization's analytics cycle time.

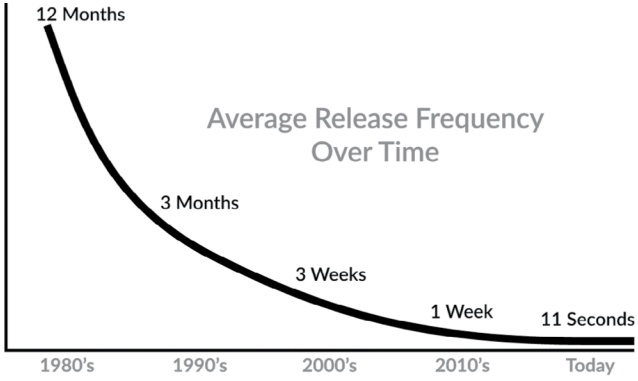


Figure 78: Software developers have reduced the cycle time for new releases by orders of magnitude using automation and process improvements

ANALYTICS IS CODE

At this point some of you are thinking this has nothing to do with me. I am a data analyst/sci- entist, not a coder. I am a tool expert. What I do is just a sophisticated form of configuration. This is a common point of view in data analytics. However, it leads to a mindset that slows down analytics cycle time.

Tools vendors have a business interest in perpetuating the myth that if you stay within the well-defined boundaries of their tool, you are protected from the complexity of software development. This is ill-considered.

Don't get us wrong. We love our tools, but don't buy into this falsehood.

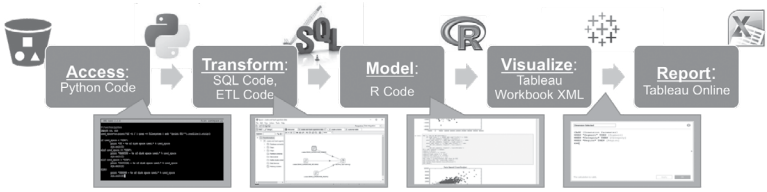


Figure 79: From data access to visualization to reports, there is code running at every stage of the data operations pipeline

The $100B analytics market is divided into two segments: tools that create code and tools that run code. The point is — data analytics is code. The data professional creates code and must own, embrace and manage the complexity that comes along with it.

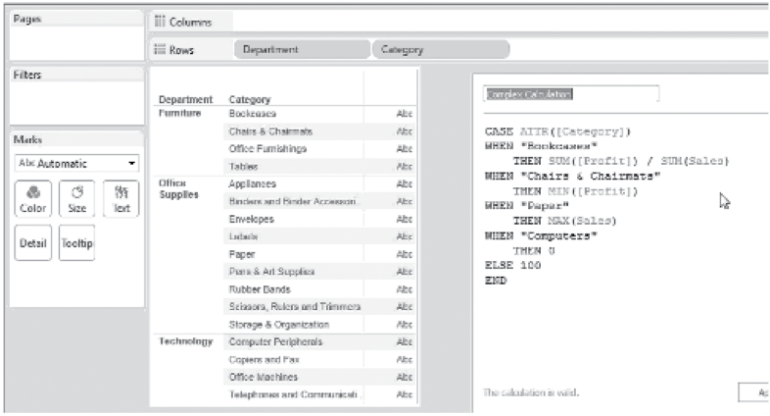


Figure 80: Tableau files are stored as XML, and can contain conditional branches, loops and embedded code.

Figure 79 shows a data operations pipeline with code at every stage of the pipeline. Python, SQL, R — these are all code. The tools of the trade (Informatica, Tableau, Excel, …) these too are code. If you open an Informatica or Tableau file, it's XML. It contains conditional branches (if-then-else constructs), loops and you can embed Python or R in it.

Remember our 20-line SQL change that took six months to implement? The problem is that analytics systems become so complex that they can easily break if someone makes one misbegotten change. The average data-analytics pipeline encompasses many tools (code generators) and runs lots of code. Between all of the code and people involved, data operations becomes a *combinatorially* complex hairball of systems that could come crashing down with one little mistake.

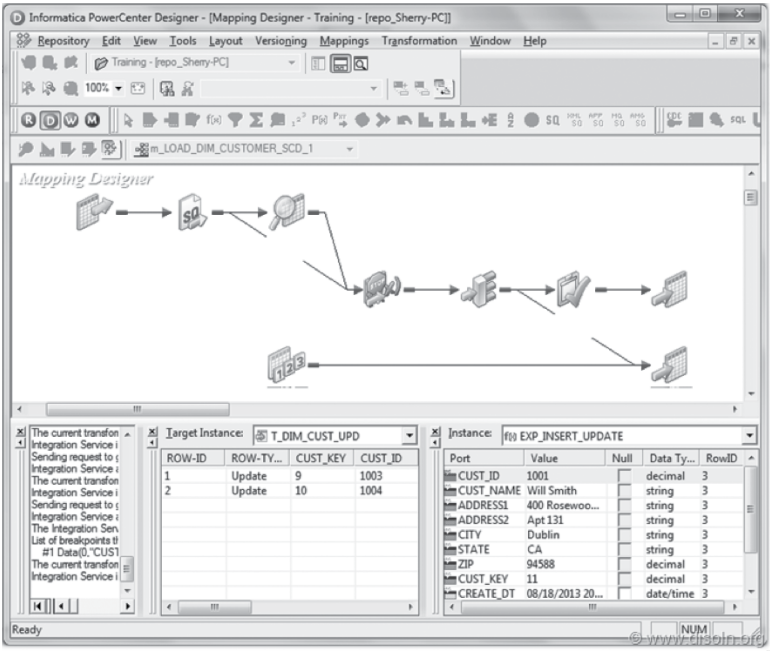


Figure 81: Informatica presents a UI that creates ETL in an XML format that is then converted to Java and executed on the machine.

For example, imagine that you have analytics that sorts customers into five bins based on some conditional criterion. Deep inside your tool's XML file is an if-then-else construct that is responsible for sorting the customers correctly. You have numerous reports based off of a template that contains this logic. They provide information to your business stakeholders: top customers, middle customers, gainers, decliners, whales, profitable customers,…

There's a team of IT engineers, database developers, data engineers, analysts and data scientists that manage the end to end system that supports these analytics. One of these individuals makes a change. They convert the sales volume field from an integer into a decimal. Perhaps they convert a field that was US dollars into a different currency. Maybe they rename a column. Everything in the analytics pipeline is so interdependent; the change breaks all of the reports that contain the if-then-else logic upon which the original five categories are built. All of a sudden, your five customer categories become one category, or the wrong customers are sorted into the wrong bins. None of the dependent analytics are correct, reports are showing incorrect data, and the VP of Sales is calling you hourly.

At an abstract level, every analytic insight produced, every deliverable, is an interconnected chain of code modules delivering value. The data analytics pipeline is best represented by a directed acyclic graph (DAG). For example, see figure 82.

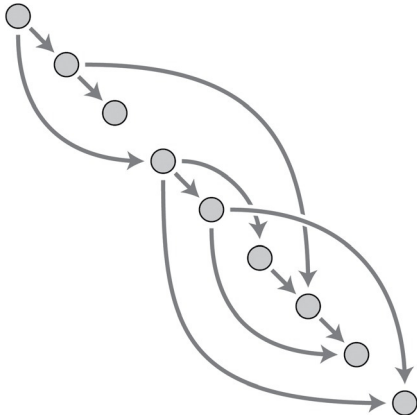


Figure 82: The Directed Acyclic Graph (DAG) models the steps in the data analytics pipeline

Whether you use an analytics tool like Informatica or Tableau, an Integrated Development Environment (IDE) like Microsoft Visual Studio (figure 83) or even a text editor like Notepad, you are creating code. The code that you create interacts with all of the other code that populates the DAG that represents your data pipeline.

To automate impact analysis, think of the end-to-end data pipeline holistically. Your test suite should verify software entities on a stand-alone basis as well as how they interact.

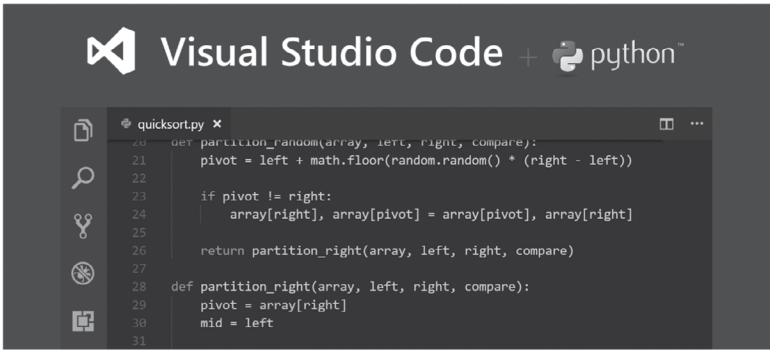


Figure 83: Developers write SQL, Python and other code using an integrated development environment or sometimes a simple editor like Notepad.

TYPES OF TESTS

The software industry has decades of experience ensuring that code behaves as expected. Each type of test has a specific goal. If you spend any time discussing testing with your peers, these terms are sure to come up:

• **Unit Tests** – testing aimed at each software component as a stand-alone entity

• **Integration Tests** – focus on the interaction between components to ensure that they are interoperating correctly

• **Functional Tests** – verification against functional specification or user stories.

• **Regression Tests** – rerun every time a change is made to prove that an application is still functioning

• **Performance Tests** – verify a system's responsiveness, stability and availability under a given workload

• **Smoke Tests** – quick, preliminary validation that the major system functions are operational

TESTS TARGET DATA OR CODE OR BOTH

It's also helpful to frame the purpose and context of a test. Tests can target data or code and run as part of the data operations pipeline. Location balance, historical balance and statistical process controls (time balance) tests are directed at the data flowing through an operations pipeline. The code that runs the data processing steps in the pipeline is fixed. The code is tightly controlled and only changed via a release process. Data that moves through operations, on the other hand, is variable. New data flows through the pipeline continuously. As figure 84 shows, the data operations pipeline delivers value to users. DataOps terms this the Value Pipeline.

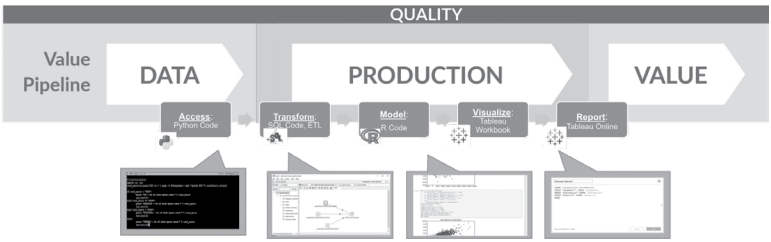


Figure 84: Data Operations: The Value Pipeline

The development of new analytics follows a different path, which is shown in figure 85 as the Innovation Pipeline. The Innovation Pipeline delivers new insights to the data operations pipeline, regulated by the release process. To safely develop new code, the analyst needs an isolated development environment. When creating new analytics, the developer creates an environment analogous to the overall system. If the database is terabytes in size, the data

professional might copy it for test purposes. If the data is petabytes in size, it may make sense to sample it; for example, take 10% of the overall data. If there are concerns about

	Data Fixed	Data Variable
Code Fixed		Value Pipeline
Code Variable	Innovation Pipeline	

Table 5: In the Value Pipeline code is fixed and data is variable. In the Innovation Pipeline, data is fixed, and code is variable.

privacy or other regulations, then sensitive information is removed. Once the environment is set up, the data typically remains stable.

In the Innovation Pipeline code is variable, but data is fixed. Tests target the code, not the data. The unit, integration, functional, performance and regression tests that were mentioned above are aimed at vetting new code. All tests are run before promoting (merging) new code to production. Code changes should be managed using a version control system, for example GIT. A good test suite serves as an automated form of impact analysis that can be run on any and every code change before deployment.

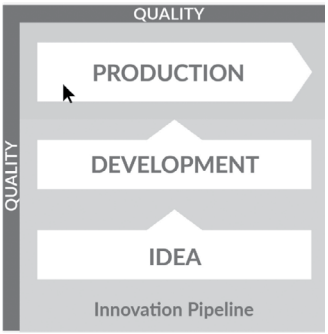


Figure 85: New analytics are developed in the Innovation Pipeline

Some tests are aimed at both data and code. For example, a test that makes sure that a database has the right number of rows helps your data and code work together. Ultimately both data tests and code tests need to come together in an integrated pipeline as shown in figure 86. DataOps enables code and data tests to work together so all around quality remains high.

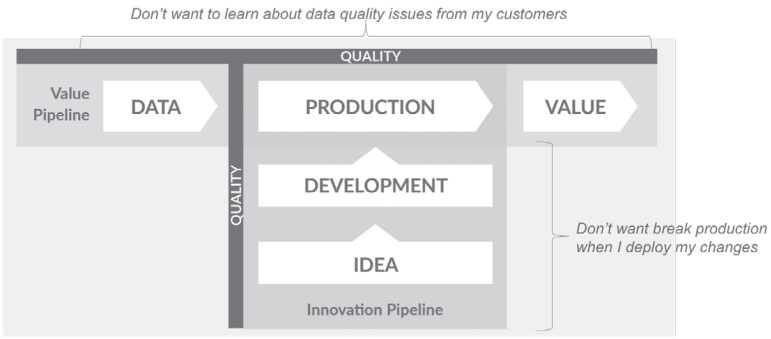


Figure 86: Ultimately the Value and Innovation Pipelines work together to maintain data and code quality

A unified, automated test suite that tests/monitors both production data and analytic code is the linchpin that makes DataOps work. Robust and thorough testing removes or minimizes the need to perform manual impact analysis, which avoids a bottleneck that slows innova- tion. Removing constraints helps speed innovation and improve quality by minimizing analyt- ics cycle time. With a highly optimized test process you'll be able to expedite new analytics into production with a high level of confidence.

20 new lines of SQL? You'll have it right away.

Build Trust Through Test Automation and Monitoring

"Trust takes years to build, seconds to break, and forever to repair."

We recently talked to a data team in a financial services company that lost the trust of their users. They lacked the resources to implement quality controls so bad data sometimes leaked into user analytics. After several high-profile episodes, department heads hired their own people to create reports. For a data-analytics team, this is the nightmare scenario, and it could have been avoided.

Organizations trust their data when they believe it is accurate. A data team can struggle to produce high-quality analytics when resources are limited, business logic keeps changing and data sources have *less-than-perfect* quality themselves. Accurate data analytics are the product of quality controls and sound processes.

The data team can't spend 100% of its time checking data, but if data analysts or scientists spend 10-20% of their time on quality, they can produce an automated testing and monitoring system that does the work for them. Automated testing can work 24x7 to ensure that bad data never reaches users, and when a mishap does occur, it helps to be able to assure users that new tests can be written to make certain that an error never happens again. Automated testing and monitoring greatly multiplies the effort that a data team invests in quality.

DATA FLOW AS A PIPELINE

Think of data analytics as a manufacturing pipeline. There are inputs (data sources), process- es (transformations) and outputs (analytics). A typical manufacturing process includes tests
at every step in the pipeline that attempt to identify problems as early as possible. As every manufacturer knows, it is much more efficient and less expensive to catch a problem in incoming inspection as opposed to finished goods.

Figure 87 depicts the data-analytics pipeline. In this diagram, databases are accessed and then data is transformed in preparation for being input into models. Models output visualizations and reports that provide critical information to users.

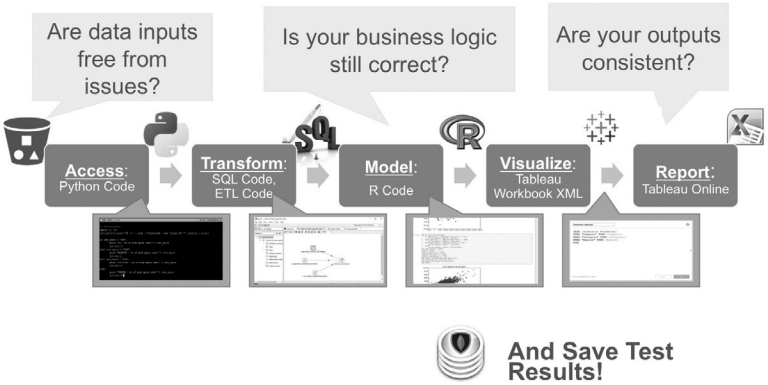


Figure 87: Testing each stage of the data-analytic pipeline

Along the way, tests ask important questions. Are data inputs free from issues? Is business logic correct? Are outputs consistent? As in lean manufacturing, tests are performed at every step in the pipeline. For example, data input tests are analogous to manufacturing incoming quality control. Figure 88 shows examples of data input, output and business logic tests.

Data input tests strive to prevent any bad data from being fed into subsequent pipeline stages. Allowing bad data to progress through the pipeline wastes processing resources and increases the risk of never catching an issue. It also focuses attention on the quality of data sources, which must be actively managed — manufacturers call this *supply chain management*. Data output tests verify that a pipeline stage executed correctly. Business logic tests validate data against tried and true assumptions about the business. For example, perhaps all European customers are assigned to a member of the Europe sales team.

Test results saved over time provide a way to check and monitor quality versus historical levels.

Inputs	**Verifying the inputs to an analytics processing stage** Count Verification - Check that row counts are in the right range, ... Conformity - US Zip5 codes are five digits, US phone numbers are 10 digits, ... History - The number of prospects always increases, ... Balance - Week over week, sales should not vary by more than 10%, ... Temporal Consistency - Transaction dates are in the past, end dates are later than start dates, ... Application Consistency - Body temperature is within a range around 98.6F/37C, ... Field Validation - All required fields are present, correctly entered, ...
Business Logic	**Checking that the data matches business assumptions** Customer Validation - Each customer should exist in a dimension table Data Validation - 90 percent of data should match entries in a dimension table
Output	**Checking the result of an operation, for example, a cross-product join** Completeness - Number of customer prospects should increase with time Range Verification - Number of physicians in the US is less than 1.5 million

Figure 88: Tests validate data inputs and outputs and verify that data is consistent with business logic.

FAILURE MODES

A disciplined data production process classifies failures according to severity level. Some errors are fatal and require the data analytics pipeline to be stopped. In a manufacturing setting, the most severe errors "stop the line."

Some test failures are warnings. They require further investigation by a member of the data analytics team. Was there a change in a data source? Or a redefinition that affects how data is reported? A warning gives the data-analytics team time to review the changes, talk to domain experts, and find the root cause of the anomaly.

Many test outputs will be informational. They help the data engineer, who oversees the pipeline, to monitor routine pipeline activity or investigate failures.

Severity	Required Action
Error	Stop the pipeline
Warning	Investigate the failure
Informational	Be aware of information

Table 6: Actions required for different failure modes

TYPES OF TESTS

The data team may sometimes feel that its work product is *under a microscope*. If the analytics look "off," users can often tell immediately. They are experts in their own domain and will often see problems in analytics with only a quick glance.

Finding issues before your internal customers do is critically important for the data team. There are three basic types of tests that will help you find issues before anyone else: location balance, historical balance and statistical process control.

LOCATION BALANCE TESTS

Location Balance tests ensure that data properties match business logic at each stage of processing. For example, an application may expect 1 million rows of data to arrive via FTP. The Location Balance test could verify that the correct quantity of data arrived initially, and that the same quantity is present in the database, in other stages of the pipeline and finally, in reports.

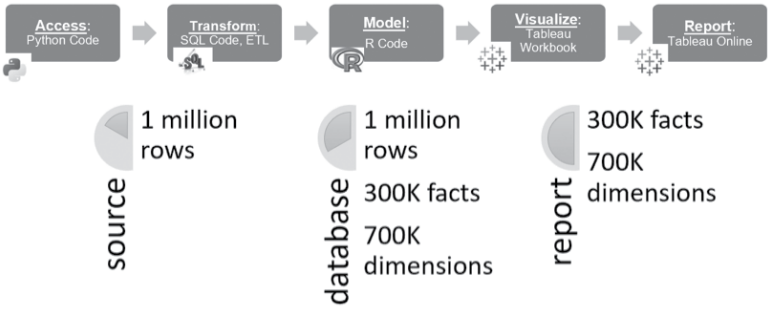


Figure 89: Location Balance Tests verify 1M rows in raw source data, and the corresponding 1M rows / 300K facts / 700K dimension members in the database schema, and 300K facts / 700K dimension members in a Tableau report

HISTORICAL BALANCE

Historical Balance tests compare current data to previous or expected values. These tests rely upon historical values as a reference to determine whether data values are reasonable (or within the range of reasonable). For example, a test can check the top fifty customers or suppliers. Did their values unexpectedly or unreasonably go up or down relative to historical values?

It's not enough for analytics to be correct. Accurate analytics that *"look wrong"* to users raise credibility questions. Figure 90 shows how a change in allocations of SKUs, moving from pre-production to production, affects the sales volumes for product groups G1 and G2. You can bet that the VP of sales will notice this change immediately and will report back that the analytics look *wrong*. This is a common issue for analytics — the report is correct, but it reflects poorly on the data team because it *looks wrong* to users. *What has changed?* When confronted, the data-analytics team has no ready explanation. *Guess who is in the hot seat.*

Historical Balance tests could have alerted the data team ahead of time that product group sales volumes had shifted unexpectedly. This would give the data-analytics team a chance to investigate and communicate the change to users in advance. Instead of hurting credibility, this episode could help build it by showing users that the reporting is under control and that the data team is on top of changes that affect analytics. *"Dear sales department, you may notice a change in the sales volumes for G1 and G2. This is driven by a reassignment of SKUs within the product groups."*

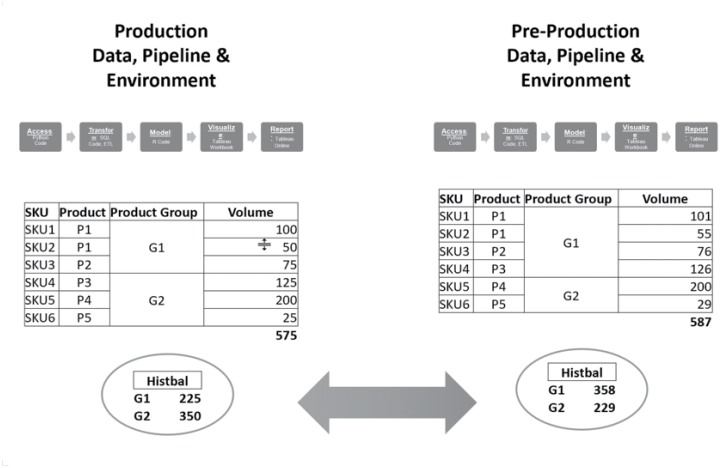


Figure 90: It's not enough for analytics to be correct. Accurate analytics that "look wrong" to users raise credibility questions.

STATISTICAL PROCESS CONTROL

Lean manufacturing operations measure and monitor every aspect of their process in order to detect issues as early as possible. These are called Time Balance tests or more commonly, statistical process control (SPC). SPC tests repeatedly measure an aspect of the data pipeline screening for error or warning patterns. SPC offers a critical tool for the data team to catch failures before users see them in reports.

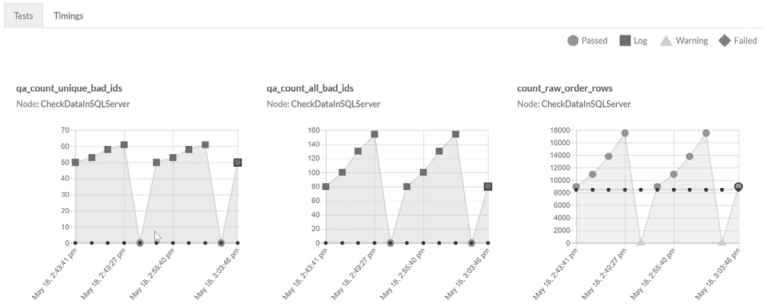
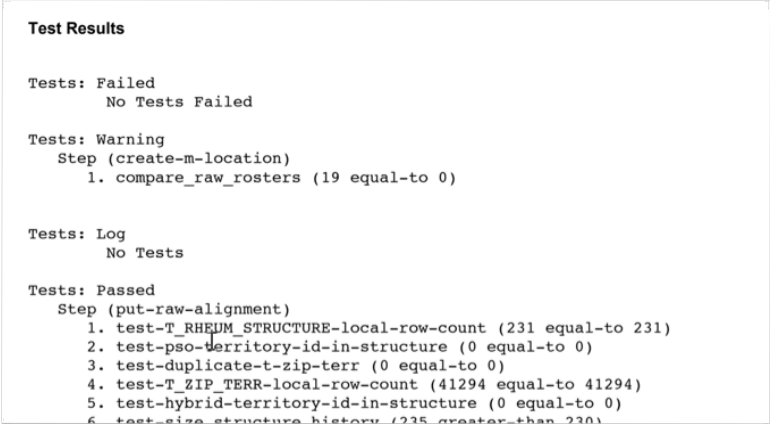


Figure 91: Statistical Process Control tests apply numerical criteria to data-analytics pipeline measurements

NOTIFICATIONS

A complex process could have thousands of tests running continuously. When an error or warning occurs, a person on the data team should be alerted in real-time through email, text or a notification service like slack. This frees the data team from the distraction of having to periodically poll test results. If and when an event takes place, they'll be notified and can take action.



Test Results

```
Tests: Failed
        No Tests Failed

Tests: Warning
   Step (create-m-location)
      1. compare_raw_rosters (19 equal-to 0)


Tests: Log
        No Tests

Tests: Passed
   Step (put-raw-alignment)
      1. test-T_RHEUM_STRUCTURE-local-row-count (231 equal-to 231)
      2. test-pso-territory-id-in-structure (0 equal-to 0)
      3. test-duplicate-t-zip-terr (0 equal-to 0)
      4. test-T_ZIP_TERR-local-row-count (41294 equal-to 41294)
      5. test-hybrid-territory-id-in-structure (0 equal-to 0)
      6. test-size-structure-history (235 greater-than 230)
```

Figure 92: Example data test result notification email

Automated tests and alerts enforce quality and greatly lessen the day-to-day burden of monitoring the pipeline. The organization's trust in data is built and maintained by producing consistent, high-quality analytics that help users understand their operational environment. That trust is critical to the success of an analytics initiative. After all, trust in the data is really trust in the data team.

How Data Analytics Professionals Can Sleep Better

LEAN MANUFACTURING SECRETS THAT YOU CAN APPLY TO DATA ANALYTICS

What could data analytics professionals possibly learn from car manufacturers? It turns out, a lot. Automotive giant Toyota pioneered a set of methods, later folded into a discipline called *lean manufacturing*, in which employees focus relentlessly on improving quality and reducing non-value-add activities. This culture enabled Toyota to grow into the one of the world's leading car companies. The Agile and DevOps methods that have led to stellar improvements in coding velocity are really just an example of lean manufacturing principles applied to software development.

Conceptually, manufacturing is a pipeline process. Raw materials enter the manufacturing floor through the stock room, flow to different work stations as work-in-progress and exit as finished goods. In data-analytics, data progresses through a series of steps and exits in the form of reports, models and visualizations. Each step takes an input from the previous step, executes a complex procedure or set of instructions and creates output for the subsequent step. At an abstract level, the data-analytics pipeline is analogous to a manufacturing process. Like manufacturing, data analytics executes a set of operations and attempts to produce a consistent output at a high level of quality. In addition to lean-manufacturing-inspired methods like Agile and DevOps, there is one more useful tool that can be taken from manufacturing and applied to data-analytics process improvement.

W. Edwards Deming championed statistical process control (SPC) as a method to improve manufacturing quality. SPC uses real-time product or process measurements to monitor and control quality during manufacturing processes. If the process measurements are maintained within specific limits, then the manufacturing process is deemed to be functioning properly. When SPC is applied to the data-analytics pipeline, it leads to remarkable improvements in efficiency and quality. For example, Google executes over one hundred million automated

test scripts per day to validate any new code released by software developers. In the Google consumer surveys group, code is deployed to customers eight minutes after a software engi- neer finishes writing and testing it.

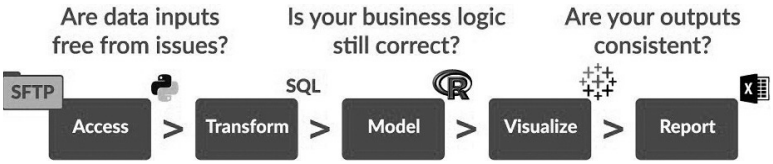


Figure 93: Tests verify the results for each intermediate step in the analytics pipeline.

In data analytics, tests should verify that the results of each intermediate step in the production of analytics matches expectations. Even very simple tests can be useful. For example, a simple row-count test could catch an error in a join that inadvertently produces a Cartesian product. Tests can also detect unexpected trends in data, which might be flagged as warnings. Imagine that the number of customer transactions exceeds its historical average by 50%. Perhaps that is an anomaly that upon investigation would lead to insight about business seasonality.

Tests in data analytics can be applied to data or models either at the input or output of a phase in the analytics pipeline. Tests can also verify business logic.

Inputs	**Verifying the inputs to an analytics processing stage** Count Verification - Check that row counts are in the right range, ... Conformity - US Zip5 codes are five digits, US phone numbers are 10 digits, ... History - The number of prospects always increases, ... Balance - Week over week, sales should not vary by more than 10%, ... Temporal Consistency - Transaction dates are in the past, end dates are later than start dates, ... Application Consistency - Body temperature is within a range around 98.6F/37C, ... Field Validation - All required fields are present, correctly entered, ...
Business Logic	**Checking that the data matches business assumptions** Customer Validation - Each customer should exist in a dimension table Data Validation - 90 percent of data should match entries in a dimension table
Output	**Checking the result of an operation, for example, a cross-product join** Completeness - Number of customer prospects should increase with time Range Verification - Number of physicians in the US is less than 1.5 million

Figure 94: Tests are applied to inputs, outputs or business logic.

The data analytics pipeline is a complex process with steps often too numerous to be monitored manually. SPC allows the data analytics team to monitor the pipeline end-to-end from a big-picture perspective, ensuring that everything is operating as expected. As an automat- ed test suite grows and matures, the quality of the analytics is assured without adding cost. This makes it possible for the data analytics team to move quickly — enhancing analytics to address new challenges and queries — without sacrificing quality.

‘The Mystery Box Full of Data Errors’

It’s a big headache – how can you fix it?

The Bane of a Data Engineering:

Do you have a dastardly dark database teaming with terrible data quality? Are the data sets around you a mystery box full of potential data errors? Is your data an enigma, wrapped in a pitch-black box, riddled with data irregularities? Or do you feel like the Forest Gump of data, where your data is like a box of chocolates, and you never know what will happen next?

Are you a member of a stressed-out, overworked data team that builds and runs data systems that produce embarrassing data errors regularly? Are you aware that only 22% of data engineers’ time is spent on innovation, but 78% on errors and manual execution (Gartner 2022)? Are you aware that a survey of 700 data engineers by DataKitchen & data.world in 2022 found that 52% of Data Engineers said errors are a significant source of burnout?

Do you have embarrassing production data errors your customer found (or, luckily, did not find)? Data errors cause your customers not to trust the data, are hard to find, take time to identify, and waste your time on re-work. Data errors can cause compliance risk and have an opportunity cost for the business because of the downtime. And wrong data or wrong reports/models themselves can cause costly business mistakes.

Do you need more time or business knowledge to create data quality validation tests? Data engineers deal with hundreds of data sets and diverse customer needs. They have backlogs on their daily task lists. So they don’t have time or energy to learn about each data set or customer to create robust production data validation tests. They need help creating tests that fit their data and customers' needs without taking too much time to set up or babysit once they go live. They also need help adding new types of tests for the business and adjusting tests when new data arrives. Can data engineers make data stewards their allies in improving data quality through testing?

What if these challenges could be fixed? What are the benefits of shining a light on data and reducing errors in production?

-

The following white paper discusses how to make embarrassing data errors a thing of the past. We will start with how data engineers are challenged to understand their data and, as a result, need help identifying problematic data records. We will also discuss how the vast majority of data engineers are so busy that they don’t know or have time to write data tests to find data errors. We will then cover ways data engineers can create automatic data tests and a process to tweak existing production tests when new data comes in. And given the complexity of data, we will discuss the need for business-level data testing.

- More Innovation – Reducing errors eliminates unplanned work that pulls data team members from their high-priority analytics development tasks. An enterprise cannot derive value from its data unless data engineers can stay focused on innovation.
- More Trust and Understanding – Errors undermine trust in data and the data team. Less confidence means less data-driven decision-making; in other words, emotionally-biased decision making. Validating the data customers use helps ensure their trust.
- Less Stress – Errors can occur at any moment. The feeling of continuous anxiety is unhealthy for team members and reduces their ability to relax and enjoy their work. Happy and relaxed people think more clearly and creatively.
- Less Embarrassment – Errors in data analytics tend to be very public. When data errors corrupt reports and dashboards, it can be highly uncomfortable for the manager of the data team. As embarrassing as having your mistakes highlighted in public, it's much worse if the errors go unnoticed. Imagine if the business makes a costly and critical decision based on erroneous data.

What Exactly Is The Problem? And Why Have We Not Solved It Already?

The data team may sometimes feel its work product is under a microscope. If the data looks "off," users can often tell immediately. Business users are experts in their domain and will often see problems in data with only a glance. It's generally an unpleasant experience for the data team to learn about analytics errors from its internal and external customers.

Problem #1: Say Nope to Hope

Data changes continuously as it flows through the system. Data can drift out of statistical range, defy data ingestion and preparation algorithms, and not meet business requirements. Manual testing of data is performed step-by-step by a person. This process tends to be expensive as it requires a precious resource, such as a data engineer, to execute test queries one at a time. Manual testing can also be prone to human error and is often too cumbersome to run frequently. Checking production data manually or not at all is a recipe for failure. Hoping your customers will not find problems in production is not a strategy. Check data with software, not a person.

Problem #2: Data Sources Gone Bad

Data changes continuously as it flows through the system. Data can drift out of statistical range, defy data ingestion and preparation algorithms, and not meet business requirements. Manual testing of data is performed step-by-step by a person. This process tends to be expensive as it requires a precious resource, such as a data engineer, to execute test queries one at a time. Manual testing can also be prone to human error and is often too cumbersome to run frequently. Checking production data manually or not at all is a recipe for failure. Hoping your customers will not find problems in production is not a strategy. Check data with software, not a person.

Your data providers occasionally give you insufficient and even incorrect or corrupted data. Get used to it. If you do not quickly find these problems in your providers' data and give them feedback, they will continually cause problems.

Problem #3: Too Much Data. Data Engineers can't understand the characteristics of every data table in their organization.

Data, especially new data, is a mystery. They can't identify obvious problems in various data sources just by looking at the data. A data engineer may deal with dozens of discrete data sets. They don't have a 'theory of the business' in their head to make judgments on data. Yet the data needs to be validated for use.

Problem #4: Too Little Time

Data Teams often have too many things on their 'to-do' list. Customers are asking for new data, people need questions answered, and the tech stack is barely running – data engineers don't have time to create tests. They have a backlog full of new customer features or data requests, and they go to work every day knowing that they won't and can't meet customer expectations.

Problem #5: It Takes Syntax, Semantics, and Pragmatics to Validate Data

To understand the time crunch that data engineers face and how it prevents them from testing their data sufficiently, we can think about three categories of data tests, using linguistics as an analogy: data syntax vs. semantic svs.pragmatics. In linguistics,Syntax is the study of sentence structure and grammar rules.While people can do what they want with language (and many often do), syntax helps ordinary language users understand how to organize words to make the most sense.

On the other hand, semantics is the study of the meaning of sentences. The sentence "Colorless green ideas sleep furiously" makes syntactic sense but is meaningless. Pragmatics takes semantics one step further because it studies the meaning of sentences within a particular context.

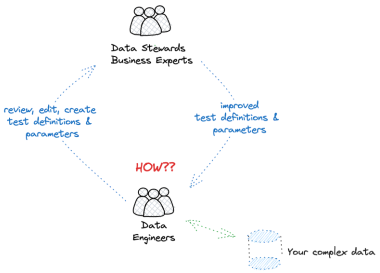
Let's apply this idea from linguistics to data engineering challenges.

- *Syntax-Based Profiling and Testing*: By profiling the columns of data in a table,you can look at values in a column to understand and craft rules about what is allowed for a column. For instance, if a column is filled with US zip codes, a row should not have the word ''bacon" in it. Data engineers need a quick way to profile data quickly and automatically cast a wide net to catch data anomalies. It's analogous to setting up a burglar alarm in your home by deploying sensors at all possible entrances to catch a burglar who may only try one window. Automatically creating tests from profile data allows teams to maintain maximum sensitivity to real problems while minimizing false positives that are not worth the follow-up.

- *Semantics-Based Business Rule Testing.* What is a meaningful test for your business? Do you know as a data engineer? For example, you can compare current data to previous or expected values. These tests rely upon historical values as a reference to determine whether data values are reasonable (or within the range of reasonable). For example, a test can check the top fifty customers or suppliers. Did their values unexpectedly or unreasonably go up or down relative to historical values? What is the acceptable rate of change? 10%? 50%? Data engineers are only sometimes able to make these business judgments. They must thus rely on data stewards or business customers to 'fill in the blank' on various data testing rules.
- *Pragmatics-Based Custom Testing.* Many companies have widely diverging business units. For example, a pharmaceutical company may be organized into Research and Development (R&D), Manufacturing, Marketing and Sales, Supply Chain and Logistics, Human Resources (HR), and Finance and Accounting. Each unit will have unique data sets with specific data quality test requirements. Drug discovery data is so different from manufacturing data that data test cases require unique domain knowledge or a specific, pragmatic business context based on each group's unique data and situation.

Problem #6: Data Changes and Testing Needs Change With It

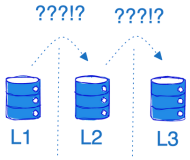
Finally, data teams must start small and quickly add more complex data tests over time. Data is not static. As their teams grow and data conditions change, they must continually add and modify their existing database profiles and test suites.



Data engineers face another challenge: understanding what tests are needed and configuring the parameters of those tests. That knowledge is often in the heads of business users or Data Stewards. How do you get those individuals involved looped in early? How can you share the responsibility to edit data test definitions and configurations with people who know your business context the best? How can we empower these non-technical people to configure tests?

Problem #7: Multiple Data Architecture Layers

Data teams today often organize their data warehouse or data lake into layers: L1, L2, and L3. The layers generally refer to different stages of data processing, storage, and access. The naming and specifics of these layers can vary somewhat depending on the context and the specific data warehousing approach, but here is a generalized description:



- **L1**-Operational Data Layer (ODL): The Source layer. This initial layer is where the data extracted from various source systems, such as operational databases, external files, web services, or third-party systems, is placed.
- **L2**-Integration Layer(IL): Also known as the Harmonization layer or the Transformation layer. Data from different source systems are cleaned, transformed, and integrated.
- L**3**-Access Layer(AL): The Presentation or Information layers. This is the layer that users send applications to interact with. The data in this layer is usually organized into facts and dimensions. This layer is where business users interact with data, often through reporting, dashboarding, and analytics tools. Data Teams face specific challenges in understanding and testing data in each layer. First, they must ensure that each layer has not inherited lost or mangled data from previous processing. They must also ensure that data properties match business logic in each layer. For example, an application may expect 1 million rows of data to arrive in L1. Via testing, please make sure that the correct quantity of data arrived in L1 and that the same quantity is present in layers L2 and L3.

Problem #7: Varied Data Engineer Team Cultures

Data Engineers adopt various development approaches. Some waterfall-driven teams need precise specifications and shareable test result artifacts to progress through various development phases. Some teams are the opposite; they are in full 'hero mode,' working night and day to add new features and data tests and rapidly fix any problems. Regardless of the approach, both teams must discover data problems rapidly and fix the errors to deliver value to their customers on time. Regardless of the approach, every data team needs a more effective and rapid method to test data

Problem #8: Will Data Tests Slow Production?

Data teams are often concerned about running data tests during the production process. Will those tests cause performance problems? Will they cause me to miss my SLA? Data engineers also don't want to duplicate data in other systems just to run tests; they need the data tests to run in the production database quickly and with low impact. They want to understand the test queries clearly and see them in the database execution log so they can quickly take the best action.

Problem #9: Data Lineage is Static Analysis in a Dynamic World

Data Teams look, at times, to Data Lineage capabilities to help reduce production errors. Data lineage answers

questions like, “Where is this data coming from, and where is it going?” Data Lineage is thus a way to describe the data assets in an organization. It is a description used to help data users understand where data came from and, with a data catalog, the content of specific data tables or files. These represent static blueprints for your data, which are valuable but insufficient. Think of it this way: if your house is on fire, you don’t want to go to town hall and rely on just the blueprints of your home to understand better how the fire could spread. You want detectors in every room so you can be alerted quickly to avoid damage. Data Lineage is the blueprint of the house; Data Testing is the set of fire detectors sending you signals in real-time. You need both to provide the complete picture. Data Lineage alone is insufficient to produce error-free data.

Problem #10: Coding Data Validation Test From Scratch

It is time-consuming to write data tests from scratch. It’s much less work to refine a small number of pre-coded tests that miss the target. It’s much more work – if not impossible in the real world – to manually create a blanket of relevant tests that can alert you when anything goes wrong. Many data syntax and semantic tests can be configured instead of coded to give you the necessary test coverage. There will always be data tests that are unique to your industry, company, or domain. Wouldn’t you like time to focus on the tests where your team can add unique value?

DataKitchen DataOps TestGen Fixes The ‘Mystery Box Full of Data Errors’

Imagine having a simple and fast way to generate and execute data quality tests without hassle, automatically. With DataOps TestGen, you can say goodbye to tedious and time-consuming manual test processes. We designed TestGen to be simple, fast, and highly efficient. It automates the entire data profiling, test generation, and execution process for SQL databases, saving you valuable time and resources. With DataOps TestGen, you can focus on what matters most - delivering trusted insight to your customers - while TestGen ensures data accuracy and quality.

DataOps TestGen creates the terms and conditions of your data contract by automatically generating valuable data tests with no coding or script writing. DataOps TestGen is your go-to "Easy Button" for database data quality monitoring and anomaly detection. The product streamlines the entire testing process, from test development through test execution, making it a breeze to identify and rectify any data errors or anomalies.

DataOps TestGen Data Profiling and Bad Data Detectors:
DataOps TestGen's first step is to profile data and produce a precise understanding of every table and column. It looks at 51 different data characteristics that have proven critical to developing practical data tests,regardless of the data domain. TestGen then performs 13 'Bad Data' detection tests, providing early warnings about data quality issues, identifying outlier data, and ensuring data are of the highest quality. The strength of your data-driven decisions is directly proportional to the quality of your data. With TestGen, we guarantee the latter so you can continually rely on the data in the database de-mystifies your data and spots complex data - without the need to write tests.

DataOps TestGen Data Quality Test Creation and Execution
TestGen produces and executes tests that validate data at rest and in motion during any production process. From data in its raw form to cleaned and prepared data sets, TestGen ensures rigorous testing, providing an extra layer of confidence in your data-driven decision-making.

One of the standout features of DataOps TestGen is the power to auto-generate data tests. With a library of 28 distinct tests automatically generated based on profiling data, TestGen simplifies the testing process and saves valuable time. These tests require minimal or no configuration, taking the heavy lifting out of your hands, so you can focus on what matters – extracting insights from your data.

TestGen also offers 11 business rule data tests that, with minimal configuration, can be used for more customized tests. These tests allow users to customize testing protocols to fit specific business requirements with a “fill in the blank” model, offering a perfect blend of speed and robustness in data testing. These types of tests ensure your data not only meets general quality standards but also aligns with your unique business needs and rules. Data stewards, who may know more about the business than a data engineer, can quickly change a setting to adjust the parameters of a data test - without coding.

DataOps TestGen Features Summary

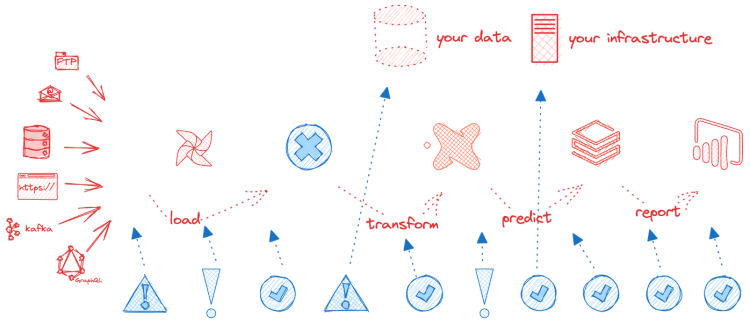
DataOps TestGen automatically creates data tests and executes those tests in your databases, saving the results. TestGen is more than just a tool; it's your strategic partner in navigating the complex world of data quality and testing. Trust in TestGen - the perfect blend of automation, adaptability, and accuracy. The capabilities of DataOps TestGen are summarized in these statistics:

Data Profiling Column Characteristics	**51**
Bad Data Detector Tests	**13**
Auto-Generated (No or Optional Configuration) Data Tests	**28**
Business Rule (Requires Configuration) Data Tests	**11**

The ‘Mystery Box Full Of Data Errors’ Is Only One Part Of The “Data Journey”

So far, this white paper focuses on the importance of testing data in databases and how DataOps TestGen can enable rapid test development and execution. However, you very likely utilize many data tools in front of that database and beyond that database: They may include Talend, Azure Data Factory, Glue, Informatica, DataBricks, Airflow, Testing Tools, other ETL Tools and Orchestrators, Data Science Tools, Dashboard Tools, bucket stores, servers, even custom tools - the potential list is very long. These tools work together to take data from its source and deliver it to your customers in a form they can use.

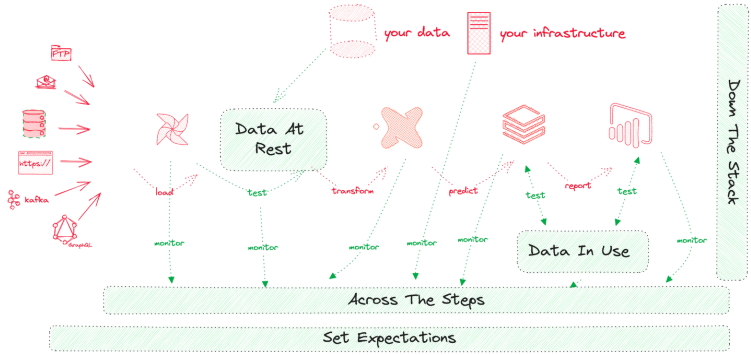
We call that multi-tool data assembly line a ‘Data Journey.’ And just like data in your database, the data across the Data Journey, and the technologies that make up the Data Journey, all have to be observed, tested, and validated to ensure success. Successful Data Journeys track and monitor all levels of the data stack, from data to tools to servers to code across all critical dimensions. A Data Journey supplies real-time statuses and alerts on start times, processing durations, test results, technology and tool state, and infrastructure conditions, among other metrics. With this information, data teams will know if everything is running on time and without errors and immediately detect the parts that didn’t.



In the illustration above, we see a typical Data Journey and the status of what is working and not working across all of its components. For each component, it's essential to compare the current state to the expected state, or the Data Journey's "expectations," at every step, from source to value delivery across multiple dimensions, including the Five Pillars Of Data Journeys:

1. Across The Steps: Check Runs,Order Of Operations,Schedules
2. Down The Stack: Monitor Metrics,Logs,And Costs
3. Data At Rest: Validate Data Quality Automatically
4. Data In Use: Test The Results Of Models,Visualizations,Delivery,And Utilization

5. Set Expectations: Compare Expected Against Reality, Alert, Analyze

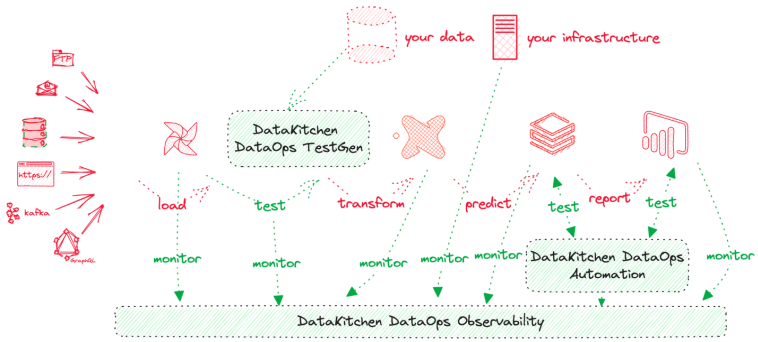


Within a Data Journey, you need to monitor a complex series of steps in an active, action-oriented way to validate the results you deliver to your customers. And you need to be alerted, in real-time, when critical expectations are not met. DataKitchen's DataOps Observability product enables this Data Journey monitoring and alerting. DataOps Observability is designed to seamlessly extract these status details and test results, including those produced by DataOps TestGen, from every Data Journey, with no changes to current jobs and processes, and compare them to expectations and alert when variances exist - allowing data teams to understand the current state of every Data Journey, enterprise-wide.

How to Embed Effective Data Tests in Data Journeys

Data In Rest'

Forcomplete' DataQualityTestingofdatabasesacrossaDataJourney,twotypesoftestsare needed. ThefirsttypeincludesDataQualityMeasurementTests.Thesetasksincludedataprofiling, lineage, and 'bad data' detection tests. The second type features *Active Data Tests* for validating data in motion during production. These are operational data tests of data in raw, cleaned, and prepared forms within databases. As described above, DataKitchen's TestGen product can help you create and execute both kinds of tests automatically with no coding.



To enable Data Quality Testing on 'Data In Use,' a solution should interact with data tools and their outputs and allow for customized, often domain-specific business tests. These include both business rule data tests and custom data tests. DataKitchen offers another test development and execution product, called DataOps Automation, that enables you to build and execute these types of tests quickly and easily.
Importantly, any tests that are developed and executed either in TestGen or Automation are automatically collected by DataKitchen DataOps Observability and compared to expectations, helping to provide a complete view of the status of every Data Journey.

Conclusion

This white paper discussed the need to automatically generate and execute data quality checks that identify errors in your data before your customer finds them. We discussed the need to test data during the production process directly in the database. We also discussed engineers' various approaches to understanding their data and identifying problematic data records. Finally, we talked about the challenge of enabling data engineers to expand from automatic data tests to a list of configurable 'fill in the blank' data tests.

D

DataOps Trail Mix Cookies

by Mike Beaverson

INGREDIENTS AND TOOLS

- 1 cup butter, room temperature
- 1 1/4 cup brown sugar
- 1/4 cup honey
- 1 tbsp vanilla extract
- 2 large eggs
- 2 1/4 cups all purpose flour
- 1/2 tsp baking soda
- 1 tsp salt
- Dash of cinnamon
- 1 bag (8-12oz) trail mix (dark chocolate, dried fruit, candy bits, nuts are all good choices)

Instructions

1. Preheat oven to 325F. Line a baking sheet with parchment paper.
2. In a large bowl, cream together butter and sugar until light and fluffy.
3. Beat in honey, vanilla, and both eggs, adding the eggs in one at a time.
4. In a medium bowl, whisk together flour, baking soda, cinnamon, and salt.
5. Working by hand or at a low speed, gradually incorporate flour mixture into honey mixture.
6. Stir in trail mix.
7. Shape cookie dough into 1-inch balls and place onto prepared baking sheet, leaving about 2 inches between each cookie to allow for the dough to spread.
8. Bake for 12-15 minutes, until cookies are golden brown.
9. Cool for 3-4 minutes on the baking sheet, then transfer to a wire rack to cool completely.

Makes about 4-dozen cookies

DATAOPS ENGINEERING

DataOps Engineer Will Be the Sexiest Job in Analytics

Years ago, prior to the advent of Agile Development, a friend of mine worked as a release engineer. His job was to ensure a seamless build and release process for the software development team. He designed and developed builds, scripts, installation procedures and managed the version control and issue tracking systems. He played a mean mandolin at company parties too.

The role of release engineer was (and still is) critical to completing a successful software release and deployment, but as these things go, my friend was valued less than the software developers who worked beside him. The thinking went something like this — developers could make or break schedules and that directly contributed to the bottom line. Release engineers, on the other hand, were never noticed, unless something went wrong. As you might guess, in those days the job of release engineer was compensated less generously than development engineer. Often, the best people vied for positions in development where compensation was better.

RISING FORTUNES

Today, the fortunes of release engineers have risen sharply. In companies that are implementing DevOps there is no more important person than the release engineer. The job title has been renamed DevOps engineer and it is one of the most highly compensated positions in the field of software engineering. According to salary surveys, experienced DevOps engineers make six figure salaries. DevOps specialists are so hard to find that firms are hiring people without college degrees, if they have the right experience.

Whereas a release engineer used to work off in a corner tying up loose ends, the DevOps engineer is a high-visibility role coordinating the development, test, IT and operations functions. If a DevOps engineer is successful, the wall between development and operations melts away and the dev team becomes more agile, efficient and responsive to the market. This has a huge impact on the organization's culture and ability to innovate. With so much at stake, it makes sense to get the best person possible to fulfill the DevOps engineer role and compensate them accordingly. When DevOps came along, the release engineer went from fulfilling a secondary supporting role to occupying the most sought-after position in the department. Many release engineers have successfully rebranded themselves as DevOps engineers and significantly upgraded their careers.

DATAOPS FOR DATA ANALYTICS

A similar change, called DataOps, is transforming the roles on the data analytics team. DataOps is a better way to develop and deliver analytics. It applies Agile development, DevOps and lean manufacturing principles to data analytics producing a transformation in data-driv- en decision making.

Data engineers, data analysts, data scientists — these are all important roles, but they will be valued even more under DataOps. Too often, data analytics professionals are trapped into relying upon non-scalable methods: heroism, hope or caution. DataOps offers a way out of this no-win situation.

The capabilities unlocked by DataOps impacts everyone that uses data analytics — all the way to the top levels of the organization. DataOps breaks down the barriers between data analytics and operations. It makes data more easily accessible to users by redesigning the data analytics pipeline to be more flexible and responsive. It will completely change what people think of as possible in data analytics.

In many organizations, the DataOps engineer will be a separate role. In others, it will be a shared function. In any case, the opportunity to have a high-visibility impact on the organization will make DataOps engineering one of the most desirable and highly compensated functions. Like the release engineer whose career was transformed by DevOps, DataOps will boost the fortunes of data analytics professionals. DataOps will offer select members of the analytics team a chance to reposition their roles in a way that significantly advances their career. If you are looking for an opportunity for growth as a DBA, ETL Engineer, BI Analyst, or another role look into DataOps as the next step. And watch out Data Scientist, the real sexiest job of the 21st century is DataOps Engineer.

Picture what you could accomplish if your organization had accurate and detailed information about products, processes, customers and the market. If your company does not have a data analytics function, you need to start one. Better yet, if data analytics is not serving as a competitive advantage in your organization, you need to *step up your game* and establish a DataOps team.

Data analytics analyzes internal and external data to create value and actionable insights. Analytics is a positive force that is transforming organizations around the globe. It helps cure diseases, grow businesses, serve customers better and improve operational efficiency.

In analytics there is mediocre and there is *better*. A typical data analytics team works slowly, all the while living in fear of a high-visibility data quality issue. A high-performance data analytics team rapidly produces new analytics and flexibly responds to marketplace demands while maintaining impeccable quality. We call this a DataOps team. A DataOps team can Work Without Fear or Heroism because they have automated controls in place to enforce a high level of quality even as they shorten the cycle time of new analytics by an order of magnitude. Want to upgrade your data analytics team to a DataOps team? It comes down to roles, tools and processes.

MEET THE DATAOPS TEAM

There are four key roles in any DataOps team. Note that larger organizations will tend to have many people in each role. Smaller companies might have one person performing multi- ple roles. *See the table down below for some key tools associated with each of the roles described as well as alternate job titles.* Most of these roles are familiar to data analytics professionals, but DataOps adds an essential ingredient that makes the team much more productive.

DATA ENGINEER

The data engineer is a software or computer engineer that lays the groundwork for other members of the team to perform analytics. The data engineer moves data from operation- al systems (ERP, CRM, MRP, …) into a data lake and writes the transforms that populate schemas in data warehouses and data marts. The data engineer also implements data tests for quality.

DATA ANALYST

The data analyst takes the data warehouses created by the data engineer and provides analytics to stakeholders. They help summarize and synthesize massive amounts of data. The data analyst creates visual representations of data to communicate information in a way that leads to insights either on an ongoing basis or by responding to ad-hoc questions. Some say that a data analyst summarizes data that reflects past performance *(*descriptive analytics*)* while future predictions are the domain of the data scientist.

DATA SCIENTIST

Data scientists perform research and tackle open-ended questions. A data scientist has domain expertise, which helps him or her create new algorithms and models that address questions or solve problems.

For example, consider the inventory management system of a large retailer. The company has a limited inventory of snow shovels, which have to be allocated among a large number of stores. The data scientist could create an algorithm that uses weather models to predict buying patterns. When snow is forecasted for a particular region it could trigger the inventory management system to move more snow shovels to the stores in that area.

Roles	Other Job Titles	Responsibilities	Skills	Tools
Data Engineer	Database Architect Data Modeler, Database Administrator, Data QA Engineer, ETL Engineer	Data lakes, Data warehouses, Data marts, Schema design	Databases, Programming, Cloud infrastructure, Simple storage	SQL, Informatica, DataStage, SSIS, Talend
Data Analyst	Data Visualization Designer, Business Data Analyst, BI Tableau Developer, Reporting Analyst, Business Intelligence Engineer	Visualizations: Charts, Graphs, Dashboards, Tables, Reports	Programming, Statistics, Machine learning, Data cleaning, Data visualization	Excel, Looker, Tableau, Qlik View, Altryx, Spotfire
Data Scientist	Machine Learning Researcher, Machine Learning Engineer, Quantitative Analyst, AI Programmer Actuary	Algorithms, Models	Domain subject matter expert, Advanced mathematics, Machine learning, Data mining tools, Programming	R, Python, SAS, SPSS
DataOps Engineer		Orchestrating the analytic pipeline, Promoting features to production, Automating quality	Agile Development, DevOps, Statistical Process Control	DataKitchen, data test frameworks, python, shell scripts

Table 7: The DataOps Team

DATAOPS IS THE PROCESS AND THE TOOLS

Many data analytics teams fail because they focus on people and tools and ignore process. This is similar to fielding a sports team with players and equipment, but no game plan describing how everyone will work together. The game plan in data analytics is included in something that we call DataOps.

DataOps is a combination of tools and process improvements that enable rapid-response data analytics, at a high level of quality. Producing analytics that are responsive, flexible, continuously deployed and quality controlled requires data analytics to draw upon techniques learned in other fields.

- **Agile Development –** an iterative project management methodology that completes software projects faster and with far fewer defects.
- **DevOps –** a software development process that leverages on-demand IT resources and automated test and deployment of code to eliminate the barriers between development (Dev) and operations (Ops). DevOps reduces time to deployment, decreases time to market, minimizes defects, and shortens the time required to resolve issues. DevOps techniques help analytics teams break down the barriers between data and ops (DataOps).
- **Lean Manufacturing –** DataOps utilizes statistical process control (SPC) to monitor and control the data analytics pipeline. When SPC is applied to data analytics, it leads to remarkable improvements in efficiency and quality. With quality continuously monitored and controlled, data analytics professionals can Work Without Fear or Heroism.

The process and tools enhancements described above can be implemented by anyone on the analytics team or a new role may be created. We call this role the DataOps Engineer.

DATAOPS ENGINEER

The DataOps Engineer applies Agile Development, DevOps and statistical process controls to data analytics. He or she orchestrates and automates the data analytics pipeline to make it more flexible while maintaining a high level of quality. The DataOps Engineer uses tools to break down the barriers between operations and data analytics, unlocking a high level of productivity from the entire team.

As DataOps breaks down the barriers between data and operations, it makes data more easily accessible to users by redesigning the data analytics pipeline to be more responsive, efficient and robust. This new function will completely change what people think of as possible in data analytics. The opportunity to have a high-visibility impact on the organization will make DataOps engineering one of the most desirable and highly compensated functions on the data-analytics team.

How To Succeed as a DataOps Engineer

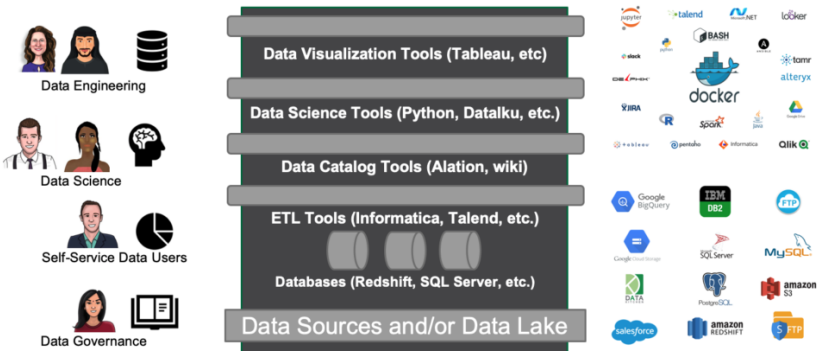
What makes an effective DataOps Engineer? A DataOps Engineer shepherds process flows across complex corporate structures. Organizations have changed significantly over the last number of years and even more dramatically over the previous 12 months, with the sharp increase in remote work. A DataOps engineer runs toward errors. You might ask what that means. After all, we all deal with errors. Errors are an inherent part of data analytics. A DataOps Engineer embraces errors and uses them to drive process improvements.

Curating Processes

The product for a data engineer is the data set. For an analyst, the product is the analysis that they deliver for a data object. For the DataOps Engineer, the product is an effective, repeatable process. DataOps Engineers are less focused on the next deadline or analytics deliverable. We're looking to create a repeatable process. Individual nuggets of code are fungible from our perspective. Success is all about process reliability and consistency.

The DataOps Engineer leverages a common framework that encompasses the end-to-end data life cycle. Our goal is to seek out opportunities for reuse in the work that other data team members share with us. It can be as simple as encapsulating common joins, unions, and filters or creating views. It can also be as challenging as carving out reusable steps from orchestrations or analytics that people share with us. The goal is to identify any steps that many different people in different roles have to do repeatedly to do their work. If we can provide shortcuts, we can create leverage and give people a boost. Many organizations take weeks to procure and prep data sets. A DataOps Engineer can make test data available on demand. If we can provide shortcuts to members of the data team, we can help improve their productivity.

DataOps Engineers have tools that we apply to all of the pipeline orchestrations that we manage. We do timeliness tracking for data sources, builds, and jobs. We want to know if a data source failed to deliver an update before a complex build is kicked off. The DataOps Engineer can automate the creation of artifacts related to data structures, such as change logs that are automatically updated. We have data profiling tools that we run to compare versions of datasets. We have automated testing and a system for exception reporting, ~~where tes~~ts identify issues that need to be addressed. All this serves to Increase transparency, which is critical to increasing trust in the process and analytics work product.



Shepherding Processes Across the Corporate Landscape

The flow of data is fundamental to the organization. You pull information from wherever it's generated, transform it and summarize it. Then you redistribute it to where it's needed for people who have to make decisions and act.

DataOps Engineers are not just focused on their narrow swimlanes. They must consider the end-to-end people, processes and tools that act upon data – figure 1. The path of information doesn't follow the lines of authority or the traditional hierarchies of an organization. We have to follow the data wherever it goes. Data paths span departments, buildings, time zones, companies, cultures, and countries. The DataOps Engineer serves as a catalyst to bring *siloed* contributors and teams together.

A previous post talked about the definition of "done." In DataOps, the understanding of the word "done" includes more than just some working code. It considers whether a component is deployable, monitorable, maintainable, reusable, secure and adds value to the end-user or customer. Sometimes people confuse "doneness" with ownership. We often refer to data operations and analytics as a factory. We distinguish between owning the assembly lines of the data factory (DataOps Engineer) and owning individual steps within the assembly lines (data scientists, engineers, etc.).

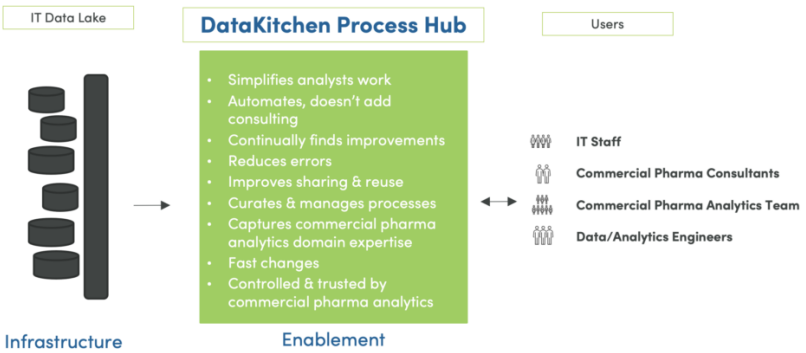
When people become concerned about defending their turf, it leads to less transparency, less reusable logic, and loss of control over source code. It is harder to QC (quality control) methods and data spread among different segments and places. More points of failure ultimately lead to less reliable results. One approach that has worked well for the DataOps Engineering team at DataKitchen is the concept of building a process hub.

Process Hub

A process hub is a coherent common framework, shared workspace, and shared set of services that coordinate tasks and amplify the value contributed by various workflow participants. For example, one of the services built by our Data Engineers is an engine that generates QC rules from baseline data. It then autogenerates QC tests based on those rules. It's an excellent way to quickly scale up to a large set of small tests for a new orchestrated process. Once this capability is in place, data scientists, analysts and engineers can take advantage of it.

The common framework of a process hub implicitly encourages collaboration, and it's essentially a pipeline from a development process to a production process. The process hub that ties development and production together shows how they relate to each other – figure 2. Development and production exist as different points within a workflow, as do the users who work within the process framework.

The process hub allows everyone to leverage everyone else's best work, and it liberates people from drudgery. For example, analysts keep working on insights. They don't have to understand how to deploy analytics into production – an automated QC and deployment orchestration performs that job. When analysts stay focused, it speeds up deployment. Another advantage to a process hub is transparency. With workflows "on the grid," the organization can view process activity at an aggregated level. You can track, measure and create graphs and reporting in an automated way. The result is a single version of the process that supports a single version of the truth.



Run Toward Errors

DataOps Engineers run towards errors. This may sound trite, but it is very challenging. Errors drive the feedback loop that makes complex processes reliable. There is complexity in data flows that cut across so many different structures that we can't possibly anticipate everything that will go wrong. We try. There are always going to be surprises. So each error is an opportunity to go back and improve reliability. Consider a machine learning example. ~~Let's say~~ a data scientist has developed a model that works perfectly with training data. You don't necessarily see its bias and variance. When subjecting it to real-world data, it may behave differently, and you must adjust accordingly. *Errors are data* just like any other data and need to be understood and analyzed for data professionals to do their jobs more effectively.

Best Practices Related to Errors

If a DataOps Engineer actively seeks to avoid errors, they will fail (or succeed less). In DataOps, avoiding errors is doing any of the following:

- Meet the spec (and only the spec) – narrowly define success so you can prematurely declare victory. In one situation, we had a partner performing updates based on matching identifiers from a data set that we provided them. They sent us back an email saying that the process had been successful. What they didn't tell us was that there were a large number of missing records and the spec said to exclude those. The missing records were due to an upstream problem. We missed critical updates, and this partner could have alerted us had they not been dogmatically following the spec.
- Demand perfection (and stifle feedback) – Perfection is usually unrealistic, and you're not going to get it. What you will get is less honest feedback. People will hide their mistakes, and you won't find out about errors until they explode in a high-profile data outage.
- Resist disruption (and avoid change) – resisting change keeps a system static and unable to adapt to changing business conditions.
- Protect your secrets (and hide error risks) – people generally don't like negative attention. Still, one thing I have found at DataKitchen is that changing the culture and attitudes around errors helps to improve processes. It really makes a difference.
- QC the end-product (not the process) – you can confirm and validate that you've made one deliverable work successfully, but if you have to repeat it again and again, you need to QC every step of a process
- Make problems disappear (without solving them) – Senior data professionals know dozens of shortcuts for making a problem go away without actually addressing it. One of the biggest mistakes is to misuse the SELECT DISTINCT statement in SQL to make duplicate records disappear. It may work around a short-term error, but it doesn't necessarily make the data more correct for the next person who might be working on a different use case. Another classic trick in RedShift is to set MAXERRORS to a high value. Doing so will sidestep a data error, but could potentially admit thousands of errors into the system. It is better to just fix the RedShift error upfront.

The DataOps approach encourages the data team to l*ove our errors*. We want to manage the possibility of errors as a routine part of our daily effort to improve the overall system. Every time we see an error, we address it with a new automated test. Our system designs are built with the expectation that people will make mistakes. We don't want to embarrass anyone. We want to empower people, not blame them. We all need to know when these issue issues arise and what we do about them.

Another DataOps best practice curates mistakes. Our team has a checklist of silly errors that recur frequently. We want to avoid making those errors a second time. We're also testing data at every step of the process. You catch more errors testing each step of a pipeline instead of just testing at the end.

Testing along the way also enables us to recover faster in a critical situation. It helps to know precisely where in a complicated and lengthy sequence an error was introduced. From personal experience, I favor a large number of simple tests rather than a few complex ones. You can't anticipate all of the different things that can wrong so casting a wide net of simple tests can be more effective.

Another best practice is to double-check what you already know is true. Some day someone may change data inputs or analytics or introduce an error in an upstream process. Testing what you already know can help you find errors in business logic that stem from future scenarios that you can't anticipate today.

DataOps Collaboration

It's essential to communicate and provide timely feedback about errors. I was involved with an upstream system that neglected to apply an overhead factor to a financial calculation. The downstream analysts caught the problem, and instead of asking the upstream team to fix it, they worked around the problem by applying the overhead factor to their downstream analytics. The upstream team had no clue.

One bright day, someone on the upstream team fixed the bug and broke all of the downstream analytics, which were now applying the overhead factor a second time. The upstream team had been trying to fix a problem but unintentionally made it worse. Timely communication is essential in data organizations.

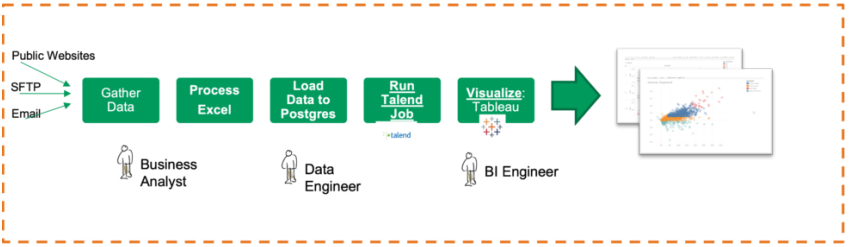
DataOps is first and foremost about collaboration. It's about reuse. It's about creating tested reliable functionality that is leveraged as a building block in a system. With a carefully constructed process hub, you can successfully minimize errors and create a culture of transparency.

A Day in the Life of a DataOps Engineer

A DataOps implementation project consists of three steps. First, you must understand the existing challenges of the data team, including the data architecture and end-to-end toolchain. Second, you must establish a definition of "done." In DataOps, the definition of *done* includes more than just some working code. It considers whether a component is deployable, monitorable, maintainable, reusable, secure and adds value to the end-user or customer. The final step is designing a data solution and its implementation

List of Challenges

A DataOps project begins with a list of challenges. Having worked with multiple customers on various use cases, scenarios and different toolchains, I believe there are commonalities in all DataOps projects. The biggest challenge is broken data pipelines due to highly manual processes. One project that I worked on involved dependencies between multiple teams and various roles. Figure 1 shows a manually executed data analytics pipeline. First, a business analyst consolidates data from some public websites, an SFTP server and some downloaded email attachments, all into Excel. She applies some calculations and forwards the file to a data engineer who loads the data into a database and runs a Talend job that performs ETL to dimensionalize the data and produce a Data Mart. The data engineer then emails the BI Team, who refreshes a Tableau dashboard.



There are numerous challenges with this process, as described below. There are no automated tests, so errors frequently pass through the pipeline. There are communication delays because of the multiple stakeholders belonging to different teams, each having its own priorities. The delays impact delivery of the reports to senior management, who are responsible for making business decisions based on the dashboard.

There's a fear of making changes to the process as it might break production. There is no process to spin up an isolated dev environment to quickly add a feature, test it with actual data and deploy it to production.

The process has no monitoring whatsoever to track the actions performed by the developers, the timeline and the final output.

Definition of Done

When is a project ready to be pushed to production? When can you declare it done? Keeping DataOps principles in mind, a project is done when you can orchestrate all the tools, team environments and processes in one single pipeline, and you can manage these different environments – development, test, production, etc. All your code is version controlled in a system like GitHub, and you're able to maintain history and track changes. The code is parameterized so that it is reusable across different environments. The DataOps pipeline you have built has enough automated tests to catch errors, and error events are tied to some form of real-time alerts.

The project is done when you can decrease the cycle time of change and quickly add and deploy a new feature. Finally, when your implementation is complete, you can track and measure your process.

DataOps Project Design and Implementation

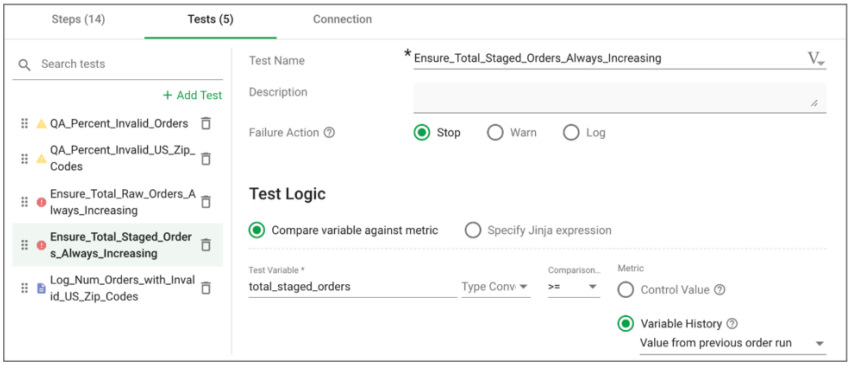
As a DataOps implementation engineer, I transformed the use case described above into a DataOps solution – figure 2. I started by gathering all the analytic "*nuggets*" from all the teams: business rules from the business analyst, the SQL scripts from the data engineer, the Talend job from the ETL developer, and the Tableau report from the BI engineer. I quickly designed the DataOps solution to orchestrate the nuggets, tools, and environments in a single pipeline.

During implementation, my focus is to automate all the processes so that the data engineers, BI engineers and analysts can spend more time implementing new features and less time fixing errors and worrying that their processes will fail.

In this project, I automated data extraction from SFTP, the public websites, and the email attachments. The automated orchestration published the data to an AWS S3 Data Lake. I use the business rules I received from the business analyst to process the source data using Python. I then orchestrated the SQL scripts, the Talend job and the Tableau dashboard refresh, all in one workflow using the APIs these tools natively provide. All the code, Talend job, and the BI report are version controlled using Git. The pipeline has automated tests at each step, making sure that each step completes successfully. Based on business rules, additional data quality tests check the dimensional model after the ETL job completes. In case of a test failure, I have alerts configured via email – you can also configure the alerts to go to Slack or Microsoft Teams or any other collaboration tool that your team uses.

Adding Tests to Reduce Stress

While implementing a DataOps solution, we make sure that the pipeline has enough automated tests to ensure data quality and reduce the fear of failure. Below is an example historical balance test. I wanted to make sure that the total number of sales orders is monotonically increasing. The test compares the current value of the total number of orders with the most recent value of the same variable – figure 3. If that number ever decreases, something is wrong. If there's an error, I configured the pipeline to stop executing. Imagine receiving a call from your CEO because sales on the CEO dashboard were off by a million dollars. With tests, errors like these are caught before the data shows up in reports.



Below are some example DataOps tests that should be added to pipelines:

- Location Balance – make sure that the number of rows in the data matches the expected value (or threshold) at each stage in the pipeline, or make sure that if you're moving some files, they're not corrupted.
- Historic Balance – compares current data to previous or expected values. These tests rely upon historical values as a reference to determine whether data values are reasonable (or within the range of reasonable).
- Statistical Process Control – applies statistical methods to control a process
- Data Completeness – check for missing data
- Data Correctness – test for incorrect data
- Business Rule-Based – ensure that data matches business assumptions

These tests could be written in any language or tool. It could be SQL, a Python script or a shell script. The environment shown in my example is from the DataKitchen UI, which makes it simple to define tests even if you have no familiarity with the underlying tools executing the pipeline.

Parameterizing Code

In this solution, the SQL code and the Python code are completely parameterized to ensure reusability. Parameterization makes it possible to deploy the code across environments without any changes. For example, the SQL code in figure 4 uses Jinja Templating, a Python templating language embedded within the SQL code itself. This approach allows the same query to run in different databases or schemas, or even a different cluster, by updating the connection information and values specified in a couple of variables. The same code we wrote to move data from S3 to RedShift or SFTP to S3 or to publish a Tableau workbook can be used across teams, projects and environments. Jinja Templating can also be used with other structured languages or scripts, like the Tableau workbook code in figure 5.

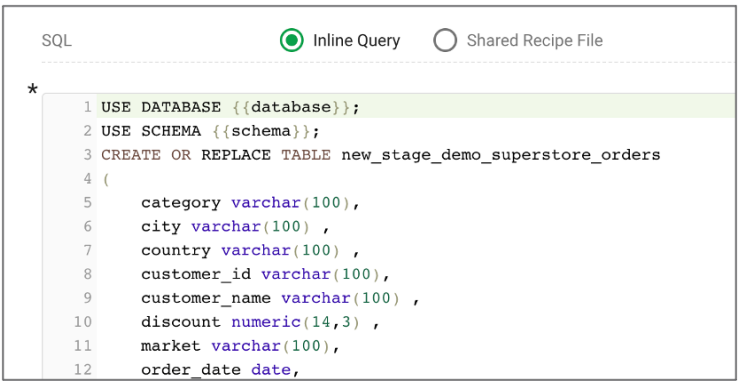
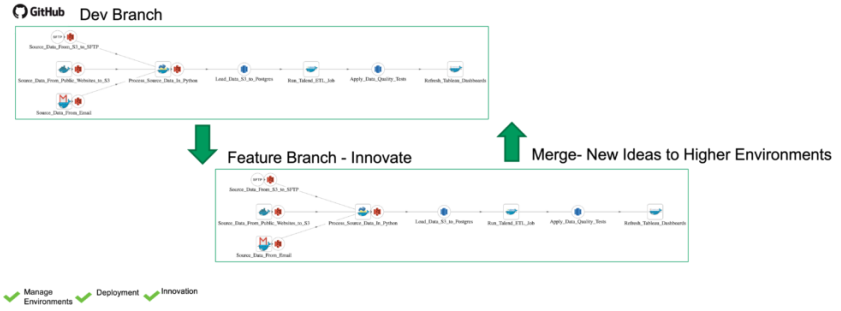


Figure 4: The database and schema are templatized using Jinja Templating to allow easy customization to different databases, schemas and clusters.

Figure 5: The Tableau and Redshift passwords are customized using Jinja Templating

Using Version Control

Adding a version control system such as GitHub enables us to manage environments, supporting faster innovation and code deployment. Suppose I want to add a new feature. In that case, I can create a feature branch and configure it to connect to an isolated environment, for example, a different S3 bucket, a separate database cluster and a dev project in the Tableau server – figure 6. I can then develop my new feature in my feature branch (sandbox environment) without breaking production. When coding is done, I test the code in the feature development environment. When it's ready, I can merge it to higher environments and rerun the same tests. This way, automated testing is built into every stage of the release and deployment workflow, and it ensures that my pipeline is delivering high-quality analytics every time it executes.



Monitoring Job Metadata

Monitoring and tracking is an essential feature that many data teams are looking to add to their pipelines. Figure 7 shows how the DataKitchen DataOps Platform helps to keep track of all the instances of a job being submitted and its metadata. The table shows the job submitted, the submitter, the start time, the duration, and the job completion status for each job.

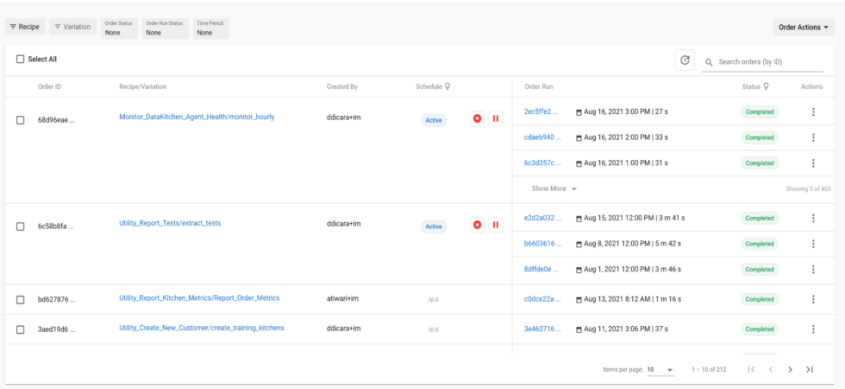


Figure 7: the DataKitchen DataOps Platform keeps track of all the instances of a job being submitted and its metadata.

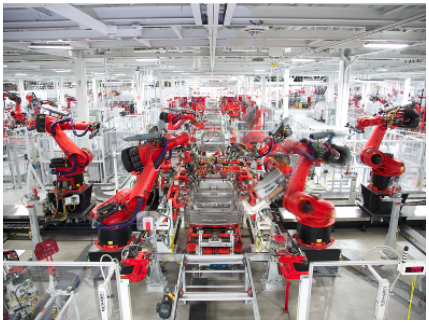
I also record all these metrics and analyze them to build a DataOps report to measure the progress in collaboration, keep track of production error rates, and keep track of the data error rates, test coverage and SLAs (service level agreements) – figure 8.

By introducing the change in our mindset, taking inspiration from methodologies, like Agile, DevOps and lean manufacturing, we can streamline the workflows, catch errors much earlier in the process, increase the productivity of the data teams and deliver high-quality analytics faster. In some cases, DataOps has helped us save hours, weeks and even months of work.

What is a DataOps Engineer?

A DataOps Engineer owns the assembly line that's used to build a data and analytic product. Data operations (or data production) is a series of pipeline procedures that take raw data, progress through a series of processing and transformation steps, and output finished products in the form of dashboards, predictions, data warehouses or whatever the business requires. We find it helpful to think of data operations as a factory.
Most organizations run the data factory using manual labor. From surveys, we know that data scientists and other data professionals spend over 50% of their time executing procedures supporting data operations.

Figure 1 is an automobile assembly line in the early 20th century. We see people lined up in a row hand assembling components. Too many data organizations run data operations like a hundred-year-old car factory. While car companies lowered costs using mass production, companies in 2021 put data engineers and data scientists on the assembly line. Imagine if a car company asked the engineers who designed cars to also build them. That's the state of data analytics today.



A DataOps Engineer transforms the picture above to the automated factory below (figure 2). Processes and workflows are highly engineered and automated. Where are the data engineers, scientists and analytics? They aren't needed on the factory floor. They are in their offices, where they belong, programming the robots and designing new automated procedures for producing continually improved versions of products, i.e., analytics.

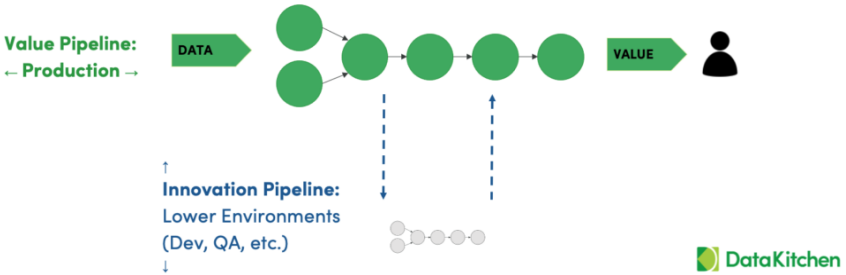
DataOps Engineer designs the data assembly line so that data engineers and scientists can produce insight rapidly and with the fewest possible errors. You might say that DataOps Engineers own the pipelines and the overall workflow, whereas data scientists and others work within the pipelines.

What is DataOps

DataOps is a set of practices, cultural norms and architecture patterns that help data professionals deliver value quickly. DataOps enables:

- Rapid experimentation and innovation for the fastest delivery of new insights to customers
- Low error rates
- Collaboration across complex sets of people, technology and environments
- Clear measurement and monitoring of results

DataOps establishes a process hub that automates data production and analytics development workflows so that the data team is more efficient, innovative and less prone to error. In this blog, we'll explore the role of the DataOps Engineer in driving the data organization to higher levels of productivity. A more technical discussion will follow in the next edition of this blog series.



Automating the Value and Innovation Pipeline

A data pipeline is a series of steps that transform raw data into analytic insights that create value. These pipelines cut across roles and organizations. Steps in the pipeline are represented in figure 3 by circles. Data engineers, scientists, analysts, governance and other data roles work inside the circles or create pipeline segments that are combined with other pipelines.

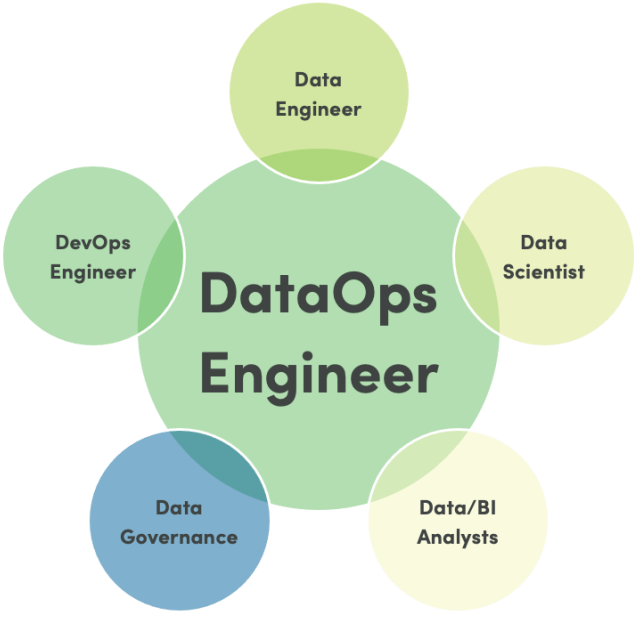
The Value Pipeline represents data operations where data progresses on its journey to charts, graphs and other analytics which create value for the organization. The Innovation Pipeline includes analytics development, QA, deployment and the rest of the change management processes for the Value Pipeline. Data professionals work at various points in these pipelines. Collectively, we want to be confident that the Value Pipeline is executing without errors, and we want to seamlessly deploy new analytics without breaking anything or creating side effects. The DataOps Engineer makes the whole system work better. The data organization wants to run the Value Pipeline as robustly as a six sigma factory, and it must be able to implement and deploy process improvements as rapidly as a Silicon Valley start-up.

data engineer builds data transformations. Their product is the data. The data scientist's products are models and segmentations. The data analyst's products are charts, graphs and visualizations. In a sense, the DataOps Engineers draw a line around all of these roles and foster greater collaboration among the data team.

When is a Task "Done"

Many people who work with data have a narrow definition of being "done." Let's say a person is building a bunch of SQL or a Jupyter notebook. They complete their work and throw it over the wall with pride. They are "done," right?
The narrow definition of "done" used by many data professionals is that it worked in a specific environment, without knowing or caring about the challenges of the people who have to deploy, monitor and maintain that component. Such a willfully blind data professional is task-focused, not value-focused. Collaboration will be difficult if you notice the following practices among the data team:

- Throw it over to production – let them figure it out
- Definition of "done" means "it worked for me"
- "I only focus on my little part"
- "Someone else's problem"
- Willful blindness
- Task – not value – focused
- Project – not product –focused
- "Hope that it works"
- Reliance on manual checks
- Lack of willingness to pull the pain forward



DataOps takes a broader view. “Done” means the component works in production and customers/users are happy with it. Instead of focusing on a narrowly defined task with minimal testing and feedback, DataOps focuses on adding value. Through automated testing, DataOps institutionalizes collaboration through orchestrated testing, monitoring, deployment and task coordination (figure 4).

Orchestrating Nuggets

Data scientists and analysts create, what we call, *nuggets* of code. Nuggets could be ETL code that drives an Informatica job, a SQL transformation, some Python or perhaps XML. No matter its consistency or purpose, nuggets must be thoroughly tested before being inserted into larger assemblies (pipelines). Below are examples of the ways that DataOps Engineers work with “nuggets” of code within a DataOps system:

- Add to pipelines
- Create tests
- Run the factory
- Automate deploys
- Work across people
- Measure success
- Enable DataOps self-services

DataOps Engineers are not usually solving data problems by creating the nuggets. They are solving process problems by using automation to test, deploy and maintain the nuggets in a system context. By automating the housekeeping related to nuggets, DataOps Engineers enable nugget creators to work more quickly and iteratively.
DataOps is applying automation to streamline workflows. As a general rule, any operation that is performed manually three times should be automated. The DataOps Engineer creates these automated orchestrations and meta-orchestrations. Below are some common examples of DataOps automation:

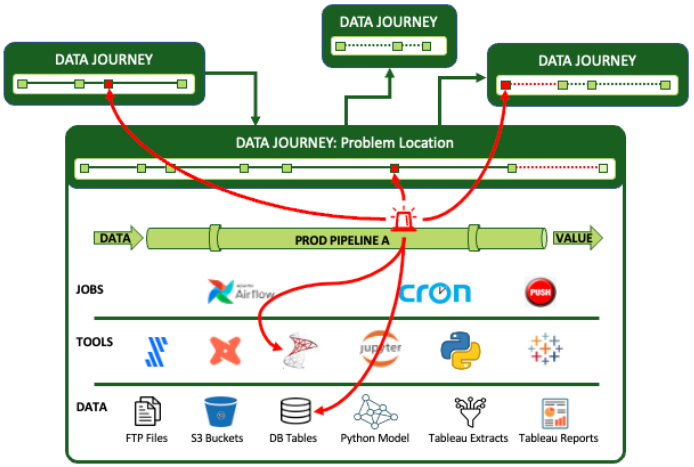
- Production orchestration – replace manual procedures that execute data operations with automated orchestrations and meta-orchestrations
- Production data monitoring/testing – create tests so you catch errors before your customers see them, monitor your production and development pipelines in real-time
- Self-service environments – give the data team a way to create on-demand development sandboxes with data and toolchains aligned with production

- Development regression and functional tests – automate development and deployment testing
- Test data automation – create test data for development on-demand
- Deployment automation – deploy with minimal keyboarding
- Shared components – standardize and reuse commonly used “nuggets” and pipelines and horizontal infrastructure
- Process measurement – construct dashboards on every aspect of the data lifecycle for unprecedented process transparency

DataOps Engineering is about taking these invisible processes and making them highly visible through an automated, shared abstraction. DataOps pulls the discovery of errors forward so they can be addressed early and never corrupt user analytics and dashboards.

The DataOps Engineer First Step: Observe The Data Journey

In the data world, we focus a lot on the data. If the data in your database needs to be corrected, has been transformed with errors, or is not clearly understood, then using that data will be a problem. However, you have many data tools in front of that database and behind that database: Talend, Azure Data Factory, DataBricks, Custom Tools, Custom Testing Tools, ETL Tools, Orchestrators, Data Science Tools, Dashboard Tools, bucket stores, servers, etc. Those tools work together to take data from its source and deliver it to your customers.



That set of multi-tool set of expectations is a ‘Data Journey. The Data Journey concept is about observing, not changing, your existing data estate. Data Journeys track and monitor all levels of the data stack, from data to tools to servers to code to tests across all critical dimensions. It supplies real-time statuses and alerts on start times, processing durations, test results, costs, and infrastructure events, among other metrics. With this information, you can know if everything ran on time and without errors and immediately detect the parts that didn’t.

The Challenges of Automation

The challenges of implementing data automation mirror some of the difficulties that we've reviewed above:

- No owner
- Not enough time spent on automation
- No one seems to care
- Everyone focused on tasks at hand instead of building better processes
- Perception that automation work is less prestigious than data work
- Unlike software dev, data can't be automated
- Manual work is the way we have always done it

When a task like automation is not recognized as important, no one takes responsibility for it. In many data organizations, there is a view that being a data scientist is the highest form of contribution. Given the impact of automation on team productivity, it should come as no surprise that top DevOps Engineers are some of the most highly compensated individuals in the software industry. We've come a long way from the era in which release engineers were considered second-class citizens.

The DataOps Engineer Second Step: Automate

The goal of the DataOps Engineer is to automate the organization's processes:

- Reduce waste.
- Promote reuse. Prevent "reinventing of the wheel."
- Lower errors and mean time to failure.
- Promote version control.
- Highlight gaps and improve testing.
- Ensure data security standards are applied to pipelines.
- Make reports on every aspect of the data team's processes and the data lifecycle.

The DataOps Engineer is a cousin of the DevOps Engineer but with familiarity with data ecosystem tools and methods. What's great and challenging about the role is that it requires several different skills in a wide range of domains. The skillset for a DataOps Engineer is as follows:

- A Scripting Language: Python, Bash
- Data Language: SQL
- Source Code Control: git
- DataOps Tool: DataKitchen
- DevOps Configuration Tool: Terraform, Puppet, Docker/K8s
- Process Skills: Agile methods & tools like JIRA
- Familiarity with the toolchain your data engineers, scientists, analysts and governance team use

As a manager or team leader, you may be wondering whether you neDataOps-type DataOps Engineer and how much of your team's resources need to be devoted to DataOps. There is no one right answer that fits all organizations. Cutting-edge software development teams spend about 23% of their time on DevOps. Data teams currently spend about 3% of their time on DataOps type tasks. We recommend increasing this level to about 15%. That could be achieved by involving everyone or by hiring a dedicated resource. The point is that investment in DataOps will pay back big time.

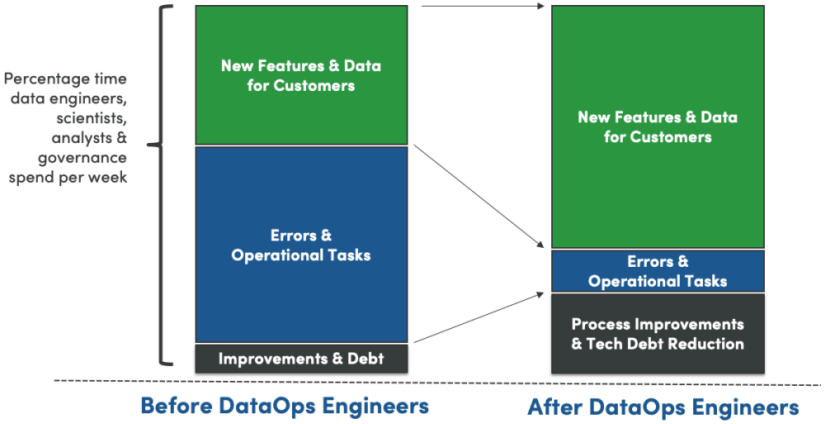


Figure 5: Time allocation before and after DataOps

The investment in DataOps impacts the whole team (figure 5). With DataOps, data engineers, scientists, analysts, and self-service users spend more time creating value, reducing technical debt or deploying changes to production, and less time chasing errors, sitting in meetings, and managing issues.

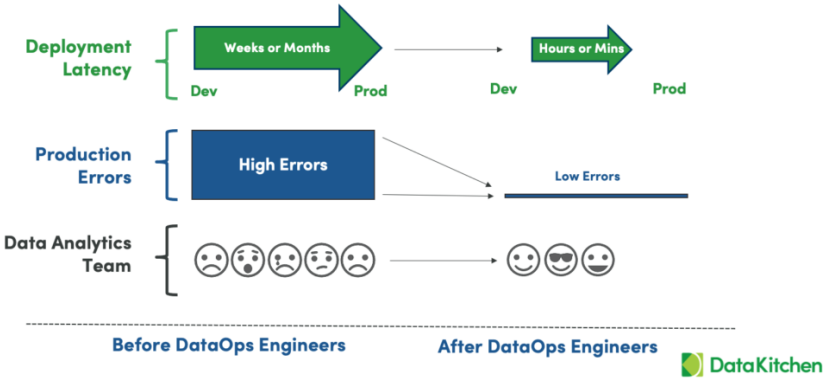


Figure 6: DataOps reduces cycle time and eliminates errors, making for a happier data analytics team.

The bottom line metrics that DataOps impacts are deployment latency and errors (figure 6). DataOps slashes deployment cycle time from weeks/months to hours/minutes. It reduces the high rate of data errors present in most data organizations to virtually zero. When productivity is high, errors are low, and users are happy, the work environment for the data team is much more enjoyable. It may sound cliche, but DataOps puts the fun back in data analytics and smiles on the faces of the data team.

Improve Business Agility by Hiring a DataOps Engineer

> *It is not the strongest of the species that survives, nor the most intelligent that survives. It is the one that is most adaptable to change.*
> *– Leon C. Megginson on* Charles Darwin "Origin of Species"

Adapt or face decline. The agile alliance defines "business agility" as the ability of an organization to sense changes internally or externally and respond accordingly in order to deliver value to its customers. Responsiveness and flexibility can enable a business to survive disruptive change and thrive in uncertain times. Companies that move slowly get left behind.

The agile alliance definition of business agility consists of two parts. First, a business has to *sense change,* and next, *respond* accordingly. If a company is slow to perceive change, then it will be slow to react.

Data-driven companies sense change through data analytics. Analytics tell the story of markets and customers. Analytics enable companies to understand their environment. Companies turn to their data organization to provide the analytics that stimulates creative problem-solving. The speed at which the data team responds to these requests is critical. A business cannot adapt to change faster than its ability to understand itself and its environment. Rapid, responsive analytics enable business agility. Slow and inflexible analytics development processes can be an early bottleneck that limits data-driven agility. Data analytics agility is the most critical and, often overlooked, component of business agility.

The Role of DataOps and the DataOps Engineer

The agility of analytics directly relates to data analytics workflows. If the data team spends half their time executing data operations without automation, they can't be agile. If it takes months to spin up a development environment, then analytics projects will become irrelevant before they are completed. If the data team is always dealing with data errors and putting out fires, then they'll be constantly pulled away from their highest priority projects.

You can transform your data analytics workflows by applying methodologies like agile development, DevOps, and lean manufacturing to data pipelines and analytics workflows. Within the data industry, this effort is called DataOps, and it is implemented by someone called a DataOps Engineer. If you want to attain greater business agility through faster, more responsive data analytics, then the DataOps Engineer should be your first hire.

DataOps Engineers implement the continuous deployment of data analytics. They give data scientists tools to instantiate development sandboxes on demand. They automate the data operations pipeline and create platforms used to test and monitor data from ingestion to published charts and graphs.

Through tools automation, the DataOps Engineer eliminates data lifecycle bottlenecks, which sap data team productivity. A DataOps Engineer who understands how to automate and streamline data workflows can increase a data team's productivity by orders of magnitude. A person like that is worth their weight in gold. The role of the DataOps Engineer goes by several different titles and is sometimes covered by IT, dev, or analyst functions. The DataOps Engineering skillset includes hybrid and cloud platforms, orchestration, data architecture, data integration, data transformation, CI/CD, real-time messaging, and containers.

The capabilities unlocked by DataOps impacts everyone that uses data analytics all the way to the top levels of the organization. DataOps breaks down the barriers between data analytics development and data operations. It makes data more easily accessible to users by redesigning the data analytics pipeline to be more flexible and responsive. It improves agility, which can positively impact a company's competitiveness. The rise of the DataOps Engineer will completely change what people think of as possible in data analytics.

With lightning-speed analytics, companies can more nimbly develop and execute strategies that build value. They will have greater success in disrupting markets and establishing a sustained competitive advantage. The DataOps Engineer plays a critical role in making agile analytics happen.

Why DevOps Tools Fail at DataOps

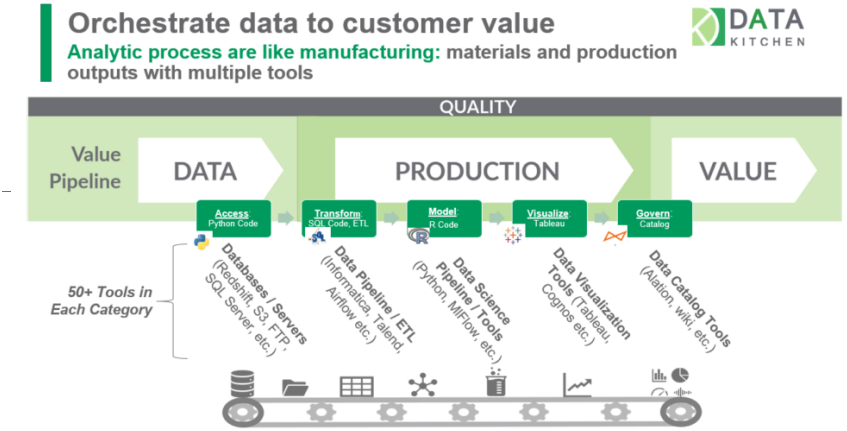
Implementing DataOps requires a combination of new methods and automation that augment an enterprise's existing toolchain. The fastest and most effective way to realize the benefits of DataOps is to adopt an off-the-shelf DataOps Platform.

Some organizations try to implement DataOps from scratch using DevOps and workflow tools. However, DataOps is not just DevOps for data, and using DevOps tools in the data science and analytics domain will not easily lead to DataOps success. DevOps tools will leave significant gaps in your DataOps processes. Here we describe the important ingredients required for DataOps, without which companies will falter on their DataOps journey.

End-to-End Production Pipelines

There are hundreds of tools for data engineering, data science, analytics, self-service, governance, and databases. Meta-orchestration of the complex toolchain across the end-to-end data analytics process is a crucial element of DataOps.

Think of your analytics development and data operations workflows as a series of steps that can be represented by a set of **directed acyclic graphs** (DAGs). Each node in the DAG represents a step in your process. In most enterprises, the production and development pipelines respectively are not one DAG; they are a DAG of DAGs. Typically, different groups within the data organization use their own preferred tools. The toolchains may include multiple orchestration tools. It's challenging for a software engineer, let alone a data scientist, to learn all of the different tools used by the various data teams (Figure 1). Furthermore, orchestration tools do not adequately address the needs of data organizations.



DataOps adds tests to the production pipeline to prevent errors from corrupting published analytics. If unit, functional or regression tests are written in a development tools environment different from production, then the tests cannot easily deploy. Popular workflow tools like Airflow, Control-M, or Azure Data Factory lack features that ease the migration of tests used in development into production tests that actively monitor and test your data flows. Seamlessly moving between development and production requires certain capabilities overlooked by typical DevOps tools, such as:

- Built-in connectors to the complex chain of data engineering, science, analytics, self-service, governance, and database tools.
- Meta-Orchestration or a 'DAG or DAGs'
- Integrated production testing and monitoring

A DataOps platform simplifies test deployment with Recipe tests. Recipes are orchestrated pipelines that easily migrate as a working unit between compatible environments. Tests can be written in a specific tool or, more portably, in the DataOps Platform domain. The DataKitchen DataOps Platform connects to your heterogeneous toolchains and provides a unified environment in which to implement tests. With a common testing environment, test authors don't have to be experts in each of the various tools and languages that comprise your end-to-end data operations pipelines. The DataOps Platform also enables tests to be added without having to write code; using a UI. For more information, view our webinar, Orchestrate Your Production Pipelines for Low Errors.

Complete Testing and Deployment Pipelines

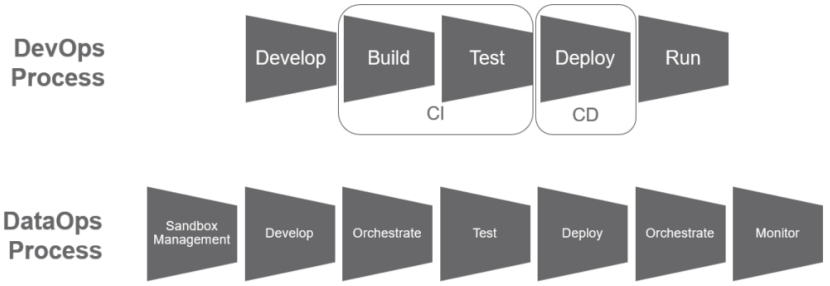
Deployment of data analytics differs from software engineering deployment. To deploy analytics from development through to production, you need four key components:

- The data you are using for testing
- The hardware/software stacks implementing analytics
- The code you have created
- All the tests that prove success

In DataOps, there are more steps and complexity than in a DevOps ‘build-test-deploy.’ (Figure 2)

A DataOps Platform enables you to extend the concept of ‘DevOps CI/CD’ to meet the needs of data science and analytics teams. It automates environment management, orchestration, testing, monitoring, governance and integration/deployment. Note that orchestration occurs twice in DataOps. DataOps orchestrates data operations and also development sandbox environments, which include a copy of data operations. DataOps executes these functions continuously:

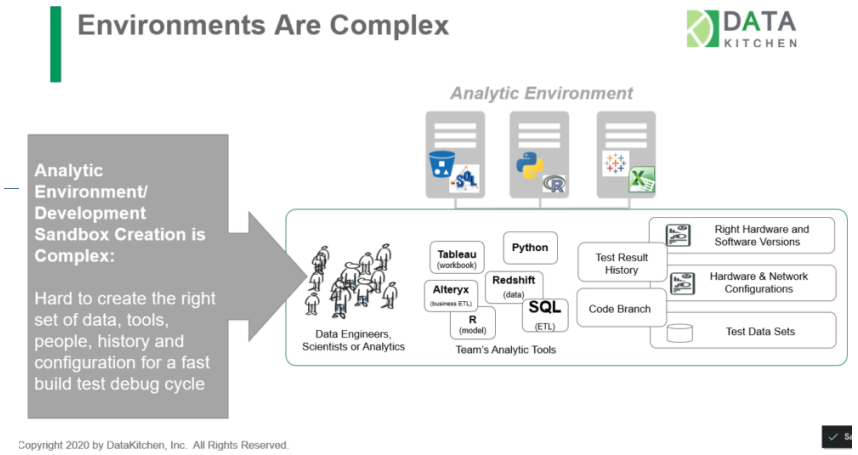
- CE (Continuous Environments)
- CO (Continuous Orchestration)
- CI / CD (Continuous Integration, Deployment)
- CT / CM / CG (Continuous Testing, Monitoring, Governance)



DevOps CI/CD tools like Jenkins or Azure Pipelines focus on the CI/CD segment of the development pipeline – the build and delivery of code. They orchestrate the software development tools, but not the data toolchains. For more information, view our webinar, Orchestrate Your Development Pipelines for Fast and Fearless Deployment.

Environment Pipelines

Compared to software development environments, creating analytic development environments is complex. The development of new data analytics requires the joining of test data, hardware-software environments, version control, toolchains, team organization, and process measurement. The DataKitchen DataOps Platform uses ‘Kitchens’ to abstract environments that contain all the pre-configured tools, datasets, machine resources, and tests – everything users need to create and innovate. Kitchens provide users with a controlled and secure space to develop new analytics and experiment.



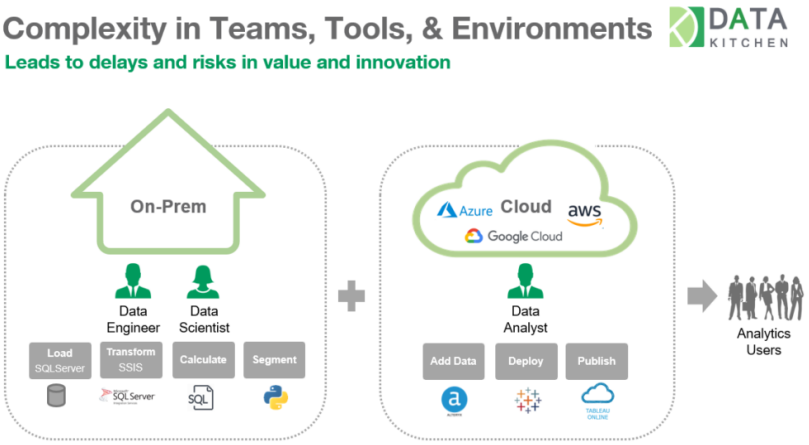
DevOps infrastructure tools like Puppet, Ansible, or Terraform lack support for self-service, analytics sandboxes. They do not offer:

- Test data management capabilities
- Integration to Git
- Simple wizards enabling data professionals to set up/shutdown/monitor sandboxes on-demand. Sandbox environments include test data, hardware-software environments, version control, and toolchains that are team, organization and process measurement aware

For more information on seamlessly spinning-up and managing repeatable analytics work environments that underpin your data production and development pipelines, see our webinar, Orchestrate Your Environment Pipelines for Reusability and Security.

Complex Team and Data Center Coordination

Data science, engineering, and analytics development and production work often take place across multiple organizations or multiple data centers (Figure 4). A DataOps Platform includes reusable analytics components and pipelines (Ingredients), permission-based access control to infrastructure and toolchains (Vaults), and a multi-agent architecture that facilitates seamless collaboration and coordination between teams working in different locations or environments.



DevOps or workflow tools lack the team and environment awareness necessary to promote reuse:

- A method to create reusable components, share them across the teams, and run the different elements together in a single process, with local control and centralized management and visibility.
- Ability to create, monitor, and remove Self-Service Data Science/Analytics Sandboxes on demand.

For more information on how to create a reliable and agile process for environment creation that balances analytics development productivity and governance with the needs of business users, see the webinar, Achieve Agility and Control with Self-Service Sandboxes.

Design for Data People

Building DataOps capabilities from scratch requires knowledge and management of 7-10 DevOps and workflow tools. This project may be a dream for some, but for others, it's an indescribable nightmare.
Data people are different from software people. Software engineers embrace complexity and celebrate each new tool. Your average software engineer enjoys unlocking the power of Git, Jenkins, AWS/Azure UI, workflow/DAG tools, testing frameworks, scripting languages, and infrastructure-as-code tools. Mastery of new tools represents career growth for a software developer.
Data professionals want to focus on analyzing data and creating models – learning new tools is a means to an end. Data people prefer a more straightforward set of abstractions and UI to deal with system complexity. While software engineers embrace toolchain complexity, data engineers and data scientists seek to avoid complexity. Self-service analytics users rarely venture outside the safe and ordered confines of a single preferred tool.

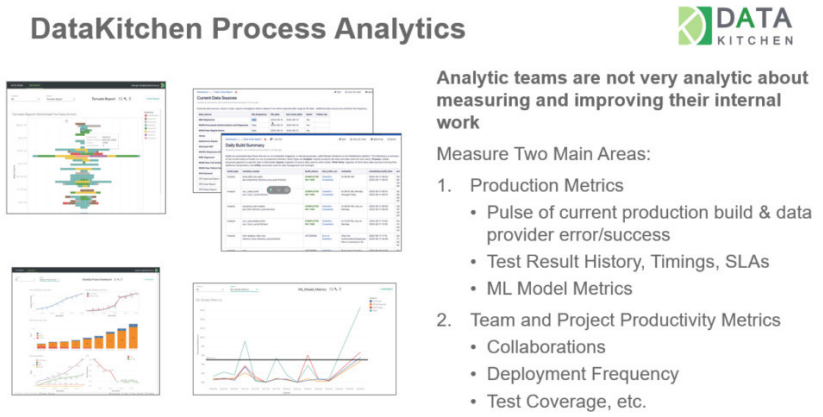
Forcing the data team to manage a complex toolchain combining 7-10 DevOps and workflow tools does not provide:

- A simple, straightforward user experience for users that wish to avoid complexity
- The ability to support a full spectrum of users, from the most technical data engineer to the most non-technical self-service user, and enable them to carry out their role in DataOps.

A DataOps Platform brings all of the tools necessary for DataOps into a single coherent platform that allows each user to interact with the toolchain at the level of simplicity or complexity that they desire.

DataOps Process Metrics

DataOps strives to eliminate waste, and DataOps metrics serve as the means to demonstrate process improvements (Figure 5). Teams need to be measured on cycle times, error rates, productivity, test coverage, and collaboration. Metrics drive teams to work in a more agile, iterative, and customer-focused manner. A DataOps Platform saves the run information from every Recipe execution to provide those critical metrics.



DevOps tools lack explicit capabilities to create and store metrics, including:

- Process lineage: An integrated store of history of all processing, including test results, timing, and code/configuration.
- DataOps process analytics reports and datamart.

Build vs. Buy?

Despite the difficulties discussed above, some data teams still consider implementing DataOps from scratch by building a system from a dozen DevOps and workflow software components. These endeavors nearly always stumble. Developing, debugging and maintaining a DataOps system itself tends to become a major bottleneck and source of unplanned work. A team might begin with big dreams but stop with a few unit tests in development, a manually-tended Jenkins deploy, and lots of glue code on Airflow, or Talend, or data factory – falling far short of a broad and robust DataOps implementation. The software industry has coined the terms “wagile” or “scrum-fall” to describe a failure to properly apply the Agile Manifesto principles. Ultimately what matters to data teams is rapid deployment, reducing errors, end-to-end collaboration, and process metrics. The faster a team advances to focusing on these process improvements, the more likely it will sustain momentum and attain DataOps excellence. Adopting a purpose-built DataOps Platform from DataKitchen can save your team significant time and frustration (Figure 6).

In competitive markets, the most innovative companies will be those that can quickly adapt to rapidly evolving market conditions. The data teams that adopt DataOps and produce robust and accurate analytics more quickly than their peers will power strategic decision-making that sustains a competitive advantage. The DataKitchen DataOps Platform can jumpstart your DataOps initiative and more quickly bring the benefits of rapid and robust analytics to your users and decision-makers.

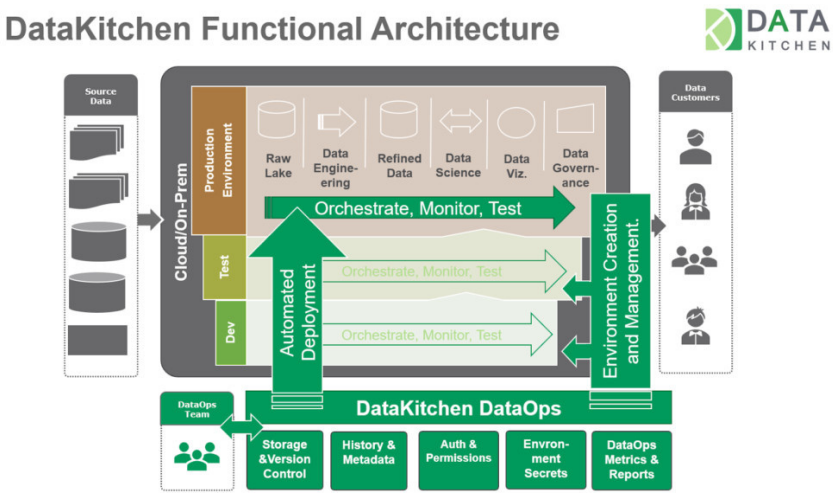
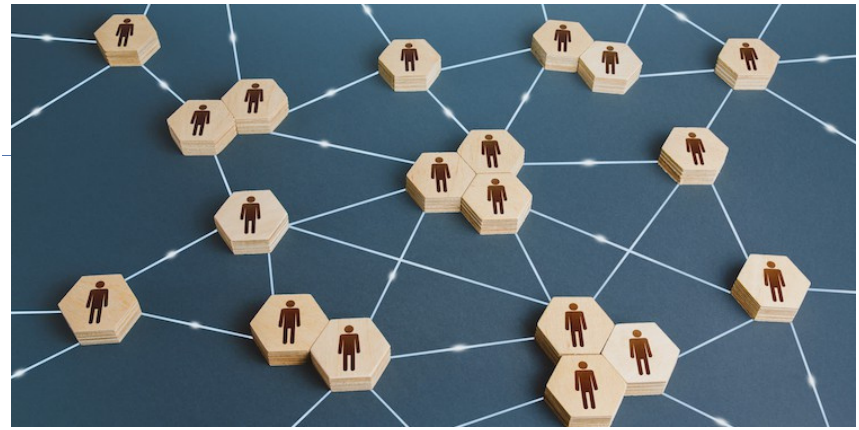


Figure 6: The DataKitchen DataOps Platform eliminates waste from the processes that manage integration, development and operation

Comparison of Approaches for DataOps Implementation

DataOps Process	DevOps Tools	DataOps Platform
End-to-End Production Pipelines		
Orchestration	x	x
• Meta-Orchestration		x
• Connectors to your complex data analytics toolchain		x
• Integrated testing and monitoring		x
Development Pipelines		
• Continuous Integration/Deployment (CI/CD)	x	x
• Continuous Meta-Orchestration (CO)		x
• Continuous Environments (CE)		x
• Continuous Testing and Monitoring (CT/CM)		x
• Continuous Governance (CG)		x
Environment Pipelines		
• Test data management capabilities		x
• Git integration		x
• On-demand infrastructure		x
Collaboration and Coordination		
• Ability to create, share, and run reusable components		x
• Meta-orchestration across environments, teams, locations		x
• Single process with local control, centralized management, and end-to-end visibility		x
• Ability to create, monitor and remove analytic sandboxes on-demand		x
User Interface/Complexity		
• Designed for data professionals and non-technical people		x
Process Metrics		
• Process lineage		x
• Process analytic reports		x
• Datamart		x

What is a Data Mesh?



The data mesh design pattern breaks giant, monolithic enterprise data architectures into subsystems or domains, each managed by a dedicated team. With an architecture comprised of numerous domains, enterprises need to manage order-of-operations issues, inter-domain communication, and shared services like environment creation and meta-orchestration. A DataOps superstructure provides the foundation to address the many challenges inherent in operating a group of interdependent domains. DataOps helps the data mesh deliver greater business agility by enabling decentralized domains to work in concert.

This post (1 of 5) is the beginning of a series that explores the benefits and challenges of implementing a data mesh and reviews lessons learned from a pharmaceutical industry data mesh example. But first, let's define the data mesh design pattern.

The past decades of enterprise data platform architectures can be summarized in 69 words. Note, this is based on a post by Zhamak Dehghani of Thoughtworks.

- **First-generation** – expensive, proprietary enterprise data warehouse and business intelligence platforms maintained by a specialized team drowning in technical debt.
- **Second-generation** – gigantic, complex data lake maintained by a specialized team drowning in technical debt.
- **Third-generation** – more or less like the previous generation but with streaming data, cloud, machine learning and other (fill-in-the-blank) fancy tools. And you guessed it, managed by a specialized team drowning in technical debt.

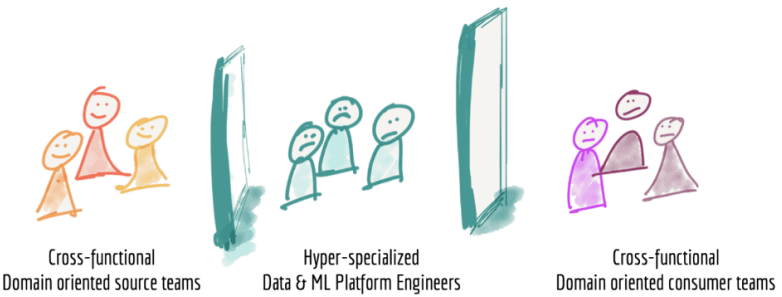
See the pattern? It's no fun working in data analytics/science when you are the bottleneck in your company's business processes. A barrage of errors, missed deadlines, and slow response time can overshadow the contributions of even the most brilliant data scientist or engineer. The problem is not "you." It lies somewhere in between your enterprise data platform architecture and the enterprise's business processes. Below are some reflections upon the failure of modern enterprise architectures to deliver on data analytics agility and reliability:

- Centralized Systems Fail – Lots of inputs and outputs; numerous fragile pipelines; hard to understand, modify, monitor, govern; billions of dollars have been (and will be) invested in vain trying to cope with this vast complexity.

- Skill-based roles cannot rapidly respond to customer requests – Imagine a project where different parts are written in Java, Scala, and Python. Not everyone has all these skills, so there are bound to be bottlenecks centering on certain people. Data professionals are not perfectly interchangeable. Further, the teams of specialized data engineers who build and maintain enterprise data platforms operate as a centralized unit divorced from the business units that create and consume the data. The communication between business units and data professionals is usually incomplete and inconsistent. Centralized enterprise data architectures are not built to support Agile development. Data teams have great difficulty responding to user requests in reasonable time frames. Figure 1 illustrates the existence of barriers between data source teams, the platform team and data consumers.

- Data domain knowledge matters – The data team translates high-level requirements from users and stakeholders into a data architecture that produces meaningful and accurate analytics. This is much easier to do when the data team has intimate knowledge of the data being consumed and how it applies to specific business use cases. A schema designer who moves from project to project may not understand the nuances behind the requirements of each data consumer.

- Universal, one size fits all patterns fail – Data teams may fall into the trap of assuming that one overarching pattern can cover every use case. We've seen many data and analytic projects, and mistaken assumptions like this one are a common cause of project underperformance or failure.



Introduction to Data Mesh

The data mesh addresses the problems characteristic of large, complex, monolithic data architectures by dividing the system into discrete domains that are managed by smaller, cross-functional teams. Data mesh proponents borrow the term "domain" from the software engineering concept of "domain-driven design (DDD)," a term coined by Eric Evans. DDD divides a system or model into smaller subsystems called domains. Each domain is an independently deployable cluster of related microservices which communicate with users or other domains through modular interfaces. Each domain has an important job to do and a dedicated team – five to nine members – who develop an intimate knowledge of data sources, data consumers and functional nuances. The domain team adopts an entrepreneurial mindset, viewing the domain as if it were a *product* and users or other domains as *customers*. The domain includes data, code, workflows, a team, and a technical environment. Data mesh applies DDD principles, proven in software development, to data analytics. *If you've been following DataKitchen at all, you know we are all about transferring software development methods to data analytics.*

The organizational concepts behind data mesh are summarized as follows.

- A five to nine-person team owns the dev, test, deployment, monitoring and maintenance of a domain.
- The team organizes around the domain, not the underlying toolchain or horizontal data pipelines.
- Domain data assets, artifacts and related services are viewed as the team's product. The product includes data and operations. Data consumers are the domain team's customers.
- Data Engineers must develop an intimate understanding of data sets to really add value.

Benefits of a Domain

A well-implemented domain has certain attributes that offer benefits to the customers or domain users:

- **Trusted** – Users wish to be confident that the data and artifacts are correct. Trust must be earned, which is why it is so important for a domain to have interfaces that enable introspection and access. Users should be able to ask the domain questions and get answers.
- **Usable by the teams' customers** – Customer experience improves by virtue of having a dedicated team that attains intimate knowledge of the data and its use cases. Also, the domain must support the attributes that are part of every modern data architecture.

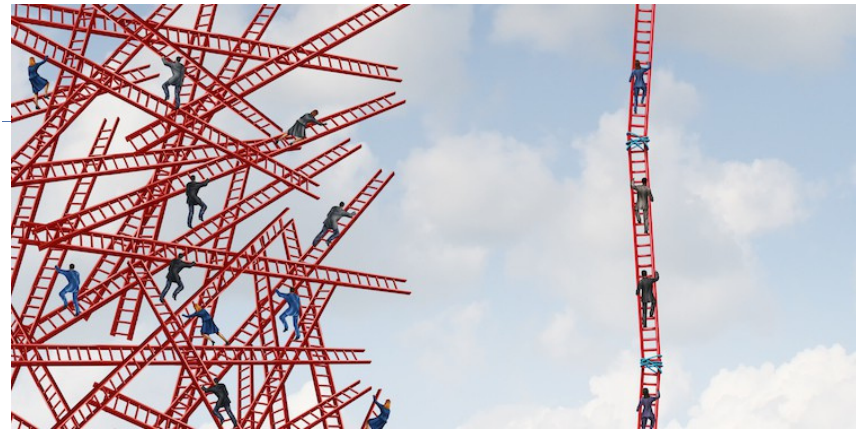
    - Discoverable – users have access to a catalog or metadata management tool which renders the domain discoverable and accessible.
    - Understandable and well-described – terms are defined in a dictionary and the domain has clean, well-designed interfaces.
    - Secure and permissioned – data is protected from unauthorized users.
    - Governed – designed with data quality and management workflows that empower data usage.
    - URL/API Driven – can easily interoperate with other domains through a hyperlink (URL) or application programming interface (API).

- **Clear accountability** – users interact with a responsive, dedicated team that is accountable to them.

    - Easy to report problems and receive updates on fixes
    - Users may request new insights/improvements and get them into production quickly.

Organizing a data architecture into domains is a first-order decision that drives organizational structure, incentives and workflows that influence how data consumers use data. Domains change the focus from the data itself to the use cases for the data. The customer use cases in turn drive the domain team to focus on services, service level agreements (SLA) and APIs. Domains promote data decentralization. Instead of relying on a centralized team, where the fungibility of human resources is an endless challenge, the domain team is dedicated and focused on a specific set of problems and data sets. Decentralization promotes creativity and empowerment. In the software industry, there's an adage, "you build it, you run it." Clear ownership and accountability keep the data teams focused on making their customers successful. As an organizing principle, domains favor a decentralized ecosystem of interdependent data products versus a centralized data lake or data warehouse with all its inherent bottlenecks.

Use DataOps With Your Data Mesh to Prevent Data Mush



In our last post, we summarized the thinking behind the data mesh design pattern. In this post (2 of 5), we will review some of the ideas behind data mesh, take a functional look at data mesh and discuss some of the challenges of decentralized enterprise architectures like data mesh. Last we'll explore how DataOps can be paired with data mesh to mitigate these challenges.

Large, centralized enterprise architectures discourage agility. The centralized architecture consists of various pipelines that ingest, process and serve data. Everyone on the data team works on their specialized part of the larger system, and they rely upon emails, documents and meetings to stay organized. Figure 1 shows the strained lines of communications that cross-functional boundaries. Roles tend to be more specialized on a big team, so there are bottlenecks. With heavyweight processes, it becomes much more bureaucratic to change any part of the architecture. With a steady stream of requests for new data sources and new analytics, the centralized team managing the platform can quickly exceed their capacity to keep up. Customers are on a journey to get insight, and they may not know exactly what they want until they see it. With large systems, it's much harder to iterate toward a solution that addresses a user's latent requirements.

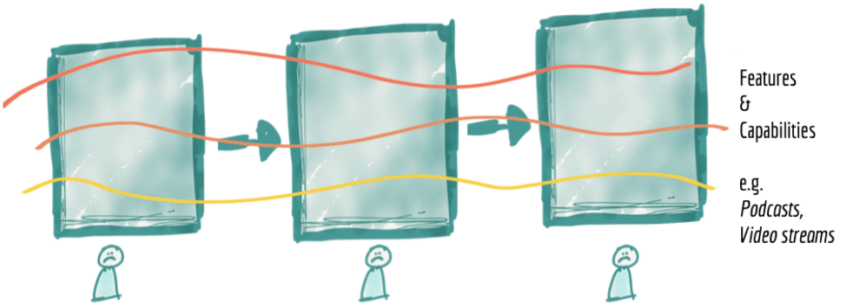


Figure 1: When you make a change to a centralized platform, you need to update each component and coordinate between several different teams. Source: Thoughtworks

Data mesh decouples the subsystem teams from each other. By partitioning the system into pieces, each domain team can work uninterrupted and at their natural iteration cadence. Over time, domain teams attain a much more intimate understanding of their data sources and uses, leading to more effective technical solutions. The data team addresses the backlog of tasks more quickly because task coordination is more straightforward with a smaller team. The data mesh team assumes total lifecycle ownership of the domain. The domain is their internal "product," and the domain product manager is their empowered mini-CEO – you may have heard the term "product thinking" with respect to a data mesh. Success means excellently serving the needs of internal customers. Team members cover all the roles, so there are more hybrid players enabling more flexibility in responding to surges in demand for a particular skill set. When something goes wrong, the domain team has all the right incentives to iterate on a solution.

Data Mesh Architecture Example

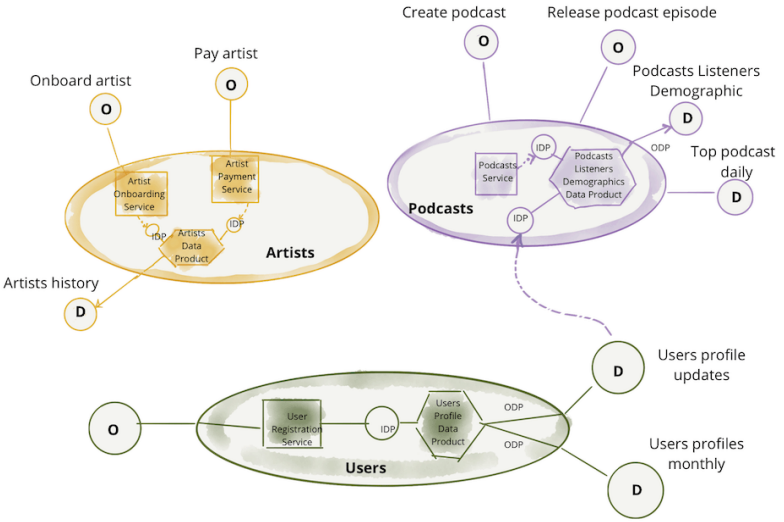
The concept of data mesh, proposed by Zhamak Dehghani, has taken the industry by storm. Below we'll look at a simple example architecture partitioned into domains to illustrate data mesh.

You may have used a media streaming application such as Spotify or SoundCloud. A streaming service has to handle a variety of activities that logically partition into different areas:

- Artists – onboard, pay, manage, …
- Podcasts – create, release, play, …
- Users – register, manage profiles, manage access, …

A centralized enterprise application would combine all of these functions and services into one monolithic architecture. A data mesh organizes each functional domain as a separate set of services. In figure 2 below, three domains are shown. A dedicated team handles artists, podcasts and users respectively, but each domain depends on data and services from other domains and operational systems. The domains in the figure are shown receiving data from input data ports (IDP) and transmitting data to output data ports (ODP).

The act of onboarding a new artist activates an artist onboarding service that updates the artists domain. When a user plays a podcast (within the podcast domain), it triggers an input into the artist's domain, which activates the artist payment service. Application scalability improves as each group independently manages its domain. For example, the users domain team could add a new data set or new microservices related to users data with complete autonomy.



Challenges Related to Data Mesh

In data analytics, there is a constant tension between centralization and decentralization. Data democratization and self-service analytics empower innovation, but the enterprise must adhere to strict governance standards. We need one version of reality, but over-centralizing analytics creates bottlenecks.

Data mesh leans the enterprise hard in the direction of decentralization. The benefits of granting autonomy to small teams are well established. The domain team develops workflows and practices aligned with user requirements and their own team culture. A group of five to nine people is large enough to vet ideas but not so large as to stifle innovation. The team develops an intimate understanding of their data and its application within the greater data organization. Domains foster a sense of ownership, accountability and an alignment of incentives.

The disadvantages of decentralization are also well known. Greater decentralization raises the overall level of chaos. Decentralization makes it harder to enforce governance, security and other policies. Decentralization may not adequately manage intellectual property. The table below summarizes the advantages and disadvantages of a decentralized organizational structure.

DataOps is the Factory that Supports Your Data Mesh

Below is our final post (5 of 5) on combining data mesh with DataOps to foster innovation while addressing the challenges of a data mesh decentralized architecture. We see a DataOps process hub like the DataKitchen Platform playing a central supporting role in successfully implementing a data mesh. DataOps excels at the type of workflow automation that can coordinate interdependent domains, manage order-of-operations issues and handle inter-domain communication. The following section will explore the DataOps-enabled data mesh in more depth.

It would be incredibly inefficient to build a data mesh without automation. DataOps focuses on automating data analytics workflows to enable rapid innovation with low error rates. It also engenders collaboration across complex sets of people, technology, and environments. DataOps produces clear measurement and monitoring of the end-to-end analytics pipelines starting with data sources. While data mesh concerns itself with architecture and team alignment, DataOps automates workflows that simplify data mesh development and operations. DataOps is also an ideal platform upon which to build the shared infrastructure needed by the data mesh domains. For example, DataOps provides a way to instantiate self-service development environments. It would be a waste of effort for each domain to create and maintain that capability independently.

A data mesh implemented on a DataOps process hub, like the DataKitchen Platform, can avoid the bottlenecks characteristic of large, monolithic enterprise data architectures. One can implement DataOps on its own without data mesh, but data mesh is a powerful organizing principle for architecture design and an excellent fit for DataOps-enabled organizations. DataOps is the scaffolding and connective tissue that helps you construct and connect a data mesh.

A discussion of DataOps moves the focus away from organization and domains and considers one of the most important questions facing data organizations – a question that rarely gets asked. *"How do you build a data factory?"* The data factory takes inputs in the form of raw data and produces outputs in the form of charts, graphs and views. The data factory encompasses all of the domains in a system architecture. Most enterprises rush to create analytics before considering the workflows that will improve and monitor analytics throughout their deployment lifecycle. It's no wonder that data teams are drowning in technical debt.

Before you can run your data factory, you have to build your data factory. Before you build your factory, you would do well to design mechanisms that create and manage your data factory. *DataOps is the factory that builds, maintains and monitors data mesh domains.*

Architecture, uptime, response time, key performance parameters – these are challenging problems, and so they tend to take up all the oxygen in the room. Take a broader view. Architect your data factory so that your data scientists lead the industry in cycle time. Design your data analytics workflows with tests at every stage of processing so that errors are virtually zero in number. Doing so will give you the agility that your data organization needs to cope with new analytics requirements. Agile analytics will help your data teams realize the full benefits of an application and data architecture divided into domains. Efficient workflows are an essential component of a successful data team initiative. One common problem is that code changes, or new data sets may break existing code. Your domain DAG (directed acyclic graph) is ingesting data and then transforming, modeling and visualizing it. It's hard enough to test within a single domain, but imagine testing with other domains which use different teams and toolchains, managed in other locations. How do you allow a local change to a domain without sacrificing global governance and control? That's important to do, not only within a single domain but between a group of interdependent domains as well.

With a DataOps process hub, like the DataKitchen Platform, testing is performed in an environment that mirrors the system. DataKitchen supports an intelligent, test-informed, system-wide production orchestration (meta-orchestration) that spans toolchains. Unlike orchestration tools like Airflow, Azure Data Factory or Control-M, DataKitchen can natively connect to the complex chain of data engineering, science, analytics, self-service, governance and database tools, and meta-orchestrates a hierarchy of DAGs. Meta-orchestration in a heterogeneous tools world is a critical component to successfully rolling out a data mesh.

The DataKitchen Platform natively provides URL access to nearly all the interfaces that are required for inter-domain communication.

Inter-Domain Communication	Question / Steps Asked	DataKitchen Support
Domain Query	"When was the last time you were updated?" Successful or failure? Warnings? "Is the data or artifacts in your domain good? Can you prove it with some test results?"	√
Process Linkage	"Ok, you start. I am done." "Ok, you start. I am done, *and* here are a bunch of parameters you need to keep going."	√
Event Linkage	"Here is an event: e.g., processing completed, error, warnings, etc."	√
Data Linkage	"We share a common table (e.g., a dimension table) in our domain."	Link to 3rd Party Tools
Development Linkage	"Can I re-create your domain in development?" "Can I see the code you used to create it?" "Can I modify that code in development?" "Is there a path to production?"	√

Table 1: DataKitchen Platform support for inter-domain communication.

The capability to execute domain queries comes from DataKitchen order runs. Process queries come from calling an order run or running a Recipe (orchestration pipeline). Event linkage is natively supported, and development linkage stems from Kitchens (on-demand development sandboxes). DataKitchen supports data linkage by integrating with tools that access and store data – there are plenty of great ones.

DataKitchen groups recipes into components called ingredients. Ingredients are composable units that can enable domains to change independently. They can also be orchestrated together within Kitchens.

DataKitchen can help you address the critical tasks of developing, deploying, and monitoring analytics related to a domain.

- Recipes – Orchestrate the whole pipeline (i.e., ingest, process/curate, serve, etc.) inside a domain. DataKitchen Recipes can serve as a master DAG as well as lower-level DAGs nested inside other DAGs.
- Monitoring Tests – Make sure the data (from suppliers and to customers) is trustworthy this aids in diagnostics (i.e., detection and localization of an issue). In the DataKitchen context, monitoring and functional tests use the same code.
- Variations – Execute data pipelines with specific parameters. Deliver the correct data product for the consumer because one size does not fit all. Variations enable the data team to create different versions of their domain to handle development, production, "canary" versions or any other change. Variations unlock a great degree of agility and enable people to be highly productive.

- **Kitchen Wizard** – Provide on-demand infrastructure to the data teams to prevent delays. One concern related to domain teams is the potential duplication of effort with respect to horizontal infrastructure. DataKitchen can be used to automate shared services. For example, DataKitchen provides a mechanism to create self-service development sandboxes, so the individual teams do not have to develop and support this capability.
- **Kitchens/Recipes/Functional** Tests – Iterate to support the “product-oriented” approach. Building the factory that creates analytics with minimal cycle time is a critical enabler for the customer focus that is essential for successful domain teams.

CONCLUSION

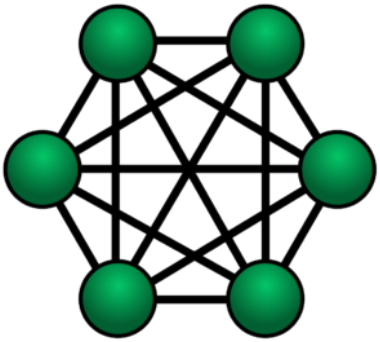
Data mesh is a powerful new paradigm which deals with the complexity in giant, monolithic data systems. As an organizing principle, it focuses on data, architecture and teams and less, the operational processes that are critically important for agile, error-free analytics. As part of your data mesh strategy, DataOps assists with the process and workflow aspects of data mesh. DataOps automates shared services preventing duplication of effort among teams. DataOps also addresses some of the complexity associated with domain inter-dependencies and enables the data organization to strike the right balance between central control/governance and local domain independence.

DataOps is the Factory that Supports Your Data Mesh

Industry analysts who follow the data and analytics industry tell DataKitchen that they are receiving inquiries about “data fabrics” from enterprise clients on a near-daily basis. Forrester relates that out of 25,000 reports published by the firm last year, the report on data fabrics and DataOps ranked in the top ten for downloads in 2020. Gartner included data fabrics in their top ten trends for data and analytics in 2019. From an industry perspective, the topic of data fabrics is on fire.

What is a Data Fabric?

Whenever a new technology or architecture gains momentum, vendors hijack it for their own marketing purposes. This is happening to the term “data fabric.” Tools vendors are creating their own definitions of “data fabric” to promote their own product and solution offerings. If you search the Internet for a definition of data fabrics you can see discussions of storage, AI augmentation, and other tools. It is also difficult to understand a new technology when people discuss it in terms of its benefits. For example, one could say that data fabrics address complexity, eliminate silos, unify toolchains and help organizations derive insights. That’s all justifiable, but the same things could be said of other technologies, so it doesn’t actually explain what data fabrics are and what they do.
Data organizations are buckling under the strain of numerous data pipelines acting on large, complex, and distributed data sets. Data fabrics purport to offer a unified approach to manage the cacophony of heterogeneous toolchains being thrown at data problems. Data fabrics are best regarded as an architecture or design concept, not a specific set of tools.



The term fabric gets its name from an architectural approach that provides full point-to-point connectivity between nodes. In Figure 1, the nodes could be sources of data, storage, internal/external applications, users – anything that accesses or relates to data. Data fabrics provide reusable services that span data integration, access, transformation, modeling, visualization, governance, and delivery. In order to deliver connectivity between all these different services, data fabrics must include connectors to data ecosystem tools.

A cynic could argue that data fabrics retrace many points from the discussion on data virtualization that began a decade ago. Data fabric enthusiasts assert that the design pattern is much more than that and reference one or more emerging data analytics tools: AI augmentation, automation, orchestration, semantic knowledge graphs, self-service, streaming data, composable data analytics, dynamic discovery, observability, persistence layer, caching and more. Data fabrics seek to harmonize all of these diverse technologies and tools – which ones depend on who is doing the talking. Some thought leaders are saying that enterprises need to implement data fabrics in order to work in an agile, customer-focused way. Data fabrics can add value, but they are not a panacea. It's important to understand that migrating to a data fabric is not guaranteed to improve business agility.

Process Constraints

Let's use an analogy. When you drive down Massachusetts Avenue in Cambridge, Massachusetts you hit a stoplight every few tenths of a mile. This goes on for miles. If a Ferrari Enzo and a Honda Civic raced from Cambridge to Arlington during rush hour, stopping at stoplights and observing traffic laws, it's a fair bet that they would arrive at around the same time. Maybe the Ferrari would zip ahead at times, but between stoplights and long lines of traffic, the inherent advantages in the Ferrari design would be handicapped or perhaps even totally neutralized.

The point is that analytics agility is more about removing obstacles and process bottlenecks than optimized tool performance. The people that data organizations hire and the tools that they buy are Ferraris. Then they place them on busy boulevards wading through long lines of traffic, waiting at stoplights, and force them to observe speed limits. Data fabrics are the latest example of putting a Ferrari on Massachusetts Ave.

The stoplights in data analytics are all of the obstacles that degrade productivity and agility: data errors, manual processes, impact review, lack of coordination, and general bureaucracy. An organization that takes seven weeks to create an analytics development environment will still take seven weeks, even after they spend millions of dollars implementing a data fabric. Tools are sexy, but you know what is really sexy? Turning data analytics into a sustainable competitive advantage because your data team can go really fast while incurring virtually zero errors. How do you create the most agile data analytics team in your industry? You start by looking at your processes and workflows.

Start with a DataOps Process Fabric

Data fabrics can be helpful, but we think it is much more effective to start with a "process fabric." A process fabric begins with a DataOps superstructure that connects with all your existing tools and orchestrates your data pipelines and workflows (Figure 2). A DataOps process fabric connects your people, processes, and tools together in a giant web of interconnectedness. Once you have that in place, you can start alleviating bottlenecks. If you have data errors that drive unplanned work, then orchestrate a battery of statistical and process controls that qualify data sources and data processing. If it takes weeks to create development environments, then use automation and orchestrate the creation of environments on-demand. Take control of your data factory with process orchestration. With fewer traffic lights, your Ferrari tools and people will accelerate into an open road.

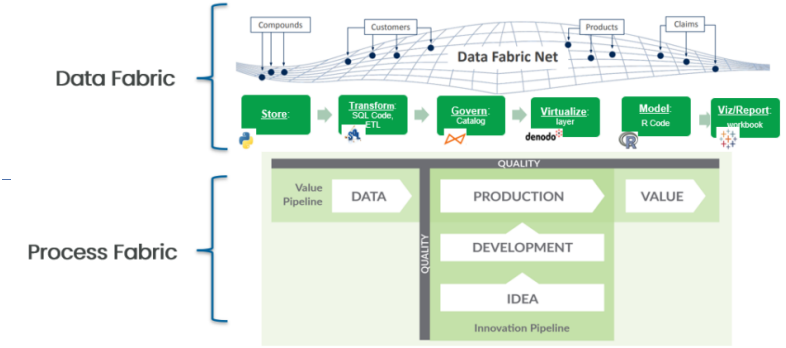


Figure 2: A process fabric begins with a DataOps superstructure that connects your existing tools and orchestrates your data pipelines and workflows.

A DataOps process fabric unifies your various toolchains into a hierarchy of observable orchestrations that we call meta-orchestration. Once that is done, your teams can focus on their specific local orchestrations without struggling with all of the task coordination that makes inter-team collaboration challenging. A process fabric unifies your existing tools and provides the platform on which you can iterate toward alleviating bottlenecks. You may find that technologies like data virtualization, augmented data catalogs, or some other components associated with data fabrics are helpful. The DataOps process fabric gives you control over your technology roadmap so you can evolve incrementally in ways that make sense given your goals and constraints. While some people see data fabrics as an ideal path to DataOps, we think the opposite is more compelling. A DataOps process fabric is a great way to begin a smooth evolution towards a data fabric.

The DataKitchen DataOps Platform provides the toolchain superstructure, meta-orchestration, observability, and environment automation that can help you identify and address process and workflow bottlenecks. If analytics agility is the goal, then process-focused optimization should drive tools decisions and a DataOps process fabric should drive evolution to data fabrics or other technologies that put your Ferrari data team on the fast track.

DataOps Triple Chocolate Peanut Butter Cookies

by Aarthy Kannan Adityan

INGREDIENTS AND TOOLS

- 1 cup butter
- 3/4 cup brown sugar
- 3/4 cup white sugar
- 1 1/2 teaspoon vanilla extract
- 3/4 cup chocolate peanut butter
- 2 eggs
- 2 1/3 cup flour
- 1 teaspoon baking soda
- 3/4 cup cocoa powder
- 1 cup semi-sweet chocolate chips
- 1 cup peanut butter chips

INSTRUCTIONS

1. Preheat oven to 350ºF
2. Soften butter to room temperature
3. Line a baking sheet with parchment paper
4. In a large bowl, cream together softened butter, brown sugar and white sugar
5. Add vanilla extract, chocolate peanut butter and eggs and mix well
6. Stir in flour, baking soda and cocoa powder and combine until blended
7. Fold chocolate chips and peanut butter chips into batter
8. Scoop batter onto prepared baking sheet using a cookie or ice-cream scoop, leaving enough space in-between for cookies to expand
9. Bake for 14-16 minutes
10. Transfer cookies to a wire rack to cool

Serving size: about 20 cookies

DataOps Examples and Case Studies

Grow Sales Using a DataOps-Powered Customer Data Platform

Data analytics can help drive corporate growth by providing customer analytics and ultimately actionable insights to the sales and marketing teams. Unfortunately, the fast-paced, dynamic nature of sales makes it difficult for the customer-facing teams to tolerate the slow and deliberate manner in which analytics is typically produced. In an earlier chapter, we identified eight major challenges of data analytics:

- **The Goalposts Keep Moving –** Sales and marketing requirements change constantly and the requests for new analytics never cease.
- **Data Lives in Silos –** Data is collected in separate operational systems and typically, none of these systems talk to each other.
- **Data Formats are not Optimized** – Data in operational systems is usually not structured in a way that lends itself to the efficient creation of analytics.
- **Data Errors** – Data will eventually contain errors, which can be difficult to resolve quickly.
- **Bad Data Ruins Good Reports** – When data errors work their way through the data pipeline into published analytics, internal stakeholders can become dissatisfied. These errors also harm the hard-won trust in the analytics team.
- **Data Pipeline Maintenance Never Ends** – Every new or updated data source, schema enhancement, analytics improvement or other change triggers an update to the data pipeline. These updates may be consuming 80% of your team's time.
- **Manual Process Fatigue** – Manual procedures for data integration, cleansing, transformation, quality assurance and deployment of new analytics are error-prone, time-consuming and tedious.
- **The Trap of "**Hope and Heroism**"** – To cope with the above challenges, data professionals work long hours, make changes (without proper testing) and "hope" for the best or just retreat into a posture of over-caution in which projects just execute more slowly.

OVERCOMING THE EIGHT CHALLENGES

If you have managed an analytics team for any period of time, you have likely encountered these and similar challenges. However, you don't have to accept the status quo. It is possible to implement processes and methodologies that address these challenges and enable your data-analytics team to improve their productivity by an order of magnitude while achieving a higher level of data quality. In this new approach to customer and market analytics, the data-analytics team executes at previously unimaginable speed, efficiency and quality:

Rapid-Response Analytics – The sales and marketing team will continue to demand a never-ending stream of new and changing requirements, but the data-analytics team will delight your sales and marketing colleagues with rapid responses to their requests. New analytics will inspire new questions that will, in turn, drive new requirements for analytics. The feedback loop between analytics and sales/marketing will iterate so quickly that it will infuse excitement and creativity throughout the organization. This will lead to breakthroughs that vault the company to a leadership position in its markets.

Data Under Your Control – Data from all of the various internal and external sources will be integrated into a consolidated database that is under the control of the data-analytics team. Your team will have complete access to it at all times, and they will manage it independently of IT, using their preferred tools. With data under its control, the data-analytics team can modify the format and architecture of data to meet its own operational requirements.

Impeccable Data Quality – As data flows through the data-analytics pipeline, it will pass through tests and filters that ensure that it meets quality guidelines. Data will be monitored for anomalies 24x7, preventing bad data from ever reaching sales and marketing analytics. You'll have a dashboard providing visibility into your data pipeline with metrics that delineate problematic data sources or other issues. When an issue occurs, the system alerts the appropriate member of your team who can then fix the problem before it ever receives visibility. As the manager of the data-analytics team, you'll spend far less time in uncomfortable meetings discussing issues and anomalies related to analytics.

Automated Efficiency – Data feeds and new analytics will be deployed using automation, freeing the data-analytics team from tedious manual processes. The analytics team will be able to focus on its highest priorities — creating new analytics for sales and marketing that create value for the company.

The processes, methodologies and tools required to realize these efficiencies combine two powerful ideas: The Customer Data Platform (CDP) and a revolutionary new approach to analytics called DataOps. **Below we'll explain how you can implement your own DataOps-powered CDP that improves both your analytics cycle time and data-pipeline quality by 10X or more.**

CUSTOMER DATA PLATFORM

A Customer Data Platform (CDP) provides sales and marketing with a unified view of all cus- tomer-related data whether internal or external, in a single integrated database. Once setup, a CDP enables the analytics team to create and manage customer data themselves, without reliance upon resources from IT or other departments. This helps sales and marketing better leverage the company's valuable data while responding to market demands quickly and proactively. The figure below shows how a CDP consolidates data from numerous databases. Each operational database becomes a data source that continuously feeds a copy of its data into a centralized CDP database.

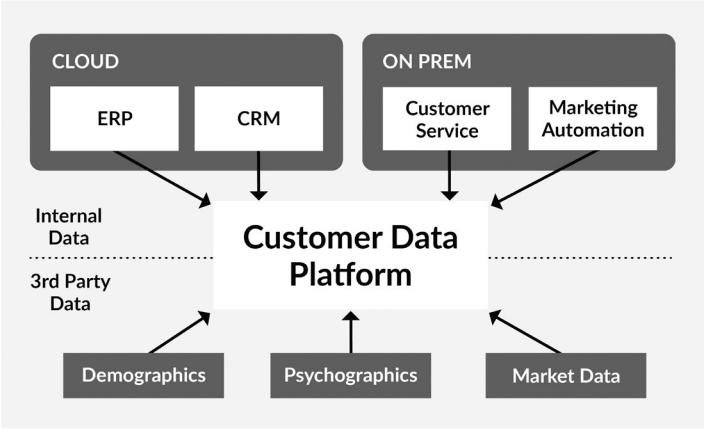


Figure 95: The Customer Data Platform consolidates data from operational systems to provide a unified customer view for sales and marketing.

DATAOPS

A CDP is a step in the right direction, but it won't provide much improvement in team productivity if the team relies on cumbersome processes and procedures to create analytics. DataOps is a set of methodologies and tools that will help you optimize the processes by

which you create analytics, manage the data-analytics pipeline and automatically deploy new analytics and data. DataOps rests on three foundational principles:

Agile Development – DataOps utilizes a methodology called Agile Development to minimize the cycle time for new data analytics. Studies show that software development projects using Agile complete significantly faster and with far fewer defects.

DevOps – In DataOps, new analytics production is automated and monitored. Automated tests verify new analytics before publishing them to sales and marketing users. This allows the analysts to focus less on the mechanics of deploying analytics and more on the creation of new insights that address sales and marketing requests. In the software development domain, the automated deployment of code is called DevOps. Prominent software industry leaders use DevOps to publish software updates many times per second while assuring quality. DataOps incorporates DevOps methods and principles to publish new analytics and data in an automated fashion.

Statistical Process Control – DataOps employs a methodology called Statistical Process Control (SPC) to assure data quality using end-to-end data pipeline automation and quality controls. SPC is a lean manufacturing method that institutes continuous testing on data flowing from sources to users, ensuring that data stays within statistical limits and remains consistent with business logic. SPC monitors data and verifies it 24x7. If an anomaly occurs, SPC notifies the data-analytics team via an automated alert. This reduces the operational burden on team members while improving data quality and reliability. Also, the quantity of data and the number of data sources can more easily scale independently of the size of the data engineering team.

When implemented in concert, Agile, DevOps and SPC take the productivity of data-analytics professionals to a whole new level. DataOps will help you get the most out of your data, human resources and integrated CDP database.

Achieving Growth Targets by Implementing a DataOps-Powered Customer Data Platform

As the leader of a data-analytics organization, your mission is to utilize data from operational systems and other sources to create insights that help the organization achieve its growth targets. Figure 96 provides a conceptual view of this flow. The dark boxes show the domain that is under the control of the analytics leader. It includes people, tools, technologies and processes that together comprise the DataOps-powered CDP.

The Eight Challenges of Data Analytics

1. The Goalposts Keep Moving
2. Data Lives in Silos
3. Data Formats are not Optimized
4. Data Errors
5. Bad Data Ruins Good Reports
6. Data Pipeline Maintenance Never Ends 7. Manual Process Fatigue
8. The Trap of Hope and Heroism

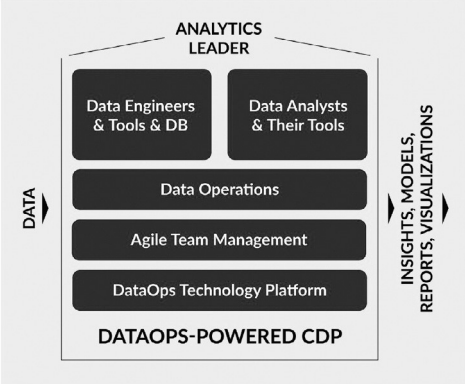


Figure 96: The resources under data-analytics control that leverage data to meet business objectives

DATA ANALYSTS AND THEIR TOOLS

The Data Analyst works to satisfy the needs of the sales and marketing department by continually delivering insights. The data analyst creates visual representations of data to communicate information in a way that leads to insights either on an ongoing basis or by responding to ad-hoc questions. With a DataOps-powered CDP, data analysts work with autonomy and speed, drawing analytics from CDP data. Analysts use tools like Tableau and Alteryx to create these insights and independently promote their investigative work into production deliverables as needed.

Every resource, technology and tool in the data-analytics organization exists to support the data analyst's ability to serve Sales and Marketing. This also applies to Data Scientists who also deliver insights directly to Sales and Marketing colleagues.

DATA ENGINEERS AND THEIR TOOLS AND DATABASES

The Data Engineer works with the IT department and all data source providers to institute automated processes that move data from various data sources into a trusted, integrated CDP database under the complete control of the data-analytics team. The CDP database may include a data lake, which provides revision history, easy access, control and error recovery.

The engineer writes transforms that operate on the data lake, creating data warehouses and data marts used by data analysts and scientists. The data engineer also implements tests that monitor data at every point along the data-analytics pipeline assuring a high level of quality.

The data engineer lays the groundwork for other members of the team to perform analytics without having to be operations experts. With a dedicated data engineering function, DataOps provides a high level of service and responsiveness to the data-analytics team.

DATA OPERATIONS

The DataOps-powered CDP provides automated support for the creation, monitoring and management of the end-to-end data pipeline. This includes stewardship of every aspect of the journey from data sources to reporting. Statistical Process Control (SPC) data-quality tests monitor each stage of the automated pipeline, alerting data engineering when data fails to meet statistical controls or match business logic.

With tests monitoring each stage of the automated data pipeline, DataOps can produce a dashboard showing the status of the pipeline. The DataOps dashboard provides a high-level overview of the end-to-end data pipeline. Is any data failing quality tests? What are the error rates? Which are the troublesome data sources? With this information at his or her fingertips, the Data Engineer can proactively improve the data pipeline to increase robustness. In the event of a high-severity data anomaly, an alert is sent to the Data Engineer who can take steps to protect production analytics and work to resolve the error. If the anomaly relates to a data supplier, data engineering can work with the vendor to drive the issue to resolution. Workarounds and data patches can be implemented as needed with information in release notes for users. In many cases, errors are resolved without the users (or the organization's management) ever being aware of any problem.

AGILE TEAM MANAGEMENT

The Agile methodology governs the creation of new analytics, producing a steady stream of valuable innovations and improvements to analytic insights in short increments of time. Agile is particularly effective in environments where requirements are quickly evolving a situation all too familiar to data-analytics professionals. Agile development is not only a method; it is also a philosophy and a mindset. Developers collaborate with sales/marketing customers, respond to change, measure progress through "delivered analytics," release frequently, seek feedback on releases, and adjust behavior to become more effective.

DATAOPS PLATFORM

The various methodologies, processes, people (and their tools) and the CDP analytics database are tied together cohesively using a technical environment called a DataOps Platform. The DataOps Platform includes support for:

- Agile project management
- Deployment of new analytics
- Execution of the data pipeline (orchestration)
- Integration of all tools and platforms
- Management of development and production environments
- Source-code version control
- Testing and monitoring of data quality
- Data Operations reporting and dashboards

The high degree of automation offered by DataOps eliminates a great deal of work that has traditionally been done manually. This frees up the team to create new analytics requested by stakeholder partners.

A DataOps Platform is not a one-size-fits-all tool. It is the central application that coordinates the various tools that drive your orchestration, testing, deployment, model deployment, development-environment management, change management, and data integration. You can create your own DataOps Platform from scratch, although partnering with a supplier can reduce time to market. Working with a partner also gives you the option of treating the entire CDP and data pipeline as a managed service, which can be initially outsourced and then partly or entirely taken over by internal resources at a later time.

DATAKITCHEN DATAOPS-CDP SOLUTION

An enterprise can outsource data engineering, databases, data operations, Agile team management and the DataOps Technology Platform to gain efficiencies. DataKitchen offers a DataOps Technology Platform as well as managed services for each of the gray boxes in figure 97. In essence, DataKitchen offers all aspects of the DataOps-powered CDP except for data analysis and data science, which rely upon vertical market expertise and close collaboration with sales and marketing.

The enterprise can also outsource the functions shown initially but insource them at a later date. Once set-up, the DataOps Platform can be easily and seamlessly transitioned to an internal team.

Customer Data Platforms promise to drive sales and improve the customer experience by unifying customer data from numerous disjointed operational systems. As a leader of the analytics team, you can take control of sales and marketing data by implementing efficient analytics-creation and deployment processes using a DataOps-powered CDP. A DataOps platform makes analytics responsive and robust. This enables your data analysts and scientists to rise above the bits and bytes of data operations and focus on new analytics that help the organization achieve its goals.

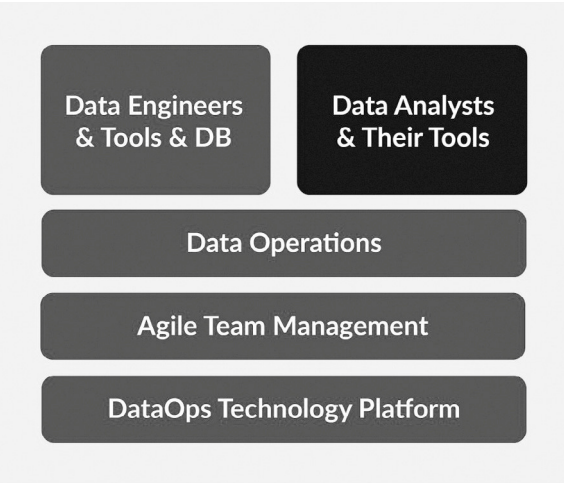


Figure 97: DataKitchen DataOps-Powered CDP and Managed Services

How a Mixed Martial Arts Fighter Would Approach Data Analytics

By James Royster, Director, Commercial Analytics, Celgene

Mixed Martial Arts (MMA) combines striking, wrestling and other fighting techniques into a unified sport. Every martial art and fighting technique has its strengths and strategic advantages. Boxing is known for punching but also provides footwork, guard position and head movement. Wrestling relies upon takedowns. Karate features striking techniques such as kicking. MMA is a hybrid of all of these (and many more) drawing upon each mode of combat as needed for a given competitive situation. If an MMA athlete competed against a boxer or karate expert, the mixed martial artist would clearly have an unfair advantage. MMA's real strength is its versatility and its ability to absorb new methods.

DataOps is the mixed martial arts of data analytics. It is a hybrid of Agile Development, DevOps and the statistical process controls drawn from lean manufacturing. Like MMA, the strength of DataOps is its readiness to evolve and incorporate new techniques that improve the quality, reliability, and flexibility of the data analytics pipeline. DataOps gives data analytics professionals an unfair advantage over those who are doing things the old way — using hope, heroism or just going slowly in order to cope with the rapidly changing requirements
of the competitive marketplace.

Agile development has revolutionized the speed of software development over the past twenty years. Before Agile, development teams spent long periods of time developing specifications that would be obsolete long before deployment. Agile breaks down software development into small increments, which are defined and implemented quickly. This allows a development team to become much more responsive to customer requirements and ultimately accelerates time to market.

Data analytics shares much in common with software development. Conceptually, the data analytics pipeline takes raw data, passes it through a series of steps and turns it into actionable information. Files, such as scripts, code, algorithms, configuration files, and many others, drive each processing stage. These files, taken as a whole, are essentially just code. As a coding endeavor, data analytics has the opportunity to improve implementation speeds by
an order of magnitude using techniques like Agile development. DevOps offers an additional opportunity for improvement.

The difficulty of procuring and provisioning physical IT resources has often hampered data analytics. In the software development domain, leading-edge companies are turning to DevOps, which utilizes cloud resources instead of on-site servers and storage. This allows developers to procure and provision IT resources nearly instantly and with much greater control over the run-time environment. This improves flexibility and yields another order of magnitude improvement in the speed of deploying features to the user base.
DataOps also incorporates lean manufacturing techniques into data analytics through the use of statistical process controls. In manufacturing, tests are used to monitor and improve the quality of factory-floor processes. In DataOps, tests are used to verify the inputs, business logic, and outputs at each stage of the data analytics pipeline. The data analytics professional adds a test each time a change is made. The suite of tests grows over time until it eventually becomes quite substantial. The tests validate the quality and integrity of a new release when a feature set is released to the user base. Tests allow the data analytics professional to quickly verify a release, substantially reducing the amount of time spent on deploying updates.

Statistical process control also monitors data, alerting the data team to an unexpected variance. This may require updates to the business logic built into the tests, or it might lead data scientists down new paths of inquiry or experimentation. The test alerts can be a starting point for creative discovery.

The combination of Agile development, DevOps, and statistical process controls gives DataOps the strategic tools to reduce time to insight, improve the quality of analytics, promote reuse and refactoring and lower the marginal cost of asking the next business question. Like mixed martial arts, DataOps draws its effectiveness from an eclectic mix of tools and techniques drawn from other fields and domains. Individually, each of these techniques is valuable, but together they form an effective new approach, which can take your data analytics to the next level.

Reinvent Marketing Automation with the DataKitchen DataOps Platform

A global pharmaceutical giant sought to drive top-line growth by modernizing its marketing operations. The project included a migration to Salesforce Marketing Cloud, integrations with numerous internal and third-party data sources, and a continuous flow of data. The plan initially required eighteen months for implementation. Using the DataKitchen DataOps Platform, which automates deployment, controls quality and supports Agile development of analytics, the company was able to start delivering value in six weeks and completed the migration in about one third the time.

THE CHALLENGE OF MARKETING AUTOMATION AT SCALE

The company faced numerous difficulties when implementing marketing automation for multiple global business units:

- **Distributed Data –** The company's marketing analytics and customer data were distributed in many specialized systems that do not easily talk to each other. This made it challenging to link customer data from one system with another. Customers engage through emails, partner websites, advertisements, campaigns and across product lines. Each of these touchpoints produces a continuous stream of fine-grained opt-in/opt-out requests which all must be consolidated and synchronized. Cross-referencing customer data with third-party databases also provides valuable segmentation information.
- **Supporting Agile –** The company lacked the technical infrastructure to implement Agile development of data flows and analytics. Using a slow and inflexible development process prevented them from keeping up with the fast-paced requirements of the sales and marketing teams.
- **Iterating on Data Quality –** Analysts had trouble specifying data quality rules until they saw the data. In a non-agile environment, this caused requirements to keep changing, causing delays.
- **Continuous Change Requests –** Once a system is operational, users are inspired to request additional data sources, segmentations and other enhancements. With long development cycles, it was difficult for the team to keep up with the users' continuous demands.

AUTOMATED EFFICIENCY AND QUALITY WITH DATAKITCHEN

The company utilized the DataKitchen Platform to oversee and monitor the end-to-end data pipeline. With the DataKitchen solution, the company is now able to:

- **Automate and monitor data pipelines –** Data flows from sources to user analytics in a continuous pipeline.
- **Implement continuous deployment –** New analytics are tested and deployed to users with speed and confidence using automation.

• **Control quality –** Data is continuously monitored for anomalies with alerts and dashboards that provide real-time information about data quality and operations.
• **Manage sandboxes –** Development environments are created as needed to prevent enhancements from disrupting operations.

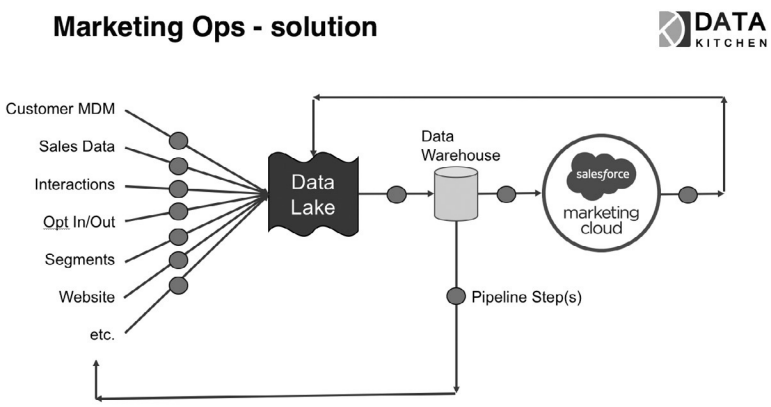


Figure 98: With DataKitchen, marketing automation data flows continuously from numerous sources through the analytics pipeline with efficiency and quality.

BUSINESS IMPACT

With the DataKitchen Platform, the company was able to break the long 18-month project into sprints and began to deliver value in six weeks. The agility of the DataKitchen DataOps approach enabled the analytics team to rapidly respond to changing user requirements with a continuous series of enhancements. Users no longer waited months to add new data sources or make other changes. The team can now deploy new data sources, update schemas and produce new analytics quickly and efficiently without fear of disrupting the existing data pipelines.

DataKitchen's lean manufacturing control helped the team be more proactive addressing data quality issues. With monitoring and alerts, the team is now able to provide immediate feedback to data suppliers about issues and can prevent bad data from reaching user

analytics. All this has led to improved insight into customers and markets and higher impact

marketing campaigns that drive revenue growth.

DataKitchen's DataOps Platform helped this pharmaceutical company achieve its strategic goals by improving analytics quality, responsiveness, and efficiency. DataKitchen software provides support for improved processes, automation of tools, and agile development of

new
analytics. With DataKitchen, the analytics team was able to deliver value to users in 1/10th the time, accelerating and magnifying their impact on top line growth.

Meeting the Product Launch Challenge with DataOps

> *"So much of what we do involves business questions that are fire drills. Executives want answers as quickly as possible. The infrastruc- ture that we've set-up with DataKitchen allows us to mix and match data in new ways so that we can quickly get the answer to a question."*
>
> *—Manager, Data Analyst, Celgene*

Celgene is a $12B biopharmaceutical company committed to delivering innovative treatments for patients worldwide. Celgene relies upon DataKitchen to enable rapid-response, high-quality data analytics that help the company maximize product lifetime revenue.

It costs between $2-3B to bring a new pharmaceutical to market. When a new drug is introduced, it is already halfway through its patent life. This makes the first 6-12 months of a pharmaceutical launch critical to a product's lifetime revenue. The vendor needs up-to-date information to allocate samples, plan marketing events, and monitor progress vs. goals. With so much at stake, pharmaceutical companies like Celgene make strategic investments to maximize product adoption and adherence during the initial phase of a drug product's life cycle.

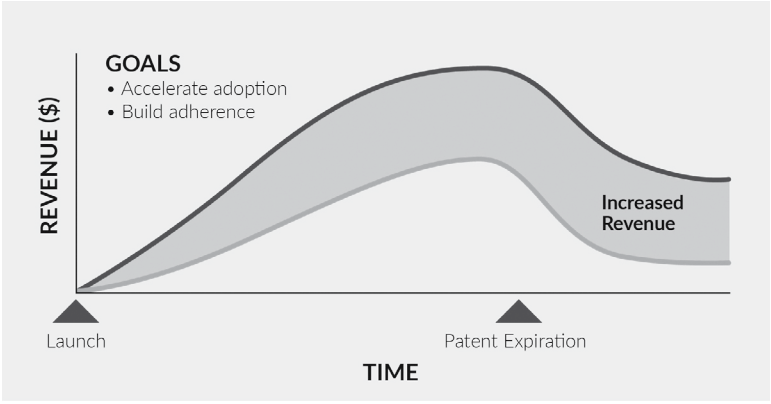


Figure 99: The first year is critical to a pharmaceutical product's lifetime revenue.

THE ROLE OF DATA IN LAUNCH SUCCESS

Celgene has found that using analytics to understand customers and markets can significant- ly improve product launch success. The analytics produced range from weekly, standardized reports to ad-hoc analyses. In the first months of a product's lifecycle, the sales and market- ing teams can't wait weeks or months for new analytics. Every new data source, question, and innovative idea demands an immediate and accurate response.

THE OBSTACLES TO RESPONSIVE, HIGH-QUALITY ANALYTICS

The Celgene data engineering and analytics teams faced many obstacles that prevented analytics responsiveness and quality. Data was organized in silos—using a variety of technologies and isolated platforms. Without the right processes and tools in place, the data engineering and analytics teams can spend a majority of their time on data engineering and pipeline maintenance. This distracts them from their main mission — producing analytic insights that help the business attain its objectives.

THE DATAKITCHEN PLATFORM

Celgene chose DataKitchen to enable data engineering that streamlines development and data operations processes, helping the data analysts and scientists to remain focused on creating value — at the speed of business. With the DataKitchen Platform, Celgene has been able to:

- **Automate orchestration –** DataKitchen automates the deployment of new analytics and performs data pipeline orchestration, freeing up the team from manual processes and enabling them to focus on extracting value from data.
- **Monitor data quality –** Statistical process control and dashboards help monitor and control the quality of the end-to-end data pipeline, and real-time alerts provide high-level visibility into incidents and provide critical information.
- **Automate deployment –** Once new features pass all their tests, just a push of a button is required to deploy into production, with confidence.

DATA SOURCES & INTEGRATIONS
IQVIA (IMS)
Symphony
Veeva (salesforce.com)
Specialty Pharmacies
Email & Web Interactions
Sales Alignments Targets and Prospects
Product Hierarchies
Customer Segments

Table 8

ANALYTICS AT SUPER SPEED

With help from DataKitchen, Celgene was able to improve the productivity of the data engineering function by an order of magnitude. With automation of their data pipeline, they were able to update 30X more visualizations per week. They were able to compress the cycle time required to produce new analytics from weeks (or months) to one day. This enabled the data analytics team to successfully address the volume of questions from sales and marketing, helping them maximize product adoption during the critical first phase of their product launch. Instead of just fighting fires, the data team felt like they had acquired *superpowers*.

METRIC	BEFORE	WITH DATAKITCHEN
Data analysts supported by one data engineer	0.5	12
Schema changes per week by one data engineer	1	12
Sales people supported by one data analyst	50	250
Cycle time to publish new visualizations	weeks/ months	next day
Visualizations updated/wk	50	1500

Table 9

MOVING FORWARD WITH DATAKITCHEN

Use of DataKitchen fostered a tight collaboration between data analytics and the business unit, unlocking creativity made possible by DataOps agility and quality. Impressed by the performance of the data-analytics team using DataKitchen, Celgene decided to expand their use of DataKitchen throughout the company.

> *"DataKitchen has enabled us to become nimble and agile when it comes to data. We are now*
> *a self-service data organization — from the marketing department to the sales reps."*
>
> *—Director, Market Insights, Celgene*

DataOps Classic Baked Macaroni and Cheese

by Joanne Ferrari

INGREDIENTS AND TOOLS

- 2 Cups Milk
- 2 Tablespoons Butter
- 2 Tablespoons All-Purpose Flour
- ½ Teaspoon Salt
- ¼ Teaspoon Freshly Ground Black Pepper
- 1 (10 oz.) Block Extra Sharp Cheddar Cheese, Shredded • ¼ Teaspoon Ground Red Pepper (Optional)
- ½ (16 oz.) Package Elbow Macaroni, Cooked

INSTRUCTIONS

1. Preheat oven to 400◦. Microwave milk at HIGH for 1 ½ minutes. Melt butter in a large skillet or Dutch oven over medium-low heat; whisk in flour until smooth. Cook, whisking constantly for 1 minute.
2. Gradually whisk in warm milk and cook, whisking constantly 5 minutes or until thickened.
3. Whisk in salt, black pepper, 1 cup shredded cheese, and if desired, red pepper until smooth; stir in pasta. Spoon pasta mixture into a lightly greased 2-qt. baking dish; top with remaining cheese. Bake at 400◦ for 20 minutes or until golden and bubbly.

NOTES

For this recipe, it is recommended that you grate the block(s) of cheese. I combine Sharp Cheddar and Swiss cheeses — my favorite. Pre-shredded varieties won't give you the same sharp bite or melt into creamy goodness over your macaroni as smoothly as block cheese that you grate yourself. You can go reduced-fat (but then it's even more important to prep your own). Grating won't take long, and the rest of this recipe is super simple. Use a pasta that has plenty of nooks to capture the cheese—like elbows, shells, or cavatappi. Try it just once, and I guarantee that Classic Baked Macaroni and Cheese will become your go-to comfort food.

DataOps Survey

Tomorrow's Forecast: Cloudy with a Chance of Data Errors

KEY FINDINGS OF THE 2019 DATAOPS SURVEY

Whatever you were planning to accomplish this week — forget about it!

Chances are good that you'll be interrupted by data errors. That is the clear message communicated by the respondents to a joint DataOps survey conducted by DataKitchen and Eckerson Research.

The survey contains feedback from 300 data-analytics professionals who work for medium to large-sized companies across multiple industries in the US and Europe. The full report will be available in June 2019, but here are the key results. The survey sheds light upon three issues that embody the analytics industry's challenges. From a recent NewVantage Partners Report, we know that despite the hype and investment, the number of companies that identify as data-driven is declining. Gartner estimated that 60% of big data projects fail. We see results in our survey that may help explain this.

THE IMPACT OF DATA ERRORS

One egregious issue is that analytics too often contain errors which erode the credibility of the data team. 30% of respondents to the DataOps survey reported more than 11 errors per month. This is a staggering figure. That means that the data team is probably moving from fighting one fire to next with little time for any value-add activity. The managers of these enterprises learn not to trust the data. Is it any wonder that companies are becoming less data-driven?

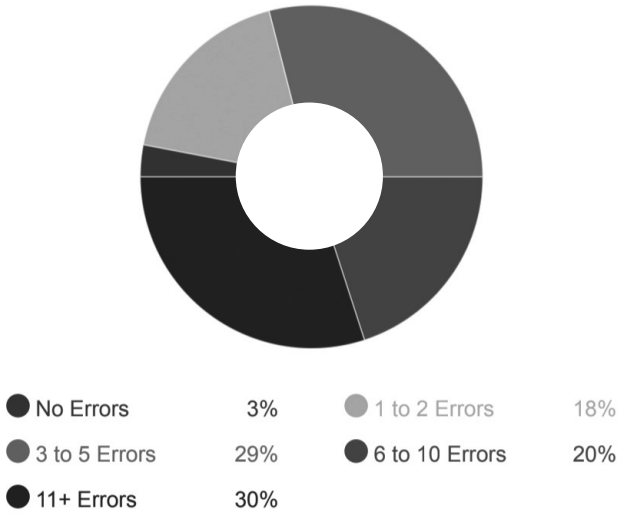
The DataOps enterprises that DataKitchen works with have less than 1 data error per year. Only 3% of the companies surveyed approached that level of quality. Another 18% reported 1-2 errors per month. Would W. Edwards Deming have considered that an acceptable failure rate? Would Toyota? In a manufacturing setting, each one of these errors could be the equiv-

alent of a product recall. For the sake of discussion, let's presume that 1-2 errors per month in an enterprise analytics context are tolerable (it's really not). That still leaves nearly 80% of companies surveyed reporting 5, 10 or even more errors per month. That has a big impact on how an organization views its data and may even explain why the average tenure of a Chief Data Officer is only around 2 years.

THE IMPACT OF DATA ERRORS

Data errors negatively impact the productivity of the analytics teams in several ways. They flood Kanban boards with new tasks. They cause unproductive context switches. Wary of making further errors, the data team may become overly cautious, working more slowly. In short, data errors are a major bottleneck that affect the entire workflow of new analytics development. We call this analytics cycle time, and it is one of the critical bottlenecks that slow the ability of analytics to create value for an enterprise.

On average, how many errors (e.g., incorrect data, broken reports, late delivery, customer complaints) do you have each month?



A short cycle time enables an analytics team to respond quickly to requests for new analytics. When analytics are produced quickly, the data team can keep pace with the endless stream of requests from the business unit. A short cycle time fosters close collaboration with business users and in our experience this unlocks an organization's creativity. However, the cycle time in most data organizations is plagued with inefficient manual processes, bureaucracy, lack of task coordination and dependencies on bottlenecks. For example, survey respondents provided interesting feedback about the time it takes to create an analytics development environment.

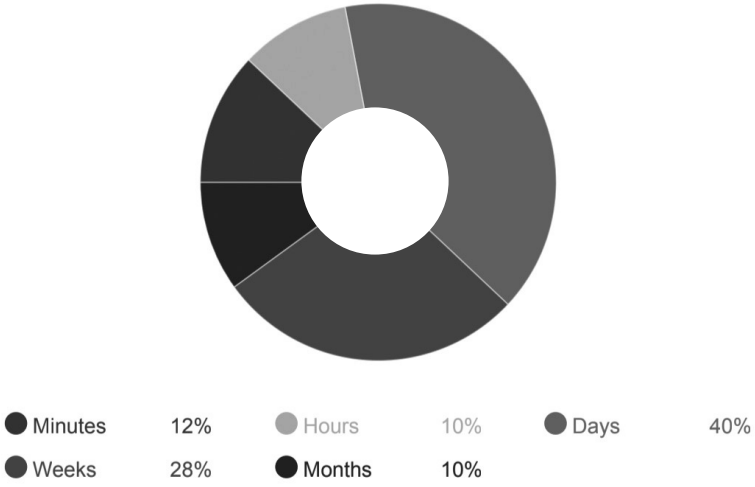
DELAYS IN ENVIRONMENT CREATION

Isolated development environments are an important way that analytics development can proceed without impacting data operations. If it takes weeks or months for IT to provision hardware and software or to make data available, the queue time severely impacts the pro- ductivity of analytics professionals.

78% of respondents indicated that it takes days, weeks or months to create a

development
environment. 38% of users surveyed report that it took weeks or months. That wait time prevents the data analytics team from even beginning to work on the critical analytics that the organization has requested. This means that their time-to-value is much slower than it should be.

On average, how long does it take your team to create a new development environment with the appropriate test data, servers, and tools?



DataKitchen Interpretation

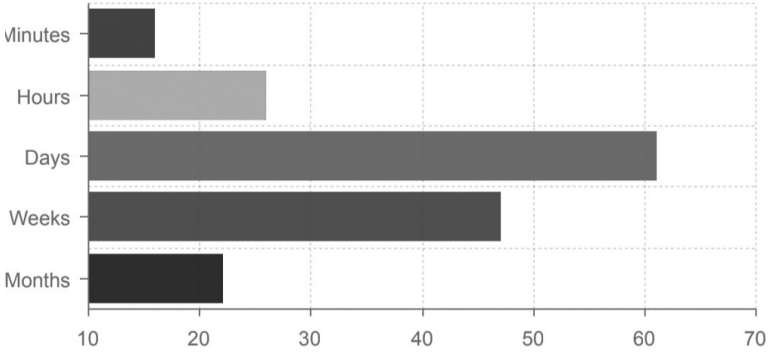
MOST COMPANIES ARE VERY SLOW CREATING NEW DEVELOPMENT ENVIRONMENTS

78%

LENGTHY DEPLOYMENT TIMES

We asked our survey respondents directly about the end-to-end cycle time of creating new analytics. In light of the above, it is not surprising that we see that far too many organizations undergo lengthy periods of time to create and deploy analytics. 76% of organizations surveyed take days, weeks or months to move from analytics development to production. If one data engineer needs to support a dozen data analysts and each data analyst needs to support hundreds of salespeople, the cycle time for new analytics must be reduced to hours or better yet, minutes. Enterprises can reach these performance targets with a DataOps approach to analytics development and deployment

On average, how long does it take to move a new or modified data analytic pipeline from development to production?



DataKitchen Interpretation

MOST COMPANIES ARE TOO SLOW TO DEPLOY CHANGES INTO PRODUCTION

76%

CONCLUSION

The three survey responses above help explain why enterprises are not able to respond to user requests for new and updated analytics in a reasonable time frame. For organizations that suffer from these constraints, the case could be made that nothing else matters but addressing these bottlenecks. The ability to rapidly produce and deploy analytics is at the heart of a data team's ability to add value. If viewed from the perspective of the Theory of Constraints, these bottlenecks limit the overall throughput of value creation. Any improve- ment in analytics development cycle time improves the overall throughput of the system. An improvement other than in the bottleneck is an illusory accomplishment when an analytics team suffers from long development cycle times.

DataOps offers a way to reduce errors, shorten the time it takes to set up a development environment, and minimize analytics development cycle time. Nothing could state more clearly why analytics organizations need a DataOps initiative now.

Pain survey with data,world

Data engineers are the backbone of the modern data-driven enterprise, and they serve in one of the most critical and celebrated roles in the tech industry. While data engineering tools and trends are frequently discussed and analyzed, conversations rarely focus on the day-to-day lives of data engineers and their lived experiences. To delve into this largely unexplored subject, DataKitchen and data.world teamed up to commission a survey intended to further understand the people behind
the keyboards. Our survey gives voice to 600 data engineers and data engineering managers1 who shared their hopes, challenges, frustrations and insight.

There are numerous challenges in the data analytics space. Recently, Gartner2 wrote that “most analytics and AI projects fail because operationalization is only addressed as an afterthought.” The average tenure of a CDO or CAO is only about 2.5 years. Our survey confirms that data engineering is not immune to these systemic forces. A full 78% of those surveyed wished that their job came with a therapist to help manage work-related stress (see Figure 1). We share the insights from our survey in the hope that enterprises will heed the call to action and institute process, workflow and other supportive changes for data engineers and similar roles.

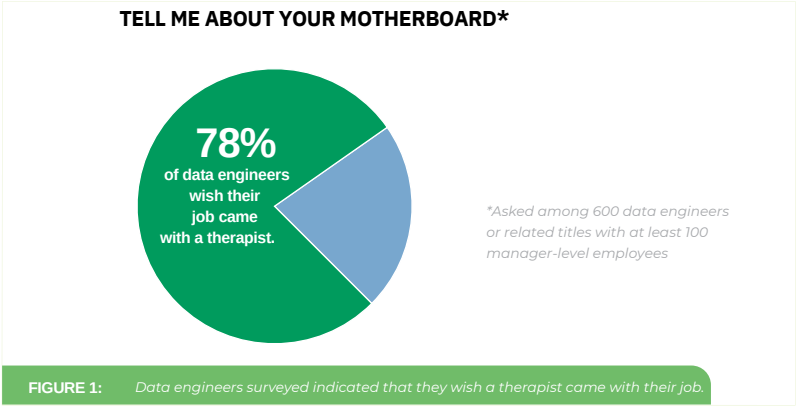


FIGURE 1: *Data engineers surveyed indicated that they wish a therapist came with their job.*

WHAT IS A DATA ENGINEER?

A data engineer is a software or computer engineer who lays the groundwork for team members, like analysts and data scientists, to perform analytics. Data engineers ensure that data is available, secure, correct, and fit for purpose. For example, the data engineer moves data from operational systems (ERP, CRM, MRP, etc.) or third-party sources into a data lake and writes the transforms that populate schemas in the data warehouses and data marts that power self-service analysis or automated charts, graphs and models.

Data engineering requires proficiency in cloud and cluster computing, ETL frameworks, batch/stream processing, orchestration, containers, coding, Agile Development, DevOps and observability. A competent data engineer influences the productivity of many others, so data engineers are valued and paid more on average than data analysts, scientists and DB admins. The life of a data engineer sounds pretty rosy when we look at it from thirty-thousand feet, but let's explore what data engineering is really like through the lens of our survey respondents.

THE DAY-TO-DAY WORK LIFE OF A DATA ENGINEER

When we asked data engineers how they feel about their daily work, 97% reported feeling *burned-out*. Data engineering is a relentlessly demanding profession in which you face a steady stream of requests from users, high-priority interruptions and ill-defined projects. The data engineers in our survey listed the challenges below as significant contributors to their feeling of burnout:

Sources of Burnout	Percent Responded
Focusing too much time on finding and fixing errors	50%
Focusing too much on maintaining data pipelines and/or manual processes	50%
Constantly playing catch up with stakeholder requests	49%
The fast pace of requests from stakeholders	48%
Lack of feedback on the products delivered	47%
Unreasonable requests from stakeholders	42%

TABLE 1: *Sources of burnout identified by data engineers and managers*

THE RELENTLESS FLOW OF ERRORS

Data errors are a huge productivity drain on data engineers. Yet 52% of those surveyed did not feel like their companies sufficiently addressed data quality issues in a rigorous and systematic way. These respondents indicated that they "frequently hope and pray that things don't break." Many enterprises ingest data from sources and transform it with minimal quality controls. When working toward a deliverable, it is tempting to just quickly produce a solution with minimal testing, push it out to the users and hope it does not break. This approach has inherent risks. Eventually, a deliverable will contain data errors, upsetting the users and harming the hard-won credibility of the data analytics team.

A data engineer in a typical enterprise knows that errors can occur at any time. With no strategy to eliminate errors, the data engineering team can only “hope” for the best — another hour, another day — until the next interruption. A typical enterprise experiences multiple data, pipeline or analytics errors per week. Managing a continuing succession of outages while trying to keep development projects on schedule and under budget is like trying to play “*whack a mole*” while simultaneously reading a book.
Data engineers know that an enterprise cannot derive value from its data unless the data team stays focused on innovation, but errors create unplanned work that can’t be ignored. For data engineers, this is a no-win situation.

MANUAL PROCESSES CROWD OUT INNOVATION

A recent Gartner3 survey showed that data professionals spent 56% of their time on operational execution and only 22% on innovation that delivers value — figure 2. Gartner describes the time spent on “operational execution” as using the data team to implement and maintain production initiatives. In other words, highly skilled data team members are asked to manually execute procedures that ingest, clean, transform, and disseminate data. Our survey confirms the magnitude of this problem, with 50% citing manual processes as an issue for data engineers.

Companies that wish to cultivate data analytics as a sustainable competitive advantage need to find a way to flip the script. Data engineers need to be spending 80% of their time on value-add initiatives and 20% on operational execution and internal meetings.

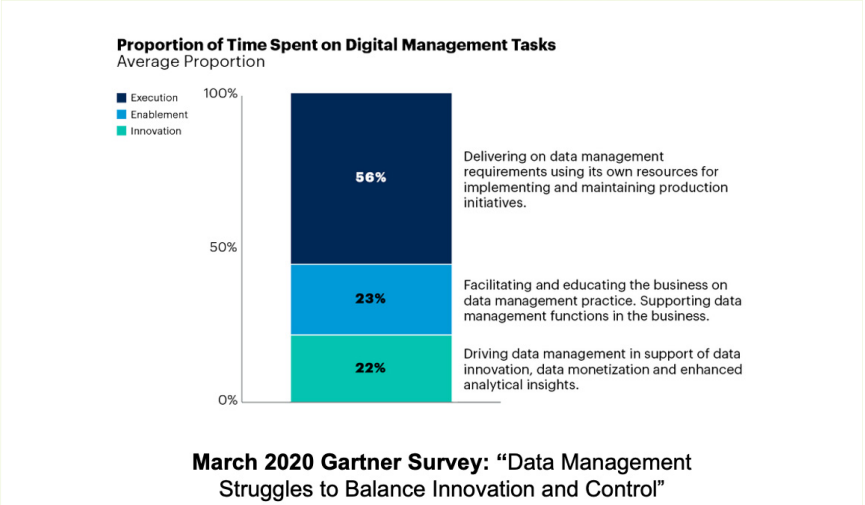


FIGURE 2: *Data professionals spend the majority of their time on operational execution. Source: Gartner (Chien, 2020).*

YOU WANT IT WHEN?

Another major cause of burnout is the steady stream of half-baked requests from stakeholders. Like everything else in our *one-click* world, analytics is now expected to happen instantaneously and reliably. Ninety-one percent of our respondents reported receiving requests for analytics with unrealistic or unreasonable expectations. 61% said this happens “often” or “all the time” — figure 3. A majority of respondents reported receiving requests for analytics that are simply not possible to complete in the time requested or not possible with the functions and features specified. This phenomenon is not a surprise since business colleagues have little understanding of the complexity required to deliver accurate charts and graphs to decision-makers.

Data engineers focus on delivering clean and accurate data, and an effective way to optimize these impractical requests is to catalog enterprise data. By implementing a data catalog platform, data engineering teams can better understand and connect all data sources, simplifying managing and monitoring data pipelines. At the end of the day, unreasonable asks will happen, and a data catalog is the proper secret force to allow data engineers to meet crazy expectations.

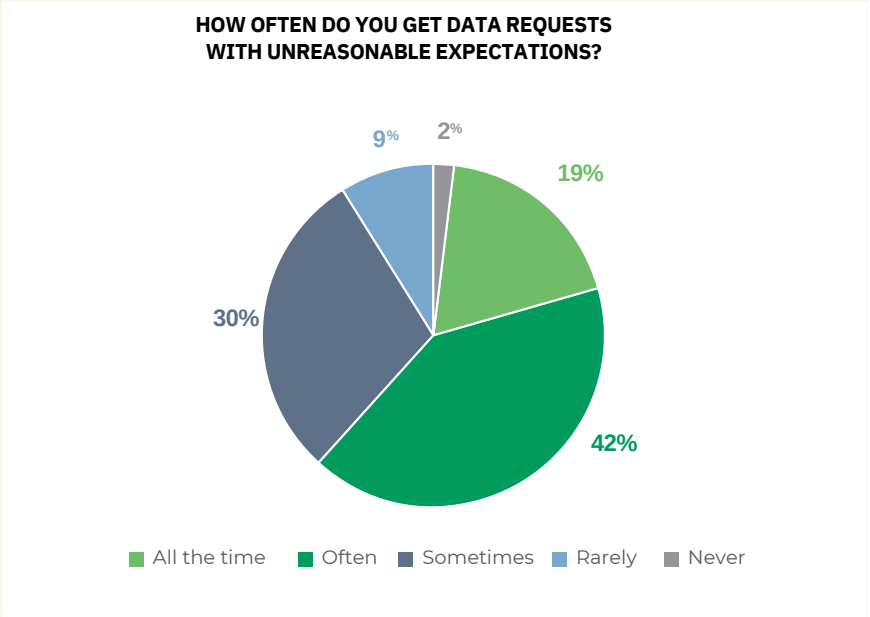


FIGURE 3: *Data engineers routinely receive unreasonable requests.*

SHAME AND BLAME

Other teams depend on the product of data engineering. Regular data outages erode trust, so when there's a problem in critical analytics, people within the organization engage in blaming and finger-pointing. In our survey, 87% of respondents said they are blamed frequently when things go wrong with the company's data and analytics. Sixty- three percent said this happens "often" or "all the time."

Public shaming can cause a range of bad feelings among data engineers. It can lead to anxiety and a reluctance to take technical risks — a significant obstacle to productivity. Shaming and blaming take the fun out of the most enjoyable aspects of data engineering working with data.

STYMIED BY GOVERNANCE BUREAUCRACY

On the topic of working with data, 69% of those surveyed said their company's data governance policies make their day-to-day job more difficult. The "lock-it-down" approach employed by many organizations lacks transparency, often resulting in more work for data engineers who are beholden to complicated processes for managing access to data sources.

Enterprises can alleviate this burden by practicing Agile Data Governance. Unlike traditional top-down data governance, Agile Data Governance opens up some traditionally restricted governance functions to a broader audience to iteratively capture the knowledge of data producers and consumers so everyone can benefit. Think of it like putting access on rails: making data fully auditable and predictable simplifies its management, so data engineers are free to work on more impactful projects.

WORK-LIFE *IMBALANCE*

Data engineers work hard and deserve to spend their downtime relaxing. The problem is that issues and errors create unplanned work, which forces data engineers to work long, irregular schedules. Eighty-nine percent of data engineers in our survey reported frequent disruptions to work-life balance due to unplanned work. Fifty percent said this occurs "often" or "all the time."

Data engineers work overtime to compensate for the gap between performance and expectations. When a deliverable is met, data engineers are considered heroes. However, "heroism" is a trap. Heroes give up work-life balance. Yesterday's heroes are quickly forgotten when there is a new deliverable to meet. The long hours eventually lead to burnout, anxiety and even depression. Heroism is difficult to sustain over a long period, and it ultimately just resets expectations at a higher level without addressing the root cause of productivity bottlenecks.

I'M OUTTA HERE

In light of what we have learned about burnout, obstacles to productivity, high-profile shaming and lack of work-life balance, perhaps it isn't a surprise that over 70% of the data engineers surveyed indicated that they are likely to leave their current company in the next twelve months. Even more surprising is that 79% of those surveyed have considered abandoning the field of data engineering entirely — figure 4. This suggests that data engineers don't believe that the challenges that they experience are specific to their particular job situation or enterprise. The problems are industry-wide and can't be avoided simply by changing companies. These data engineers feel that the profession of data engineering is broken. Can it be fixed?

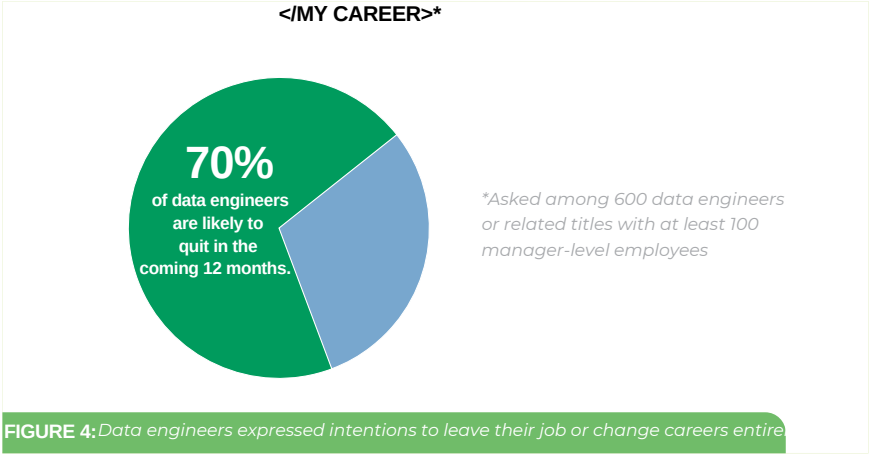


FIGURE 4: *Data engineers expressed intentions to leave their job or change careers entire*

TOOLS SERVE WORKFLOWS

Tools vendors have learned that they can garner significant attention by claiming that their tool alone will solve a particular data problem. Still, our surveyed group of data engineers see through the hype. Over 89% agreed with the statement that "cutting edge tools for managing data and building analytics are ineffective without processes that deploy, monitor and manage analytics throughout the lifecycle." In truth, tools are not an end in themselves. They serve lifecycle workflows.

According to quality pioneer W. Edwards Deming, 94% of problems are "common cause variation." To decrease this variation, you must focus on the system or process. This observation applies equally to factories that make widgets and data organizations that produce analytics. The best way to reduce waste and eliminate errors is to improve systemic processes and workflows. When quality methods, such as lean manufacturing, are applied to the end-to-end lifecycle of data and analytics, the term of art is called "DataOps." Overall, 78% feel DataOps is essential or very important to successfully manage data processes. This was even higher among data engineering managers, 91% of whom recognized the importance of DataOps — see Figure 5 below.

WHAT IS DATAOPS?

DataOps is a collection of technical practices, workflows, cultural norms, and architectural patterns that enable:

- Rapid innovation and experimentation, delivering new insights to customers with increasing velocity
- Extremely high quality and very low error rates
- Collaboration across complex arrays of people, technology, and environments
- End-to-end observability with clear and precise measurement, monitoring and transparency of results

In practical terms, DataOps employs automation to streamline analytics development and data operations workflows. DataOps also integrates testing, monitoring and observability into production data pipelines. By automating mundane tasks, DataOps enables data engineers to focus on higher-value activities. For example, data professionals can create sandbox environments for new analytics development projects on-demand — with a few clicks. DataOps slashes the time and effort required to release analytics to production by incorporating DevOps continuous integration, delivery and deployment into analytics development workflows. DataOps also institutes process and workflow metrics, so there is unprecedented transparency across the data lifecycle.

DataOps addresses each of the sources of burnout listed above. By introducing cataloging and observability into data operations and analytics development, DataOps helps the data team catch errors before they become critical outages. It automates data pipelines and prioritizes last mile governance, democratizing data and improving collaboration between groups using hierarchies of orchestration. DataOps also slashes development cycle time by automating non-value-add aspects of analytics creation, enabling the data engineering and analytics team to keep up with the steady stream of requests by users. It also implements Agile Development, so the team stays focused on development that adds tangible value and receives immediate feedback from stakeholders.

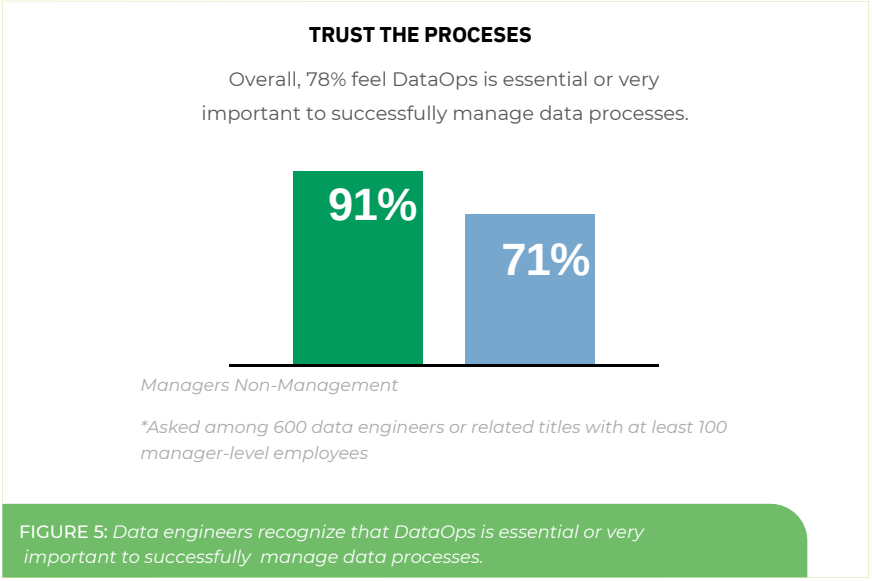


FIGURE 5: *Data engineers recognize that DataOps is essential or very important to successfully manage data processes.*

SAVING DATA ENGINEERS WITH DATAOPS AGILITY

The data engineers and managers in our survey spoke clearly about the challenges facing the data engineering profession. They feel burned-out by the relentless battle against data errors, inefficient manual processes, unreasonable requests, shaming and blaming, governance bureaucracy, and lack of work-life balance. The problem has reached a point where most individuals in the data engineering role are considering abandoning the profession altogether. Addressing data engineer burnout should be every organization's top priority.

Our survey participants also articulate a call to action. Enterprises can institute process change using methodologies like DataOps to orchestrate workflows that eliminate errors, accelerate delivery and deployment, foster collaboration and improve process transparency. By addressing the challenges of data engineering from a systemic and process perspective, DataOps can play a foundational role in enhancing the employee experience. Empowering people with processes and technology that enable greater communication, collaboration, integration, and automation will raise the quality and agility of analytics while putting the fun back into the data engineering role.

Additional Recipes

DataOps Vegan Corn Chowder

by Eran Strod

INGREDIENTS AND TOOLS

Cashew Cream

- 1 cup cashews soaked in water for at least 2 hours
- 2 cups veg stock
- 4 teaspoons cornstarch (can sub tapioca starch if desired)
- Drain the cashews. In a blender, combine all the ingredients and work for 2 to 5 minutes or until smooth, scraping down the sides with a rubber spatula several times. Set aside.

Soup

- 1 Tablespoon olive oil
- 1 large onion coarsely chopped
- 2 celery ribs, chopped
- 3 cups veg broth
- 1 large carrot chopped
- 1 red pepper diced (could sub 1 bag Frozen mixed-vegetables, thawed in a pinch)
- 1 potato, diced
- 3 ears of fresh corn (cut the kernels off and scrape the corn cobs for corn milk to add to the soup)
- Can of corn

INSTRUCTIONS

1. Heat the oil in 4-quart pot
2. When hot add onion and celery with a pinch of salt, cook until start to soften.
3. Add carrots and potatoes
4. Add corn and red pepper and stir-fry for 10 minutes
5. Add 3 cups veg stock and the corn milk
6. Bring to a boil, lower the head and cover — simmer 10 min or until veg tender
but not overcooked.
7. Stir in Cashew Cream and stir gently for 7 minutes until nicely thickened.
8. Blend up to half the soup to make more liquid and add it back in
9. Add salt & pepper to taste, depending on the type of veg stock you used.

My own adaptation of a vegan New England Clam Chowder recipe from the Boston Globe from Isa-does-it by Isa Chandra Moskowitz

DataOps Energy Bytes

by Eric Estabrooks

INGREDIENTS AND TOOLS

- 1 cup rolled oats
- ⅓ cup coconut flakes
- ½ cup peanut butter
- ½ cup ground flax seed
- ½ cup chocolate chips •
⅓ cup raw honey
- 1 tsp vanilla

INSTRUCTIONS

1. Add rolled oats, coconut flakes, nut butter, flax seed, honey, and vanilla to a
mixing bowl and mix.
2. Mix well so that you can form the balls easily.
3. Add chocolate chips if using or other desired mix ins.
4. Chill the mixture in the fridge for an hour so that balls will bind together.
5. Roll the balls into about a 1-inch diameter.

DataOps Resources

The Agile Manifesto	http://agilemanifesto.org/
DatOps Blog	http://bit.ly/2Ef2Hto
The DataOps Manifesto	http://dataopsmanifesto.org
DataOps News	http://bit.ly/2ORDlUr http://bit.ly/2PygnSb
DataOps SlideShare	http://bit.ly/2UFcKO8
DataOps Videos	http://www.scrumguides.org
Scrum Guides	https://en.wikipedia.org/wiki/Statistical_process_control
Statistical Process Control	https://en.wikipedia.org/wiki/W._Edwards_Deming
W. Edwards Deming	http://bit.ly/2DnlqR1
Wikipedia DataOps	https://en.wikipedia.org/wiki/DevOps
Wikipedia DevOps	

About the Authors

Christopher Bergh is a Founder and Head Chef at DataKitchen where, among other activi- ties, he is leading DataKitchen's DataOps initiative. Chris has more than 25 years of research, engineering, analytics, and executive management experience.

Previously, Chris was Regional Vice President in the Revenue Management Intelligence group in Model N. Before Model N, Chris was COO of LeapFrogRx, a descriptive and predictive analytics software and service provider. Chris led the acquisition of LeapFrogRx by Model N in January 2012. Prior to LeapFrogRx Chris was CTO and VP of Product Management of MarketSoft (now part of IBM) an innovative Enterprise Marketing Management software. Prior to that, Chris developed Microsoft Passport, the predecessor to Windows Live ID, a distributed authentication system used by 100s of Millions of users today. He was awarded a US Patent for his work on that project. Before joining Microsoft, he led the technical architecture and implementation of Firefly Passport, an early leader in Internet Personalization and Privacy. Microsoft subsequently acquired Firefly. Chris led the development of the first travel-related e-commerce web site at NetMarket. Chris began his career at the Massachusetts Institute of Technology's (MIT) Lincoln Laboratory and NASA Ames Research Center. There he created software and algorithms that provided aircraft arrival optimization assistance to Air Traffic Controllers at several major airports in the United States.
Chris served as a Peace Corps Volunteer Math Teacher in Botswana, Africa. Chris has an M.S. from Columbia University and a B.S. from the University of Wisconsin-Madison. He is an avid cyclist, hiker, reader, and father of two college age children.

Gil Benghiat is a Founder and VP of Products at DataKitchen where he is focusing on DataKitchen users and the Agile data practices.

Gil has held various technical and leadership roles at Solid Oak Consulting, HealthEdge, Phreesia, LeapFrogRx (purchased by Model N), Relicore (purchased by Symantec), Phase For-
ward (IPO and then purchased by Oracle), Netcentric, Sybase (purchased by SAP), and AT&T Bell Laboratories (now Nokia Bell Labs).

Gil's career has been data oriented starting with collecting and displaying network data at AT&T Bell Labs, managing data at Sybase, collecting and cleaning clinical trial data at PhaseForward, integrating pharmaceutical sales data at LeapFrogRx, protecting patient and financial data at Phreesia, processing claims data at HealthEdge, and liberating data at Solid Oak Consulting.

Gil holds an M.S. in Computer Science from Stanford University and a Sc.B. in Applied Mathematics/Biology from Brown University. He completed hiking all 48 of New Hampshire's, 4,000 peaks and is now working on the New England 67, and is the father of one high school and two college age boys.

Eran Strod works in marketing at DataKitchen where he writes white papers, case studies and the DataOps blog. Eran was previously Director of Marketing for Atrenne Integrated Solutions (now Celestica) and has held product marketing and systems engineering roles at Curtiss-Wright, Black Duck Software (now Synopsys), Mercury Systems, Motorola Computer Group (now Artesyn), and Freescale Semiconductor (now NXP), where he was a contributing author to the book "Network Processor Design, Issues and Practices."

Eran began his career as a software developer at CSPi working in the field of embedded computing.
Eran holds a B.A. in Computer Science and Psychology from the University of California at Santa Cruz and an M.B.A. from Northeastern University. He is father to two children and enjoys hiking, travel and watching the New England Patriots.

DATA KITCHEN