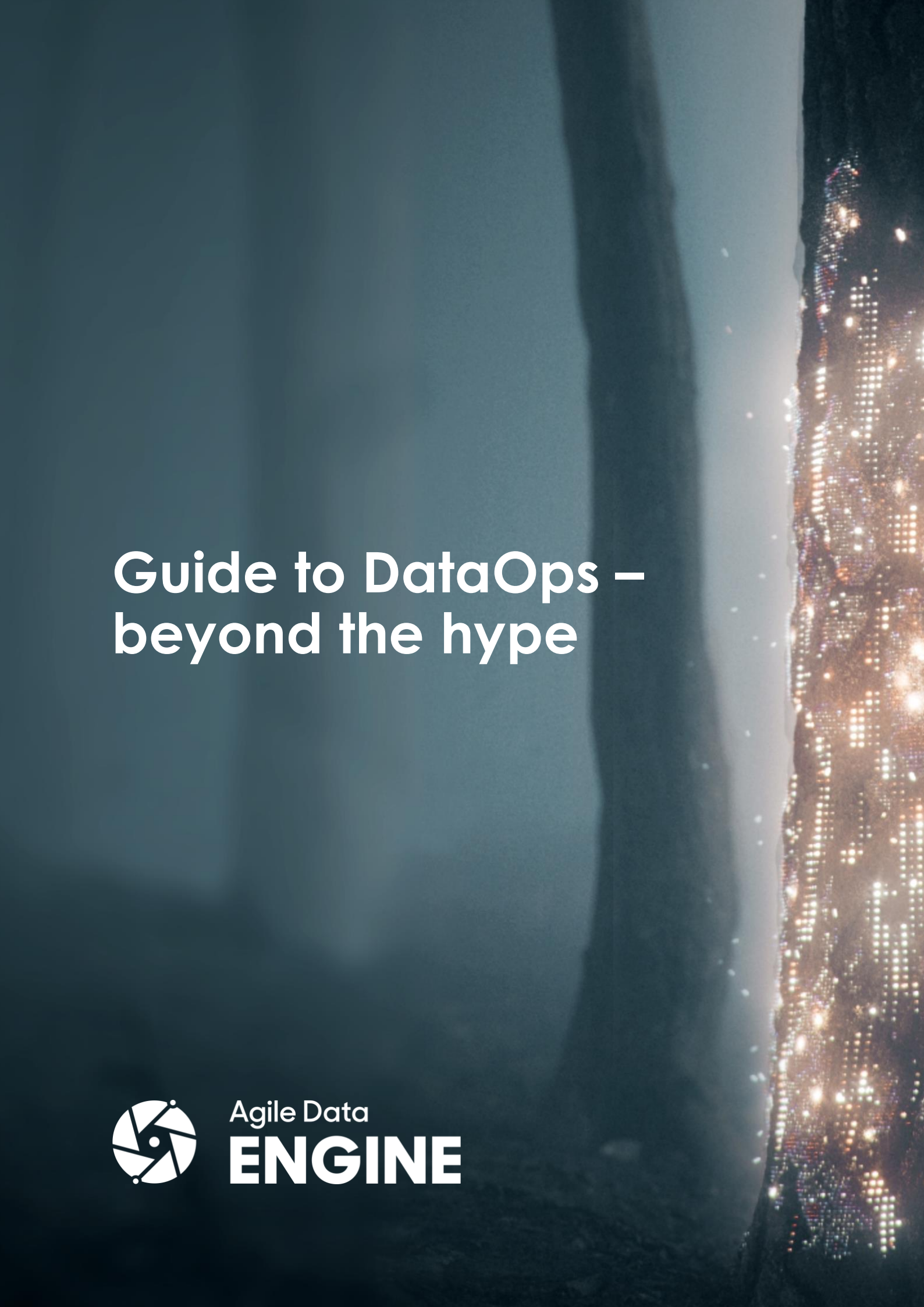
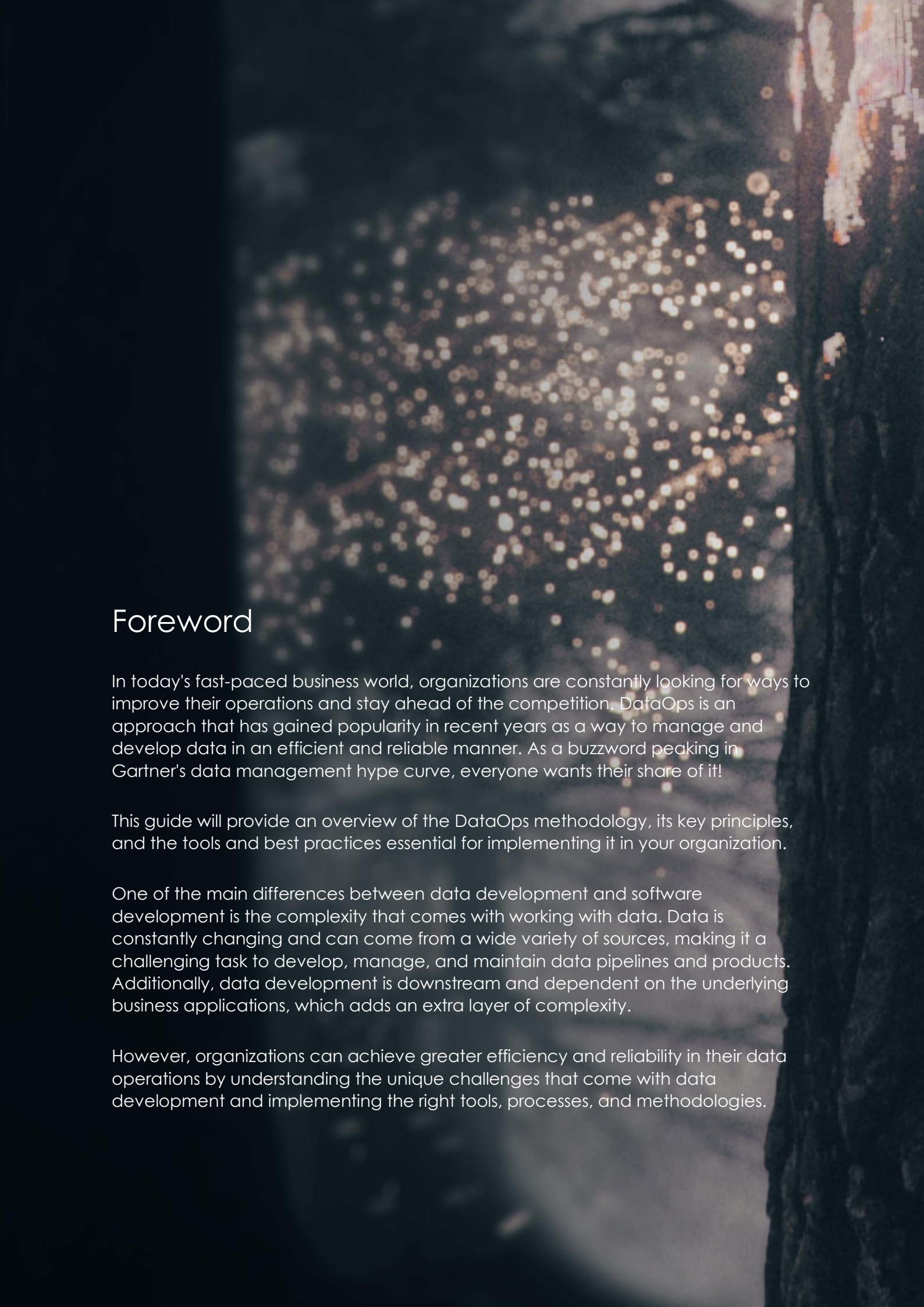


Guide to DataOps – beyond the hype

Agile Data ENGINE

Table of Contents



Foreword

In today's fast-paced business world, organizations are constantly looking for ways to improve their operations and stay ahead of the competition. DataOps is an approach that has gained popularity in recent years as a way to manage and develop data in an efficient and reliable manner. As a buzzword peaking in Gartner's data management hype curve, everyone wants their share of it!

This guide will provide an overview of the DataOps methodology, its key principles, and the tools and best practices essential for implementing it in your organization.

One of the main differences between data development and software development is the complexity that comes with working with data. Data is constantly changing and can come from a wide variety of sources, making it a challenging task to develop, manage, and maintain data pipelines and products. Additionally, data development is downstream and dependent on the underlying business applications, which adds an extra layer of complexity.

However, organizations can achieve greater efficiency and reliability in their data operations by understanding the unique challenges that come with data development and implementing the right tools, processes, and methodologies.

Background – Managing complexity

In recent history, data management has evolved significantly with the shift from traditional on-premise data warehouses to data platforms built on the public cloud. This change is driven by the increasing volume and variety of data, and the need for advanced analytics and AI/ML.

The architecture of a modern data platform has become more complex due to the use of diverse SaaS/PaaS services, and the need to manage the entire data value chain. There are different approaches to face this complexity, some are more managed than others.

This chapter provides background on the changing landscape in data management and data development industry that created the need for DataOps. We'll also very briefly discussed the different approaches different organizations have taken to managing complexity in their architecture.

The evolution of data management

First of all, the real question is: What has changed in Data management in recent history?

Data Platform or Data Warehouse architecture, whatever you want to call it, has evolved quite drastically in the past ten years. We have seen a jump from traditional on-premise data warehouses with "one-size-fits-all" DBMS systems and ETL software to data platforms built on the public cloud leveraging SaaS/PaaS services with endlessly scalable object storage and database management systems designed for huge analytical workloads.

The volume and variety of data have increased exponentially. With it, the need and desire for data and analytics have shifted the landscape from traditional business intelligence to a diverse set of use cases like advanced analytics AI/ML, and all other kinds of data products.

Today, data provides value throughout its whole life cycle independently and not just as a part of a specific business application. This, together with the ability to buy services rather than investing in infrastructure, has made the architecture of a modern data platform more complex.

The role of data governance and architecture in DataOps

DataOps does not really state what kind of architecture you should have. However, by nature, your architecture should support incremental development and have distinct environments for, at least, development and production.

Because of the nature of data platforms, the probability of having a wide variety of data, including sensitive data, in your data platform is quite high. For this reason, especially when working with the public cloud, you should put high-level data governance and architecture principles in place before starting the actual development.

This includes ensuring the security of the data platform by having identity and access management in place and secure network architecture, ensuring the scalability of your architecture, and at least giving a thought for the cost management. Your data platform should also be aligned with the company compliance policies from the beginning.

From a data development perspective, after the big architectural decisions are made, the development phase is mostly about producing new content in the

platform in the form of new data sets and applying different kinds of business rules and requirements to them.

When building new functionalities or the requirements differ from earlier ones, the architecture work can and should be done incrementally like all the rest of the development. When bringing in new data sets, you should always validate compliance requirements against existing ones to ensure everything goes as defined.

One thing to note, especially when operating in the EU, is general data protection regulation. Keep in mind that all GDPR-related data is much more expensive and requires to be handled with care.

It is also a good idea to validate your data platform governance policies with an experienced cloud engineer if the team lacks experience working in the public cloud.

Approaches to managing complexity in the architecture

The variation and complexity in services bought and built have increased the skill set required to develop and operate these platforms and how to manage the whole value chain. To handle the situation, we have at least a couple of different approaches:

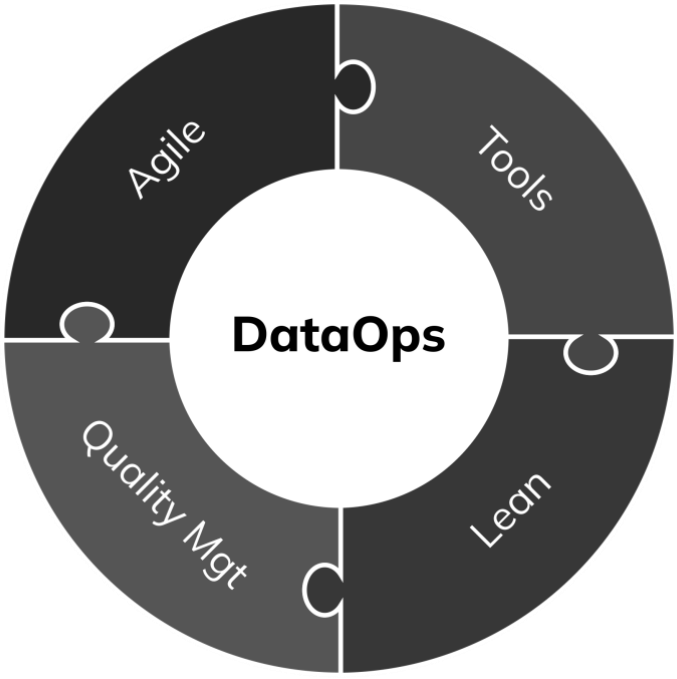
1. **There is no obligation to go to the cloud.** If your data warehouse is purely running financial reporting, for example, and after careful assessment, you can't see any other real requirements for a diverse set of data products now or in the near future, or for any reason you still have to run the on-premise data warehouse anyway, you might not benefit going to the public cloud. In this post, we won't dive deeper into this option.
2. **You can make an informed decision to run 'spaghetti architecture' on your data platform.** This means you give your developers the freedom to create 'quick and dirty' solutions to fulfill the requirements. When refactoring or maintenance is needed, the solution is discarded and redeveloped rather than spent time fixing it. In this post, we won't go into the details of this option either.
3. **Adopt a disciplined and agile way of data development to ensure continuous and predictable value creation and delivery by utilizing DataOps practices and tools in the development and operations of your data platform.** The rest of the post will focus on this option.

On a side note, we want to emphasize that there exists an endless combination of different options, and you should always pay extra attention to privacy and security,

whatever method or approach you choose, especially when going into the public cloud.

Cornerstones of DataOps

DataOps is DevOps adapted to data development and operations. But there are also key differences in the natures of software and data development. In **chapter two**, we will take a closer look at these differences, and how they affect DataOps.



The way we see it, DataOps consists of four main areas: Agile development methods, tools, quality management and Lean.

Chapter three will focus on agile development as the foundation of DataOps.

In **chapter four**, we discuss the DataOps tools and choosing the right ones for your team.

The **fifth** and last **chapter** focuses on quality management and Lean, and how to balance between time to market, performance, and quality.

DevOps or DataOps – What is the distinction between data and software development

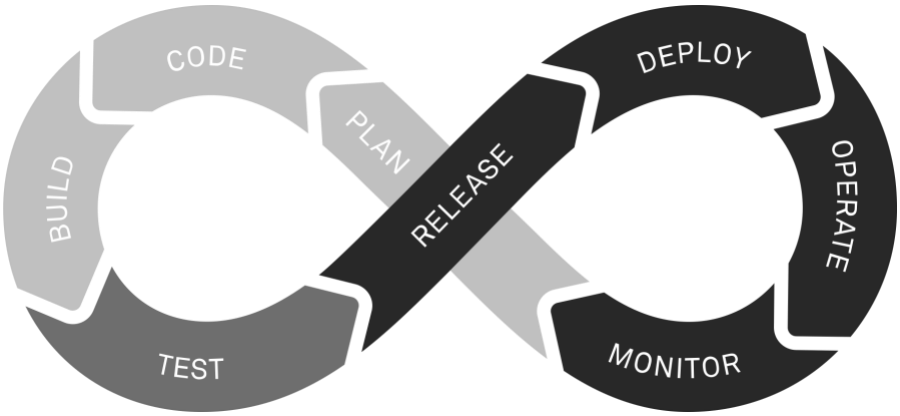
DataOps is a development methodology and set of processes and tools that aim to continuously provide quality data pipelines, products, and analytics in time and with predictable costs.

In this chapter, we will explore the similarities and differences between DataOps and DevOps, and the specific challenges and considerations of data development. We'll discuss the dependency on underlying business applications, the constantly changing nature of data, and the different skill sets required for data engineering and software development.

DevOps: Combining development and operations to improve quality, efficiency, and speed of software delivery

DevOps is a software development methodology that combines processes and practices to improve quality, efficiency, and speed in development, deployment, and delivery while making the software more reliable and the maintenance and operations of the software as smooth as possible.

In other words, DevOps combines or unifies the development and operations. It shifts the thinking from project to product and takes the whole life cycle into account. In practice, this means introducing agile development methods, like Scrum, and automating as much as possible, i.e., testing, version control, CI/CD, etc.



A crucial part of DevOps is measuring the development process and making it leaner. This is done by removing obstacles and waste from the processes and by continuously trying to improve the performance of the development team. Unifying operations in the development team improves the quality of the software as it is in the developers' interest to make the software work well and require as little maintenance and operation work as possible.

One of the core things we should not forget in DevOps comes from agile methods: collaboration and communication. This means there should be clear communication channels between all relevant stakeholders and a continuous feedback loop from end users to developers.

Understanding DataOps: data development and operations

DataOps is not about data and operations, but DevOps adopted to data development. DataOps is a development methodology and a set of processes and tools that aim to continuously provide quality data pipelines, products, and analytics in time and with predictable costs.

To understand how DataOps differs from DevOps, we need to understand how data development differs from software development. In a way, you could oversimplify it like this:

Business applications are about automating and helping with business processes and functions. Data development is about deriving insight from the data created by the said business processes and applications.

Navigating the challenges of upstream data sets

One of the major differences is that data and analytics development is always downstream development and dependent on the underlying business applications, while software development is most commonly focused on the functionalities of a stand-alone business application. Getting the data out and combining it with data extracted from other business applications creates an extra layer of complexity to data development.

You also have to keep in mind that data is constantly changing. In most cases, the upstream business applications do not have integrated development or test data sets, even if the production usually does, to some extent. This creates different challenges for developing the data products as they typically focus on the data content rather than the functionalities.

Understanding the required skillset

Another major difference is that working with data emphasizes a slightly different skill set than software development. As a modern data engineer, it helps to have experience and a decent understanding of programming. However, you should have a firm grasp of SQL and relational databases and some mathematical domains such as set logic and statistics. On top of that, having the required skills and an understanding of data modeling is a huge benefit because data modeling is all about communication and semantics, which is necessary when integrating data from several sources.

That said, the required skill set of data engineers and how you put together your DataOps team heavily depends on the specific needs of your organization.

First of all, you should put some up-front effort into the data platform architecture and consider and analyze at least the following questions:

- How your data platform architecture fits your enterprise architecture?
- What kind of possible DevOps/DataOps tools your organization possibly already has in use?
- what kind of expertise is available in the market?

Agile development – the foundation for enabling DataOps

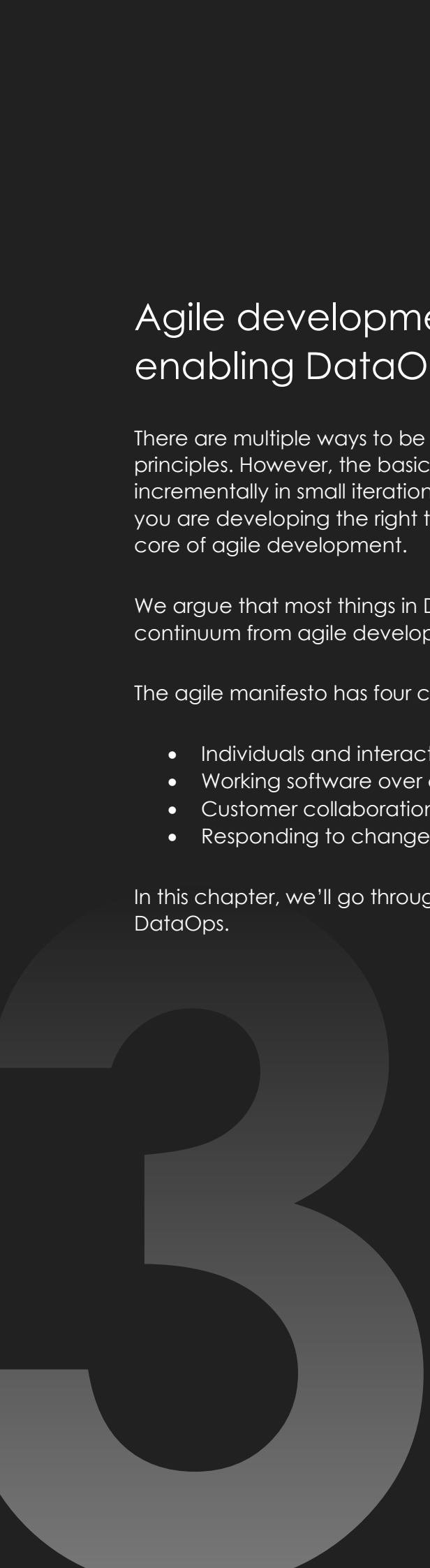
There are multiple ways to be agile, but they all have common values and principles. However, the basic idea for agile development is to do the development incrementally in small iterations, deliver often, and get constant feedback to ensure you are developing the right things. Communication and collaboration are at the core of agile development.

We argue that most things in DevOps or DataOps are just natural evolution or a continuum from agile development.

The agile manifesto has four core statements:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

In this chapter, we'll go through these statements and how they are reflected in DataOps.



Individuals and interactions over processes and tools

When we state that individuals and interactions over processes and tools, we would be really careful not to take that as an instruction not to have the needed processes and tools in place. In DataOps, tools and processes are one of the main contributors to the development. However, a well-performing team only performs well with proper means of internal and external communications. Even the best processes and tools won't help you if the product does not match what was expected.

There is an extra layer of communication in data development as you have to have well-established communication channels to the technical side of the business applications, but also with the subject matter experts and other relevant stakeholders.

Well-performing DataOps team should have all the processes and tools in place. As a matter of fact, we've never seen so disciplined and process-oriented teams than well-organized and performing DataOps teams.

Working software over comprehensive documentation

Development should always focus on producing working software over comprehensive documentation, but this does not mean that the documentation is non-existent. It means that the documentation should be on the correct level.

You should remember that in a correctly organized DataOps team, there is also the 'Ops' part. This means that whoever – within the team – is responsible for the production support or operations of the data platform should be able to sort out the possible issues or at least be able to delegate the problems to the correct address. When you organize your Ops with circulating production support within the team, it will become in their best interest to develop good enough data products that maintaining and running the platform does not mean constantly extinguishing fires. It also means that the documentation has to be on the correct level, or the issues bounce back to the actual developer who built it.

We can – and should – expect a certain level of professionalism from the data developers and engineers. It doesn't mean that the documentation has to be on a level that a random IT guy can sort out the problem based on the documentation. But a data engineer familiar with the technologies used should be.

Customer collaboration over contract negotiation & Responding to change over following a plan

Customer collaboration over contract negotiation and responding to change over following a plan go hand in hand. The basic idea is that the team should provide the output as a service. When required, we should be able to correct the course in the right direction without heavy contract negotiations or hanging on a plan that everyone understands is obsolete. That said, we still should have a plan and contracts and not just start doing random stuff!

Achieving agile success: time, experience, and support from the organization

It takes time and experience to get an agile team to perform well. It also requires new thinking and actions from the surrounding organization. If the relevant stakeholders around your product do not understand or are not able to support agile delivery, it is hard to succeed.

It usually pays off to hire an agile coach for at least a couple of months to get the team on the right track. And when you do so, don't break up the team without a good reason: it always takes time to get a new team to perform well, even when the individuals are experienced.

DataOps tools – understanding and selecting the right stack and approach for your team

When it comes to building a successful DataOps team, one of the crucial factors to consider is the tools and technology used. But with a plethora of options available, how do you decide which tools are the best fit for your organization?

When building your DataOps tool capabilities, there are multiple factors to analyze: Skills, experience, and preferences of the team, requirements of your organization, cost, and – most importantly – what you are trying to achieve!

In this chapter, we have a closer look at how the DataOps tool stack differs from DevOps tools. We also give guidelines on choosing the correct tools.

One tool stack does not fit all

Tools used in DevOps are commonly found in DataOps as well. We have a stack of tools in DataOps that include version control, CI/CD pipelines, automated infrastructure configuration, ETL/ELT tools, orchestration, monitoring, data lineage, and data modeling tools. Choosing the correct tools for your DataOps team should be based on the needs of your organization and level of experience of the people, and what is expected from them.

There is no singular all-encompassing answer to the tooling. Some of the complexity of DataOps comes from the tooling. There is quite a variety of different kinds of technological solutions to organize the stack to help your DataOps team to succeed.

There is a huge amount of open-source software you can leverage to most parts of your DataOps tool stack. Your team could even develop some of the needed tools with a reasonable effort, depending on their experience and skills. Neatly packaged SaaS/PaaS tools and platforms are also available that will combine most of the required tool stack. The real question behind the tooling is: Buy versus build?

In order to answer the question, we need to dive a bit deeper into the differences in the development and operations of software and data.

Leveraging the right mix of software and data warehouse development skills for optimal performance

Tool stack has some variation between DevOps and DataOps, but the bigger difference comes from the skill set differences and experience between software developers and data warehouse developers.

Software developers usually have much broader experience working with and developing the tool stack. It takes time to set up different environments, CI/CD pipelines, etc. The time to set up these naturally vary based on experience working with the chosen tools. It also takes time to master all these tools and to get the development process performing.

You should keep in mind that the tools are only one part of the DataOps methodology. The more tools you have, the more complexity it creates, and the more time it takes the team to master these. People with data management or warehousing backgrounds tend to have less experience with these tools. Usually, you need both software engineering and data warehouse development skills in your

team, and there is demand for a solution that enables a smooth development process no matter which background the developer comes from.

Choosing and maintaining the right tools for DataOps

We can't forget the operations, either. When a business application fails, there might be various reasons behind it, like malfunctioning hardware, incorrect software configuration, insufficient resources, human error, bad design, or problems with security. When data pipelines – or whatever kind of data products you are building – fail, the reason might be any of the above. In addition, failures or changes in the business applications can also be the root cause, as the data platform is downstream and dependent on them.

If you built your DataOps tool stack, you must maintain it yourself in addition to the data platform. If you leverage SaaS/PaaS services on your stack, you outsource the responsibility to the service provider. This, of course, comes with a (subscription) price, but then again, is the engineering work free? If you have multiple DevOps and DataOps teams, then it might pay off to have a centralized team responsible for the tools, but as we are writing this (beginning of 2023), it seems that the demand for people on the market with expertise in DevOps tooling is much higher than the availability.

Either way, there is no definitive answer to this question. It all depends on the people, the complexity of your environment, budget, scalability, and the team's focus, and of course, this all should be aligned with your enterprise architecture.

Quality management in DataOps – balancing between time to market, performance, and quality

Lean development and Total Quality Management are two key principles that are essential to achieving success in DataOps. But how do they work together? And how do they apply to data development?

Getting rid of all the excess waste from the development processes is the core idea of Lean. However, there exists a delicate balance between efficiency and quality of development. As well-defined metrics based on data are the foundation of informed decision-making, we should apply the same standards to the data development as well.

In this chapter, we will explore the intersection of Lean and Total Quality Management in DataOps and how they can help organizations streamline their development processes and ensure high-quality data products.

Combining quality management and lean

Total Quality Management is the often-referred methodology to arrange quality management in DevOps or DataOps. The idea of Total Quality Management is to involve everyone in the quality improvement process and make it a continuous part of the process in the product's life cycle.

We will not go through the details of Total Quality management in this post. Still, we want to point out a couple of key principles that should be applied in data development and operation quality management as well as the actual data product quality management:

- As in Lean, there should be continuous improvement and elimination of waste of the products and processes
- The optimization should be done on the whole process from design to delivery rather than just individual parts of the process
- The decisions on the optimization should be based on data and metrics (how can you even expect to optimize something without understanding it and without proper facts?)

These principles overlap with Lean principles, and from the Lean point of view, we should focus on value by prioritizing the backlog.

We claim that quality management and lean go tightly hand in hand. In DataOps, we can analyze quality management and Lean from two perspectives:

1. What is the quality and how lean is the development process of the data products
2. how lean are the data pipelines, and what is the data quality of the product.

Streamlining the development

From the development perspective, the idea is to continuously measure and analyze the process and make improvements and adjustments accordingly. When talking about the data development process, it is rather ironic if the development processes are not actually analyzed and optimized based on data and metrics. However, the weight on the optimization should not only be on the throughput of the team. As said, on a higher level, there is the quality present as well.

It is not the easiest task to optimize the throughput and quality in a scalable cloud environment where the resources scale and the costs follow. As the engineering

work also has a price tag, it is a delicate balance between time to market, performance, and scalability of the data pipelines and products – not forgetting about the Ops part with the maintenance costs. In the end, what we would like to see is a productive DataOps team and a well-working data platform delivering value with predictable run and development costs. The more standardized development patterns you have in place, the more predictable development and running costs you will have on your platform!

One solution for streamlining or speeding the development is to automate as many parts of the development process as possible. Pattern-based development methodologies and architecture make automation usually more effective. However, you should always pay attention to the effort-gain ratio when starting to automate tasks; if you save one day of work by using ten days to automate the task, is it worth it?

Balancing with the data quality

From the pipeline or product perspective, the focus should be architecturally consistent, robust, and trustworthy data pipelines and products that perform within the requirements. And then there is data quality.

Data quality can, of course, be measured and monitored as part of the DataOps, but data quality issues rarely have anything to do with data development. Mostly the quality issues of the business applications became visible on reports or analytics built on top of the data platform.

Some of the data quality issues can be addressed on the data platform, but you should always keep in mind that the correct place to fix the data quality issues is where the data is produced. The further downstream you go, the higher cost and the less impact the fix has. From the operations perspective, these and all other types of fixes should be done in a way that repairs the whole issue, not just the symptom.

You can read more on data quality management from our whitepaper: **Data Quality Monitoring with Agile Data Engine**.


Agile Data
ENGINE