1ST EDITION

Azure Data and AI Architect Handbook

Adopt a structured approach to designing data and AI solutions at scale on Microsoft Azure

OLIVIER MERTENS
BREGHT VAN BAELEN

Azure Data and AI Architect Handbook

Adopt a structured approach to designing data and AI solutions at scale on Microsoft Azure

Olivier Mertens

Breght Van Baelen

packt

Azure Data and AI Architect Handbook

Group Product Manager: Ali Abidi

Publishing Product Manager: Anant Jain

Content Development Editor: Priyanka Soam

Technical Editor: Devanshi Ayare

Copy Editor: Safis Editing

Project Coordinator: Farheen Fathima

Proofreader: Safis Editing

Indexer: Sejal Dsilva

Production Designer: Prashant Ghare

Marketing Coordinator: Vinishka Kalra

First published: August 2023

Production reference: 1210225

Published by Packt Publishing Ltd.

Grosvenor House

11 St Paul's Square

Birmingham

B3 1RB

ISBN 978-1-80323-486-1

www.packtpub.com

Contributors

About the authors

Olivier Mertens is a cloud solution architect for Azure data and AI at Microsoft, based in Dublin, Ireland. In this role, he assisted organizations in designing their enterprise-scale data platforms and analytical workloads. Next to his role as an architect, Olivier was selected as an advanced cloud expert for AI. In this role as a domain expert, he has led the technical expertise of his field in the corporate markets of Europe, the Middle East, and Africa at Microsoft. Before his time at Microsoft, he worked as a data consultant at Microsoft Partners in Belgium.

Olivier is a lecturer at PXL Digital Business School, a keynote speaker for AI, and holds a master's degree in information management, a postgraduate degree as an AI business architect, and a bachelor's degree in business management.

Breght Van Baelen is a Microsoft employee based in Dublin, Ireland, and works as a cloud solution architect for the data and AI pillar in Azure. He provides guidance to organizations building large-scale analytical platforms and data solutions. In addition, Breght was chosen as an advanced cloud expert for Power BI and is responsible for providing technical expertise in Europe, the Middle East, and Africa. Before his time at Microsoft, he worked as a data consultant at Microsoft Gold Partners in Belgium.

Breght led a team of eight data and AI consultants as a data science lead. Breght holds a master's degree in computer science from KU Leuven, specializing in AI. He also holds a bachelor's degree in computer science from the University of Hasselt.

About the reviewers

Aleksei Zhukov is a fully certified Microsoft data engineer and architect. He has 10+ years of practical experience in different roles closely related to data – mobile network design engineer, data analyst, BI and DWH developer, architect and team lead, DWH product owner, and ETL developer. Different business domains are also covered for professional years (mostly, telecommunication and consulting). He is well known in the Power BI world as one of the top official community contributors.

Aaron Saikovski has over 30 years of commercial information technology expertise, spanning a broad range of technologies and industries. His skill set is centered around software and platform engineering, specifically in the Microsoft Azure platform. He works primarily in the platform, DevOps, and site reliability engineering space. Aaron has a software development background in GoLang, Python, PowerShell, and Bash scripting. He specializes in **Infrastructure as Code** (**IaC**) technologies such as Terraform, Bicep, and ARM templates.

Remon van Harmelen started his career as a software developer. Later, he became intrigued by Azure and made a career change to become an Azure consultant. He worked for several consulting parties and as part of multiple multinational projects, ranging from lift-and-shift migration to creating data analytical platforms. Now working for Microsoft as a cloud solution architect, he spends his days supporting customers on their Azure journeys.

Acknowledgement

This goes out to our highly skilled colleagues whose contributions have had a direct impact on the development of this book: Justin Venter, Diogo Vaz Guedes, Frederic Van Kelecom, Luke Moloney, David Browne and to our managers, Karim Shawki and Gary Keegan, for the continued support.

Table of Contents

1

Advanced Analytics Using AI

AI is transforming businesses across various industries rapidly. Especially with the surge in popularity of **large language models** (**LLMs**) such as ChatGPT, AI adoption is increasing exponentially. Microsoft Azure provides a wide range of AI services to help organizations build powerful AI solutions. In this chapter, we will explore the different AI services available on Azure, as well as the roles involved in building AI solutions, and the steps required to design, develop, and deploy AI models on Azure.

Specifically, we will cover the following:

- The different roles involved in building AI solutions
- The questions a data architect should ask when designing an AI solution
- An overview of Azure AI services, including Azure Machine Learning, Azure AI Model Catalog, and Azure Open AI Service
- An introduction to the concept of MLOps
- A reference architecture for a single-tenant RAG solution in Azure and Azure OpenAI chat

By the end of this chapter, you will have a good understanding of the role of the data architect in the world of data science. Additionally, you will have a high-level overview of what the data scientist and machine learning engineers are responsible for.

Knowing the roles in data science

The Azure cloud offers an extensive range of services for use in advanced analytics and data science. Before we dive into these, it is crucial to understand the different roles in the data science ecosystem.

A well-structured data platform involves multiple professionals who handle various aspects of data processing, from ingestion to raw storage to transformation, data warehousing, and eventually, visualization and dashboarding. The advanced analytics component is more separated from the entire solution, in the sense that most data architectures can perform perfectly without it. This does not take away from the fact that adding advanced analytics such as machine learning predictions can be a valuable enhancement to a solution.

The environment for advanced analytics introduces some new roles. The most prominent are the data scientist and the machine learning engineer, which we will look at in a bit more detail, starting with the following figure. Other profiles include roles such as data labelers and citizen data scientists.

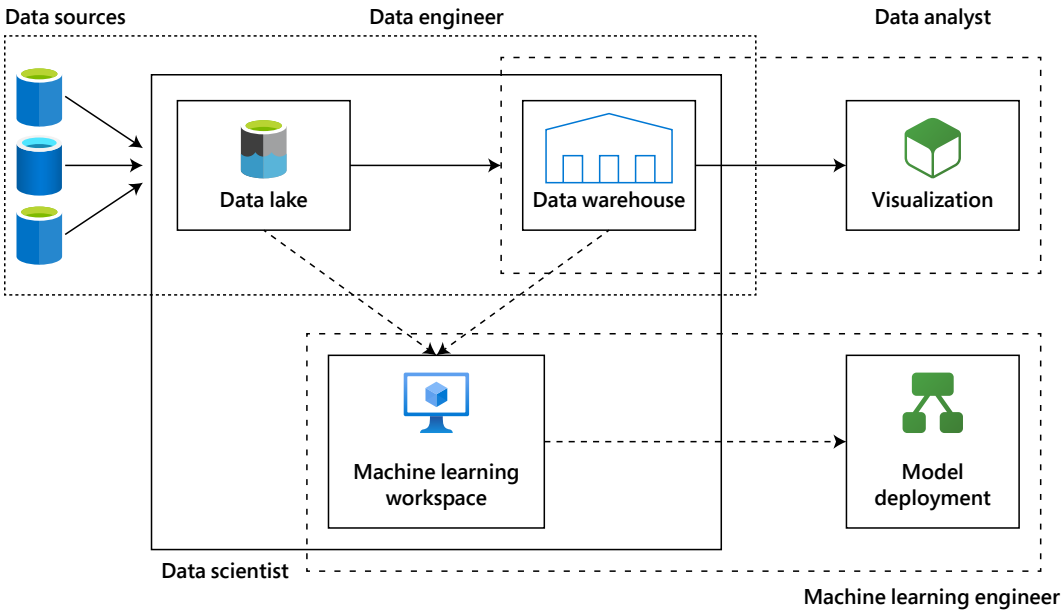

Figure 1.1: An overview of the core components that each data role works with

Figure 1.1 shows a simplified data solution with a machine learning component attached to it. This consists of a workspace to build and train machine learning models and virtual machine clusters to deploy them in production.

The data scientist is responsible for building and training the machine learning model. This is done through experimenting with data, most of the time stemming from the data lake. The data scientist will often use data from the bronze or silver tier in the data lake (i.e., the raw or semi-processed data). Data, in the gold tier or the data warehouse, is often transformed and aggregated in ways that make it convenient for business users to build reports with. However, the data scientist might want to perform different kinds of transformations, which focus more on the statistical relevance of certain features within the data to optimize the training performance of a machine learning model. Regardless, in some cases, data scientists will still interact with the gold layer and the data warehouse to pull clean data for experimentation.

Using this data, data scientists will perform **exploratory data analysis** (**EDA**) to get initial insights into the dataset. This is followed by data cleaning and feature engineering, where features are transformed, or new features are derived to serve as input for the machine learning model. Next, a model is trained and evaluated, resulting in a first prototype. The experimentation does not stop here, however, as machine learning models have hyperparameters that can be adjusted, which might lead to increased performance, while still using the same dataset. This last process is called hyperparameter tuning. Once this is completed, we will arrive at the cutoff point between the responsibilities of a data scientist and a machine learning engineer.

The machine learning engineer is responsible for the **machine learning operations** (**MLOps**). Depending on the exact definition, this usually encompasses the later stages of the machine learning model life cycle. The machine learning engineer receives the finished model from the data scientist and creates a deployment for it. This makes the model available through an API so that it can be consumed by applications and users. In later stages, the model needs to be monitored and periodically retrained, until the end of its life cycle. This is a brief summary, but the MLOps process will be explained in more detail further in this book.

Next, *Figure 1.2* provides an overview of the processes that take place in the MLOps cycle and who the primary contributor to each step is.

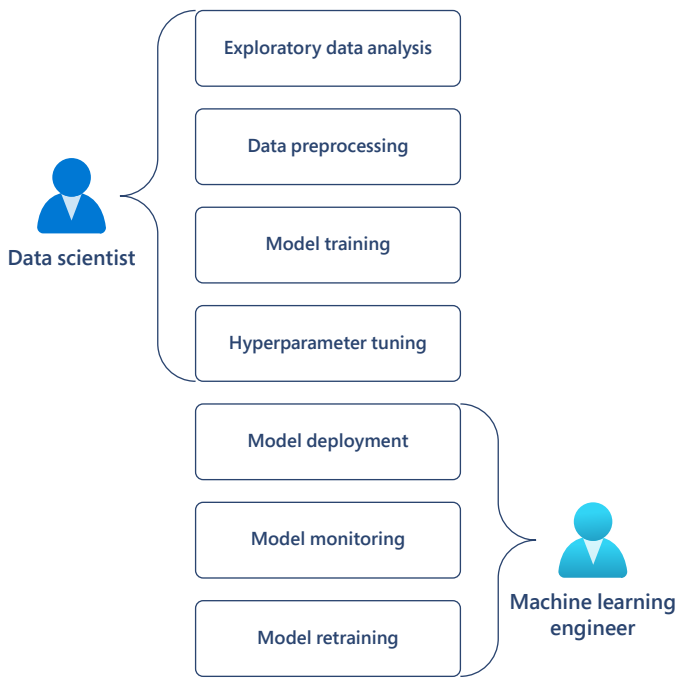

Figure 1.2: The steps of the data science workflow and their executors

Finally, what we are most interested in is the role of the cloud data architect in this environment. First, the architect has to think about the overall AI approach, part of which is deciding whether to go for custom development or not. If custom machine learning model development is involved, the architect will have to decide on a data science environment, or workspace, where the data scientists can experiment.

However, the architect will have more involvement in the work of a machine learning engineer. The optimal working of MLOps is considerably more dependent on good architectural design than the typical prototyping done by data scientists. Here, the architect is responsible for deciding on deployment infrastructure, choosing the right monitoring solutions, version control for models, datasets, code, retraining strategies, and so on.

A lot of the value that an architect brings to machine learning projects comes from design choices outside of the data science suite. The data architect can greatly facilitate the work of data scientists by envisioning efficient data storing structures at the data lake level, with a strong focus on silver (and bronze) tiers with good data quality. Often, extra pipelines are required to get labeled data ready to be picked up by the data scientists.

Designing AI solutions

Generative AI presents **independent software vendors** (**ISVs**) with a unique opportunity to enhance their existing offerings and deliver innovative, unique experiences to their customers. Utilizing the Microsoft cloud ecosystem, ISVs can seamlessly integrate AI capabilities into their applications, ensuring scalability, security, and performance.

Understanding the Microsoft cloud ecosystem

Microsoft cloud offers a comprehensive suite of services, including Azure AI, Microsoft 365, and Microsoft Fabric. This integrated platform enables ISVs to develop and deploy AI-driven applications efficiently, tapping into a vast array of tools and resources. By utilizing these services, ISVs can create AI solutions that are both robust and adaptable to ever-changing market demands.

Approaches to building AI solutions

ISVs can adopt various strategies to incorporate AI into their products, each tailored to specific objectives and resources:

- **Extending Microsoft Copilots**: For ISVs aiming to enhance user experiences within Microsoft applications such as Teams, Word, or Outlook, integrating proprietary data and services into existing Microsoft Copilots is advantageous. This can be achieved by developing plugins or utilizing Microsoft Graph connectors, allowing users to access specialized functionalities directly within familiar interfaces.

- **Creating custom copilots**: ISVs seeking to embed AI assistants within their own applications can utilize tools such as the Microsoft Graph API, Copilot Studio, or the Teams AI Library. They can also use their own custom data connectors. This approach facilitates the development of conversational agents that provide contextual assistance, thereby increasing user engagement and enhancing productivity.
- **Building end-to-end AI experiences**: For comprehensive AI solutions, ISVs can utilize Azure AI and Fabric to construct applications from the ground up. This method offers full control over the AI models and infrastructure, enabling the creation of highly customized solutions that address specific business challenges.

Key considerations in AI solution design

When designing AI solutions, ISVs should focus on several critical aspects when creating custom copilots, extending copilots, or building end-to-end AI experiences for the Microsoft Azure Marketplace:

- **User experience (UX) design**: Ensure that the AI solutions are user-friendly and intuitive by designing interfaces that are easy to navigate and understand. Additionally, make sure they are accessible to users with disabilities by following accessibility guidelines and standards.
- **Data management**: Selecting the appropriate data architecture is crucial. Azure offers various storage and processing options, such as Azure Data Lake, Azure Databricks, Azure Synapse Analytics, and Fabric to handle large datasets effectively.
- **Model deployment and MLOps**: Implementing robust MLOps practices ensures seamless deployment, monitoring, and maintenance of AI models. Azure Machine Learning provides tools to streamline these processes, facilitating continuous integration and delivery of AI solutions.
- **Scalability**: Design AI solutions to handle varying loads efficiently by optimizing algorithms and infrastructure for consistent performance. Utilize scalable infrastructure from Azure to dynamically adjust resources based on demand, ensuring the solution can grow with user needs.
- **Interoperability**: Ensure that AI solutions seamlessly integrate with existing enterprise systems and workflows through APIs, connectors, and other integration tools. Additionally, design them to work across different platforms and devices, providing a consistent experience for all users.
- **Ethical AI**: Implement strategies to identify and mitigate biases in AI models to ensure fair and unbiased outcomes. Enhance user trust and understanding by providing clear explanations of how AI decisions are made.
- **Continuous improvement**: Establish mechanisms for collecting user feedback and continuously improving AI solutions based on this input. Keep AI models and software up to date with the latest advancements and security patches.
- **Cost management**: Carefully plan and manage the budget for developing, deploying, and maintaining AI solutions. Utilize cost management tools offer by Azure to monitor expenses and explore cost-effective options for data storage, processing, and model training.

- **Security and compliance**: Ensuring data privacy and compliance with industry standards is paramount. Utilizing security features in Azure, such as **role-based access control** (**RBAC**) and data encryption, helps protect sensitive information and maintain customer trust.
- **Legal and regulatory compliance**: Ensure compliance with data sovereignty laws, which may require data to be stored and processed within specific geographic regions. Adhere to industry-specific regulations and standards, such as HIPAA for healthcare or GDPR for data protection in the EU.

Utilizing the Microsoft commercial marketplace

To maximize reach and streamline the distribution of AI solutions, ISVs can utilize the Microsoft commercial marketplace, which includes platforms such as Azure Marketplace and Microsoft AppSource. These platforms provide avenues to showcase applications to a global audience, facilitating customer acquisition and growth.

Offer listing options

When publishing AI solutions, ISVs can choose from various listing options to align with their business models:

- **Free Trial**: Allows customers to experience the solution at no cost for a limited period, ranging from 30 days to six months, depending on the offer type. This approach enhances discoverability and enables potential customers to evaluate the solution's capabilities within their specific business scenarios.
- **Contact Me**: Enables customers to reach out directly for more information, fostering personalized engagement and tailored solutions. This option is ideal for complex solutions requiring consultation or customization.
- **Get It Now**: Facilitates immediate purchase and deployment of the solution. This option supports various pricing models, including subscriptions and usage-based pricing, and is suitable for solutions ready for market scaling. Microsoft is responsible for billing and collections.

Best practices for marketplace success

To enhance visibility and customer engagement, ISVs should consider the following best practices:

- **Optimize offer listings**: Craft clear and compelling descriptions that articulate the value proposition, target audience, and key benefits. Utilize relevant keywords to improve searchability and ensure that visuals, such as logos and screenshots, are professional and reflective of the brand.
- **Engage in go-to-market activities**: Utilize various Microsoft marketing resources and programs to co-market solutions, participate in events, and access sales enablement tools. Collaborating with Microsoft can amplify reach and credibility in the marketplace.

- **Gather and act on customer feedback**: Implement mechanisms to collect customer reviews and feedback. Continuous improvement based on user insights fosters customer satisfaction and loyalty, leading to positive word-of-mouth and increased adoption.

By thoughtfully designing AI solutions and strategically utilizing the Microsoft cloud ecosystem and commercial marketplace, ISVs can effectively meet customer needs, expand their market presence, and drive business growth.

Is AI the right solution?

This can be further refined to the necessity of an inductive solution, compared to a deductive one. Business rulesets are deductive; machine learning is inductive. Business rules will provide you with a solid answer if the condition for that rule is met. Machine learning models will provide you with answers that have a high probability but not certain ones.

The big advantage of machine learning is its ability to cover cases in a much more granular manner, whereas business rules must group various cases within a single condition so as to not end up with an absurd or even impossible number of rules. Look at image recognition, for example. Trying to make a rule set for every possible combination of pixels that might represent a human is simply impossible. Knowing this, evaluate the proposed use case and confirm that the usage (and correlating costs) of AI is justified for this solution.

Do we opt for pre-trained models or a custom model?

Although this question is more focused on implementation than qualification, it is crucial to answer it first, as it directly impacts the following two questions. As with many aspects of IT, the key is not reinventing the wheel. Does your use case seem generic or industry-agnostic? If so, there are probably existing machine learning models, often offering far superior performance (general knowledge-wise) than your own data could train a model to have. Companies such as Microsoft and partners such as OpenAI invest heavily in getting these pre-trained models to cutting-edge standards.

It may be that the solution you want to create is fairly generic, but certain aspects make it a bit more niche. An example could be a text analytics model in the medical industry. While text analytics models are great at the general skill of language understanding, they might have some issues with grasping the essence of industry-specific language out of the box. In this case, an organization can provide some of its own data to fine-tune the model to increase its performance on niche tasks, while maintaining most of the general knowledge from its initial training dataset. Most pre-trained AI models on Azure, available in Azure AI services and Azure OpenAI Service, are fine- tuneable. When out-of-the-box models are not an option, then we need to look at custom development.

Pre-trained models

Pre-trained models are AI models that have been trained on large datasets and are ready to use for various applications. They offer a quick and cost-effective way to implement AI without requiring extensive development effort

Advantages of pre-trained models

Pre-trained models provide several benefits, making them a great choice to implement AI solutions without the complexities of building models from scratch:

- **Cost-effective**: Pre-trained models are generally more affordable as they eliminate the need for extensive data collection and model training.
- **Fast implementation**: These models can be quickly deployed, allowing businesses to start utilizing AI capabilities without significant delays.
- **Proven effectiveness**: Pre-trained models have been tested and validated on large datasets, ensuring a certain level of reliability and performance.
- **Minimal resource requirements**: They require less computational power and technical expertise compared to building custom models from scratch.
- **Easier maintenance**: Updates and improvements to pre-trained models are often managed by the provider, reducing the maintenance burden on your team.

Drawbacks of pre-trained models

While pre-trained models offer convenience and cost savings, they also come with certain limitations that may impact their effectiveness in specialized applications:

- **Limited customization**: Pre-trained models may not perfectly align with your specific business needs, leading to suboptimal performance in niche tasks.
- **Lower accuracy for specialized tasks**: They might not perform as well on domain-specific problems where specialized knowledge is required.
- **Limited flexibility**: There is less control over the model architecture and training data, which can be a limitation for highly specialized applications.
- **Reliance on external providers**: Dependence on third-party providers for updates and support can be a drawback if the provider's priorities change.

Custom models

Custom AI models are built and trained specifically for a business's unique needs. They require more investment in time and resources but offer greater control and adaptability.

Advantages of custom models

For businesses with specialized requirements, custom AI models provide significant advantages by offering tailored solutions and enhanced performance:

- **Tailored to specific needs**: Custom models are designed to address specific business challenges, ensuring higher relevance and accuracy.
- **Better performance for niche tasks**: They can be fine-tuned to excel in specialized tasks that pre-trained models might struggle with.
- **Competitive advantage**: Custom models can provide unique capabilities that differentiate your business from competitors.
- **Adaptability and scalability**: These models can be continuously improved and scaled to meet evolving business requirements.
- **Enhanced data control**: You have full control over the data used for training, ensuring it is relevant and high-quality.

Drawbacks of custom models

Despite their advantages, custom models come with challenges that businesses must carefully consider before investing in development:

- **High costs**: Developing custom models can be expensive due to the need for specialized expertise, computational resources, and extensive data collection.
- **Long development time**: Building and fine-tuning custom models can be time-consuming, delaying the deployment of AI solutions.
- **Requires technical expertise**: Significant technical knowledge and experience are required to develop and maintain custom models.
- **Data challenges**: Ensuring the availability of clean, well-labeled, and relevant data can be a major hurdle.
- **Maintenance and updates**: Ongoing maintenance and updates are necessary to keep the model performing optimally, which can be resource intensive.

Key considerations

When deciding between pre-trained and custom models, consider the following factors:

- **Business objectives**: Align the choice with your specific business goals and the problems you aim to solve.
- **Resource availability**: Assess the availability of financial, technical, and human resources to support the chosen approach.

- **Time constraints**: Consider the urgency of deploying the AI solution and the time required for development.
- **Data availability**: Evaluate the quality and quantity of data you have for training custom models.
- **Scalability needs**: Determine the scalability requirements of your AI solution and how each approach meets those needs.

Is data available?

If we opt for custom development, we will need to bring our own data. The same goes for fine-tuning an existing model, though to a lesser extent. Is the data that we need available? Does an organization have a significant volume of historical data stored already in a central location? If this data is still spread across multiple platforms or sources, then this might indicate it is not the right time to implement AI. It would be more valuable to focus on increased data engineering efforts in this situation. In the case of machine learning on Azure, data is ideally stored in tiers in Azure Data Lake Storage.

Keep in mind that machine learning model training does not stop after putting it into production. The performance of the production model will be constantly monitored, and if it starts to drift over time, retraining will take place. Do the sources of our current historic data still generate an adequate volume of data to carry out retraining?

There is still a common misunderstanding that large volumes of data are a necessity for any high-performant model. It's key to know here that even though the performance of a model still scales with the amount of training data, more and more new techniques have been developed to allow for valuable performance levels to be reached with a limited data volume.

Is the data of acceptable quality?

Just like the last question, this only counts for custom development or fine-tuning. Data quality between sources can differ immensely. There are different ways in which data can be of bad quality. Some issues can be solved easily; others can be astonishingly hard. Some examples of poor data quality are as follows:

- **Inaccurate data**: This occurs when data is incorrect or contains errors, such as typos or missing values. This is not easy to solve and will often result in fixes required at the source.
- **Incomplete data**: This occurs when data is missing important information or lacks the necessary details to be useful. In some cases, data scientists can use statistics to impute missing data. In other cases, it might depend on the specific model that is being developed. Certain algorithms can perform well with sparse data, while others are heavily affected by it. Knowing which exact algorithms should not be in the scope of the architect but, rather, the data scientists.
- **Outdated data**: This occurs when data is no longer relevant or useful due to changes in circumstances or the passage of time. If this data is statistically dissimilar to data generated in the present, it is better to remove this data from the training dataset.

- **Duplicated data**: This occurs when the same data is entered multiple times in different places, leading to inconsistencies and confusion. Luckily, this is one of the easiest data quality issues to solve.
- **Biased data**: This occurs when data is influenced by personal biases or prejudices, leading to inaccurate or unfair conclusions. This can be notoriously hard to solve and is a well-known issue in the data science world. We will come back to this later when discussing responsible AI.

This concludes the qualifying questions of AI implementation. One remaining factor to consider is the **return on investment** (**ROI**), however, before calculating the investment, we need to have more knowledge on the exact implementation. This will be the focus of the next set of questions.

Low code or code first?

The answer to which approach should be chosen depends on people, their skill sets, and the complexity of the use case. In the vast majority of cases, code-first solutions are preferred, as they come with considerably more flexibility and versatility. Low-code simplifies development significantly, often by providing drag-and-drop interfaces to create workflows (or, in this case, machine learning pipelines). While low-code solutions often benefit from rapid development, this advantage in speed is slowly shrinking. Due to advancements in libraries and packages, generic code-first models can now be developed in a shorter amount of time than before.

While code-first solutions cover a much broader set of use cases, they are simply not feasible for every organization. Data scientists tend to be an expensive resource and are often in high demand, with competition due to a limited supply in the labor market. Luckily, low-code platforms are advancing rapidly to address this issue. This allows citizen data scientists (non-professionals) to create and train machine learning models easily, although the performance will still be inferior compared to professional code-first development.

As a rule of thumb, if a professional data science team is available and it has already been decided that custom development is the way forward, choose a code-first solution.

What are the requirements for the AI model?

Now, we will dive deeper into the technicalities of machine learning models. Note that not all answers here must come from the data architect. It is certainly a plus if the architect can think about things such as model selection with the data scientists, but it is not expected of the role. Leave it to the data science and machine learning team to have a clear understanding of the technical requirements for the AI model and allow them to utilize their expertise.

The minimum accepted performance is probably the most straightforward requirement. This is a defined threshold on the primary metric of a model, based on what is justifiable for the use case to progress. For instance, a model might need to have a minimum accuracy of 95% to be economically viable and continue toward production.

Next, latency is an important requirement when the model will be used to make real-time predictions. The larger the model and the more calculations that need to happen (not counting parallelism), the longer it will take to make a prediction. Some use cases will require a prediction latency within milliseconds, which can be achieved with lightweight model selection and specialized infrastructure.

Another requirement is the size of the model, which directly relates to the hosting costs when deployed into production, as the model will have to be loaded in RAM while the deployment runs. This is mostly a critical requirement for IoT Edge use cases, where AI models are deployed on a small IoT device and make predictions locally before sending their results to the cloud. These devices often have very limited memory, and the data science team will have to figure out the most efficient model that can fit on the device.

With the rapidly growing adoption of LLMs, such as the GPT-model family, power consumption has become an increasingly critical concern. Years ago, this was a negligible topic in most use cases, but with the massive size of today's cutting-edge models, it is now unavoidable. Whether these models are hosted privately or in the cloud, power consumption incurs costs directly or indirectly. For natural language use cases specifically, consider whether the traditional (and significantly cheaper) text analytics models in Azure AI services can do the job effectively before heading straight for LLMs.

Batch or real-time inferencing?

When a model is finished and ready for deployment, the architect will have to decide on the type of deployment. On a high level, the choice is between batch scoring or predicting in real time. Typically, when machine learning predictions are used to enrich data already being batch- processed in an OLAP scenario, the machine learning model can perform periodical inferencing on large batches. The model will then be incorporated as an extra transformation step in the ETL pipeline. When using machine learning models in applications where users expect an instant prediction, real-time endpoints are required.

When deploying our model to an endpoint, the architecture might differ based on the type of inferencing, a topic we will look into in greater detail later in this chapter.

Is explainability required?

Explainable AI (**XAI**) has been on the rise for quite a while now. For traditional machine learning models, it was relatively easy to figure out why a model came to which conclusion, through statistical methods such as feature importance. With the rise of deep learning models, which are essentially black-box models, predictions have become increasingly difficult to explain.

Techniques have been developed to approximate the decision-making process of a black-box model. For instance, in the case of the mimic explainer, a traditional (and by nature interpretable) machine learning model is trained to mimic the black-box model and extract things, such as feature importance, from the mimic model. However, this remains an approximation rather than a definitive explanation.

Therefore, it is key to figure out how crucial explainability is for the use case. In cases that significantly affect humans, such as predicting credit scoring using AI, interpretability is a must. In cases with minimal or no impact on human lives, interpretability is more of a nice-to-have. In this instance, we can opt for a black-box model if this provides increased predictive performance.

What is the expected ROI?

Once the key qualifying questions have been answered and decisions have been made to fulfill technical requirements, we should have sufficient information to calculate an estimated ROI. This will be the final exercise before giving the green light to start implementation, or at least the development of a proof of concept.

If we know which approach to use, which model to train, and which type of deployment to use, we can start mapping it to the right Azure service and perform a cost calculation. The expected costs are then compared to the projected added value of a machine learning model.

Optimal performance of a machine learning model

As a side note to calculating the ROI, we need to have an idea of what the optimal performance level of a machine learning model is. This is where the academic and corporate worlds tend to differ. Academics focus on reaching the highest performance levels possible, whereas business will focus on the most efficient ratio between costs and performance. It might not make sense for a business to invest largely in a few percent increase in performance if this marginal increase is not justified by bringing adequate value to compensate.

Understanding AI on Azure

Microsoft has a globally leading role in terms of cloud-based AI. This is all thanks to performant infrastructure, strategic partnerships, and heavy investment in machine learning services.

The AI offering on Azure can be classified into two distinct categories:

- Pre-trained AI models
- Workspace for data scientists and machine learning engineers

The first contains a range of ready-to-use and pre-trained AI models that can be quickly implemented (and combined), allowing for an innovative way to process unstructured data or enhance applications with machine learning features. The latter provides the environment for a data science team to create their own custom models and maintain them throughout their life cycle.

The pre-trained models are, presently, available in three services:

- Azure AI services
- Azure AI Model Catalog
- Azure OpenAI Service

Azure AI services is a collection of models meant to mimic most human functionalities, which can be used in various ways. The Azure OpenAI Service, a result of the close collaboration between Microsoft and its partner OpenAI, encompasses large-scale models to generate natural language, code, and images.

For machine learning workspaces, the go-to option is Azure Machine Learning. Azure Machine Learning provides an Azure-native data science environment, featuring a complete MLOps framework and many built-in connectors to other Azure resources. Azure Databricks can be an alternative for data science teams who prefer Spark, although Azure Machine Learning is increasingly getting Spark capabilities as well, due to integrations with Azure Synapse Analytics. Whereas a full MLOps framework comes as a built-in feature of the Azure Machine Learning SDK, Databricks uses the open source MLflow framework to deliver MLOps functionalities. The MLOps functionalities refer to things such as model and dataset versioning, machine learning pipeline management, model life cycle management, and model deployment (monitoring).

Let's dive deeper into each of the key AI services on Azure.

Azure AI services

Azure AI services holds many pre-trained models, covering numerous industry-agnostic use cases. It is divided into three categories:

- Speech
- Vision
- Language

All these models are readily available as API endpoints in a serverless solution. This means costs are only incurred for requests sent to the model. Hosting costs are not charged unless an organization chooses a private deployment, which is possible with a subset of the models.

Speech

Azure AI Speech offers several models that can be used for speech-to-text, text-to- speech, speech translation, and speaker recognition.

The **speech-to-text** (**STT**) model is used to transcribe speech, enabling easy transformation from audio data into text. It supports real-time and batch transcription, speaker diarization, and custom language models. Real-time transcribing is convenient to add instant subtitles to any video stream, while batch transcription benefits call centers or interviewers, allowing them to turn all their recordings into text, which can then be further processed by text analytics models.

Speaker diarization

Speaker diarization is the ability to distinguish speech from multiple speakers (at the same time). This can be useful to segment parts of a transcription of a business meeting, for example. When combined with a speaker recognition model, which is also available in Azure, we can instantly transcribe who said what in a conversation.

The **text to speech** (**TTS**) model generates natural-sounding text-to-speech voices. It supports standard and neural TTS voices, custom voice creation, and **speech synthesis markup language** (**SSML**) for advanced customization. With SSML, users can adjust certain features of the synthetic voice if the default pronunciation is not satisfactory. It allows scaling pitch, adjust the speaking rate and volume, change the pronunciation and emphasis, and so on. TTS is often used at scale when creating audiobooks and the like. Nowadays, a lot of solutions are also popping up where TTS is used on the output of generative language models to quickly create a personal AI (home) assistant.

The **speech translation** model builds upon other models and is used to translate spoken audio in real time. It can output either text or audio and is, like the STT and TTS models, able to be fine-tuned. Among other capabilities, this technology allows the creation of solutions that can do real-time translation during voice or video calls.

The **speaker recognition** model identifies and verifies the speaker in a conversation and also supports fine-tuning. As previously mentioned, it is great to use in combination with the speaker diarization features of the STT model. This is perfect for use cases such as interview audio transcribing, where we want to automatically distinguish an interviewer and interviewee in the generated transcription.

Vision

Azure AI Vision entails models such as Azure AI Custom Vision, image analysis, spatial analysis, facial recognition, and optical character recognition.

Custom Vision is a core component of the Vision models in Azure AI services. It allows for convenient training of computer vision models, used for image classification or object detection. Organizations can simply upload their own data and train a computer vision model without a letter of code. The service also has an environment to label images, with a machine learning model in the background that will suggest labels after a while, which significantly speeds up the process as labeling efforts continue.

Beyond Custom Vision, there is a range of models used for **image analysis**. The latest Florence foundation model understands over 10,000 concepts and objects, which it can detect in the images or classify into categories. With this knowledge, it is also capable of auto-generating alt-text for pictures (a brief description of what is shown). This can also be useful in cases where we want to have images as input for text-based foundation models (such as the GPT models). In this scenario, we can use the caption or description of an image as input for a language model.

Spatial analysis models are used to detect people or vehicles and track their movements in 3D space. These models tend to go very well with CCTV footage, for either security monitoring business intelligence, such as estimating queue lengths in a store or airport. Another example is (near)-collision detection for traffic cameras by detecting the speed and distance between two vehicles, in order to map highly dangerous road segments.

Facial recognition models are also included in Azure AI services and are capable of recognizing facial features and linking them to a person. However, this model is gated due to responsible AI guidelines from Microsoft, just like the models in the Azure OpenAI Service, as we will see later in the section on this service. As this model could be used for harmful actions at scale, access needs to be requested through a special application process before the model can be used.

Lastly, **optical character recognition** (**OCR**) models can automatically locate and extract text in images. It is a great way of turning images into text and can be valuable in cases of document digitalization. Since the latter has become a very popular use case, Microsoft has developed Form Recognizer, a service built on top of OCR to automatically extract key-value pairs from images of documents or PDF files. This way, the text in a document is extracted and instantly organized to conform to the document's structure.

Language

The core language models include the following:

- Entity recognition
- Sentiment analysis
- Question answering
- Conversational language understanding
- Translation

The **entity recognition** models provide the ability to quickly extract concepts and objects from text data. A more specific version, **named entity recognition** (**NER**) can extract the names of people and places from text. These models can be useful detecting personally identifiable information in documents or extracting key stakeholders from lengthy contracts.

Sentiment analysis models can understand the sentiment of a given phrase. When passing large chunks of text data, a sentiment score is returned on a phrase-by-phrase basis. This can determine whether a text (especially reviews) is written in a positive or negative way. When used in combination with entity recognition, it is possible to extract sentiments on different concepts, such as different products in a single review. A restaurant can use something like this to determine which dishes are favored and which could use improvement.

Question-answering models enable conversational layer over knowledge bases such as FAQs or other documents. It is one of the components to create a custom chatbot. This is used in combination with a **conversational language understanding** model that focuses on interpreting the goal of a sentence or command, along with other types of information. Nowadays, however, many of these models are being outperformed by highly capable LLMs, but they still provide a relatively cheap alternative.

The **translator** model is able to translate texts in over 100 languages and dialects, with the option to fine-tune very domain-specific data.

With the rapid rise of LLMs such as the GPT family—available via the Azure OpenAI Service—it is unclear what the future holds for the language models in Azure AI services. Although the performance of LLMs is arguably better, they require a lot more memory and power to run. Therefore, the language models in Azure Cognitive Services still provide a cheaper and more energy-efficient method of processing natural language text data.

Azure AI Model Catalog

The Azure AI Model Catalog offers a comprehensive and diverse selection of pre-trained AI models, allowing ISVs to seamlessly integrate cutting-edge AI capabilities into their applications. By leveraging these models, you can significantly reduce time-to-market, eliminate the need for extensive model training, and enhance your AI-powered solutions with state-of-the-art technology for text, image, video, and time-series applications.

Diverse selection of pre-trained AI models

The catalog features a broad array of industry-leading AI models that cater to various use cases, including **natural language processing** (**NLP**), computer vision, **retrieval-augmented generation** (**RAG**), generative AI, and time-series forecasting. These models have been optimized for different levels of performance, scalability, and cost efficiency, ensuring that ISVs can find the right fit for their unique application requirements.

The catalog includes models from renowned AI providers, such as:

- **OpenAI**: A leader in foundation models, OpenAI provides some of the most advanced models for text, image, and video processing. This includes the widely used **GPT models**, which have demonstrated exceptional performance across multiple benchmarks.
- **Phi** : Designed for cost-effective AI application development, Phi offers **small language models** (**SLMs**), delivering superior latency and lower deployment costs.
- **Meta**: Open-source LLMs ranging from 7 billion to 70 billion parameters, allowing ISVs to integrate robust, scalable AI solutions into their applications.
- **Mistral AI**: Known for state-of-the-art reasoning performance, Mistral's models are designed to enhance critical thinking and analytical tasks in AI-powered applications.

- **Cohere**: A leading AI provider focused on RAG, enabling search capability and knowledge retrieval on both structured and unstructured data alike.
- **Hugging Face**: One of the largest AI model repositories, Hugging Face offers thousands of pre-trained models for various applications, from text generation to image analysis.
- **Nixtla**: Specializing in **time-series forecasting**, Nixtla provides transformer-based models that help businesses improve predictive analytics and trend forecasting.
- **Core42**: Focused on Arabic NLP, Core42's models enable applications tailored for Arabic-speaking markets.
- **Stability AI**: A leader in **text-to-image** and **video generation**, Stability AI provides creative models that power content generation and artistic applications.
- **NTT Data**: Offers a bilingual Japanese and English small language model, enabling ISVs to build cross-language AI applications with enhanced contextual understanding.
- **AI21**: A provider of **foundation chat completion models**, AI21 delivers enterprise-grade conversational AI models that enhance chatbot and virtual assistant functionalities.

Advantages of using the Azure AI Model Catalog

By integrating pre-trained models from the Azure AI Model Catalog, ISVs gain access to powerful AI capabilities without the need to train their own models or develop custom API integrations. Some of the key benefits include:

- **Seamless integration**: ISVs can quickly deploy and test AI models via **API-based access**, ensuring faster implementation and development cycles.
- **Reduced development costs**: Using pre-trained models eliminates the need for intensive computing resources, making AI more accessible and cost-efficient.
- **Scalability and flexibility**: The catalog provides a variety of models optimized for different tasks at different price points, enabling businesses to scale AI solutions based on performance and cost considerations.
- **Access to open source models**: Many models within the catalog are open-source and available to use for commercial use cases, giving ISVs the freedom to innovate and customize AI applications at very little cost.
- **Enterprise-ready AI**: From conversational AI to AI-enhanced search to time series forecasting, the Azure AI Model Catalog empowers ISVs to build solutions that drive business value and user engagement with security and privacy treated as first-class citizens.

Whether enhancing search capabilities, generating content, improving business forecasting, or building intelligent chatbots, you can utilize the best-in-class models available through Azure AI Model Catalog. With scalability, cost-efficiency, and enterprise-grade performance, the catalog ensures that you have the tools you need to drive AI innovation at scale.

Azure OpenAI Service

OpenAI's innovations, especially with GPT models and DALL-E, have revolutionized AI research and applications. Microsoft has invested in OpenAI, with the partnership utilizing Azure infrastructure to deploy OpenAI models. In this partnership, OpenAI focuses on research and innovation, developing new models and iterating on existing ones. Microsoft handles the enterprise-scale go-to-market strategy, offering robust infrastructure, technical guidance, and reliable SLAs. This enables large organizations to integrate, fine-tune, and deploy OpenAI models on their own data, as well as manage private instances of these models.

At the time of writing, Azure OpenAI Service offers access to the following models:

Models	Description
o-series models	Reasoning models designed for advanced problem-solving with enhanced focus and capabilities.
GPT-4o, GPT-4o mini, and GPT-4 Turbo	The most capable Azure OpenAI models, including multimodal versions that can process both text and images as input.
GPT-4o audio	Models optimized for low-latency, "speech in, speech out" conversational interactions or audio generation.
GPT-4	A set of models that enhance GPT-3.5, capable of understanding and generating natural language and code.
GPT-3.5	A set of models that improve upon GPT-3, capable of understanding and generating both natural language and code.
Embeddings	Models that convert text into numerical vector representations, enabling tasks like text similarity and semantic search.
DALL-E	Models that generate original images based on natural language descriptions.
Whisper	Preview models designed for speech-to-text transcription and translation.
Text to speech (Preview)	Preview models that synthesize text into natural-sounding speech.

Table 1.1: Models offered by Azure OpenAI Service

Azure OpenAI deployment types

Azure OpenAI offers different deployment options to accommodate various business needs, ensuring flexibility in performance, scalability, and compliance. Selecting the right deployment type helps optimize resource utilization and model efficiency.

- **Multi-tenant deployments**: This is the default deployment type where multiple customers share Azure OpenAI's infrastructure. It provides access to pre-deployed models, reducing the need for manual setup and management. This deployment is cost-effective, scalable, and maintained by Microsoft, making it suitable for general AI applications where high availability and security are key.

- **Provisioned throughput deployments**: This deployment provides dedicated AI capacity for businesses that require consistent and predictable performance. By allocating a fixed portion of Azure AI resources, it eliminates resource contention and ensures stable response times. It is ideal for enterprise applications with strict performance requirements, such as large-scale AI workloads or real-time processing needs.
- **On-premises and air-gapped deployments (future)**: Microsoft is developing deployment options that allow Azure OpenAI models to run in on-premises or fully isolated environments. These deployments are designed for organizations operating in highly regulated industries, such as government or defense, where cloud-based AI services may not be viable due to data privacy and security restrictions.

Each deployment type offers different levels of control, scalability, and security. Multi-tenant deployments work well for general use, provisioned throughput deployment is best for businesses needing consistent performance, and air-gapped solutions will cater to industries with strict regulatory requirements.

Table 1.3 gives a brief overview of which deployment to choose depending on your use case.

Deployment type	Best for
Multi-tenant	Startups, developers, and general AI applications
Provisioned throughput	Businesses needing consistent, scalable AI performance
Air gapped (upcoming)	Industries with strict data privacy and security regulations

Table 1.2: Choosing a deployment type

Fine-tuning and private deployments

As a data architect, it is important to understand the cost structure of these models. The first option is to use the base model in a serverless manner. Similar to how we work with Azure AI services, users will get a key for the model's endpoint and simply pay per prediction. For DALL-E, costs are incurred per 100 images, while the GPT and Codex models are priced per 1,000 tokens. For every request made to a GPT model, all tokens of the input prompt and the output are added up to determine the cost of the prediction.

Tokens

In natural language processing, a token refers to a sequence of characters that represents a distinct unit of meaning in a text. These units do not necessarily correspond to words, although for short words, this is mostly the case. Tokens are used as the basic building blocks to process and analyze text data. A good rule of thumb for the English language is that one token is, on average, four characters. Dividing your total character count by four will make a good estimate of the number of tokens.

Azure OpenAI Service also grants extensive fine-tuning functionalities. Up to 1 GB of training data can be uploaded per Azure OpenAI instance for fine-tuning. This may not sound like a lot but note that we are not training a new model from scratch. The goal of fine-tuning is to retrain the last few layers of the model to increase performance on specific tasks or company-specific knowledge. For this process, 1 GB of data is more than sufficient.

When adding a fine-tuned model to a solution, two additional costs apply. On top of the token-based inference cost, we need to consider both training and hosting costs. The hourly training cost can be quite high due to the amount of hardware needed but compared to the inference and hosting costs during a model's life cycle, it remains a relatively minor expense. Next, since we are not using the base model anymore and, instead, our own "version" of the model, we will need to host the model ourselves, resulting in an hourly hosting cost.

Now that we have covered both pre-trained model collections, Azure AI services, and Azure OpenAI Service, let's move on to custom development using Azure Machine Learning.

RAG for LLMs

For ISVs incorporating LLMs into their applications, **retrieval-augmented generation (RAG)** is a powerful technique to enhance accuracy and relevance. Instead of fine-tuning a model, RAG retrieves relevant data from an external knowledge base and injects it into the prompt. This ensures responses are grounded in authoritative, up-to-date information without requiring expensive retraining.

Unlike traditional LLM implementations that rely solely on pre-trained knowledge, RAG dynamically retrieves external, real-time data before generating a response. This ensures that the AI outputs remain up-to-date, contextually accurate, and grounded in authoritative sources. Best of all, you can use out-of-the-box LLMs without having to spend all the time, money and effort fine-tuning.

How RAG works in Azure

Microsoft Azure provides a scalable and robust ecosystem for implementing RAG, leveraging cloud-native databases and search capabilities to retrieve and inject relevant knowledge into LLM queries. The RAG process involves the following four key parts:

Data ingestion

Before an LLM can provide enriched responses, it needs access to structured and unstructured knowledge bases. ISVs can index diverse data sources using powerful indexing and search capabilities from Azure:

- **Azure AI Search**: Enables semantic search over large document repositories
- **Azure Cosmos DB**: Supports globally distributed document retrieval
- **Azure Database for PostgreSQL**: Provides structured query access to relational data
- **Azure SQL Database**: Serves as a secure relational store for business-critical applications

These services collectively allow ISVs to integrate internal documentation, customer-specific data, regulatory guidelines, or proprietary knowledge into AI-driven workflows.

User query processing

When a user submits a query, the system interprets intent and retrieves the most relevant content from the indexed knowledge base. Semantic search and embedding-based retrieval techniques in Azure ensure that the retrieved content is not only keyword-matched but also contextually aligned with the user's request using the following:

- **Vector embeddings**: Queries and documents are converted into numerical representations, allowing the system to retrieve results based on conceptual similarity rather than just text matching.
- **Hybrid search**: Combines traditional keyword-based searches with semantic similarity.
- **Metadata filtering**: Ensures that responses adhere to security policies, access controls, and user-specific permissions.

Context injection

Once the most relevant passages are retrieved, they are appended to the user's original query before being sent to the LLM for processing. This expands the model's context window, allowing it to generate responses that are grounded in real-time knowledge, personalized and domain aware.

Response generation

With the additional context injected, the LLM processes the query and generates a well-informed, contextually accurate response. This approach reduces AI hallucinations by anchoring responses to externally retrieved data.

Key benefits of RAG for ISVs

By implementing RAG-powered AI solutions, ISVs can significantly lower their costs by eliminating the need for fine-tuning and by enabling ISVs to use lighter-weight, general-purpose LLMs. ISVs can also deliver more accurate, reliable solutions less prone to hallucinations. This is especially important for regulated industries like healthcare, finance, and legal. Furthermore, you don't have to upload domain-specific knowledge into the LLM itself; instead, you keep that data in an outside, secure cloud data store. Lastly, since RAG is so simple and quick to implement, ISVs can achieve a faster time to market.

For ISVs looking to implement RAG in their applications can utilize the robust AI ecosystem offered by Azure by following these steps:

1. **Choose an LLM and retrieval strategy**:
    - Use **Azure OpenAI Service** for GPT-4, or **Azure AI Model Catalog** for other models.
    - Store and retrieve knowledge using **Azure AI Search**, **Cosmos DB**, or **SQL Database**.

2. **Implement query processing and context injection**:
    - Utilize vector search capabilities in **Azure AI Search** for semantic retrieval.
    - Dynamically inject retrieved knowledge into user queries before sending them to the LLM.
3. **Deploy and scale securely**:
    - Optimize performance with **Azure Kubernetes Service** (**AKS**) for scalable inference.
    - Ensure compliance with the Azure built-in security and identify management (e.g., Azure AD, Private Link).

By following these steps, ISVs can create LLM-based applications grounded in internal data for customers interested in chat-with-your-own data use cases.

Azure Machine Learning and MLOps

Azure Machine Learning is a workspace primarily focused on supporting custom development and maintenance of machine learning models throughout their whole life cycle. To understand the full capabilities of Azure Machine Learning, we first need to introduce the concept of MLOps.

MLOps is a core function of machine learning engineering that focuses on streamlining the process of taking machine learning models to production and then maintaining and monitoring them. MLOps is often the result of collaboration between data scientists and machine learning engineers.-

In other words, MLOps is an engineering discipline that aims to unify machine learning system development and machine learning system deployment, in order to standardize and streamline the continuous delivery of high-performing models in production.

As shown in the following figure, MLOps can be divided into the following steps:

1. Training the machine learning model.
2. Packaging and containerizing the machine learning model.
3. Validating the machine learning model.
4. Deploying the machine learning model.
5. Monitoring the machine learning model.
6. Retraining the machine learning model.

This eventually forms a feedback loop, also shown in the following figure.

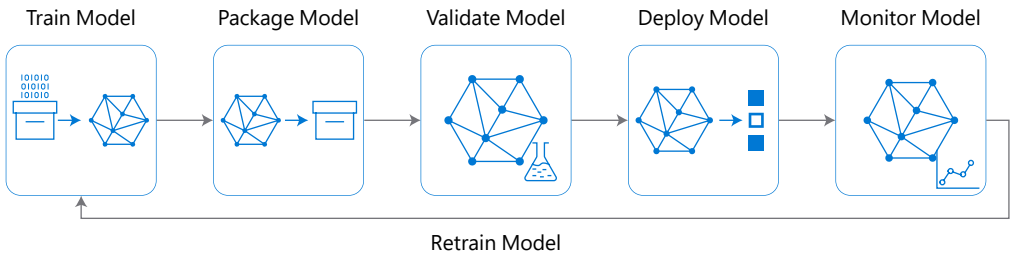

Figure 1.3: A visual overview of the MLOps flow and model feedback loop

Let's cover the steps in detail now:

1. **Training the machine learning model**: In this overview, training the model includes all initial development by data scientists, including exploratory data analysis, data pre-processing, and model prototyping (training and hyperparameter tuning).

   The Azure Machine Learning workspace provides many options to perform these tasks. As well as code-first options such as notebooks and scripts, automated machine learning (no-code) and designer (low-code/drag-and-drop) are performant alternatives aimed at citizen developers.

   Automated machine learning accepts a dataset and trains a variety of models on it. For citizen data scientists, this is a great way to easily acquire a trained model. However, professional data science teams can also use this, just in a different context. When starting a new project and dataset, a data scientist can run an automated machine learning job within minutes and get a good idea of which type of models perform significantly better on the dataset. This gives a strong indication of what algorithms can be further explored during custom development.

   For large jobs, such as model (re)training or batch inference, we can use the machine learning pipelines provided by Azure Machine Learning. Machine learning pipelines form an essential component in AI development. They conveniently orchestrate entire end-to-end machine learning workflows, from data pre-processing to model training, validation, and deployment. These pipelines are often used when the code remains relatively unchanged over time, typically after the experimentation phase.

2. **Packaging the machine learning model**: This refers to packaging the machine learning model within a container. This ensures that it will run the same way on any machine regardless of its environment. By packaging the files that run the model along with the Python packages and other dependencies required to run the model, you can rerun the code anywhere you'd like.

3. **Validating the machine learning model**: Although model validation is listed as a separate step in the MLOps process, in reality it is often performed during or immediately after training. For deep learning models, this happens during the training process; in between every pass throughout the entire training dataset (also called **epochs**), to find the ideal moment to stop training. Without going too deep into the mathematics of machine learning, model performance can deteriorate when training for too long, often referred to as overfitting, or when training is too short, known as underfitting.

4. **Deploying the machine learning model**: From this step onwards, the machine learning engineer comes into play. While the data scientist has a good understanding of mathematics and statistics, the machine learning engineer usually brings DevOps and containerization (Docker, Kubernetes, etc.) knowledge to the table.

   When deploying a model and making it available through an API, we need three things:

   - A compute target
   - An environment
   - An entry script

   For compute targets, when deploying into production, we often make use of an **Azure Kubernetes Service** (**AKS**) cluster. With the newer version of the Azure Machine Learning SDK (V2), it is possible to have managed endpoints called **Azure Machine Learning Managed Endpoints** to outsource some of the cluster management. Another option is deploying a single container using **Azure Container Instances** (**ACI**), although this is usually used for prototyping, as it does not provide a scalable solution. Models can also be deployed on IoT Edge devices or on-premises hardware (using Azure Arc, a service to simplify hybrid cloud architectures), as shown in the following figure.

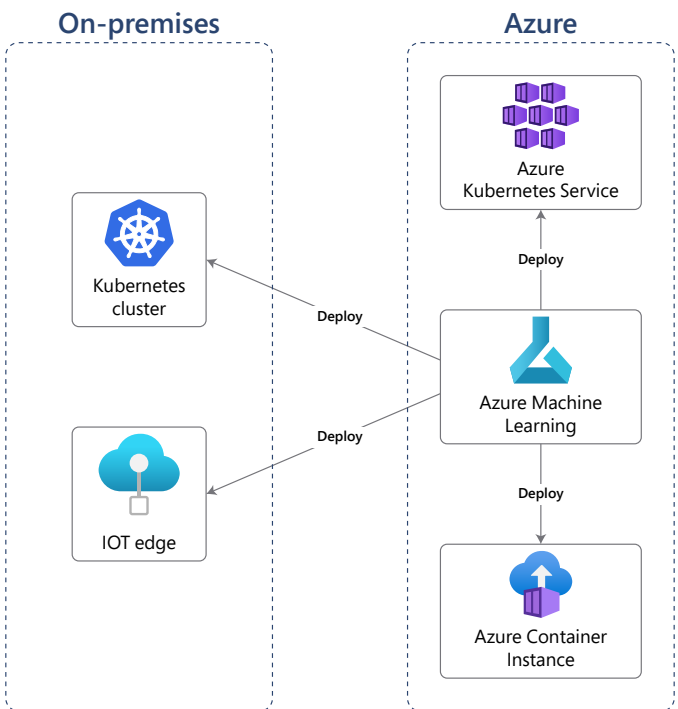

Figure 1.4: Common deployment options for trained machine learning models in Azure Machine Learning

Next, an environment needs to be provided to the deployment container, holding all the necessary dependencies for the model. For this, we can either upload our own custom environment to Azure Machine Learning or use one of the curated (pre-built) environments available in the workspace.

The final required component is the entry script. This is a brief script that fetches the model from the model registry in Azure Machine Learning, loads it into memory, transforms the incoming data from `POST` requests in the right format for the model (note that this is model-specific, so most entry scripts are unique), makes a prediction, and returns it. The following code snippet is an example of a minimalistic entry script:

```
import os
import pickle
import json
import numpy as np

# This function runs when the container starts
def init():
    global model
    # Get the path in the model registry
    model_path = os.path.join(os.getenv('AZUREML_MODEL_DIR'),
'model.pkl')
    # Load model in memory
    model = pickle.load(open(model_path, 'rb'))

# This function runs for every request made to the model
endpoint
def run(data):
    try:
        # Unpack the incoming data
        body = json.loads(data)
        sample = body['data']

        # Reshape the data
        sample = np.array(sample).reshape(1, -1)

        # Predict and return the prediction
        pred = model.predict(sample)
        return str(pred)

    except Exception as e:
        return str(e)
```

5. **Monitoring the machine learning model**: Good monitoring is the result of collaboration between the machine learning engineer and the data scientist. The machine learning engineer mainly monitors the health of the deployment (less so when using the managed online endpoints). For this, the engineer can make use of Azure Monitor.

   The data scientist will monitor the performance of the model as time goes on. For this, scientists can use built-in data drift monitoring in the workspace. When monitoring data drift, two datasets are defined in the workspace, the baseline and target dataset. A dataset can be dynamic (for example, the last two weeks of data) or static (all data from a certain month).

   The baseline dataset is a static dataset referring to the training data for the model currently in production. The target dataset is usually a dynamic dataset referring to the most recent weeks or months of data. Checks are continuously performed between the baseline and the target dataset to check whether they remain statistically similar.

   Once the **statistical difference** (also called **drift magnitude**) exceeds a threshold, this means our model may have become outdated. The new, incoming data has significantly changed, and our model might require retraining. Before we move on to model retraining, we have to perform one more check – the health of the data sources. For instance, if a sensor breaks and faulty data is ingested, this will also be seen as data drift. Once we can rule out that this happened, we can start the retraining process.

6. **Retraining the machine learning model**: By retraining the model, we restart the continuous MLOps loop of training, validating, deploying, and monitoring. For retraining, we use deployed machine learning pipelines. Machine learning pipelines can be deployed when the code does not change anymore, which is the case for retraining and batch inference. The pipeline can still be dynamic, using pipeline parameters. These pipelines can be triggered using an API call. Data drift monitoring can kick off a retraining pipeline, but the best practice is to have a human actor in between to check the case of faults at the data source.

The efficiency of data drift monitoring and retraining pipelines is heavily dependent on the work of the data architect and data engineer. A well-structured data lake and well-architected ETL pipelines go a long way and greatly improve the quality of MLOps processes.

Here are some best ML Ops practices:

- **Use branching strategies**: Data scientists work in topic branches off the master branch to maintain a structured workflow.
- **Automate CI pipelines**: Code commits to the repository trigger a **continuous integration** (**CI**) pipeline for automated processing.
- **Provision infrastructure as code**: The first pipeline run provisions essential resources such as ML workspaces, compute targets, and data stores.
- **Ensure code and data quality**: Every new code commit runs unit tests and data quality checks before proceeding.

- **Train the model**: Execute training scripts and algorithms, producing a model file stored in the run history.
- **Evaluate model performance**: Compare the new model against the production model; proceed only if the new model performs better.
- **Register the best model**: Version control the best-performing model by registering it in the Azure ML Model registry.
- **Package and validate the model**: Use the Azure ML CLI to package and validate the model before deployment.
- **Deploy with Azure ML and DevOps**: Registered models can be deployed using Azure ML and Azure DevOps for streamlined integration.
- **Define a release pipeline**: Coordinate deployments through Azure Pipelines, integrating artifacts from Azure ML, Azure Repos, and GitHub.
- **Utilize gated releases**: Deploy to a staging/QA environment first, requiring manual approval before final deployment.
- **Deploy to production**: Once approved, deploy the model scoring service to AKS and conduct final validation.

By following these best practices, ISVs can become guides for their customers in establishing sound data science practices that align well with other engineering teams.

AI architectures on Azure

To understand the addition of AI components in a larger solution, we will take a look at some common architectures. First, let's examine a single-tenant RAG solution in Azure. Then, let's look at Azure OpenAI chat basic architecture.

Single-tenant RAG solution

This is one of the most sought-after solution architectures. A company has data and wants users to be able to easily find the data and receive answers to commonly asked questions. Data could be company policy, historical financial performance, sales data, HR policies, company history, the address book, legal filings, pending patents, etc. It could be nearly anything. Within this architecture, presented in *Figure 1.5*, all of the company's data is in Azure Blob Storage and indexed with Azure AI Search.

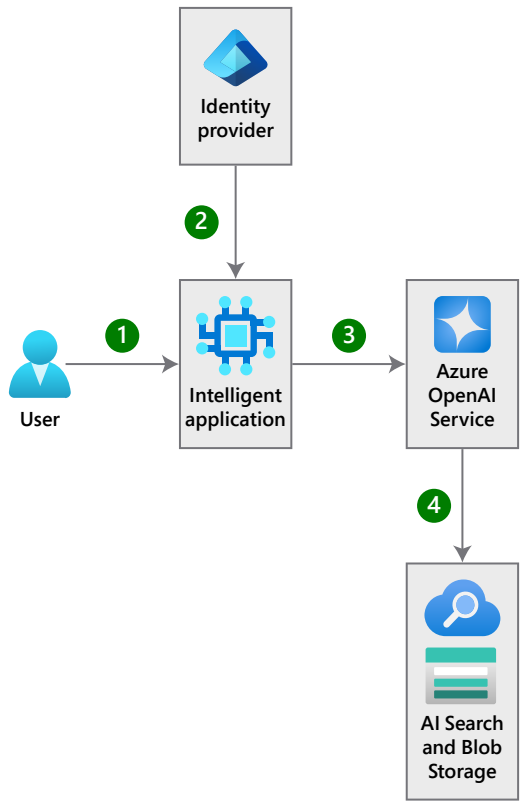

Figure 1.5: Single-tenant RAG solution architecture

On top of that data sits Azure OpenAI Service, which uses one of its LLMs, probably GPT-4o, to query the data using natural language. The ISV-built intelligent application gives users a graphical web interface which includes the ability to chat with your own data.

Then, the user issues a request through the intelligent web application. An identity provider authenticates the requestor. Once the user is authenticated, the application calls the Azure OpenAI-hosted LLM to perform the user's query. It then fetches grounding data within the indexed data and uses that data as a part of the context in forming the LLM's response. Then, the results are returned to the user via the intelligent application.

Multi-tenant versions of this architecture are also available via the Azure architecture center at https://learn.microsoft.com/azure/architecture/ai-ml/guide/secure-multitenant-rag.

Azure OpenAI chat basic architecture

Another common scenario is private GPT. Companies would love their employees to use an AI chatbot, but don't want them disseminating carefully guarded information to the open web. To accomplish this, ISVs can build their own chatbots within Azure as shown in *Figure 1.6*.

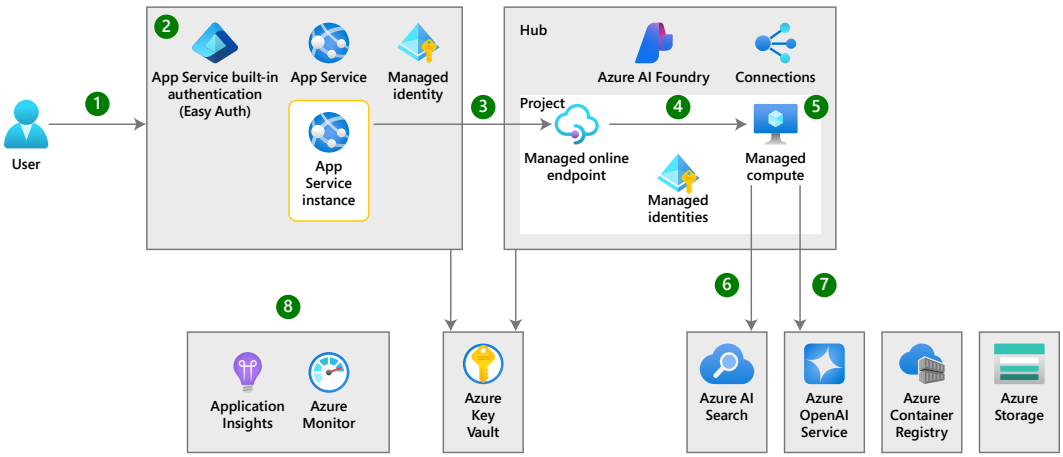

Figure 1.6: Azure OpenAI chat basic architecture

Within this architecture, there's a client application hosted on App service that presents a chat UI to the end user. The user sends an HTTPS request to the default domain on azurewebsites.net. That default domain directs traffic to the App Service's built-in public IP. The TLS connection is established between the client and App Service, with Azure managing the certificate. Easy Auth ensures authentication via Microsoft Entra ID.

When a user sends a message, the chat UI interacts with APIs, and the API connects to an Azure Machine Learning managed online endpoint, which routes the request to managed compute where the orchestration logic runs. This logic extracts the user's query and retrieves relevant grounding data from Azure AI Search. The enriched prompt is then sent to Azure OpenAI for processing.

Application Insights logs the original request, API calls, and interactions with the managed online endpoint, storing them in the same Azure Monitor Logs workspace as Azure OpenAI telemetry. This setup ensures end-to-end observability and monitoring across the system.

Summary

This chapter focused on data science and AI on Azure. We started by outlining the different roles involved in a data science team, including the responsibilities of data architects, engineers, scientists, and machine learning engineers, and how the collaboration between these roles is key to building successful AI solutions.

We then focused on the role of the data architect when designing an AI solution, outlining the questions they should ask themselves for a well-architected design.

Next, we delved into the various AI services offered by Azure, including Azure AI services, Azure Machine Learning, Azure AI Model Catalog, and Azure OpenAI Service. For Azure AI services, we saw speech, vision, and language models.

We then introduced the concept of MLOps, along with mapping the different steps in the MLOps process to components with Azure Machine Learning, the workspace used for custom AI development.

Finally, we introduced the reference architectures for a single-tenant RAG solution in Azure and Azure OpenAI chat.

In the next chapter, we will explore how to enforce data governance and compliance, essential components for scalable and highly performant data platforms.

2

Enterprise-Level Data Governance and Compliance

As organizations continue to generate and consume vast amounts of data, it has become increasingly important to manage this data in a strategic and structured manner. This is where data governance comes in, providing a framework for managing data as a valuable organizational asset. With data governance practices in place, we can ensure that an organization's data is accurate, secure, and accessible, while also complying with relevant regional or industry-specific regulations. Without proper data governance practices, we run the risk of struggling with disparate data sources and inaccurate data and having difficulties in ensuring data privacy and security. As such, data governance and compliance have become essential components of any organization's data strategy.

In this chapter, you will learn about the importance of data governance, the roles involved in its implementation, and tools such as Microsoft Purview that can be used to facilitate data governance. Additionally, we will explore two assessment frameworks and two data governance models to help you establish and apply effective data governance practices in your organization.

By the end of this chapter, you will have a solid understanding of the concepts of data governance and compliance, the required profiles, Microsoft Purview as a governance service, and several proven frameworks to assist in the adoption of data governance in an organization.

We will be covering the following topics in the chapter:

- The importance of data governance and compliance
- Governing data with Microsoft Purview
- Applying enterprise-level data governance

The importance of data governance and compliance

Businesses are becoming more and more data driven as time goes on. They have good reasons to do so, as revenue, time-to-market, profits, and customer satisfaction increase greatly for data-driven businesses. Given this trend, the need for data governance is higher than ever.

The goals of data governance can be summarized as follows:

- Managing an ever-growing data landscape
- Overcoming data silos
- Increasing data agility
- Complying with data regulations

More data-driven organizations mean the rapid growth of the global data volume, but also a larger data landscape per organization. The latter will cause serious issues in the long run when managed incorrectly. With a strong data governance strategy alongside the right tools and services, a data landscape will remain clear and structured at scale. The data landscape tends to grow exponentially with the business. When the business grows, new subsidiaries or acquisitions bring new data, but existing departments also start to use more **software as a service** (**SaaS**) applications (such as software for planning, HR management, decision-making, and so on) to develop their own applications, or generate more data in another way.

Next, overcoming silos is nothing new in the data world. Data silos could refer to databases of different departments that are not integrated with each other. On a larger scale, subsidiaries or acquired organizations also tend to be seen as silos because their data often resides in different data centers, cloud tenants, or even cloud providers. Organizations have long been working on centralizing data in a (cloud-based) data lake and/or data warehouse. Data governance solutions such as Microsoft Purview provide insights into every silo without the immediate need for data migration. In mature scenarios with a central data warehouse, Microsoft Purview grants insights into all the surrounding data transformation processes, source databases, and downstream data usage – for instance, in dashboards or reports. An example is shown in the following figure:

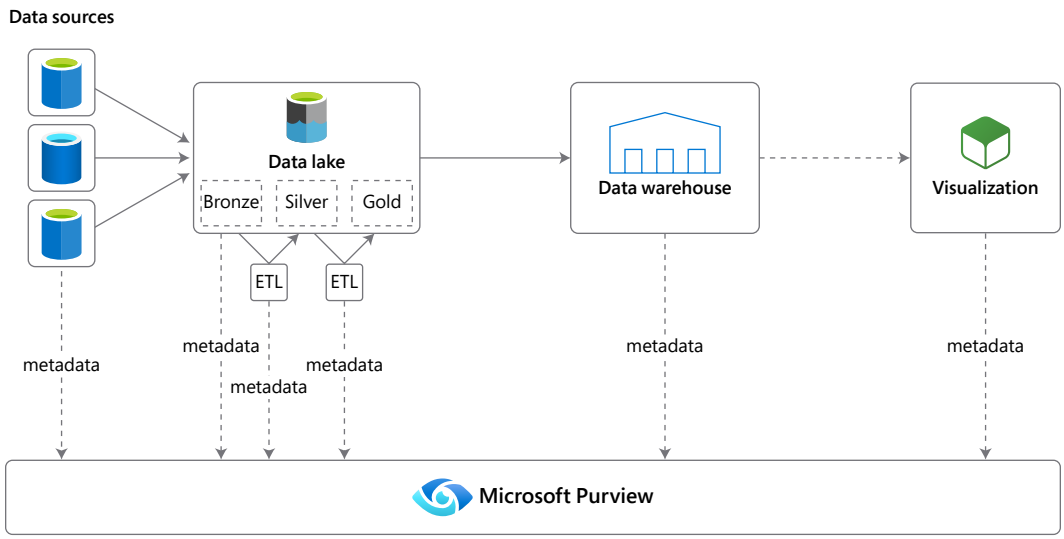

Figure 2.1: Architecture representing Microsoft Purview ingesting metadata from the entire data platform

Effective data governance should also result in a vast increase in data agility. This comes as a consequence of having fast organization-wide insights into all data. Data strategists such as the **chief data officer** (**CDO**) can adapt rapidly to changes in the data. These changes can reflect new trends or external influences or quickly provide results from altered internal business processes.

Good data governance also forms the basis for data compliance. Without governance, it is not possible to efficiently verify whether data fully complies with various regulations, be they regional or industry specific. With hefty fines for violations, this is something organizations should avoid at all costs. An example of regional regulation is the **General Data Protection Regulation** (**GDPR**) in the **European Union** (**EU**), focusing on data protection and privacy. Simple examples of industry-specific regulations include the handling and storage of sensitive data regarding client-attorney privilege in the legal sector or patient data in the medical sector.

Now that we have seen the results of strong data governance, let's dive deeper into the roles that are essential for designing, implementing, and enforcing such a strategy.

The roles in data governance and compliance

The field of data governance and compliance is becoming increasingly interdisciplinary. Many different roles collaborate to design and execute the strategy at the enterprise level. In this domain, we introduce a few new roles:

- C-level executives:
  - **Chief data officer** (**CDO**)
  - **Chief information officer** (**CIO**)
  - **Chief technology officer** (**CTO**)
  - **Chief information security officer** (**CISO**)
- Data administrator
- Data steward
- Data owner
- **Subject-matter expert** (**SME**)

Executives, data administrators (in a data governance board), and stewards are typically situated in a hierarchical structure such as the following:

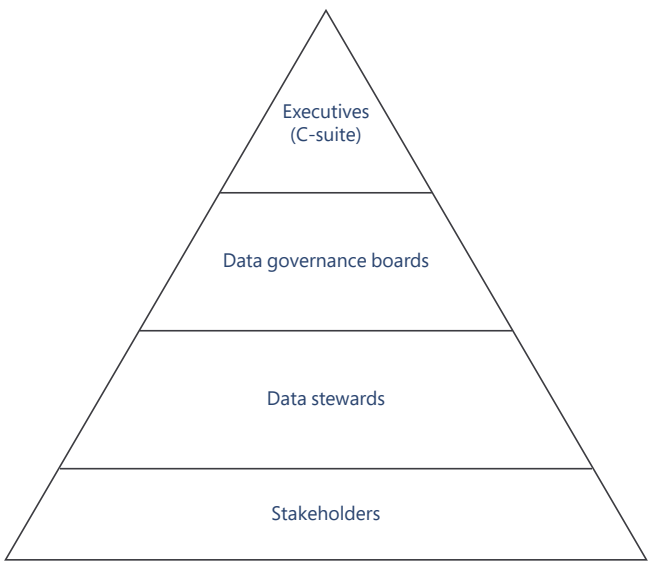

Figure 2.2: Data governance organizational model

We will look at all data governance roles in more detail, starting with the C-suite executives.

Executive roles in data governance

The roles of CDO and CISO are relatively new but are becoming increasingly important as companies continue to rely more heavily on data and technology.

A CDO is responsible for managing an organization's data assets and ensuring that data is properly stored, managed, and used to achieve the company's objectives. This includes developing data governance policies and procedures, managing data quality and integrity, and overseeing data analytics and reporting. The CDO works closely with other senior leaders to ensure that data is effectively used to support business decisions and drive innovation. More specifically, in data governance, the CDO will (indirectly) oversee the core governance and compliance team, consisting of data officers with elevated permissions to enforce governance.

A CISO, on the other hand, is responsible for managing an organization's information security program. This includes developing and implementing security policies and procedures, managing security risk assessments, and overseeing security monitoring and incident response. The CISO ensures that the organization's information is protected from cyber threats and designs strategies for responding to security incidents.

The CTO and CIO typically cover the entire technical landscape of the organization: the CTO might focus more on the technological aspects of the company's products and services while the CIO focuses on the management and security of the company's IT systems. Although the scope of the CTO and CIO is broader than the data domain, in less mature or more traditional organizations without a CDO, they may assume CDO responsibilities in data governance and compliance.

Data governance boards

Next is the data governance board. It is a governing body that strategizes data governance programs in collaboration with the CDO, raises awareness of its importance, approves enterprise data policies and standards, prioritizes related projects, and enables ongoing support. The board typically includes delegates from the core function teams of marketing, sales, finance, and HR – critical teams that depend on accurate data to perform their jobs effectively.

In its simplest form, a robust data governance team consists of data administrators and data stewards. Data administrators oversee the implementation of the entire data governance program.

Data stewards

A data steward is a person who is responsible for managing the data assets at a departmental or business unit level. They ensure that data is properly managed, maintained, and used to meet the needs of their specific area of the organization. Although the role of a data steward may vary between organizations, in mature environments, it comes down to the following areas:

- **Data quality**: Ensuring that the data within their department is accurate, complete, and consistent
- **Data access**: Managing who has access to the data within their department and ensuring that access is appropriate and secure

- **Data security**: Ensuring that the data within their department is protected from unauthorized access or breach
- **Data governance**: Helping to develop and enforce data governance policies and procedures within their department or business unit
- **Data usage**: Ensuring that the data within their department is being used appropriately to meet the needs of the organization
- **Data training**: Providing guidance and training to their colleagues on the proper use of data within their department

Data stewards work closely with other stakeholders, including the governance board and end users, to ensure that the data at the operational level is properly integrated, maintained, and aligned with the overall data strategy of the organization.

Data owner and SME

Contrary to the data steward, the data owner and SME are not full-time roles. These are dedicated individuals for a data asset, whose contact information will be visible to those who seek more information about the asset.

The data owner carries the final responsibility for the data asset. The data owner overlooks the data governance, data quality, security, and compliance for its data asset. To accomplish this, the data owner can utilize data stewards. By appointing a data owner, thereby having someone responsible for every single data asset in the organization, the risk of an ever-growing unmanageable spaghetti of data assets is mitigated.

The SME is, just like the data owner, an individual appointed to give insights into the data. The SME is typically someone with hands-on experience with the data and good theoretical knowledge of the field and use cases. They can provide clarity on data columns, values, aggregations, derivations, and so on.

Now that we have covered the importance of data governance and the roles that are applicable to this domain, let's move on to Microsoft Purview. This tool will allow us to manage organization-wide data assets conveniently.

Governing data with Microsoft Purview

Microsoft Purview is a unified data governance service that allows organizations to manage and govern data that resides anywhere, from on-premises data stores to multi-cloud and even SaaS data.

Microsoft Purview enables data discovery by providing data scanning and classification for assets across the entire data landscape. Metadata and descriptions of discovered data assets are integrated into a holistic map of the data estate, as shown in the following figure:

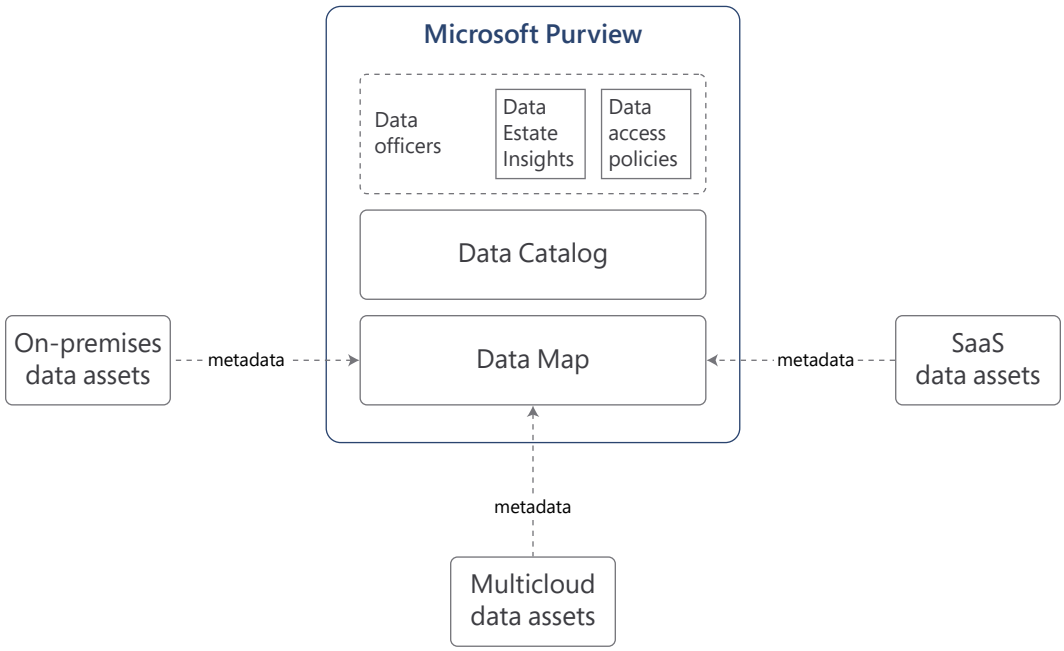

Figure 2.3: Microsoft Purview components

Microsoft Purview is built upon core components: Data Insights, Data Catalog, and Data Map. The Microsoft Purview Data Map holds important metadata from multi-cloud, SaaS, and on- premises sources. Data officers can also set data access policies, which are straightforward.

The service consists of three core components:

- Microsoft Purview Data Map
- Microsoft Purview Data Catalog
- Microsoft Purview Data Estate Insights

Let's take a look at each of them.

Data Map

Data Map provides the foundation for discovery and governance. It is a unified map of all data assets and their relationships and enables more effective governance for the data estate. It functions as a knowledge graph that forms the basis for the Microsoft Purview Data Catalog. It is crucial to get the data map right, as it significantly impacts on the performance of all other governance and discovery efforts.

We will now cover the different concepts within the data map, starting with the connection to various data sources.

Sources

The first step of building a data map is connecting to all data sources. Most of these, including third-party software, can make use of built-in connectors to rapidly create a link between the data source and Microsoft Purview. By registering an external data store as a source, the connection information for that data source is securely kept within Microsoft Purview, allowing for easy manageability.

Using this method, Microsoft Purview connects to an organization's entire data landscape, going from Azure services to on-premises data sources, Power BI, Amazon Web Services, Google Cloud Platform, SAP, Salesforce, Snowflake, and many more. This means that in a typical data platform—sources, data lake, data warehouse, and reporting—Microsoft Purview has information on the whole data journey from start to finish.

Scans

Microsoft Purview does not copy the data from the source to its service. This would incur a significant amount of unnecessary costs. Instead, it periodically scans the registered data sources and stores metadata of all data assets. For example, Microsoft Purview will ingest metadata such as the following:

- Data asset name
- File size
- Schema (columns, data types, and so on)

On top, Microsoft Purview will apply classifications and labels during the scanning process, which will be further discussed later in this chapter.

The use of scans is the main cost driver for Microsoft Purview. Therefore, it is possible to apply scan rule sets to perform periodic scanning in the most cost-efficient way possible.

Scans can be scoped and individually scheduled, on different levels of granularity. Organizations can decide what sources to scan, what file types, and which classifications should be applied. For example, all structured data in a data lake can be scanned daily, while unstructured file types are scanned twice per week.

Optimizing the scan interval according to the importance of the data and the speed at which it changes is essential, as this greatly affects the resulting costs incurred from the service.

Data asset

A data asset, in the context of Microsoft Purview, is an abstract concept and can point toward different layers in the data hierarchy. Data assets include data storage services, workspaces, tables, files, dashboards, and reports. For example, an SQL table is considered a data asset, as well as the schema, database, and SQL server it resides in. Data assets can be grouped logically using hierarchical collections.

Lastly, the scans that are performed by Microsoft Purview on the data sources utilize Azure Data Factory pipelines under the hood. This means that in terms of compute or integration runtimes, the approach is very similar to that of Azure Data Factory.

Classifications

As mentioned before, Microsoft Purview applies classifications while scanning the data sources. Classifications categorize columns according to a specific pattern and are useful to automatically detect certain types of data. More specifically, **personally identifiable information** (**PII**) can be distinguished upon scanning, so that sensitivity labels can be applied where necessary. The organization can then use these labels to enforce policy compliance.

Classification rules are set up using regular expressions in combination with a threshold of match percentage. These regular expressions can be applied to column names or data values. For instance, we could apply a rule set looking for the words phone or mobile, while also matching the cell values to a specific phone number pattern. Microsoft Purview has a vast range of classifications readily available, such as passport numbers (for different countries), credit card numbers, and email addresses. The next pillar of Microsoft Purview we will cover is the Data Catalog, which builds upon the metadata in the Data Map.

Data Catalog

The Data Catalog provides a complete overview of the data landscape with automated data discovery, data lineage, and sensitive data classification, along with some other features. The data catalog is to be used by business end users to find any data asset (given that permissions allow it) and receive key information on it.

Within the data catalog, we will zoom in on the semantic search, data asset overview, and business glossary.

Semantic search

Semantic search in Microsoft Purview enables users to easily locate data assets across the organization's entire data landscape. This makes a task that used to be extremely cumbersome, awfully simple. End users can simply browse the data catalog to retrieve any data asset within their scope of permissions.

Data assets

When a data asset is retrieved, users are able to access various metadata, such as the following:

- Schema
- Data lineage
- Contacts
- Related data assets

We are able to view the schema (if applicable) of the data asset – the column names, column descriptions, data types, and the classifications and sensitivity labels applied by Microsoft Purview. These columns can also be linked to certain terms in the business glossary to create more clarity on otherwise ambiguous column names.

Next to the schema, we get access to the data lineage, one of the most powerful and sought-after features within Microsoft Purview. It provides a comprehensive, visual overview of the journey of data from its origin to its destination(s). End users can easily trace the flow of data through ETL pipelines, understand transformations happening along the way, and gain insights into how data is used within the organization. Furthermore, Microsoft Purview allows us to drill down on specific columns in aggregated datasets to see where the data originated and in which dashboards and reports it is being used. The lineage is also useful for root cause analysis and data quality analysis. The following figure shows an example of data lineage in Microsoft Purview from start to finish:

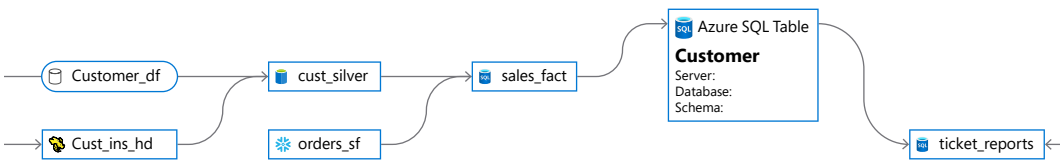

Figure 2.4: An example of a data lineage in Microsoft Purview

We get a view of the upstream sources and downstream usage of data in the chosen data asset (in this case, a SQL table).

When using Azure-native data pipelines, such as in Azure Data Factory or Synapse Analytics, this data lineage is automatically generated. For this reason, it is recommended to use Azure-native services for ETL pipelines, as this greatly increases the convenience of building up data lineages. However, it remains possible to add custom data lineages, yet this requires some custom coding and making use of the underlying Apache Atlas API.

Every data asset has a list of relevant contacts. These contacts are divided by role. Using this list, we can easily discover who is the designated data owner, the stewards governing the data asset, and the SMEs, along with their contact details. Additionally, it is possible to define custom roles and add them to the list.

Microsoft Purview also provides the option to retrieve relevant data assets, such as parent and child assets. For instance, when looking up a SQL table, Microsoft Purview also returns links to the schema, database, and server asset.

Lastly, access can be requested instantly from the data asset page and a `.pbix` file can be downloaded to quickly open up the data in a dashboard or report for analysis.

Business glossary

The business glossary is the final component of the data catalog. It serves as a centralized tool that allows the business to define and manage business terms, concepts, and relationships. It helps organizations to establish a common understanding of the meaning and context of the data they use across various teams, systems, and processes. For larger corporations, a business glossary becomes absolutely essential, as the smallest ambiguity in term definitions can have a major impact on larger workflows.

An organization can benefit in many ways from leveraging a business glossary such as the one in Microsoft Purview. The business glossary provides a single location to manage all business terms and definitions, centralizing the overhead management. Furthermore, it establishes a standardized naming convention in a clear manner. This helps to avoid confusion and ensure everyone is using the same terminology.

Next, the business glossary provides a search and discovery feature that allows users to quickly find the relevant business terms and definitions they need. Lastly, the glossary is integrated with other components of Microsoft Purview, such as data assets. As mentioned before, we can link certain data columns to business terms in the glossary to provide more clarity.

In the end, the data catalog can be visualized as follows:

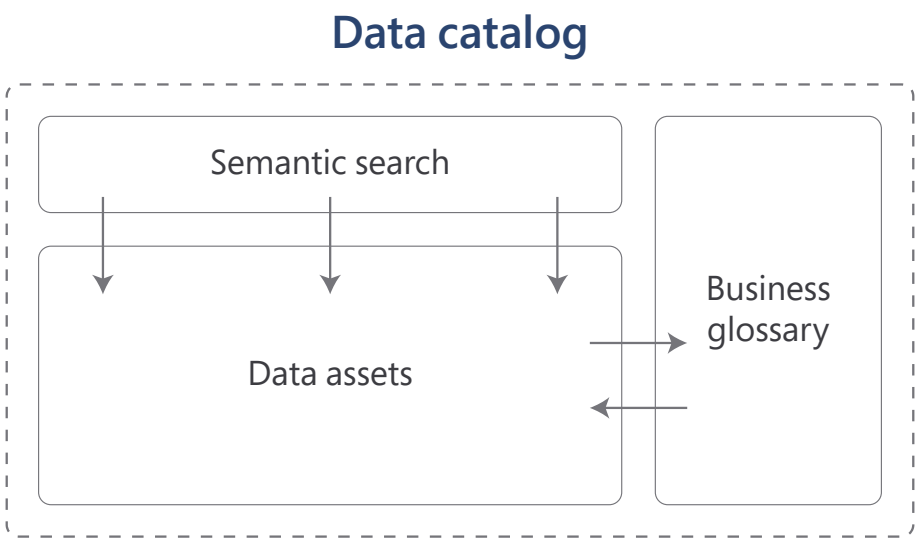

Figure 2.5: The data catalog provides a powerful semantic search on top of the data assets, accompanied by a business glossary

After exploring the Data Map and Data Catalog, there is one pillar left to cover: Data Estate Insights.

Data Estate Insights

Data Estate Insights provides a high-level overview of the assets that have been scanned into the Data Map and views key gaps that can be closed by governance stakeholders for better governance of the data estate. Access to Data Estate Insights should be more restricted than the data catalog. It is mainly to be used by executives and the governance boards.

Data Estate Insights comprises a collection of pre-built dashboards, showing aggregated information about the data landscape. More specifically, an executive or a member of the governance board can utilize these dashboards to get insights on the following topics:

- **The state of data stewardship**: Here we can see the level of data curation across all data assets. A data asset is seen as fully curated when it has an assigned data owner, description, and classification.
- **Data catalog adoption**: This shows the usage of the data catalog's semantic search, which data assets were the most retrieved, which keywords were the most popular, and so on.
- **Insights on data assets**: Aggregated information is available here on data assets that can be drilled down in many ways, such as Microsoft Purview collection, source type, and file type.
- **Curation of the business glossary**: This shows both the curation of the business term in the glossary (does it have an owner, a definition, ...?) as well as their level of linkage with data assets.
- **Insights on classifications and sensitivity labels**: As the name suggests, this shows information on the usage and curation of both classifications and sensitivity labels.

In summary, Data Estate Insights is a restricted area within Microsoft Purview that provides leaders with a cross-tenant bird's-eye view of the current state of the organization's data landscape.

These were the three main pillars of Microsoft Purview: Data Map, Data Catalog, and Data Estate Insights. Understanding the tool is one thing, but implementing data governance requires transformational steps to be taken on the business level as well. Let's find out how we can do this.

Applying enterprise-level data governance

Not every cloud data architect will have to be as well-versed in the following part, as the implementation of data governance is something that happens organization-wide, rather than on the level of a single data solution. It would mainly involve a chief architect, CDO, or CIO.

Data governance is essential for any data estate at scale, yet we cannot implement data governance blindly. As we will see later, data governance can be implemented too late or too early.

Neglecting the need for governance over a longer period of time, until the data platform reaches full-scale maturity, will cause many issues in the long run. The longer governance, compliance, and data management are neglected, the more technical debt is built up. The data landscape will turn into an unmanageable swamp, making it hard or near impossible to clean up and organize. Therefore, the key takeaway here is to think about data governance from the start.

On the other hand, data governance can be implemented too fast. First, the organization needs a good idea of the designated roles, such as the data owners and data stewards. Second, the business has to decide on a solid data governance and management strategy. To do this, we can utilize a couple of frameworks. It is recommended to follow one of the proven assessment frameworks (to assess the organization's maturity) and a governance framework (for the actual governance model to be used). Jumping straight into implementing data governance while disregarding these crucial steps runs the risk of having an unscalable governance model, resulting in a chaotic governance structure further down the road.

Preparing the business for data governance and management

Before deciding on a governance framework, the main goal is to understand the organization's maturity level when it comes to data. This can be done in various ways, but two widely adopted assessment frameworks are as follows:

- **Data Management Capability Assessment Model** (**DCAM**) assessment
- **Data Management Association International** (**DAMA**) **Data Management Body of Knowledge** (**DMBOK**)

Let's take a closer look at these two approaches.

DCAM assessment

The first step that an organization can take to prepare for implementing data governance and management is to perform a DCAM assessment. The DCAM is a framework designed to assess an organization's data management capabilities across various domains. This model helps organizations evaluate their current state of data management and identify areas for improvement in their data management practices.

The DCAM assessment typically involves a comprehensive evaluation of an organization's data management capabilities, including data governance, data architecture, data integration, data quality, and data security. The assessment model is based on a maturity model approach, where organizations are evaluated against a set of criteria that reflect best practices for each domain. The criteria are organized into maturity levels, which indicate the level of maturity an organization has reached in each domain.

DCAM defines maturity levels based on the following six dimensions:

- **Governance**: The ability to manage data as a strategic asset across the enterprise
- **People**: The skills and competencies needed to manage data effectively
- **Process**: The methods and procedures used to manage data across its life cycle
- **Technology**: The tools and infrastructure used to manage data
- **Data**: The quality, consistency, and completeness of data across the enterprise
- **Business**: The alignment of data management with business objectives

The DCAM assessment typically follows a structured approach involving a series of steps that include data gathering, analysis, and reporting. The assessment may be conducted by internal or external auditors, and the findings are typically presented in a report that outlines the areas of strengths and weaknesses in the organization's data management capabilities. The report may also include recommendations for improvement and remediation.

DAMA DMBOK

DAMA DMBOK is a framework developed by DAMA that provides a comprehensive guide to data management practices and principles. It can be seen as a solid alternative to DCAM. Much like the DCAM framework, DMBOK is also used to evaluate the data maturity level of the organization.

The DMBOK framework consists of a set of guidelines, best practices, and standards that cover all aspects of data management, including data governance, data architecture, data quality, metadata management, data modeling, and many other topics. The framework is organized into 10 functional areas:

- Data governance
- Data architecture management
- Data development
- Database operations management
- Data security management
- Data integration management
- Document and content management
- Reference and master data management
- Data warehousing and business intelligence management
- Metadata management

Each functional area contains a set of activities, tasks, and deliverables that are essential for effective data management. The DAMA DMBOK framework is intended to be a flexible guide that organizations can adapt to their specific needs and requirements.

Comparing the DCAM and DAMA DMBOK frameworks

Now that we have discussed both frameworks, we can start comparing them. The following table shows the differences between DCAM and DMBOK, or more specifically, DMBOK2 versus DCAM 2.2.

Level	Level name		Level description	
	DAMA-DMBOK2	DCAM® 2.2	DAMA-DMBOK2	DCAM® 2.2
Level 0	No capability	Non initiated	No organized data management practices or formal enterprise processes for managing data	Ad-hoc data management (performed by heroes)
Level 1	Initial/Ad hoc	Conceptual	• Little or no governance • Limited tool set • Roles defined within silos • Controls applied inconsistently • Data quality issues not addresses	Initial planning activities (white board sessions)
Level 2	Repeatable	Developmental	• Emerging governance • Introduction of a consistent tool set • Some roles and processes defined • Growing awareness of impact of data quality issues	Engagement underway (stakeholders being recruited and initial discussions about roles, responsibilities, standards and processes)
Level 3	Defined	Defined	• Data viewed as an organizational enabler • Scalable processes and tools • Reduction in manual processes • Process outcomes are more predictable	Data management capabilities established and verified by stakeholders (roles and responsibilities structured, policy and standards implemented, glossaries and identifiers established, sustainable funding)
Level 4	Managed	Achieved	• Centralized planning and governance • Management of risks related to data • Data Management performance metrics • Measurable improvements in data quality	Data management capabilities adopted and compliance enforced (sanctioned by executive management, activity coordinated, adherence audited, strategic funding)
Level 5	Optimization	Enhanced	• Highly predictable processes • Reduced risk • Well understood metrics to manage data quality and process quality	Data management capabilities fully integrated into operations (continuous improvement)

Table 2.1: A comparison between DMBOK and DCAM maturity levels

While both frameworks categorize organizations into six levels of maturity, these levels cannot be directly mapped against each other. As just described, DCAM defines maturity levels based on 6 dimensions that reflect an organization's overall data governance and management capability, while DMBOK defines maturity levels based on 10 knowledge areas.

Data governance frameworks

Once an acceptable maturity level has been defined (around level 2 for DMBOK and level 2-3 for DCAM), the organization can think about actually implementing data governance. Much like the initial assessment, it would be strongly recommended to follow a proven framework. The difference now is that we are looking for a data governance model.

Two well-known data governance models are as follows:

- The centralized data governance model
- The federated data governance model

Each model has its own benefits and drawbacks. The optimal choice for the governance model depends on the size of the business, its structure, and management needs.

Centralized data governance is a model in which a single governing body or team within an organization is responsible for managing and maintaining all of the organization's data. This governing body creates and enforces policies and procedures for how data is collected, stored, and used across the organization. This model is typically more hierarchical and has a top-down approach to managing data. It is most convenient and ideal for smaller businesses. Large organizations can also incorporate the centralized model if their business units are well integrated and data management requirements are roughly similar across the organization.

On the other hand, the federated data governance model is a more decentralized approach to managing data. In this model, multiple governing bodies or teams within an organization are responsible for managing different sets of data. Each team creates and enforces policies and procedures for the data under their purview (pun intended). This model is typically more collaborative and allows for greater flexibility in how data is managed. However, without careful coordination between business units, it runs the risk of resulting in data governance silos – the exact thing we are trying to avoid.

The federated data governance model is often used in larger organizations that have multiple business units, where the data requirements and governance needs vary greatly between them. By decentralizing governance, each business unit can manage its data more efficiently and effectively, while still adhering to overall organizational standards.

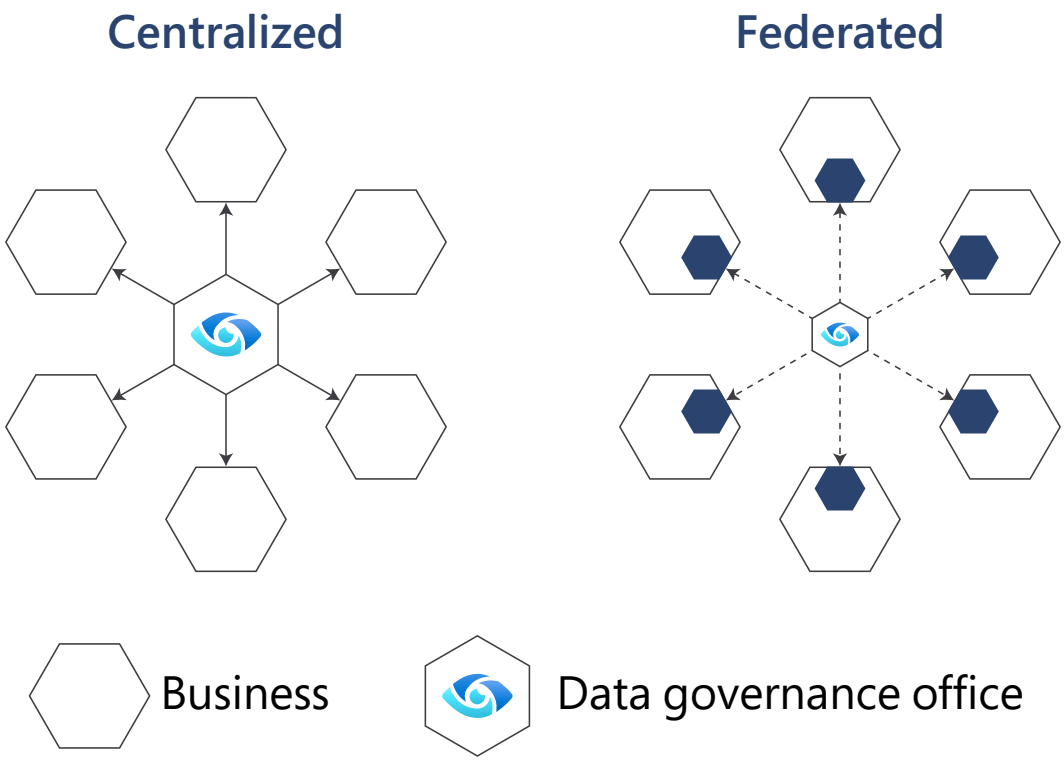

Figure 2.6: The centralized and federated data governance model

To conclude, the main difference between the two models is that centralized data governance is managed by a single governing body, whereas federated data governance is managed by multiple governing bodies or teams.

Summary

Data governance and compliance are critical components of any data strategy. In this chapter, we explored the importance of data governance, its goals, and the roles involved in its implementation, along with Microsoft Purview and data governance frameworks.

We highlighted four main goals of data governance, namely managing an ever-growing data landscape, overcoming data silos, increasing data agility, and complying with data regulations.

The chapter also discussed the different roles involved in data governance. Executives, governance boards, data stewards, data owners, and SMEs all have a part to play in ensuring that data is managed effectively and efficiently.

We explored Microsoft Purview as a core component for the implementation of data governance and compliance, which provides a Data Map, Data Catalog, and Data Estate Insights. This tool can help organizations understand their data landscape (even in its most decentralized form) and make better decisions about how to manage it.

Applying data governance in an organization requires a structured approach. We discussed two assessment frameworks, DCAM and DAMA DMBOK, which can help organizations evaluate their current level of data maturity and provide a clear path forward. We also explored two data governance models, centralized data governance and federated data governance, which organizations can use to determine the best approach for their needs.

Briefly, data governance and compliance are essential for organizations that want to manage their data effectively and comply with regulations. With the right tools, roles, and frameworks in place, organizations can achieve their data governance goals and realize the benefits of better data management.