INSPARI
a valantic company

Microsoft Fabric Festival
Dive into the Lakehouse

INSPARI
a valantic company

Today’s speakers

Thor Wetche

Data Engineer

thwe@inspari.dk

Mathias Ragn

Senior Data Engineer

mrag@inspari.dk

Agenda

1 Microsoft Fabric for Data Engineers

2 Demo Project, Architecture and Approach

3 Demo and Choice discussion

4 Wrap-up & Takeaways

Personal Data

Age:

41

Job Title:

Data Engineer

Company:

EnergyCorp

Data Engineer

Data Engineer

Taking data to the next level

Personal profile

Kayla is a Data Engineer at EnergyCorp

Tasks:

- Specialization in extracting, transforming and loading data
- Designing and building scalable data pipelines
- Exposing data models

Skills

ETL Processes

Python

Data modeling

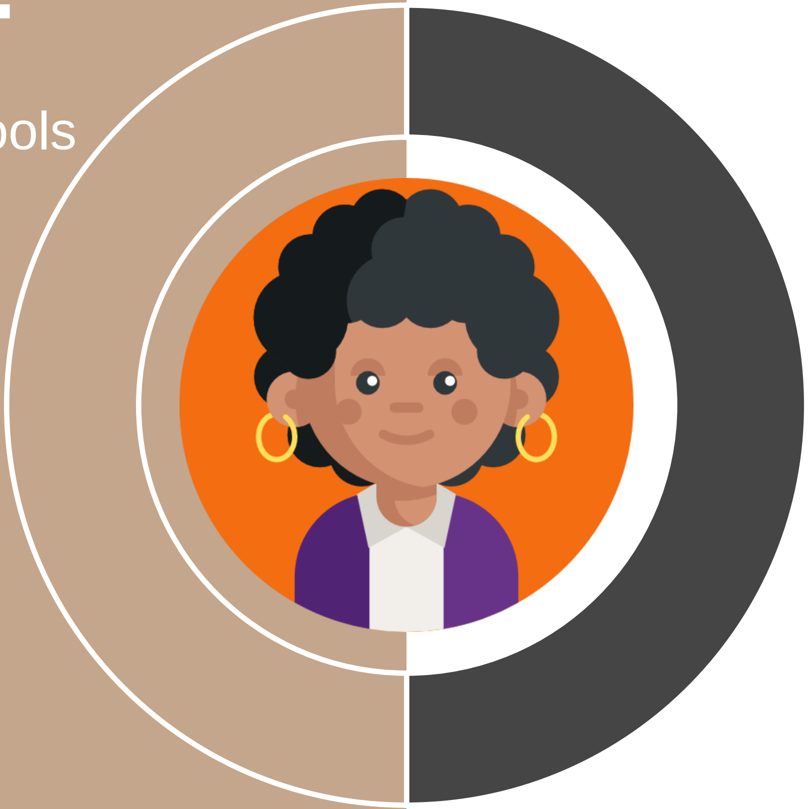
SQL

What are Kayla’s needs?

- Robust data processing tools
- Options for interacting with data sources
- Scalable infrastructure
- Tailor deliveries to users
- Futureproof platform

What are Kayla’s needs?

- Robust data processing tools
- Options for interacting with data sources
- Scalable infrastructure
- Tailor deliveries to users
- Futureproof platform



What are Kayla’s pains?

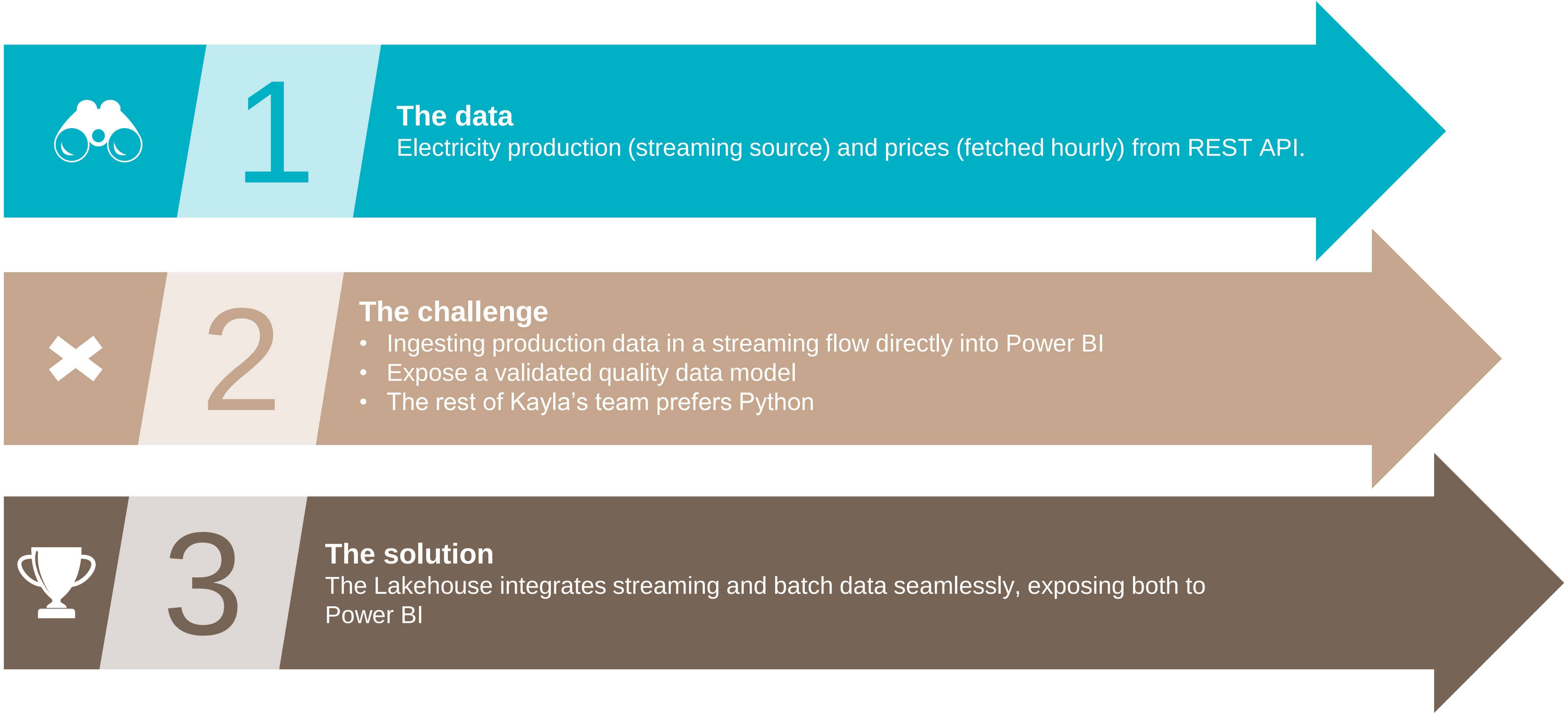
- Too many rigid tools
- Integration challenges
- Infrastructure cannot be scaled
- Outdated technology and lack of support

Fabric as an Engineering Enabler

- Many options for powerful data processing and integration
- Intuitive and familiar user experience
- Proof of concepts without managing infrastructure
- Fabric is continually updated and enhanced

A wild project appeared!

Analyzing Electricity Prices and Energy Production in Denmark



1

The data

Electricity production (streaming source) and prices (fetched hourly) from REST API.

2

The challenge

- Ingesting production data in a streaming flow directly into Power BI
- Expose a validated quality data model
- The rest of Kayla’s team prefers Python

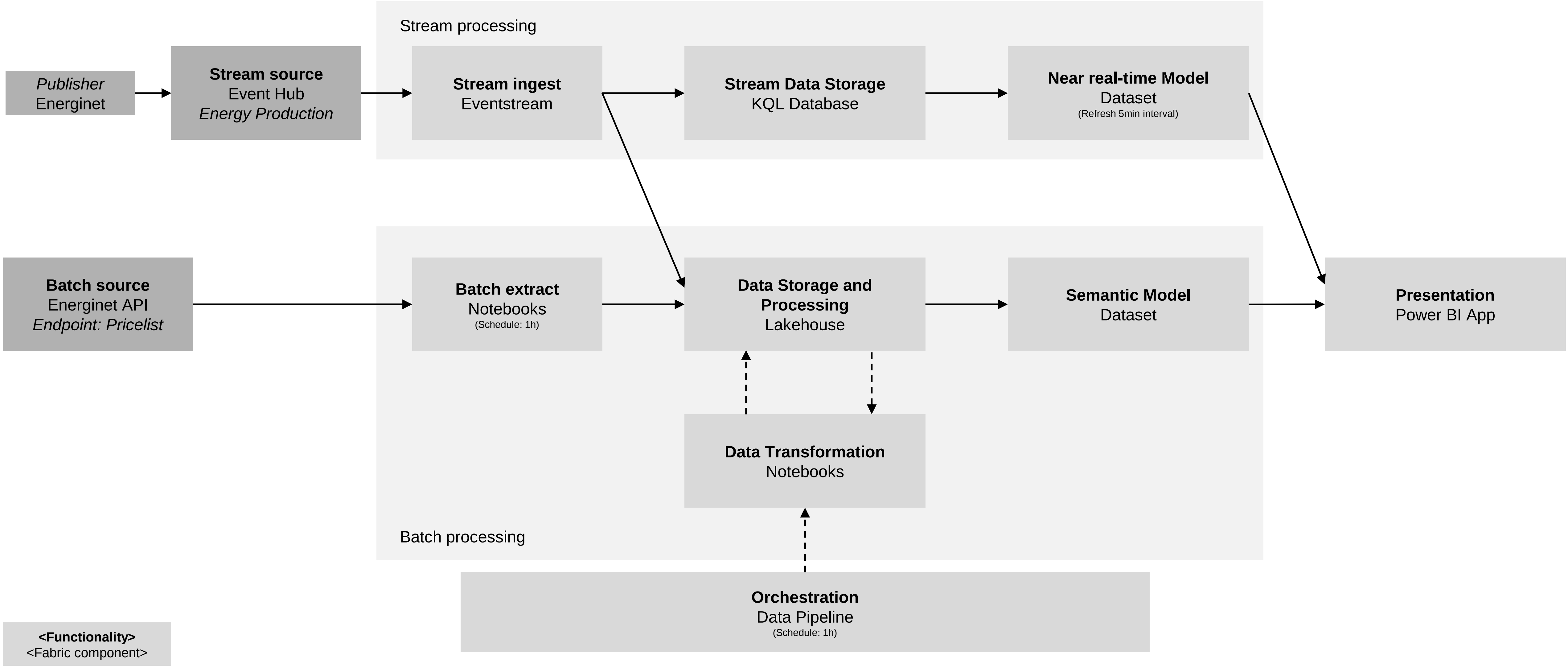
3

The solution

The Lakehouse integrates streaming and batch data seamlessly, exposing both to Power BI

INSPARI
a valantic company

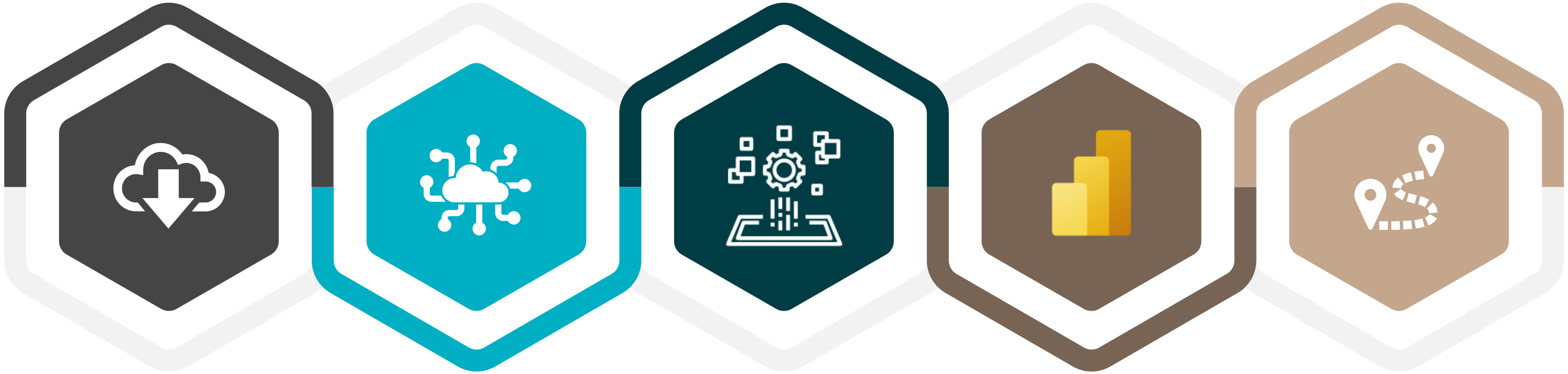
A Lambda architecture allows for real-time insights for business users and long-term persistence for AI/ML use cases



INSPARI
a valantic company

Solution Buildup

Starting point: requirements have been delivered by management



1. Ingest price data

Data is pulled hourly from REST API using a Notebook

3. Transform data

Python executed in Notebooks are utilized to transform data into classic star schema

5. Orchestration

Workloads are triggered by the use of Data Factory

2. Stream production data

Data is streamed from event hub into KQL and Lakehouse

4. Blend stream & Batch in Power BI

Data is exposed in Power BI

Solution Buildup

Starting point: requirements have been delivered by management

1. Ingest price data

Data is pulled hourly from REST API using a Notebook

2. Stream production data

Data is streamed from event hub into KQL and Lakehouse

3. Transform data

Python executed in Notebooks are utilized to transform data into classic star schema

4. Blend stream & Batch in Power BI

Data is exposed in Power BI

5. Orchestration

Workloads are triggered by the use of Data Factory

1. Ingest Price Data

Requirements

- Ingest data on an hourly schedule
- Integrate to an external API

Considerations for Experience choice

A Notebook was chosen because:

- **Pro:** Ability to be scheduled by other resources, e.g. Pipelines

- **Pro:** Ability to integrate to external APIs and handle integration errors using Python

- **Pro:** Can easily persist data in a Lakehouse experience

- **Con:** Testability and code re-use is not well supported in a Notebook experience

What other options do I have?

- Azure Data Factory/Fabric Pipeline

- Azure Functions

- Logic App

- Fabric Data Flow Gen 2.

Solution Buildup

Starting point: requirements have been delivered by management

1. Ingest price data

Data is pulled hourly from REST API using a Notebook

2. Stream production data

Data is streamed from event hub into KQL and Lakehouse

3. Transform data

Python executed in Notebooks are utilized to transform data into classic star schema

4. Blend stream & Batch in Power BI

Data is exposed in Power BI

5. Orchestration

Workloads are triggered by the use of Data Factory

2. Stream Production data

Requirements

- Integrate to Event Hub that receives streaming data

Considerations for Experience choice

An Eventstream and KQL Database was chosen because:

- **Pro:** Ability to write to many sinks (KQL Database, Lakehouse) by single Experience (Eventstream)
- **Pro:** High Ease of use
- **Pro:** Scales to large amount of messages
- **Con:** Transformation capabilities are few.

What other options do I have?

- Azure Stream Analytics
- Spark Streaming / Spark Job Definition
- Azure Functions

Solution Buildup

Starting point: requirements have been delivered by management

1. Ingest price data

Data is pulled hourly from REST API using a Notebook

3. Transform data

Python executed in Notebooks are utilized to transform data into classic star schema

5. Orchestration

Workloads are triggered by the use of Data Factory

2. Stream production data

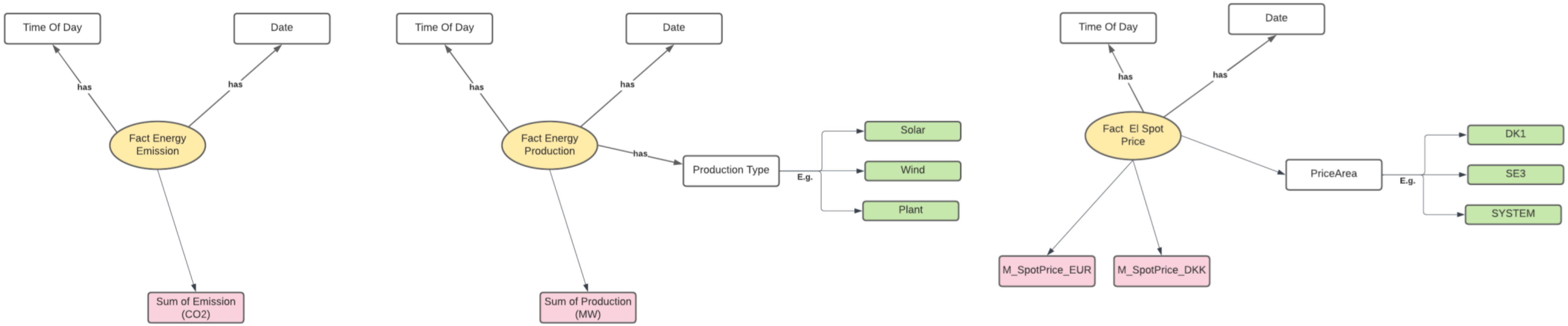
Data is streamed from event hub into KQL and Lakehouse

4. Blend stream & Batch in Power BI

Data is exposed in Power BI

Conceptual model

Understand before you build



3. Transform data

Requirements

- Integrate natively with the Lakehouse experience
- Allow for defining transformations in Python, SQL, etc.
- Allow for Version Control and Collaboration

Considerations for Experience choice

As the team surrounding Kayla prefer to code in Python, Notebooks was chosen because:

- **Pro:** Fits well for the company’s context
- **Pro:** Highly flexible with great version control
- **Pro:** Rich ecosystem of libraries and tools
- **Con:** Performance overhead and dependency management of libraries / versioning

What other options do I have?

- Warehouse with SQL:
  - Highly performant
  - Widely used language
  - Limited flexibility

Solution Buildup

Starting point: requirements have been delivered by management

1. Ingest price data

Data is pulled hourly from REST API using a Notebook

3. Transform data

Python executed in Notebooks are utilized to transform data into classic star schema

5. Orchestration

Workloads are triggered by the use of Data Factory

2. Stream production data

Data is streamed from event hub into KQL and Lakehouse

4. Blend stream & Batch in Power BI

Data is exposed in Power BI

4. Blend stream & Batch in Power BI

Requirements

- Possibility of exposing data in a Semantic Mode
- Possibility of creating Measures using DAX

Considerations for Experience choice

A Power BI Semantic model was chosen because:

Pro: Fabric ensures a cohesive and consistent user experience across different tools and platforms

Pro: Data sharing capabilities

Con: Limited in terms of highly specialized or custom visualizations compared to a fully custom web application

What other options do I have?

- Azure Data Explorer
  - Great for showing real-time analytics, but less so for a dimensional model
- Custom Web Application
  - Highly customizable
  - High effort and expensive compared to other solutions

Solution Buildup

Starting point: requirements have been delivered by management

1. Ingest price data

Data is pulled hourly from REST API using a Notebook

2. Stream production data

Data is streamed from event hub into KQL and Lakehouse

3. Transform data

Python executed in Notebooks are utilized to transform data into classic star schema

4. Blend stream & Batch in Power BI

Data is exposed in Power BI using a composite model

5. Orchestration

Workloads are triggered by the use of Data Factory

5. Orchestration

Requirements

- Simple to set up
- Must work with Artifacts in Fabric

Considerations for Experience choice

Data Factory is chosen as orchestration tool because:

Pro: Seamless integration with other Fabric artifacts

Pro: Trigger functionality

Con: Best suited for simple workflows

What other options do I have?

- Azure Functions
- Spark Job Definitions

Solution Wrap-up

Starting point: requirements has been delivered by management

1. Ingest price data

Data is pulled hourly from REST api using a Notebook

2. Stream production data

Data is streamed from event hub into KQL and Lakehouse

3. Transform data

Python executed in Notebooks are utilized to transform data into classic star schema

4. Blend stream & Batch in Power BI

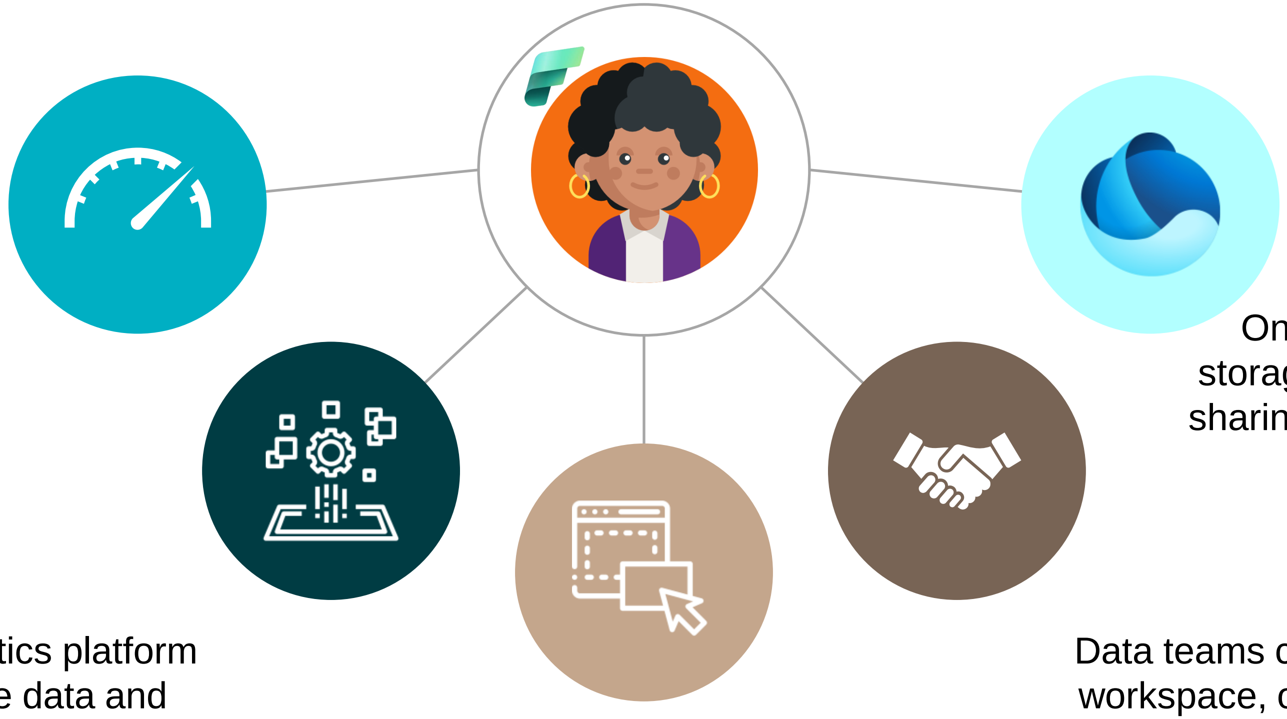
Data is exposed in Power BI

5. Orchestration

Workloads are triggered by the use of Data Factory

Takeaways

Microsoft Fabric benefits for the Data Engineer



Rapid development
Not dealing with infrastructure allows for rapid building, testing failing and succeeding

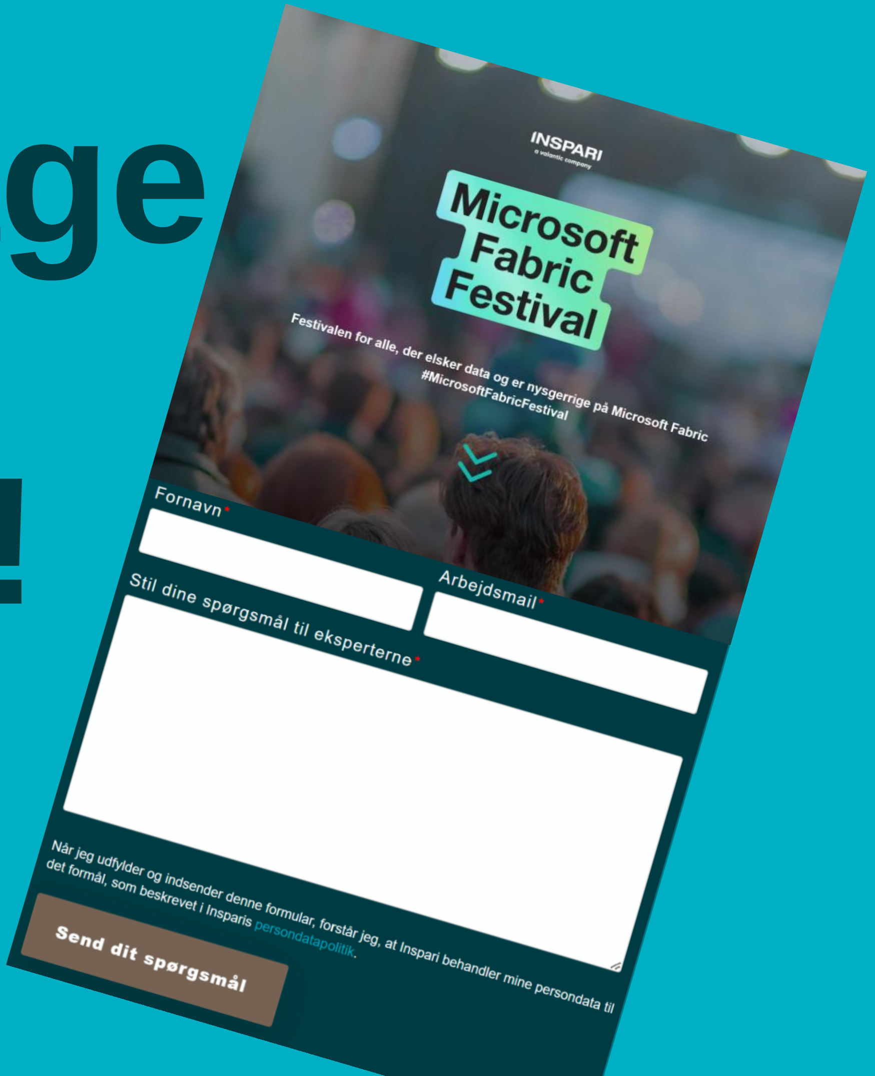
End-to-end solution
A complete, unified analytics platform that brings together all the data and analytics tool you need

Skillset compatibility
Engineers can work with different frameworks and languages

OneLake: OneDrive for data
OneLake provides a single, unified storage system, where discovery and sharing of data of any format between users and applications is easy

Collaboration
Data teams can collaborate in a single workspace, on the same copy of data, with centralized administration, governance, and compliance tools

See you Backstage at the Festival, ask me anything!



Microsoft Fabric Festival

Festivalen for alle, der elsker data og er nysgerrige på Microsoft Fabric
#MicrosoftFabricFestival

Fornavn*

Arbejdsmail*

Stil dine spørgsmål til eksperterne*

Når jeg udfylder og indsender denne formular, forstår jeg, at Inspari behandler mine persondata til det formål, som beskrevet i Insparis persondatapolitik.