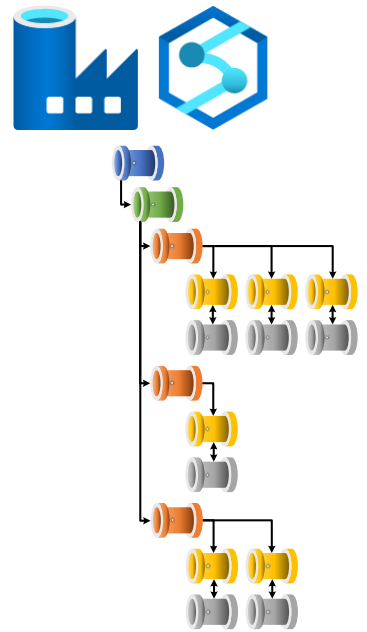
Creating a Metadata Driven Processing Framework



Using Azure Integration Pipelines

Paul Andrew | Group Manager & Analytics Architect

Microsoft® MVP Most Valuable Professional

avanade

@MrPaulAndrew

In/MrPaulAndrew

MrPaulAndrew.com

https://github.com/mrpaulandrew

CommunityEvents

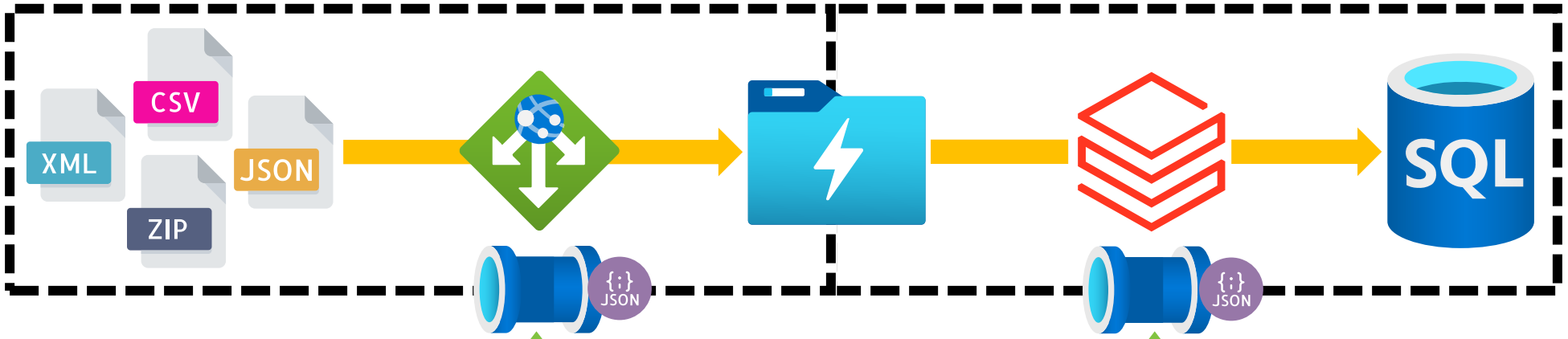
Demo code, content and slides from various community events.

C++

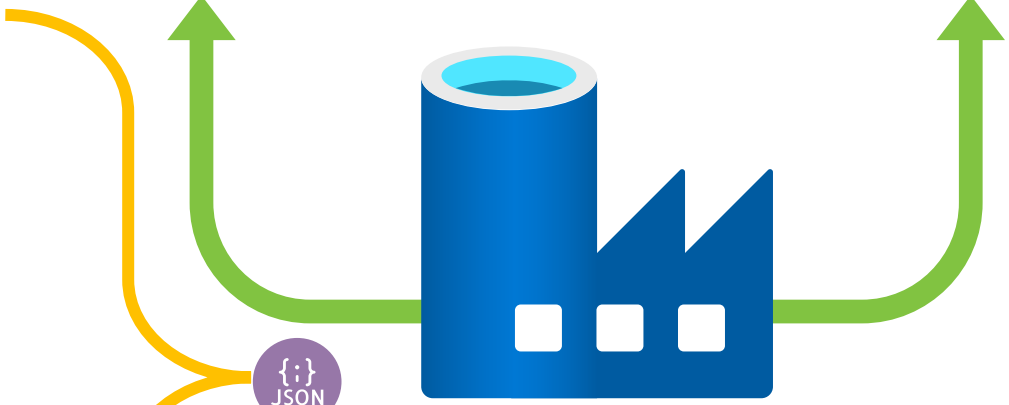
{Event/Location}-{Month}-{Year}

Data Factory – A Quick Overview

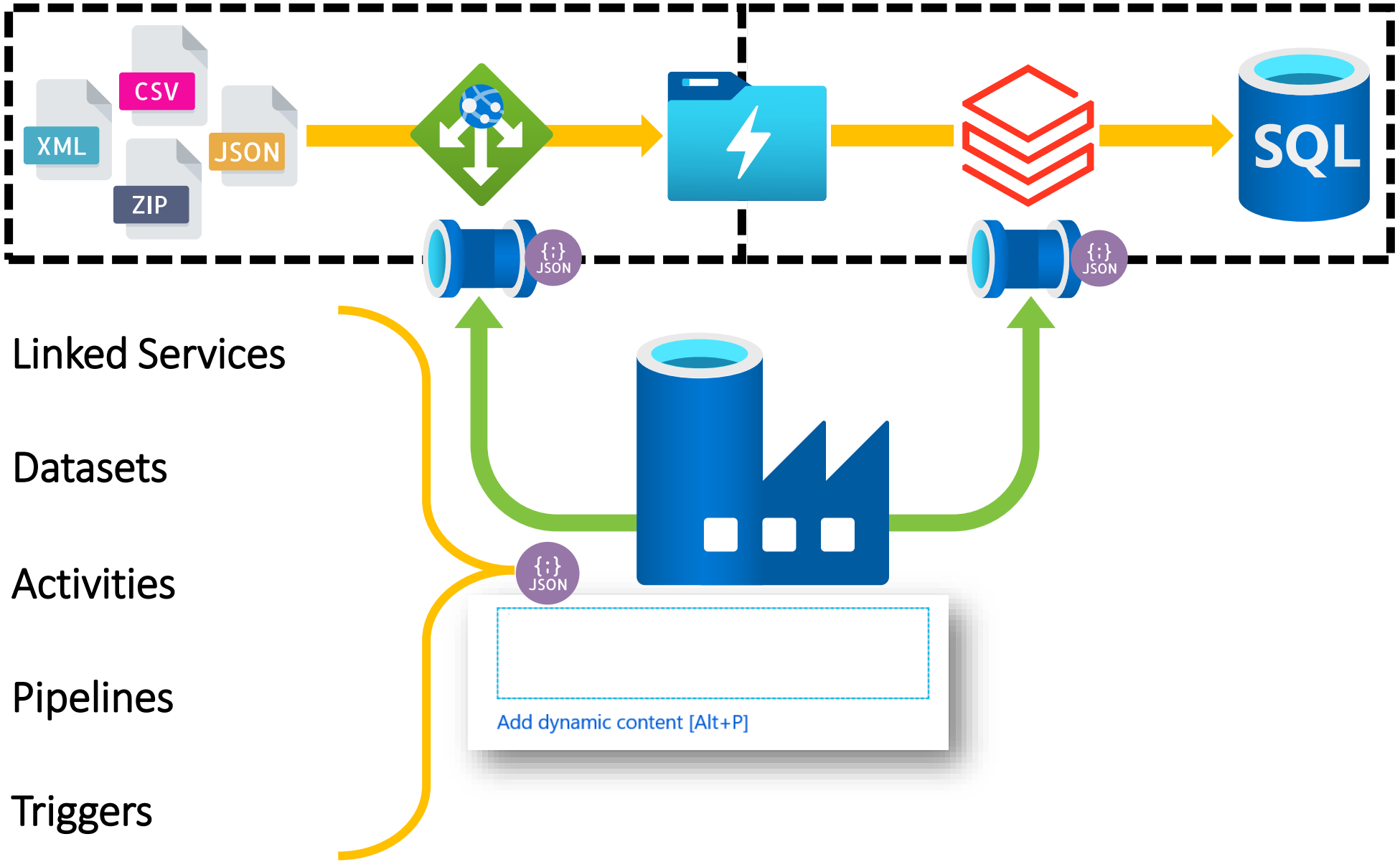
Data Factory A Quick Overview



1. Linked Services
2. Datasets
3. Activities
4. Pipelines
5. Triggers



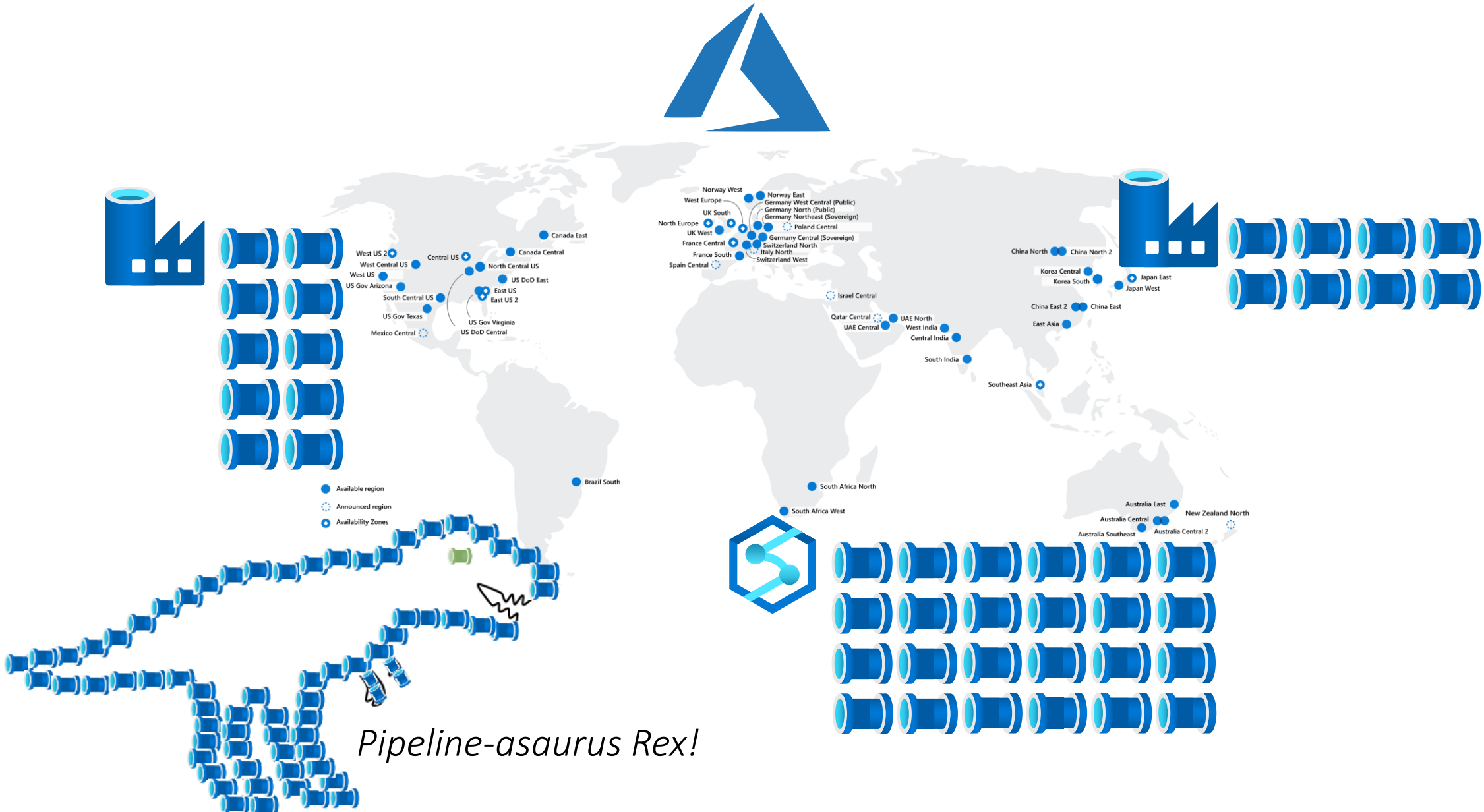
Data Factory Components – Add Dynamic Content



1. Linked Services
2. Datasets
3. Activities
4. Pipelines
5. Triggers

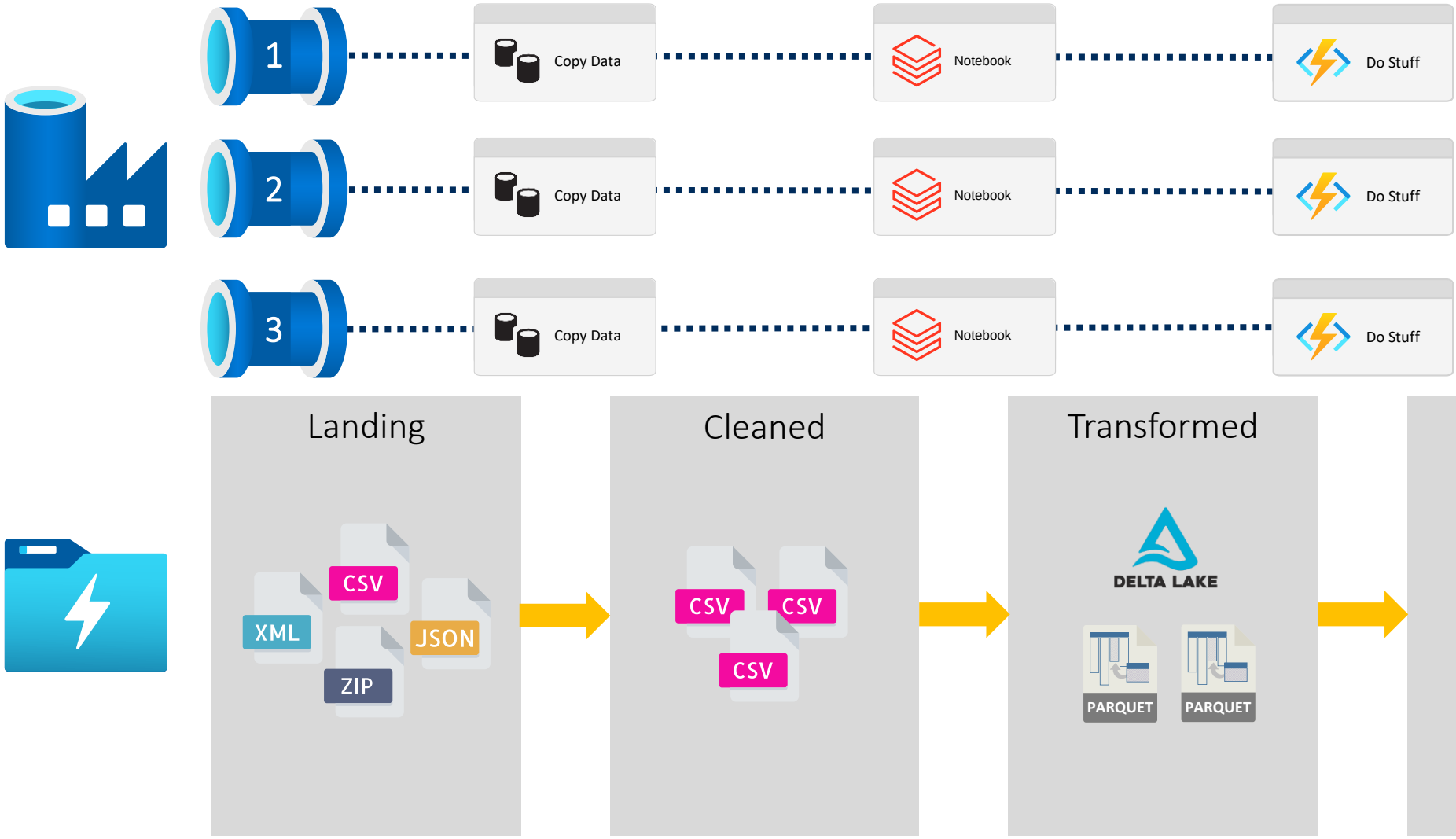
Problem

Problem: How should we structure our Integration Pipelines?

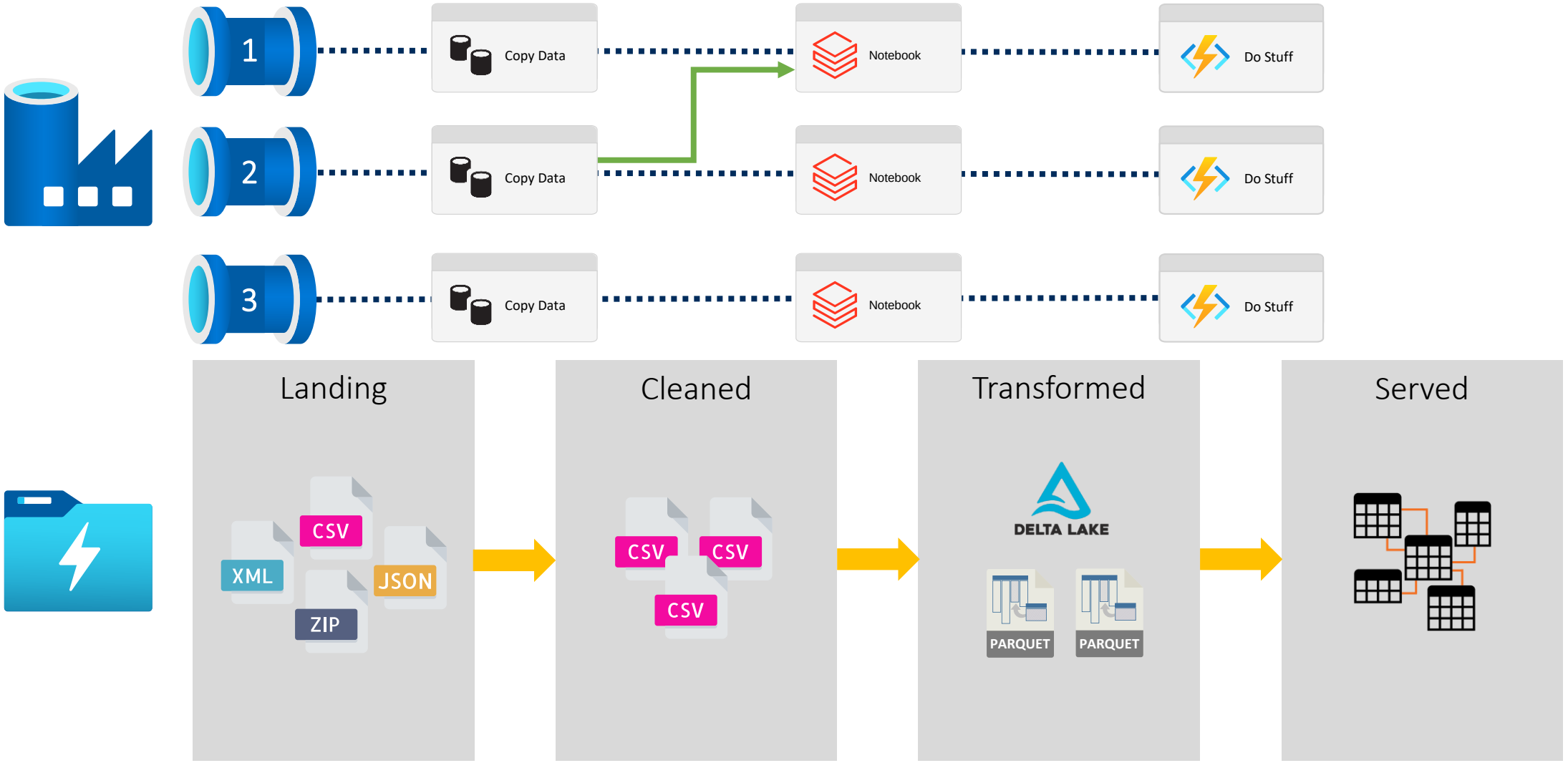


Pipeline-asaurus Rex!

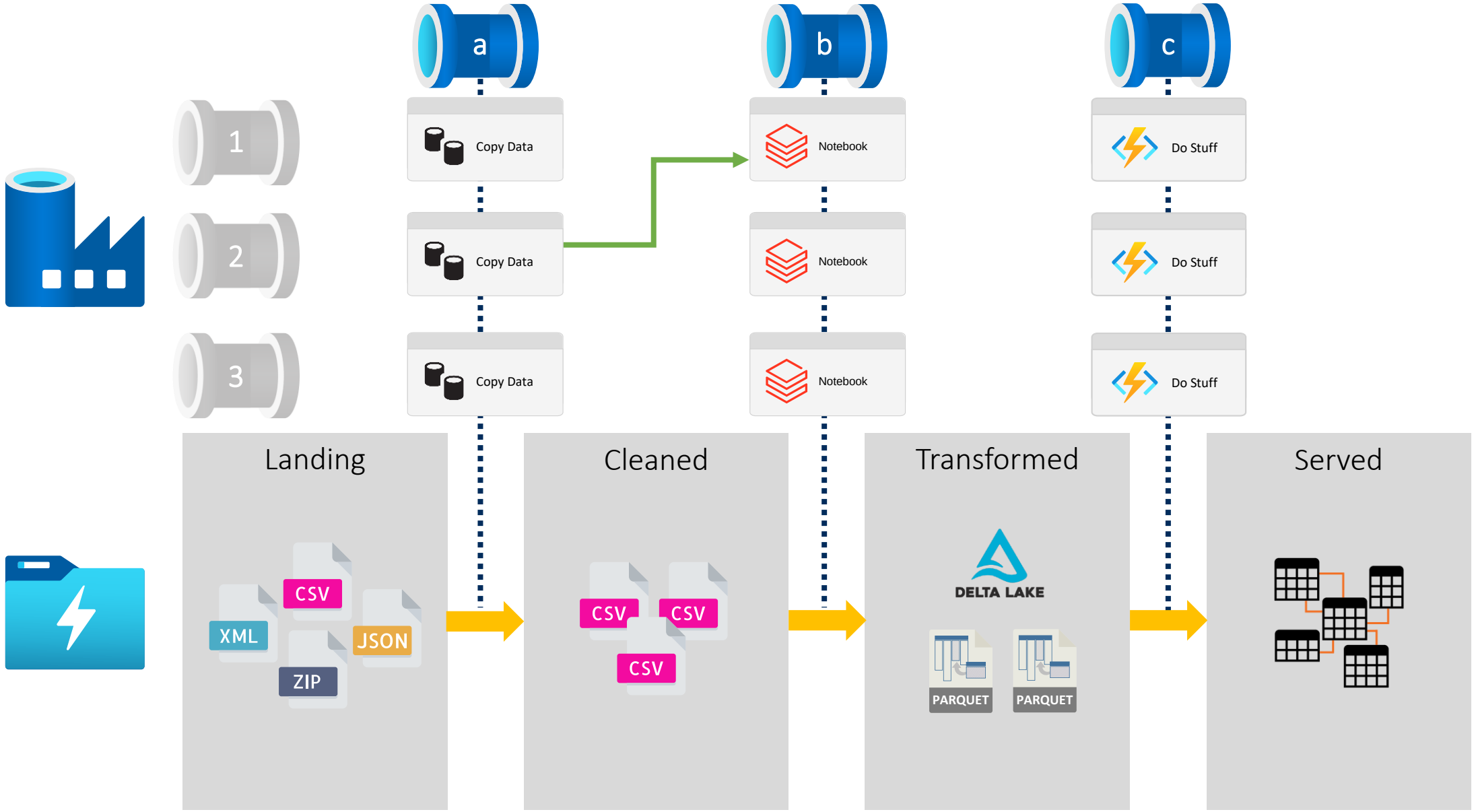
Problem


1
Copy Data
Notebook
Do Stuff
2
Copy Data
Notebook
Do Stuff
3
Copy Data
Notebook
Do Stuff
Landing
CSV
XML
JSON
ZIP
Cleaned
CSV
CSV
CSV
Transformed
DELTA LAKE
PARQUET
PARQUET
Served

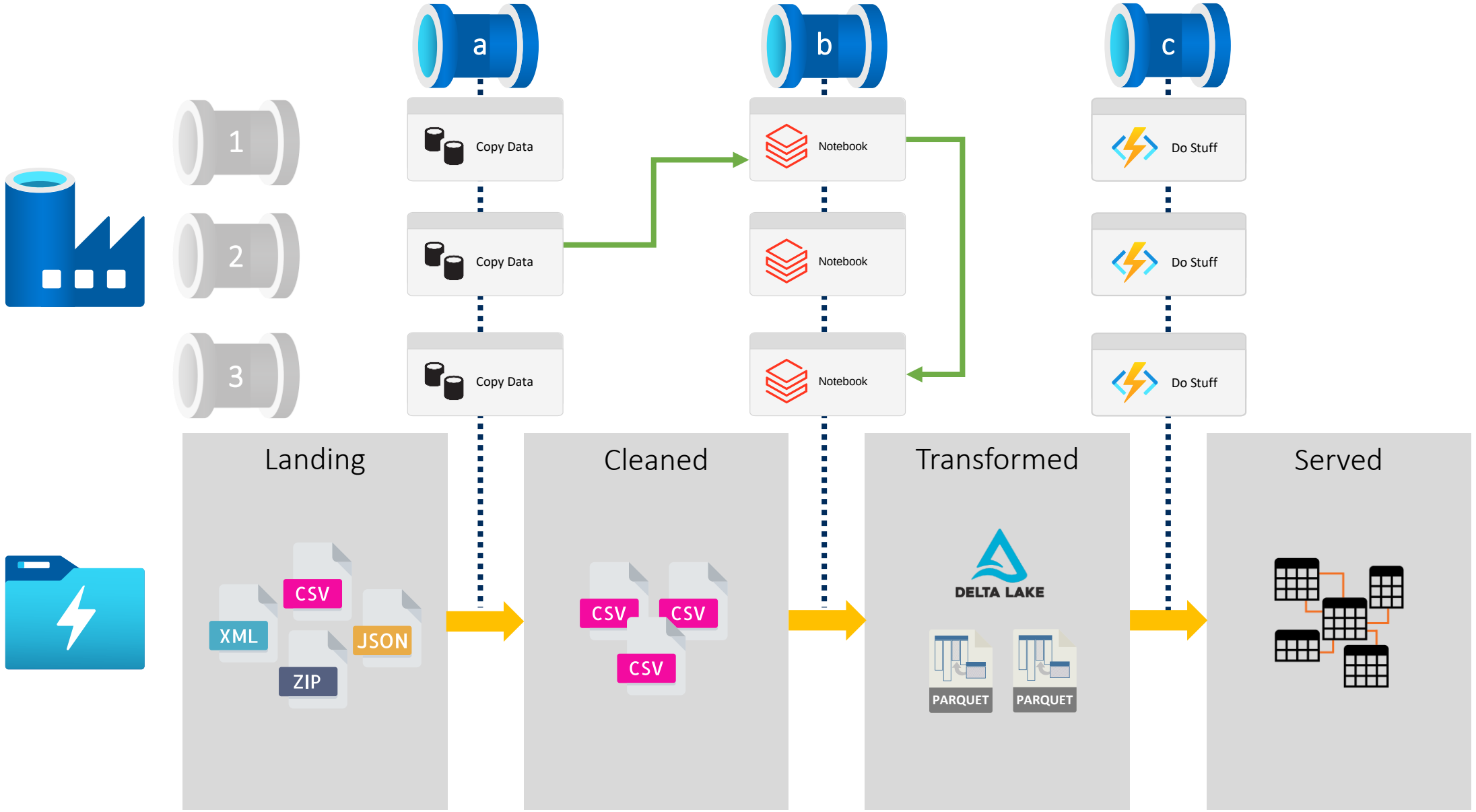
Problem


1
2
3
Copy Data
Copy Data
Copy Data
Notebook
Notebook
Notebook
Do Stuff
Do Stuff
Do Stuff
Landing
Cleaned
Transformed
Served
CSV
XML
JSON
ZIP
CSV
CSV
CSV
DELTA LAKE
PARQUET
PARQUET

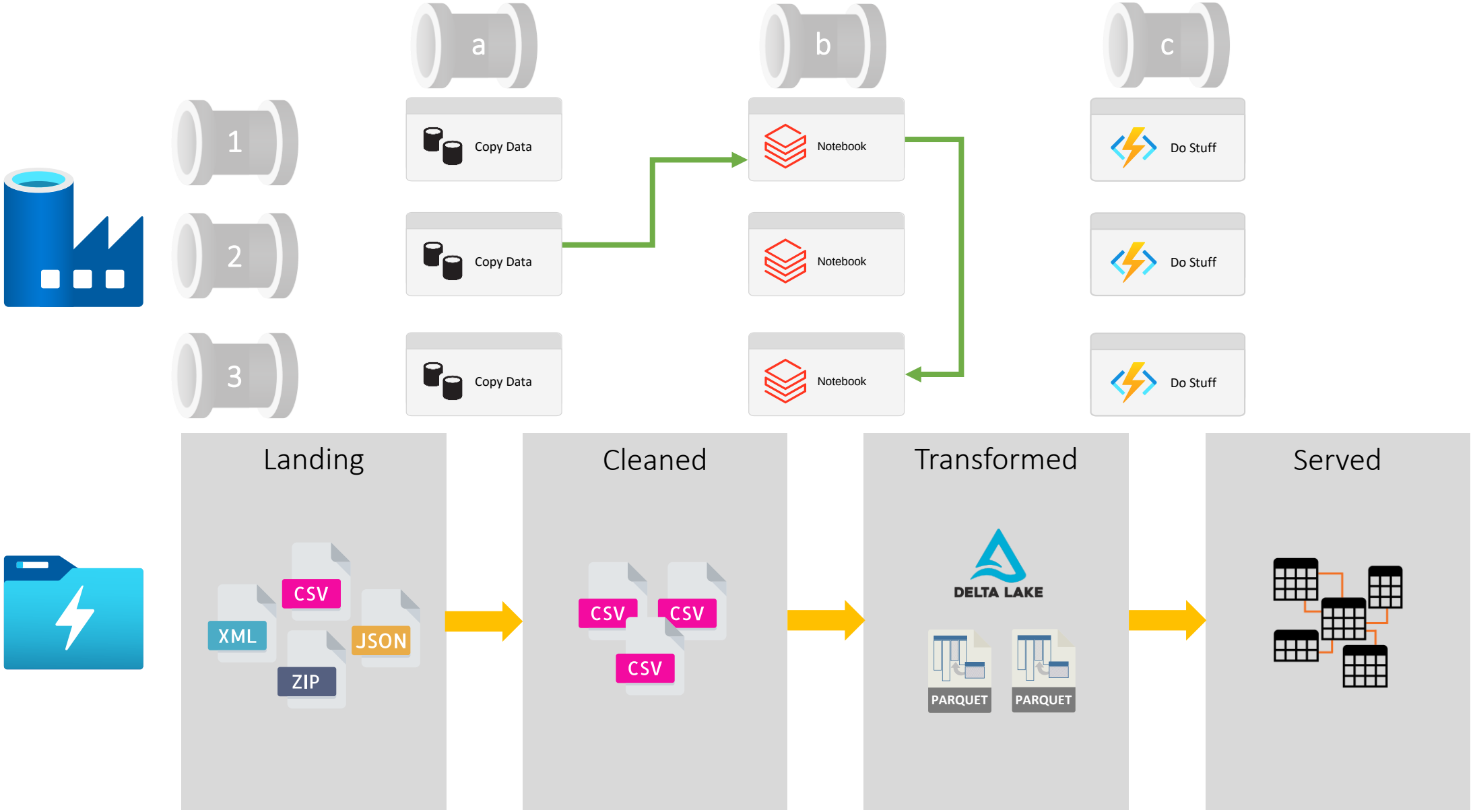
Problem


a
b
c
1
2
3
Copy Data
Copy Data
Copy Data
Notebook
Notebook
Notebook
Do Stuff
Do Stuff
Do Stuff
Landing
Cleaned
Transformed
Served
CSV
XML
JSON
ZIP
CSV
CSV
CSV
DELTA LAKE
PARQUET
PARQUET

Problem

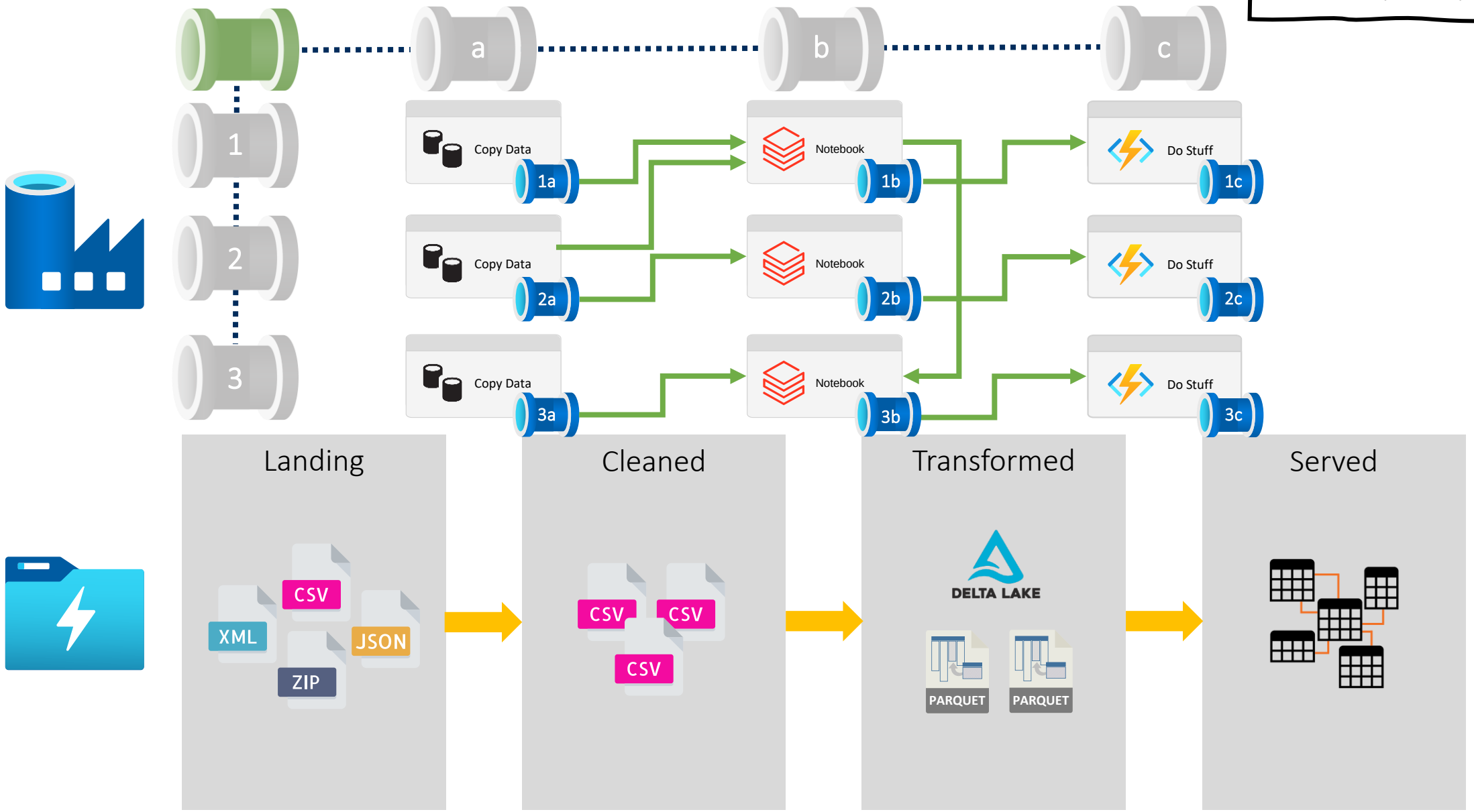

a
b
c
1
2
3
Copy Data
Copy Data
Copy Data
Notebook
Notebook
Notebook
Do Stuff
Do Stuff
Do Stuff
Landing
Cleaned
Transformed
Served
CSV
XML
JSON
ZIP
CSV
CSV
CSV
DELTA LAKE
PARQUET
PARQUET

Problem

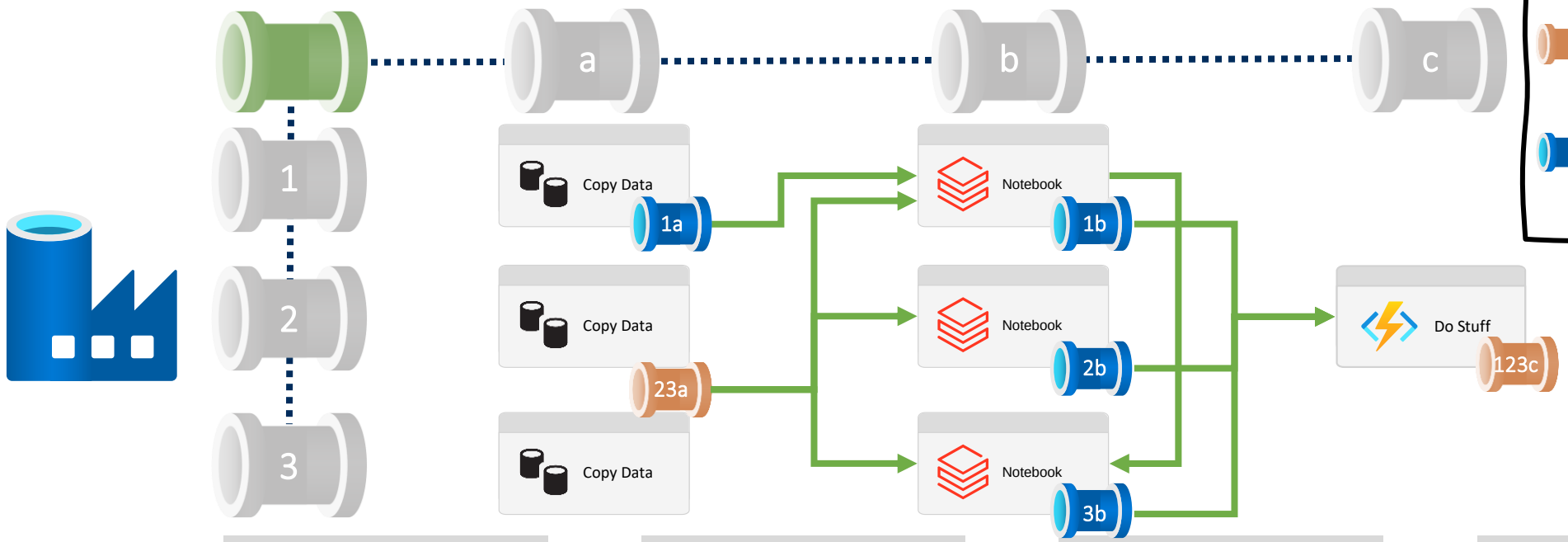

a
b
c
1
2
3
Copy Data
Copy Data
Copy Data
Notebook
Notebook
Notebook
Do Stuff
Do Stuff
Do Stuff
Landing
Cleaned
Transformed
Served
CSV
XML
JSON
ZIP
CSV
CSV
CSV
DELTA LAKE
PARQUET
PARQUET

Problem

Only 40 Activities per Pipeline.



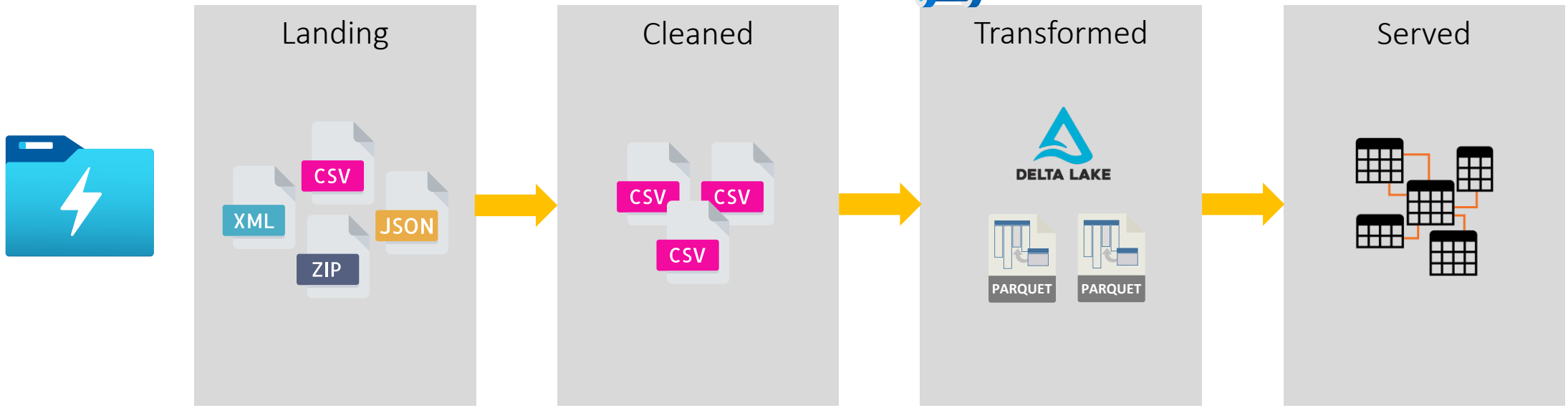
Problem



Grandparent pipeline for all processing.

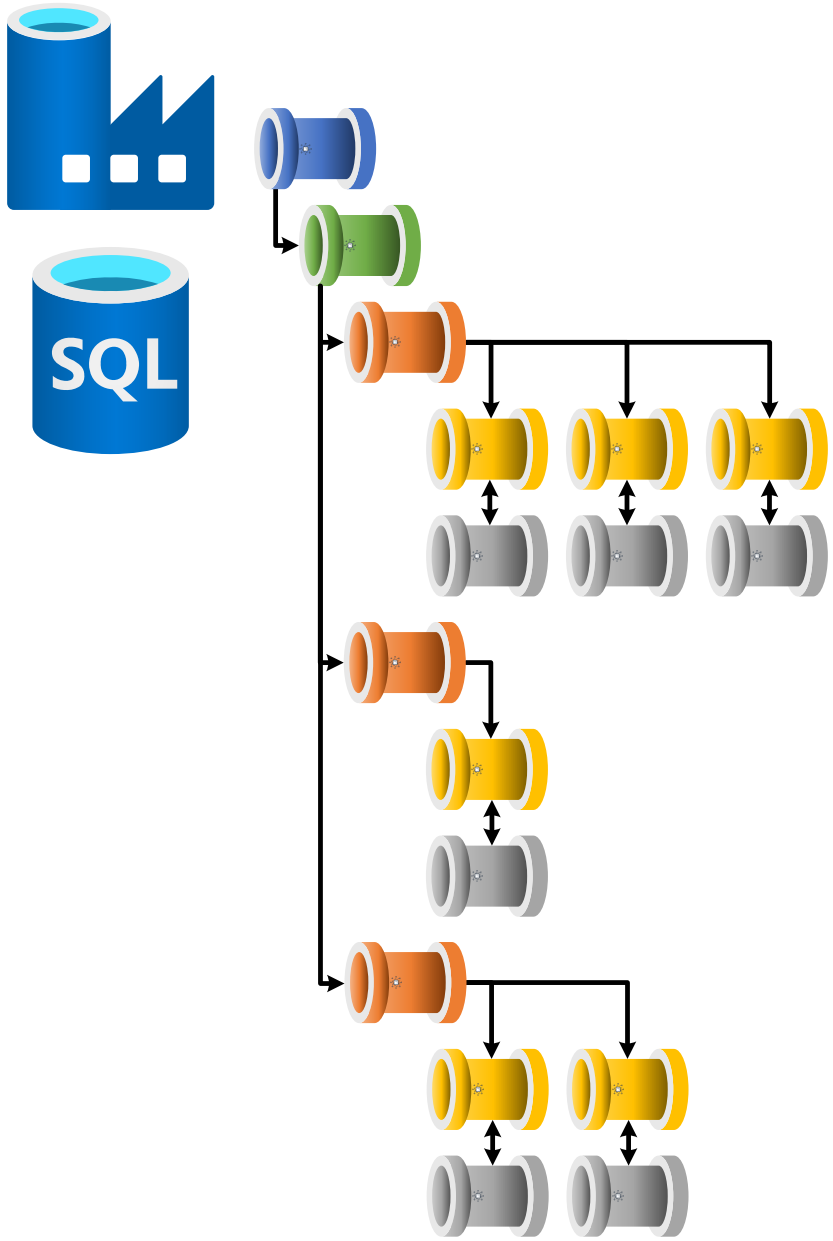
Parent pipeline to consolidate work.

Child pipelines for low level dependencies.

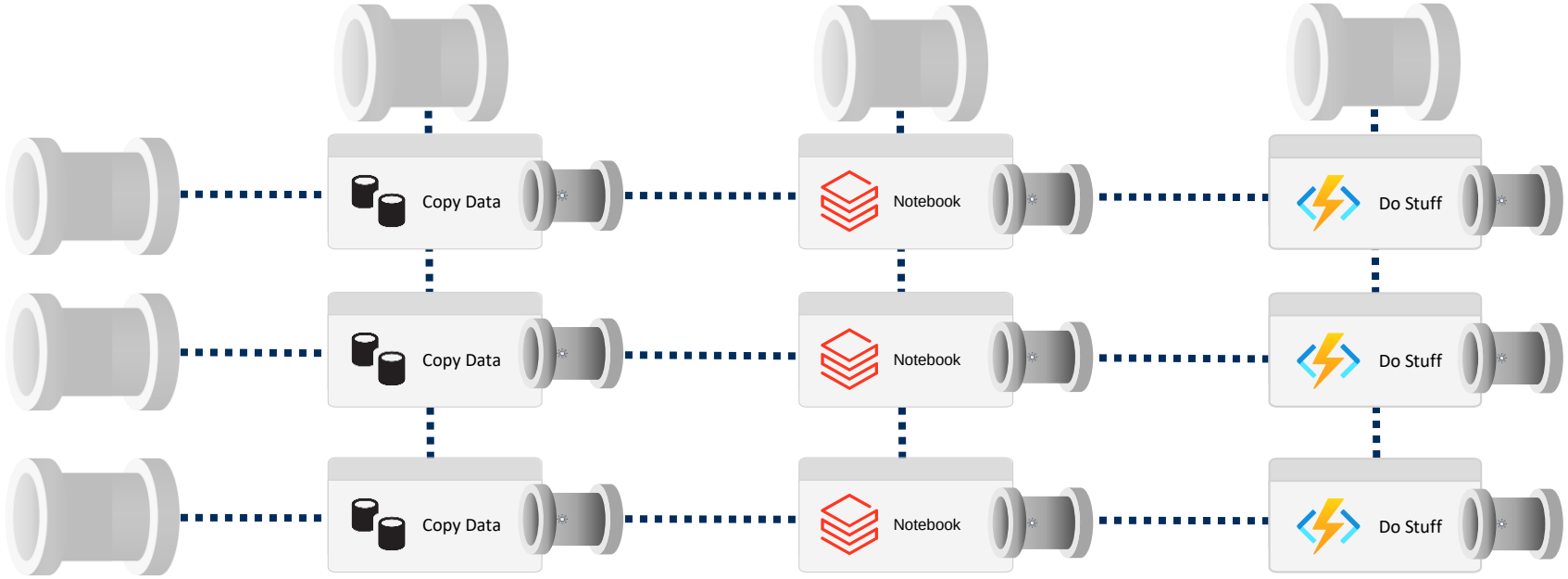


Solution

Solution: Use Metadata to Drive Data Factory Pipelines



Solution

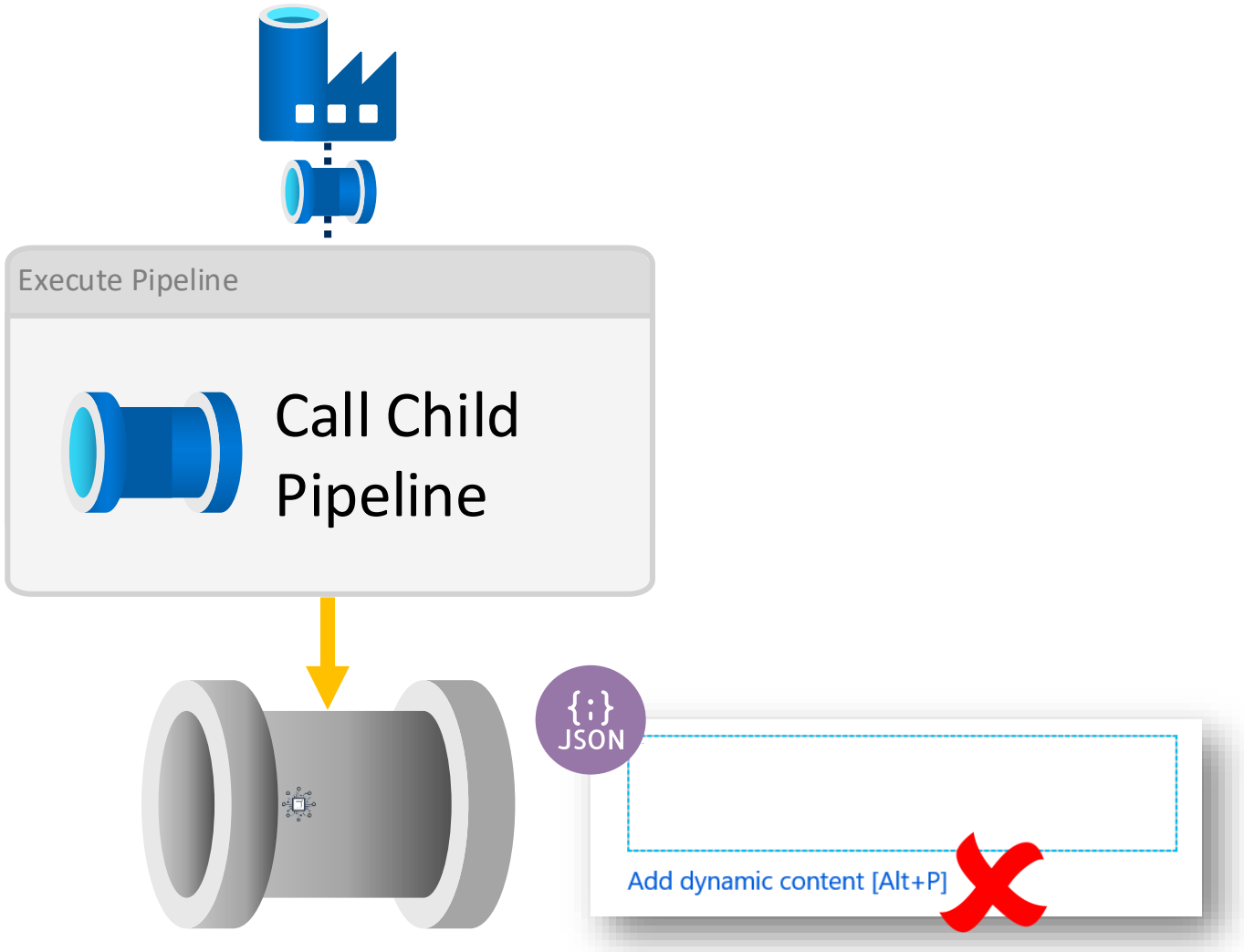


Stages
1
2
3

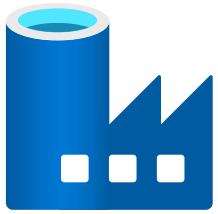
Pipelines
a
b
c
d
e
f
g
h
i

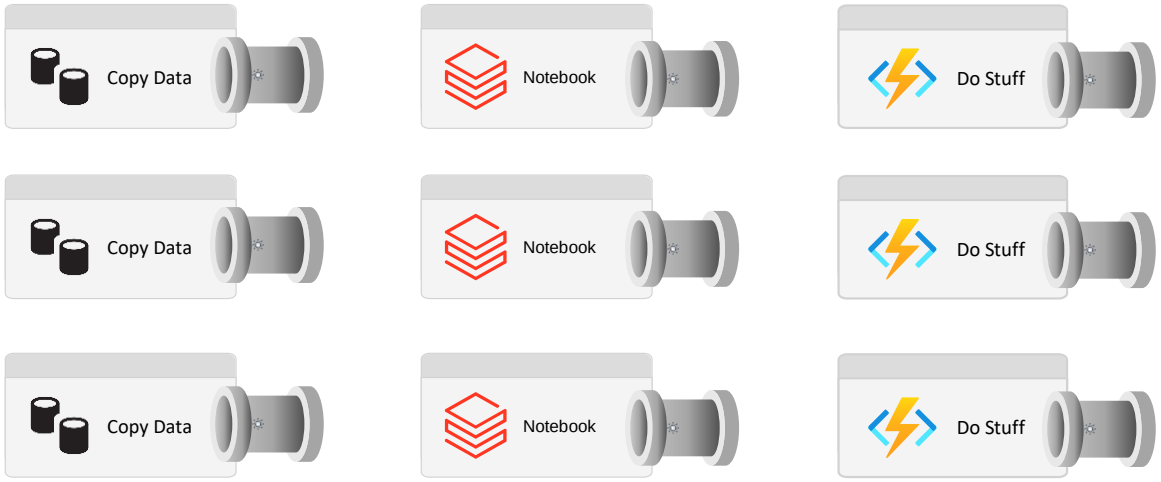
Stage	Pipeline
1	a
1	b
1	c
2	d
2	e
3	f
3	g
3	h
3	i

One More Problem



Calling Our Worker Pipelines



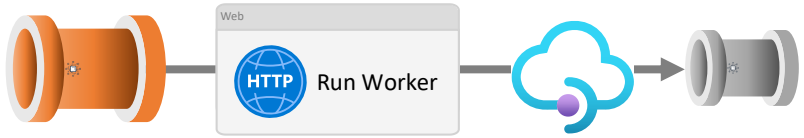


Stages
1
2
3

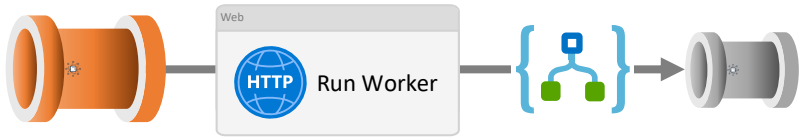
Pipelines
a
b
c
d
e
f
g
h
i

Stage	Pipeline
1	a
1	b
1	c
2	d
2	e
3	f
3	g
3	h
3	i

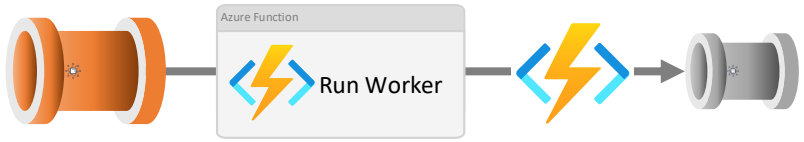
Option 1:



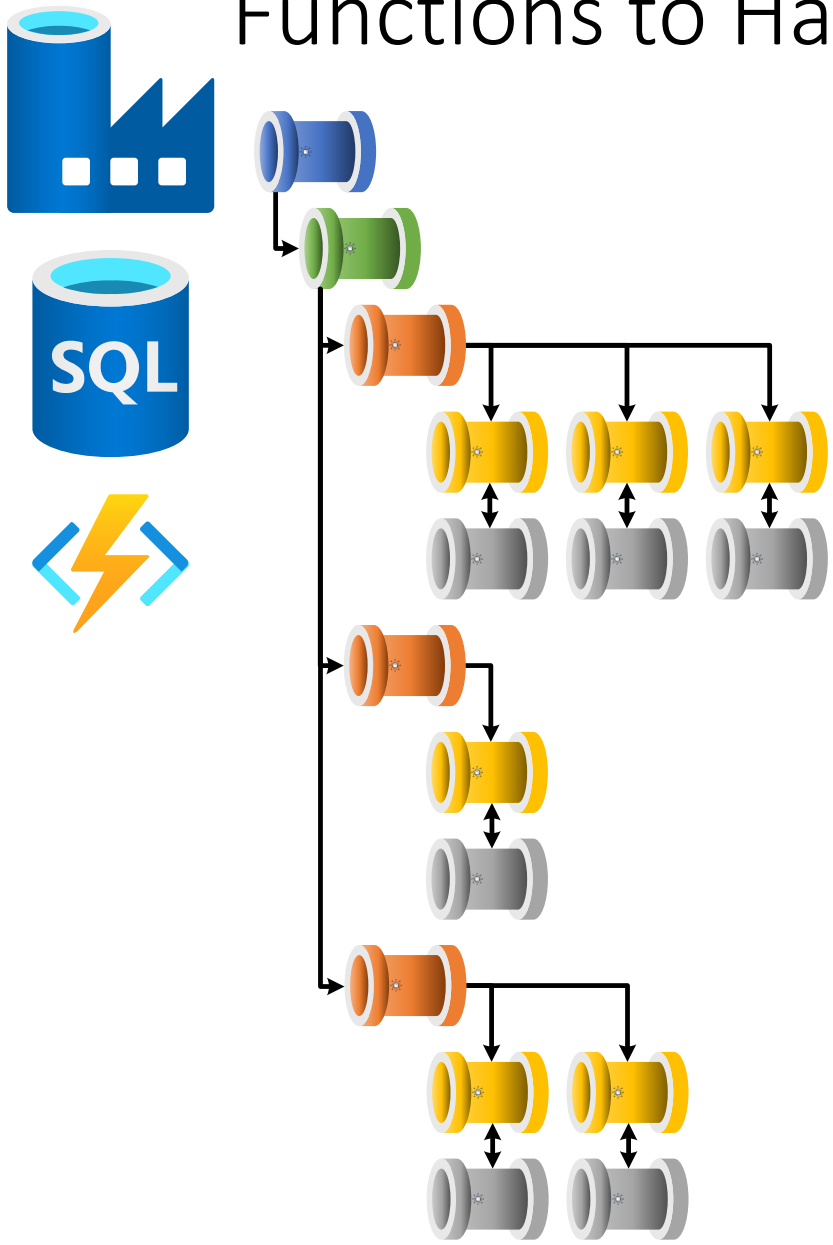
Option 2:



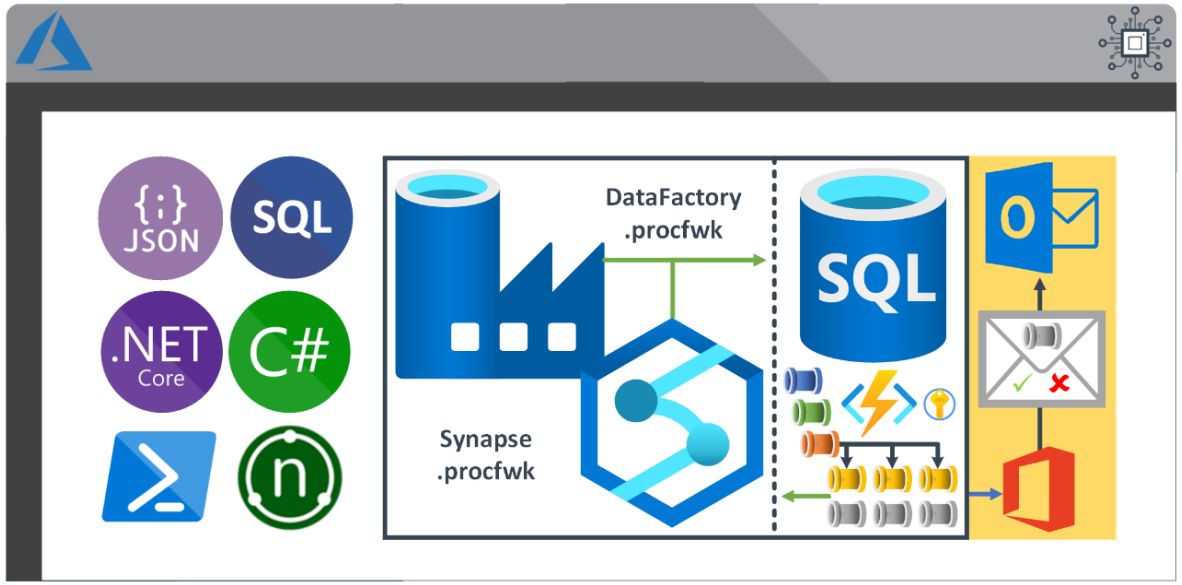
Option 3:



Solution: Use Metadata to Drive Data Factory Pipelines & Functions to Handle the Worker Execution

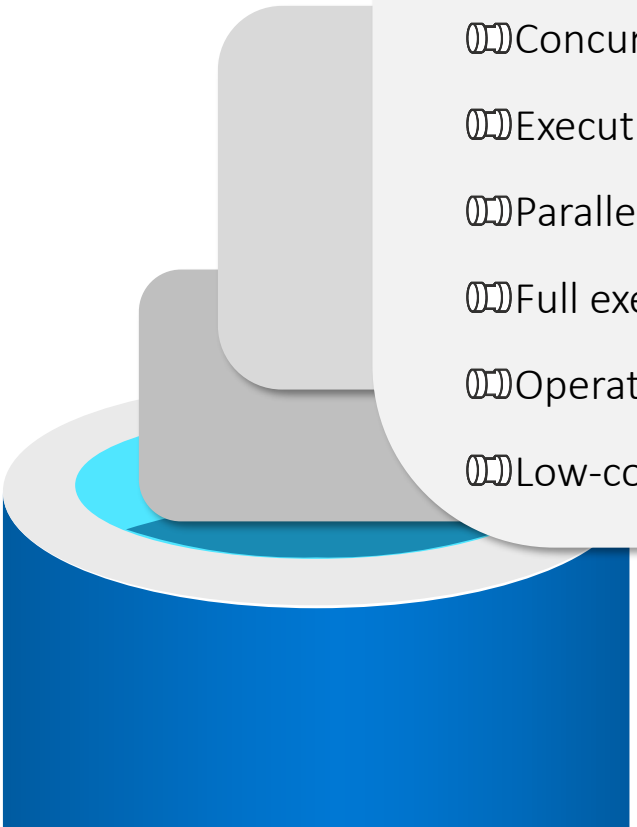


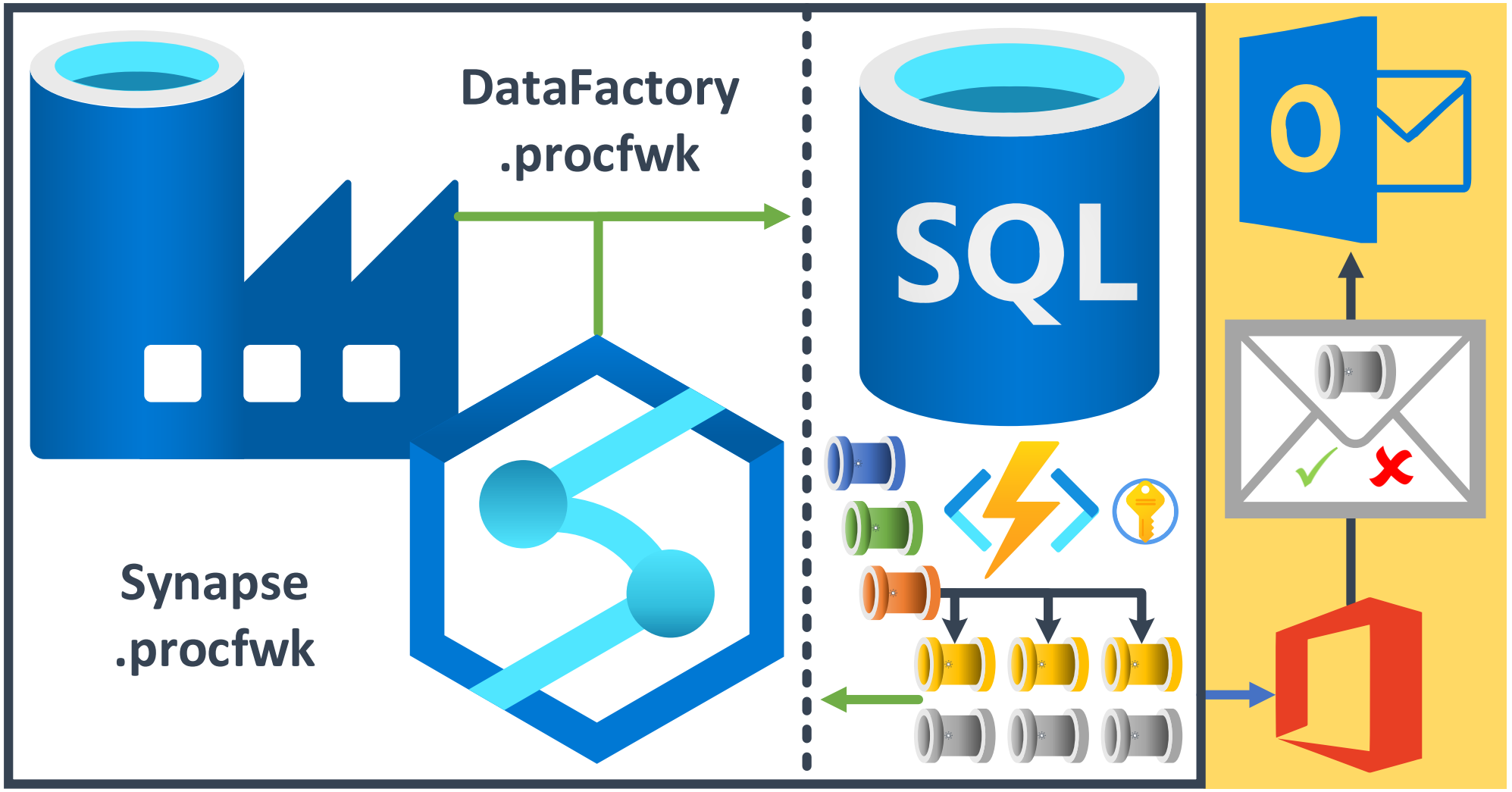
Introducing procfwk.com

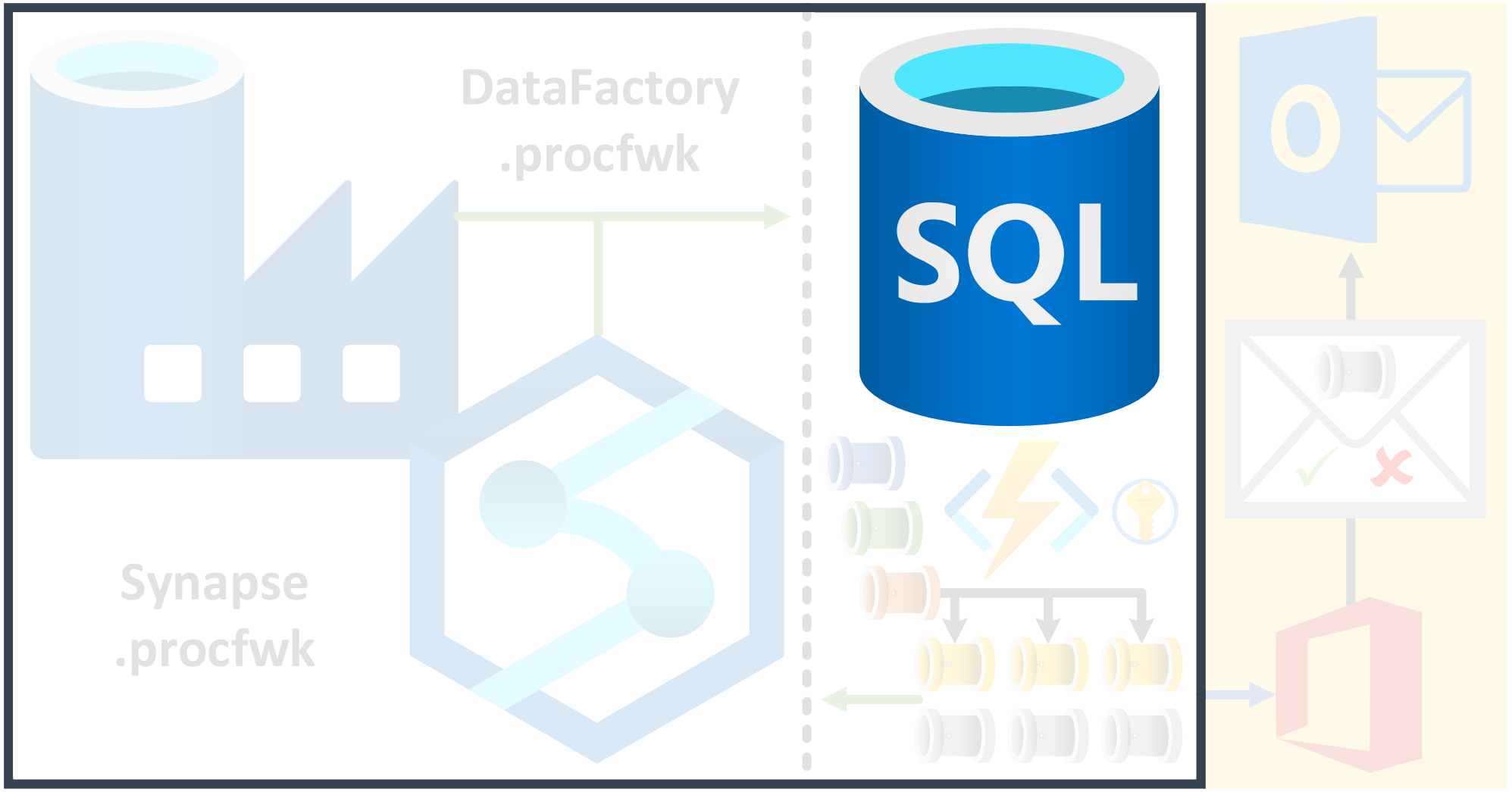


procfwk Features

- Granular metadata control.
- Metadata integrity checking.
- Global properties.
- Complete pipeline dependency chains.
- Concurrent batch executions.
- Execution restart-ability.
- Parallel execution stages.
- Full execution and error logs.
- Operational dashboarding.
- Low-cost orchestration.
- Disconnection between framework and worker pipelines.
- Cross Tenant/Subscription/Data Factory control flows.
- Pipeline parameter support.
- Simple troubleshooting.
- Easy deployment.
- Email alerting.
- Automated testing.
- Azure Key Vault integration.
- Is pipeline already running checks.




DataFactory
.procfwk
SQL
Synapse
.procfwk


DataFactory
.procfwk
SQL
Synapse
.procfwk

Framework Database

AlertOutcomes (procfwk)
- OutcomeBitPosition
- PipelineOutcomeStatus
- BitValue

PipelineAlertLink (procfwk)
- AlertId
- PipelineId
- RecipientId
- OutcomesBitValue
- Enabled

Recipients (procfwk)
- RecipientId
- Name
- EmailAddress
- MessagePreference
- Enabled

Batches (procfwk)
- BatchId
- BatchName
- BatchDescription
- Enabled

PipelineParameters (procfwk)
- ParameterId
- PipelineId
- ParameterName
- ParameterValue
- ParameterValueLastUsed

Pipelines (procfwk)
- PipelineId
- OrchestratorId
- StageId
- PipelineName
- LogicalPredecessorId
- Enabled

BatchStageLink (procfwk)
- BatchId
- StageId

PipelineDependencies (procfwk)
- DependencyId
- PipelineId
- DependantPipelineId

Stages (procfwk)
- StageId
- StageName
- StageDescription
- Enabled

Orchestrators (procfwk)
- OrchestratorId
- OrchestratorName
- OrchestratorType
- IsFrameworkOrchestrator
- ResourceGroupName
- SubscriptionId
- Description

PipelineAuthLink (procfwk)
- AuthId
- PipelineId
- OrchestratorId
- CredentialId

Subscriptions (procfwk)
- SubscriptionId
- Name
- Description
- TenantId

ServicePrincipals (dbo)
- CredentialId
- PrincipalName
- PrincipalId
- PrincipalSecret
- PrincipalIdUrl
- PrincipalSecretUrl

Tenants (procfwk)
- TenantId
- Name
- Description

Configuration & Behaviour

Properties (procfwk)
- PropertyId
- PropertyName
- PropertyValue
- Description
- ValidFrom
- ValidTo

Core Metadata

Execution Handling

BatchExecution (procfwk)
- BatchId
- ExecutionId
- BatchName
- BatchStatus
- StartDateTime
- EndDateTime

ExecutionLog (procfwk)
- LogId
- LocalExecutionId
- StageId
- PipelineId
- CallingDataFactoryName
- ResourceGroupName
- DataFactoryName
- PipelineName
- StartDateTime
- PipelineStatus
- EndDateTime
- AdfPipelineRunId
- PipelineParamsUsed

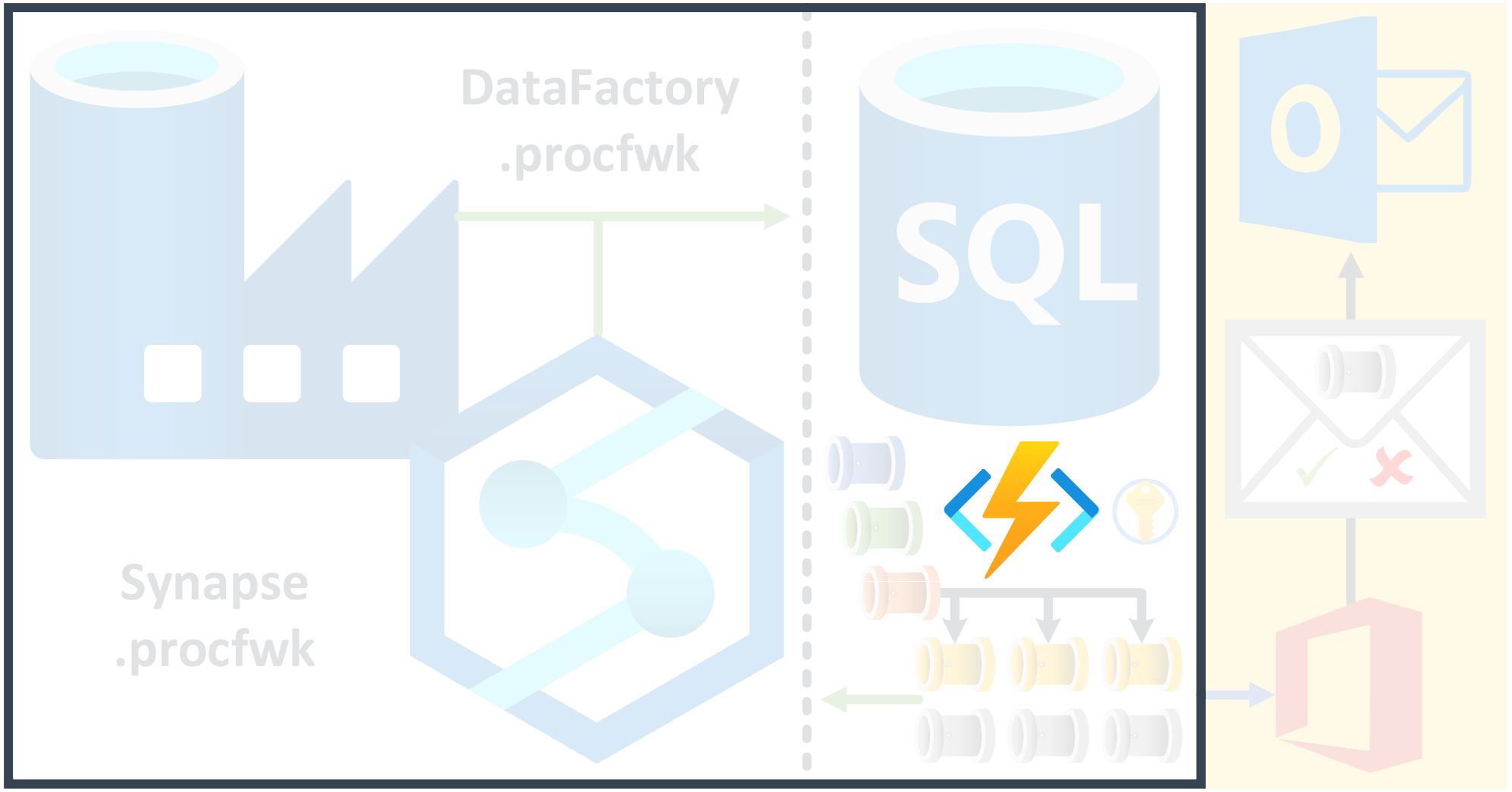
CurrentExecution (procfwk)
- LocalExecutionId
- StageId
- PipelineId
- CallingDataFactoryName
- ResourceGroupName
- DataFactoryName
- PipelineName
- StartDateTime
- PipelineStatus
- LastStatusCheckDateTime
- EndDateTime
- IsBlocked
- AdfPipelineRunId
- PipelineParamsUsed

ErrorLog (procfwk)
- LogId
- LocalExecutionId
- AdfPipelineRunId
- ActivityRunId
- ActivityName
- ActivityType
- ErrorCode
- ErrorType
- ErrorMessage

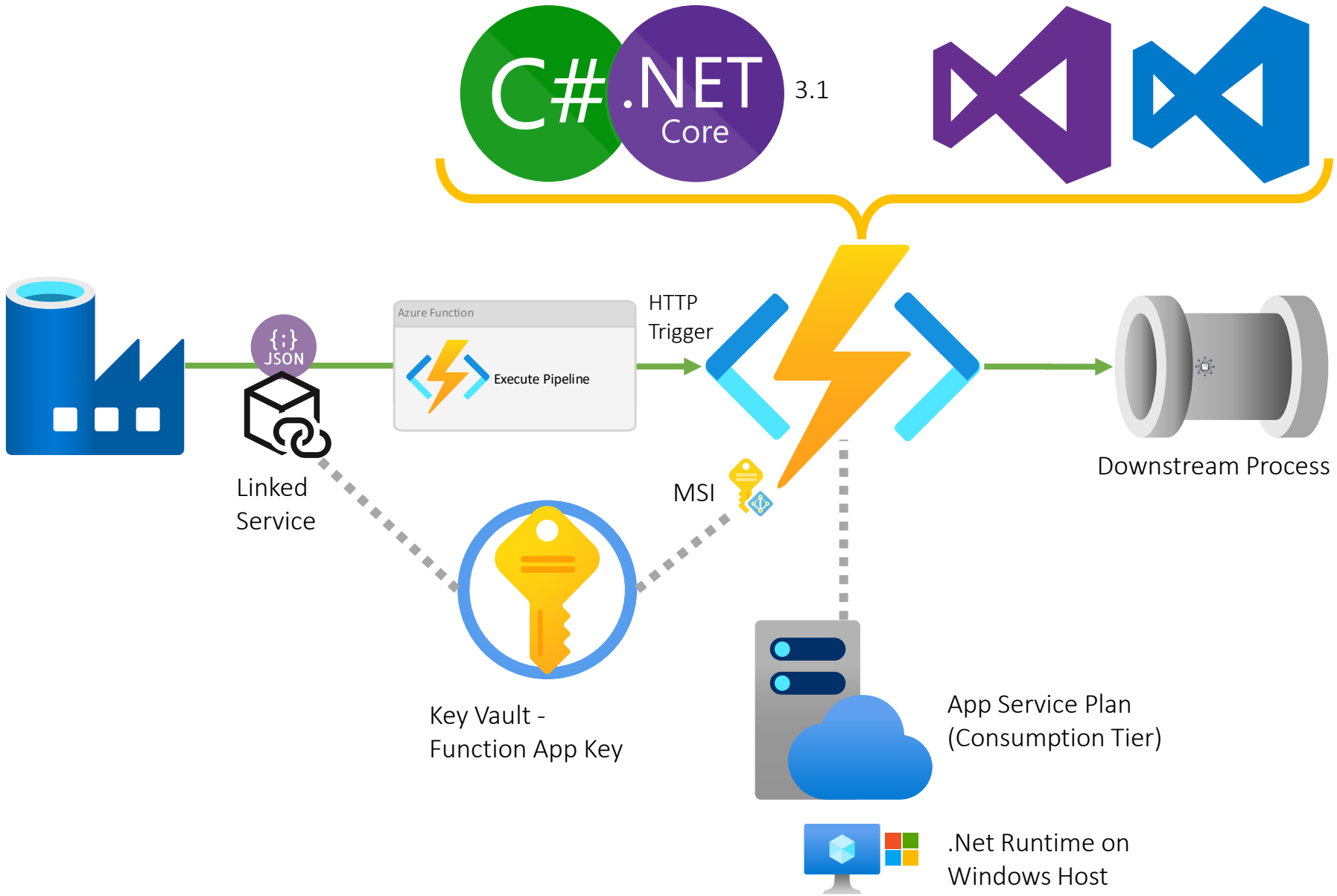
Location & Authentication

Email Alerting

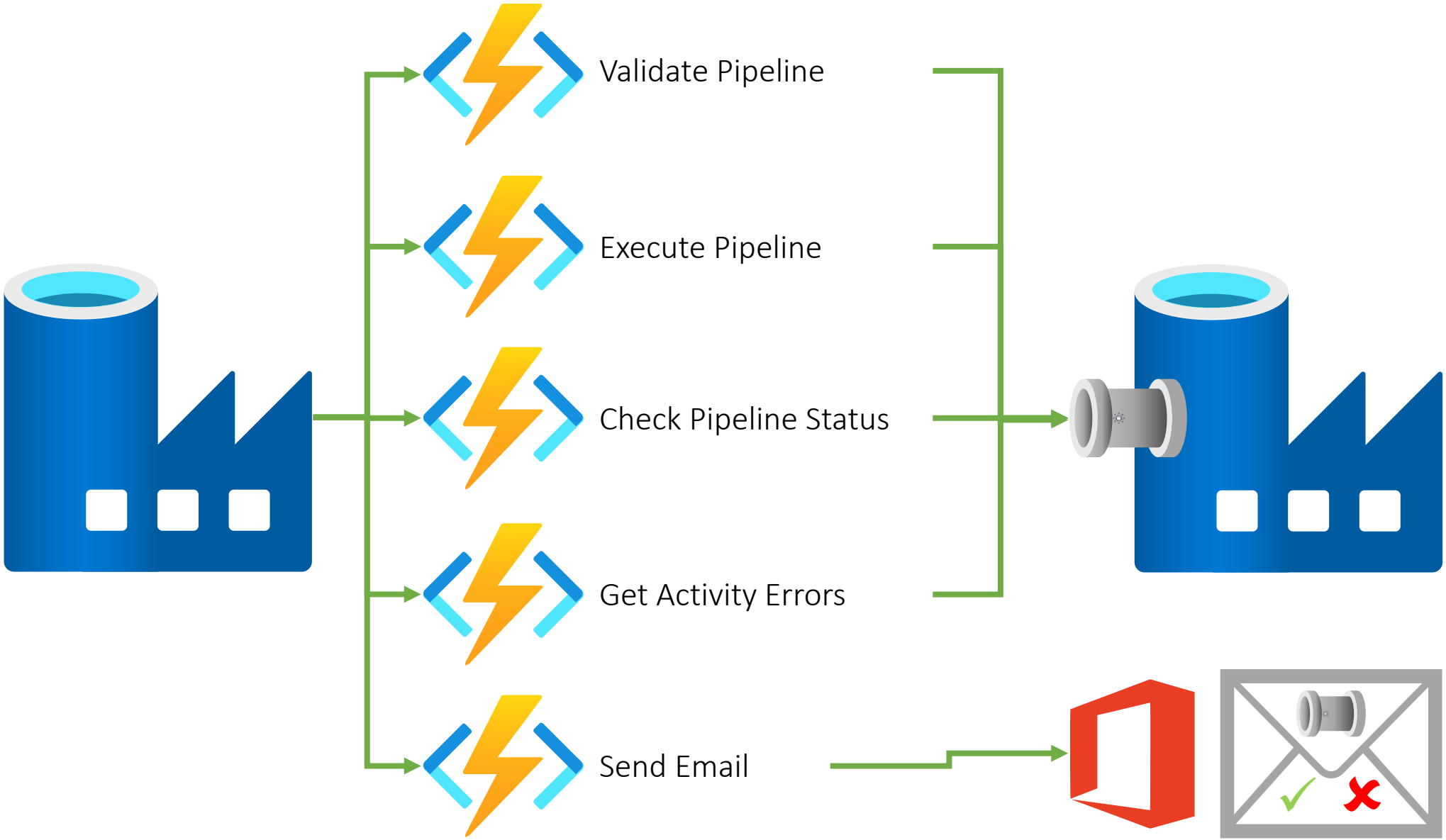
Runtime & Logging

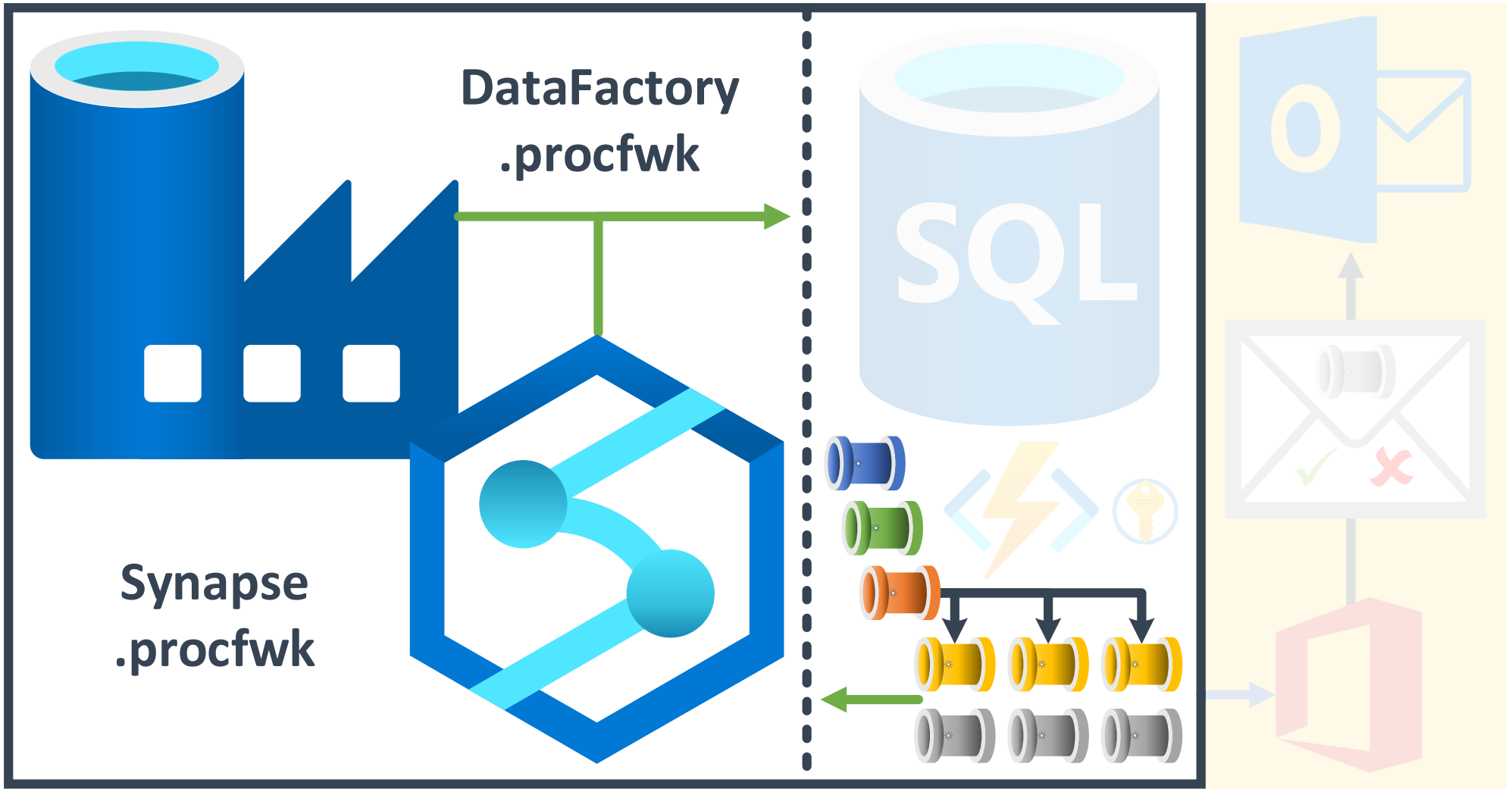

DataFactory
.procfwk
SQL
Synapse
.procfwk

Functions Creation & Configuration

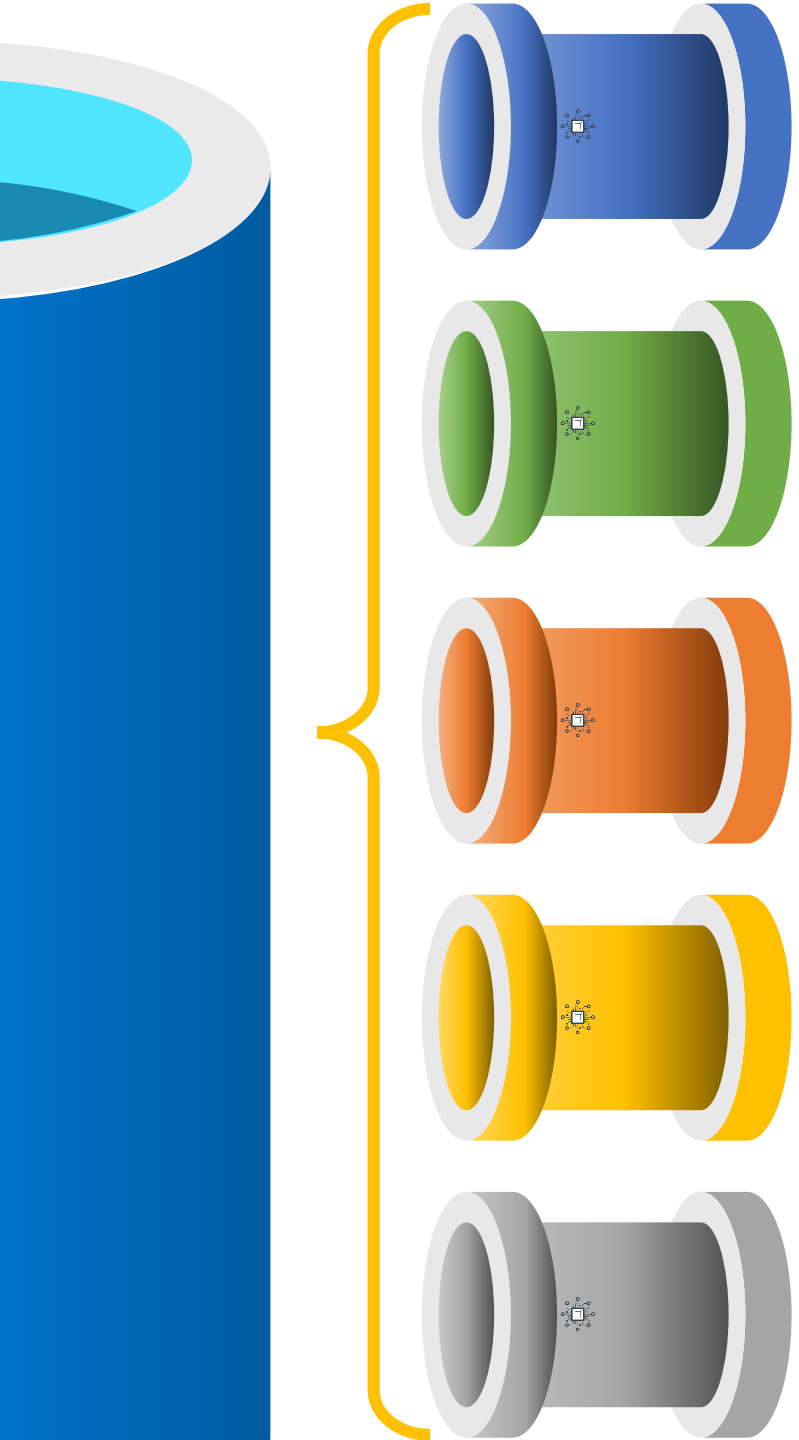


procfwk Functions




DataFactory
.procfwk
SQL
Synapse
.procfwk

Framework Pipeline Hierarchy



- Grandparent

Role: Optional level platform setup, for example, scale up/out compute services ready for the framework to run.

- Parent

Role: Execution run wrapper for batches and execution stage iterator.

- Child

Role: Scale out triggering of worker pipelines within the execution stage(s).

- Infant

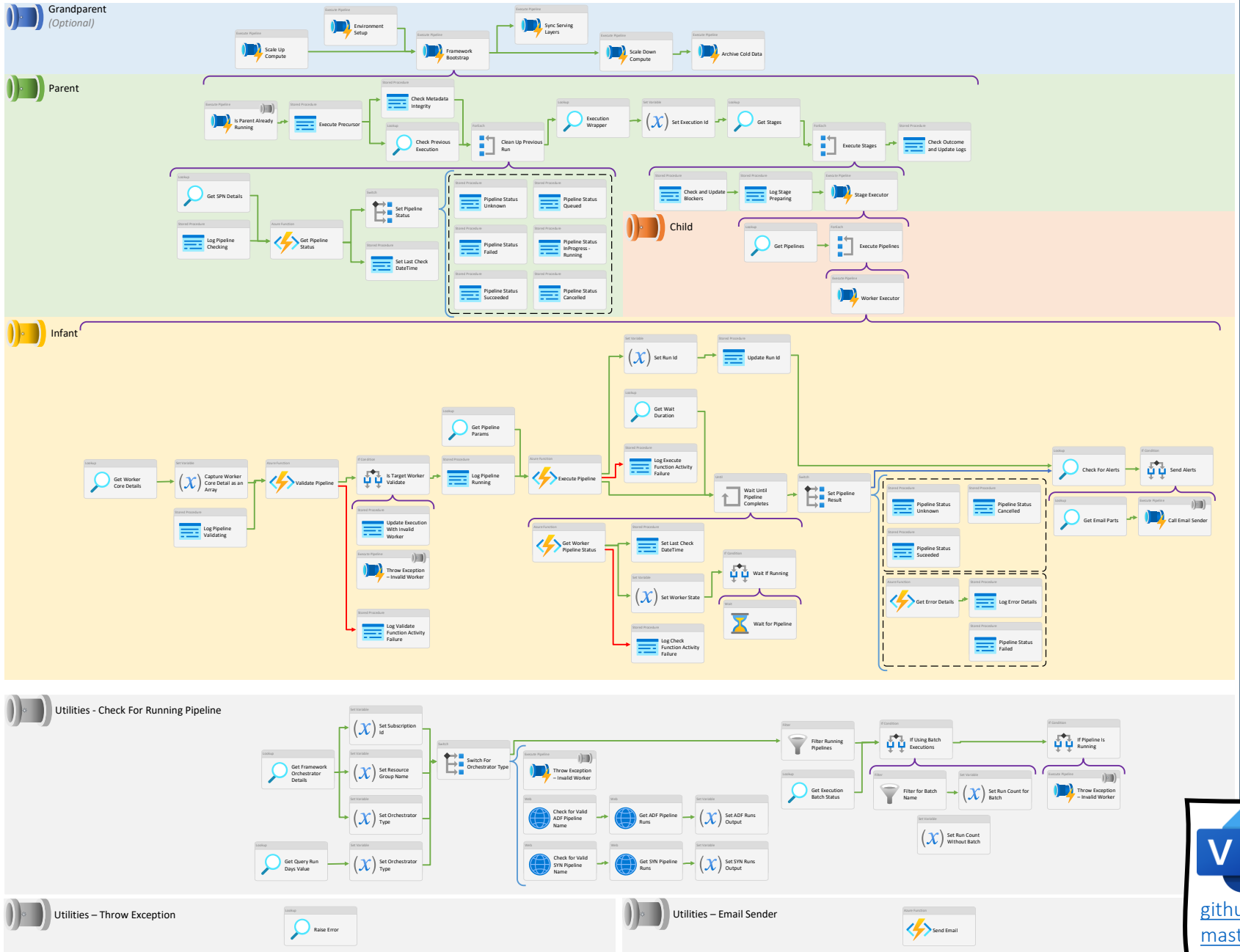
Role: Worker validator, executor, monitor and reporting of the outcome for the single worker pipeline.

- Worker

Role: Anything specific to the process needing to be performed.

Processing Framework - Activity Chain

Orchestration Framework



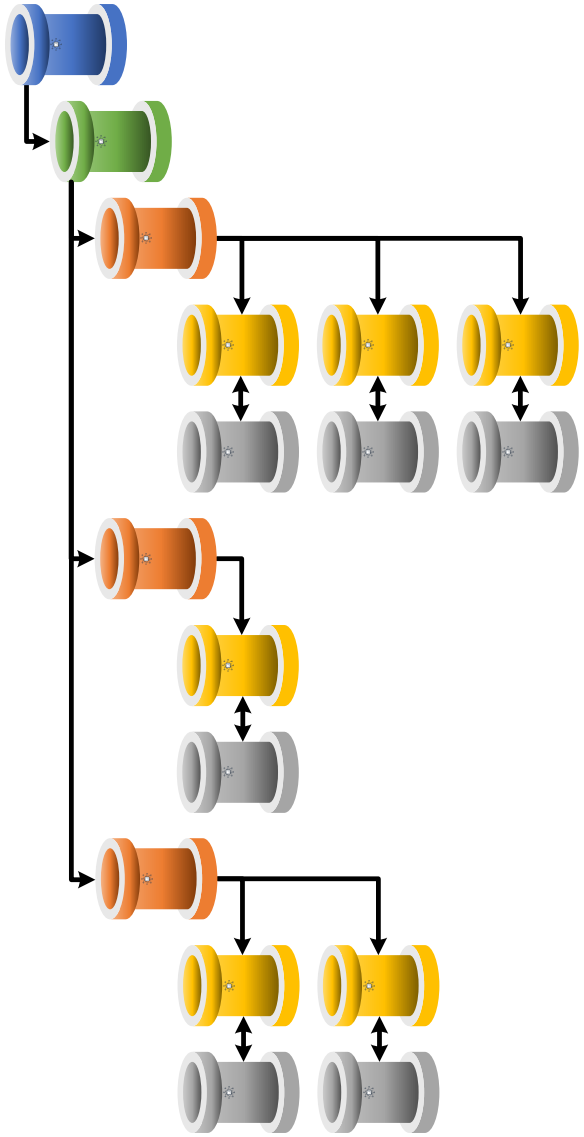
Worker Pipelines

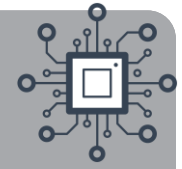
- Worker 1 - Extract
- Worker 2 - Clean
- Worker 3 - Transform
- Worker 4 - Load
- Worker 5 - Serve
- Worker n - .....

Go to Visio file in GitHub:

github.com/mrpaulandrew/procfwk/blob/master/Images


DEMO





Thank you for listening…

Paul Andrew

Blog: mrpaulandrew.com
YouTube: c/mrpaulandrew
Email: paul@mrpaulandrew.com

Twitter: @mrpaulandrew
LinkedIn: In/mrpaulandrew

GitHub: github.com/mrpaulandrew