Beyond the distributed monolith

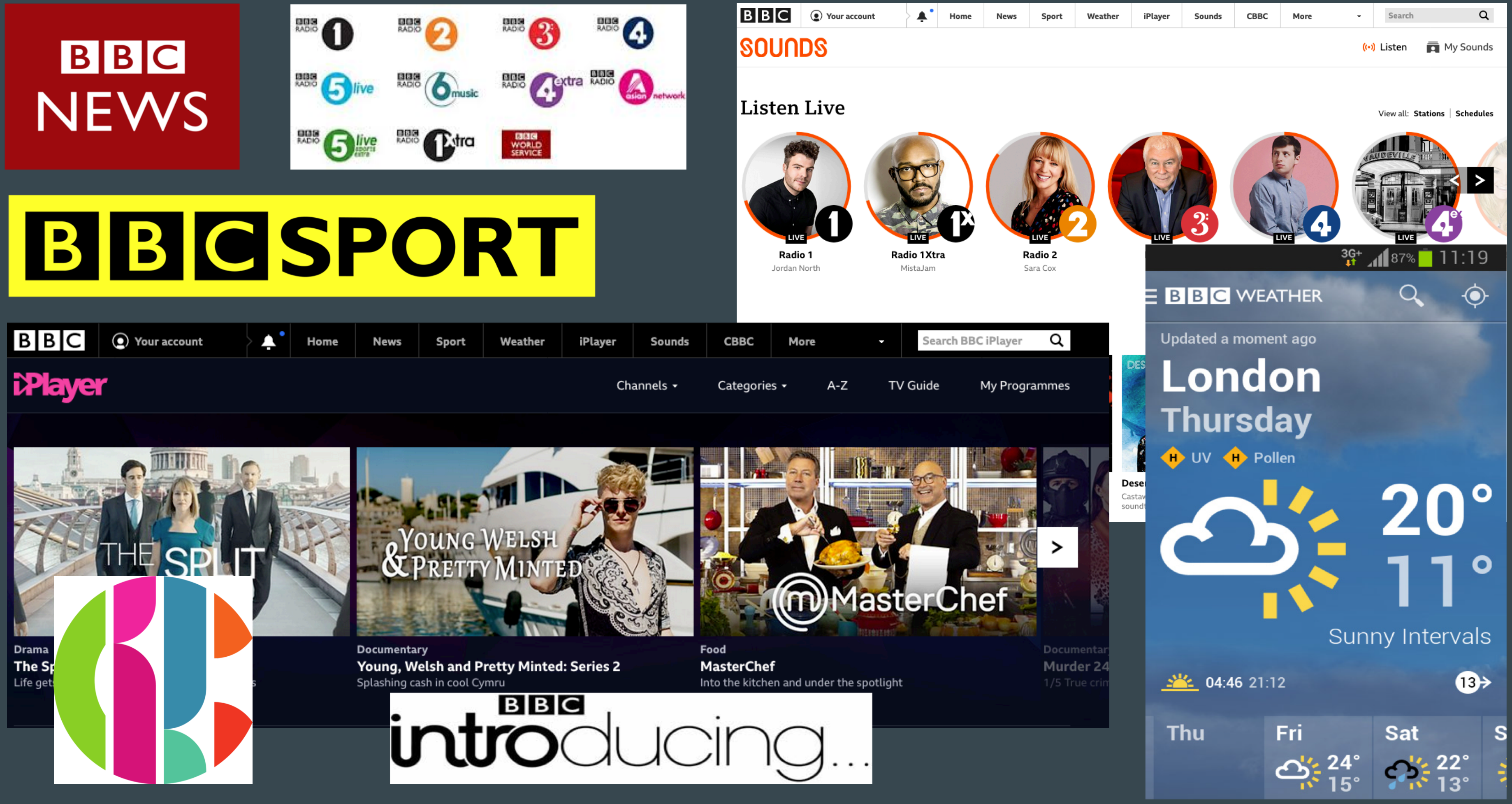
•••

Blanca Garcia Gil

Principal Systems Engineer, BBC

@blanquish

Table of contents


BBC NEWS
BBC SPORT
BBC Your account Home News Sport Weather iPlayer Sounds CBBC More Search
SOUNDS
Listen My Sounds
Listen Live
View all: Stations Schedules
Radio 1
Jordan North
Radio 1Xtra
MistaJam
Radio 2
Sara Cox
BBC WEATHER
Updated a moment ago
London
Thursday
UV Pollen
20°
11°
Sunny Intervals
04:46 21:12
Thu Fri Sat
24° 15°
22° 13°
BBC Your account Home News Sport Weather iPlayer Sounds CBBC More Search BBC iPlayer
iPlayer
Channels Categories A-Z TV Guide My Programmes
Drama
Documentary
Young, Welsh and Pretty Minted: Series 2
Splashing cash in cool Cymru
Food
MasterChef
Into the kitchen and under the spotlight
BBC introducing...


A more
personal
BBC


Why?

Privacy promise

A more *personal* BBC for you

A better BBC for *everyone*

https://www.bbc.co.uk/usingthebbc/privacy-promise

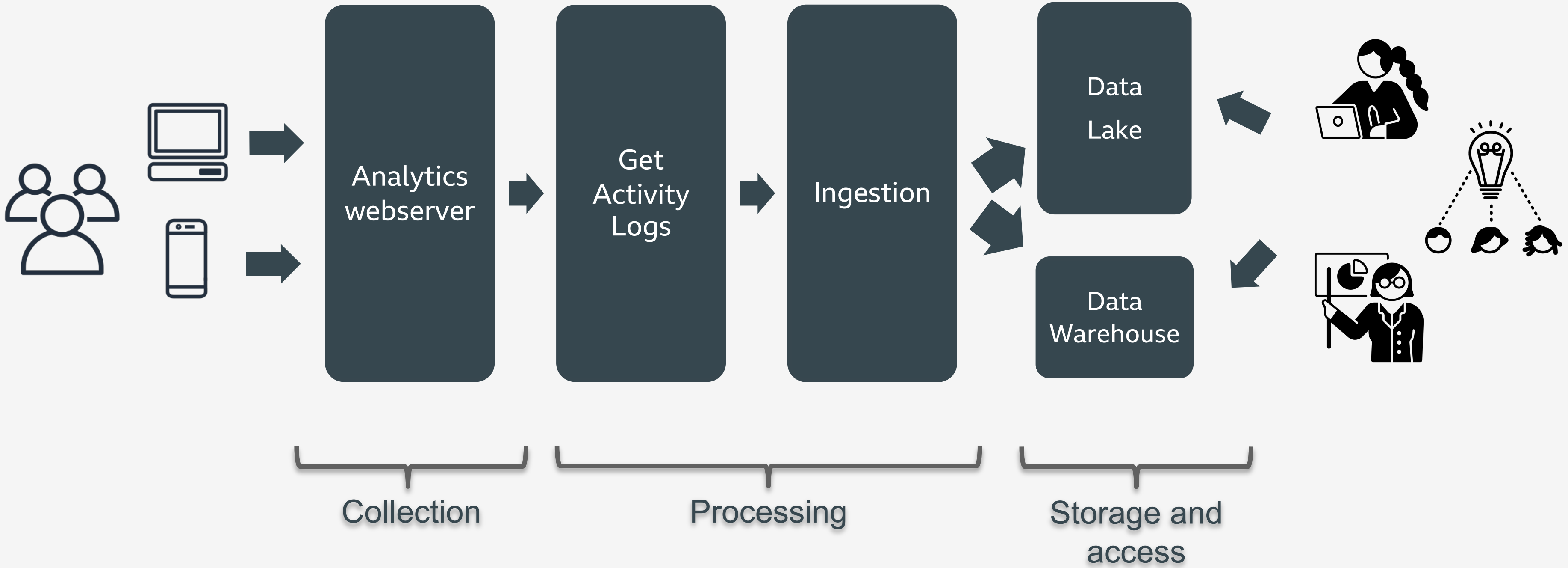
The acceptable face of personalisation

Dr Who

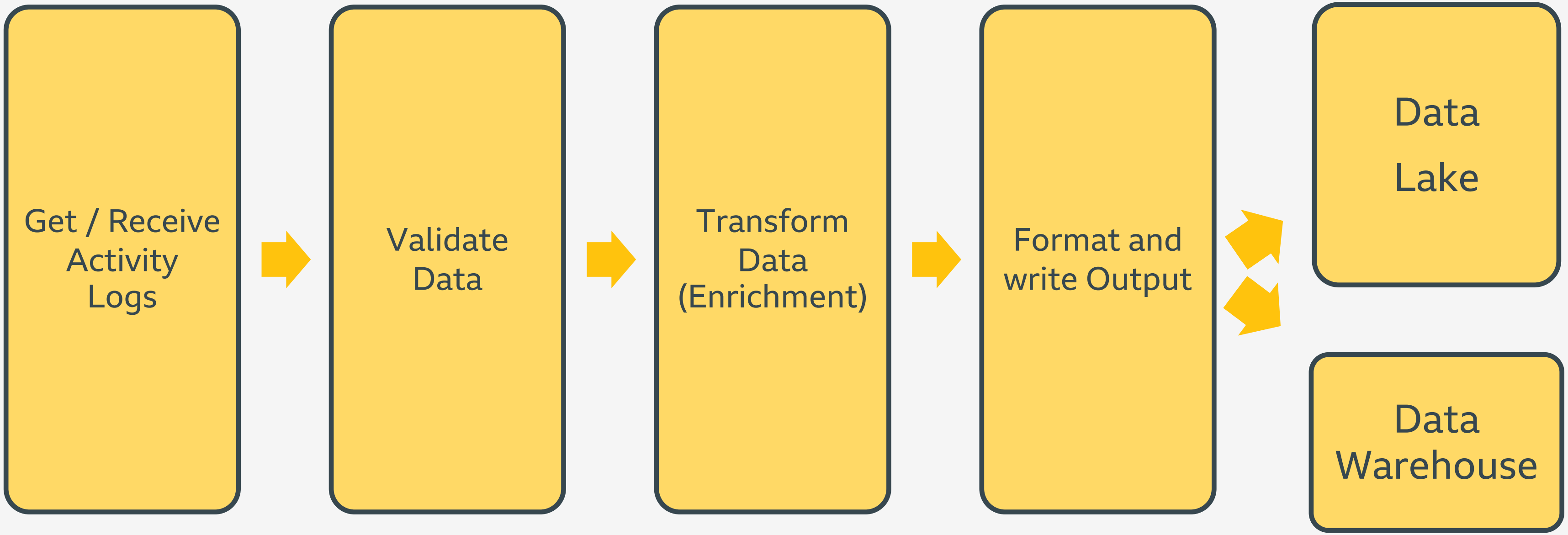
Daleks

2. How does analytics processing work?

Typical data analytics end to end pipeline

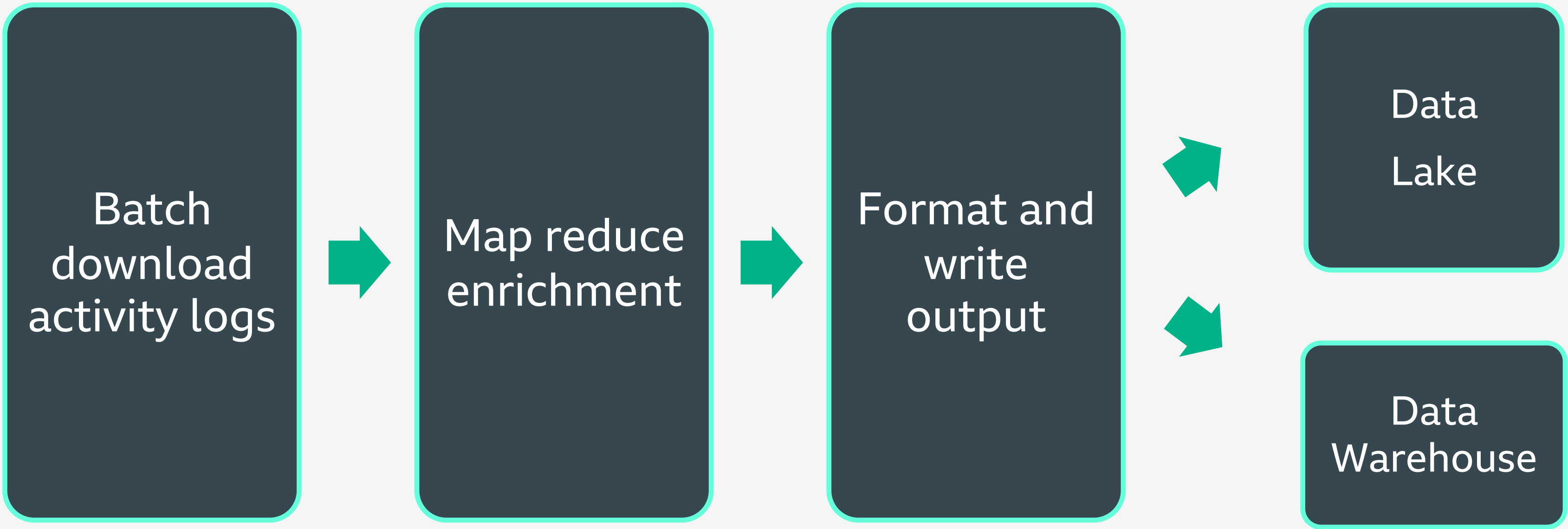


Typical data analytics ingestion flow

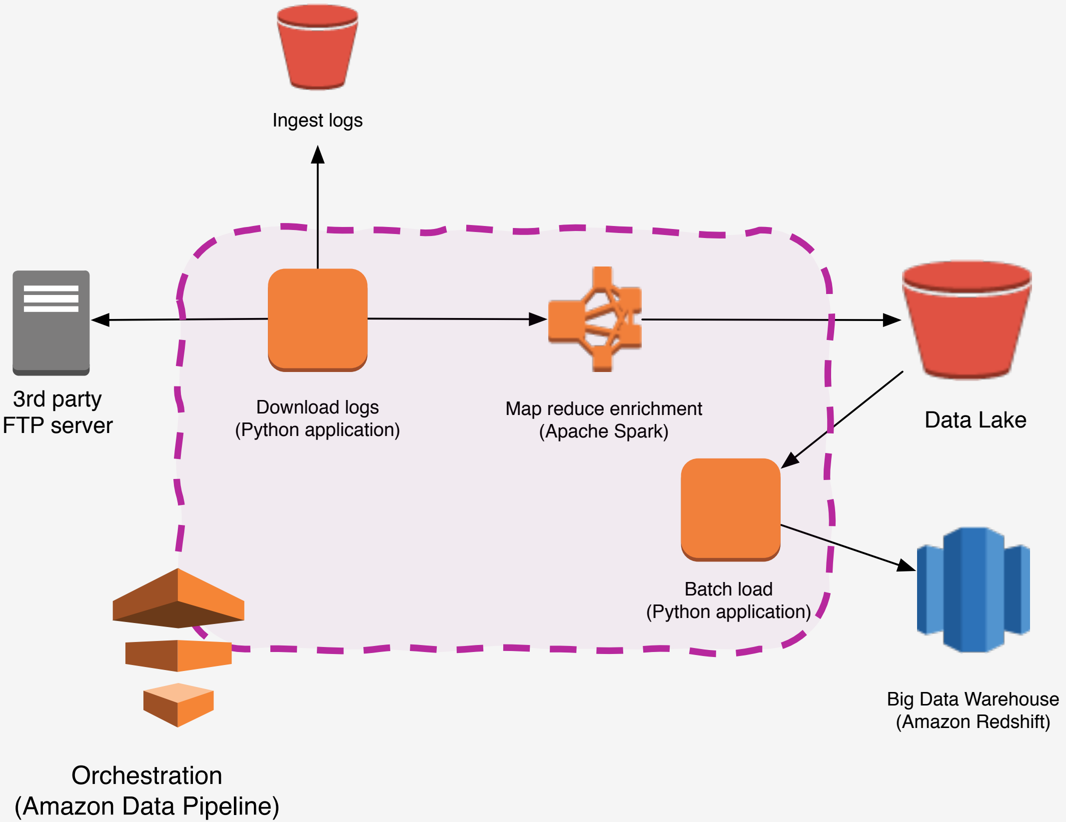


3. The distributed monolith and lessons learnt

How our data analytics pipeline architecture was



How our data analytics pipeline architecture was



Microservices
+
Big Data

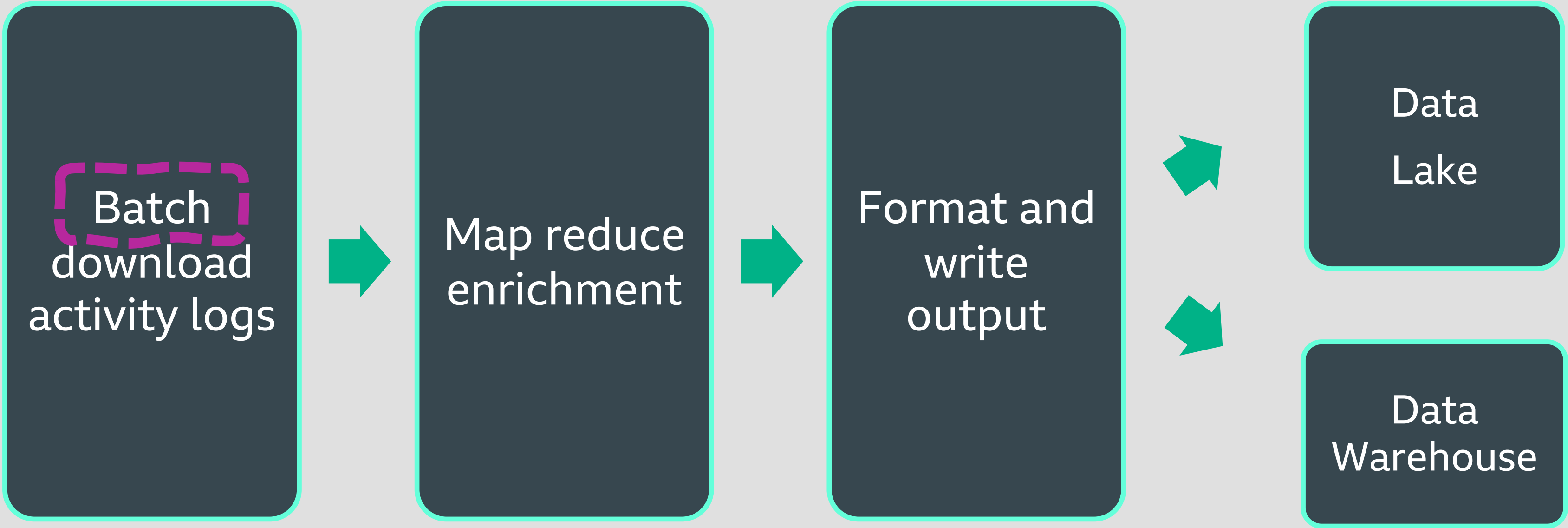
Highest scale data pipeline within our team

Billions of messages per day

Peta byte scale data lake

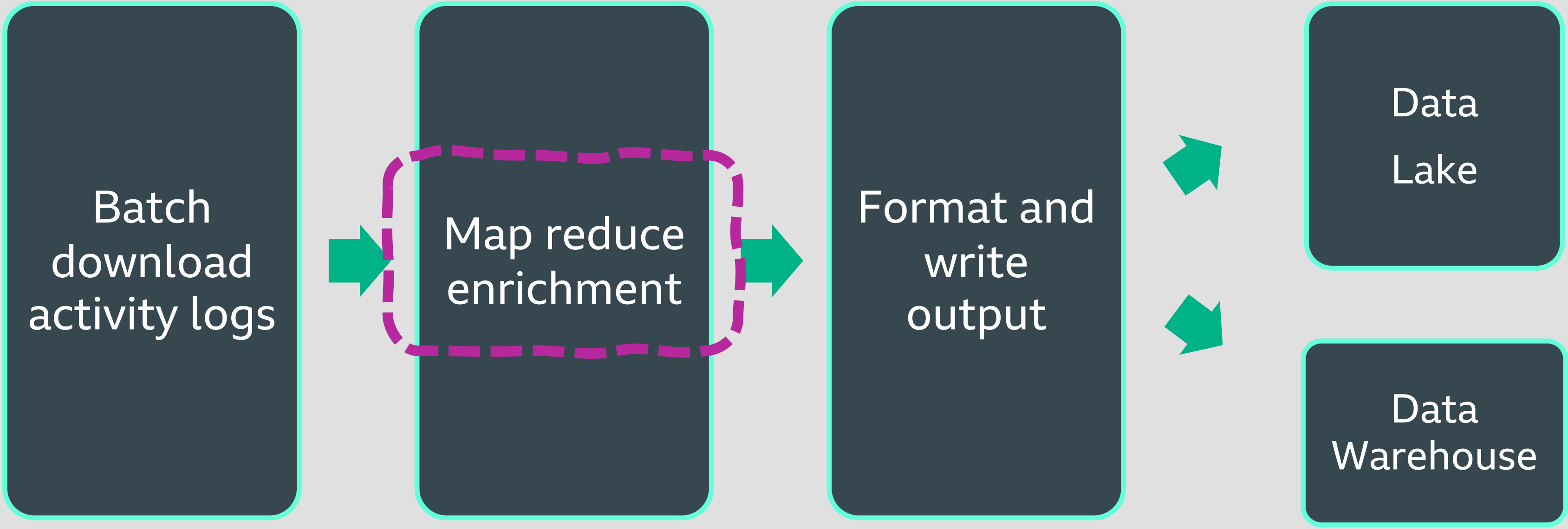
Lessons learnt

Lesson #1: batch processing



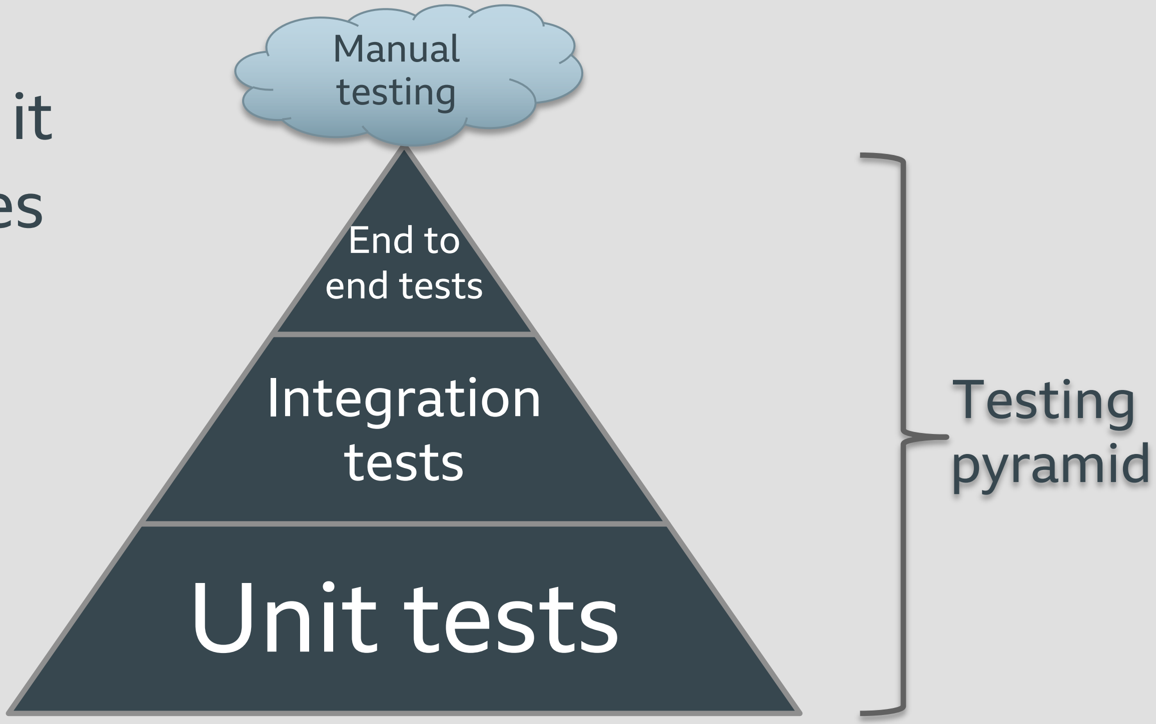

Finding needles

Lesson #2: validating input and testing

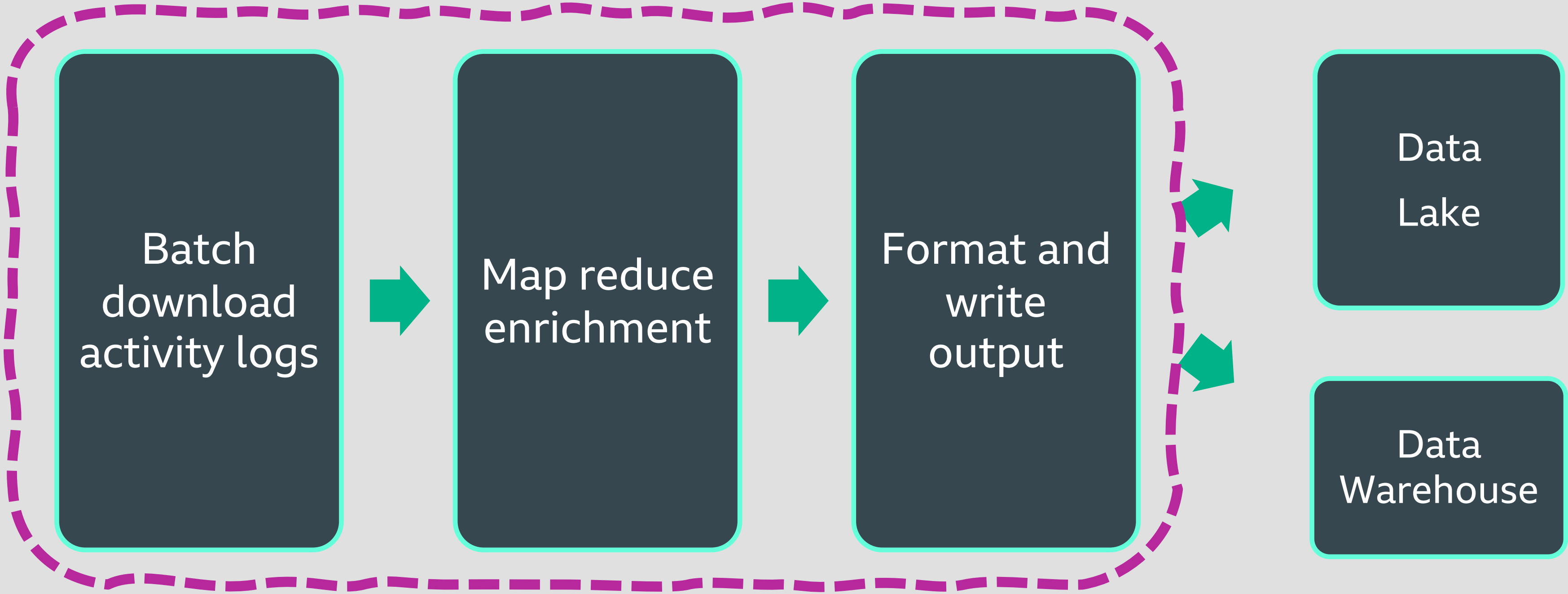


Testing

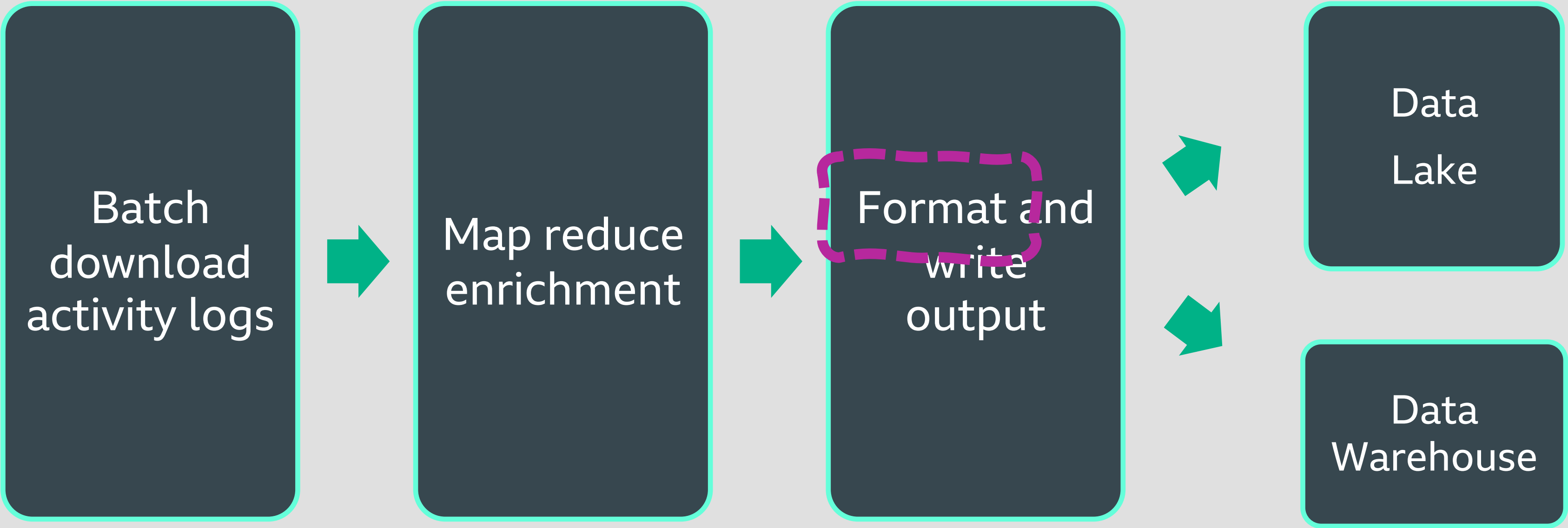
Very few tests made it hard to make changes to the code with confidence



Lesson #3: tight coupling



Lesson #3: tight coupling



Lesson #4: Monitoring

Is every alert worth your team being interrupted?

Lesson #5: understanding our traffic patterns

Big News Days

Data volume is ever increasing

Lesson #6: Cost effective solution

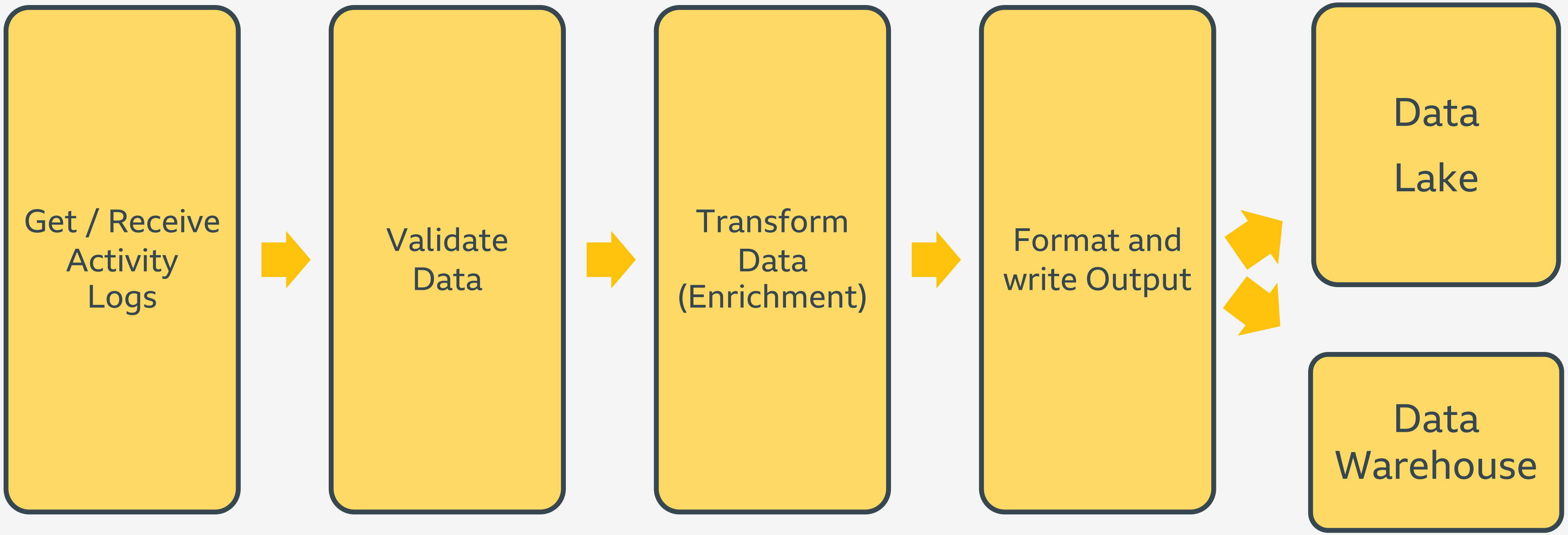
Lesson #7: Getting feedback from our internal users

4. Designing a new analytics ingestion architecture

“Small, autonomous services that work together, modelled around a business domain”

Definition of microservices from “Building microservices” book by Sam Newman

Revisiting the typical data analytics ingestion flow



Smaller problems to solve

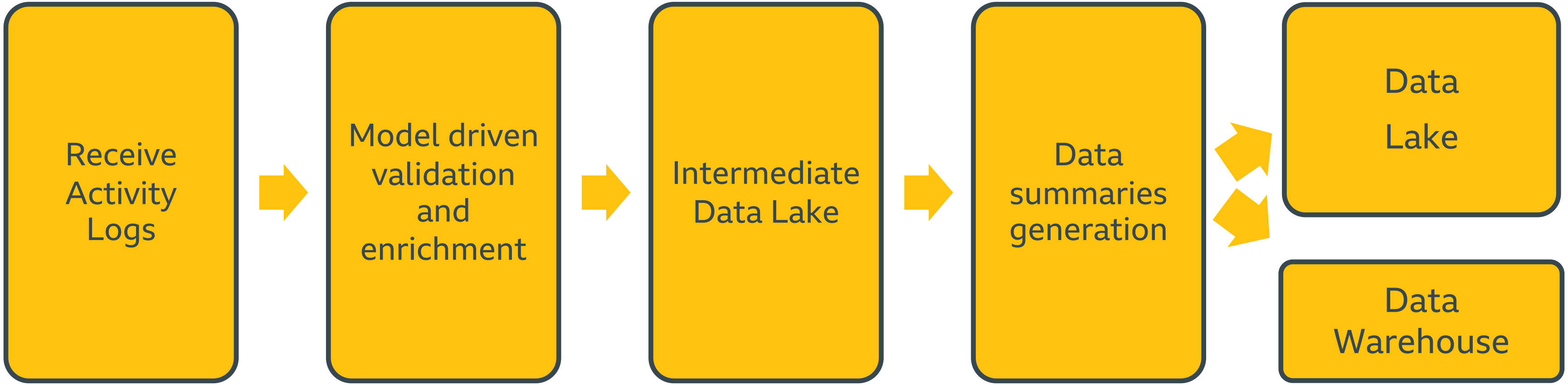
Squads to focus on each bounded context:

1. Receiving activity logs and keeping track of the data received
2. Model driven validation, enrichment and transformation to columnar format
3. Data aggregations (summaries) based on users' needs to make easier getting value from the data

Different approaches to testing

- Auto generated data (load testing)
- Production-like data (quality)

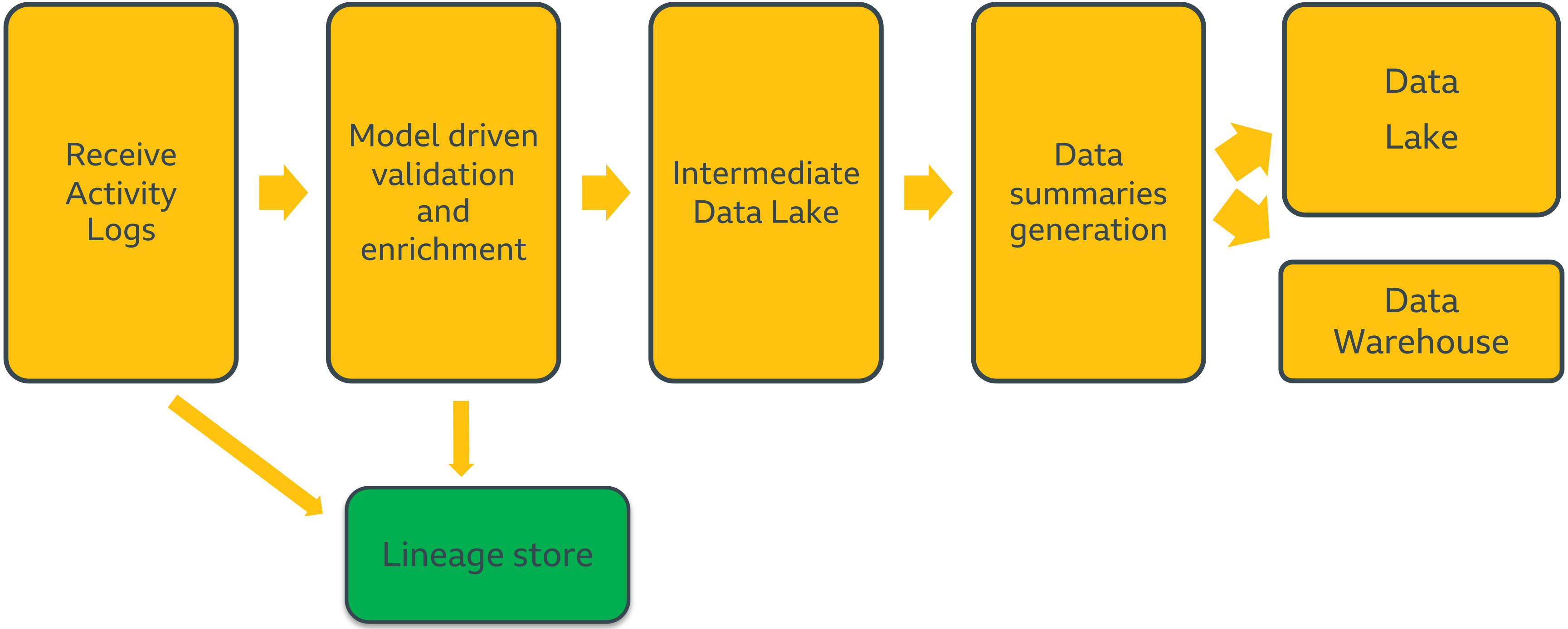
New pipeline architecture



Keeping track of the data

Introduced a lineage store

New pipeline architecture

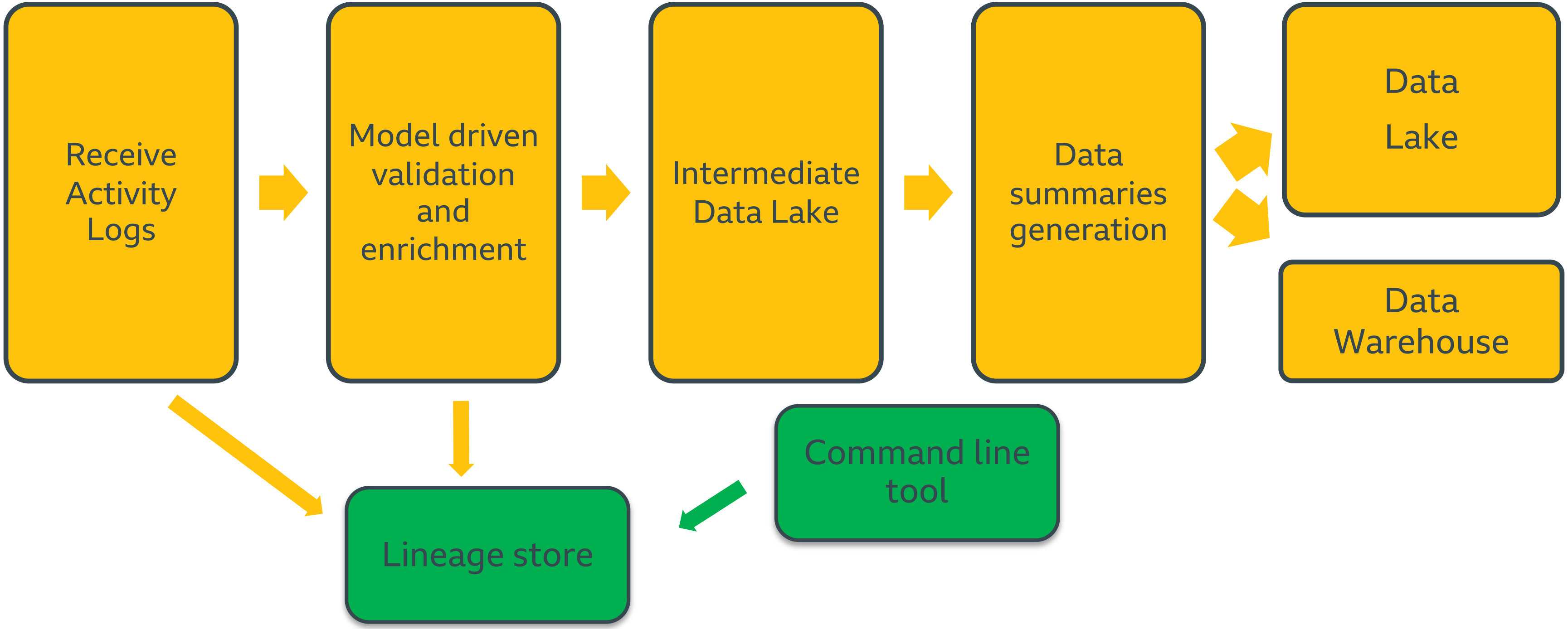


Operating the new architecture

Minimize the knowledge needed to be able to share the operational load within the team

💗 command line tools

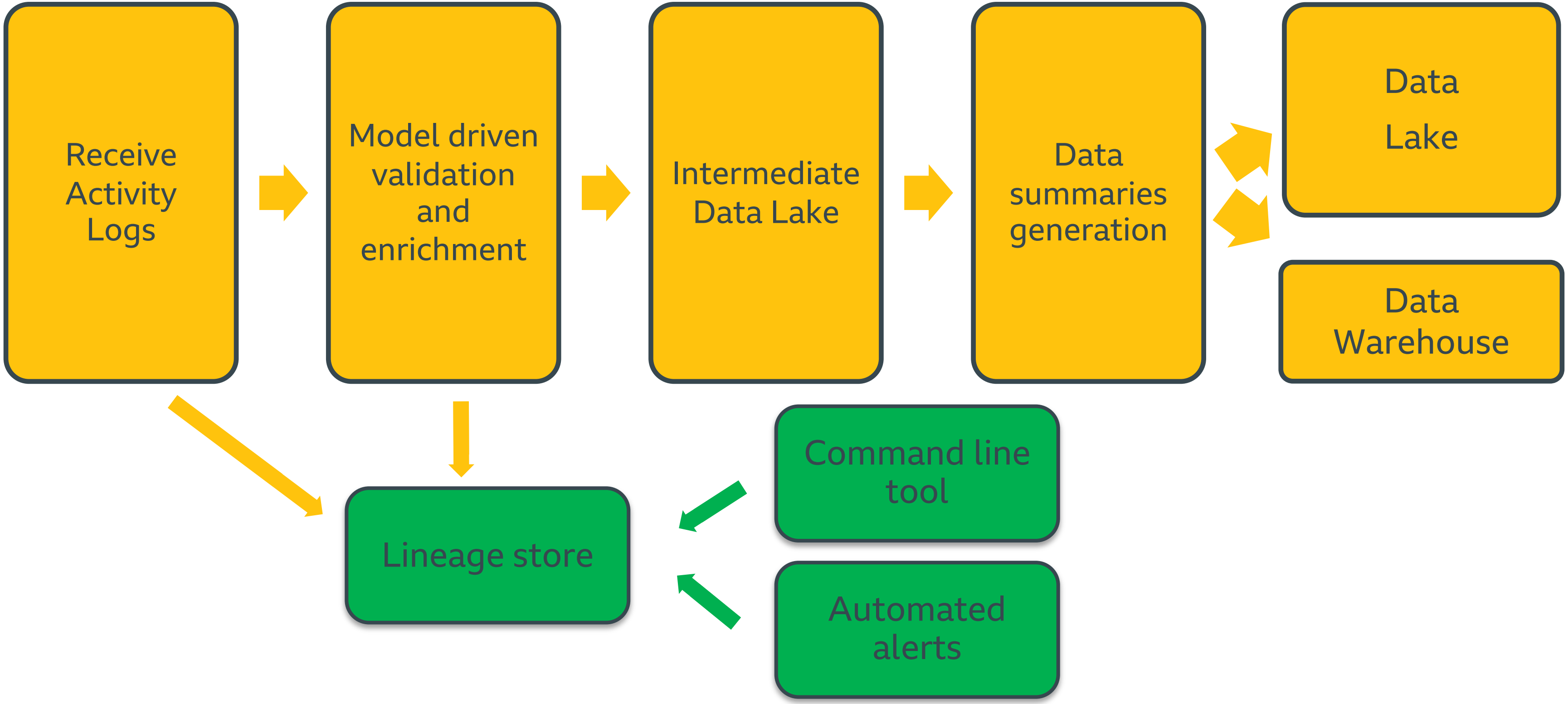
New pipeline architecture



Operating the new architecture

Alerting on missing data

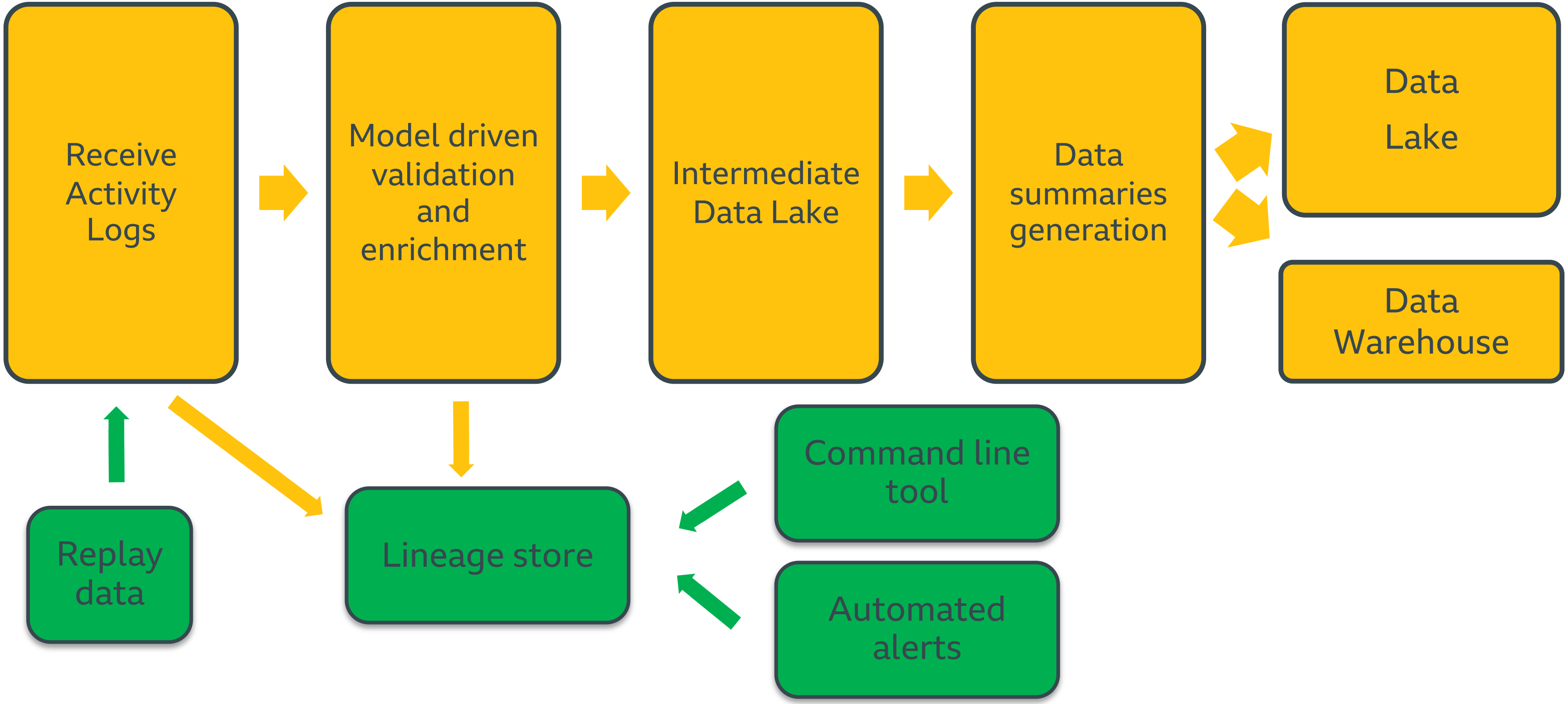
New pipeline architecture



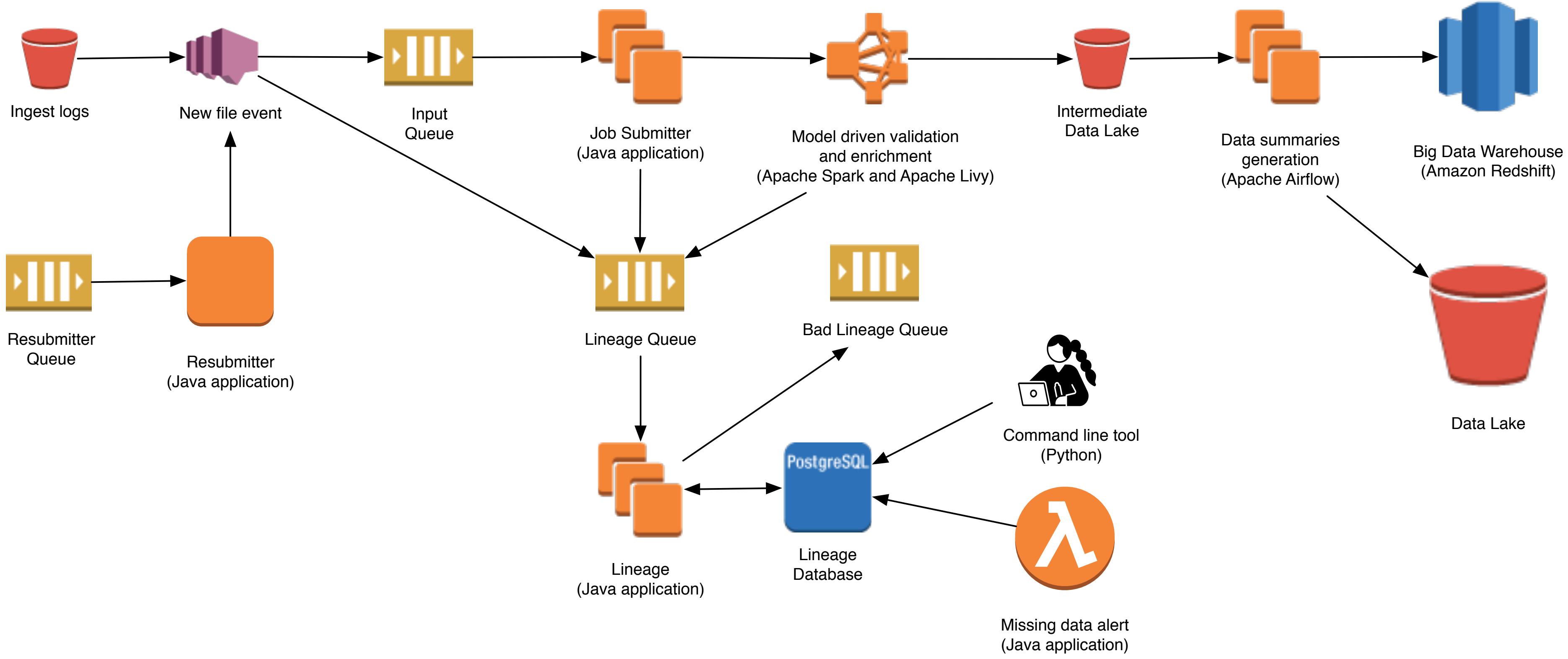
Operating the new architecture

Replaying data without stopping everything else

New pipeline architecture



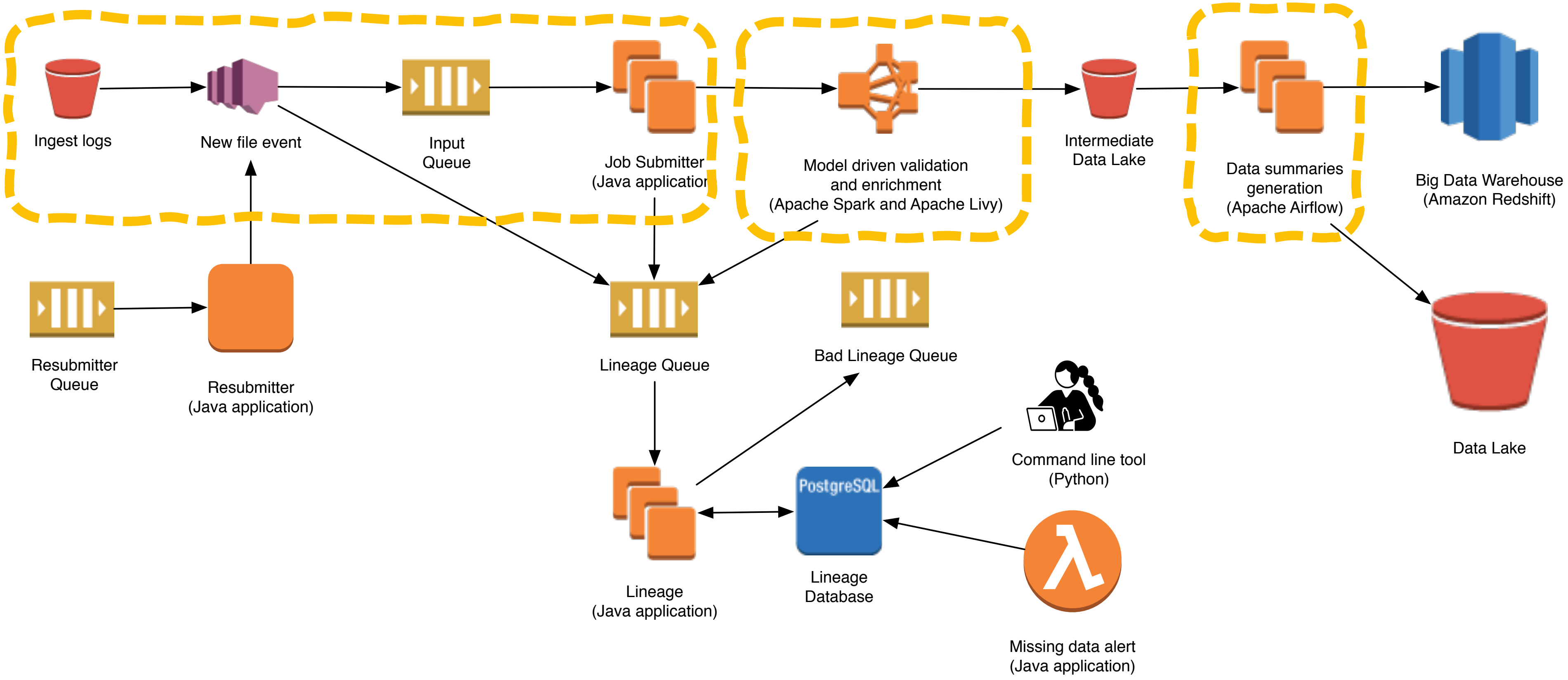
New pipeline architecture



Benefit #1

Enabling team to choose fit for purpose tech and architecture

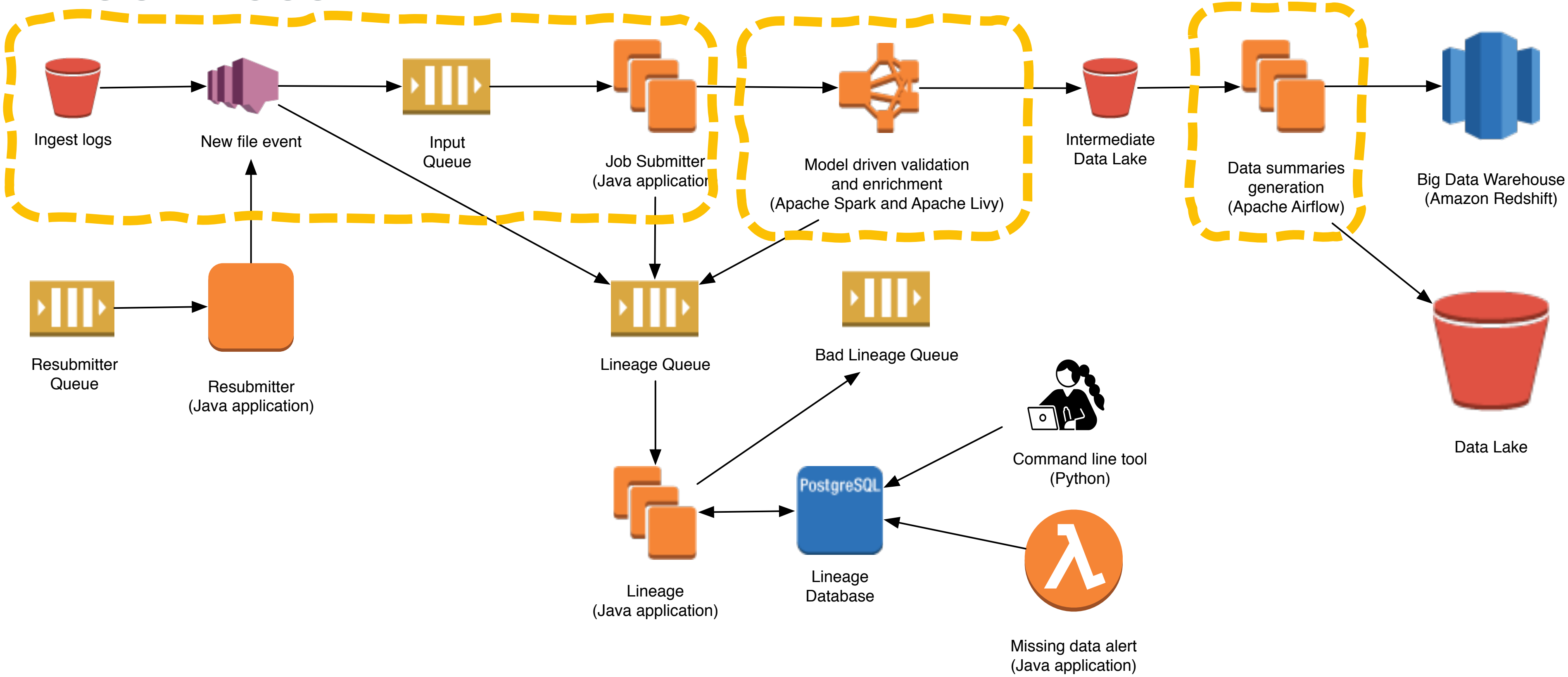
Benefit #1: choose fit for purpose tech



Benefit #2

Make it easier to change or replace microservices

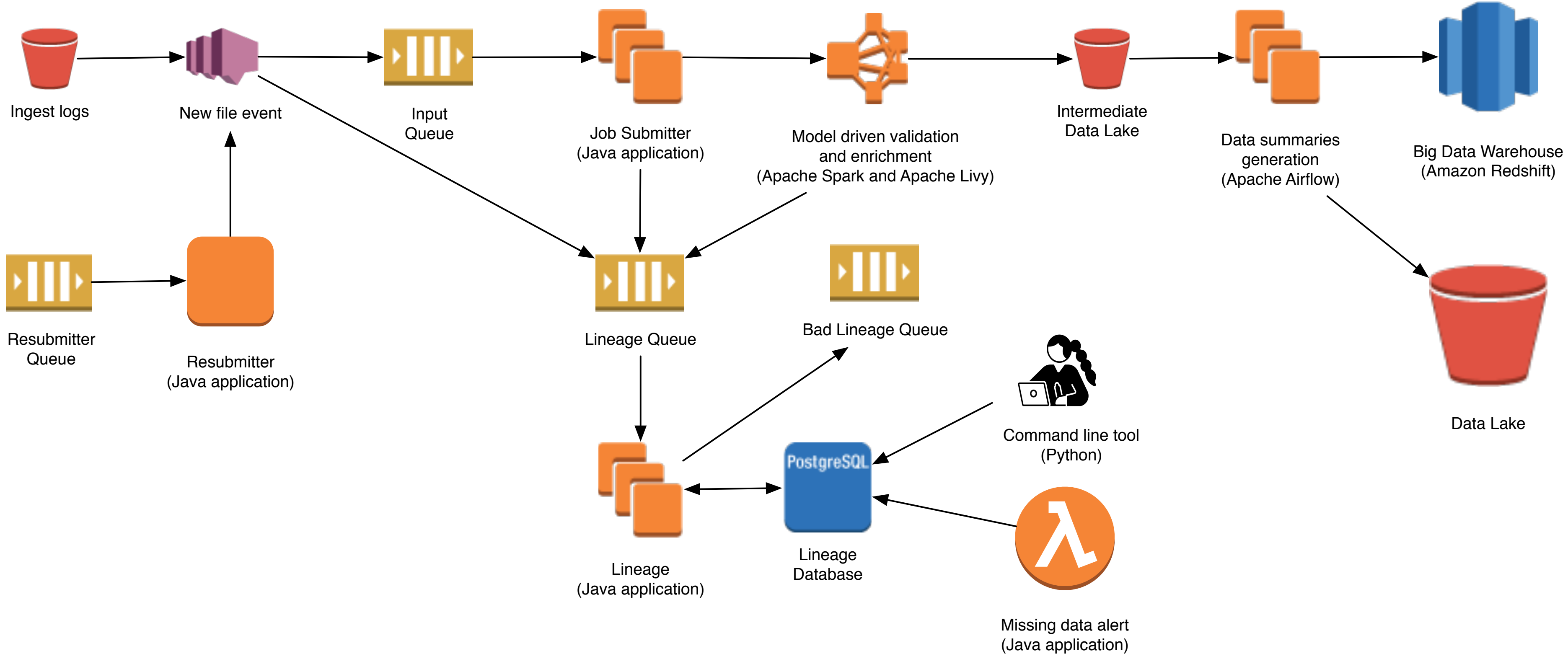
Benefit #2: make it easier to change or replace services



Benefit #3

Isolate failure

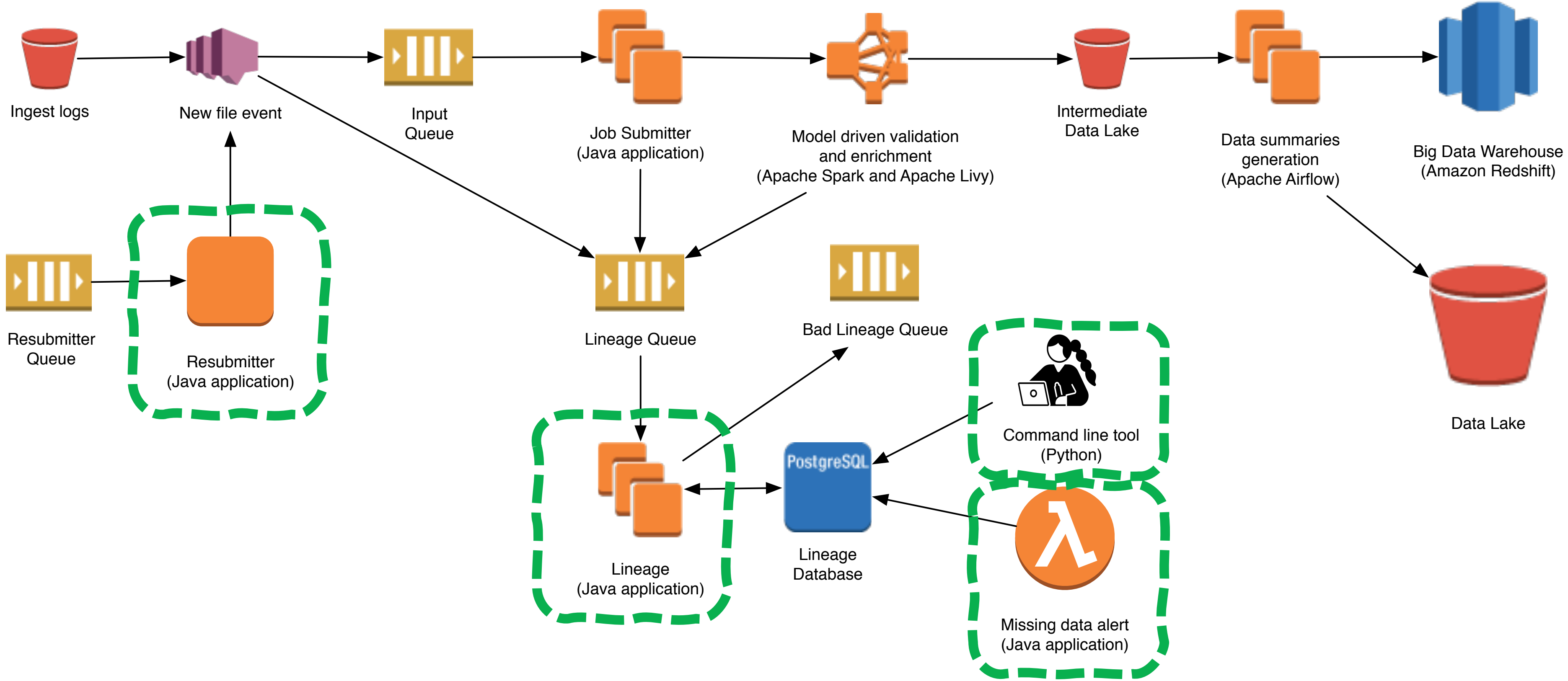
Benefit #3: isolate failure



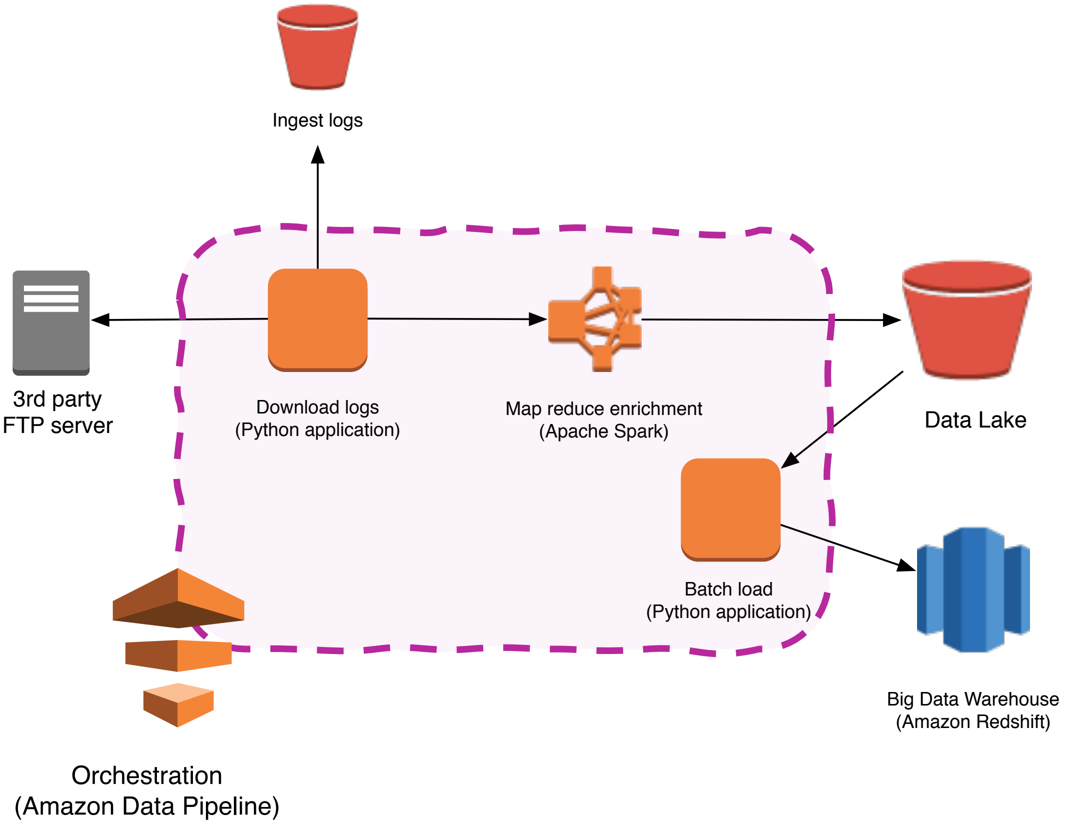
Benefit #4

Organic system growth as we operate it and learn

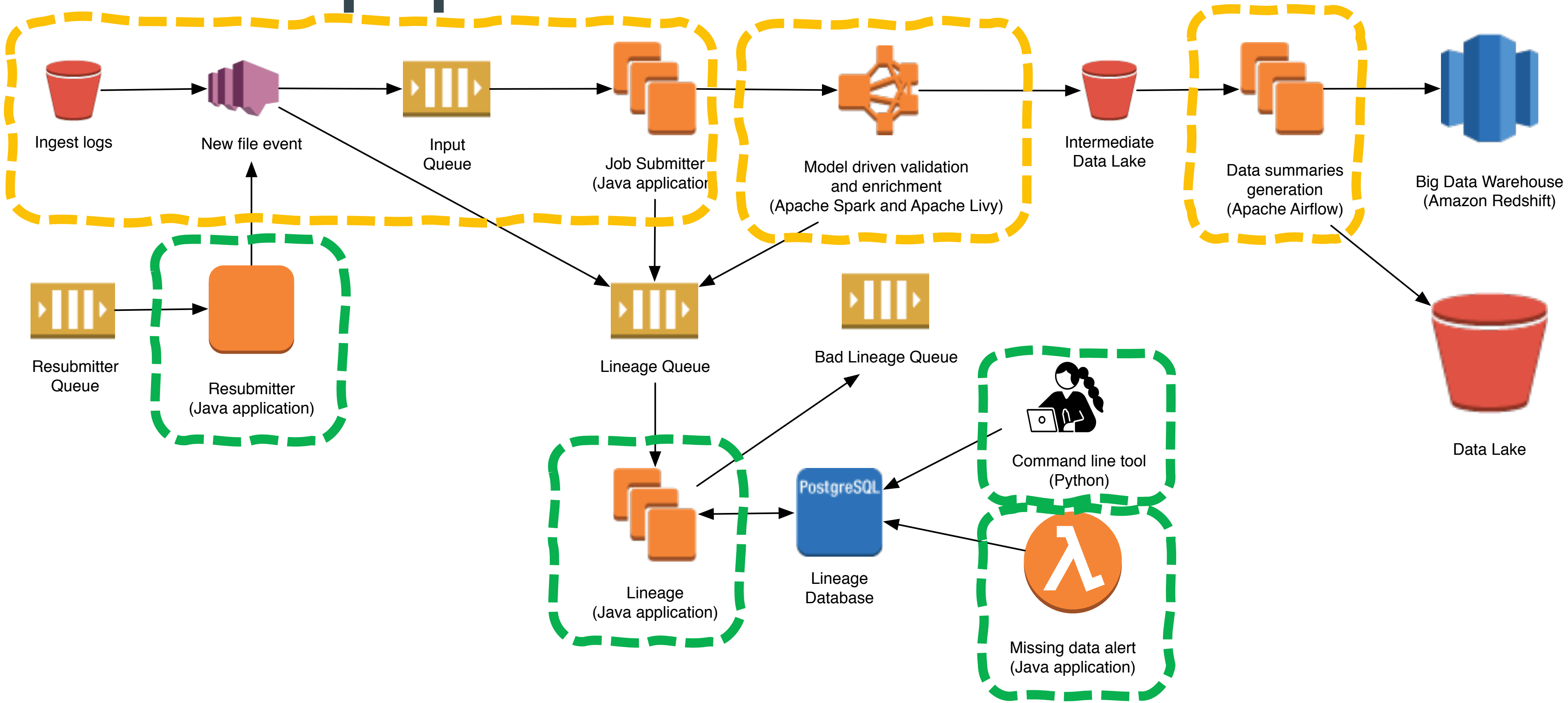
Benefit #4: organic system growth



How our data analytics pipeline architecture was



New pipeline architecture



Main takeaways

Design with change in mind, you can’t predict how your traffic will evolve over time

Make sure everyone in the team can triage live issues

Choosing languages and tools which the whole team owns

5. The future of the Data Platform

Challenges as we look into the future

1. How will this architecture evolve as our data load increases?

Challenges as we look into the future

2. What are the future usage requirements for our data platform?

Challenges as we look into the future

3. How can we make it easier for our users to self serve while keeping the data secure?


A data story
KILLING EVE

Thank you!

@blanquish