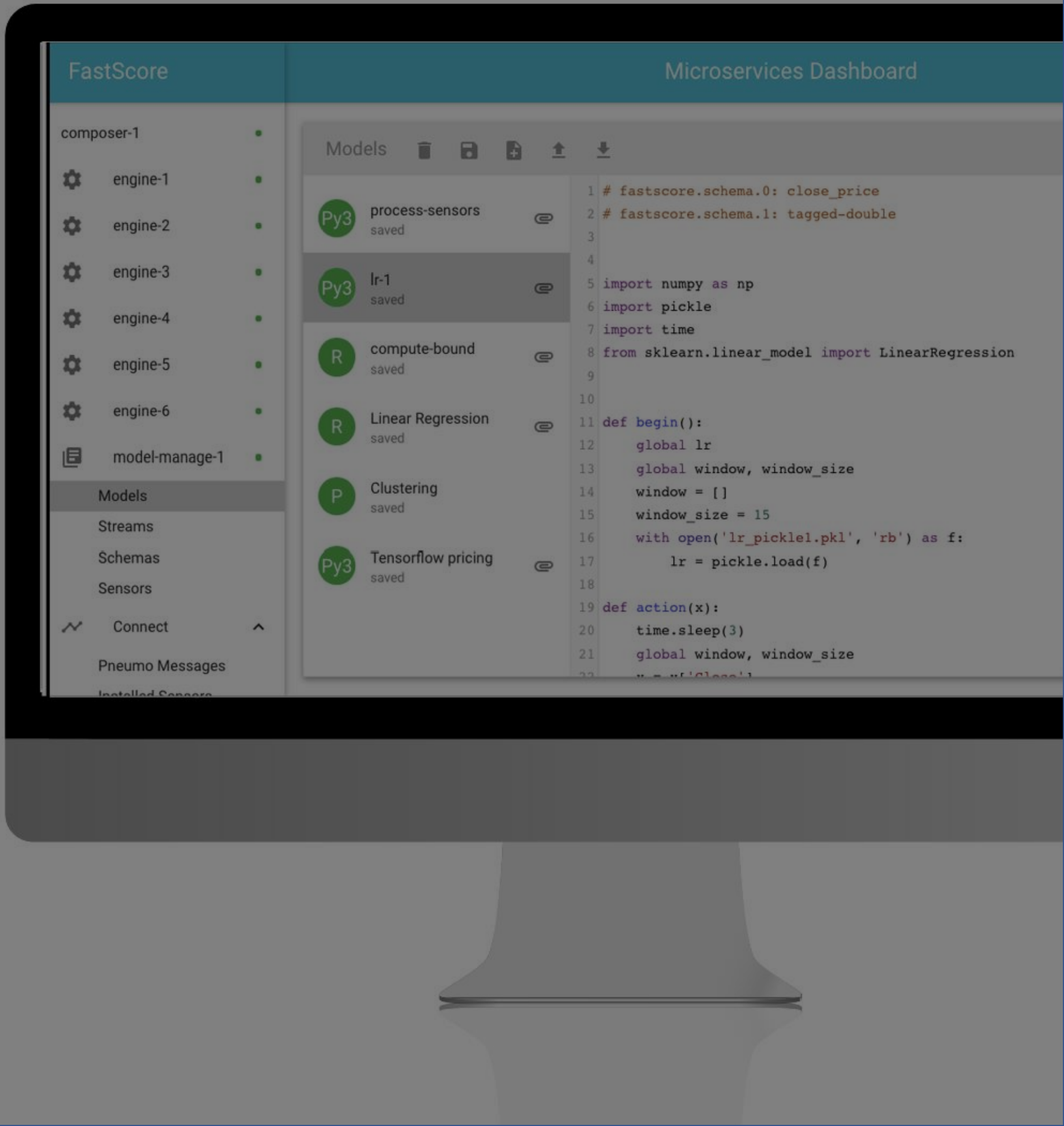
Achieving AI and ML Operational Excellence

Presented by Rehgan Avon, Product Manager



open data

Featured Speaker

About Rehgan Avon

- Studied Integrated Systems Engineering specializing in Data Analytics from Ohio State University
- Launched the Big Data and Analytics Association at OSU
- Founded and coordinates the Women in Analytics Conference in Columbus, Ohio
- Product Manager at Open Data Group

"We learn more by looking for the answer to a question and not finding it than we do from learning the answer itself." – Lloyd Alexander

"If I had an hour to solve a problem and my life depended on the solution, I would spend the first 55 minutes determining the proper question to ask for once I know the proper question, I could solve the problem in less than 5 minutes." – Albert Einstein

Market Trend & Drivers

2018 Predictions

- Applied Machine Learning and AI gain traction in a majority of Fortune 500 companies, across all lines of business
- Open Source adoption continues to slowly erode traditional monolithic platform dominance of enterprise analytics
- Select analytic workloads will be transitioned to cloud (or hybrid) infrastructure, creating cost and application opportunities
- Achieving Operational Excellence on deployed ML and AI models will become as important as creating new models
- Commoditization (e.g. cloud) is allowing the enterprise to benefit from unprecedented available compute and storage at historically low costs.

Gartner

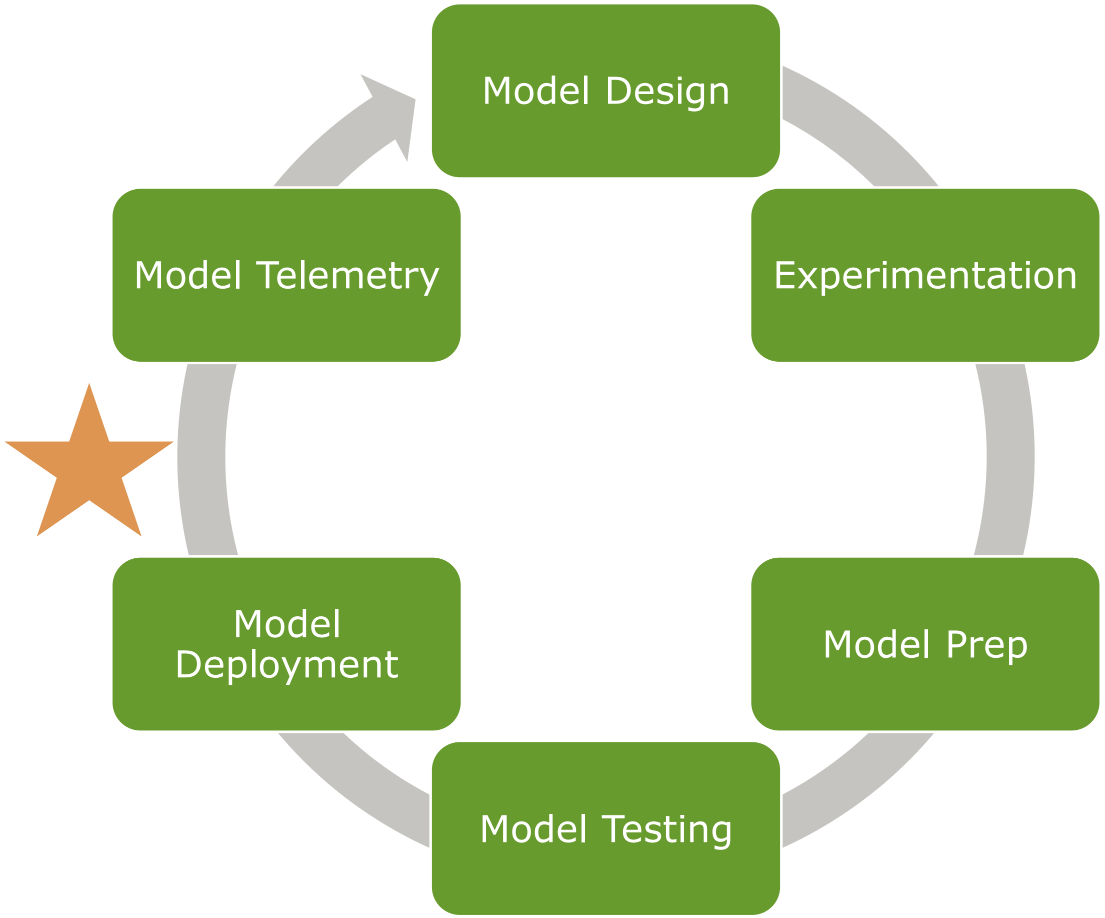
Market trends and insights

- *By 2020, more than 40% of data science tasks will be automated, resulting in increased productivity and broader usage by citizen data scientists.*[1]
- *The challenge now is to deploy and operationalize at scale.*[2]

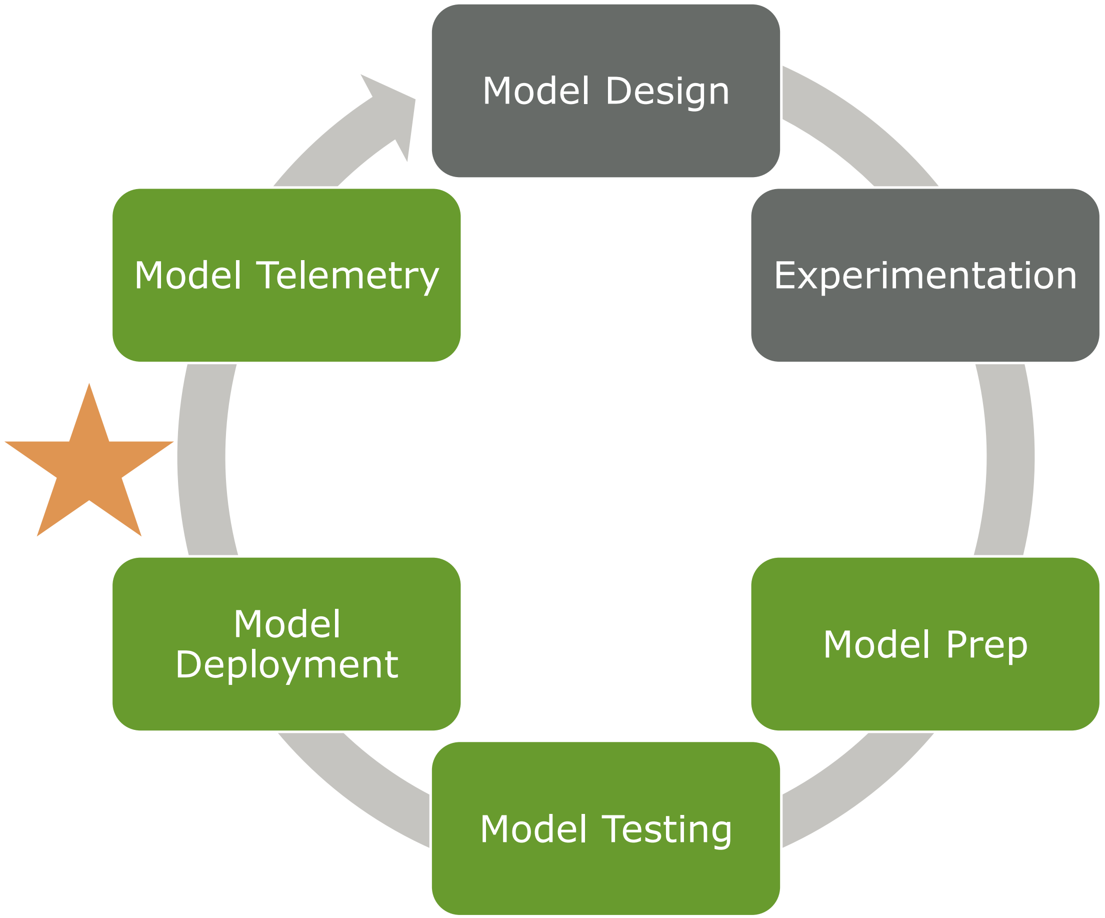
[1] *"Gartner Says More Than 40 Percent of Data Science Tasks Will Be Automated by 2020", Susan Moore et al January 2017*

[2] *"How to Operationalize Machine Learning and Data Science Projects", E Brethenoux et al July 2018*

Model Development Life Cycle



Model Development Life Cycle



Top Process Challenges

Volume of Changes

Diversity of Teams Involved

Rapid Iteration

What is the solution?

Goal: Build for high quality improvements, quickly

Top Process Challenges

Volume of Changes

Diversity of Teams Involved

Rapid Iteration

Goal: Build for high quality improvements, quickly

Top Process Challenges

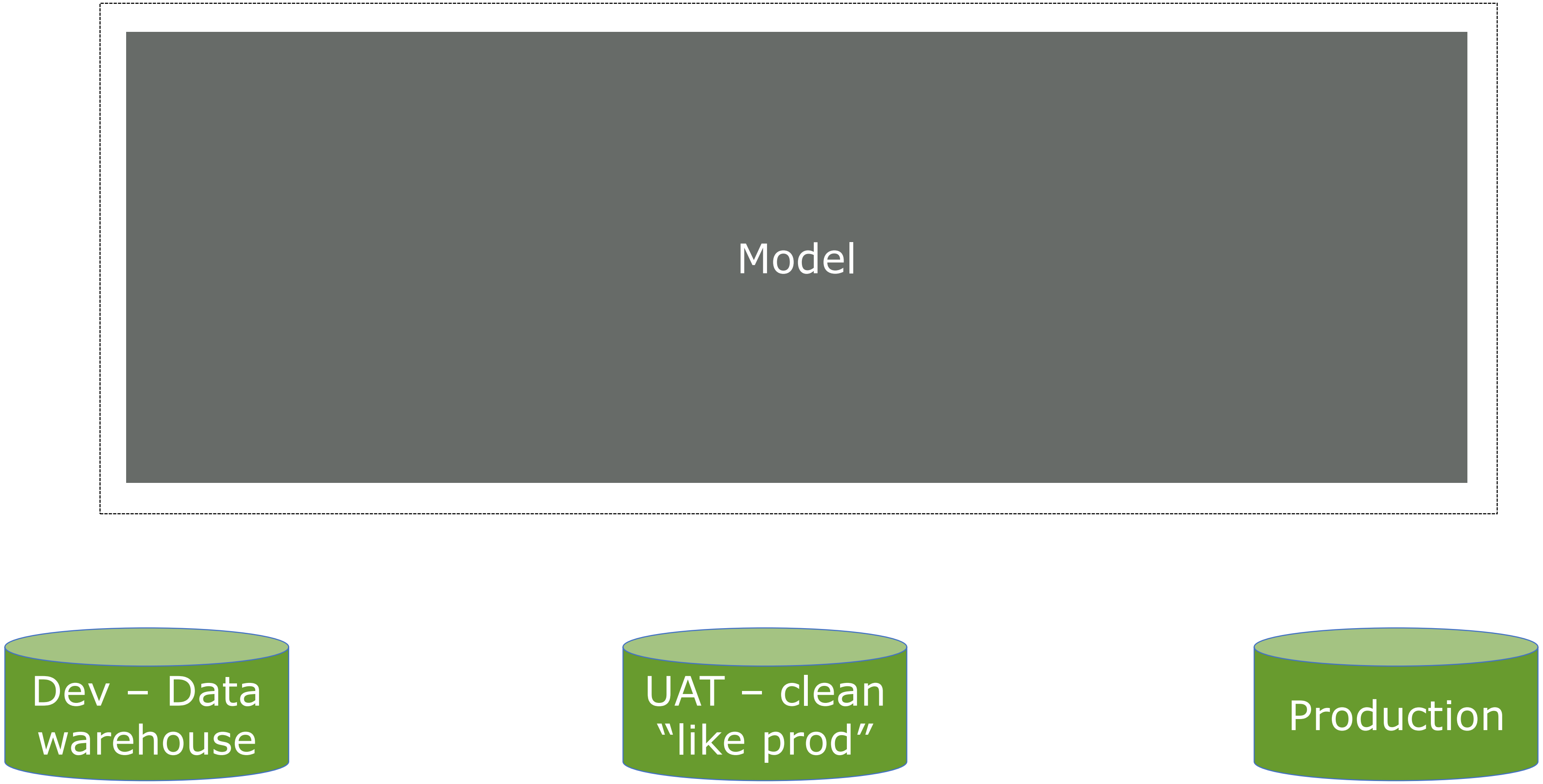
Volume of Changes

Data Locations
Infrastructure
Surrounding Systems
Security Requirements

Goal: Build for high quality improvements, quickly

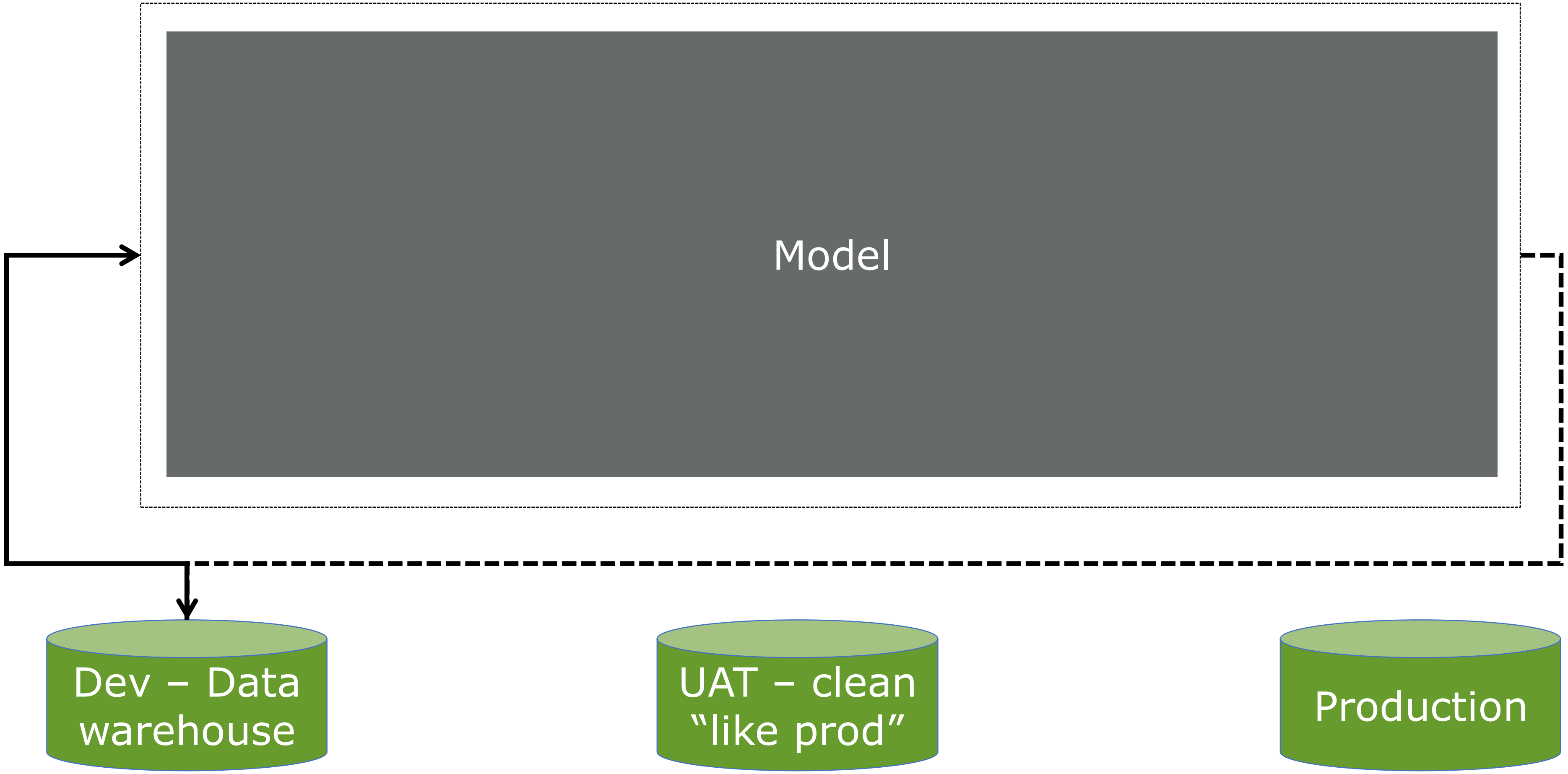
What Can Change to Deployment?

Data Locations



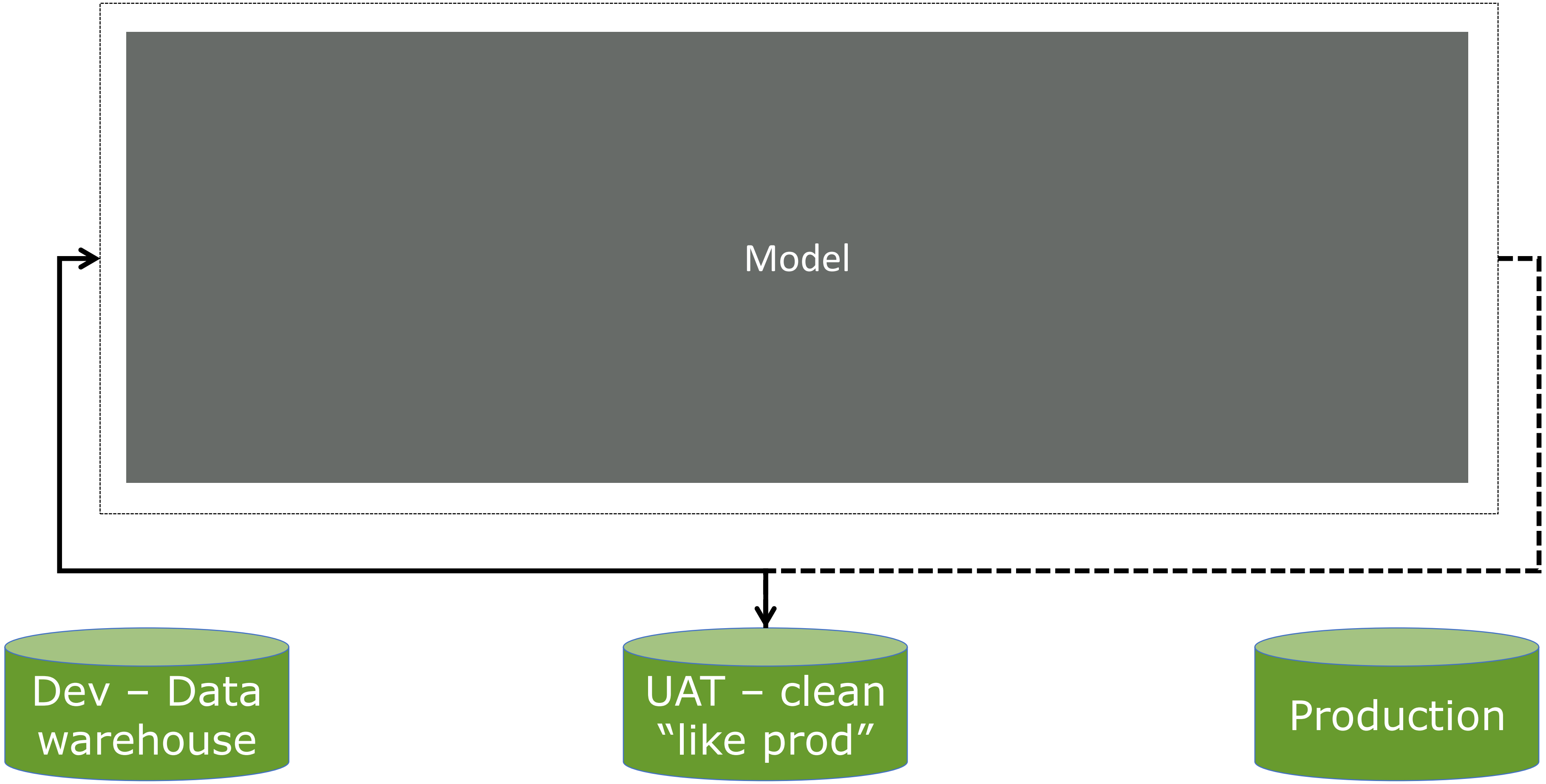
What Can Change to Deployment?

Data Locations - Dev



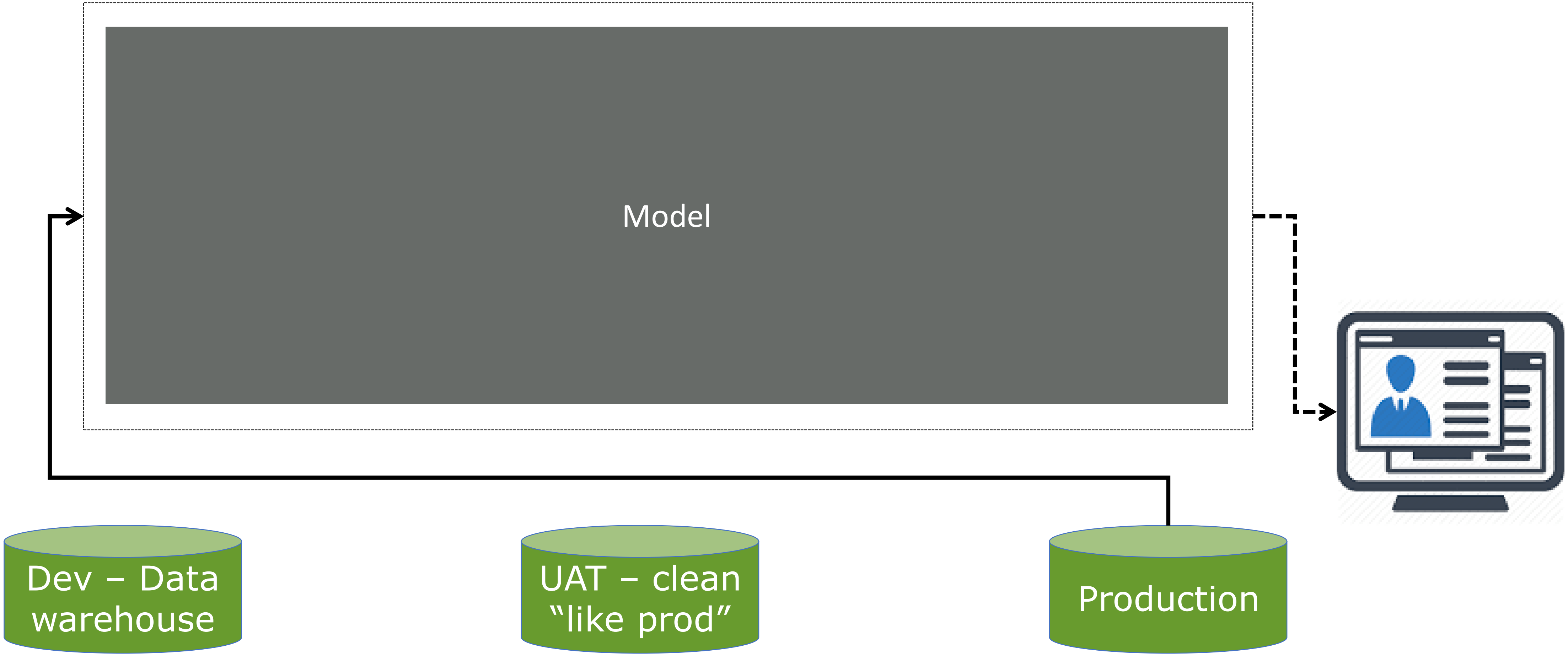
What Can Change to Deployment?

Data Locations - Test



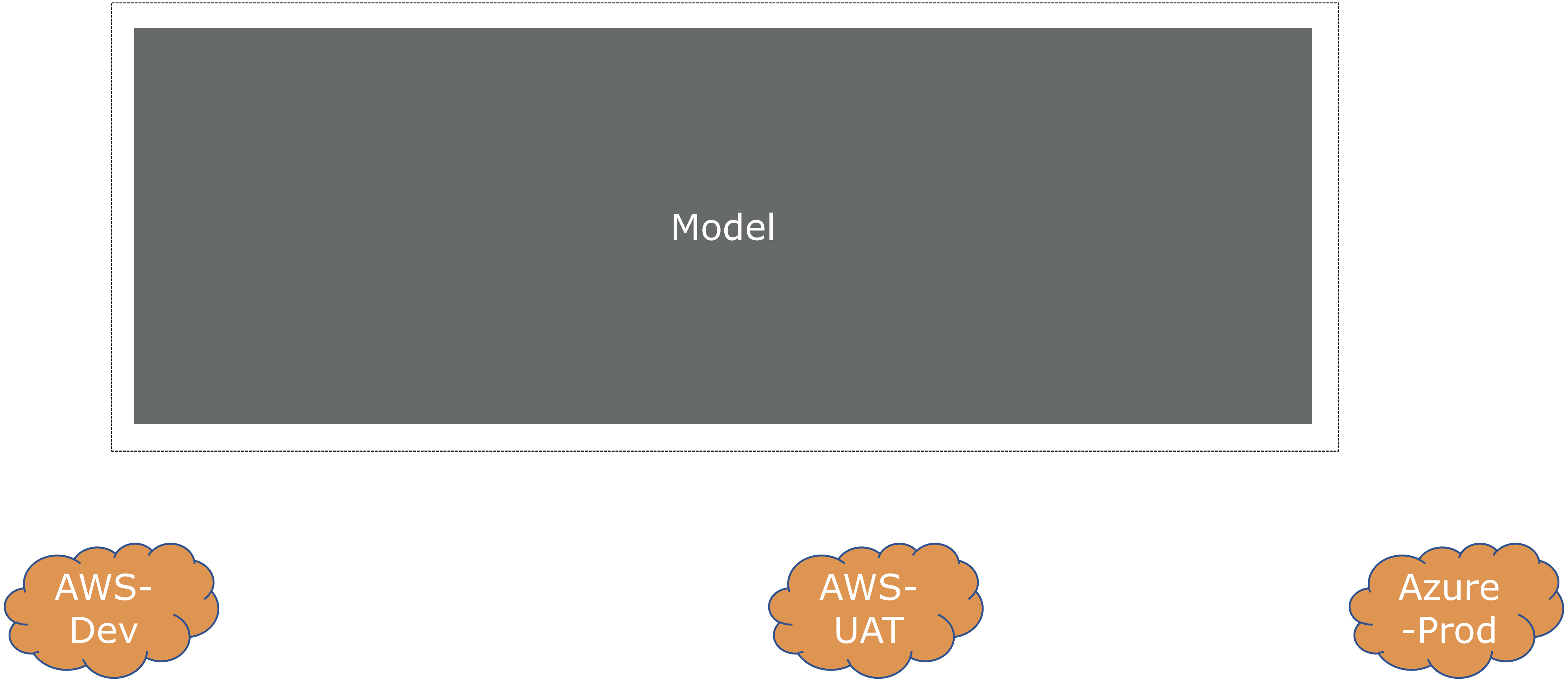
What Can Change to Deployment?

Data Locations - Production



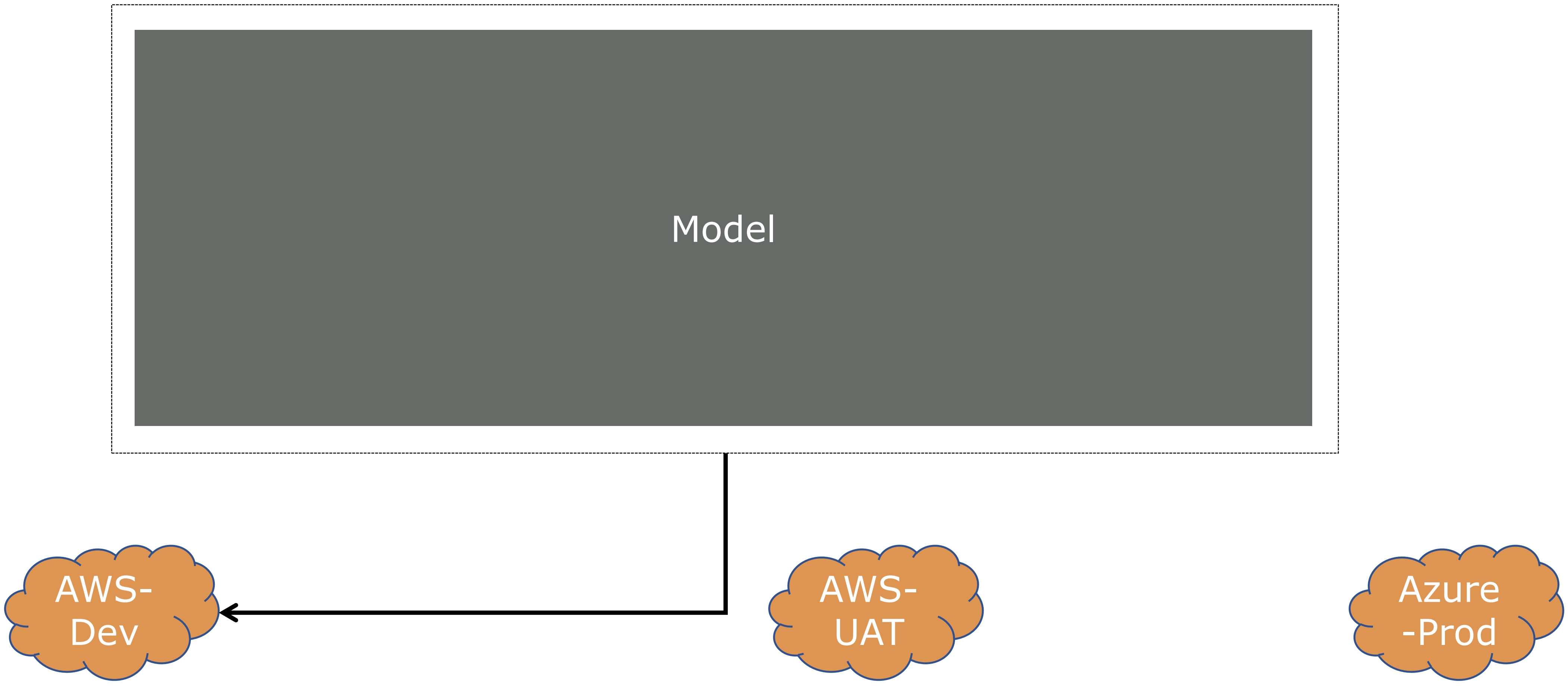
What Can Change to Deployment?

Model Environment



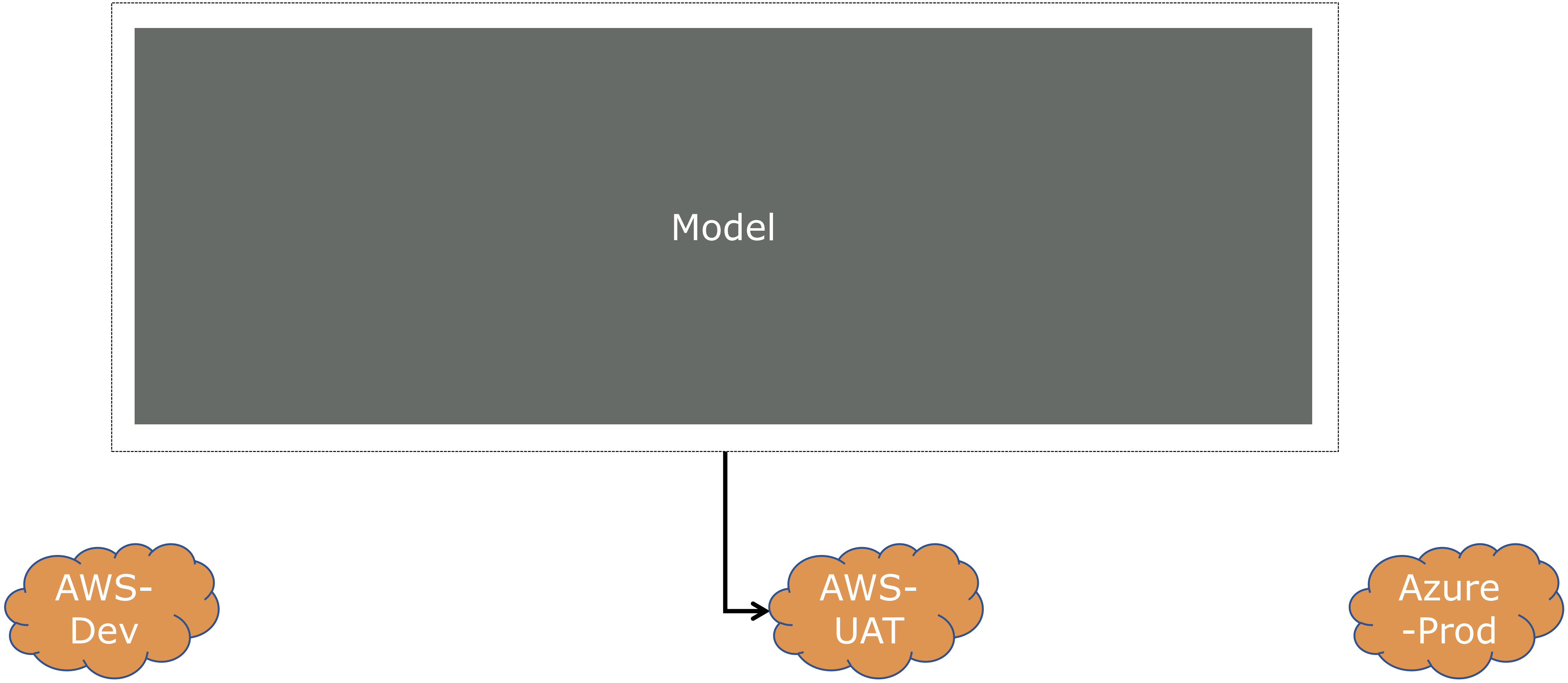
What Can Change to Deployment?

Model Environment - Dev



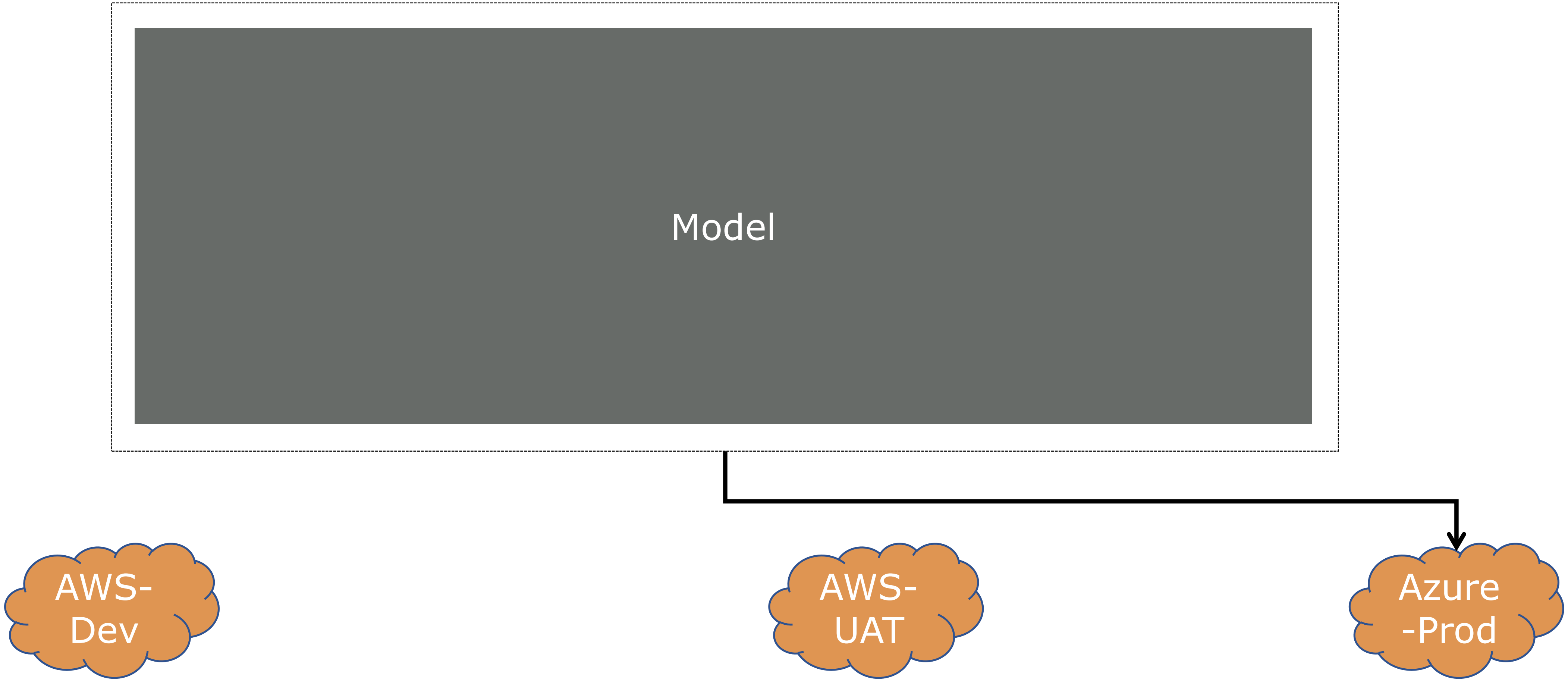
What Can Change to Deployment?

Model Environment - UAT



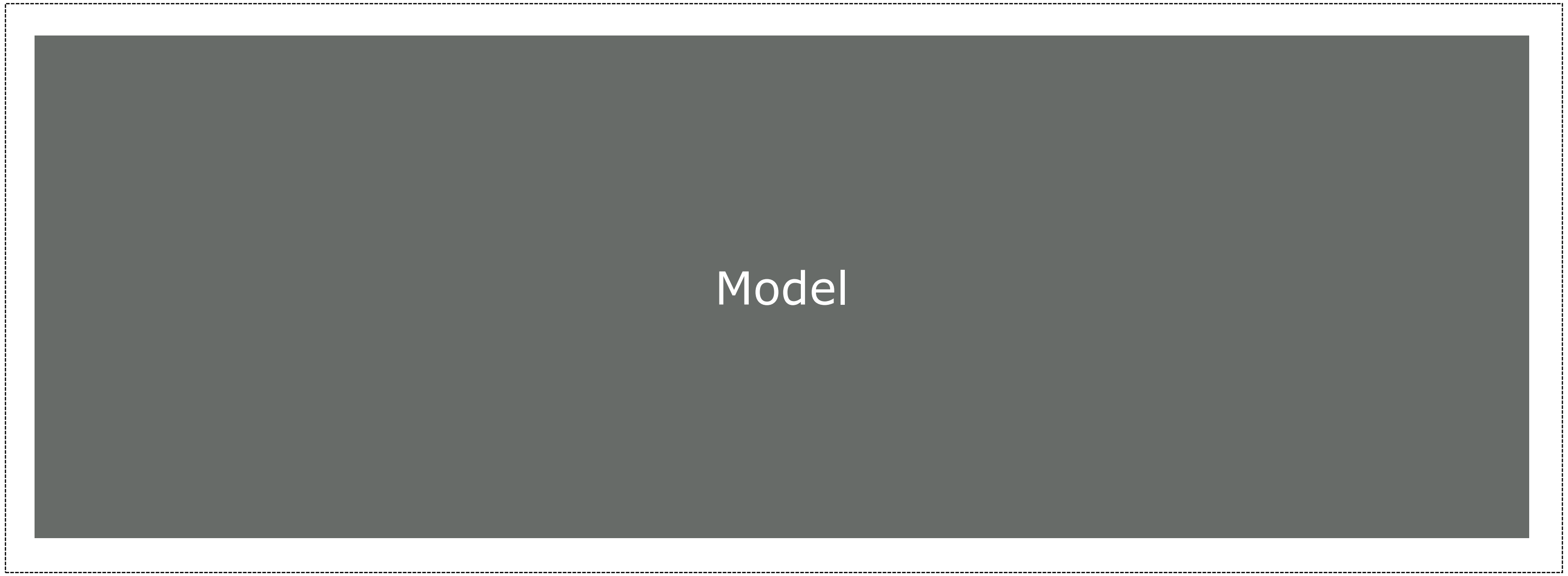
What Can Change to Deployment?

Model Environment - Prod



What Can Change to Deployment?

Model Environment



*Must support model language and libraries

What Can Change to Deployment?

	Dev	UAT	Production
SCM	Open	Restricted	Read only
Meta data	All versions	Test results	All results
Security	DS test against sandbox	AI eng test against prod data	No access

Top Process Challenges

Volume of Changes

Diversity of Teams Involved

Rapid Iteration

What is the solution?

Goal: Build for high quality improvements, quickly

Top Process Challenges

Volume of Changes

Diversity of Teams Involved

Rapid Iteration

What is the solution?

Goal: Build for high quality improvements, quickly

Who is Involved and What do they Care About?

Business

SLAs and Acceptance Criteria

IT

Security, costs, sustaining

Data Science

Build an impactful model

Engineering

Automation, scale, tooling

Top Process Challenges

Volume of Changes

Diversity of Teams Involved

Rapid Iteration

What is the solution?

Goal: Build for high quality improvements, quickly

Top Process Challenges

Volume of Changes

Diversity of Teams Involved

Rapid Iteration

What is the solution?

Goal: Build for high quality improvements, quickly

Changes *After* Deployment

What causes the need for rapid iteration?

- Doesn’t run
  - Data structure
  - Environment updates
- Doesn’t run “good” enough
  - Statistical properties of data
  - Model standards
- Doesn’t run fast enough
  - Data volume change
  - Resource allocation

Top Process Challenges

Volume of Changes

Diversity of Teams Involved

Rapid Iteration

What is the solution?

Goal: Build for high quality improvements, quickly

Creating Meaningful Abstractions to Supplement the Process

Abstraction	Description
Model Artifacts – Coefficients	Referenceable statistics on trained data
Execution Ready Model	Scoring Code
Model Data Definitions	Avro Schemas
Data Transport Descriptors	JSON file with necessary information
Model Environment Requirements	Mode library/code dependencies
Performance Monitoring Models (Telemetry)	Models custom to report performance statistics
Sensors	Computational performance tracking

Core Concepts to Enable MDLC

Previous View

Model

Core Concepts to Enable MDLC

23 lines (17 sloc) | 622 Bytes

Download | Raw | Blame | History

```python
# fastscore.input: gbm_input
# fastscore.output: gbm_output

import cPickle
import numpy as np
import pandas as pd

from sklearn.ensemble import GradientBoostingRegressor
from sklearn.pipeline import Pipeline

import imp

def begin():
    FeatureTransformer = imp.load_source("FeatureTransformer",
                                         "score_auto_gbm/FeatureTransformer.py")
    global gbmFit
    with open("score_auto_gbm/gbmFit.pkl", "rb") as pickle_file:
        gbmFit = cPickle.load(pickle_file)

def action(datum):
    score = list(gbmFit.predict(pd.DataFrame([datum])))[0]
    yield score
```

Core Concepts to Enable MDLC

31 lines (30 sloc) | 1.09 KB

```
{
  "type": "record",
  "name": "CarRecord",
  "fields": [
    {"name": "make", "type": "string"},
    {"name": "fuelType", "type": "string"},
    {"name": "aspiration", "type": "string"},
    {"name": "numDoors", "type": "string"},
    {"name": "bodyStyle", "type": "string"},
    {"name": "driveWheels", "type": "string"},
    {"name": "engineLocation", "type": "string"},
    {"name": "wheelBase", "type": "double"},
    {"name": "length", "type": "double"},
    {"name": "width", "type": "double"},
    {"name": "height", "type": "double"},
    {"name": "curbWeight", "type": "int"},
    {"name": "engineType", "type": "string"},
    {"name": "numCylinders", "type": "string"},
    {"name": "engineSize", "type": "int"},
    {"name": "fuelSystem", "type": "string"},
    {"name": "bore", "type": "double"},
    {"name": "stroke", "type": "double"},
    {"name": "compressionRatio", "type": "double"},
    {"name": "horsepower", "type": "int"},
    {"name": "peakRPM", "type": "int"},
    {"name": "cityMPG", "type": "int"},
    {"name": "highwayMPG", "type": "int"},
    {"name": "price", "type": "int"}
  ]
}
```

2 lines (1 sloc) | 19 Bytes

```
{"type": "double"}
```

Core Concepts to Enable MDLC

14 lines (13 sloc) | 195 Bytes


```
{
  "Transport": {
    "Type": "file",
    "Path": "/root/data/input_data.jsons"
  },
  "Envelope": {
    "Type":"delimited"
  },
  "Encoding": "json",
  "Schema": {
    "$ref":"gbm_input"
  }
}
```


13 lines (12 sloc) | 225 Bytes


```
{
  "Version": "1.2",
  "Loop": false,
  "Transport": {
    "Type": "kafka",
    "BootstrapServers": ["172.17.0.1:9092"],
    "Topic": "input"
  },
  "Envelope": null,
  "Encoding": "json",
  "Schema": {"$ref":"gbm_input"}
}
```

What Can Change?

Transport Modes

Data Schema - Input	Model Execution Code	Data Schema - Output
Data Transport mode		Data Transport mode

Will depend on use case, phase, and application needs: Batch, on-demand or "streaming"

Core Concepts to Enable MDLC

Language Agnostic

Data Schema - Input	Model Execution Code	Data Schema - Output
Data Transport mode		Data Transport mode

python

Infrastructure Independenc
(Microservice)

```
import cPickle
import numpy as np
import pandas as pd

from sklearn.ensemble import GradientBoostingRegressor
from sklearn.pipeline import Pipeline

import imp
```

Core Concepts to Enable MDLC

Data Schema - Input

Model Execution Code

Data Schema - Output

Data Transport mode

Data Transport mode

The Unit of Execution for a Model

Model Telemetry

Data Schema - Input

Model Execution Code

Data Schema - Output

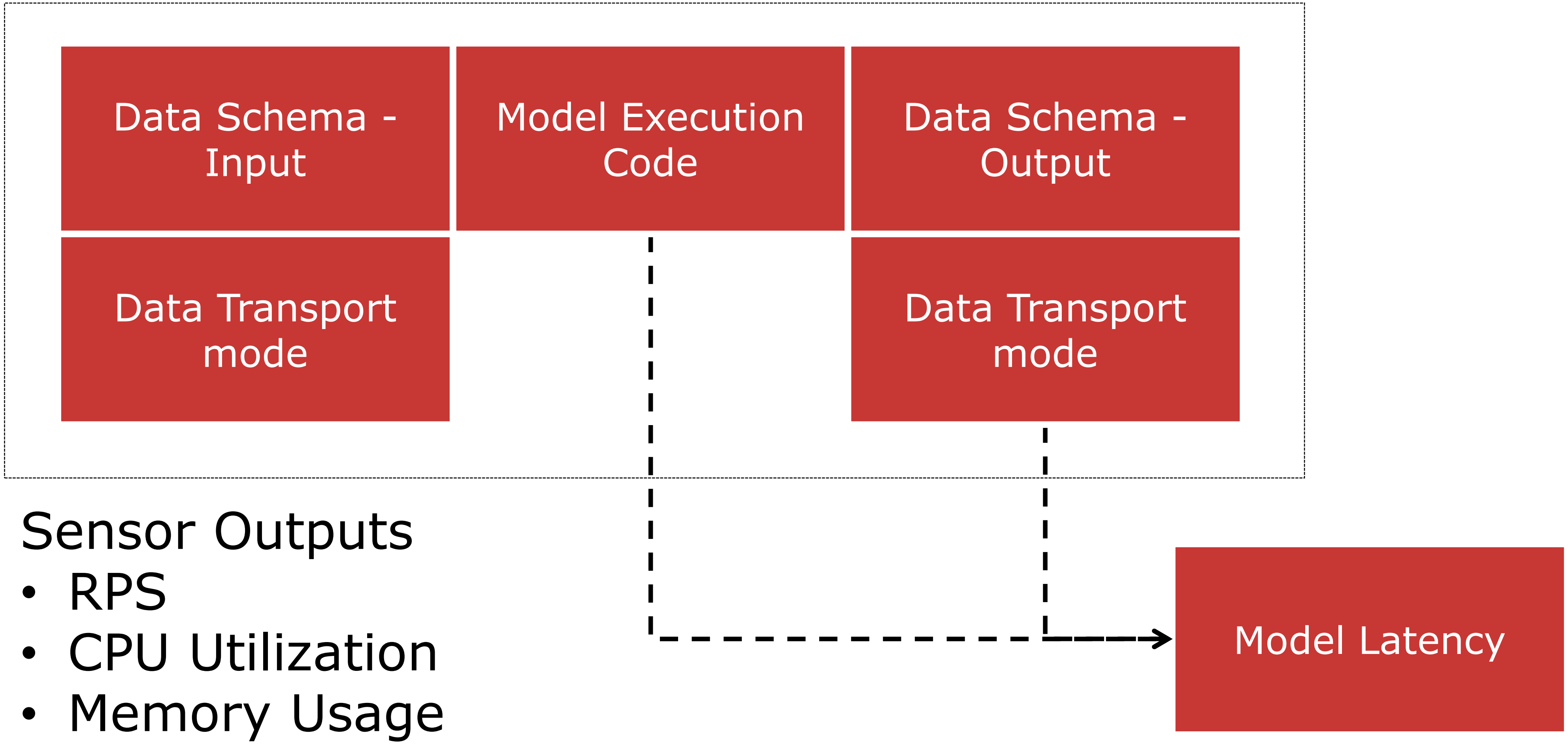
Data Transport mode

Data Transport mode

Model Acceptance

Model Operations
- Model Drift
- Data Relationships
- Final Acceptance Threshold

Model Telemetry

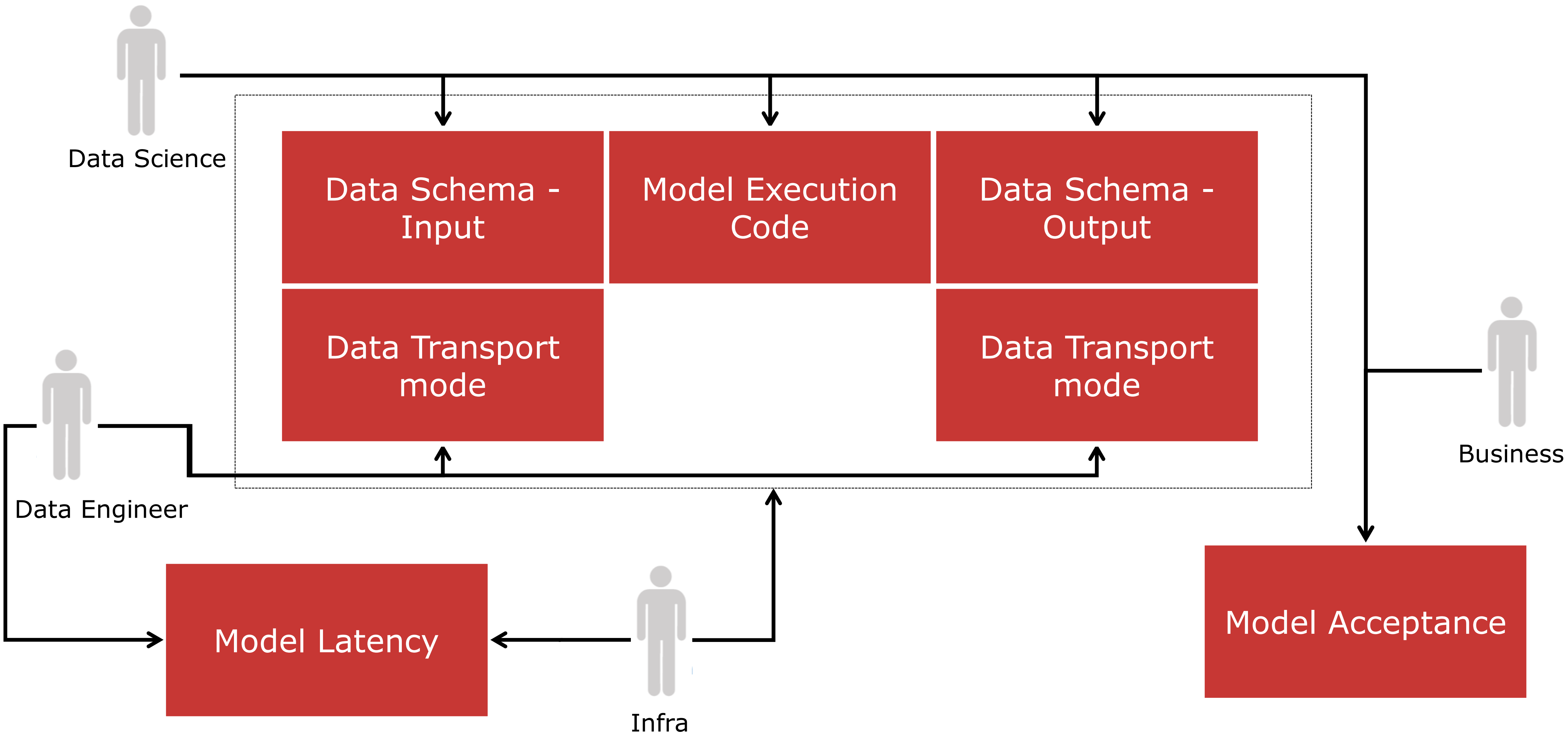


Sensor Outputs
- RPS
- CPU Utilization
- Memory Usage

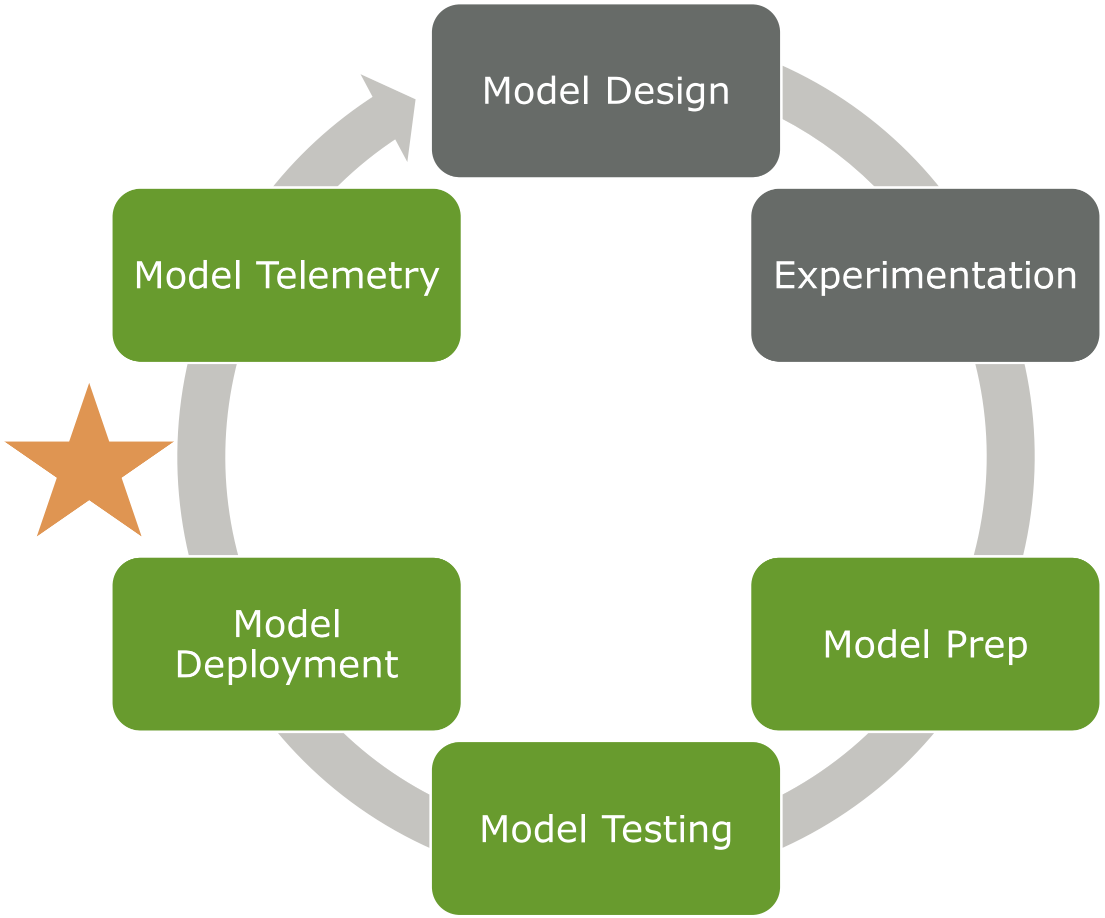
Creating Meaningful Abstractions to Supplement the Process

Abstraction	Description
Model Artifacts – Coefficients	Referenceable statistics on trained data
Execution Ready Model	Scoring Code
Model Data Definitions	Avro Schemas
Data Transport Descriptors	JSON file with necessary information
Model Environment Requirements	Mode library/code dependencies
Performance Monitoring Models (Telemetry)	Models custom to report performance statistics
Sensors	Computational performance tracking

Who Owns What?



Model Development Life Cycle



Model Prep

What assumptions are made during creation?

- Model execution environment will be consistent
- Model will receive and produce the same type of data
- Model will perform to SLAs
  - Latency
  - Accuracy (or other performance metric)

Model Testing

- Unit Testing – does the model execution properly
- End-to-End Testing - does the analytic workflow execute properly
- Acceptance Testing – does the model perform to SLAs
- Comparison Testing – A/B Testing

Model Deployment

Pre-Production (Technical Production)

- Production data
- Production Infrastructure
- Outputs to review

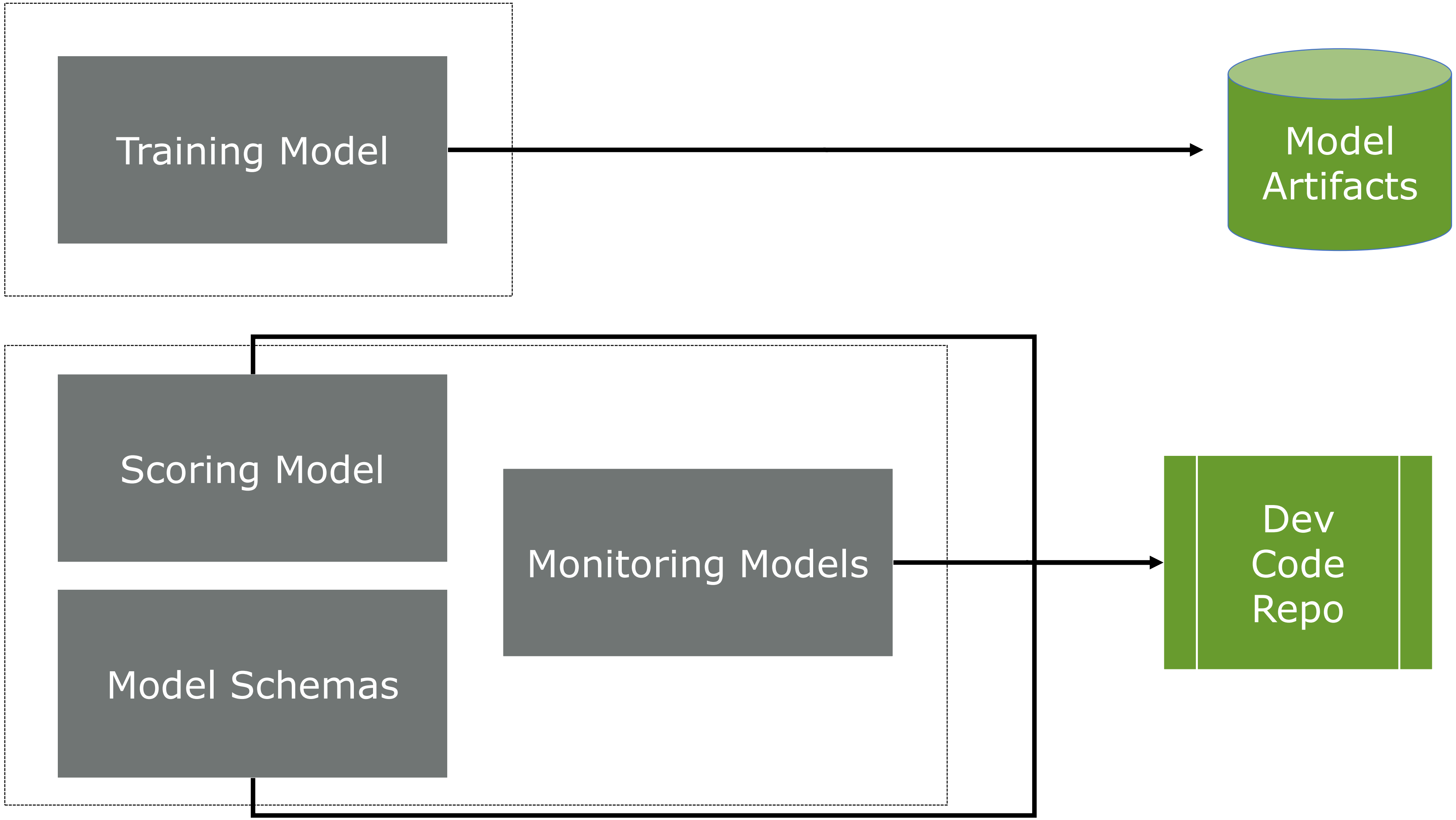
Production (Business Production)

- Production data
- Production Infrastructure
- Outputs to production systems/operations for usage

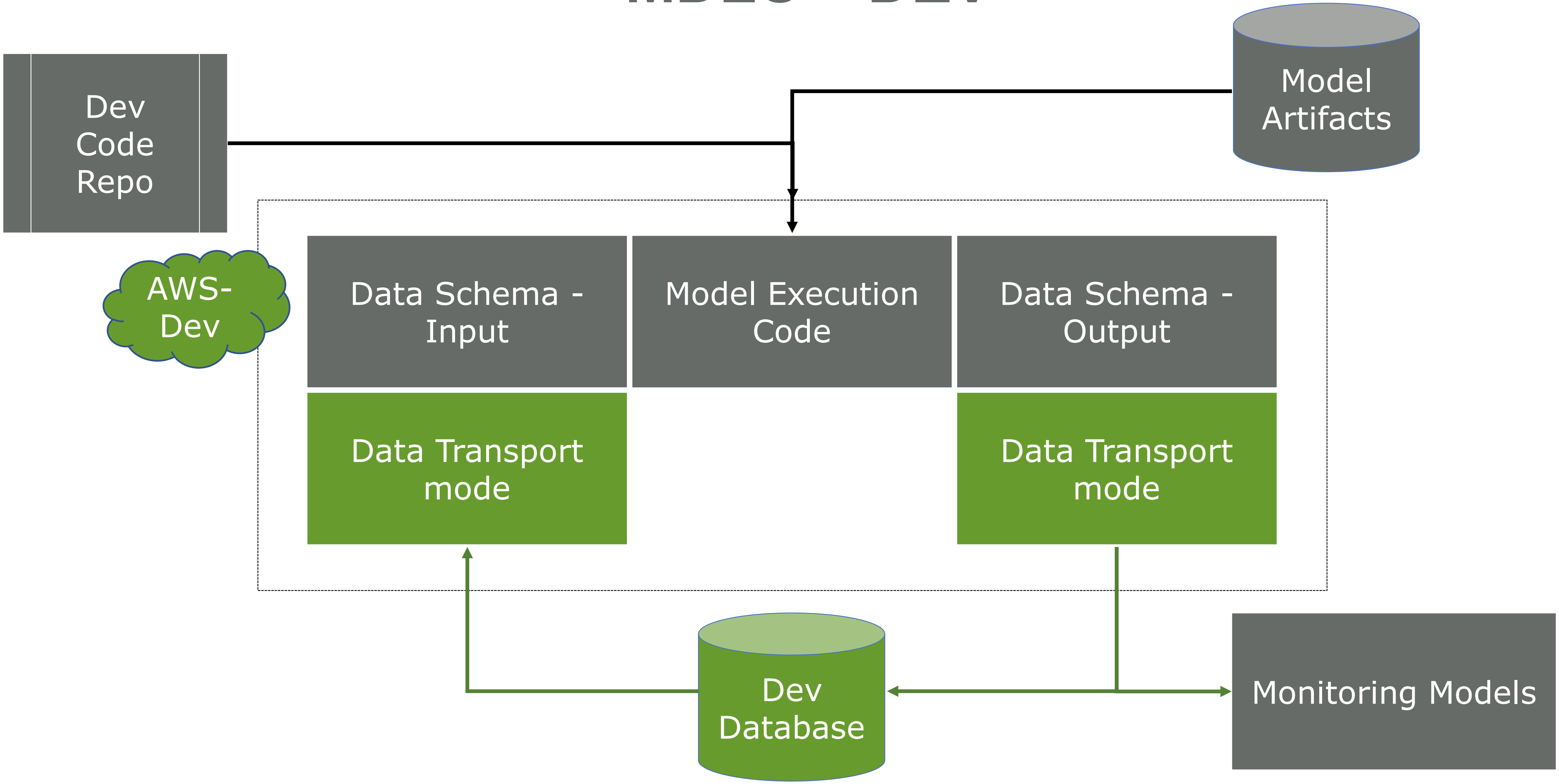
Model Performance Monitoring

- What aspects of the model are important to watch?
- What statistic identifies the quality of outputs?
- What is the calculated limit that would describe an "unacceptable" output?
- How do you define an "unacceptable" output?
- How many times does the model reach the threshold before the limit is hit?
- What actions are taken when the limit is hit?
- Who is responsible for taking action?

MDLC – LAB DEV

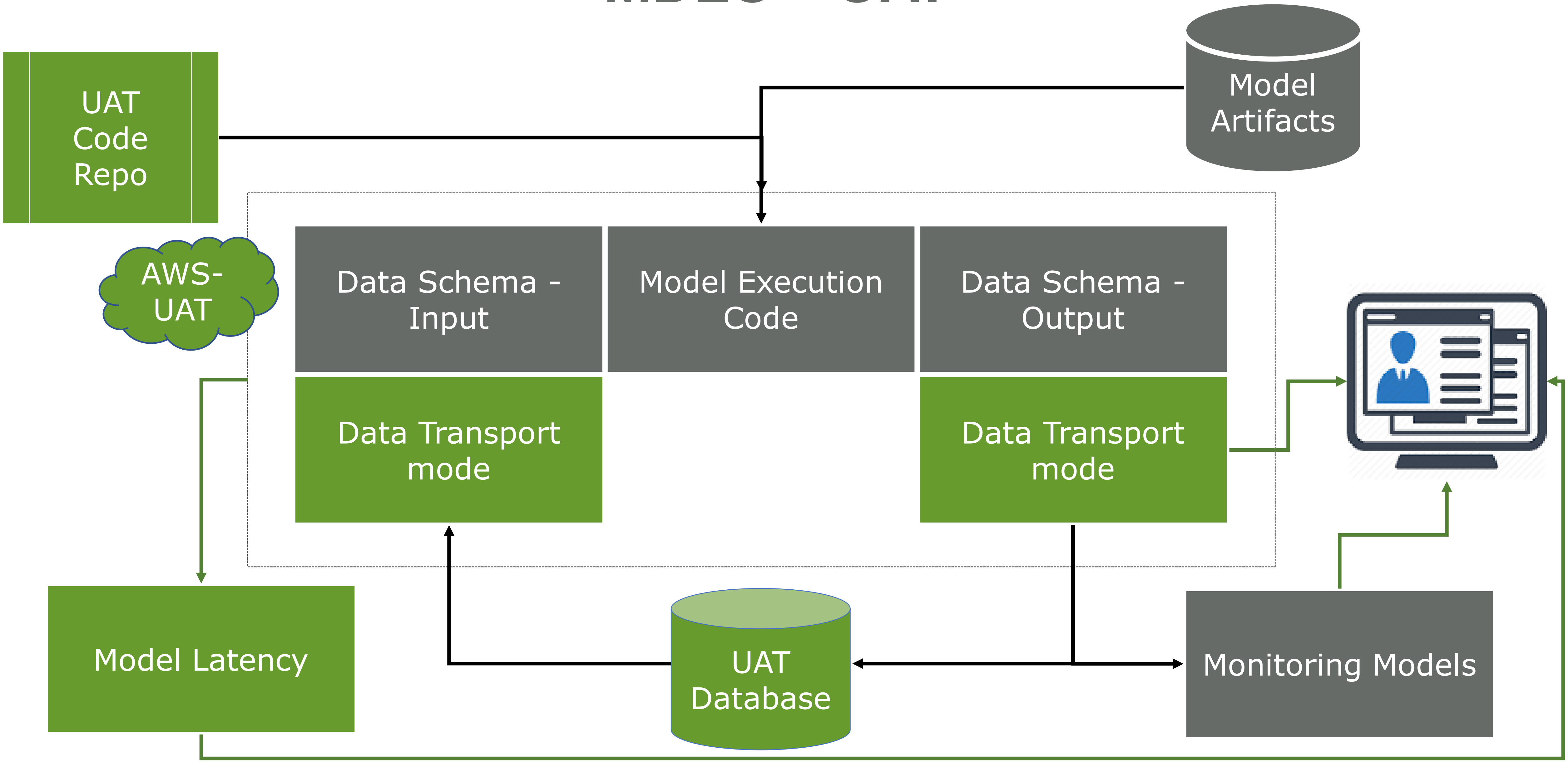

Training Model
Model Artifacts
Scoring Model
Monitoring Models
Model Schemas
Dev Code Repo

MDLC – DEV



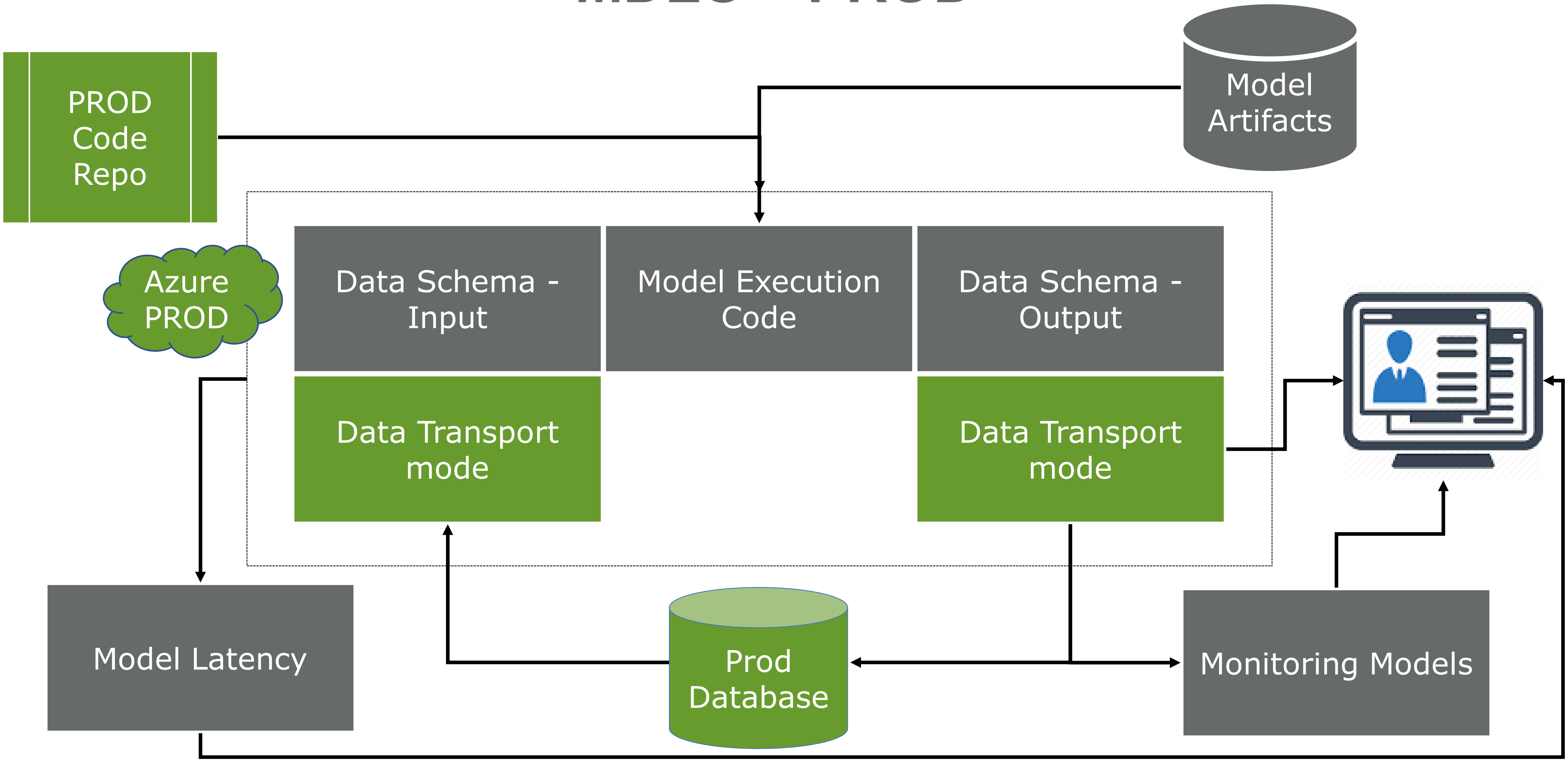
Dev Code Repo
Model Artifacts
AWS-Dev
Data Schema - Input
Model Execution Code
Data Schema - Output
Data Transport mode
Data Transport mode
Dev Database
Monitoring Models

MDLC – UAT



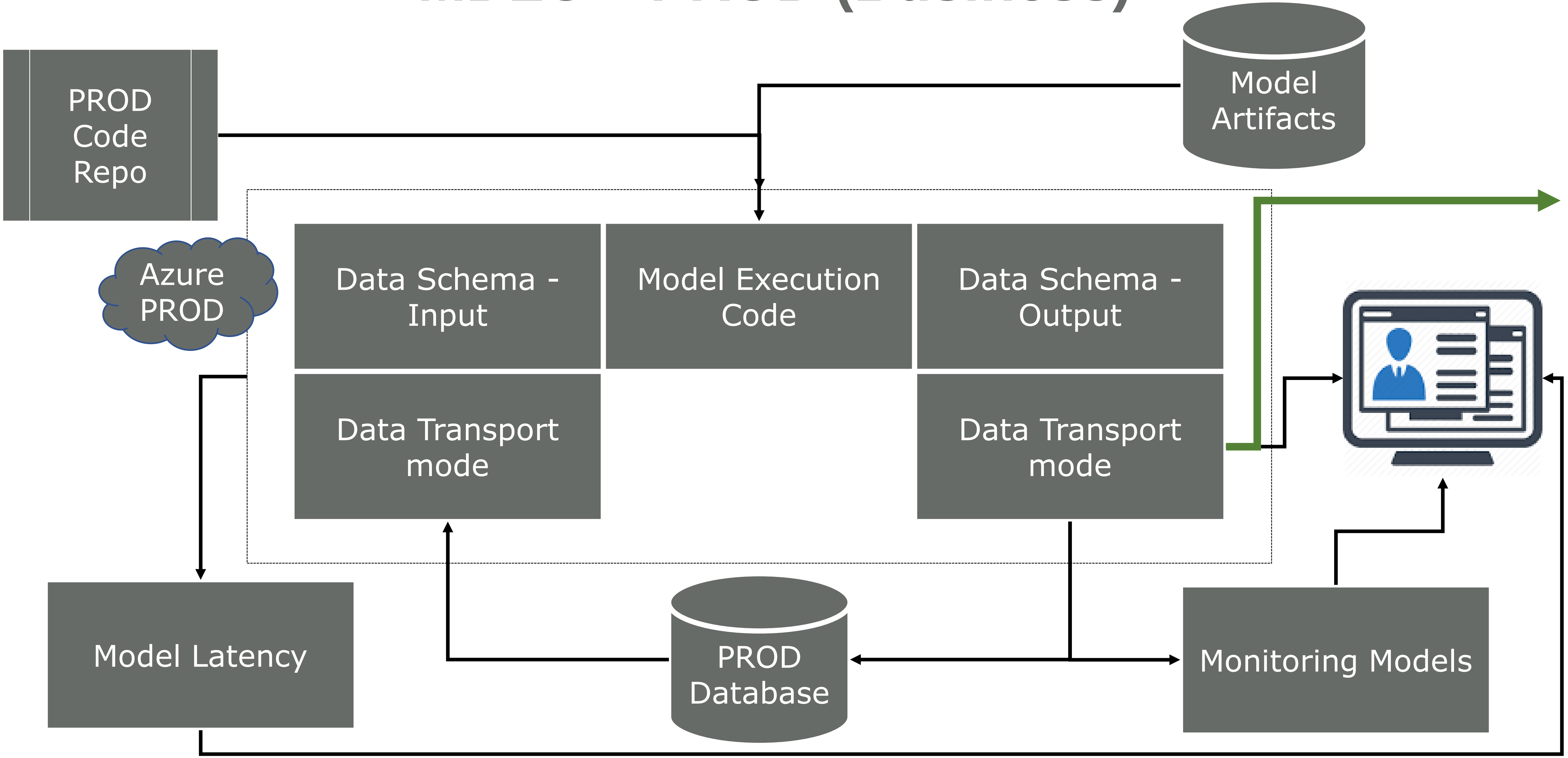
UAT Code Repo
Model Artifacts
AWS-UAT
Data Schema - Input
Model Execution Code
Data Schema - Output
Data Transport mode
Data Transport mode
Model Latency
UAT Database
Monitoring Models

MDLC – PROD



PROD Code Repo
Model Artifacts
Azure PROD
Data Schema - Input
Model Execution Code
Data Schema - Output
Data Transport mode
Data Transport mode
Model Latency
Prod Database
Monitoring Models

MDLC – PROD (Business)



Goal: Build for high quality improvements, quickly

Thank You