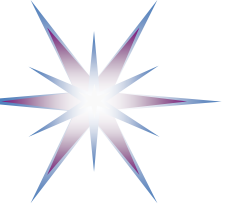


From DevOps to MLOps and DataOps:

Operationalising Data Science Analytics and ML

Yuri Demchenko
CCI Group
University of Amsterdam

Outline

- Data Science Project Management
- Data Science/Data Mining Process models
  - CRISP-DM, TDSP, ASUM
- Data Science and ML pipeline
- Operationalising Data Science Analytics and ML: DataOps, MLOps and platforms
  - DataOps processes and requirements
- MLOps with Azure
- MLOps with AWS
- Summary and take away
- Additional information
  - General project management framework
  - Details on CRISP-DM process

Historical outline

- Data Science is claimed to be mentioned in 1960s
- DevOps is a trend since 2009
- Cloud Computing as a facilitator and a platform for Infrastructure as Code since 2011
- Big Data started as a buzz word since 2013
- DataOps papers – since 2018
- MLOps platforms – since 2019

- What are factors and trends to declare DataOps and MLOps?

Data Science Project Management

- Data Science development process and DevOps
  - Data Science methodology and Research Methods
- Process Models
  - CRISP-DM, CRoss-Industry Standard Process for Data Mining
  - ASUM, Analytics Solutions Unified Method (IBM)
  - TDSP, Team Data Science Process (Microsoft)

Big Data Infrastructure and computing facility management

- Large scale storage
- Databases and Data Warehouse
- Applications, licenses

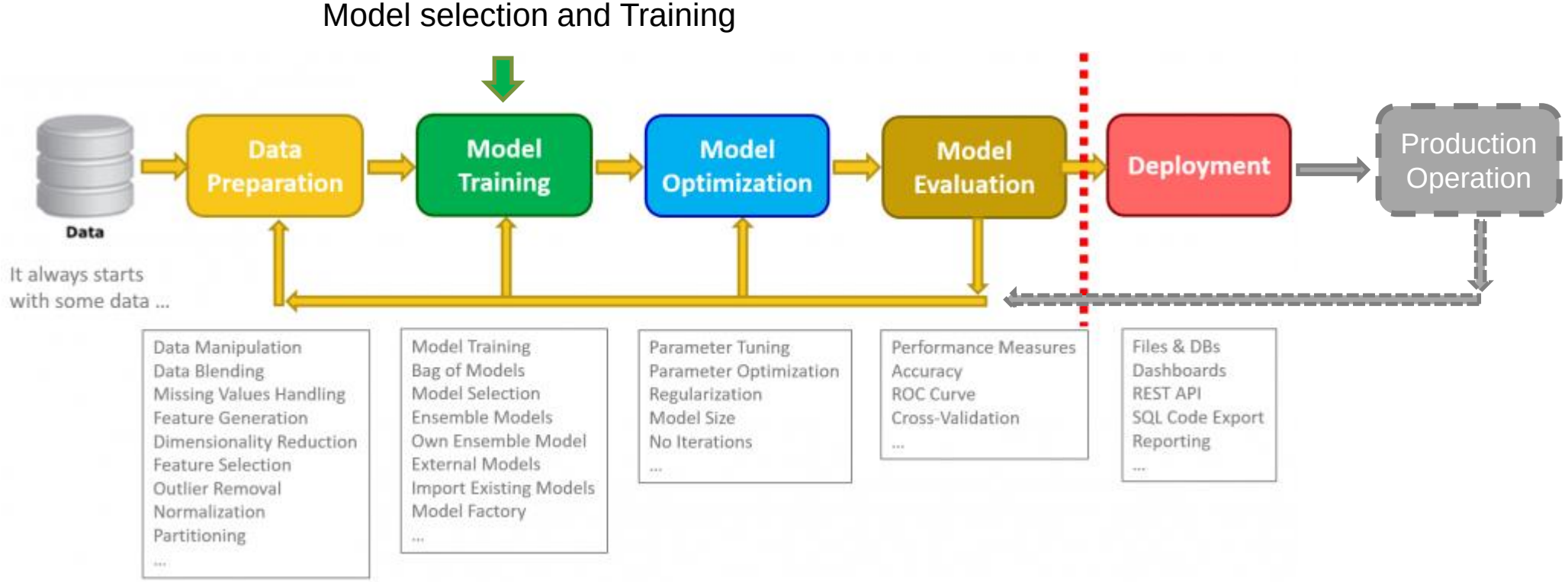
Data Science Development Process

Data Science process is dealing with the data pipelines that include stages:

- **Collecting data** from multiple sources, also blending process or business data with external data such as environmental data or social media data that can be obtained via WebAPI or web scraping
- **Working with data** including data preparation, cleaning, filtering, and reformatting for modeling needs
- **Combing datasets** by joining on common attributes, consolidating attributes, build tabular data structure (such as used in popular analytics programming languages R, python, scala)
- **Feature engineering, algorithm selection**
- **Testing** before production and **validating** the model during production, in particular, detecting drift in predictive models
- Implement changes and **deploy an updated model**.

Every Data Science Project/Research is based on Data Flow/Lifecycle

https://www.knime.com/blog/analytics-and-beyond



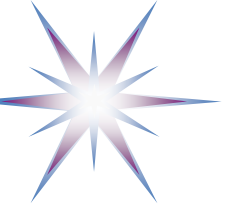
- Each phase – data preparation, model training and evaluation, and model deployment – operates on **its own dataset**. All these data sets need to be completely isolated from each other. The pollution of data sets across the data science assembly line is one of the most frequent mistakes in model production.
- The data science journey/project always starts with some ***historical data*** *or* ***sample dataset*** lying around in a repository.
- ***Data blending*** involves additional data and connecting external sources

Data Science and Research Methods

- **Data Science needs methodology**
- Research is an organized and systematic way to find answers to questions
  - Data Science uses data analysis to answer questions
- Research is a creative process
  - So does Data Science
- To effectively work as a Data Scientist you need to know Research Methods
  - Sufficient to have your confident opinion (and ask questions)

- **Data Science and Analytics projects need development and operationalization methodology and platform**

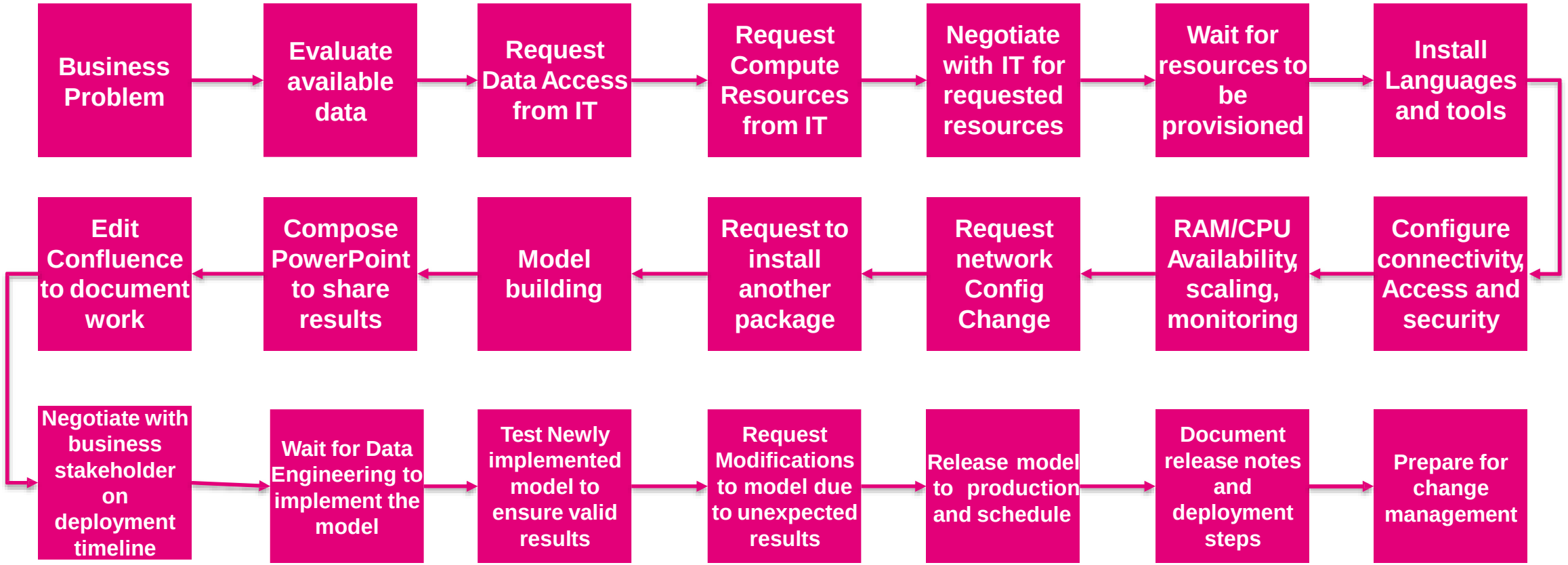
Research process cycle
- Design Experiment
- Collect Data
- Analyse Data
- Identify Patterns
- Hypothesis Explanation
- Test Hypothesis



Data Science and Big Data Infrastructure and Tools

- Challenge: *Data Science is trapped in laptops*
  - There is a big step from laptop and Jupyter Notebook to production data and production application
- Productionalising Data Science Apps means working with Big Data
  - Means Big Data Infrastructure and tools
- Data Science team need to work with Software Engineers, Operators and Infrastructure Engineers
  - Data Science teams need to know their DevOps practices
  - DevOps Engineers need to know specifics of the Data Science projects

Traditional/Old process: Multiple challenges in the process of turning data into value on existing infrastructure



"DataOps is independent approach to analytics" Andy Palmer

[ref] DataOps: 9 steps to transform your data science impact, Strata Conference 2018

DataOps and MLOps

- DataOps, MLOps is about operationalizing ML and Data Analytics

MLOps and DataOps: DevOps for ML and Data Analytics

MLOps and DataOps are extension of **DevOps** to manage data analytics and Machine Learning data flow and process.

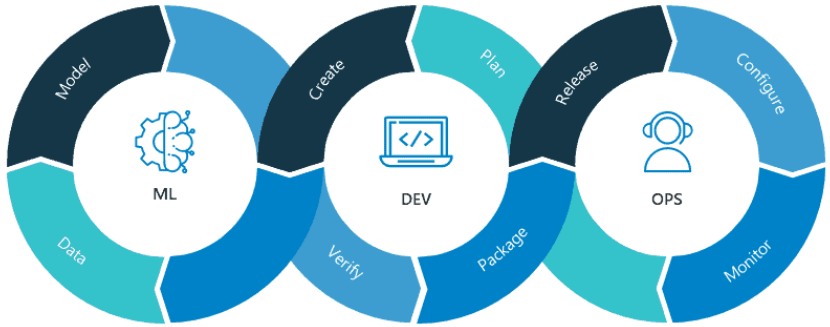
- DevOps is the practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support.
  - Develop – Build – Deploy – Operate
  - Cloud is an enabler for DevOps processes

DataOps, MLOps is about operationalizing ML and Data Analytics

- Different nature of processes at the stage of ML model development
- Benefit from the DevOps and Agile processes and culture
- Learn from DevOps experience

In fact, for the software product this is

ML + Dev + Ops



DevOps Essentials

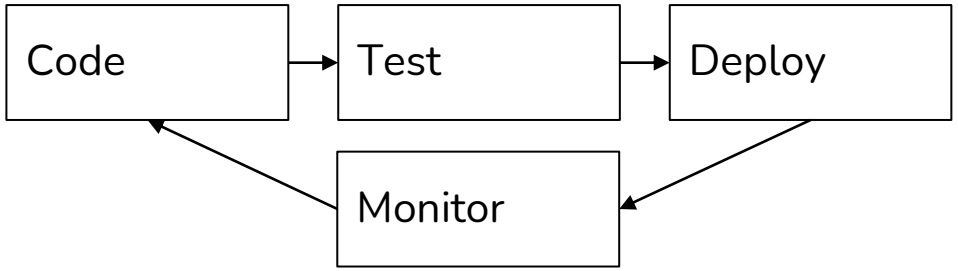
- Better Software, Faster time to market
- Movement Comes from Open Source
- Synergy of Development and Operations
- Covers the *entire* Application Lifecycle

Requirements to achieve Effective MLOps

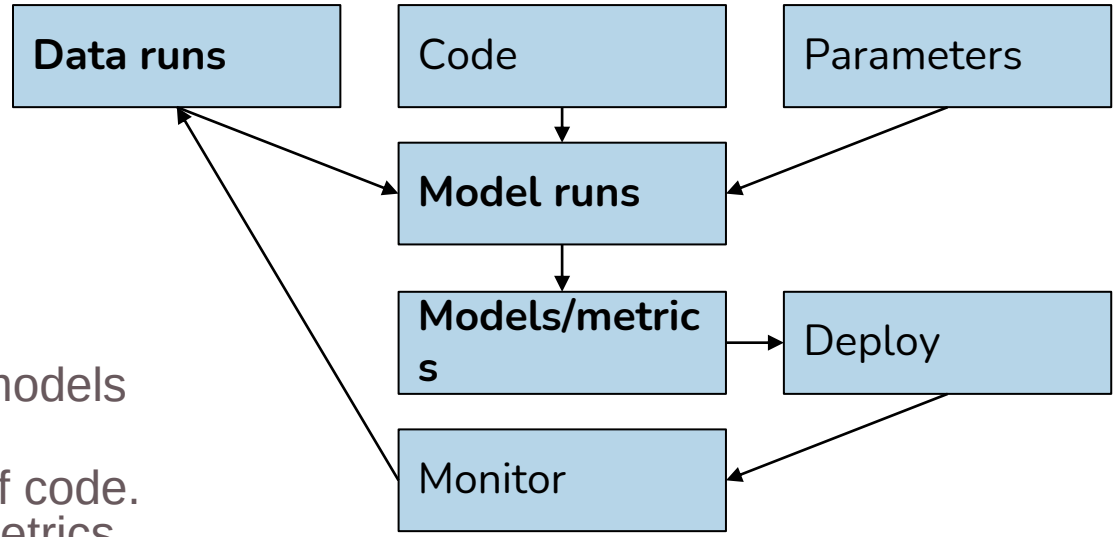
1. **Reproducible**
   Must be able to retrain a 9 month old model to within few %
2. **Accountable**
   Must be able to trace back from model in production to its provenance
3. **Collaborative**
   Must be able to do asynchronous collaboration
4. **Continuous**
   Must be able to deploy automatically & monitor statistically

Data engineering & ML are different from software development

Software Development (DevOps)



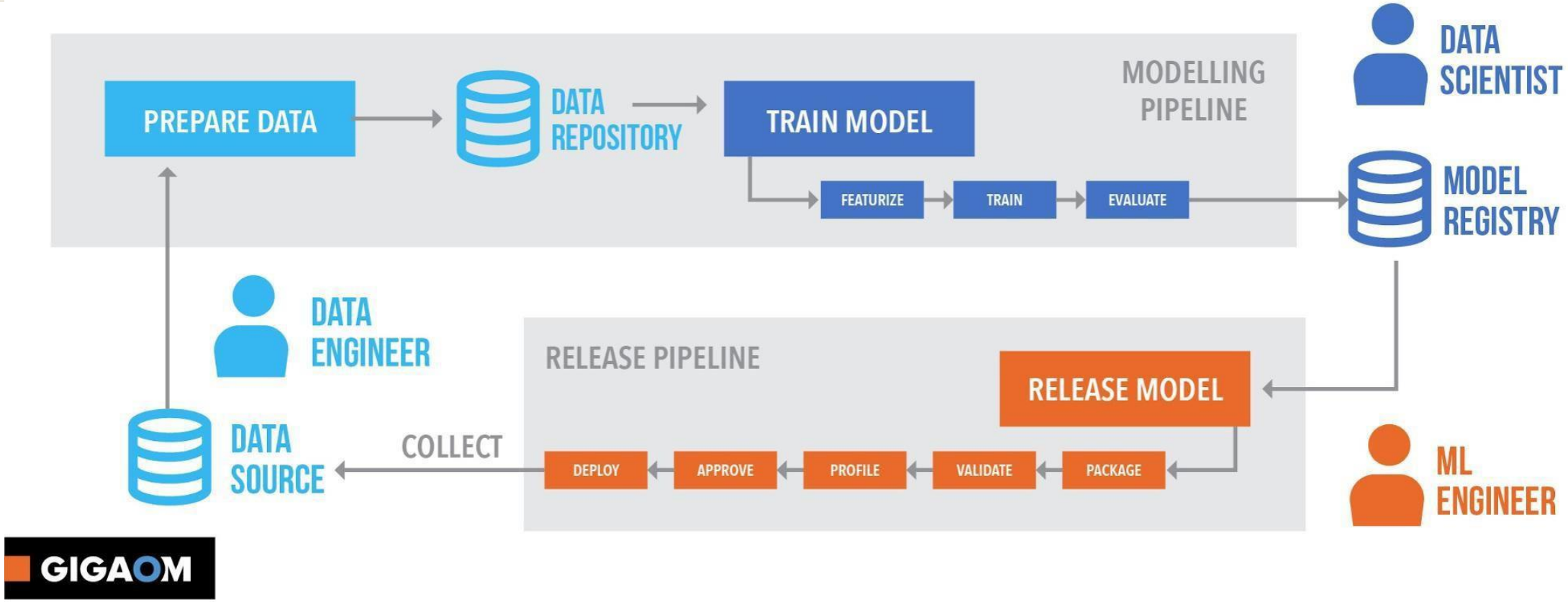
Machine Learning (MLOps)



- There are unique ways that working with data, code & models is different to just working with code.
- Normal software development tools focus on commits of code. But ML has many more moving parts: data, models & metrics.
- In ML, you need to track *runs*, bundling data, code & param versions that went into creating an intermediate dataset or a model.
  - This provides full context for reproducibility, and provenance to connect data engineering with model training to track back for accountability.
  - Requires data and model/code lineage

[ref] The Future of MLOps … *and how did we get here? By* Luke Marsden, dotscience, 2020 https://www.bristoldatascientists.org/wp-content/uploads/sites/5/2020/02/Luke-The-Future-of-MLOps.pdf

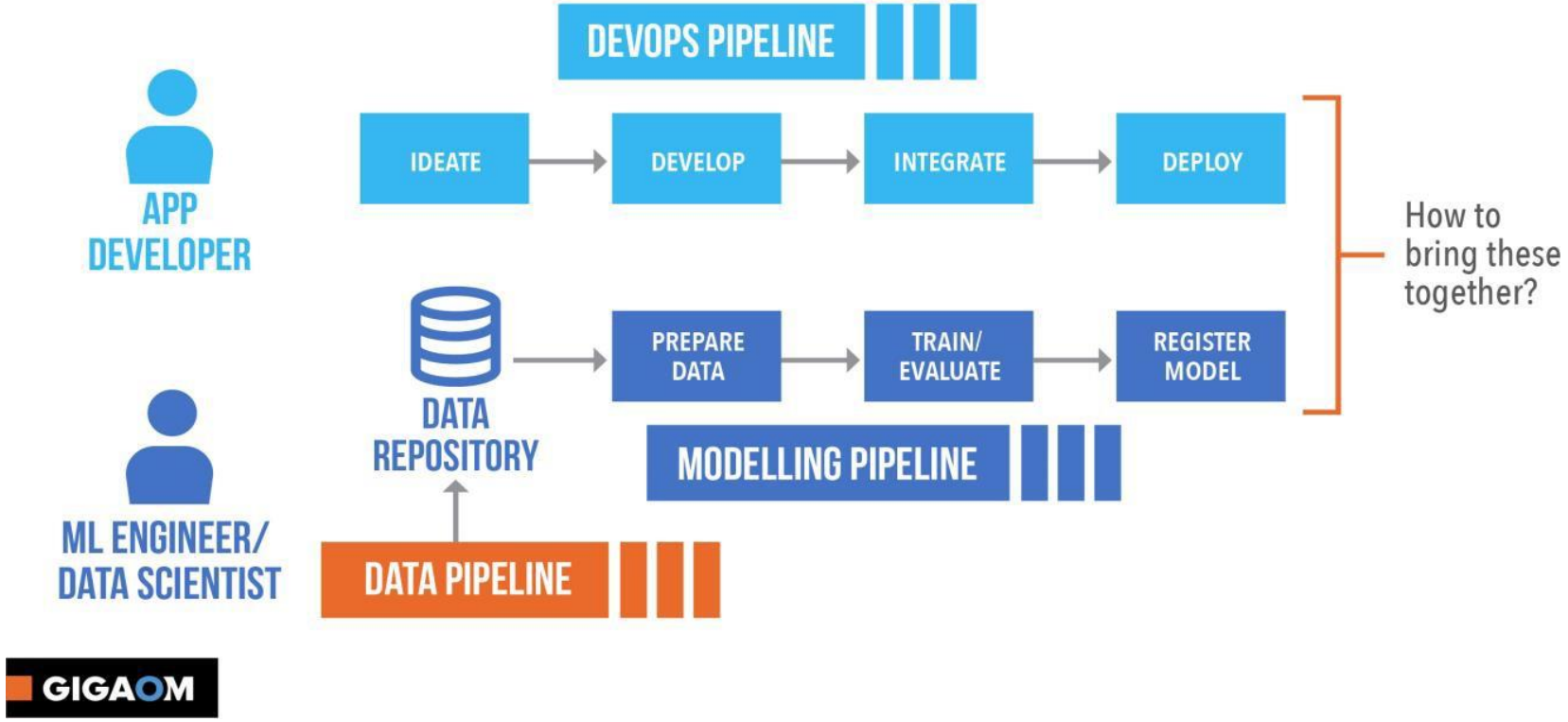
ML Application workflow



- *The ML System Incorporates a Systematic & Repeatable Workflow*

[ref] Delivering on the Vision of MLOps: A maturity-based approach, GIGAOM Report, 21 Jan 2020
https://azure.microsoft.com/mediahandler/files/resourcefiles/gigaom-Delivering-on-the-Vision-of-MLOps/Delivering%20on%20the%20Vision%20of%20MLOps.pdf
[ref] https://docs.microsoft.com/en-gb/azure/machine-learning/

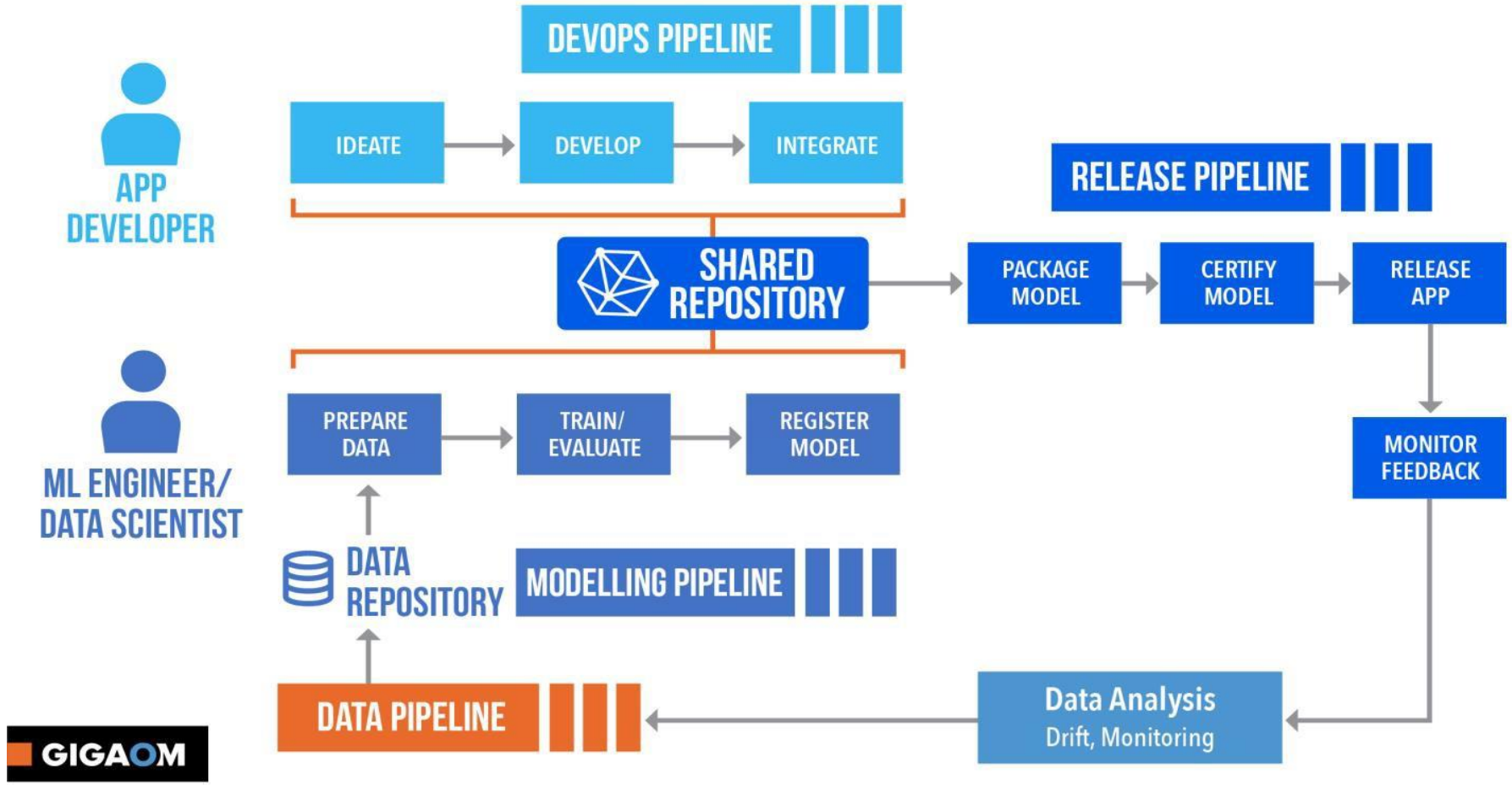
Maturity Level 1, Development & Model Creation Happen Independently



- Problem to bring application and ML model together

[ref] Delivering on the Vision of MLOps: A maturity-based approach, GIGAOM Report, 2020

Level 3, DevOps & ML Pipelines Converge to Create a Shared Release Pipeline.



- ML model and ML/AI application deployed together

[ref] Delivering on the Vision of MLOps: A maturity-based approach, GIGAOM Report, 2020

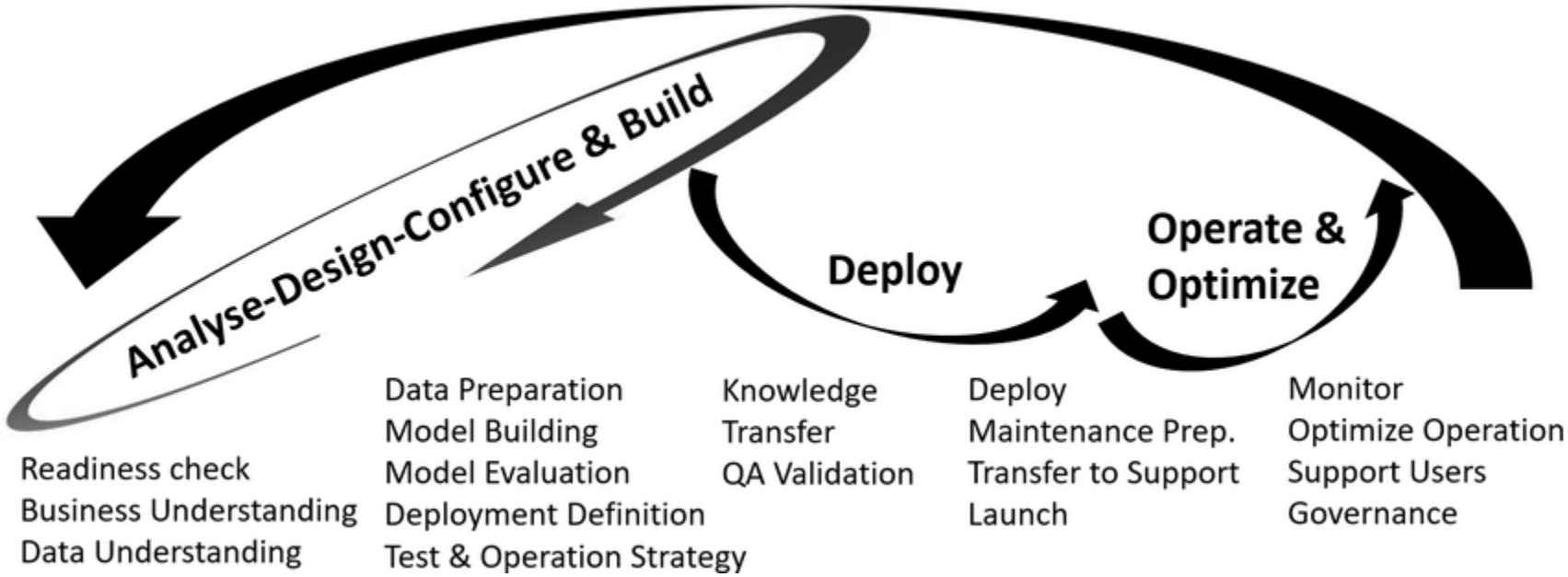
Data Science Process Models and Model Formats

- Data Science process models
  - CRISP-DM, CRoss-Industry Standard Process for Data Mining
  - ASUM, Analytics Solutions Unified Method (IBM)
  - TDSP, Team Data Science Process (Microsoft)
  - KNIME Model Process Factory (KMF)

- Data Analytics Models Formats
  - Predictive Models Markup Language (PMML)
  - Portable Format for Analytics (PFA
  - ONNX
  - TensorFlow Models

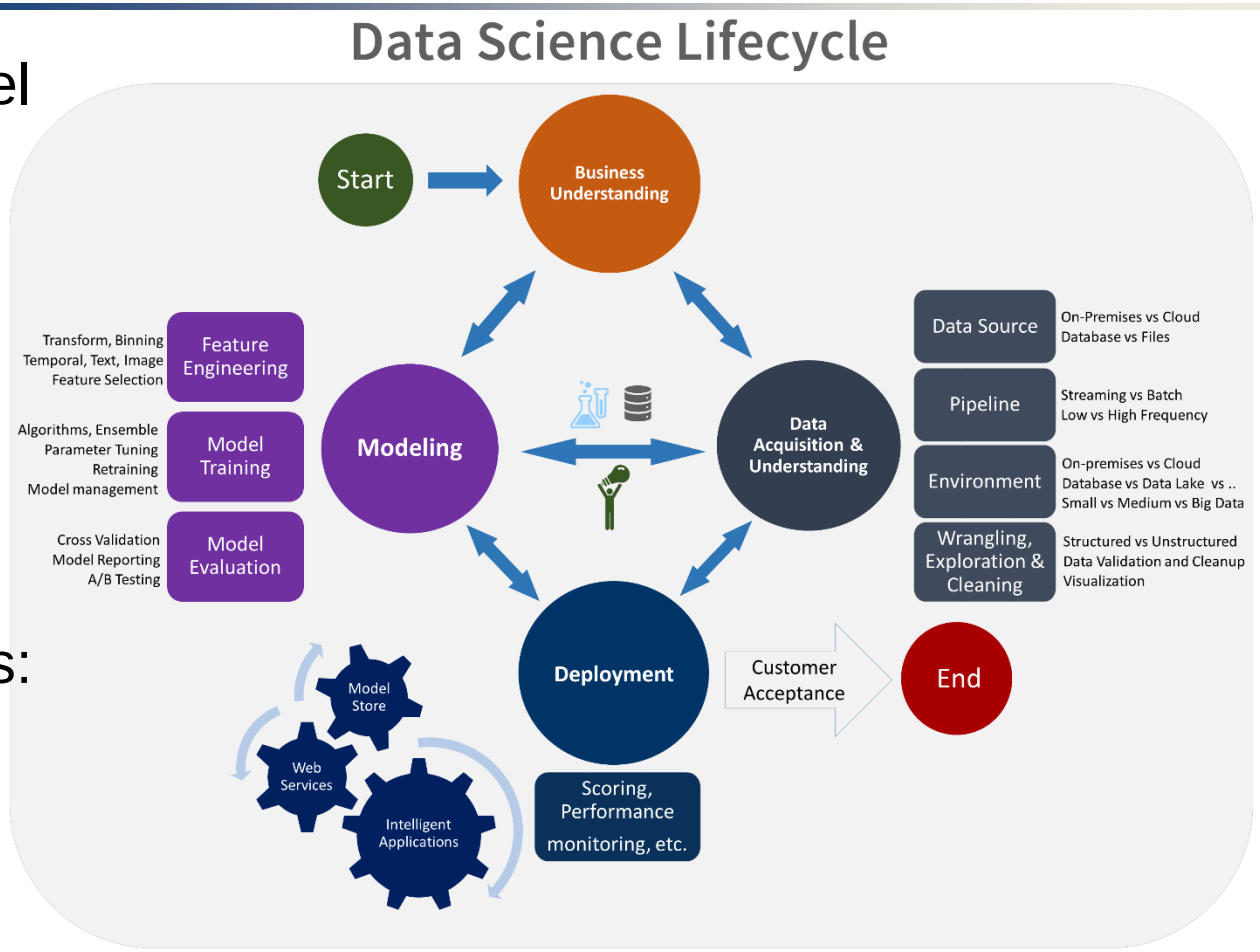
ASUM by IBM: Analytics Solutions Unified Methods

- *Analyze*. Requirements specified and agreed; contract or services agreement is signed.
- *Design*. Define all components of the solution and their relationships and dependencies, identify necessary resources.
- *Configure and Build*. The solution is developed, all components are integrated and configured.
- *Deploy*. Create a plan to run and maintain the developed solution, including configuration management and migration plan if necessary.
- *Operate and Optimize*. The solution is operational is monitoring data are collected and maintained.



Team Data Science Process (Microsoft)

- TDSP is an agile and iterative process model
  - Refactored into Azure DevOps in 2018
  - Currently supported by MLOps
- Includes components
  - A **data science lifecycle** definition
  - A **standardized project structure**
  - **Infrastructure and resources** recommended for data science projects
  - **Tools and utilities** recommended for project execution
- The lifecycle includes five sequential phases:
  - Business Understanding
  - Data acquisition and understanding
  - Modeling
  - Deployment
  - Customer acceptance



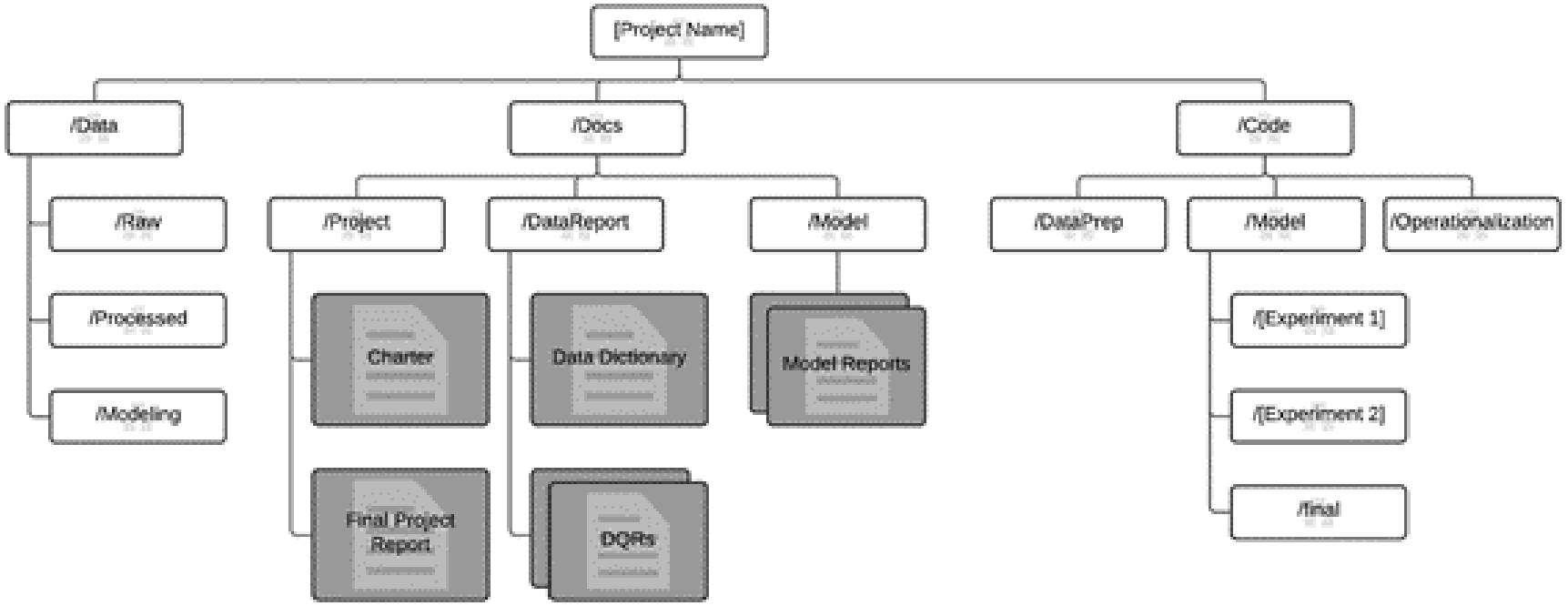
[ref] https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/overview

TDSP: Data Science Project Planning

https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/team-data-science-process-project-templates

- Microsoft Project template
- Excel template https://github.com/Azure/Azure-MachineLearning-DataScience/blob/master/Team-Data-Science-Process/Project-Planning-and-Governance/Advanced%20Analytics%20Microsoft%20Project%20Plan.xlsx
- Repository template https://github.com/Azure/Azure-TDSP-ProjectTemplate
- Walkthroughs executing the Team Data Science Process https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/walkthroughs
  - HDInsight Spark walkthroughs using PySpark and Scala These walkthroughs use PySpark and Scala on an Azure Spark cluster to do predictive analytics.
  - HDInsight Hadoop walkthroughs using Hive These walkthroughs use Hive with an HDInsight Hadoop cluster to do predictive analytics.

Recommended Git Directories structure

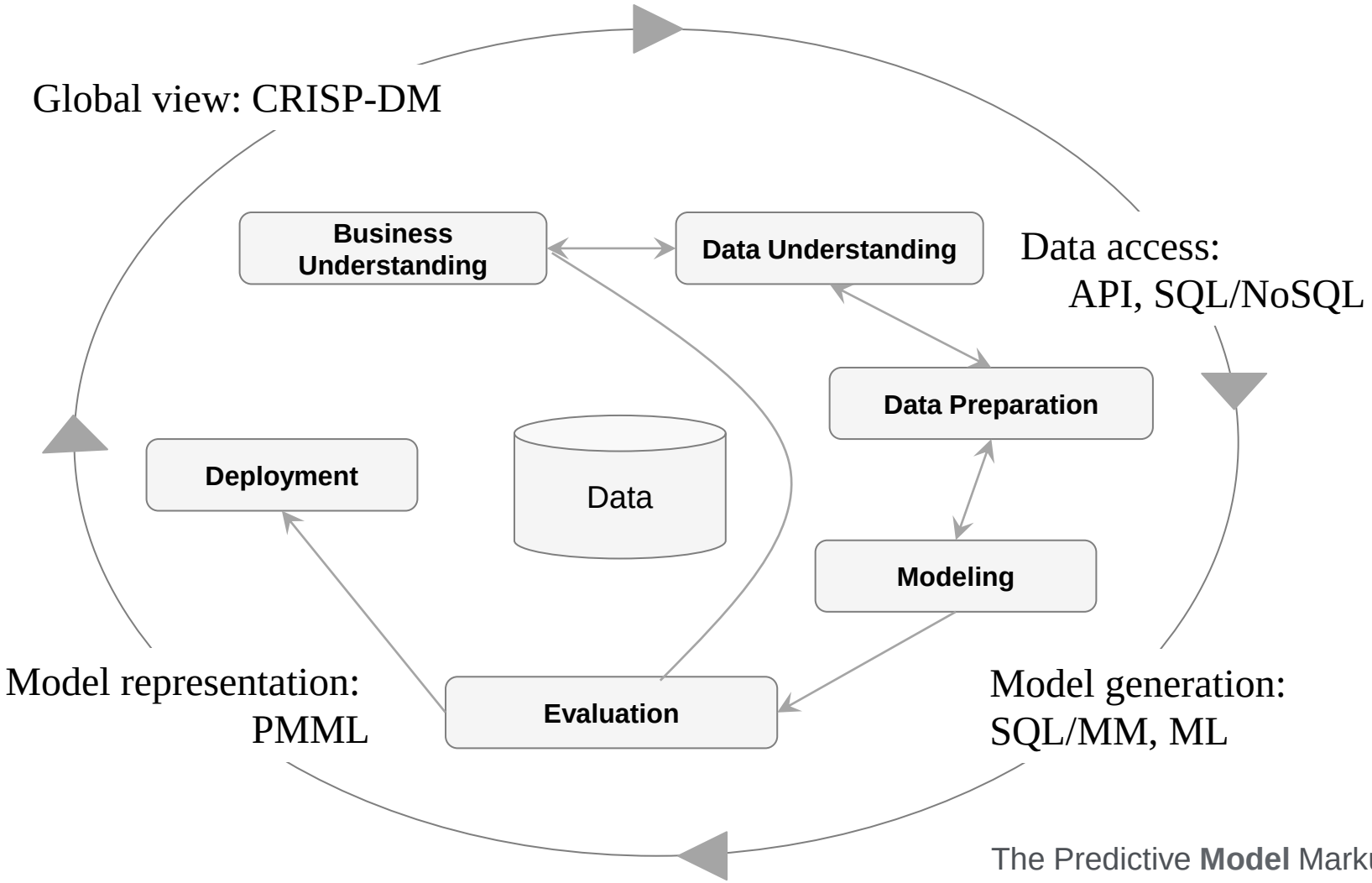


- Project_name
  - Data
    - Raw
    - Processed
    - Modelling
  - Docs
    - Project
    - DataReport
    - Model
  - Code
    - DataPrep
    - Model
    - Operationalization

What is CRISP-DM?

- Cross-Industry Standard Process for Data Mining
- Aim:
  - To develop an industry, tool and application neutral process for conducting Knowledge Discovery
  - Define tasks, outputs from these tasks, terminology and mining problem type characterization
- Founding Consortium Members: DaimlerChrysler, SPSS and NCR
  - CRISP-DM v1.0 1999, CRISP-DM v2.0 2008
- CRISP-DM Special Interest Group ~ 200 members
  - Management Consultants
  - Data Warehousing and Data Mining Practitioners

CRISP DM: Processes and Data Lifecycle - Original (1)



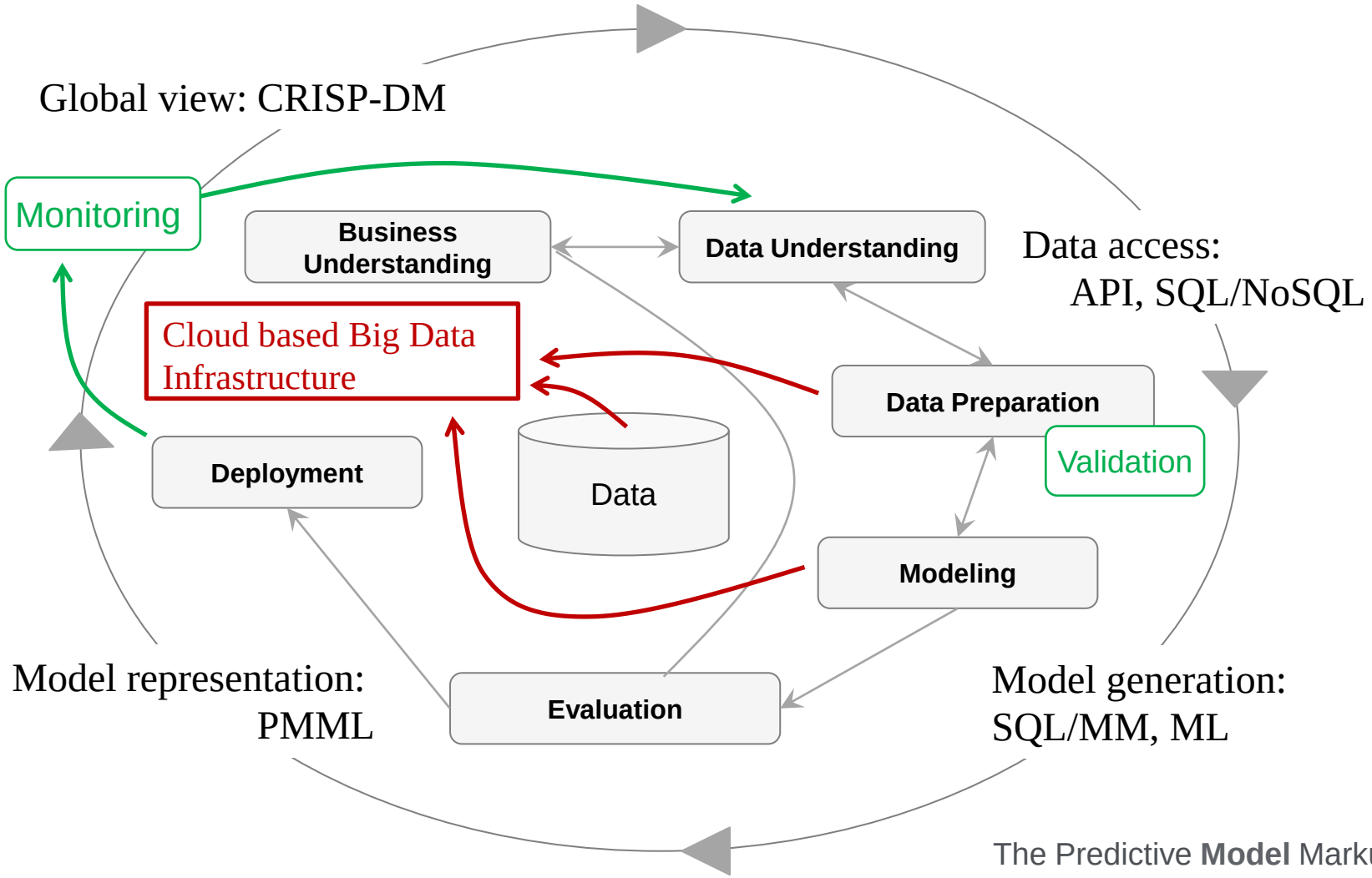
Cross Industry Standard Process for Data Mining (CRISP-DM) model and stages

- Business understanding
- Data Understanding
- Data preparation
- Modelling
- Evaluation
- Deployment

All stages are iterative with the goal to achieve effectiveness for business decision making

The Predictive **Model** Markup Language (**PMML**)

CRISP DM: Processes and Data Lifecycle – Enriched (2)



Cross Industry Standard Process for Data Mining (CRISP-DM) model and stages

- Business understanding
- Data Understanding
- Data preparation
  - Data Validation
- Modelling
- Evaluation
- Deployment
  - Process monitoring

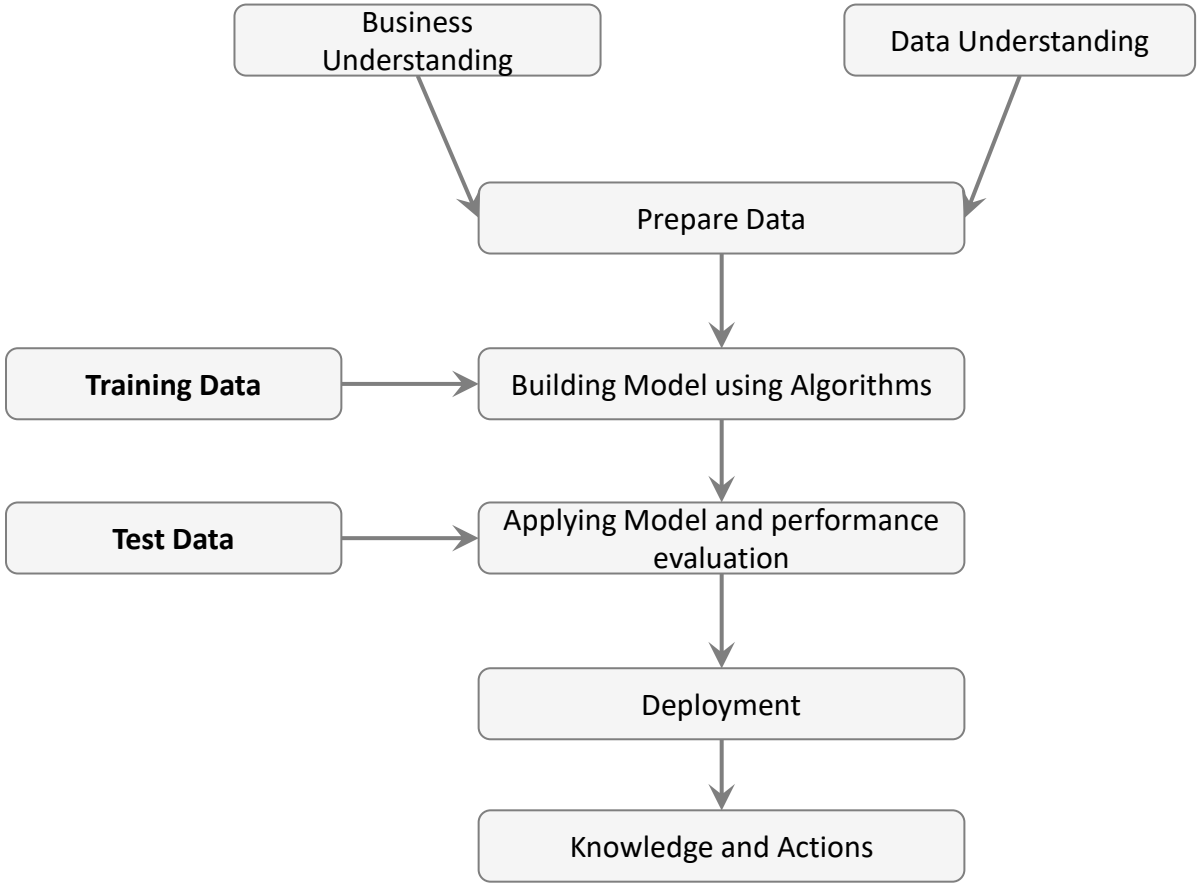
All stages are iterative with the goal to achieve effectiveness for business decision making

The Predictive **Model** Markup Language (**PMML**)

CRISP-DM Four Levels of Abstraction

- Phases/Stages
  - Core elements of the CRISP-DM model
- Generic Tasks
  - A stable, general and complete set of tasks
  - Example: Data Cleaning
- Specialized Task
  - How is the generic task carried out
  - Example: Missing Value Handling
- Process Instance
  - Example: The mean value for numeric attributes and the most frequent for categorical attributes was used

Process of Data Analysis (based on CRISP-DM)

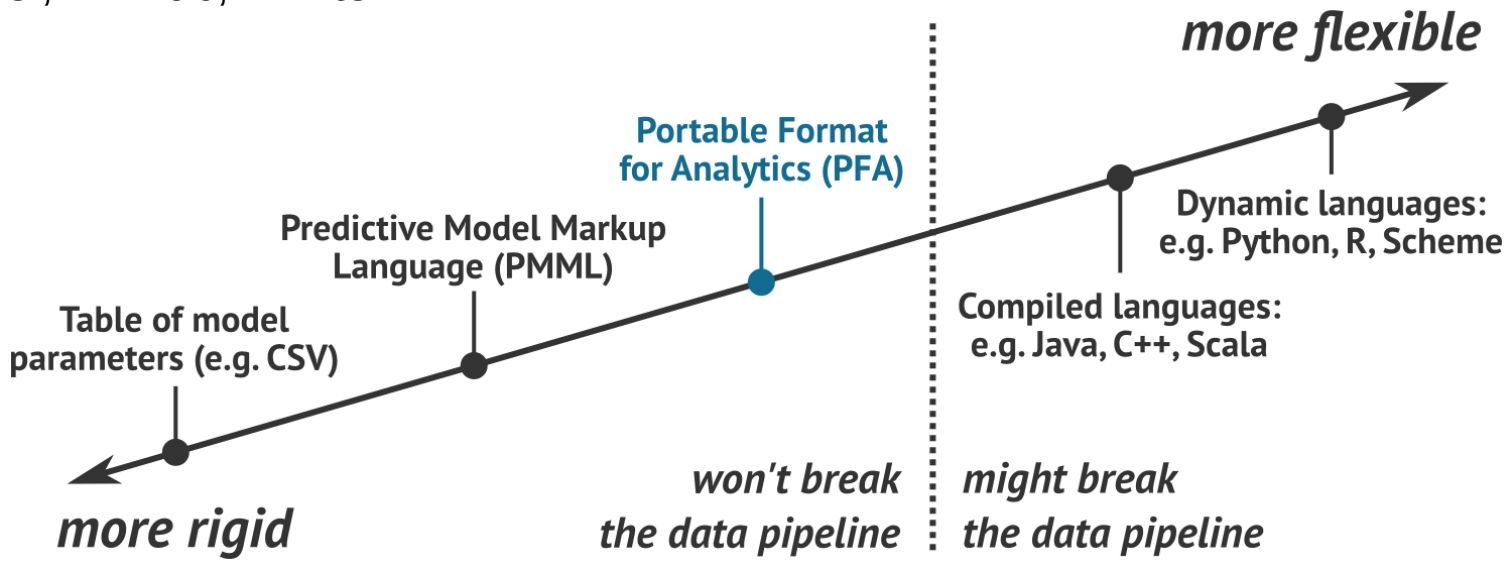


1. Prior Knowledge
2. Preparation
3. Modeling
4. Application
5. New Knowledge

- However the process is not linear, repeatedly backtracking

Data Analytics Models Formats – Step from Dev to Ops via CI/CD

- **Predictive Models Markup Language (PMML)** that have benefits of transferring a developed model to production, access to coefficients
- **Portable Format for Analytics (PFA)**, an emerging standard for statistical models and data transformation engines to ease portability across systems with algorithmic flexibility by defining composable models, pre-processing, and post-processing functions that can be built into complex workflows
- **ONNX** (Open Neural Network Exchange) - an open format built to represent machine learning models. ONNX defines
  - Common set of operators - the building blocks of machine learning and deep learning models, and
  - Common file format to enable AI developers to use models with a variety of frameworks, tools, runtimes, and compilers.
- **TensorFlow Model** format: SavedModel, TF Hub, TFLite



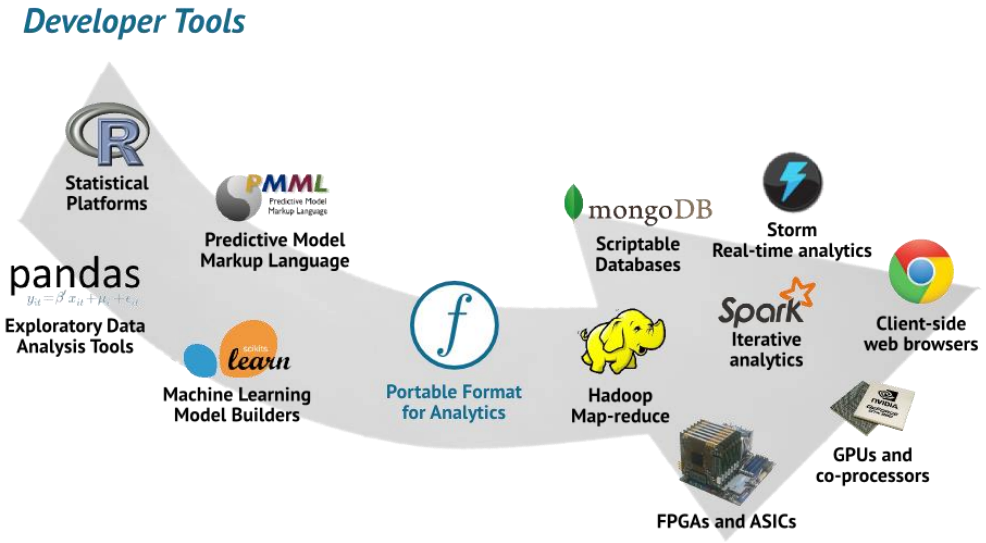
PMML Format

PMML is XML-based. Its schema follows a well-defined structure in which elements and attributes are used to define:
http://dmg.org/pmml/v4-4-1/GeneralStructure.html

- The input data through element DataDictionary
- Invalid, missing and outlier value handling strategies through element MiningSchema
- Data pre-processing via element TransformationsDictionary
- A myriad of modeling techniques via specific model elements such as: NeuralNetwork, TreeModel, SupportVectorMachineModel, Scorecard, and RegressionModel
- Post-processing of model outputs via element Output

Portable Format for Analytics (PFA)

- PFA is an emerging standard for statistical models and data transformation engines.
- PFA combines the ease of portability across systems with algorithmic flexibility: **models, pre-processing, and post-processing** are all functions that can be arbitrarily composed, chained, or built into complex workflows.
- PFA encapsulates a unit of data processing called a **scoring engine**.
- It provides a **common interface to safely deploy analytic workflows** across environments, from embedded systems to distributed data centers.
- PFA may be as simple as a raw data transformation or as sophisticated as a suite of concurrent data mining models, all described as a JSON or YAML configuration file.
- A PFA document is a JSON document with additional constraints. The JSON content describes **algorithms, data types, model parameters**, and other aspects of the scoring engine.




```
{"input": "string",
"output": {"type": "array", "items":"string"},
"cells": {
"accumulate": {"type": {"type": "array",
"items": "string"},
"init": []}},
"method": "map",
"begin":
{"log": {"rand.gaussian": [0.0, 1.0]}},
"action":
{"cell": "accumulate",
"to": {"fcn": "u.addone", "fill": {"newitem":
"input"}}},
"end":
{"log": {"rand.choice": {"cell":
"accumulate"}}},
"fcns":
{"addone":
{"params": [{"old": {"type": "array", "items":
"string"}},
{"newitem": "string"}],
"ret": {"type": "array", "items": "string"},
"do": {"a.append": ["old", "newitem"]}}},
"randseed": 12345,
"name": "ExampleScoringEngine",
"version": 1,
"doc": "Doesn't do much.",
"metadata": {"does": "notmuch"},
"options": {"timeout": 1000}}
```

TensorFlow Model Formats

- tfhub.dev hosts TensorFlow models in the SavedModel format and TF1 Hub format.
- SavedModel is the recommended format for sharing TensorFlow models.
  - You can use SavedModels from tfhub.dev without depending on the tensorflow_hub library, since this format is a part of core TensorFlow.
- TF1 Hub format - custom serialization format used in by TF Hub library.
  - The TF1 Hub format is similar to the SavedModel format of TensorFlow 1 on a syntactic level (same file names and protocol messages) but semantically different to allow for module reuse, composition and re-training (e.g., different storage of resource initializers, different tagging conventions for metagraphs).
- TFLite format - used for on-device inference
- TFJS format - used for in-browser ML

ONNX v1.8 - https://onnx.ai/ https://github.com/onnx

- Designed for interoperability of AI models: **Open Neural Network Exchange (ONNX)**
- ONNX Model Zoo is a collection of pre-trained, state-of-the-art models in the ONNX format
  - Accompanying each model are Jupyter notebooks for model training and running inference with the trained model.
  - The notebooks are written in Python and include links to the training dataset as well as references to the original paper that describes the model architecture.
- ONNX provides a definition of an extensible computation graph model, as well as definitions of built-in operators and standard data types.
- Each computation dataflow graph is structured as a list of nodes that form an acyclic graph. The graph also has metadata to help document its purpose, author, etc.
  - Nodes have one or more inputs and one or more outputs.
  - Each node is a call to an operator.
- Operators are implemented externally to the graph, but the set of built-in operators are portable across frameworks. Every framework supporting ONNX will provide implementations of these operators on the applicable data types.

Vision

- Image Classification
- Object Detection & Image Segmentation
- Body, Face & Gesture Analysis
- Image Manipulation

Language

- Machine Comprehension
- Machine Translation
- Language Modelling

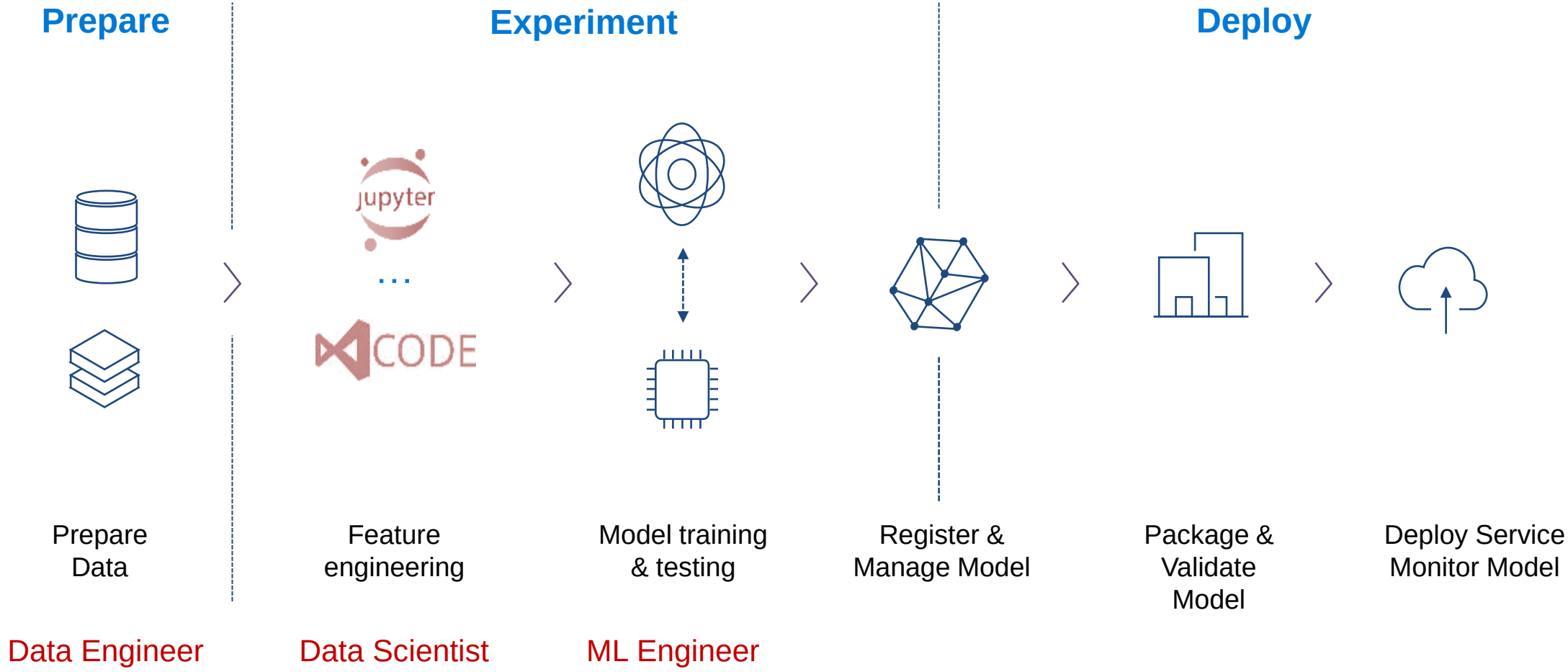
Other

- Visual Question Answering & Dialog
- Speech & Audio Processing
- Other interesting models

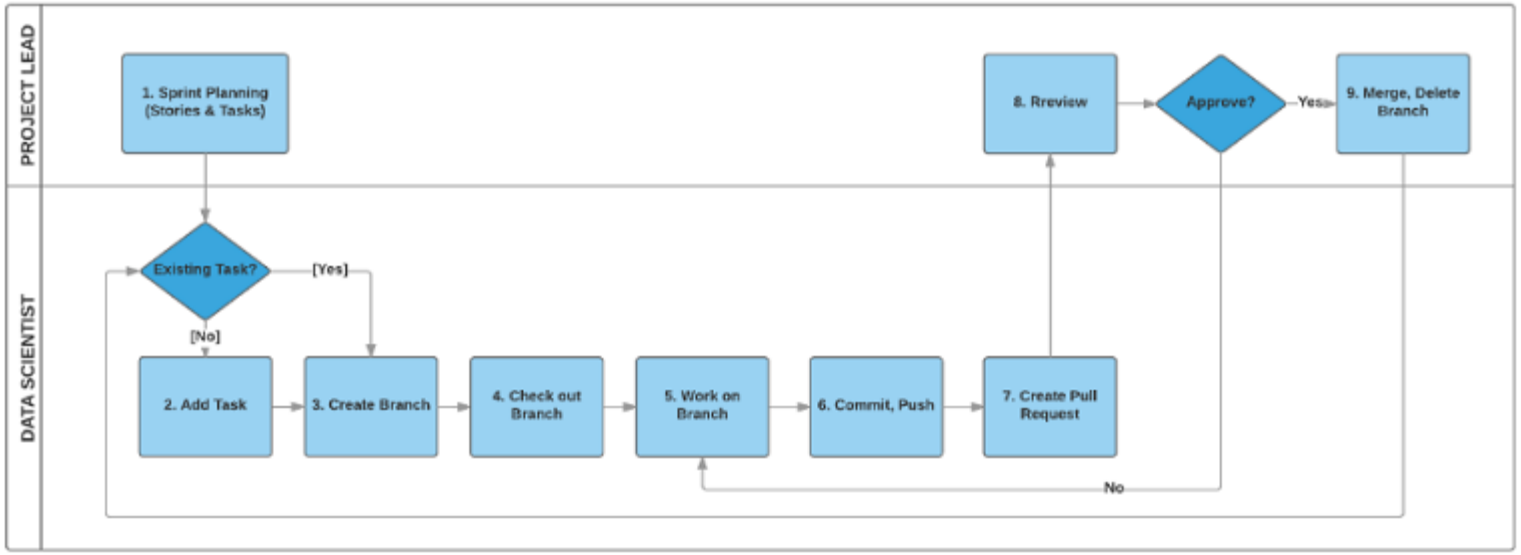
MLOps Frameworks and Platforms

- Example: Azure MLOps Framework
- Example: AWS MLOps Framework

Azure MLOps: Data Science Workflow



Azure MLOps: Execution Data Science Project



Work items:

- Feature
- User Story
- Task
- Bug

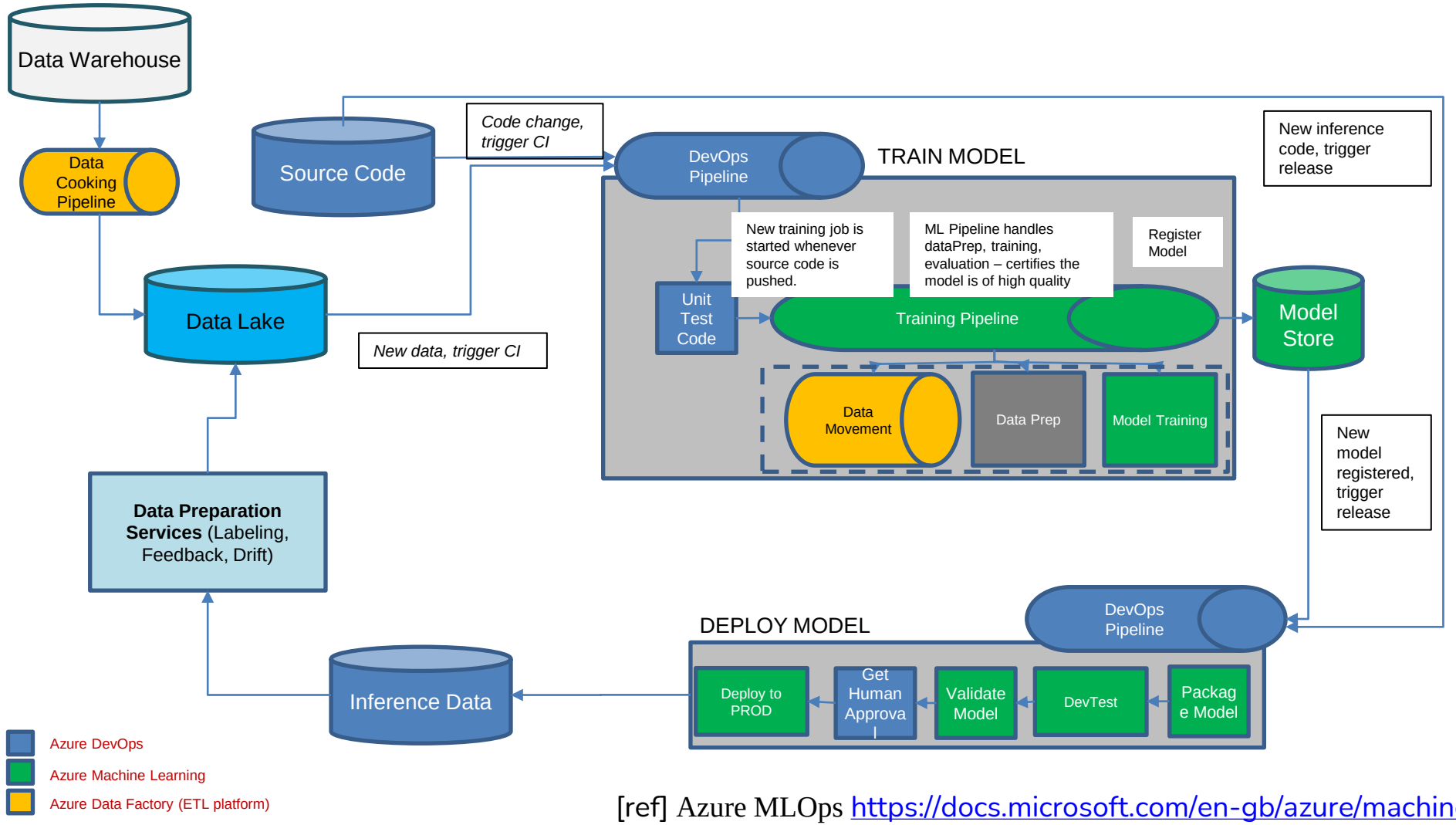
Typical Roles:

- Scrum Master
- Other Scrum/ Kanban roles

Execution steps

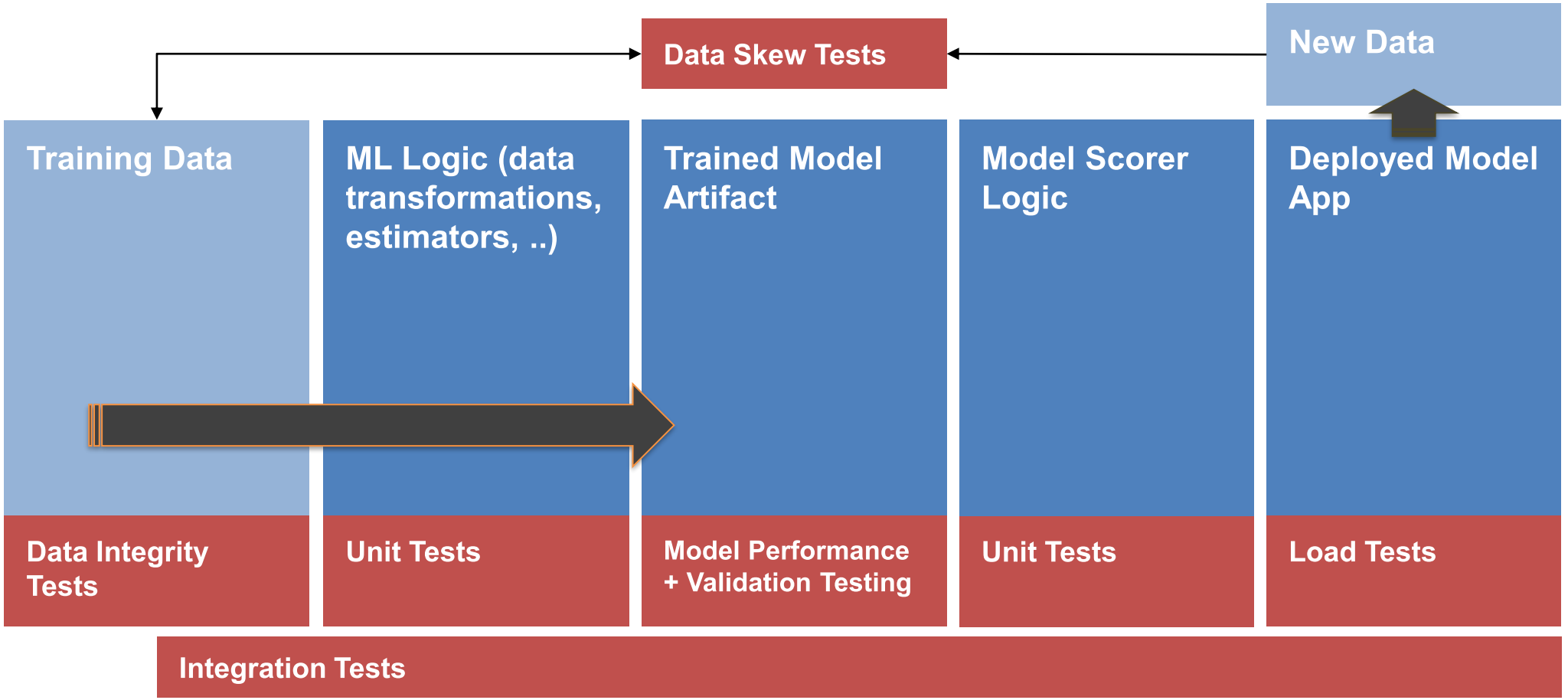
1. **Sprint planning**
2. Add a Feature
3. Add Story under Feature
4. Add a task to a Story
5. Link a work item with a git branch
6. Work on a branch and commit the changes
7. Create a pull request on VSTS
8. Review and merge
9. Data Quality Report Utility
10. Modeling Utility
11. Tracking progress of projects with Power BI dashboards

ML Model CI/CD (Machine Learning as a Service + DevOps)



[ref] Azure MLOps https://docs.microsoft.com/en-gb/azure/machine-learning/

Testing & ML: CI/CD Pipeline for ML process



[ref] Azure MLOps https://docs.microsoft.com/en-gb/azure/machine-learning/

Azure MLOps Pipelines and Azure DevOps

1) Data Pipeline

- Data is extracted from the source system(s), transformed for cleaning and preprocessing, and then loaded into a target data store (data lake).

2) Environment Pipeline

- Ensures that all these dependent libraries are loaded properly and consistently every time before proceeding further in the MLOPs Pipeline.

3) Training Pipeline

- Executes Data Pipelines and load all dependencies to do the actual training of our ML model.

4) Continuous Integration Pipeline

- Each commits undergo the code quality and unit testing

5) Testing Pipeline

- The model created should be tested properly before it can be deployed in live because your model will have to work with the existing production code of the application. So unless your model passes all the testing and validations. Only if testing is successful, the model is allowed for the deployment pipeline.

6) Continuous Deployment Pipeline

- This pipeline takes care of the deployment process.

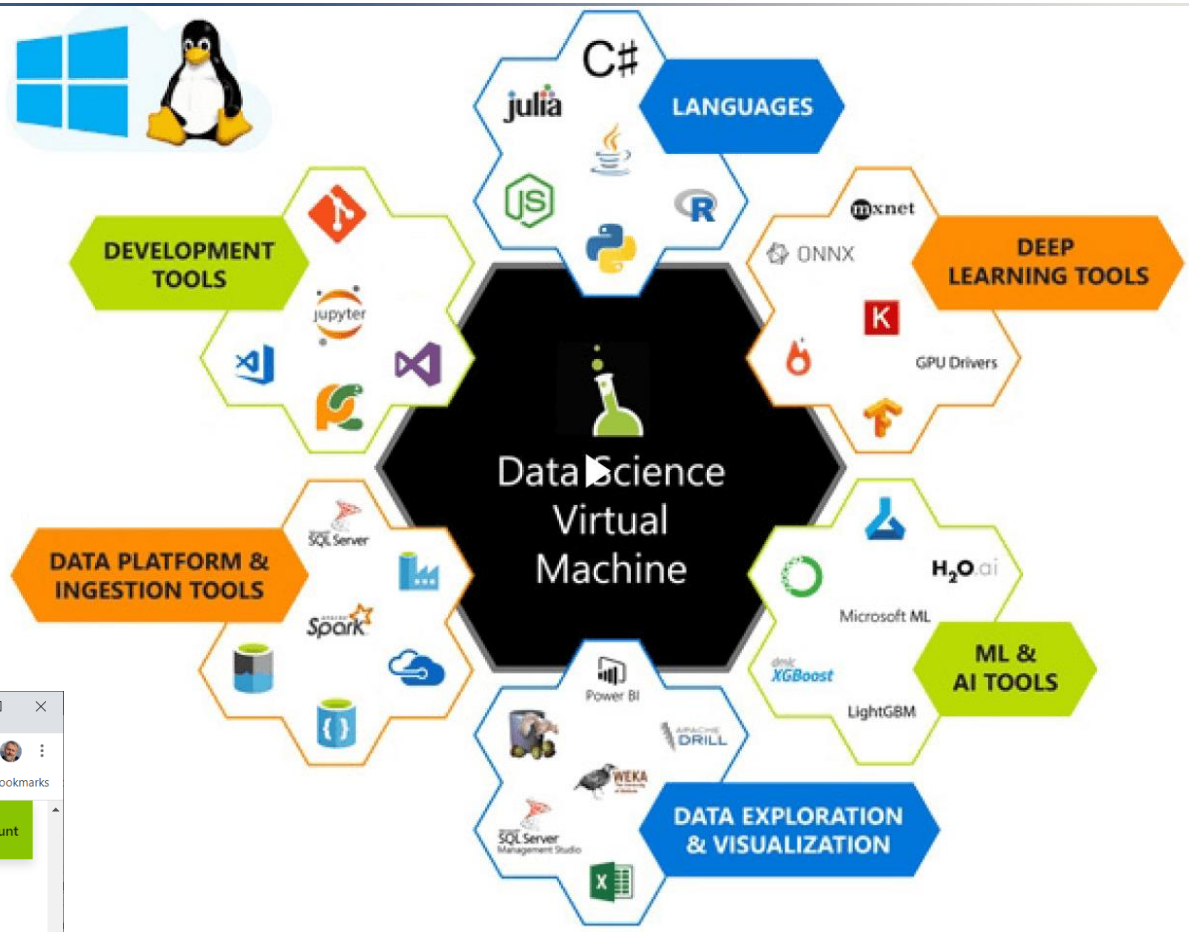
MLOPs with Azure DevOps services

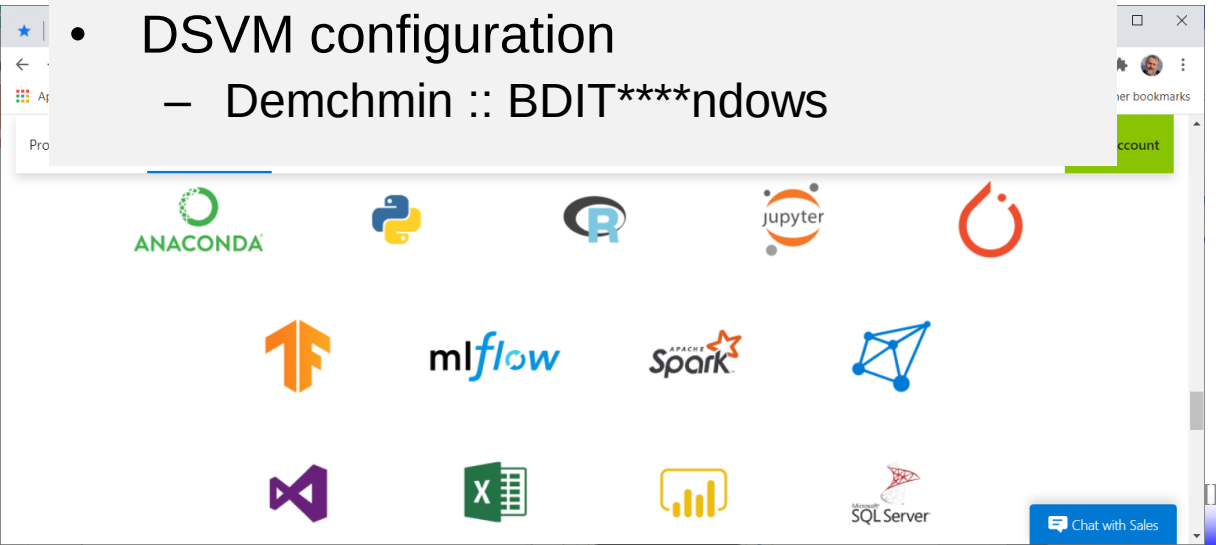
- **Azure Board** for better collaboration within the team and agile planning.
- **Azure Pipelines** for creating CI/CD pipeline architecture.
- **Azure Repos** for git repository to ensure proper version control of the codebase.
- **Azure Artifacts** for managing dependencies in the codebase.
- **Azure Test Plan** for planned and exploratory testing solutions.

Azure – Data Science Virtual Machine (DSVM)

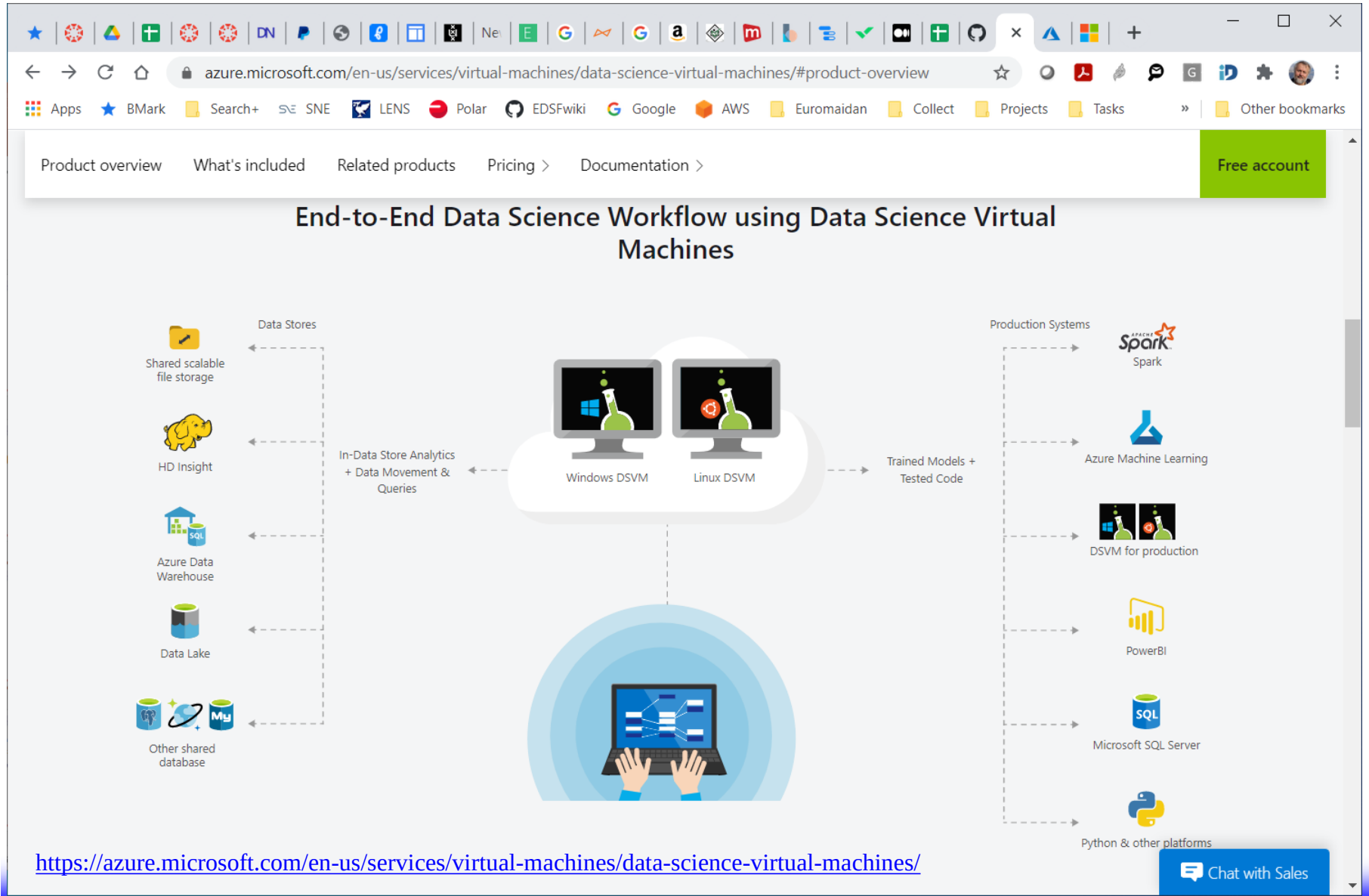
https://azure.microsoft.com/en-us/services/virtual-machines/data-science-virtual-machines/

- Pre-Configured virtual machines in the cloud for Data Science and AI Development
  - Python, R, ONNX, + Microsoft AutoML
  - https://docs.microsoft.com/en-us/azure/machine-learning/data-science-virtual-machine/tools-included
- Standard Configuration Standard_DS3_v2 – 4 vcups – 1 GiB
  - @ 328.12 USD/mo
- Expected to be used as a virtual desktop
- DSVM configuration
  - Demchmin :: BDIT****ndows





End-to-End Data Science Workflow using Azure Data Science Virtual Machines



https://azure.microsoft.com/en-us/services/virtual-machines/data-science-virtual-machines/

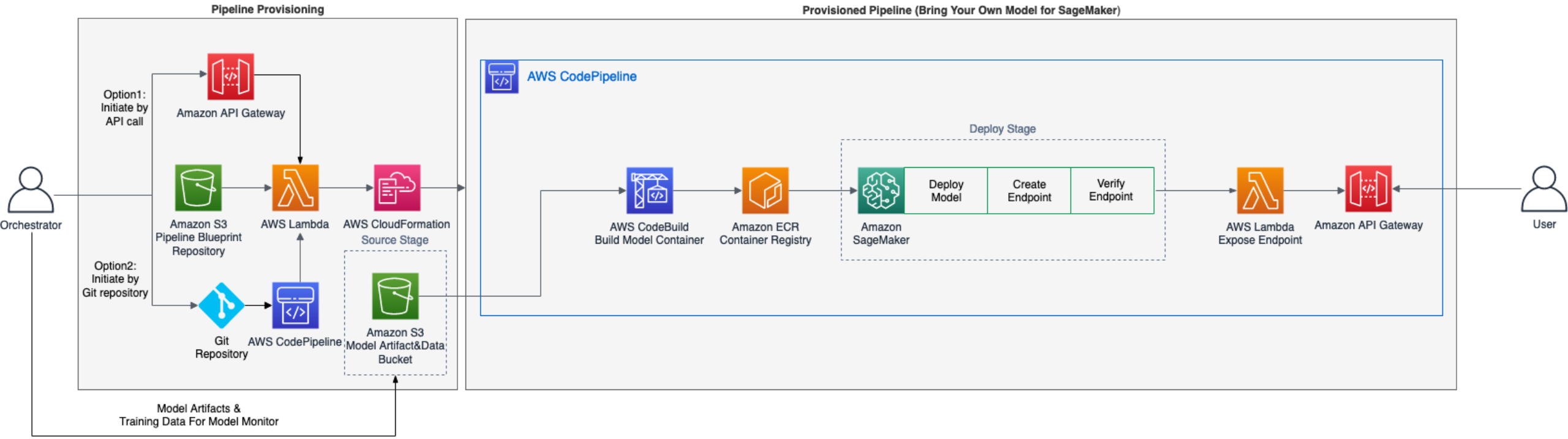
- DSVM connected to data source and to operationalised targets

AWS MLOps Framework Components

- **AWS Sagemaker – Integrated IDE for ML**
  - **Prepare - Build – Train&Tune – Deploy&Manage**
- AWS CodeCommit
- AWS CodePipeline
- AWS CodeBuild
- AWS CodeDeploy

AWS MLOps Framework

https://aws.amazon.com/solutions/implementations/aws-mlops-framework/#



- MLOps pipeline provisioning CodePipeline and SageMaker for Model deployment
  - Using CloudFormation template - https://s3.amazonaws.com/solutions-reference/aws-mlops-framework/latest/aws-mlops-framework.template
  - Source code https://github.com/awslabs/aws-mlops-framework
  - Container based – using ECR
- Configured for using AWS CDK - Code Development Kit (https://aws.amazon.com/cdk/), can also integrates with Terraform
- SageMaker Neo enables machine learning models to train once and run anywhere in the cloud and at the edge

SageMaker Process steps

- **Label** (Prepare)
  Set up and manage labeling jobs for highly accurate training datasets within Amazon SageMaker, using active learning and human labeling
- **Build**
  Connect to other AWS services and transform data in Amazon SageMaker notebooks
- **Train**
  Use Amazon SageMaker's algorithms and frameworks, or bring your own, for distributed training
- **Tune**
  Amazon SageMaker automatically tunes your model by adjusting multiple combinations of algorithm parameters
- **Deploy**
  Once training is completed, models can be deployed to Amazon SageMaker endpoints, for real-time predictions
- **Discover & Grow**
  Find, buy, and deploy ready to use model packages, algorithms, and data products in AWS Marketplace

SageMaker Studio – Jupyter Notebook based

[Ref] https://aws.amazon.com/sagemaker/

AWS Code Development Kit (CDK)

- Defines AWS Infrastructure (IaC) in one of languages: TypeScript, JavaScript, Python, Java, or C#
- Defines one or more stacks. Stacks (equivalent to AWS CloudFormation stacks) contain constructs
  - **AWS CloudFormation-only or L1 (short for "level 1").** These constructs correspond directly to resource types defined by AWS CloudFormation.
    - AWS CloudFormation resources always have names that begin with Cfn. For example, in the Amazon S3 module, CfnBucket is the L1 module for an Amazon S3 bucket.
  - **Curated or L2.** These constructs are carefully developed by the AWS CDK team to address specific use cases and simplify infrastructure development. For the most part, they encapsulate L1 modules, providing sensible defaults and best-practice security policies. For example, in the Amazon S3 module, Bucket is the L2 module for an Amazon S3 bucket.
  - **Patterns or L3.** Patterns declare multiple resources to create entire AWS architectures for particular use cases. All the plumbing is already hooked up, and configuration is boiled down to a few important parameters. In the AWS Construct Library, patterns are in separate modules from L1 and L2 constructs.

SageMaker Model Monitor

- Amazon SageMaker Model Monitor is a capability of Amazon SageMaker that continuously monitors machine learning (ML) models in production, detects deviations such as data drift that can degrade model performance over time, and alerts you to take remedial actions.
- Amazon SageMaker Model Monitor can collect prediction requests and responses from your endpoints, analyze the data collected in production, and compare it against your training or validation data to detect deviations.
- Amazon SageMaker Model Monitor can use built-in rules to detect drift for structured data sets, add data transformations before you run the built-in rules, or write your own custom rules.

SageMaker Neo

- SageMaker Neo enables machine learning models to train once and run anywhere in the cloud and at the edge
- Neo automatically optimizes
  - Gluon, Keras, MXNet, PyTorch, TensorFlow, TensorFlow-Lite, and ONNX models
  - for inference on Android, Linux, and Windows machines based on processors from Ambarella, ARM, Intel, Nvidia, NXP, Qualcomm, Texas Instruments, and Xilinx.
- Neo is tested with computer vision models available in the model zoos across the frameworks.
- SageMaker Neo supports compilation and deployment for two main platforms: cloud instances (including Inferentia) and edge devices.
  - Inferentia instances inf1 improve ML performance up to 30% and 45% lower cost than G4, low latency with 100 Gbps network

Amazon SageMaker Built-in Algorithms

https://docs.aws.amazon.com/sagemaker/latest/dg/algos.html

Supervised Learning

- General purpose algorithms that can be used for either classification or regression problems.
  - Linear Learner Algorithm
  - Factorization Machines Algorithm
  - XGBoost Algorithm
  - K-Nearest Neighbors (k-NN) Algorithm
- Algorithms specialized tasks during feature engineering and forecasting from time series data.
  - Object2Vec Algorithm
  - DeepAR Forecasting Algorithm

Unsupervised Learning: Algorithms for clustering, dimension reduction, pattern recognition, and anomaly detection.

- Principal Component Analysis (PCA) Algorithm
- K-Means Algorithm
- IP Insights
- Random Cut Forest (RCF) Algorithm

Textual Analysis: Algorithms for analysis of textual documents, document classification or summarization, topic modeling or classification, and language transcription or translation.

- BlazingText algorithm
- Sequence-to-Sequence Algorithm
- Latent Dirichlet Allocation (LDA) Algorithm
- Neural Topic Model (NTM) Algorithm

Image Processing: Algorithms for image classification, object detection, and computer vision.

- Image Classification Algorithm
- Semantic Segmentation Algorithm
- Object Detection Algorithm

Infrastructure components to support MLOps pipeline

- Data Lakes
  - Heterogeneous data storage for general Big Data and Data Science Analytics
  - Semantic Azure Data Lake gen 2 storage
- Data Fabrics
  - To support AI/ML process with massive data use and continuous data generation
  - Data fabrics essentially add a semantic layer to data lakes to smooth the process of modeling data infrastructure, reliability and governance
- Azure Data Factory vs AWS Glue vs AWS Data Pipeline
- Big Data computation platform: Hadoop and Spark
- Cloud Computing and Cloud Storage

Data and Data Analytics Applications: Paradigm Shift

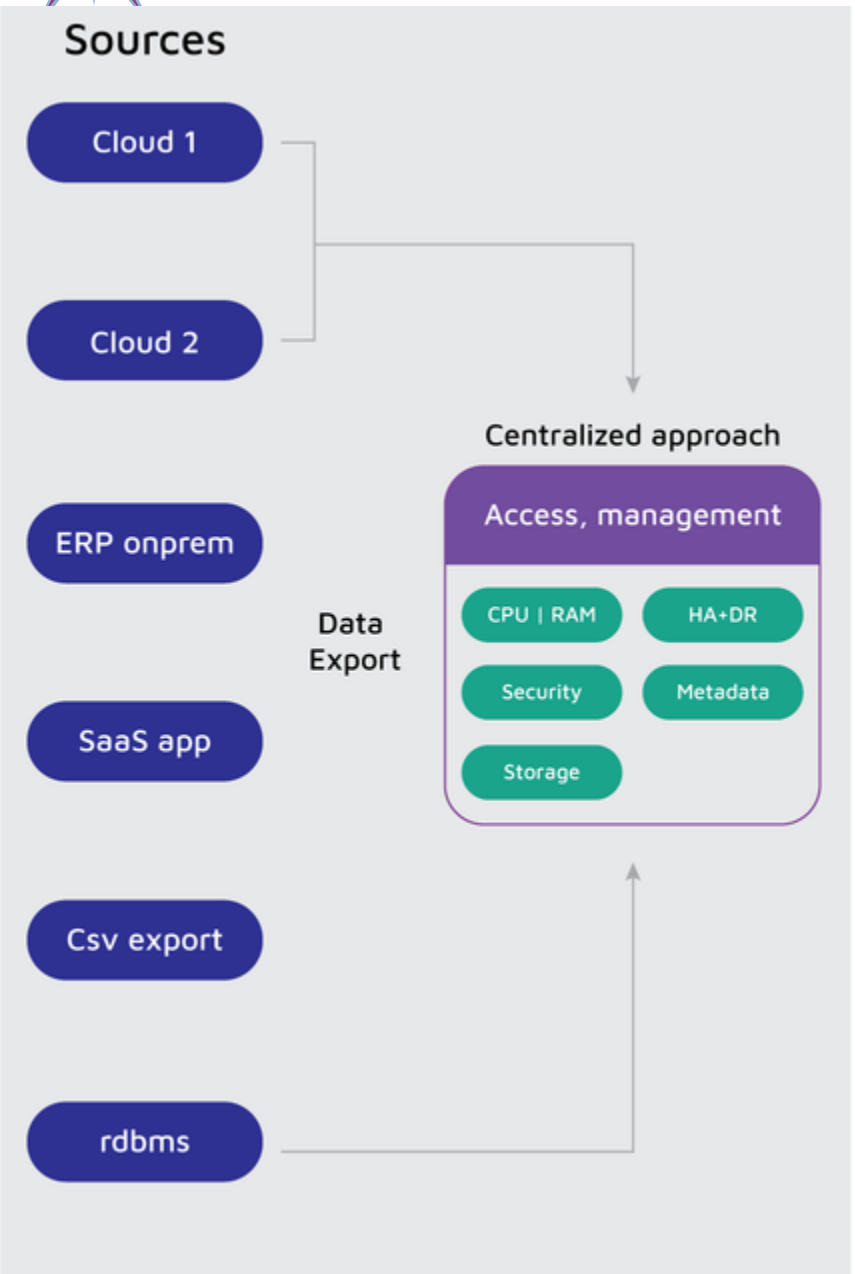
- The concept of a data fabric has emerged to describe a new approach to support agile development of data-driven applications, analytics, and artificial intelligence (AI).

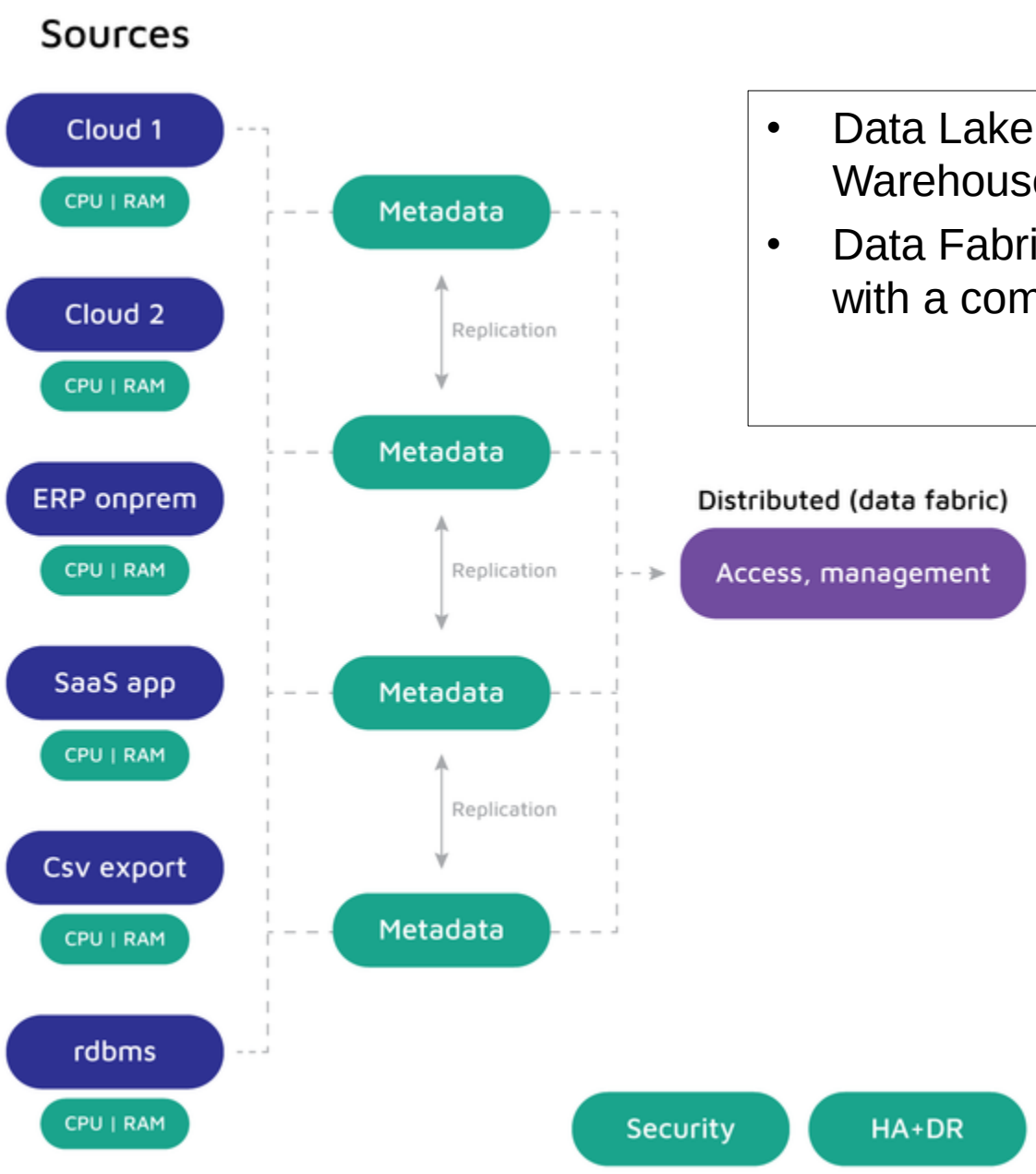
- The traditional approach of having applications dictate how data is organized and stored is going through a major transformation driven by increasing data volumes and the advent of more complex applications, particularly new AI workloads that require large data volumes
  - Semantic layer over Data Lakes

- The term data fabric has emerged to describe a solution to reduce complexity and support agility to address these requirements.
  - There is, however, much confusion in the market created by a proliferation of different approaches all describing themselves as data fabrics.
- Consider the technical capabilities a data fabric must have in order to reduce complexity and enable agility
  - Moving data to compute
  - Moving compute to data
  - Data routing and composing + Streaming

Data Fabric as next Data Management Platform

https://lingarogroup.com/blog/is-data-fabric-the-future-of-data-management-platforms/





- Data Lake as alternative to Data Warehouse
- Data Fabric unites DL and DW with a common semantic layer

12 rules in defining Data Fabrics

- **How data are stored**
  1. Linear scalability
  2. Architected to support scale, performance, and consistency
  3. Distributed metadata in the fabric
- **How data is accessed**
  4. Mixed data access from multiple protocols
  5. Distributed multi tenancy
  6. Global namespace
  7. Integrated data streaming for AI
- **How data are distributed**
  8. Distributed location support
  9. Multi master replication
  10. Location awareness
- **How data are secured**
  11. Capability to serve as a system of record
  12. Data security and governance within the fabric
- **How data quality is assured**
  13. Data quality assurance and model improvement

Azure Data Factory

- Azure Data Factory is a cloud-based data integration service for creating ETL and ELT pipelines
  - Allows creating data-driven workflows for orchestrating and automating data movement and data transformation
  - Full support for CI/CD of data pipelines using Azure DevOps and GitHub.
    - Incrementally develop and deliver ETL processes before publishing the finished product
  - Highly scalable, general purpose integration cloud based data and workflows
  - ADF supports both pre- and post-load transformations. Users apply transformations either by using a GUI to map them, or in code using Power Query Online. Azure Data Factory supports a wide range of transformation functions.
  - ADF integrates with about 80 data sources, including SaaS platforms, SQL and NoSQL databases, generic protocols, and various file types
    - It supports around 20 cloud and on-premises data warehouse and database destinations.

- Provides similar services to SQL Server Integration Services (**SSIS**) which is enterprise ETL tool for applications running on Microsoft SQL Server (see details https://www.timmitchell.net/post/2020/07/16/comparing-ssis-and-azure-data-factory/)

AWS Data Pipeline and AWS Glue

- AWS Data Pipeline is focused on data transfer for moving data from sources to analytics destinations
  - Data Pipeline supports preload transformations using SQL commands
  - Pipeline created graphically through a console, using the AWS CLI with a pipeline definition file in JSON format, or programmatically through API calls
  - Supports four types of data nodes as sources and destinations: DynamoDB, SQL, Redshift tables, and S3 locations.
- AWS Glue is more focused on ETL
  - AWS Glue is a serverless data integration service that makes it easy to discover, prepare, and combine data for analytics, machine learning, and application development.
  - AWS Glue Studio to create, run, monitor ETL workflow
  - AWS Glue automates much of the effort required for data integration.
    - AWS Glue crawls data sources, identifies data formats, and suggests schemas to store your data.
    - Automatically generates the code to run data transformations and loading processes.

Summary and take away

- With rapid development of Data Science Analytics and ML applications demand for agile technologies and processes increases
- DataOps and MLOps is based on applying DevOps principles to Data Science Analytics and ML projects
  - Both domains are well developed and their fusion facilitates operationalization of DSA and AI/ML
- Working with large amount of complex data requires new infrastructure solutions such as Data Lakes and Data Fabrics
  - Imaging future AI/ML appliances that requires “re-charging” not only batteries but also data and models
- Emerging MLOps and DataOps open new professional possibilities for SE and DevOps engineers

Additional References

- From DataOps, DevOps to ModelOps
  https://www.techzine.nl/blogs/data/432735/na-dataops-devops-nu-ook-modelops/
- What is DataOps by IBM - https://www.ibm.com/nl-en/analytics/dataops
- DataOps: Industrializing Data and Analytics Strategies for Streamlining the Delivery of Insights
  https://s3.amazonaws.com/eckerson/content_assets/assets/000/000/195/original/DataOPS.pdf?1534882627
- DataOps: Simplify Managing Complex Data Environments
  https://www.sentryone.com/dataops-overview
- DataOps is NOT Just DevOps for Data
  https://medium.com/data-ops/dataops-is-not-just-devops-for-data-6e03083157b7

Additional Materials

- General Project/Process Management Frameworks
- Data Science Analytics Platforms – Gartner Report 2021
- DataOps Manifesto (historical)
- CRISP-DM detailed tasks by stages

General Project/Process Management Frameworks

- ITIL, ISO and others
- ITIL (Information Technology Infrastructure Library) and PMP (Project Management Professional)
- PMBOK by Project Management Institute (PMI) and Project Life Cycle

ITIL Structure and Process Model

- 5 lifecycle phases (for each phase: one identically named core publication):
  - Service strategy
  - Service design
  - Service transition
  - Service operation
  - Continual service improvement
- Process model consisting of 26 ITSM processes

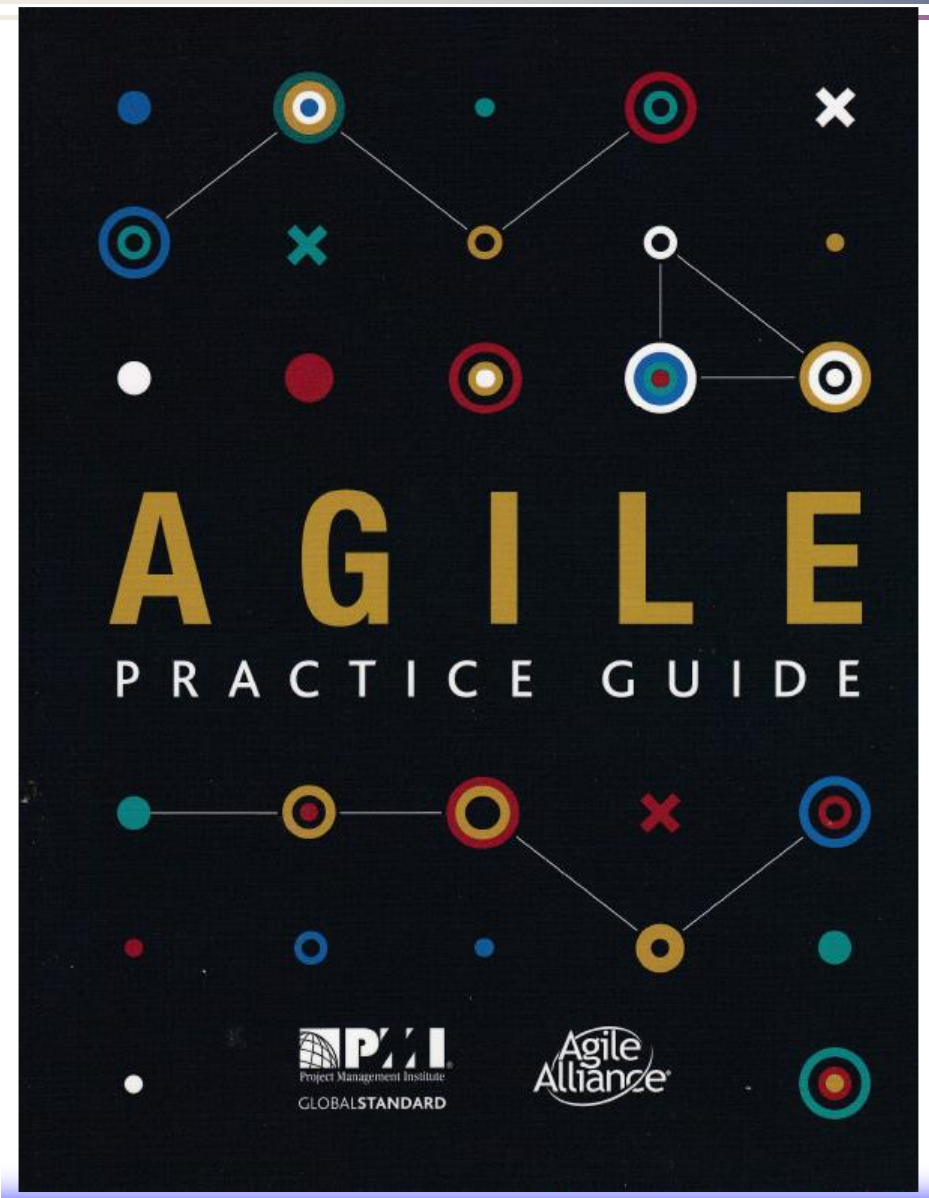
PM BoK - The Five Process Groups/Phases

- *Initiating* - Processes to define and authorize a project or project phase
- *Planning* - Processes to define the project scope, objectives and steps to achieve the required results.
- *Executing* - Processes to complete the work documented within the Project Management Plan.
- *Monitoring and Controlling* - Processes to track and review the project progress and performance. This group contains the Change Management.
- *Closing* - Processes to formalize the project or phase closure.

PMI Planning

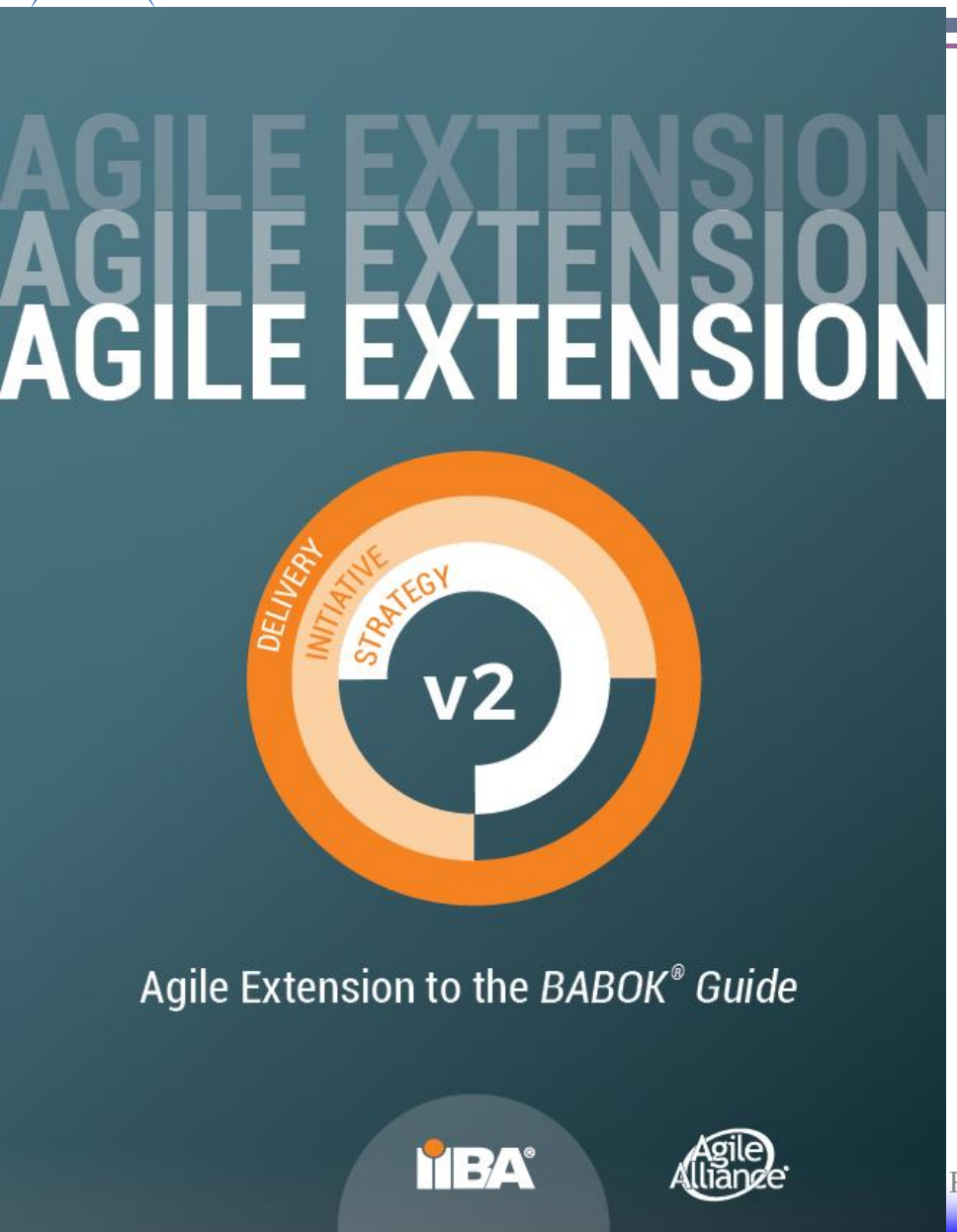
- **Planning phase** is key to successful project management and focuses on developing a roadmap that everyone will follow
- **Result:** Roles and responsibilities are clearly defined. Set of documents created during to ensure the project will stay on track:
  - **Scope Statement** – A document that clearly defines the business need, benefits of the project, objectives, deliverables, and key milestones.
  - **Work Breakdown Schedule (WBS)** – Kind of visual representation that breaks down the scope of the project into manageable sections for the team.
  - **Milestones** – Identify high-level goals that need to be met throughout the project and include them in the Gantt chart.
  - **Gantt Chart** – A visual timeline that to plan out tasks and visualize project timeline.
  - **Communication Plan** – Includes messaging around the project and a schedule of when to communicate with team members based on deliverables and milestones.
  - **Risk Management Plan** – Identify all foreseeable risks. Common risks include unrealistic time and cost estimates, customer review cycle, budget cuts, changing requirements, and lack of committed resources.

PMI Agile Practice Guide



- **An Introduction to Agile** describes the *Agile Manifesto* mindset, values, and principles. It also covers the concepts of definable and high-uncertainty work, and the correlation between the Lean, Kanban, and Agile approaches.
- **Life Cycle Selection** introduces the various life cycles discussed in the practice guide and covers suitability filters, tailoring guidelines, and common combinations of approaches.
- **Implementing Agile: Creating an Agile Environment** talks about critical factors to consider when creating an Agile environment such as servant leadership and team composition.
- **Implementing Agile: Delivering in an Agile Environment** discusses how to organize a team and common practices the team can use for delivering value on a regular basis. It provides examples of empirical measurements for the team and for reporting status.
- **Organizational Considerations for Project Agility** explores organizational factors that impact the use of Agile practices, such as culture, readiness, business practices, and the role of a project management office (PMO).

PMI Agile Extension to the BABOK



Principles of Agile Business Analysis:

- See the Whole
- Think as a Customer
- Analyze to Determine What is Valuable
- Get Real Using Examples
- Understand What is Doable
- Stimulate Collaboration and Continuous Improvement
- Avoid Waste

DevOps Trends to reflect in SE4DevOps courses

- 2020 Upskilling: Enterprise DevOps Skills Report – Part 2
- https://www.thedevopsleader.com/2020-upskilling-enterprise-devops-skills-report/
- **Agile adoption (81%), DevOps adoption (75%)** and ITIL adoption (25%) have grown since 2019 research work
- **Site Reliability Engineering (SRE)** has risen from 10% adoption in 2019 to 15% in 2020.
- Additional philosophies such as **Value Stream Management (19%) and System Thinking (13%)** are also being leveraged.
- DevOps for Data Science and ML/AI projects

DSML Platforms: Gartner Magic Quadrant 2021

Data Science and Machine Learning (DSML) Platform

- Offers a mixture of basic and advanced functionality essential for building DSML solutions (primarily predictive and prescriptive models).
- Supports the incorporation of these solutions into business processes, surrounding infrastructure, products and applications.
- Supports the sustainable consumption of insights derived from the platform, and offers functionality to quantify and track the value of data science projects.
- Supports variously skilled data science professionals ("data scientist" is an inconsistently applied job title and professional distinction) — a DSML platform's user base is often made up of professionals with diverse technical and business backgrounds.
- Supports multiple tasks across the data science life cycle

[ref] Gartner Magic Quadrant 2021

Data Lifecycle Tasks support by DSML platforms

- Problem and business context understanding
- Data ingestion
- Data preparation
- Data exploration
- Feature engineering
- Model creation and training
- Model testing
- Deployment
- Monitoring
- Maintenance
- Data and model governance
- Explainable artificial intelligence (XAI)
- Business value tracking
- Collaboration

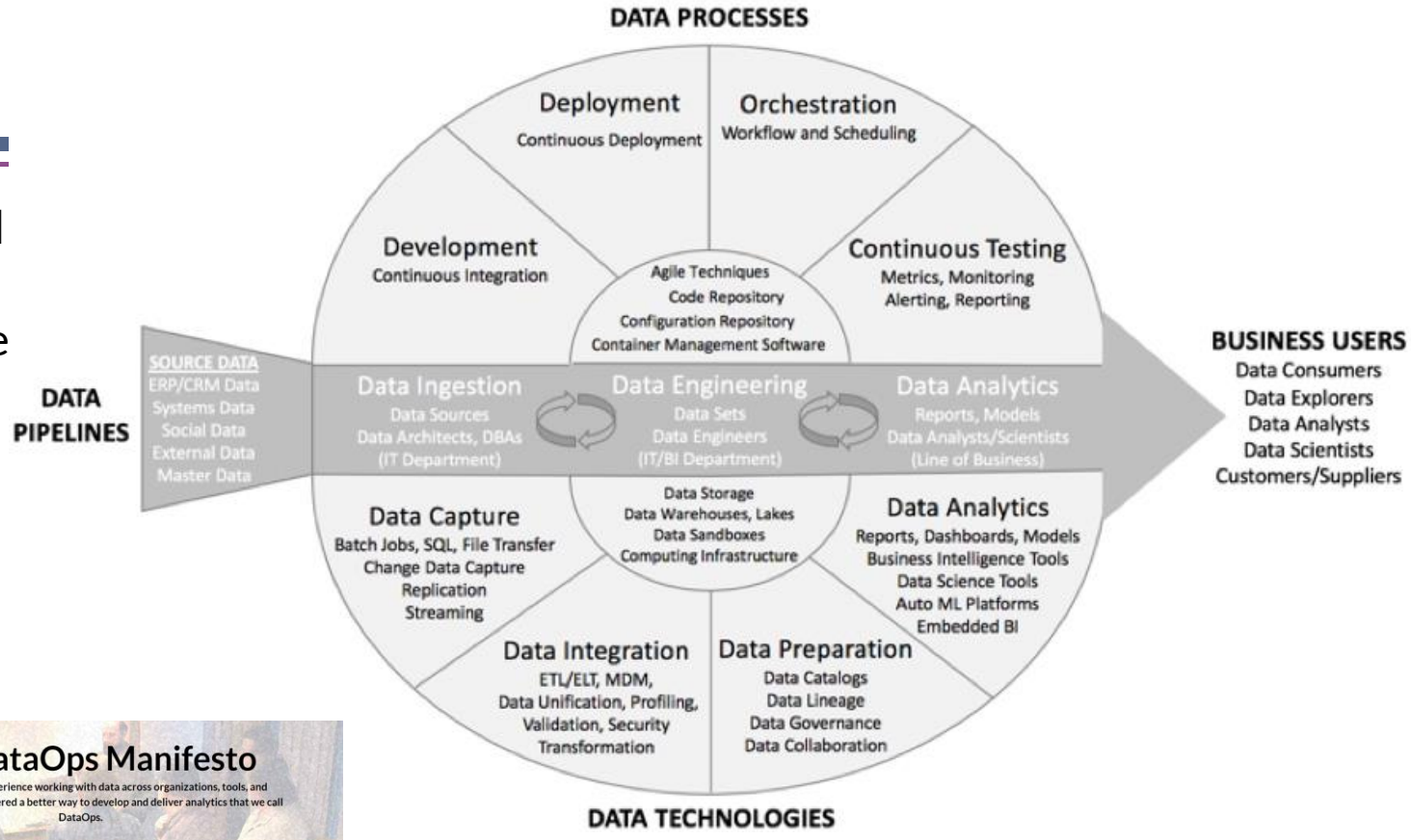
[ref] Gartner Magic Quadrant 2021

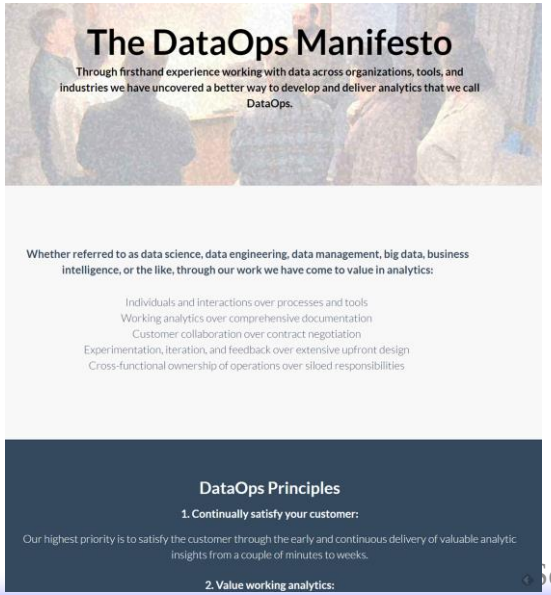
12 rules in defining Data Fabrics - Defined

1. Linear scalability	**The fabric should be able to scale without limits with growing data volumes, number of files, and concurrent client access.**
2. Architected to support scale, performance, and consistency	A data fabric should ensure data consistency locally and provide a simple consistency model for developers to implement across locations. For example, expecting a developer to implement tradeoffs between scale, performance, and consistency is not tenable.
3. Distributed metadata in the fabric	A fabric should allow metadata to be distributed across all data storing nodes to avoid failure points or bottlenecks.
4. Mixed data access from multiple protocols	A fabric needs to provide equal support for disparate data types and access methods. A fabric needs to fully integrate multiple data types, not simply attempt to orchestrate across different underlying datastores to provide links to separate silos of data. The ability to have multiple file and object protocols access and update the same data eliminates ETL functions, allows a broader range of software applications, and delivers low-latency processing for complex applications.
5. Distributed multi-tenancy	To effectively support a wide range of applications and users as well as manage and secure the fabric, organizations need the ability to create logical separations in the fabric for administration, access, update, and execution. The fabric should be able to isolate heavy loads and provide protected resources. It should rebalance automatically based on load changes or after failures.
6. Global namespace	A data fabric needs a global namespace that supports a view of and ability to access data regardless of how it is distributed across physical locations including on-premises, cloud, and edge.
7. Integrated data streaming for AI	The line is blurring between data in motion and data at rest, particularly with IoT and AI applications that need to operate in very small event window to impact the business. A data fabric needs to support integrated streaming to easily ingest and integrate data in motion with data at rest with a common management, security, and analytics framework.
8. Distributed location support	A data fabric must provide the ability to deploy and execute across on-premises, cloud, and edge locations with centralized management and fully integrated functionality across the entire infrastructure. A fabric needs to stretch across locations, not simply have the ability to install and run in different locations.
9. Multi-master replication	A data fabric requires multi-master replication across multiple locations to support distributed operations with transactional integrity.
10. Location awareness	Though a fabric is uniform, it must understand and control the placement of specific data and jobs for cost, performance, and compliance reasons. Distant data should be addressable and accessible from afar as well as replicable locally.
11. Capability to serve as a system of record	A data fabric requires features that prevent data corruption and provide backup and disaster recovery capabilities. Without these, a fabric is only appropriate for a narrow set of uses that are not impacted by data loss.
12. Data security and governance within the fabric	Data must be secured within the fabric and not be a function of the access method. As applications and access methods expand, security becomes more complex and open to compromises if it is not secured at the lowest level of the fabric.

DataOps Manifesto

- The DataOps Manifesto: Through firsthand experience working with data across organizations, tools, and industries we have uncovered a better way to develop and deliver analytics that we call DataOps.
  - https://www.dataopsmanifesto.org/
- The Ultimate Guide to DataOps: Product Evaluation Criteria and Selection
  - https://www.truedataops.org/





[ref] https://research.eckerson.com/ultimate-guide-to-dataops

DataOps Manifesto: DataOps Principles

1. Continually satisfy your customer:
2. Value working analytics:
3. Embrace change:
4. It's a team sport:
5. Daily interactions:
6. Self-organize:
7. Reduce heroism:
8. Reflect:
9. Analytics is code:
10. Orchestrate:
11. Make it reproducible:
12. Disposable environments:
13. Simplicity:
14. Analytics is manufacturing:
15. Quality is paramount:
16. Monitor quality and performance:
17. Reuse:
18. Improve cycle times:

CRISP-DM Phases - Details

- Business understanding
- Data Understanding
- Data preparation
- Modelling
- Evaluation
- Deployment

Business Understanding Phase - Objectives

Determining Business Objectives

- 1. Gather background information
  - Compiling the business background
  - Defining business objectives
  - Business success criteria
- 2. Assessing the situation
  - Resource Inventory
  - Requirements, Assumptions, and Constraints
  - Risks and Contingencies
  - Cost/Benefit Analysis
- 4. Determining data science goals
  - Data science goals
  - Data science success criteria
- 5. Producing a Project Plan

Business Understanding Phase

- Understand the business objectives
  - What is the status quo?
    - Understand business processes
    - Associated costs/pain
  - Define the success criteria
  - Develop a glossary of terms: speak the language
  - Cost/Benefit Analysis
- Current Systems Assessment
  - Identify the key actors
    - Minimum: The Sponsor and the Key User
  - What forms should the output take?
  - Integration of output with existing technology landscape
  - Understand market norms and standards
- Task Decomposition
  - Break down the objective into sub-tasks
  - Map sub-tasks to data mining problem definitions
- Identify Constraints
  - Resources
  - Law e.g. Data Protection
- Build a project plan
  - List assumptions and risk (technical/financial/business/ organisational) factors

Data Understanding Phase

- **Data Understanding**
  - Proceeds with activities aimed at:
    - Getting familiar with the data
    - Identifying data quality problems
    - Discovering first insights into the data
    - Detecting interesting subsets to form hypotheses for hidden information
- Collect Data
  - What are the data sources?
    - Internal and External Sources (e.g. Axiom, Experian)
    - Document reasons for inclusion/exclusions
    - Depend on a domain expert
    - Accessibility issues
      - Legal and technical
  - Are there issues regarding data distribution across different databases/legacy systems
    - Where are the disconnects?

- Data Description
  - Document data quality issues
    - requirements for data preparation
  - Compute basic statistics

- Data Exploration
  - Simple univariate data plots/distributions
  - Investigate attribute interactions
  - Data Quality Issues
    - Missing Values
      - Understand its source: Missing vs Null values
    - Strange Distributions

Data Preparation Phase

- **Data Preparation**
  - Covers all activities to construct the final dataset (data that will be fed into the modeling tool(s)) from the initial raw data
  - Data preparation tasks are likely to be performed multiple times, and not in any prescribed order
  - Tasks include table, record, and attribute selection, as well as transformation and cleaning of data for modeling tools
- Integrate Data
  - Joining multiple data tables
  - Summarisation/aggregation of data
- Select Data
  - Attribute subset selection
    - Rationale for Inclusion/Exclusion
  - Data sampling
    - Training/Validation and Test sets
- Data Transformation
  - Using standard functions for data transformation
  - Factor/Principal Components analysis
  - Normalization/Discretisation/Binarisation
- Clean Data
  - Handling missing values/Outliers
- Data Construction
  - Derived Attributes

The Modelling Phase

- Selection of the appropriate modelling technique
  - Data pre-processing implications
    - Attribute independence
    - Data types/Normalisation/Distributions
  - Dependent on
    - Data mining problem type
    - Output requirements
- Develop a testing regime
  - Sampling
    - Verify samples have similar characteristics and are representative of the population
- Build Model
  - Choose initial parameter settings
  - Study model behaviour
    - Sensitivity analysis
- Assess the model
  - Beware of over-fitting
  - Investigate the error distribution
    - Identify segments of the state space where the model is less effective
  - Iteratively adjust parameter settings
    - Document reasons of these changes

The Evaluation Phase

- **Evaluation**
  - At this stage, a model (or models) that appears to have high quality, from a data analysis perspective, has been built
  - Before proceeding to final deployment of the model, it is important to more thoroughly evaluate the model, and review the steps executed to construct the model, to be certain it properly achieves the business objectives
  - A key objective is to determine if there is some important business issue that has not been sufficiently considered
  - At the end of this phase, a decision on the use of the data mining results should be reached

- Validate Model
  - Human evaluation of results by domain experts
  - Evaluate usefulness of results from business perspective
    - Define control groups
    - Calculate lift curves
    - Expected Return on Investment
- Review Process
- Determine next steps
  - Potential for deployment
  - Deployment architecture
  - Metrics for success of deployment

The Deployment Phase (1)

- Creation of the model is generally not the end of the project
- Even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it
- Depending on the requirements, the deployment phase can be as simple as generating a report or as complex as implementing a repeatable data mining process
- In many cases it will be the customer, not the data analyst, who will carry out the deployment steps
- However, even if the analyst will not carry out the deployment effort it is important for the customer to understand up front what actions will need to be carried out in order to actually make use of the created models

The Deployment Phase (2)

- Knowledge Deployment is specific to objectives
  - Knowledge Presentation
  - Deployment within Scoring Engines and Integration with the current IT infrastructure
    - Automated pre-processing of live data feeds
    - XML interfaces to 3rd party tools
  - Generation of a report
    - Online/Offline
  - Monitoring and evaluation of effectiveness
- Process deployment/production
- Produce final project report
  - Document everything along the way