Summary of "Machine Learning Operations (MLOps): Overview, Definition, and Architecture"

Yusuke Minami

"Machine Learning Operations (MLOps): Overview, Definition, and Architecture"

8 Sections

1. Introduction
2. Foundations of DevOps
3. Methodology
4. Results
5. Architecture and Workflow
6. Conceptualization
7. Open Challenges
8. Conclusion

[1. Introduction]

ML community has focused extensively on the building of ML models, but not on
(a) building production-ready ML products
(b) automate & operate in a real-world setting

To address these issues, we explore MLOps, the emerging ML engineering practice.
What is MLOps?
To answer this question, we researched:

(a) principles
(b) components
(c) roles
(d) architecture

[2. Foundations of DevOps]

- emerged in the years 2008/2009 and aims to reduce issues in software development
- the goal of eliminating the gap between development and operations
- ensures automation with continuous integration, continuous delivery, and continuous deployment (CI/CD), thus allowing for fast, frequent, and reliable releases.
- designed to ensure continuous testing, quality assurance, continuous monitoring, logging, and feedback loops.

[2. Foundations of DevOps] 6 groups of DevOps tools

- collaboration and knowledge sharing (e.g., Slack, Trello, GitLab wiki)
- source code management (e.g., GitHub, GitLab)
- build process (e.g., Maven)
- continuous integration (e.g., Jenkins, GitLab CI),
- deployment automation (e.g., Kubernetes, Docker),
- monitoring and logging (e.g., Prometheus, Logstash)

[3. Methodology]

1. Literature review
2. Tools
3. Interviews

[3. Methodology] 1. Literature review

27 peer-reviewed articles found by search query:

- "Machine Learning" AND either of:
  - "DevOps"
  - "CICD"
  - "Continuous Integration"
  - "Continuous Delivery"
  - "Continuous Deployment"
- "MLOps"
- "CD4ML"

[3. Methodology] 2. Tools

11 tools

Appendix

Table 1. List of evaluated technologies

	Technology Name	Description	Sources
Open-source examples	TensorFlow Extended	TensorFlow Extended (TFX) is a configuration framework providing libraries for each of the tasks of an end-to-end ML pipeline. Examples are data validation, data distribution checks, model training, and model serving.	[7,10,26,46] [δ, θ]
	Airflow	Airflow is a task and workflow orchestration tool, which can also be used for ML workflow orchestration. It is also used for orchestrating data engineering jobs. Tasks are executed according to directed acyclic graphs (DAGs).	[26,40,41] [α, β, ζ, η]
	Kubeflow	Kubeflow is a Kubernetes-based end-to-end ML platform. Each Kubeflow component is wrapped into a container and orchestrated by Kubernetes. Also, each task of an ML workflow pipeline is handled with one container.	[26,35,40,41,46] [α, β, γ, δ, ζ, η, θ]
	MLflow	MLflow is an ML platform that allows for the management of the ML lifecycle end-to-end. It provides an advanced experiment tracking functionality, a model registry, and model serving component.	[11,32,35] [α, γ, ε, ζ, η, θ]
Commercial examples	Databricks managed MLflow	The Databricks platform offers managed services based on other cloud providers' infrastructure, e.g., managed MLflow.	[26,32,35,40] [α, ζ]
	Amazon CodePipeline	Amazon CodePipeline is a CI/CD automation tool to facilitate the build, test, and delivery steps. It also allows one to schedule and manage the different stages of an ML pipeline.	[18] [γ]
	Amazon SageMaker	With SageMaker, Amazon AWS offers an end-to-end ML platform. It provides, out-of-the-box, a feature store, orchestration with SageMaker Pipelines, and model serving with SageMaker endpoints.	[7,11,18,24,35] [α, β, γ, ζ, θ]
	Azure DevOps Pipelines	Azure DevOps Pipelines is a CI/CD automation tool to facilitate the build, test, and delivery steps. It also allows one to schedule and manage the different stages of an ML pipeline.	[18,42] [γ, ε]
	Azure ML	Microsoft Azure offers, in combination with Azure DevOps Pipelines and Azure ML, an end-to-end ML platform.	[6,24,25,35,42] [α, γ, ε, ζ, η, θ]

[3. Methodology] 2. Tools (cont'd)

	GCP - Vertex AI	GCP offers, along with Vertex AI, a fully managed end-to-end platform. In addition, they offer a managed Kubernetes cluster with Kubeflow as a service.	[25,35,40,41] [α, γ, δ, ζ, θ]
	IBM Cloud Pak for Data (IBM Watson Studio)	IBM Cloud Pak for Data combines a list of software in a package that offers data and ML capabilities.	[41] [γ]

[3. Methodology] 3. Interviews

8 professionals found in Linkedin

Table 2. List of interview partners

Interviewee pseudonym	Job Title	Years of experience with DevOps	Years of experience with ML	Industry	Company Size (number of employees)
Alpha (α)	Senior Data Platform Engineer	3	4	Sporting Goods / Retail	60,000
Beta (β)	Solution architect / Specialist for ML and AI	6	10	IT Services / Cloud Provider / Cloud Computing	25,000
Gamma (γ)	AI Architect / Consultant	5	7	Cloud Provider	350,000
Delta (δ)	Technical Marketing & Manager in ML / AI	10	5	Cloud Provider	139,995
Epsilon (ε)	Technical Architect - Data & AI	1	2	Cloud Provider	160,000
Zeta (ζ)	ML engineering Consultant	5	6	Consulting Company	569,000
Eta (η)	Engineering Manager in AI / Senior Deep Learning Engineer	10	10	Conglomerate (multi-industry)	400,000
Theta (θ)	ML Platform Product Lead	8	10	Music / audio streaming	6,500

[4. Results] Principles within Components

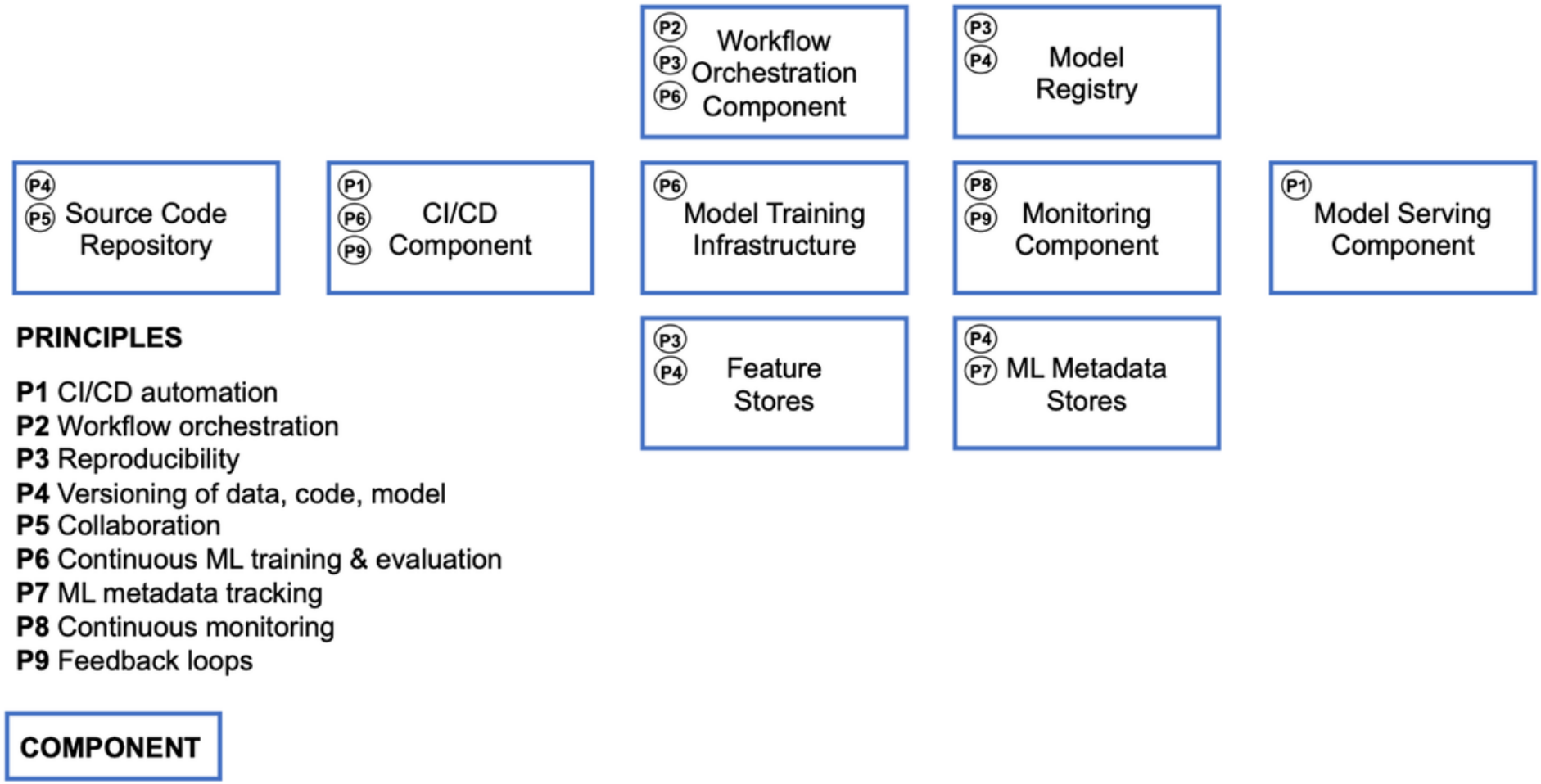


Figure 2. Implementation of principles within technical components

[4. Results] 1. Principles

- P1. CI/CD automation: continuous integration/delivery/deployment
- P2. Workflow orchestration: coordinate tasks based on relationships & dependencies
- P3. Reproducibility: obtain the exact same results
- P4. Versioning of data, model, and code
- P5. Collaboration: work collaboratively on data, model, and code
- P6. Continuous ML training & evaluation: retraining of ML models based on new data
- P7. ML metadata tracking/logging: used parameters, the resulting performance metrics, data and code used
- P8. Continuous monitoring: infrastructure resources and model serving performance
- P9. Feedback loops: e.g. model engineering to feature engineering, monitoring to retraining

[4. Results] 2. Components

- C1. CI/CD Component (P1, P6, P9) e.g. Jenkins, GitHub actions
- C2. Source Code Repository (P4, P5) e.g. Bitbucket, GitLab, GitHub, Gitea
- C3. Workflow Orchestration Component (P2, P3, P6) e.g. Apache Airflow, Kubeflow Pipelines, Luigi, AWS SageMaker Pipelines, Azure Pipelines
- C4. Feature Store System (P3, P4) e.g. Google Feast, Amazon AWS Feature Store
- C5. Model Training Infrastructure (P6) e.g. Kubernetes, Red Hat OpenShift
- C6. Model Registry (P3, P4) e.g. MLflow, AWS SageMaker Model Registry, Microsoft Azure ML Model Registry, Nepture.ai
- C7. ML Metadata Stores (P4, P7) e.g. Kubeflow Pipelines, AWS SageMaker Pipelines, Azure ML, IBM Watson Studio, MLflow
- C8. Model Serving Component (P1) e.g. KServing, TensorFlow Serving, Seldion.io
- C9. Monitoring Component (P8, P9) e.g. Prometheus with Grafana, ELK stack, TensorBoard, Kubeflow, MLflow, AWS SageMaker model monitor, cloud watch

[4. Results] 3. Roles

- R1. Business Stakeholder (similar roles: Product Owner, Project Manager)
- R2. Solution Architect (similar role: IT Architect)
- R3. Data Scientist (similar roles: ML Specialist, ML Developer)
- R4. Data Engineer (similar role: DataOps Engineer)
- R5. Software Engineer
- R6. DevOps Engineer
- R7. ML Engineer/MLOps Engineer

[4. Results] 3. Roles around ML Engineer/MLOps Engineer

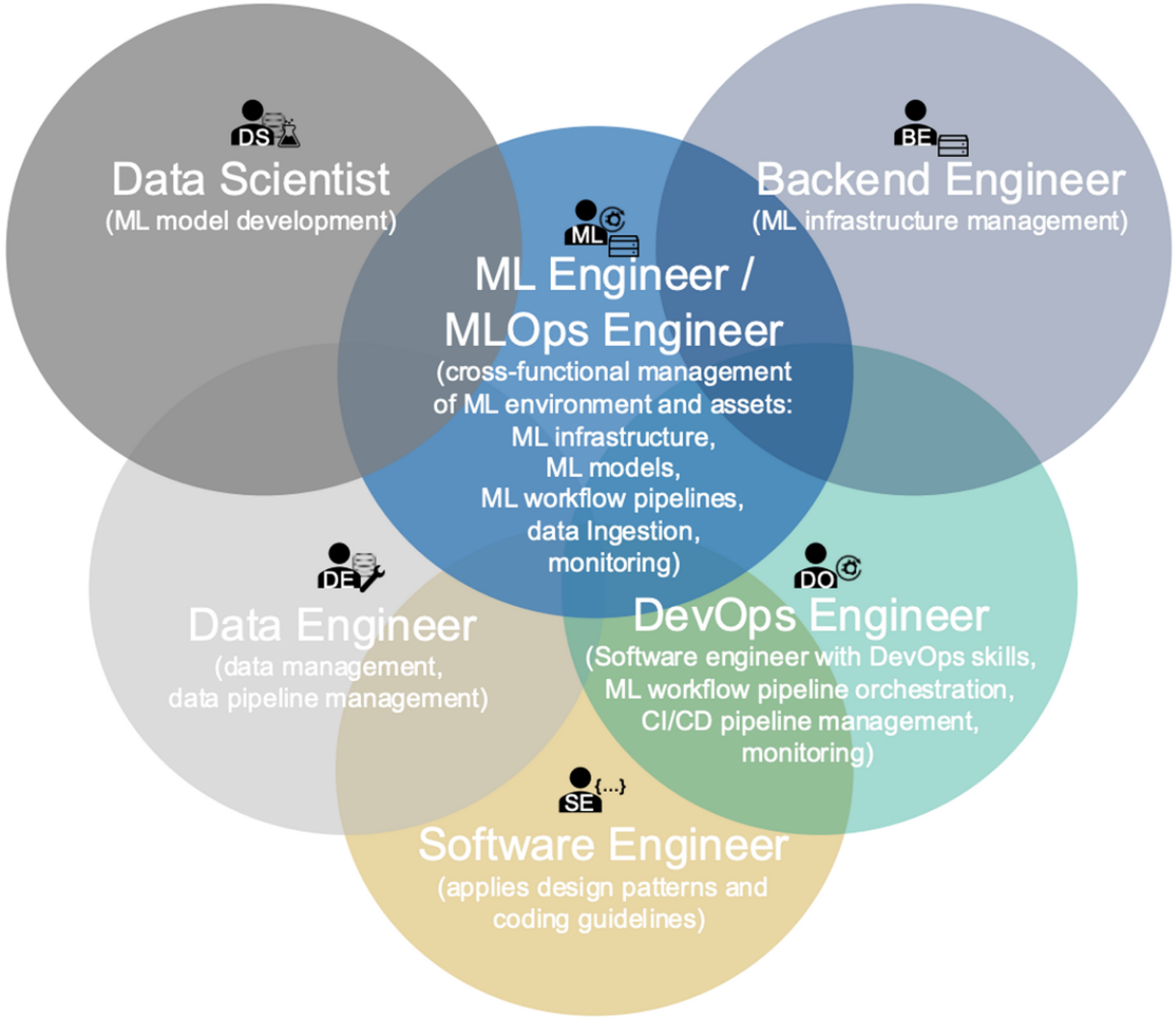


Figure 3. Roles and their intersections contributing to the MLOps paradigm

[5. Architecture and Workflow]

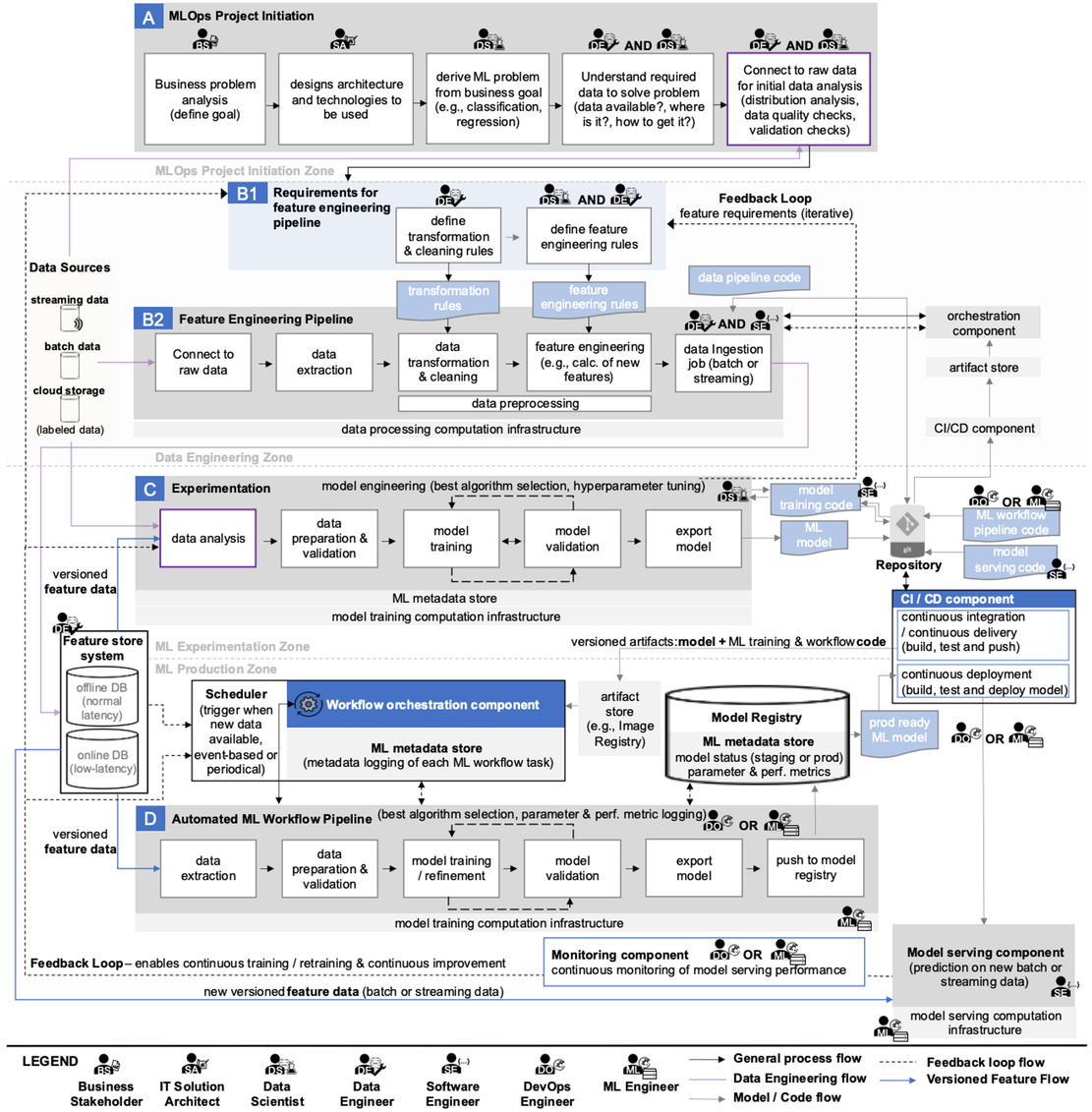
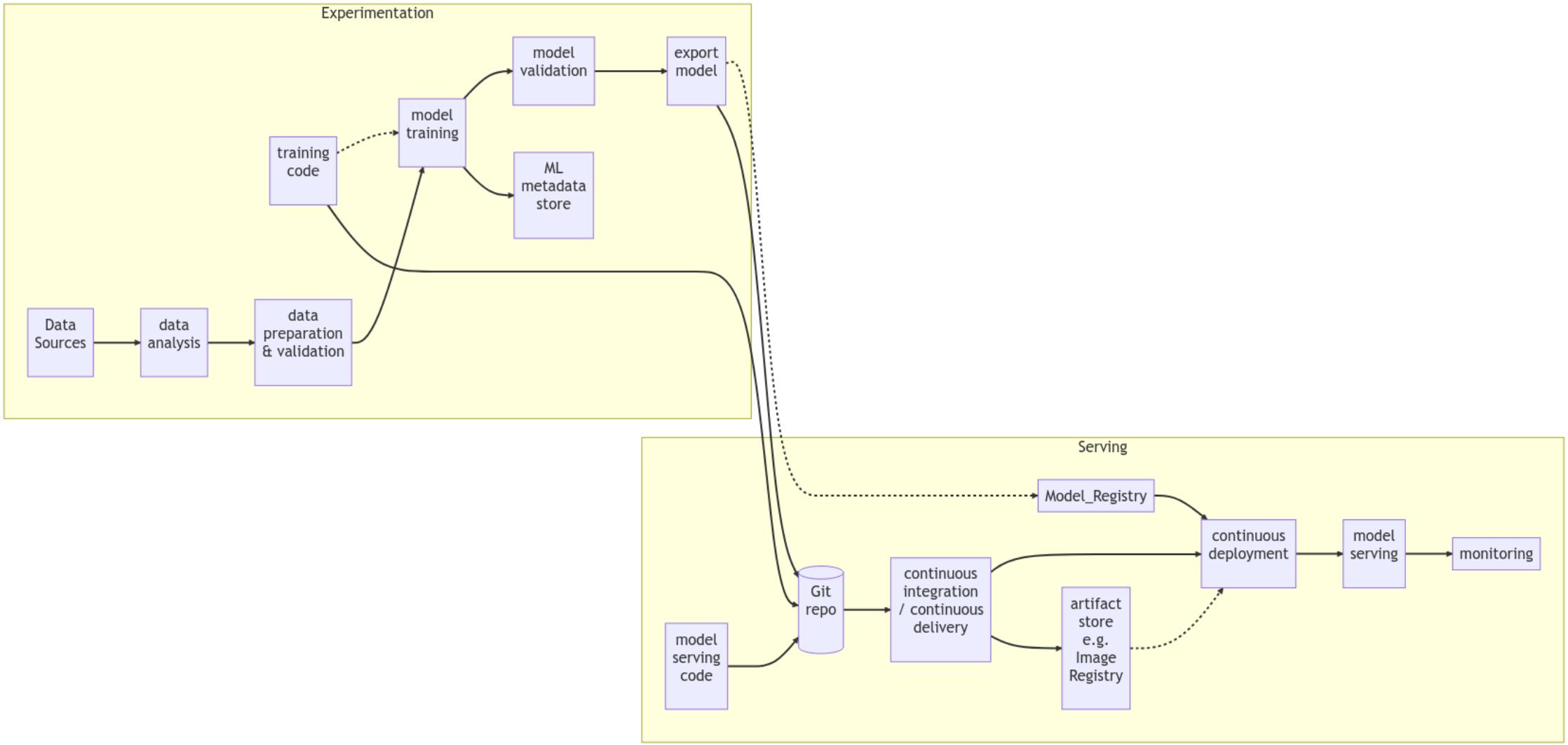


Figure 4. End-to-end MLOps architecture and workflow with functional components and roles

[5. Architecture and Workflow] MLOps workflow (simplified)



flowchart LR subgraph Serving repo --> CI[continuous\nintegration\n/ continuous\ndelivery\n\n] --> CD[continuous\ndeployment\n\n] --> serving[model\nserving\n\n] --> monitoring CI --> artifact_store[artifact\nstore\n e.g. \nImage \n Registry\n\n] -.-> CD serving_code[model\nserving\ncode\n\n] --> repo Model_Registry --> CD end subgraph Experimentation export -.-> Model_Registry training_code -.-> training sources[Data\nSources\n\n] --> analysis[data\nanalysis\n\n] --> prep[data\npreparation\n& validation\n \n] --> training[model\ntraining\n\n] --> validation[model\nvalidation\n\n] --> export[export\nmodel\n\n] --> repo[(Git\nrepo\n\n)] training_code[training\ncode\n\n] --> repo training --> metadata[ML\nmetadata\nstore\n\n] end

[6. Conceptualization] Define MLOps

- MLOps (Machine Learning Operations) is a paradigm including:
  - best practices
  - concepts
  - development culture
- engineering practice that leverages:
  - machine learning
  - software engineering (especially DevOps)
  - data engineering.
- bridge the gap between development (Dev) and operations (Ops).

[7. Open Challenges]

- Organizational challenges
  - need culture shift away from model-driven toward product-oriented
  - teams work in silos rather than in cooperative setups
  - decision-makers need to be convinced that an increased MLOps maturity and a product-focused mindset will yield clear business improvements
- ML system challenges
  - difficult to precisely estimate the necessary infrastructure resources
  - require a high level of flexibility in terms of scalability of the infrastructure
- Operational challenges
  - require governance, versioning of data, model, and code to ensure robustness and reproducibility
  - failures can be a combination of ML infrastructure and software

[8. Conclusion]

- the academic space has focused intensively on machine learning model building and benchmarking, but too little on operating complex machine learning systems in real-world scenarios
- MLOps and its associated concepts were defined
- common understanding will hopefully assist successful ML projects in the future

References for this summary

- Original paper: https://arxiv.org/abs/2205.02302
- Summary blog: https://nsakki55.hatenablog.com/entry/2023/01/23/102630

Appendix

MLOps Maturity Levels

- Level 0-2 by Google: https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning
- Level 0-4 by Microsoft: https://learn.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-maturity-model
- Level 1-7 by MLOps Community: https://mlops.community/mlops-maturity-assessment/