
# Auditability in ML Systems v1.0.0

## Gastvortrag - Software Engineering in KI Systemen – Universität Leipzig

12.06.2023

AI generated

# Agenda

1. About me
2. Why do we want auditable ML systems
3. Definitions
4. AI Act - Requirements
5. Auditability from a technical perspective

# About me

# About me

***Jürgen Stary***

*Solution Architect MLOps @*
*AI Factory DB Systel GmbH*

- Since 2015 worked in various roles with a focus on building and operating ML Systems and applications
- My favorite question is „Why?“

#gernePerDu
#prototypingInEnterprise
#systemsThinking
#challengeMe

/jaystary


Perception vs Reality ☺

# About Rachel

**Dr. Rachel Hagemann**

*Senior AI Specialist @*
*AI Factory DB AG*

- Spent the last 10+ years building, applying and testing mathematical/ML algorithms.
- Drive to understand the strengths and weaknesses of the ‚OG algorithms' and ‚new Sheet' tools in the ML tool box.
- Motivated to see what makes an algorithm break.
- #SafeAI

# Background - What do architects do?

Product

## ENTERPRISE ARCHITECT (EA)

**Technology Focus**

**Strategy Focus**

**Key Competencies**

- Master of EA Frameworks (i.e. TOGAF, Zachman Framework)
- Uncovers operational gaps
- Analyzes information through data models and architecture diagrams
- Communicates the value of new IT strategies and keeps stakeholders informed of ongoing initiatives

https://www.leanix.net/de/ty/comparing-it-architecture-roles-poster

## SOLUTION ARCHITECT (SA)

**Technology Focus**

**Strategy Focus**

**Key Competencies**

- Coordinates ongoing activities
- Translates the design concept to IT operations
- Defines a best-fit solution for existing problems
- Ensures technological risks are accounted for and solutions meet necessary requirements

## TECHNICAL ARCHITECT (TA)

**Technology Focus**

**Strategy Focus**

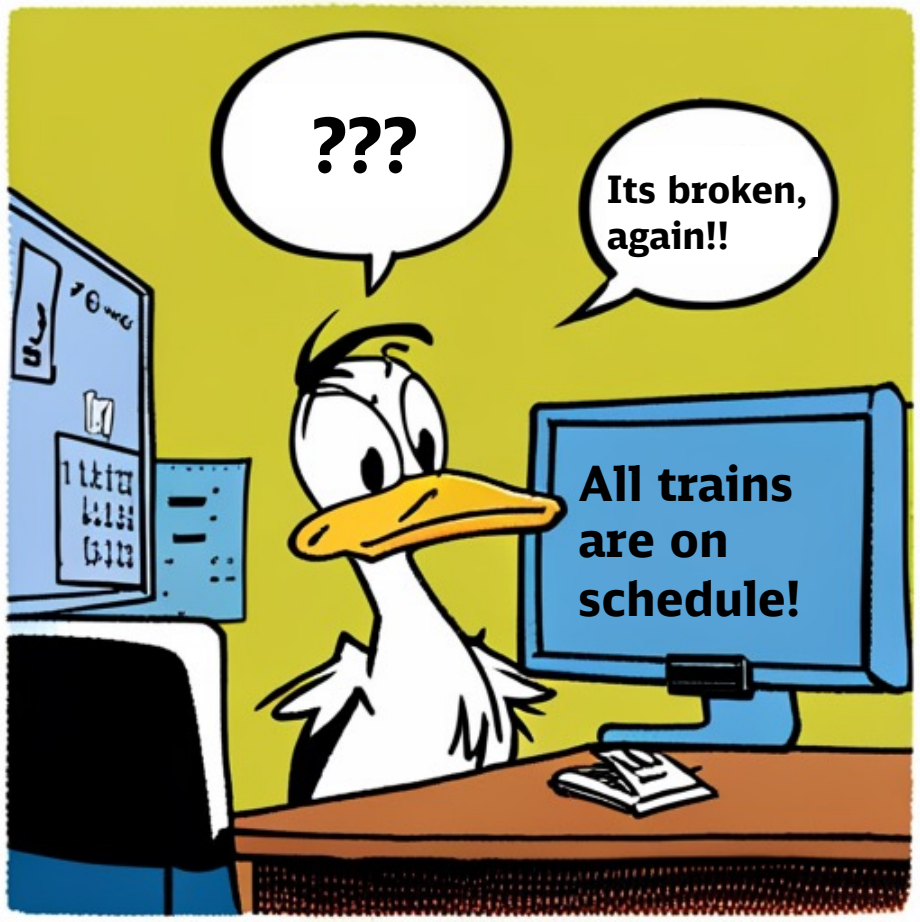
**Key Competencies**

- High level of in-depth expertise (i.e. Python, Java)
- Provides recommendations to address potential threats
- Implements technical processes to roll out solutions
- Delivers fully functional products in a timely manner for the end user

Engineering

# Why do we want auditable systems

# *“Anything that can go wrong, will go wrong”* v1.0.0

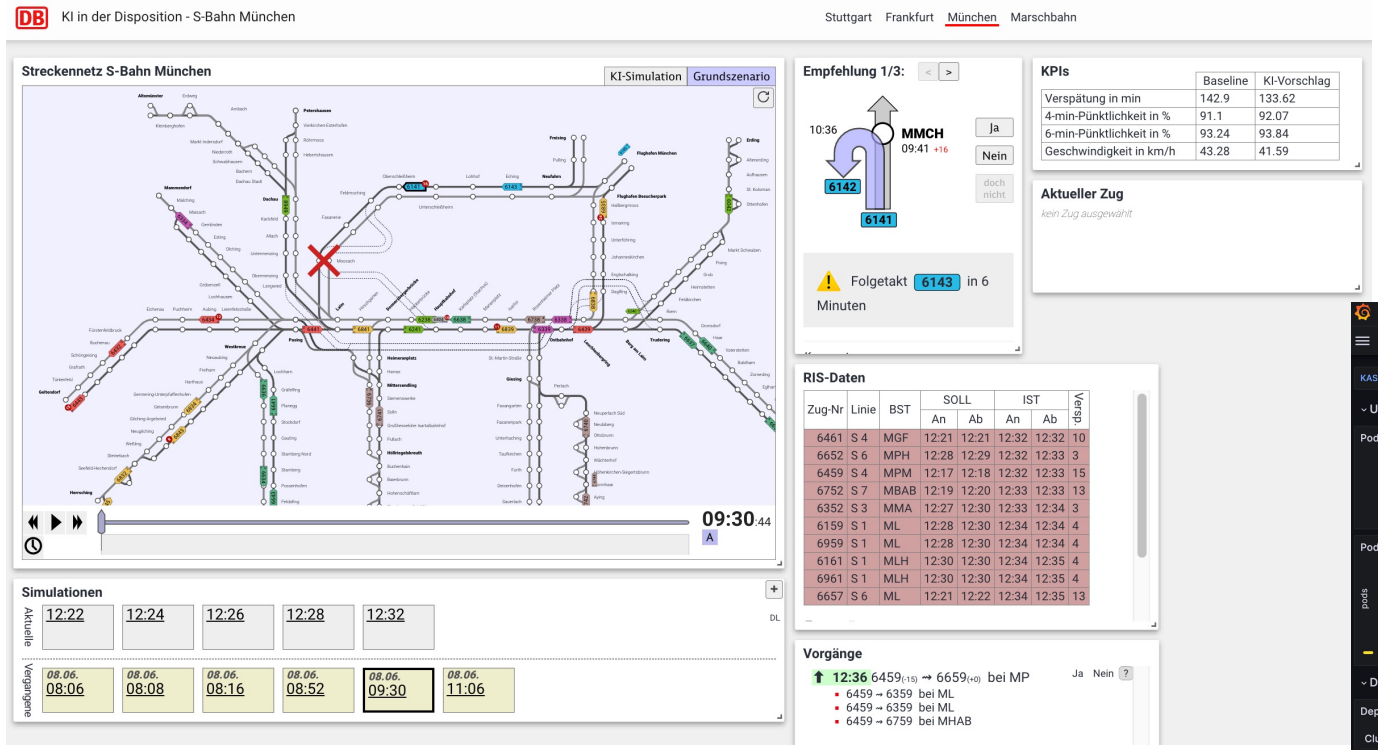



AI generated

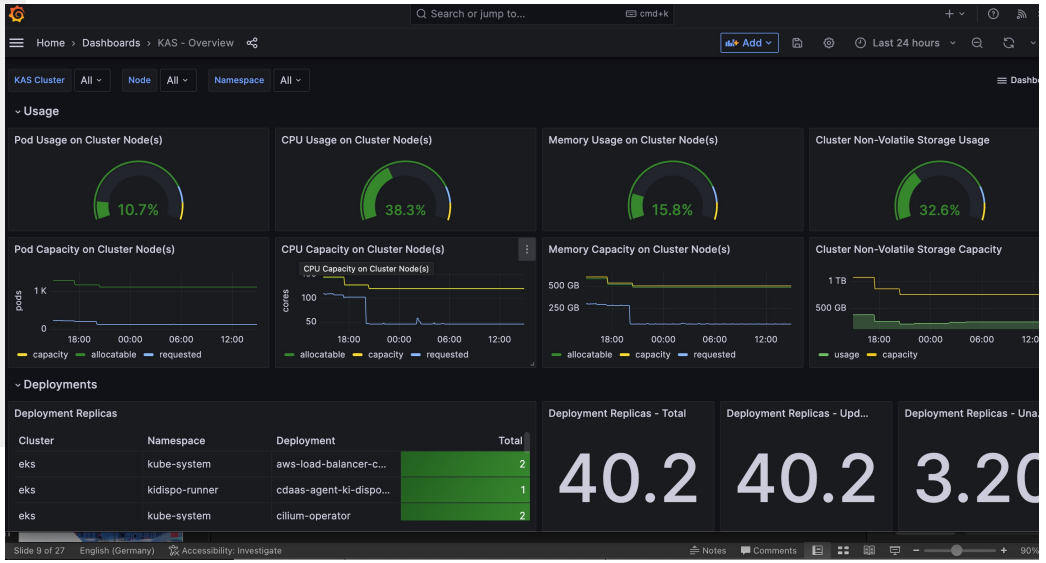
# A ML System - KI in der Disposition

**Reinforcement learning algorithms support human decision making in complex environments to minimize train delay / increase „Netzfrequenz“**

## Frontend

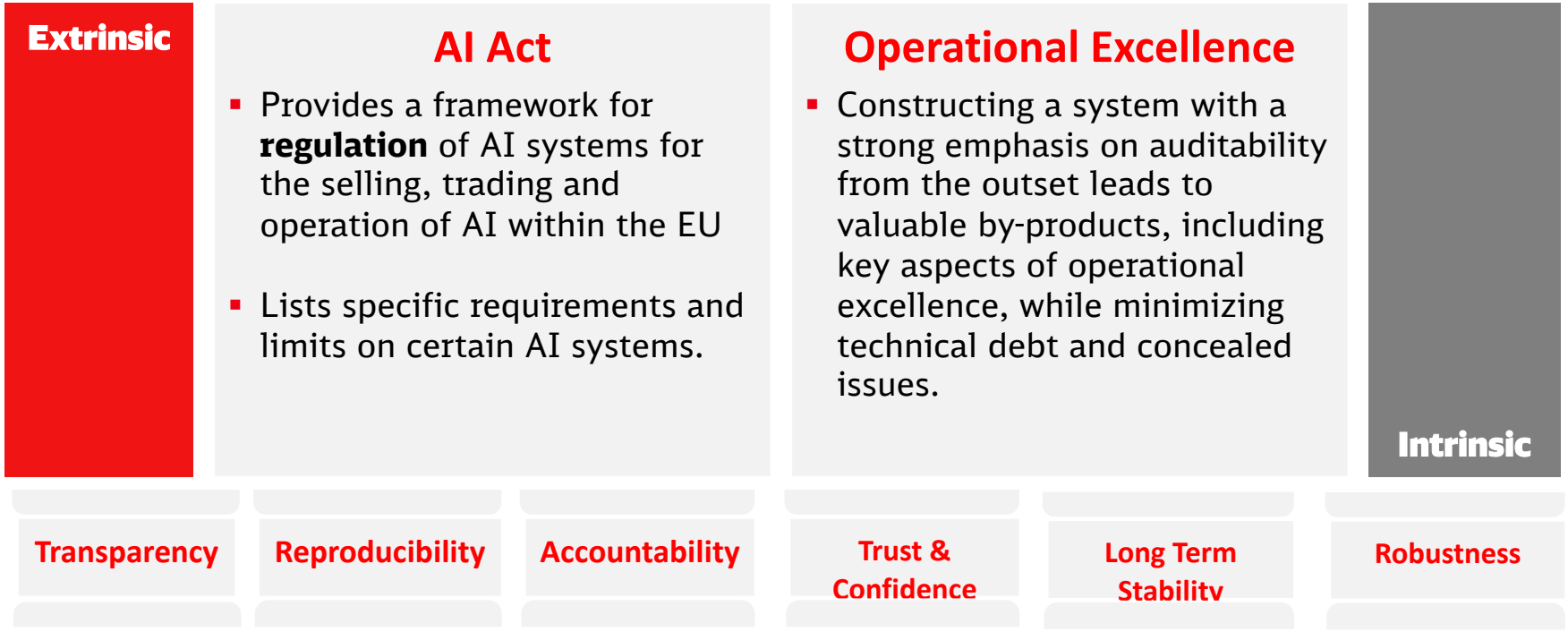



## Monitoring




https://digitale-schiene-deutschland.de/AI-Prototyping
https://www.handelsblatt.com/unternehmen/handel-konsumgueter/schienenverkehr-bahn-bringt-kuenstliche-intelligenz-aufs-gleis/27222608.html

# Foundations of auditable ML Systems



# Definitions

# Definition of auditability

A good auditable system is **accessible** and **transparent** to each respective stakeholder.

## External Stakeholders

**Requires:**

- An overview of all AI/applications Systems at the company
- Designation for each AI application (high risk/moderate risk/low risk)
- Documentation of the processes, rolls and responsibilities for the ‚high risk' systems
- Documentations of compliance to current laws and regulations

## Developer

**Requires:**

- Technical documentation on
  - Data/Data quality/Restrictions/Contacts
  - Tooling
  - Feature Integration
  - ....
- Standards and processes for working with data, deploying models ...
- Limitations (IP/Licenses/Security) on tools/code/opensource etc.

# Stakeholders

1

## External Stakeholders (Extrinsic)

- EU/Country Government
- Regulatory Authorities
- Customers / Users
- Third-Party Auditors
- ...

## Concerns

- ... revolve around:
  - compliance with laws/regulations
  - system performance
  - impact on society
  - upholding values and human rights
  - ...

2

## Internal Stakeholders (Intrinsic)

- Management
- Data Scientists / ML Engineers
- DevOpsSec/MLOps Teams
- Compliance (Liability)
- ...

## Concerns

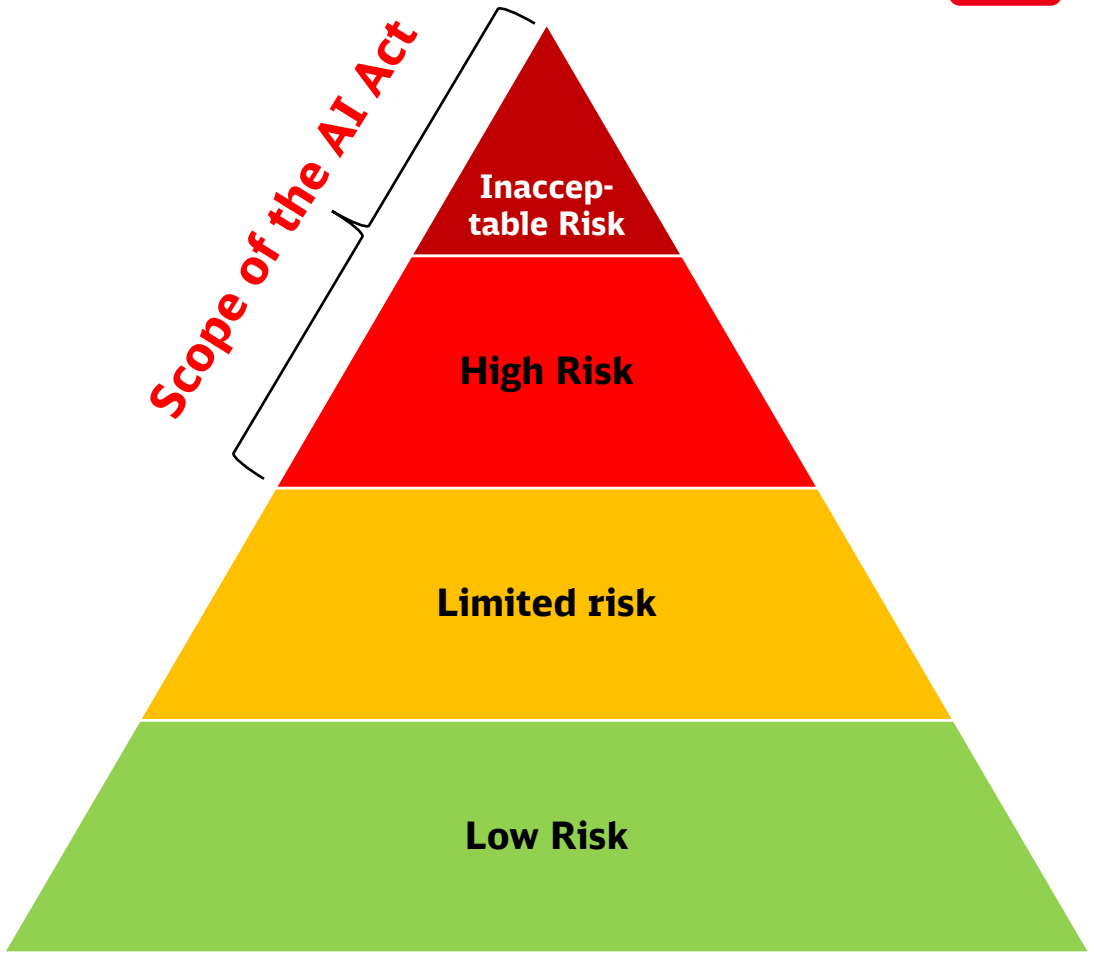
- ... revolve around:
  - Risk assessment
  - Efficient development, deployment and ops
  - Internal guidelines and policies
  - System performance & reliability
  - Organizational goals & objectives

# AI Act – Requirements

# Highlights – AI Act

- Regulates the use of AI Systems that are of:
  **Inacceptable risk**
    - social scoring
    - remote biometric identification
    - …

  **High risk**
    - critical infrastructure
    - employment
    - safety components
    - …



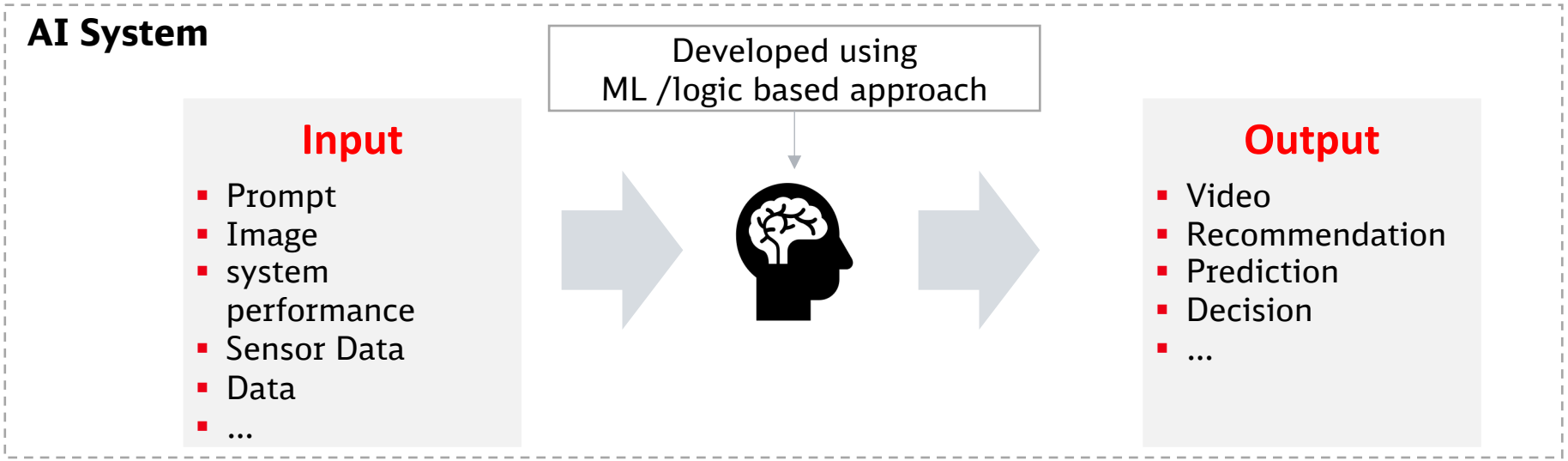

- Excludes scientific and product research

- Final voting expected at the end of 2024

- Estimated Costs for non-compliance  up to*:
    - max(30 000 000 EUR , 6 % total worldwide annual turnover)
    - SMEs & start-ups, 3% of worldwide annual turnover

* Proposal is still in negotiations, however foundational aspects are not expected to change

# Definition of AI (17.05.2023) – AI Act




## Current (paraphrased) General definition of AI system (6a, 6b)

AI systems should **infer** the way to achieve a given set of human-defined **objectives** based on **inputs** (data,prompt, image, video, ...) using **machine learning** and/or logic- and knowledge based approaches in order to produce an **output** (prediction, suggestion, video,decision...) .

## Autonomy

The concept of the autonomy of an AI system relates to the degree to which such a system functions without external influence human involvement.

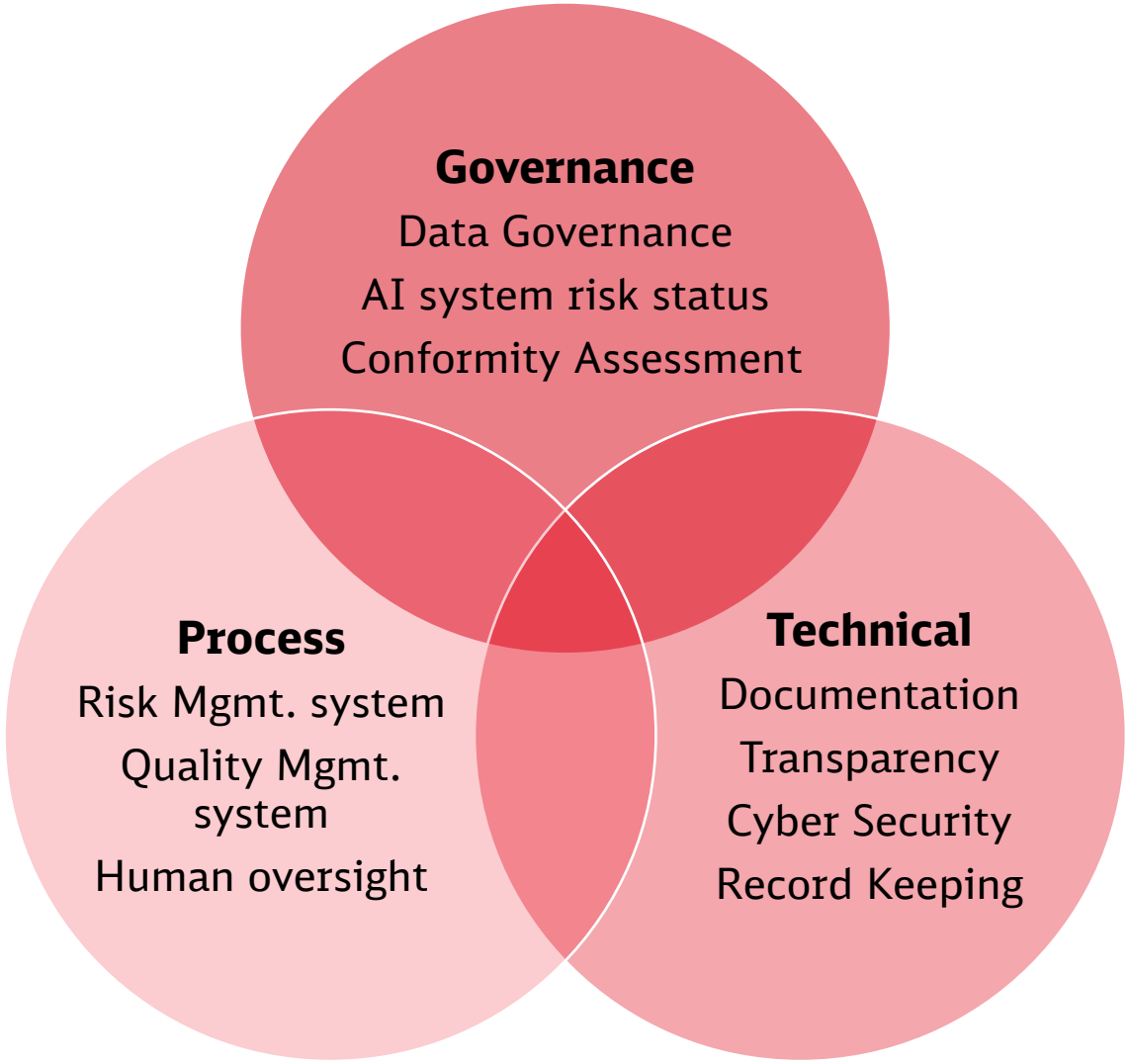
## Distinction with 'classical' software

The definition should make a distinction between the software of **artificial intelligence** and more **classic** simpler software systems and programming approaches.

## Out of scope

A system that uses **rules defined solely by natural persons** to automatically execute operations should not be considered an AI system.

# Requirements – AI Act




## Transparency through Technology

Solutions need
**Automation** and **Integration**
in the development process

# Auditability from a technical perspective

# Modern ML Systems produce a wide variety of audit relevant artefacts

**1** **From exploratory origins** → **2** **To coherence** → **3** **Our requirements**

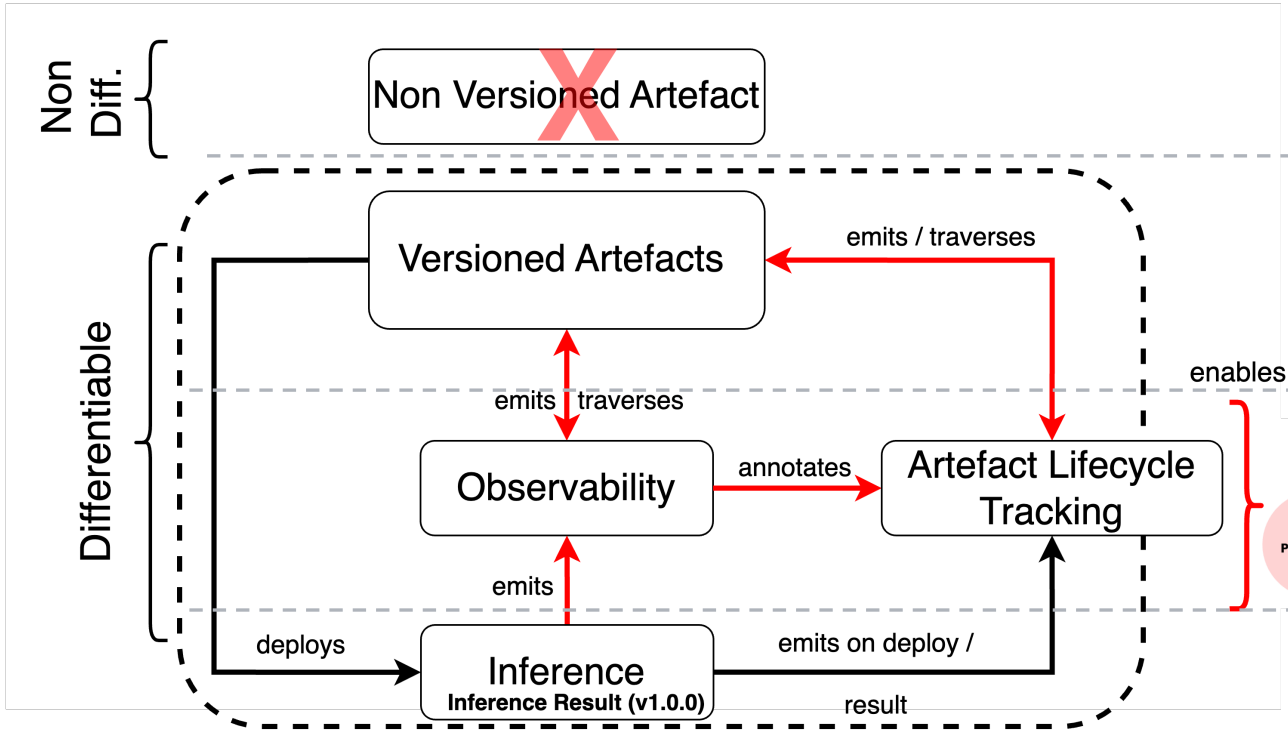
Increasing coherence by implementing requirements through the application lifecycle …




enables traversing the system

- Simple to implement for low-/limited risk use-cases
- “GitOps” / structured automated Versioning as **first order principles**
- Utilization of platform tooling
  - Engineers should be familiar with tooling
  - Enabling concept validation and quick wins for use-cases
  - Artefact tracking through available Sources of Truths
- **Enabling a high degree of automation in processes**

# A mental model for working with artefacts




**Layers:**

Non versioned artifacts cannot be integrated

**Development Layer** Versioned artefacts as a result of continuous development processes, e.g.:
- Data
- Algorithms
- 3rd Party Information

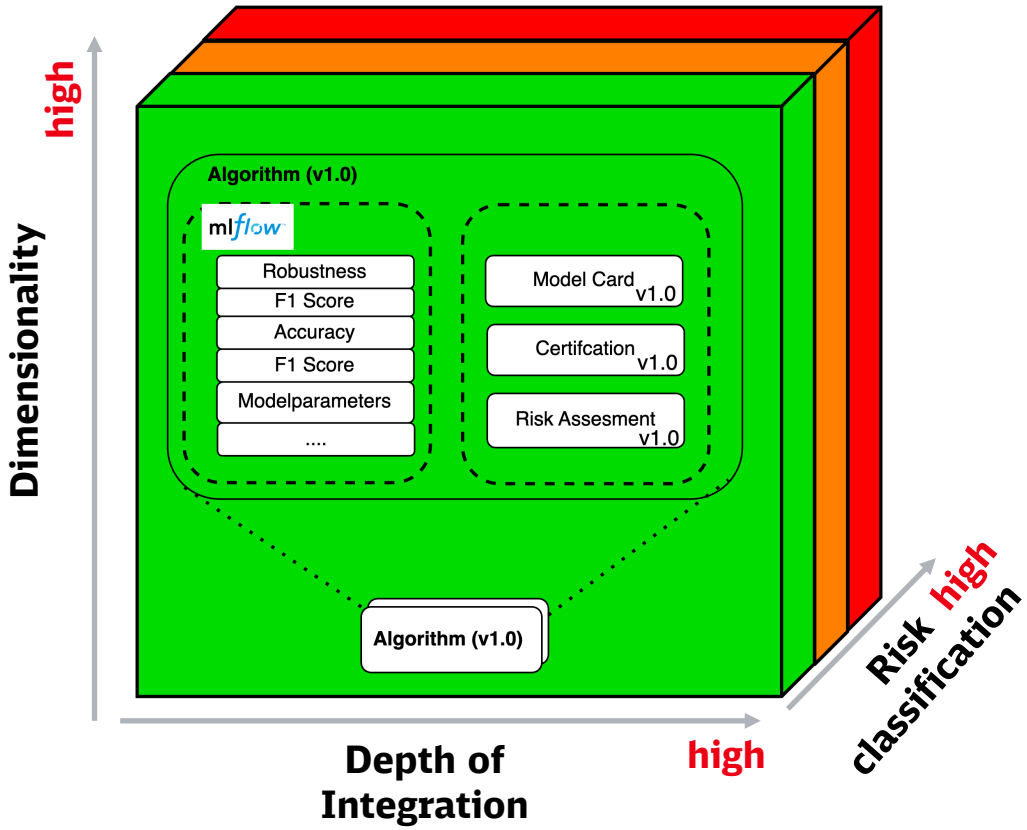
**Aggregation Layer** Structured data as a result of Development / Runtime processes
-> annotating tracked artefacts with additional information

**Inference Layer** Deployed Artefacts in a inference environment annotated with aggregate information from previous layers.
Querying its state provides a snapshot of the current system meta data

**Coherence:**
- Given a versioned artefact, we must be able to traverse the system, its artifacts and their history to ensure our defined foundations

**Steps towards Coherence:**
- Integrating versioned artefacts starting from the end adding them to tracking increases the degree of understanding about the system

# Challenges with artefact tracking




**Dimensionality**
- The level of detail at which artifacts are tracked

**Depth of Integration**
- The extent to which artifacts within our system are tracked

**Risk Categorization**
- The criteria for determining which properties of artifacts require monitoring

**Use-case specific requirements**

- Industry specifics (e.g. regulated industries)
- Tamper proofness …
- Multitenancy on a per project/user level …
- ….

**Capturing artefacts at higher dimensions requires specialized tooling**

- Current market offerings still early phase
- Platform components **>>** OSS / Buy **>>** Make

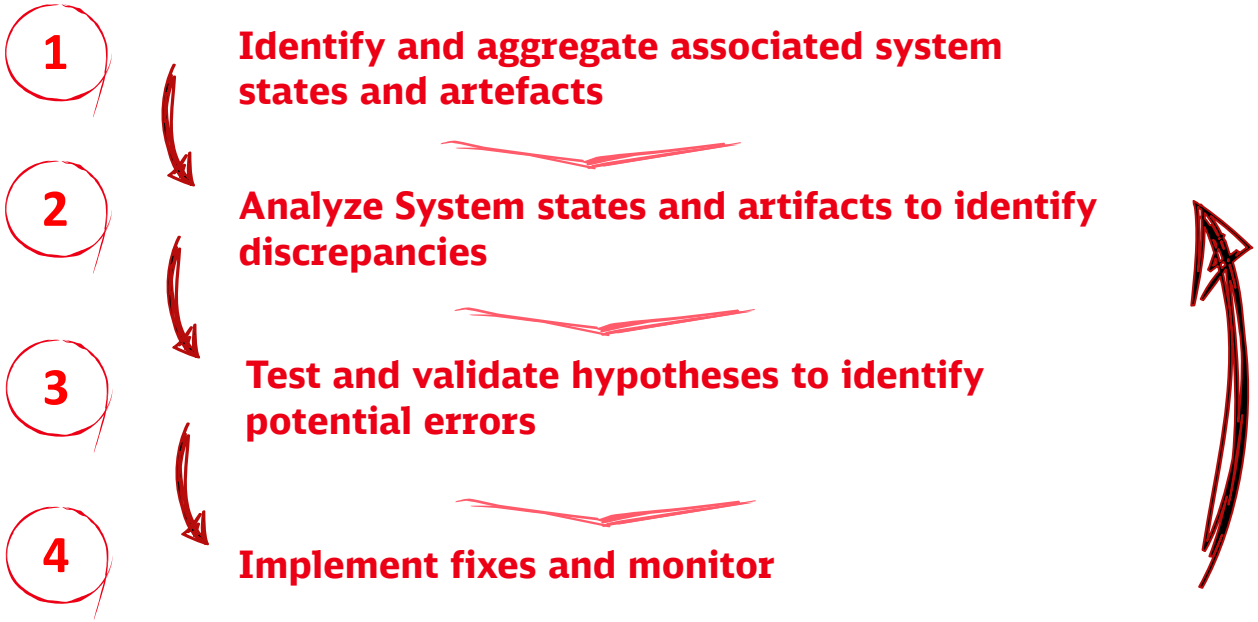
**Specialized tooling requires maintenance and additional education from users and engineers**

- Requires additional discipline, consistency embedded in workflows and culture
- Gaps in automation -> possibility for gaps in our DAG, impacting lower level integration

# Common artefacts in ML Systems

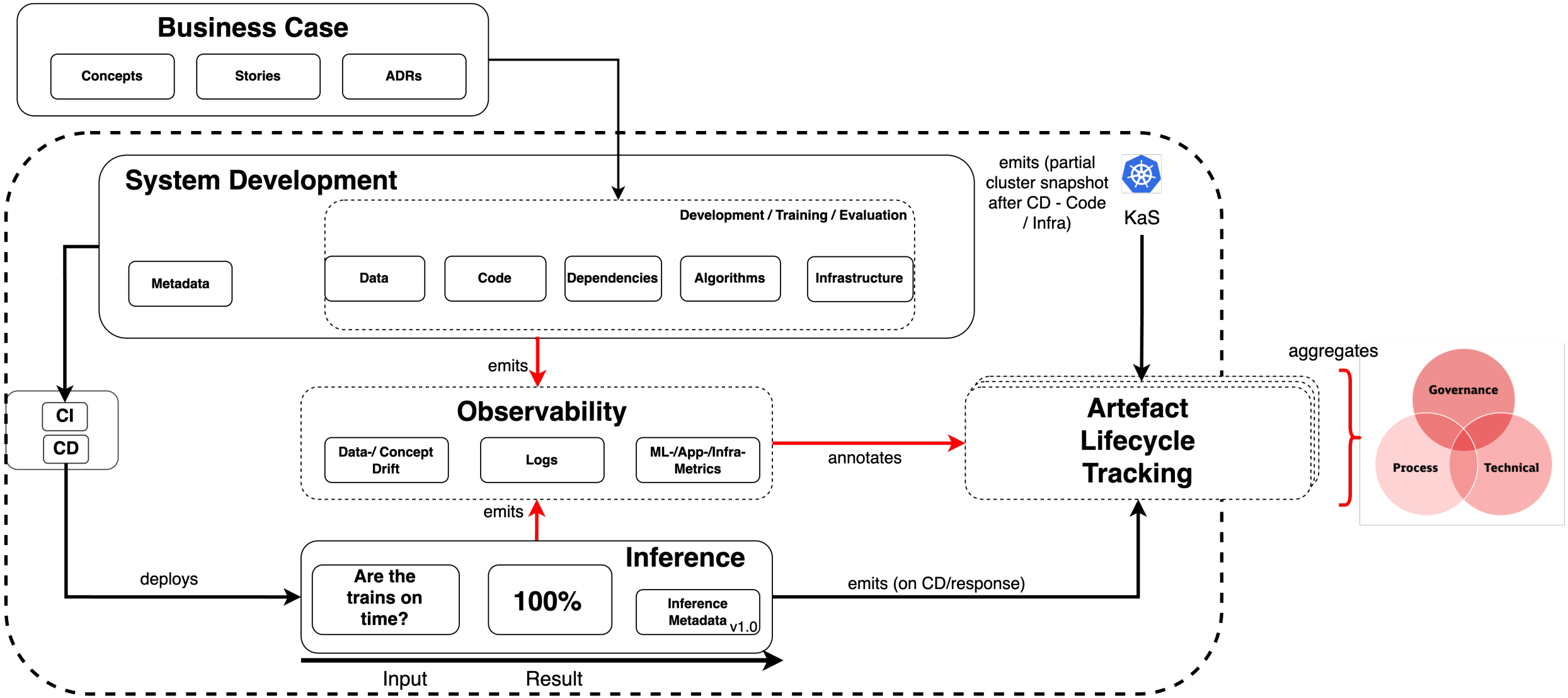
| Artefact | Description | Source of Truth |
|---|---|---|
| **Algorithm Input** | ./. | ... |
| **Inference Result** | ./. | ... |
| **Inference Metadata** | Process information about Inference, e.g. Algorithm Version, Latency, Telemetry, ... | ... |
| **Observability** | Logs and Emits from Development / Inference Layer | ... |
| **Data** | References datasets, data slices, ... on a data source | ... |
| **Code** | ./. | ... |
| **Dependencies** | Utilized package and/or engine dependencies | ... |
| **Algorithms** | Aggregation of Code, Data, Parameters, Certifications as a result of the alogrithm development process | ... |
| **Metadata** | References ML related artefact metadata | ... |
| **Infrastructure Metadata** | References to training / evaluation infrastructure | ... |
| **CICD** | References to Pipelines, Qualitygates, owners of a deployment, .. | ... |
| **Concepts / Stories / ADRs ...** | Planning & Documentation artefacts that result in development | ... |

# Navigating the system




**An ML system built upon defined principles inherently facilitates a coherent environment. This allows both external and internal stakeholders to readily discover answers to their specific inquiries.**

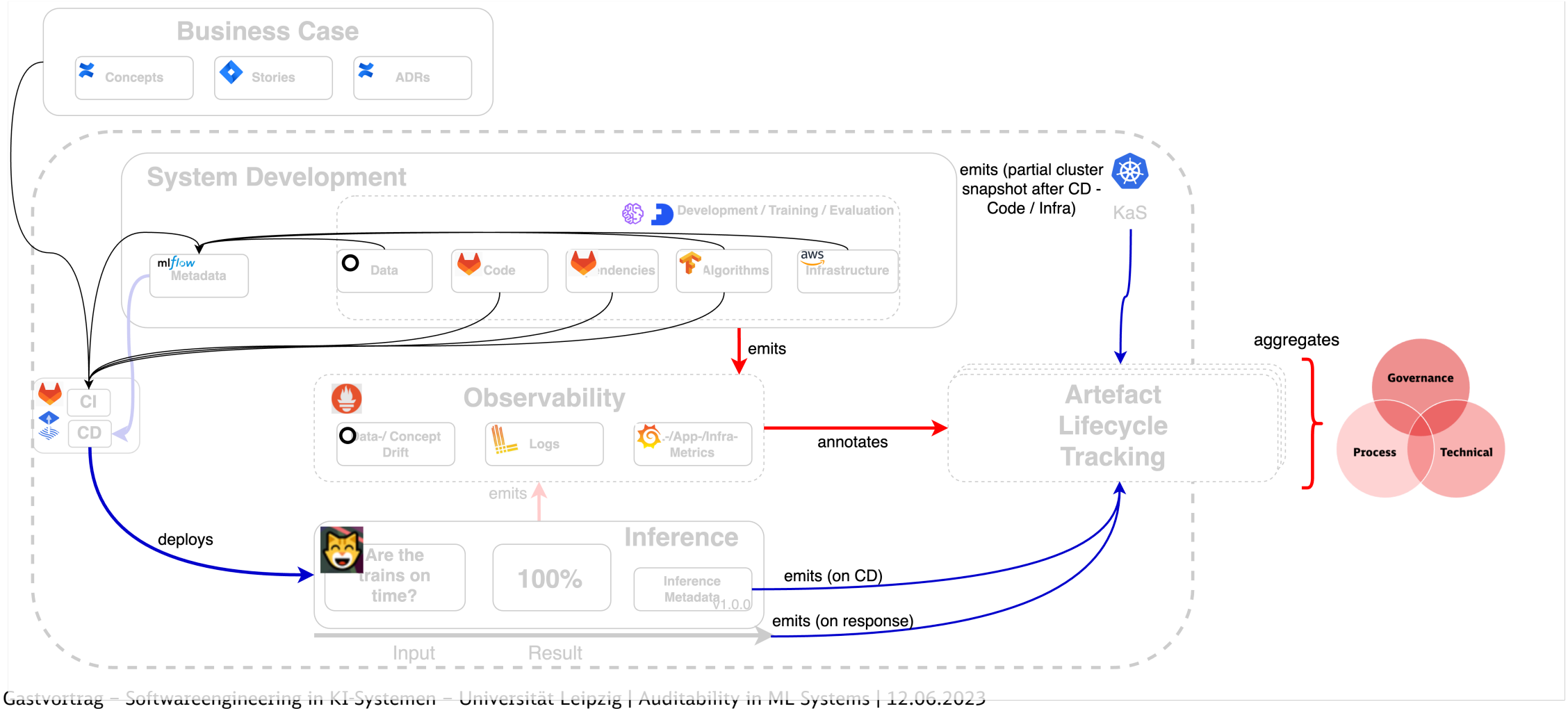
# Filling our mental model with artefacts



# All streets must lead to Rome eventually...

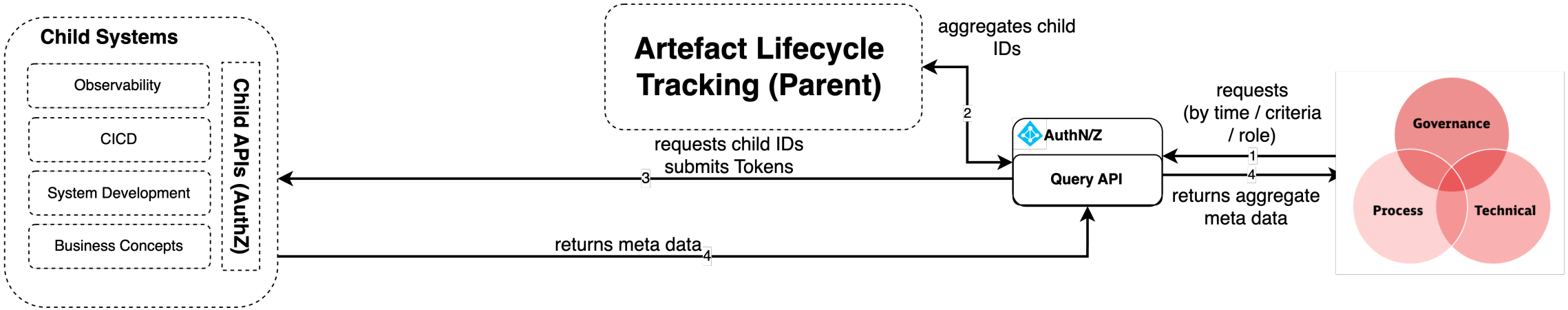
**Additional considerations for implementation:**

- Running everything in a common/integrated environment (e.g. K8s) simplifies tracking and process implementation
- Minimizing Sources of Truth simplifies data aggregation and traceability
- Being able to capture all state changes in the system (e.g. on / after a CD / declarative Infra) simplifies understanding



# Retrieving our system state

**By adding a permissioned query api in front of the overall application we can query all components and aggregate the relevant meta data by defined criteria**



AI generated

# Thank you for your interest!

***P.S. db.de/ai-jobs***