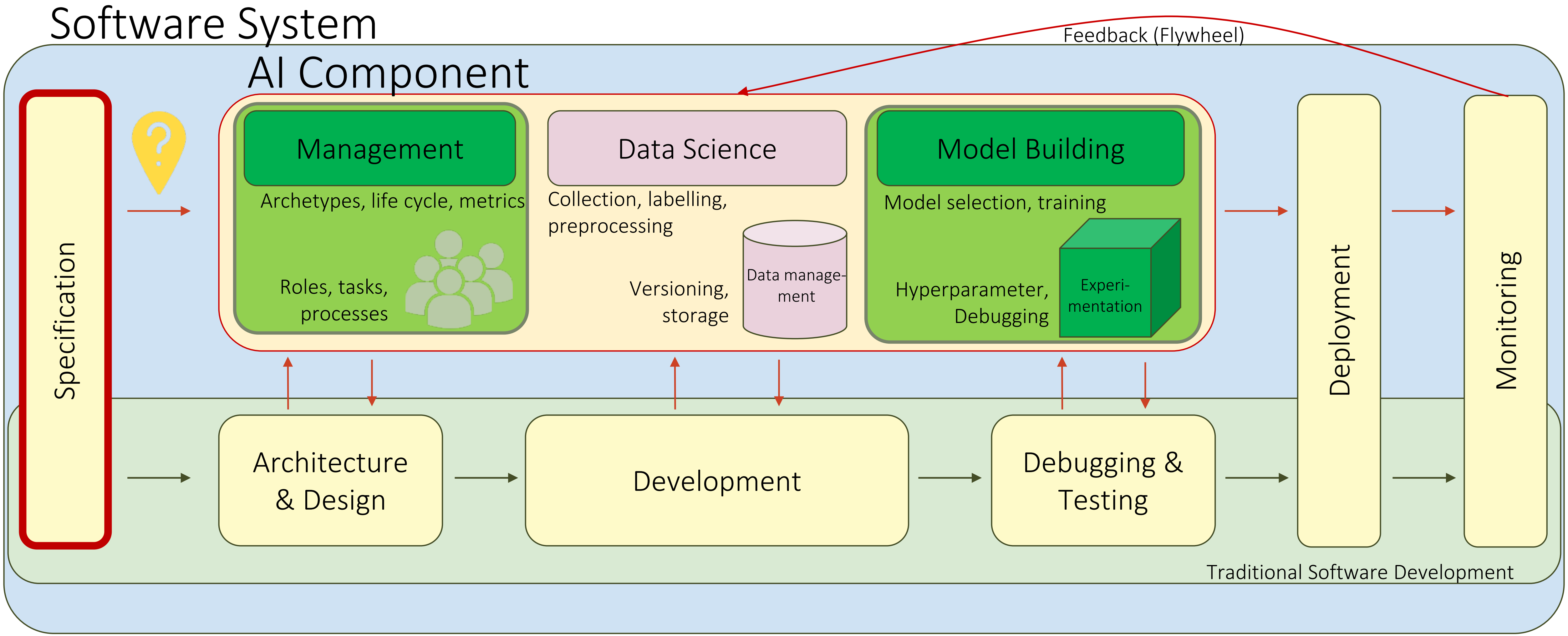
Software Engineering for AI-Enabled Systems

SOFTWARE SYSTEME

UNIVERSITÄT LEIPZIG

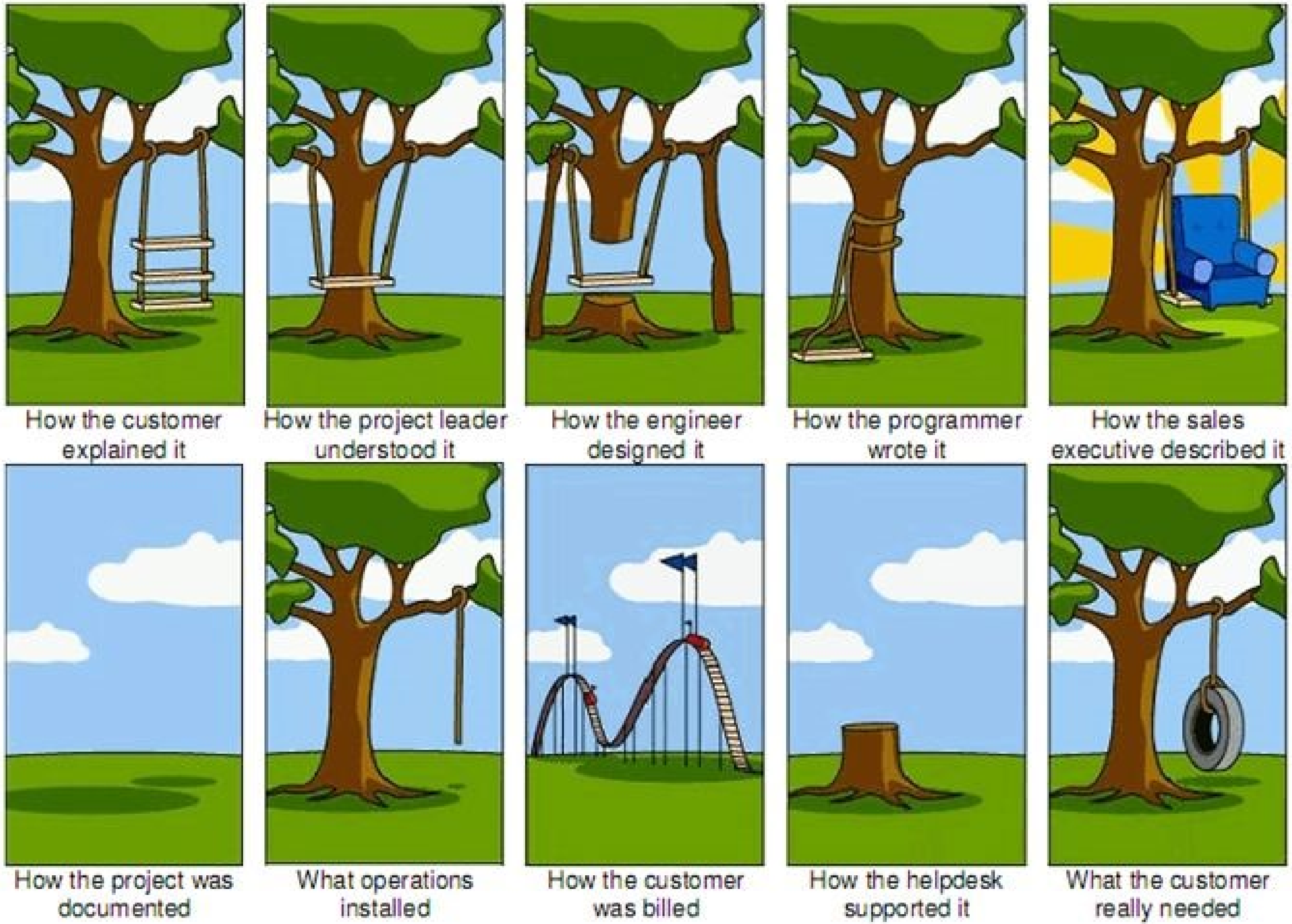
Prof. Dr.-Ing. Norbert Siegmund
Software Systems



What is a proper specification for an AI model / component?

- Specify accuracy levels
- Specify success metrics related to business value
- Specify how to measure failure
- Specify how to detect failure
- Specify hardware and non-functional properties, such as inference time

Topic I:
Requirements & Specification (Basics)


How the customer explained it
How the project leader understood it
How the engineer designed it
How the programmer wrote it
How the sales executive described it
How the project was documented
What operations installed
How the customer was billed
How the helpdesk supported it
What the customer really needed

Requirements Engineering (RE)

Requirements specify **properties, functionality, use cases, quality** on the system to be developed. RE is a **systematic way of iteratively** developing requirements via different steps.

Challenges:

- Customers not knowing or able to describe what they want
- Different languages, conflicting requirements, evolution

Types and techniques:

- Functional requirements, elicited via scenarios, walkthroughs, use cases
- Non-functional / quality requirements, elicited via stakeholder survey

Do never assume that you know what the customer wants!
This is even more the case for AI-enabled SWS! **Do not skip RE!**

Types of Requirements

Functional requirements: Specify function (features) of a system

Examples: Interface to a payment system, email notification, order system, logistics, management, etc.

Non-functional requirements: Specify all properties, abilities, conditions, and behaviors of the system that are not associated with a functionality

Examples: Performance, energy consumption, privacy, safety, security, reliability, development cost

Constraints: Specify restrictions on the implementation of the system

Examples: Must run on system X; must deliver a result in X seconds; must finish dev in 180 days

RE Process

Requirements elicitation

Identify stakeholders of the system (entities or persons affected by it), gather requirements of all stakeholders (e.g., interviews, workshops)

Identify scenarios, use cases, walkthroughs

Natural language, models, formulas, observations, code, artifacts

Challenges: System boundaries unclear, incomplete understanding, unnecessary information, unclear terms, obvious inform. omitted, vague requirements, volatile, conflicting views

Requirements modeling & specification

Translate vague requirements into actionable (conflict-free), unambiguous specifications that can later be tested or verified

Use case diagrams, formal requirements specifications, user stories

Challenges: Multitude of vague with different scales and dimensions, conflicting requirements, concretizing requirements, verifiable metrics, capturing of pre-/post-conditions

Requirements validation & documentation

Validate the specification with customers (see problems for elicitation)
Find appropriate forms for validation (prototypes, mockups, etc.)
Volere template, snow cards

Challenges: Amount of documentation, overview, validation process

Requirements Elicitation

Retrieving (all) requirements from (all) stakeholders to obtain a system description

Focus: Customer

Methods:

- Survey techniques (questionnaires, interviews)
- Creativity techniques (brainstorming, change of perspective, etc.)
- Document-centric techniques (system archeology, requirements reuse, etc.)
- Observation techniques (field observation, etc.)
- Support techniques (workshops, mind maps, prototypes, recordings, cards, etc.)

Requirement vs. Specification

Requirement: Customer-oriented description (often in natural language) about the desired properties of the software; usually written in the language of the domain (specific words and terminology);
Helps to understand what the software is supposed to do

Specification: Developer-oriented precise description and terminology of the functional and non-functional features of the software and their respective constraints;
Helps to understand how to develop the software and verify its correctness

Requirements Specification

Turning diverse, ambiguous requirements to concrete, actionable, unambiguous specifications that a software engineer can understand (and implement).

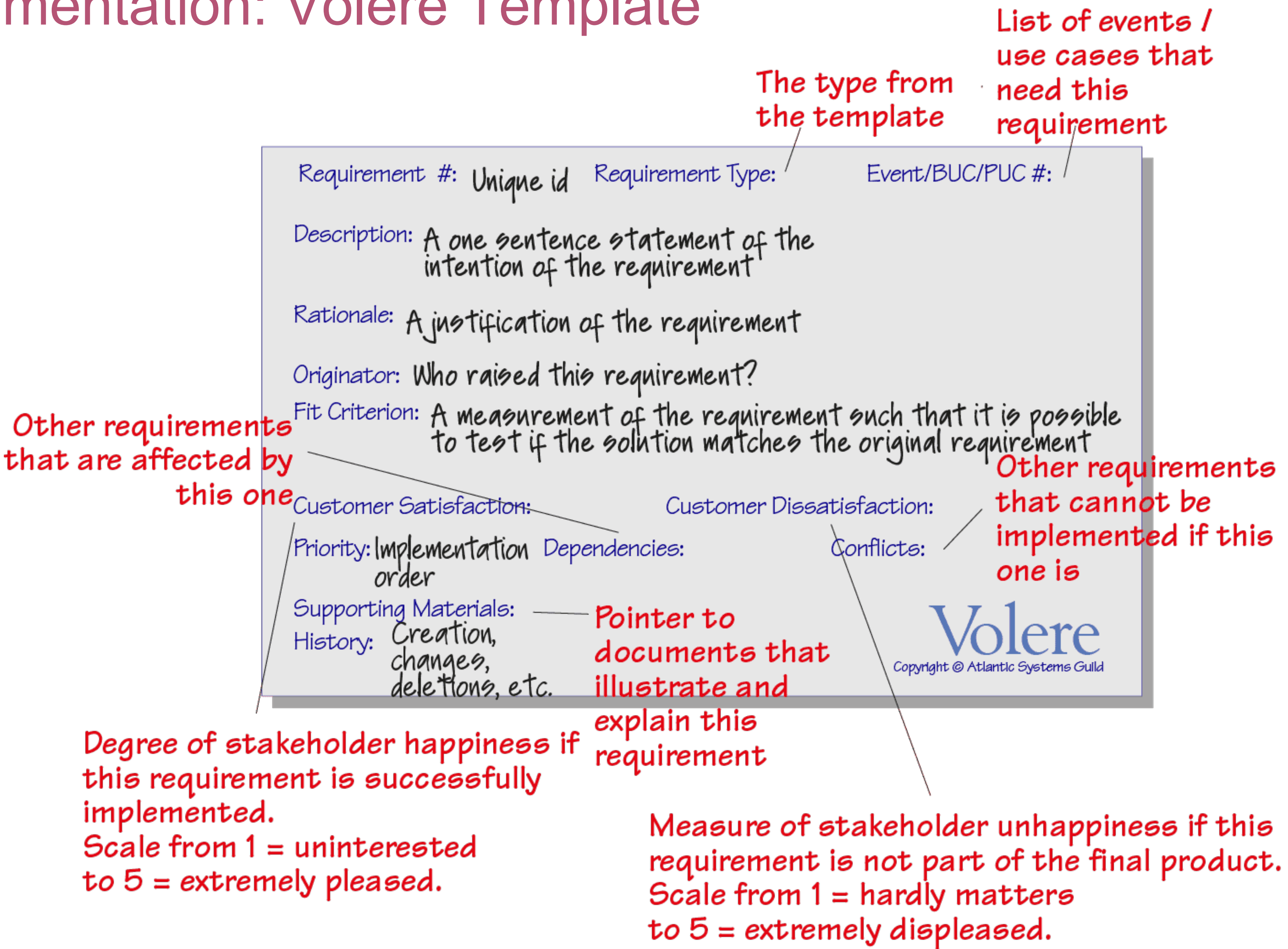
Specifications can be verified by testing or formal methods.

Focus: Developer

Methods:

- Use case diagrams
- IEEE software requirements specification (SRS)
- User stories

Documentation: Volere Template



Topic II:
Requirements Engineering for AI-Enabled Systems

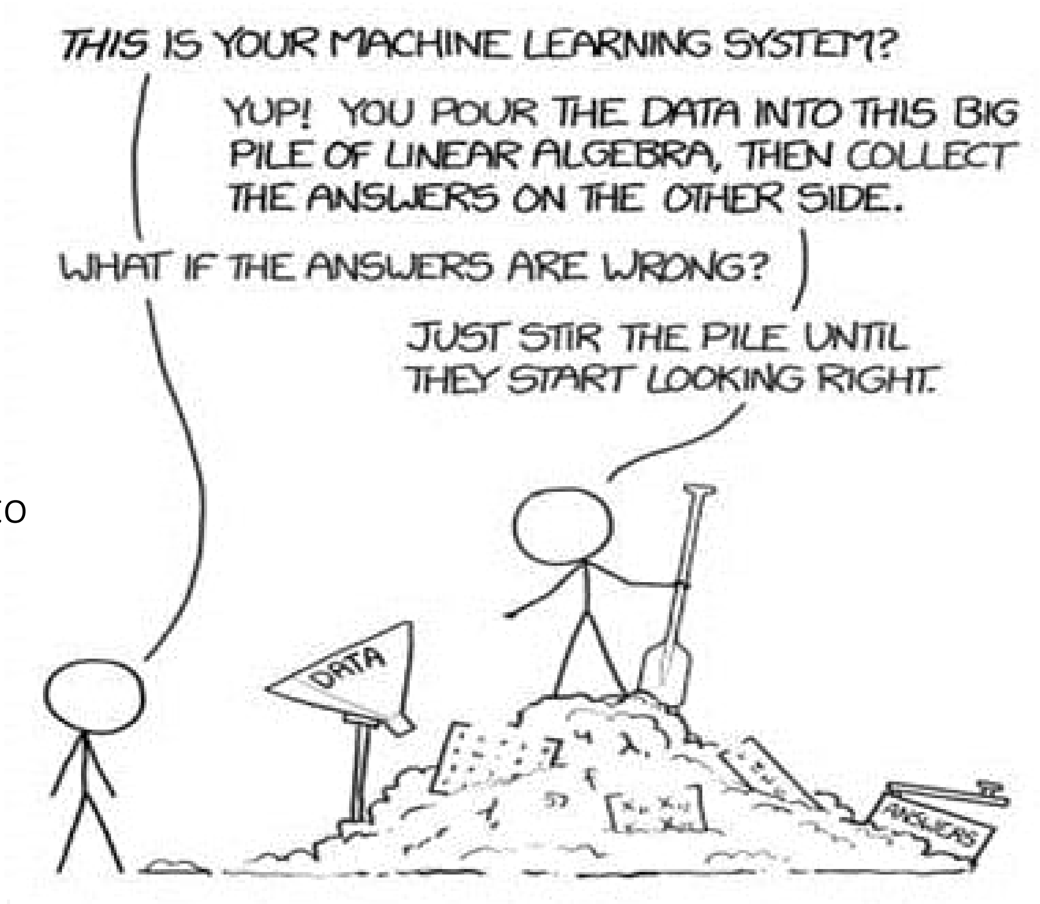
Requirements engineering might be the most difficult activity for the development of ML-based systems.*

What distinguishes good from bad data?

Do we have a kind of assurance what comes out?

What can we expect from “good” data?

Is the outcome what we want to achieve?



Can we define what is supposed to come in?

Can we measure the correctness of the outcome?

Can we assure that the outcome is unbiased?

And how would we define or monitor bias?

AI-Specific Challenges for RE

Source: Laura Pullum: Verification and Validation of Systems in Which AI is a Key Element. SEBOK

According to practitioners, AI pose challenges due to:

- **Missing oracle:** No ground truth or trustworthy tool, failing to be able to define what is the correctness criteria for system outputs
- **Imperfection:** An AI system is never 100% accurate
- **Uncertainty of untested data:** Uncertainty on how the AI system behaves for unseen data (see, for example, adversarial examples)
- **High dependency of behavior on training data:** System behaves in dependence on what data is fed during training and not how the behavior is (formally) specified

Leads to typical characteristics of AI systems:

- Erosion of determinism
- Unpredictability and unexplainability of individual outputs (Scully et al. 2014)
- Unanticipated, emergent behavior, and unintended consequences of algorithms
- Difficulty of maintaining consistency and weakness against slight changes in inputs (Goodfellow et al., 2015)

Specification & Requirements Areas

Context specification: Define the context in which the AI-component is deployed and operates

Data specification: Define requirements on the used data to ensure correct/desired functional and non-functional behavior of the system

Model specification: Define model type, parameters, metrics, dimensions, adaptability, portability.

Specification of metrics for real-time monitoring of deployed models: Enable a continuous improvement and early data/model shift detection

Human factor specification: Define how humans react on, for example, automated decisions

Ethical requirements: Define what ethical decisions are made by the system and how are ethical aspects will be addressed.

Non-functional / quality requirements: Define requirements on quality attributes (explainability, legal req.)

Hardware requirements: Specify the HW systems the AI component trains and inferences on.

Context Specification

Define under which (environmental / contextual) conditions the AI-component will operate. Optionally specify conditions for data/model shift.

Rational: The model cannot be placed in another context without re-training and testing. Considers the famous no-free-lunch-theorem.

Examples:

- Autonomous driving only on highways with clear sight
- Speech recognition only on specified hardware (microphones)
- Recommendations only for trusted / registered users on a subset of products
- Fault detection on goods only for certain types and faults

Data Specification & Requirements I

"According to IBM spokesperson Edward Barbani, Watson for Oncology started off by using real patient data. But this made it difficult to do updates each time the guidelines changed, so the scientists switched to using hypothetical cases. "Synthetic cases allow you to treat and train Watson on a variety of patient variables and conditions that might not be present in random patient samples, but are important to treatment recommendations," he said."

SCIENCE TECH HEALTH

IBM's Watson gave unsafe recommendations for treating cancer

Doctors fed it hypothetical scenarios, not real patient data

By Angela Chen | @chengela | Jul 26, 2018, 4:29pm EDT

"Watson for Oncology was supposed to synthesize enormous amounts of data and come up with novel insights. But it turns out most of the data fed to it is hypothetical and not real patient data. That means the suggestions Watson made were simply based off the treatment preferences of the few doctors providing the data, not actual insights it gained from analyzing real cases."

Data Specification & Requirements II

Data has specific (probability) distributions on which the AI-component rests on. Define the:

- Expected and operational input distributions
- Training, validation, and testing data
- Define additional sources by obtaining hypotheses about data from stakeholder
- Data origins: users, platforms, sensors, environments, 3rd parties, etc.
- Data quality: correctness, trustworthiness, accuracy, noise, diversity, timely
- Data properties: privacy, accountability, accessibility, lack of bias,

Rational: Poor data quality and false / outdated labels may cause low AI performance. Leaking data from learning to testing may cause overestimation of AI performance. Failing to separate data origins may cause training on data that is not available in production.

Model Requirements & Specification

Specify model parameters and success metrics, as well as basic assumption on the type of model (e.g., supervised vs. unsupervised; online vs. batch learning; classification vs. regression).

Rational: Defining the type of model allows to select appropriate training data and deployment scenarios. Aligns with project goal and context specification. Specification of good success metrics and model parameters (e.g., size of the model) enables to validate that model improvement aligns with project goal.

Model Metrics beyond Accuracy

Typical model metrics do not reflect all requirements.

Do not state requirements on how your model performs on typical ML metrics, such as F1-score, log-loss, MSR, precision & recall, KL divergence, variance of error, etc.,

Instead formulate requirements on how your model might perform on variations and slices of data, on different learning and optimization procedures, on different thresholds for predictions and classifications, on misaligned or malicious data, etc.

Accuracy: How well the model performs on the task ⇐ Model ⇒ Non-functional model properties: How the model achieves the accuracies

Size -> inference time, memory usage
Interpretability -> Usage scenarios, human in the loop
Robustness -> Reduced risk, lower re-training frequency
Fairness -> Model accuracy on protected input features

From Requirements to Specification: Robustness

Idea: Specify robustness benchmark to test the deployed model against subsets of data that differ in a well-defined way

Differences to consider:
- Data from another time period
- Data from another geographic region
- Data from different set of users
- Data from different sensors
- Data from different origins (vendors, self-collected, production lines)
- Data from different environments (e.g., voice recordings with different background noise)

Important: Training is performed on a different data set (different in the mentioned characteristics / distributions)

Specification of Metrics & Monitoring Requirements I

Metrics and key performance indicators (KPIs) aim to measure the success and accuracy of the deployed AI component. Needs to directly be linked to the goals of software system (review slides on goals of the project).

Rational: Enable goal-oriented development and continuous improvement of the AI component. Determine when context specification becomes invalid during runtime. Feedback on KPIs in production.

Examples:

- Number of misclassifications reported by users
- Number of repeated commands in speech recognition
- Number of manually detected faulty products after AI inspection

Specification of Metrics & Monitoring Requirements II

Requires deep understanding of ML and statistics

- Statistical phenomena (e.g., Simpson’s paradox)
- ML evaluation metrics (precision, recall, F1, Lift)
- Understanding of prior data distributions to set target quality goals

Hard to communicate with customers (false expectations, lack of knowledge)

Focus on a single success metric to ease measuring process [Andrew Ng, DeepLearning.AI, 2017]

“A 2016 audit by the University of Texas found that the cancer center spent $62 million on the project before canceling it. A deeper look at these two projects reveals a fundamental mismatch between the promise of machine learning and the reality of medical care—between “real AI" and the requirements of a functional product for today's doctors.” (IEEE Spectrum, 2019, Eliza Strickland)

Non-functional / Quality Requirements of AI-Systems

Accountability	AI system need to be accountable for its decisions, actions, and effects on stakeholders
Controllability	Degree to which an external agent can intervene in the AI's processes (see Skynet)
Explainability	Ability of explaining how an AI system has reached to a certain output
Interpretability	Degree to which humans can reason on the results of an AI system
Reliability	Consistency of AI output (unlike reliably of a SW system, focusing on up-time)
Resilience	Ability to recover from malfunctions
Robustness	Degree to which a system is keeping functioning when an error occurs during execution
Transparency	Ability to follow all steps in the AI system leading to a decision, subsumes reproducibility
Performance	Either time related (execution / inference time) or model-quality related (accuracy, error rate, etc.)
Resources	Energy or memory consumption (especially for edge hardware)
Communication	Latency, throughput, request handling (especially for scaling in cloud systems)

Explainability Example

IBM Watson can either agree with the treatment of doctors or suggest an alternative. If agreeing, there is no benefit in using Watson. For an alternative suggestion, we would need to know why Watson disagrees with the Doctors statement. However, Watson could not explain the divergence, so the alternative couldn’t be followed. Hence, there is no benefit in using Watson.

For this insight, we do not have to implement Watson, but rather do a proper requirements analysis strongly coupled with the business value (i.e., project goal).

Explainability II

Rational: Need to explain what has been learned and why certain predictions have been made. Otherwise, we cannot trust, verify, and understand decisions made by AI.

Explainability depends on the used AI/ML technique. Classical techniques, such as linear regression or classification trees are easier to assess than neural networks. The importance of explainability might constrain the choice of technique.

Some scenarios require explanations more than predictive power, e.g., to find relations between input and output. Requirements should specify what and in which degree a prediction or model needs to be explained and whether we can sacrifice prediction accuracy for that.

Legal and Regulatory Requirements

The use of AI systems gets more and more regulated. It is paramount to define the project regulatory boundaries to deploy a legal product.

Rational: Processes must conform to regularity restrictions and may require proof for not using illegal features.

Examples: General Data Protection Regulation (GDPR) constraints that personal data can only be used in ways specified by an explicit consent.
In essence: We would need to know what the ML model delivers before we developing it.

Which features are relevant? Since we do not know in advance, we need consent for all potentially relevant features! But, if we get consent for all potentially relevant features, we usually require actually only a small subset, which in turn makes getting the consent for all unused features illegal again.

Regulations require to make it possible to remove personal data. The ability to remove this from a model is challenging without retraining the entire model.

Specification of Human Factors

Define how users will work with the AI component and react on its (mis-)behavior.

Rational: Users need to accept and trust AI decisions (see fail of IBM's DeepBlue for medicine).

Examples:

- How would an expert take an AI decision that contradicts her own?
- Will a driver of an autonomous car react quickly enough in situation that is not covered by the context specification? What warning time is needed?
- Can my AI voice assistant ask clarification questions and if so how often?
- Can the limited human capacity take quickly enough the information that is provided by the AI?

Ethical Requirements & Specification

The purpose of ML is to discriminate data and learn patterns. However, some forms of discrimination are unethical and must be avoided. Requirements for AI systems must be derived from ethical principles and codes.

Rational: We must consider ethical aspects from the very beginning of a project as ethical issues occur in all phases in the life cycle. Enables to validate whether the produced system adds to the ethical requirements.

Example:

- EU Ethics guidelines for trustworthy AI
- Fairness and bias wrt. minorities, genders, etc. (i.e., protected features/attributes)
- Decision making (autonomous driving, credit ratings, insurance pricing, etc.)
- See lecture on ethics later.

Elicit Ethical Requirements

Adapt from abstract requirements to the project's scope and context. Have a diverse team to be able to do that task in the first place.

Apply SE practice of stakeholder-based requirements elicitation:

- Determine all stakeholders who work with or are affected by the software (e.g., include pedestrians' view on autonomous cars -> not driving with full speed through a puddle)
- Determine ethical stakes of these stakeholders (e.g., what makes a driver good? Allow others to overtake? Always wait for pedestrians to let them cross the street?)

Remember: Detecting fairness issues and acting on them is only possibly if you are aware of them in the first place!

Specify Fairness Measures for the ML Project

Is my following process fair, bias-free, and does not discriminate against protected features?

Data collection and labelling: Origin of data sources, distribution of data, correctness of/bias-free labels

Typical counter measures:

- Over/Undersampling, but may lead to undesirable spurious correlations
- Obtaining distribution of minority attributes and collecting more data on that
- Maintain a balanced test set to counter it on the algorithmic side
- Re-weight the importance of minority data points
- Remove some protected attributes from the data, but keep proxy variables in mind

Modelling: Algorithm penalizing minority data points, learning on protected features

Typical counter measures:

- Min-Diff training to penalize model for differences in predictions on different distributions of data
- Rate constraint to ensure a lower bound of accuracy on subsets of data

Adaptation and Re-Learning I

Dr. Mark Kris, Memorial Sloan Kettering's lead Watson trainer:

He noted that treatment guidelines for every metastatic lung cancer patient worldwide recently changed in the course of one week after a research presentation at a cancer conference. "Changing the system of cognitive computing doesn't turn around on a dime like that," he said. "You have to put in the literature, you have to put in cases."

Adaptation and Re-Learning II

Depending on the application scenario, AI-systems may require frequent adaptations to the environment or re-learning due to changes in the data.

Rational: Specifying under which conditions an adaptation of the ML model is necessary, how frequent re-learnings are to be expected, and how substantial the model must adapt to new data is key to develop the right system.

Example:

- Regulatory changes (new laws, removal of access to features, etc.).
- Data and model drift (see later).
- Business case changes.
- Interactive system (based on immediate feedback).

Example Quality Requirements

Application scenario: Edge deployment / edge AI

- Requirements on memory consumption, computation power, energy consumption
- Frequency of model updates, retraining?

Application scenario: Cloud deployment

- Requirements on latency, scalability, inference time

Application scenario: Integrated deployment

- Requirements on hardware, inference time, memory consumption
- Bandwidth?, ...

Hardware Requirements

Specify the hardware setup for learning and inference and whether it runs locally, on an edge device, on end device, or at a data center. Define whether the HW needs to be built, bought, or rented. Distinguish between training and inference hardware. Take different deployment scenarios into account.

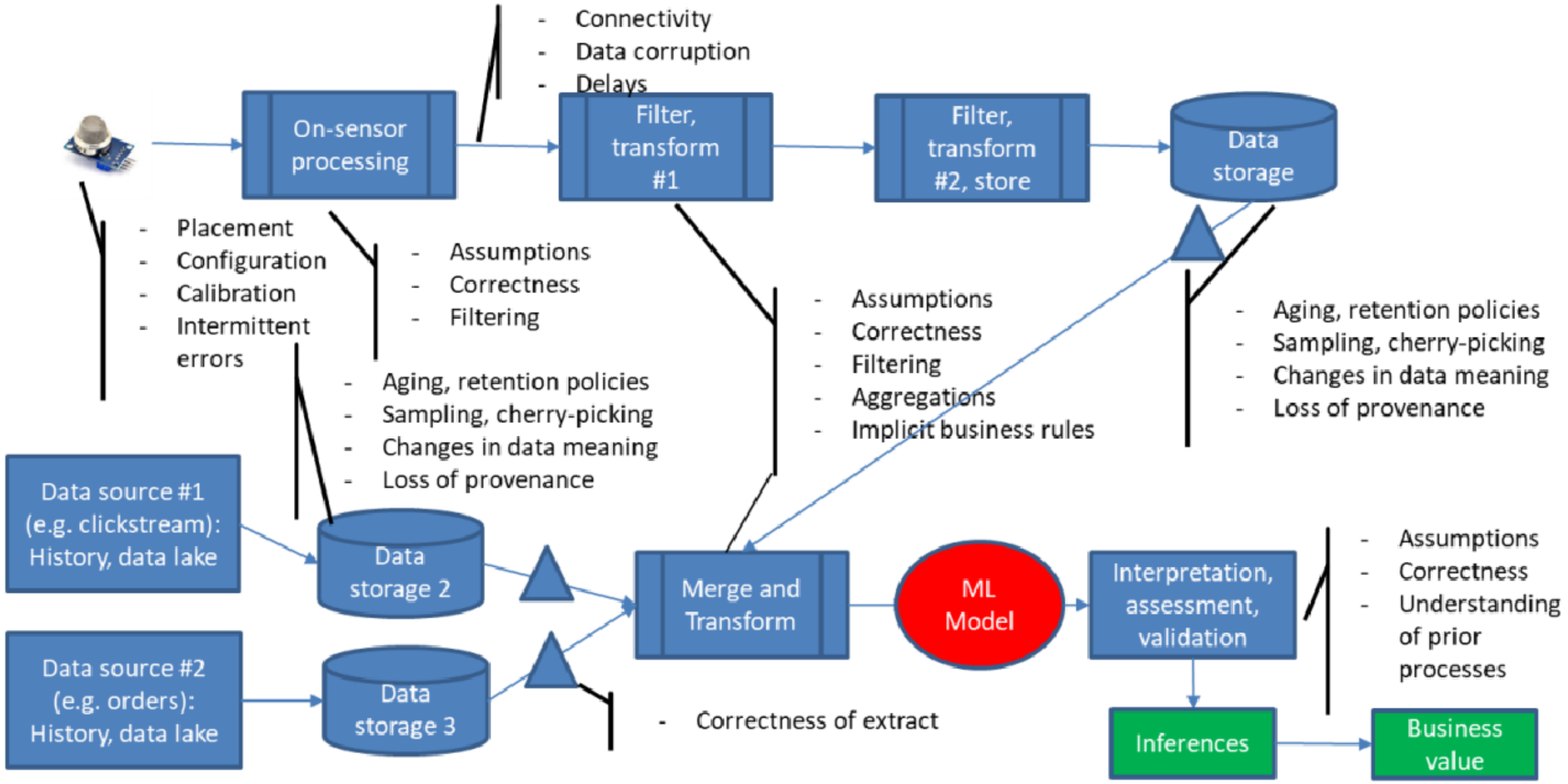
Rational: HW can be a significant cost driver and is crucial when scaling the AI component is necessary to achieve the project goal. The amount of data to handle may require special HW setups and affects other non-functional requirements (e.g., inference time).

Example:

- A huge language model is incompatible to offline on-device inference.
- A data center is inappropriate when connection is not guaranteed.

Document Data Catalog and Pipeline

Verify assumptions and requirements: Is the data from a data lake the same as the one in production?
Data catalog and pipeline documentation identifies potential threats to this assumption.



Source	Contents	Duration	Quantity	Comments
Data source #1: data lake	Clickstream data	Jan 2018–Jan 2019	1.6M	User IP address only; user name not known
Data source #2: data lake	Order history	June 1 2016–Oct 3 2018	55k orders	Format stored in changed on Jan 1 2018 Final order only (not change history) Orders with errors are deleted
Sensor data	Readings from factory sensors. Streaming data is batched and stored	90 days history retention only	50/sec; 5k/sec expected	Data cleaning unknown; is perceived outlier data being dropped?

Documenting Models Using Model Cards

- **Model Details**. Basic information about the model.
  - Person or organization developing model
  - Model date
  - Model version
  - Model type
  - Information about training algorithms, parameters, fairness constraints or other applied approaches, and features
  - Paper or other resource for more information
  - Citation details
  - License
  - Where to send questions or comments about the model
- **Intended Use**. Use cases that were envisioned during development.
  - Primary intended uses
  - Primary intended users
  - Out-of-scope use cases
- **Factors**. Factors could include demographic or phenotypic groups, environmental conditions, technical attributes, or others listed in Section 4.3.
  - Relevant factors
  - Evaluation factors
- **Metrics**. Metrics should be chosen to reflect potential real-world impacts of the model.
  - Model performance measures
  - Decision thresholds
  - Variation approaches
- **Evaluation Data**. Details on the dataset(s) used for the quantitative analyses in the card.
  - Datasets
  - Motivation
  - Preprocessing
- **Training Data**. May not be possible to provide in practice. When possible, this section should mirror Evaluation Data. If such detail is not possible, minimal allowable information should be provided here, such as details of the distribution over various factors in the training datasets.
- **Quantitative Analyses**
  - Unitary results
  - Intersectional results
- **Ethical Considerations**
- **Caveats and Recommendations**

Proposed by Google as a method for transparent model reporting, but could act as a basis for requirements collection.

Requirements Engineering Team

Requirements engineer
- RE methods (interviews, workshops, etc.)
- SE background (life cycle)
- ML basics (process, experimentation, etc.)

Data scientists
- Data quality requirements
- Trustworthiness, provenance

Legal person
- Regulatory knowledge
- Discrimination and fairness obligation

optional

UX / HCI designer / social scientist
- AI-Human interaction
- AI effect on human / society

Software engineer / ML expert
- Non-functional characteristics / deployment
- ML metrics / feasibility

Research Attempts: Specification of ML Components I

Rahimi et al.: Toward Requirements Specification for Machine-Learned Components

Task: Specify the ML component for an automated detection of pedestrians

Problem:

- Unambiguous specification is extremely difficult, since domain concepts and characteristics are often unknown, incomplete, or ambiguous
- Transfer from requirement to specification unclear: “position of a pedestrian must be accurate within 0.5m” -> how to decompose into *verifiable* specifications?
- Domain-specific context is often underspecified: “What is a pedestrian and in which environment does she act?” If this is unclear how can we assume to know what data to collect and which model to train and test?
- How can we trace the concept of pedestrian detection through data acquisition and model training?

Research Attempts: Specification of ML Components II

Rahimi et al.: Toward Requirements Specification for Machine-Learned Components

Idea: Step-Wise Process of Context Modelling und Verification

Domain Benchmark

Goal: Find a generic set of characteristics that matter in the perception of the concept

Example: What do people perceive to be important features of pedestrians?

Method: Analyze the Web for the keyword "pedestrian" and obtain relevant terms. Model and visualize the terms via an ontology. Manually review for completeness & correctness.

Dataset Interpretation

Goal: Analyze collected dataset to find characteristics of the concept within the data

Example: Analyze label information from pedestrian images (e.g., via Google vision).

Method: Obtain labels / terms from data analysis and align them in an ontology. Correlate the two ontologies to find mismatches, missing characteristics, or missing data.

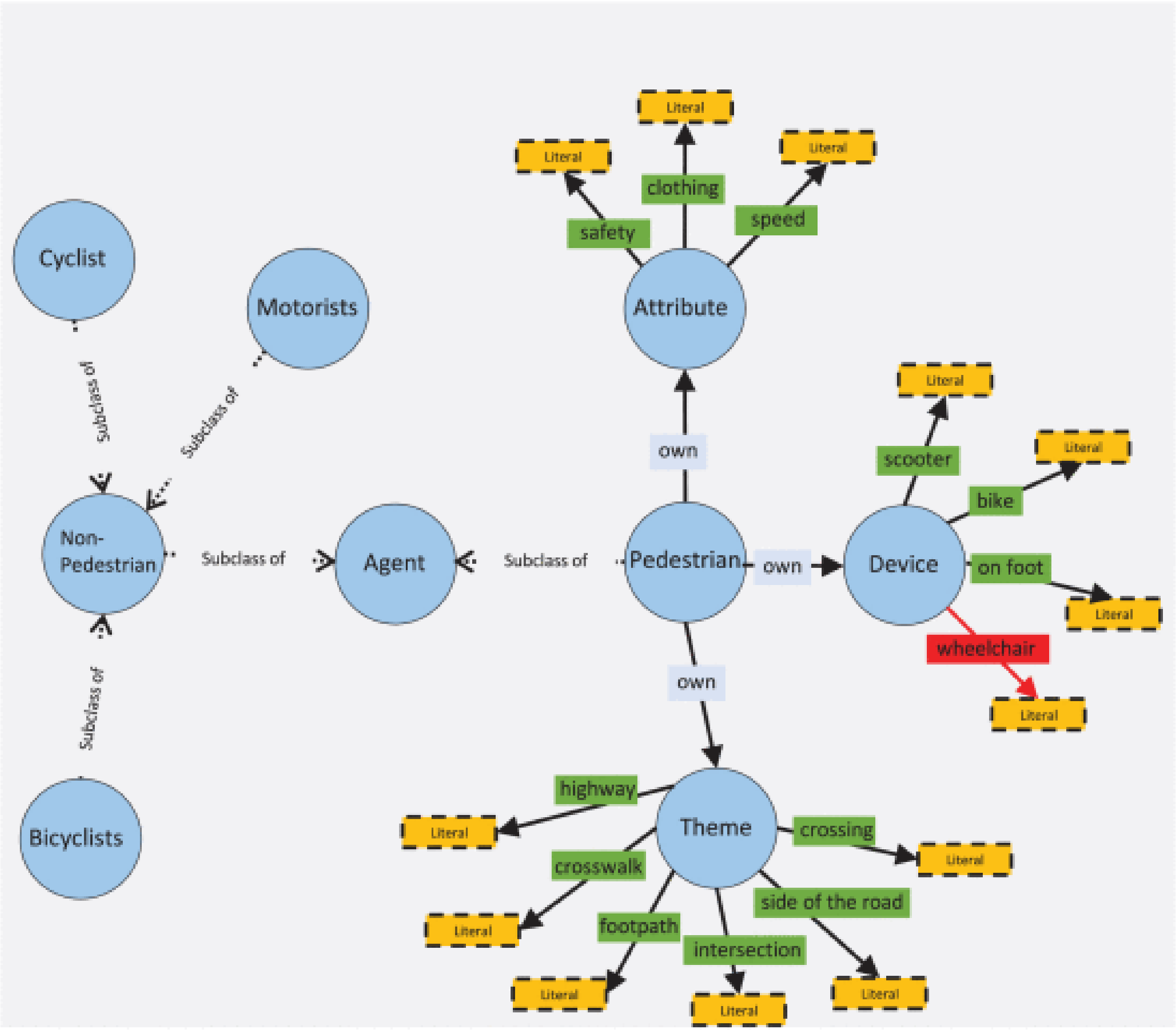
Model Interpretation

Goal: Analyze the learned model to understand its capability of representing the domain concept.

Example: Extract features from the pedestrian detection model and check whether they discriminate along identified domain characteristics.

Method: Interpretability techniques, weights of neurons and coefficients, etc. Correlate all ontologies and check for consistency.

Research Attempts: Specification of ML Components III



Check whether the dataset contains samples for all these cases.

↑

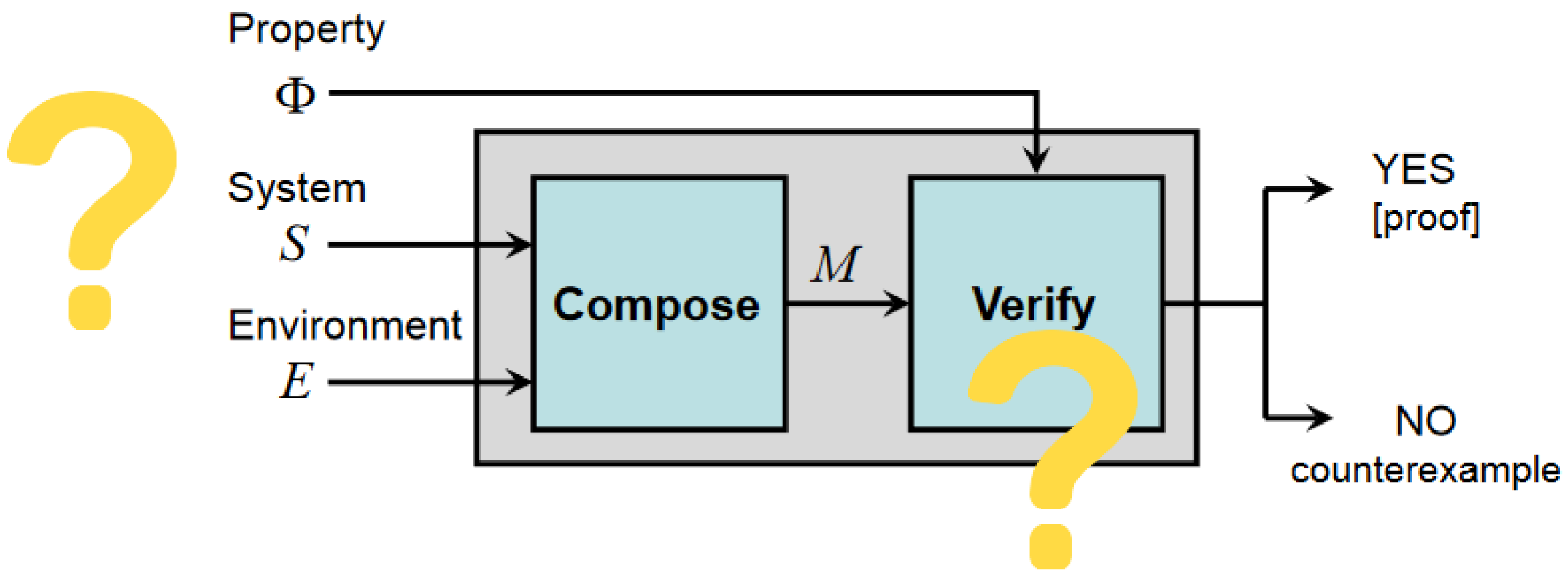
"The pedestrian detector component shall be able to detect pedestrians *on foot*, on a *scooter* and on a *wheelchair*"

↓

Verify specification to test whether component can detect all these situations.

Research Attempts: Towards Verified AI I

Idea: Design and specify an AI system in such a way that it has (provable) assurance of correctness via use of formal methods from software engineering.



Research Attempts: Towards Verified AI II

Input modelling: Environment (how to design an environment model?)

- Unknown variables: Requires a well-defined interface between the system S and the environment E. How to define all features of a vision or LiDAR sensor for autonomous driving that influence the system?
- Modeling right fidelity: requires formalism of uncertainty as probabilistic assumptions are being made by the model; existing logic models are highly over-approximate and render the verification useless
- Modeling human behavior: Tried for a long time with limited success. Requires guarantees for their prediction accuracy and convergences

Research Attempts: Towards Verified AI III

Input modelling: System (how to yield a model of tractable size?)

- High-dimensional input space: Not only high dimensional data, but also hybrid variables (discrete + continuous)
- High state space: State space defines the search space and millions of variables would make an approach infeasible
- Adaptation and evolution: Reinforcement learning (e.g., for robots) improve over time, but are incompatible with design-time specification and would require also adaptable specifications
- Modeling system context: ML models act only on a specified context, but formally describing it, is unclear.

Research Attempts: Towards Verified AI IV

Formal specification: a precise mathematical statement of what the system is supposed to do

- Hard-to-Formalize task: A component that distinguishes pedestrians from cars, etc. requires a formal definition of each type of concept
- Quantitative vs. Boolean specification: Unclear how to combine these formalisms
- Data vs. formal requirements: data is just ground truth behavior whereas formal specification is a mathematical property defining a set of correct behaviors. How to combine both?

New advances in formal methods need to be made (e.g., learning of probabilistic programs to model the system or environment) to enable a verified AI.

From Practice: Checklist by Michael Perlin

https://github.com/ttzt/catalog_of_requirements_for_ai_products

Problem

- What is the problem we want to solve?
- Which (strategic) business objective is it linked to?
- What are the current solutions/workarounds (if any)?
- Why do we think using machine learning will add value? (Links to similar cases in the industry, papers, research)
- Which parts of the system will use predictions? Which decisions/actions are possible based on the model's predictions? Which user journey(s) include using the model, and what impact does it have?
- What input must the model accept, and what output does it have to produce?

Metrics

- Which metric(s) will be used to measure the model's performance?
- What is/are the minimal value(s) of metrics for running the model in production?
- How are the performance measures aligned with the business objectives?
- What is the performance of the current solution?
- Is there a way to estimate the value added by machine learning by using historical data?

Dataset

- Any selection bias during data collection ? (explanation)
- Are there missing values? If so:
  - Do you know the causes?
  - Do they occur at random? (explanation)
- Are there any known issues with the correctness/accurateness of the data ?
- Where is data stored? Is it accessible from the infrastructure the DS team is using for training and serving?
- If data is structured or semi-structured: do you have documentation for each attribute?

Sources

Vogelsang & Borg: Requirements Engineering for Machine Learning: Perspectives from Data Scientists

Ishikawa & Yoshioka: How do engineers perceive difficulties in engineering of machine-learning systems? - Questionnaire survey

Laura Pullum: Verification and Validation of Systems in Which AI is a Key Element at SEBOOK

Sculley et al.:.Machine Learning: the high interest credit card of technical debt. In NIPS 2014 Workshop on Software Engineering for Machine Learning (SE4ML)

Goodfellow et al.:. Explaining and harnessing adversarial examples. In International Conference on Learning Representations (ICLR), May 2015.

Mitchell et al.: Model Cards for Model Reporting at Conference on Fairness, Accountability, and Transparency 2019.

Seshia et al. Verified AI at arxiv, 2020.

Guizzardi et al.: Ethical Requirements of AI Systems at 23rd Canadian Conference on Artificial Intelligence 2020.

D. H. Wolpert, “The lack of a priori distinctions between learning algorithms,” Neural computation, vol. 8, no. 7, pp. 1341–1390, 1996.

Heyn et al.: Requirements Engineering of AI-intensive Systems at WAIN'21 1st Workshop on AI Engineering

IEEE Spectrum: How IBM Watson overpromised and underdelivered on AI health-care.

Rahimi et al.: Toward Requirements Specification for Machine-Learned Components at International Requirements Engineering Conference Workshops, 2019.