Experiment: Network Anomaly Lifecycle

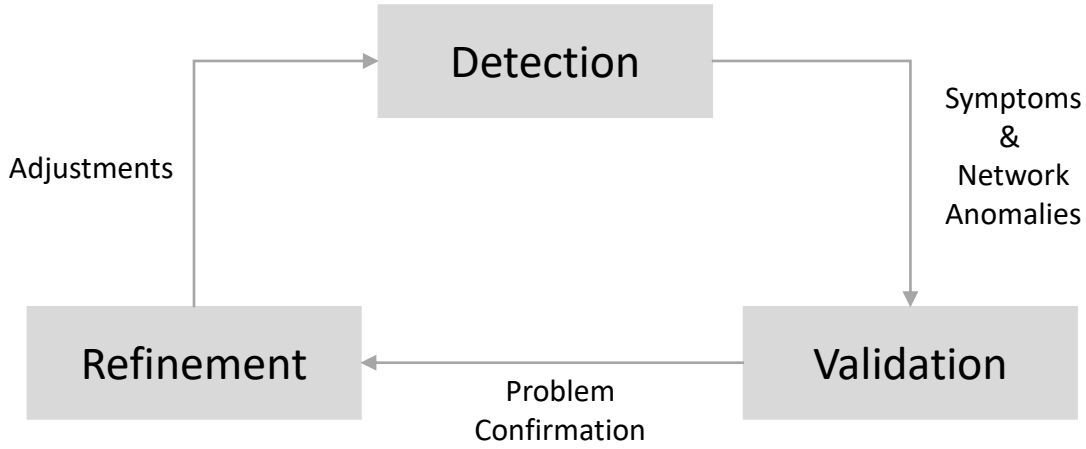
NMOP – Anomaly Detection Interim Meeting

11th September 2024

Presenter: Vincenzo Riccobene

Authors: Vincenzo Riccobene, Antonio Roberto, Thomas Graf, Wanting Du, Alex Huang Feng

Network Anomaly Detection Lifecycle

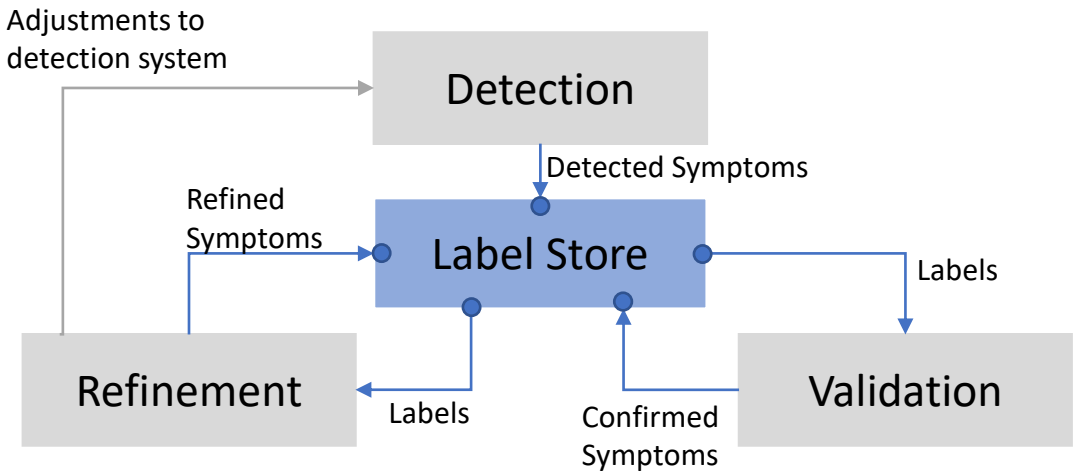


- Network anomaly detection requires **continuous learning** from new network behaviours
- Based on empirical evidence, a **lifecycle** has been identified as a suitable solution to **learn and incorporate** this learning iteratively into the anomaly detection process
- This learning can be **"codified"** by a set of labels
  - **Labels → symptoms**
  - Symptoms grouped into **network anomalies**
- Network anomalies can be updated (i) over time and in (ii) different stages of the lifecycle

  Goal: Provide more information to be fed into the detection stage

- Proposed solution: introduce a **label store** into the network anomaly architecture
- Functionality of the label store: **persistency**, **upgrade** and **retrieval** of labels for multiple actors across the 3 lifecycle stages
- **Actors**:
  - **In the Detection stage**: Network Engineers and/or Automatic detectors
    - Rule-based detectors
    - ML-based detectors
  - **In the Validation stage**: Network Engineers manually validating the labels
  - **In the Refinement stage**: Data Scientists and/or Automatic Refiners
    - Systems automatically refining detection systems, based on the validated labels
- Detectors and Refiners can be implemented by operators, vendors, etc.

Problem Statement (from Draft)



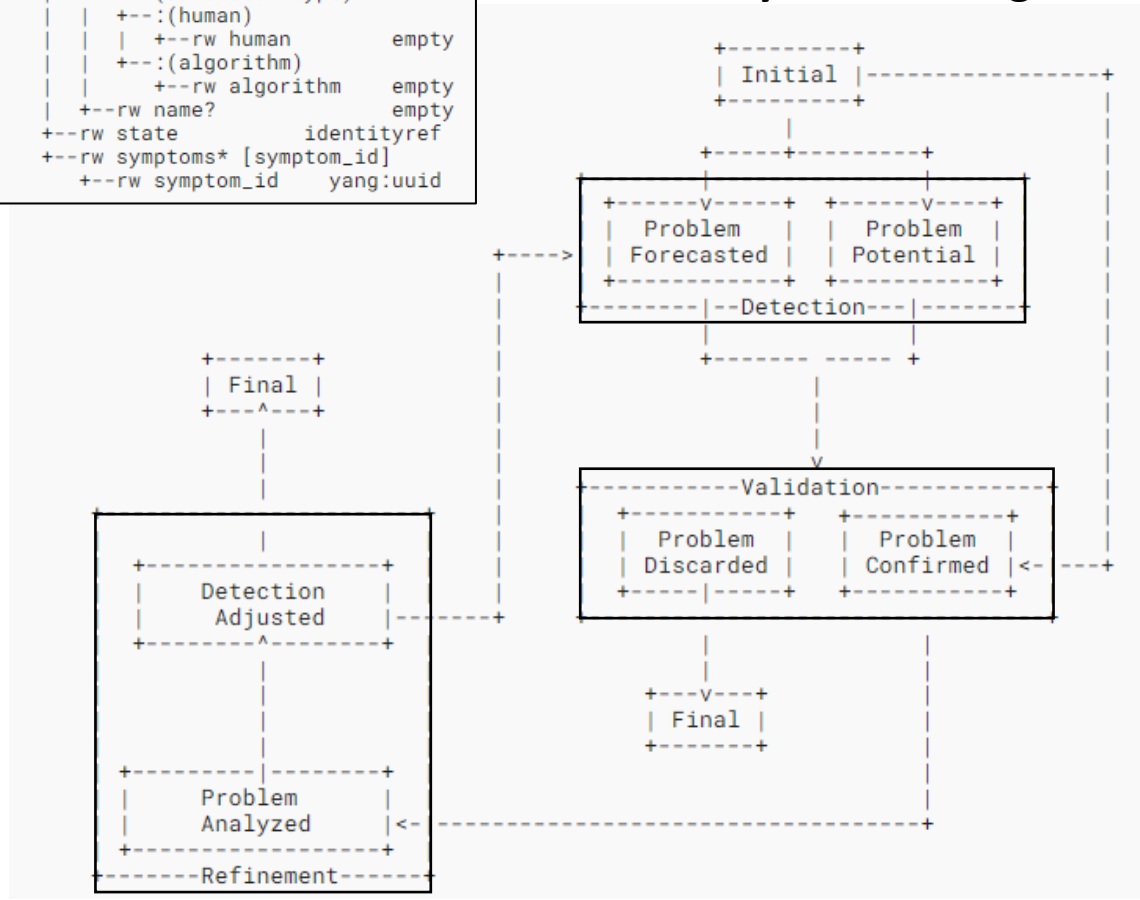
- The label store supports this exchange and it must expose an API to provide this service.
- The data model of the API (for the label exchange) is subject to the following requirements:
  - It must be **semantically consistent** from a networking point of view → let's use the semantic metadata draft
  - It must be both **human and machine readable**
  - It must allow different detection, validation and refinement solutions to **interoperate**
  - It must provide support for network experts **for validation and adjustment** of network anomaly labels

Any additional requirements worth adding to the list?

Data Model for data exchange

```
module: ietf-network-anomaly-metadata
  +--rw network-anomalies
     +--rw network-anomaly* [id version]
        +--rw id                yang:uuid
        +--rw version           uint32
        +--rw description?      string
        +--rw annotator
        |  +--rw (annotator-type)
        |  |  +--:(human)
        |  |  |  +--rw human          empty
        |  |  +--:(algorithm)
        |  |     +--rw algorithm      empty
        |  +--rw name?                empty
        +--rw state             identityref
        +--rw symptoms* [symptom_id]
           +--rw symptom_id     yang:uuid
```

State machine of the anomaly label management



Experiment Plan

Experiments Goals

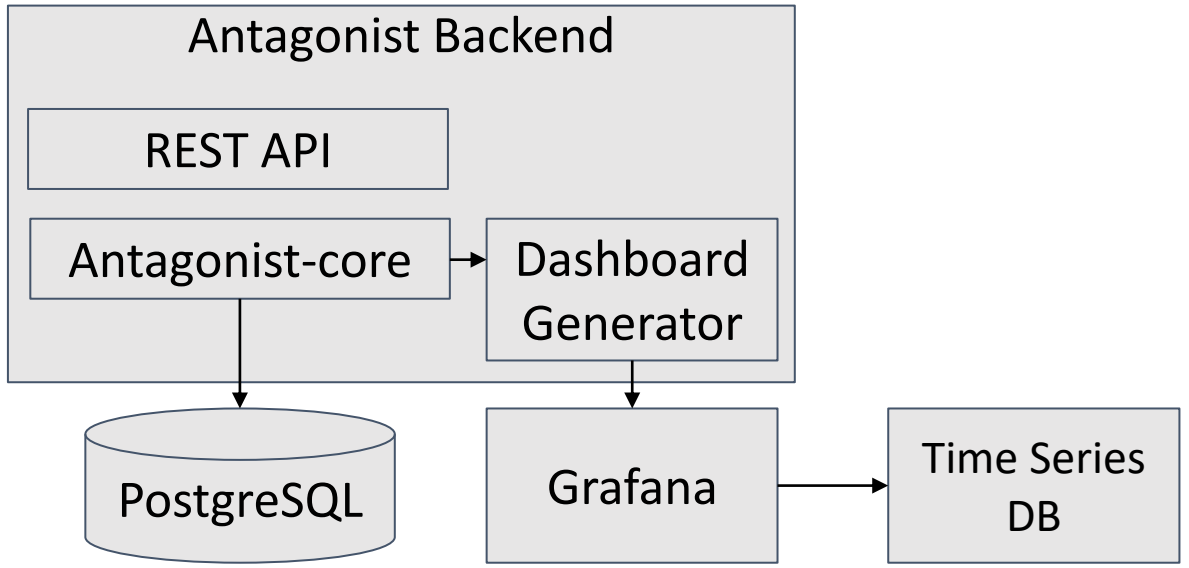
- Define and validate a suitable data model for label exchange between different actors and detection stages
- Validate the data model in a wide set of use case scenarios
- Validate the data model with real network data

Demonstrate interaction with different kinds of detection:
- **Human-based detection, validation and refinement**
- **Machine Learning -based detection**
- **Automated Rule-based detection** [Work in Progress]

Work done so far – Antagonist

Code Repo: https://github.com/vriccobene/antagonist
It includes: instructions for deployment and demo data
It's a PoC, so any help needed please get in touch

- Implement Anomaly Label persistency and retrieval
- Implement Anomaly Label exposure via the API
- Integrate with timeseries data
- Implement GUI for data exploration and validation
- Implement ML-based use case and integration
- Implement automatic dashboard generation



Human-based Detection

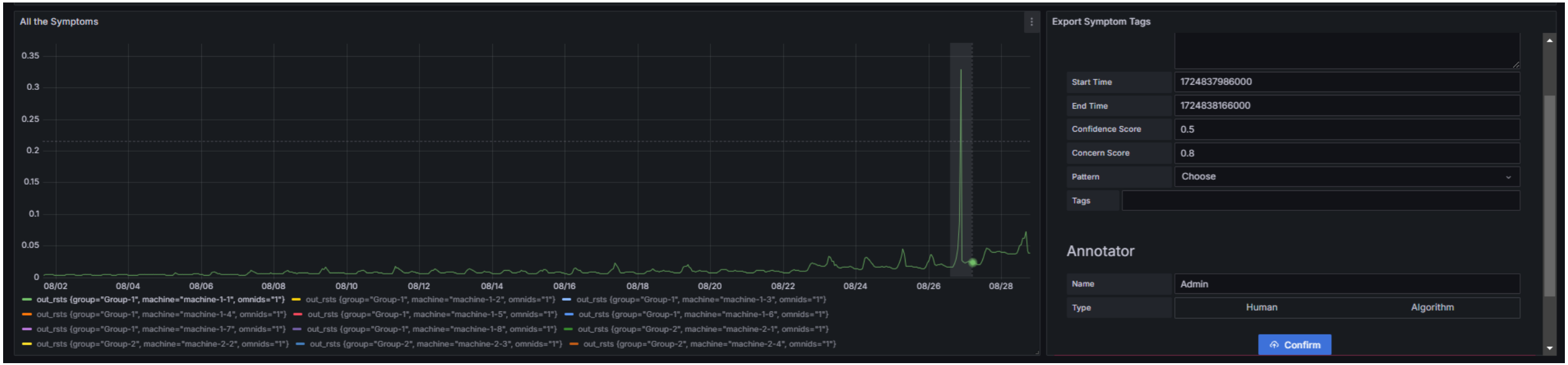
Using Antagonist, the user can **select a specific metric and tag new symptoms and network anomalies** specifying all the information defined in the data model, including the following two items (see draft-netana-nmop-network-anomaly-semantics-02):

- **Concern Score**: how badly the experienced symptom is service impacting

can be used to **prioritize symptoms**

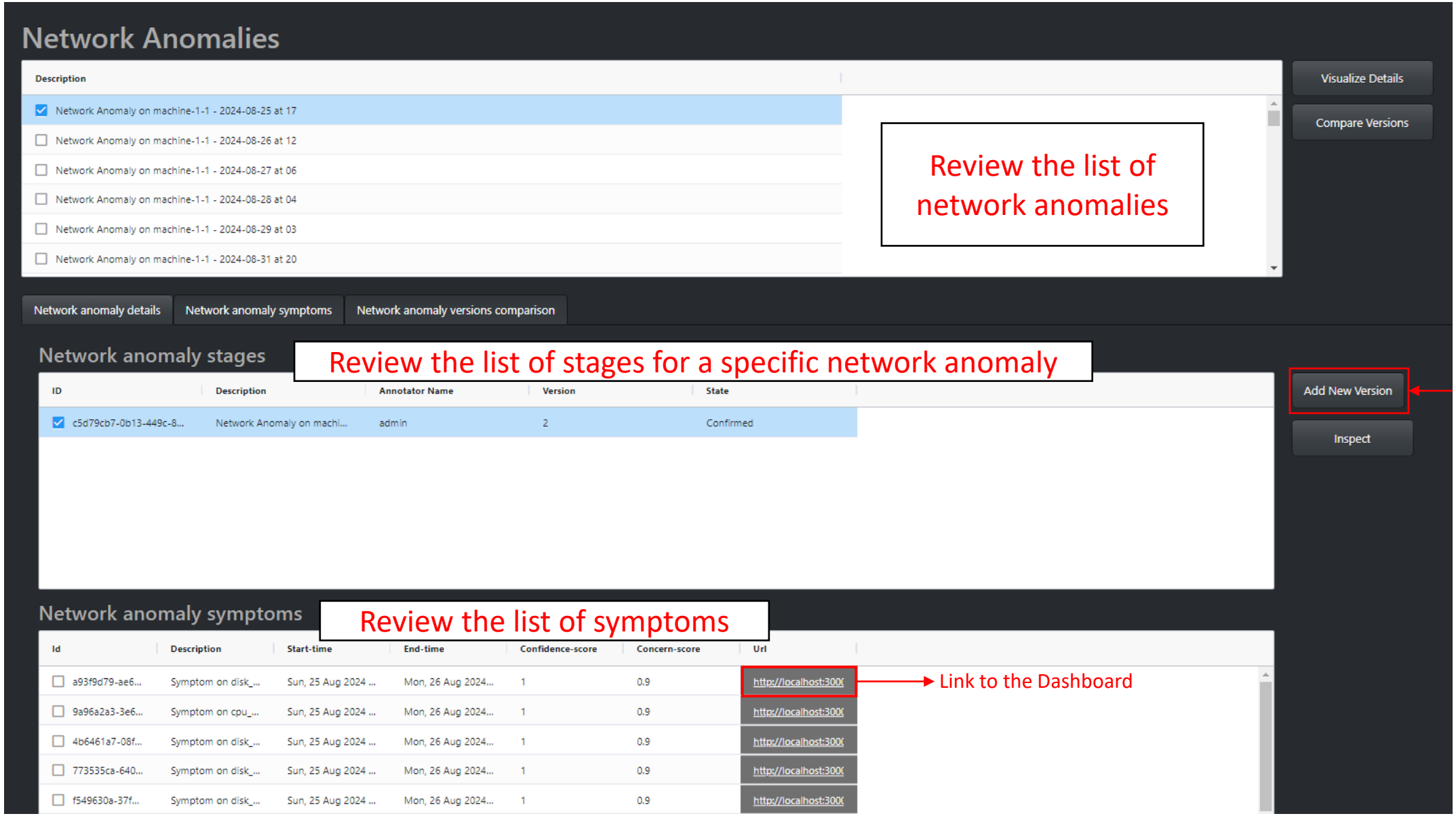
- **Confidence Score**: how sure you are this symptom is impacting or not

can be used to identify symptoms that **require validation**



Currently this is done in Grafana, but via the REST API any tool can be integrated

Human-based validation



Add new stages for the network anomaly (Add new symptoms and remove existing ones)

Human-based validation

Symptom Time Series Dashboard

Check the details of the symptom annotation

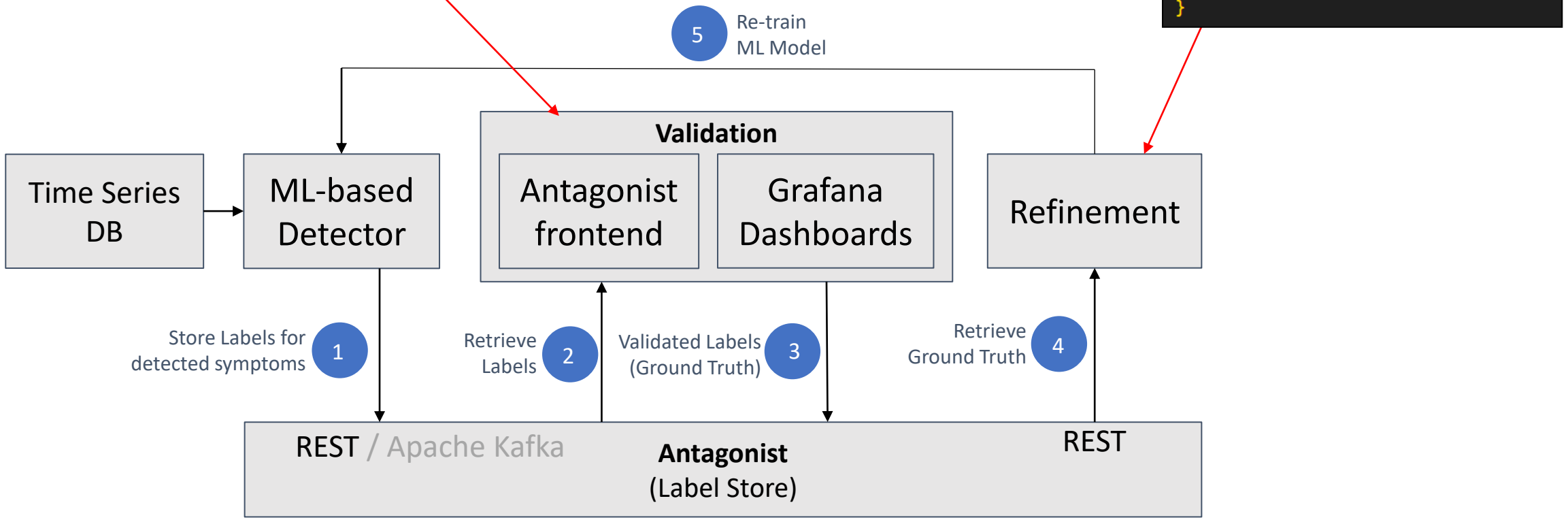
Machine Learning based detection and refinement

A Network Engineer has to validate only those symptoms for which the ML had low confidence (Active Learning)

```
# Get Labels for Validation
query_params={
    "start_time": "2024-07-01T19:41",
    "end_time": "2024-07-01T20:41",
    "min-confidence-score": 0.2,
    "max-confidence-score": 0.8
}
```

A Data scientist only wants to use the symptoms for which we have high confidence and high service impact in order to achieve high detection accuracy

```
# Get Ground Truth for Retraining
query_params={
    "start_time": "2024-07-01T19:41",
    "end_time": "2024-07-01T20:41",
    "min-confidence-score": 0.8,
    "min-concern-score": 0.5,
}
```



Experiment Roadmap

What was achieved so far

✓ Validation of Human label management

✓ Validation of ML-based anomaly detection

❑ Validation of rule-based *(WIP – data model validation)*

What needs to be finalized

- Formalize connection between metrics and symptoms
- Improve performance and scalability
- Finalize YANG data model validation

Roadmap

✓ Phase 1: Implement very basic PoC for data retrieval, API exposure and GUI (based on Grafana)

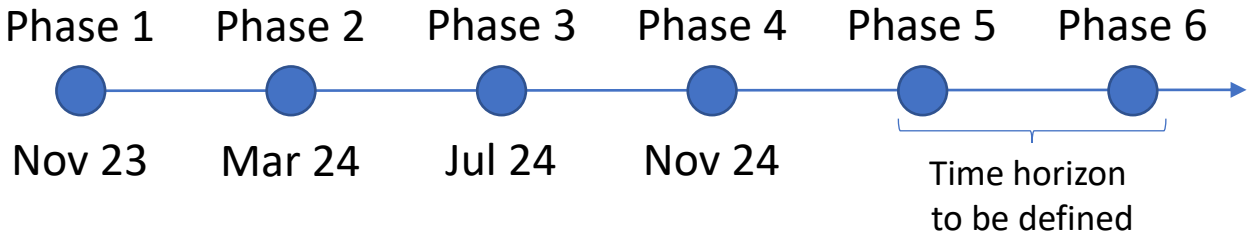
✓ Phase 2: Enhance GUI and extend API

✓ Phase 3: Validate the PoC with AIOps related data and a ML-based anomaly detector

❑ **Phase 4: Finalize validation of the PoC with SAIN [RFC 9417-9418] (as a rule-based anomaly detector)**

❑ Phase 5: Integrate with Swisscom Lab Environment

❑ Phase 6: Finalize YANG data model



❑ **Validation of the approach with more operators:**

- **Does the process fit with your processes?**
- **Does the data model support your use cases?**
- **Is this problem something you would like to cooperate?**