

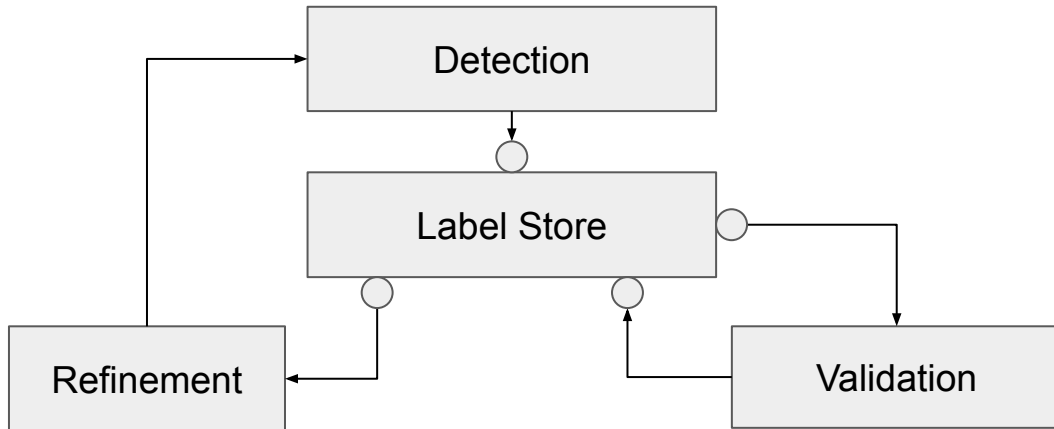
Antagonist

Network Anomaly Lifecycle and Semantic

NMOP WG - IETF 123 - Madrid (July 2025)

Network Anomaly Lifecycle - Antagonist

The Label Store is responsible to **persist and expose network anomaly labels** in a standardized format to support **all the stages of the life cycle**.



The goal of Antagonist is to provide a **single pane of glass** to:

1. Persist Anomaly Labels
2. Understand how Anomaly Detection is performing
3. Validate the labels generated and provide feedback
4. Refine the detection behaviour to improve results

Antagonist - What has been done for this iteration

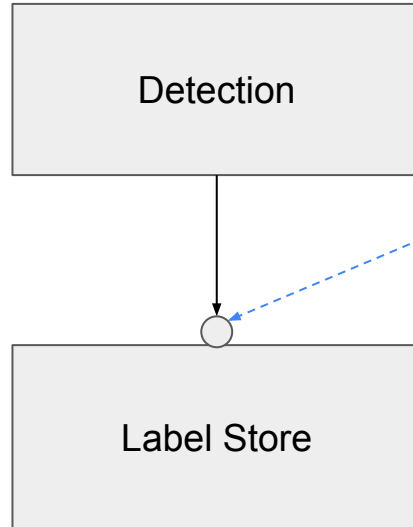
Experiment: Validating the draft while developing the Label Store

- Started refactoring Antagonist from PoC to production-ready
 - Implemented APIs using fastapi
 - Implemented DB integration using SQLAlchemy
- Validation of the PoC based on data from Swisscom
 - The data has been analysed to identify the key indicators and symptoms
 - **While trying to integrate the data, the need to make a clear reference to operational data was identified**
- Started the implementation of an improved GUI to support all the necessary use interaction to support the Lifecycle

The refactored code will soon be released in open source (some testing is still ongoing).

Antagonist Data Model - Detection

The detection is performed by an automated algorithm, which detects anomalies and produces notifications, which are persisted in the Label Store



The detection interface has been validated in the past with multiple Detection models

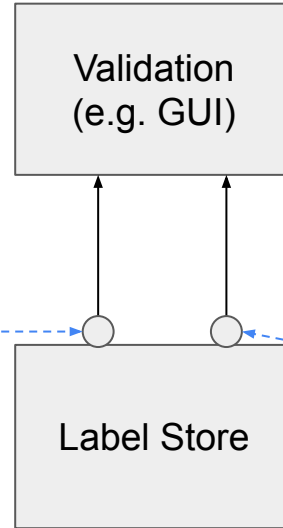
```
notifications:
+---n relevant-state-notification
  +--ro publisher
  | +--ro id?          yang:uuid
  | +--ro name        string
  | +--ro version?    string
  +--ro id            yang:uuid
  +--ro uri?          inet:uri
  +--ro description?  string
  +--ro start-time    yang:date-and-time
  +--ro end-time?     yang:date-and-time
  +--ro strategy?     string
  +--ro confidence-score? score
  +--ro concern-score  score
  +--ro (service)?
  +--ro anomaly* [id revision]
    +--ro id          yang:uuid
    +--ro revision    yang:counter32
    +--ro uri?        inet:uri
    +--ro state       identityref
    +--ro description? string
    +--ro start-time  yang:date-and-time
    +--ro end-time?   yang:date-and-time
    +--ro confidence-score? score
    +--ro pattern?    identityref
    +--ro annotator
    | +--ro id?        yang:uuid
    | +--ro name       string
    | +--ro version?   string
    | +--ro annotator-type? enumeration
    +--ro symptom!
      +--ro id          yang:uuid
      +--ro concern-score  score
```

Antagonist Data Model - Validation

In order to perform the validation, the user needs to **visualize the anomalies**, and **make changes** as necessary

Retrieve all the relevant states in a given time window

```
+--rw relevant-state
  +--rw id                yang:uuid
  +--rw uri?              inet:uri
  +--rw description?     string
  +--rw start-time       yang:date-and-time
  +--rw end-time?        yang:date-and-time
  +--rw strategy?        string
  +--rw confidence-score? score
  +--rw concern-score    score
  +--rw (service)?
  +--rw anomaly* [id revision]
    +--rw id                yang:uuid
    +--rw revision          yang:counter32
```

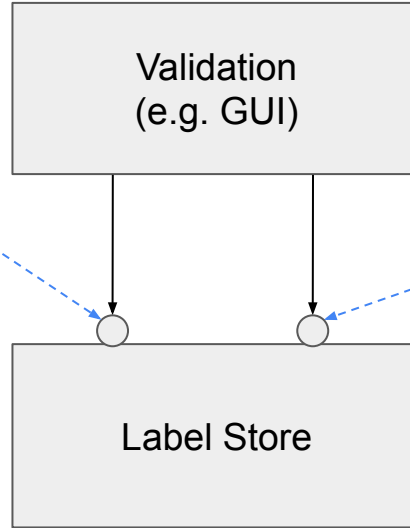


Retrieve all the anomalies in a given time window

```
+--rw anomaly* [id revision]
  +--rw id                yang:uuid
  +--rw revision          yang:counter32
  +--rw uri?              inet:uri
  +--rw state             identityref
  +--rw description?     string
  +--rw start-time       yang:date-and-time
  +--rw end-time?        yang:date-and-time
  +--rw confidence-score? score
  +--rw pattern?         identityref
  +--rw annotator
  | +--rw id?             yang:uuid
  | +--rw name            string
  | +--rw version?        string
  | +--rw annotator-type? enumeration
  +--rw symptom!
    +--rw id              yang:uuid
    +--rw concern-score  score
```

Antagonist Data Model - Validation

```
+--rw anomaly* [id revision]
+--rw id                yang:uuid
+--rw revision          yang:counter32
+--rw uri?              inet:uri
+--rw state             identityref
+--rw description?     string
+--rw start-time       yang:date-and-time
+--rw end-time?        yang:date-and-time
+--rw confidence-score? score
+--rw pattern?         identityref
+--rw annotator
| +--rw id?            yang:uuid
| +--rw name           string
| +--rw version?       string
| +--rw annotator-type? enumeration
+--rw symptom!
+--rw id                yang:uuid
+--rw concern-score    score
```



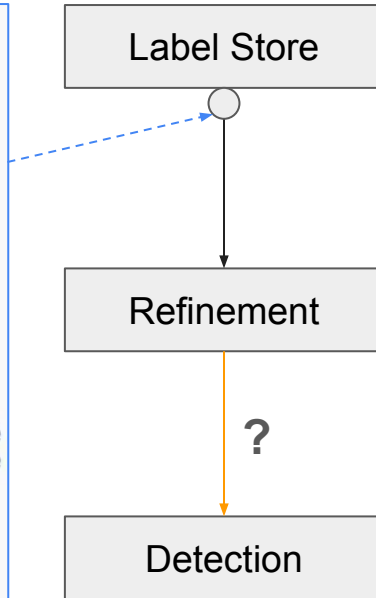
Once the changes have been made from the user, these can be persisted on the Label Store again to keep track of the considerations made during post-mortem.

```
+--rw relevant-state
+--rw id                yang:uuid
+--rw uri?              inet:uri
+--rw description?     string
+--rw start-time       yang:date-and-time
+--rw end-time?        yang:date-and-time
+--rw strategy?        string
+--rw confidence-score? score
+--rw concern-score    score
+--rw (service)?
+--rw anomaly* [id revision]
+--rw id                yang:uuid
+--rw revision          yang:counter32
+--rw uri?              inet:uri
+--rw state             identityref
+--rw description?     string
+--rw start-time       yang:date-and-time
+--rw end-time?        yang:date-and-time
+--rw confidence-score? score
+--rw pattern?         identityref
+--rw annotator
| +--rw id?            yang:uuid
| +--rw name           string
| +--rw version?       string
| +--rw annotator-type? enumeration
+--rw symptom!
+--rw id                yang:uuid
+--rw concern-score    score
```

We might need to consider adding the fields “revision” and “state” to the relevant state too.

Antagonist Data Model - Refinement

```
+--rw relevant-state
+--rw id                yang:uuid
+--rw uri?              inet:uri
+--rw description?     string
+--rw start-time       yang:date-and-time
+--rw end-time?        yang:date-and-time
+--rw strategy?        string
+--rw confidence-score? score
+--rw concern-score    score
+--rw (service)?
+--rw anomaly* [id revision]
  +--rw id                yang:uuid
  +--rw revision          yang:counter32
  +--rw uri?              inet:uri
  +--rw state             identityref
  +--rw description?     string
  +--rw start-time       yang:date-and-time
  +--rw end-time?        yang:date-and-time
  +--rw confidence-score? score
  +--rw pattern?         identityref
  +--rw annotator
  | +--rw id?             yang:uuid
  | +--rw name            string
  | +--rw version?       string
  | +--rw annotator-type? enumeration
+--rw symptom!
  +--rw id                yang:uuid
  +--rw concern-score    score
```

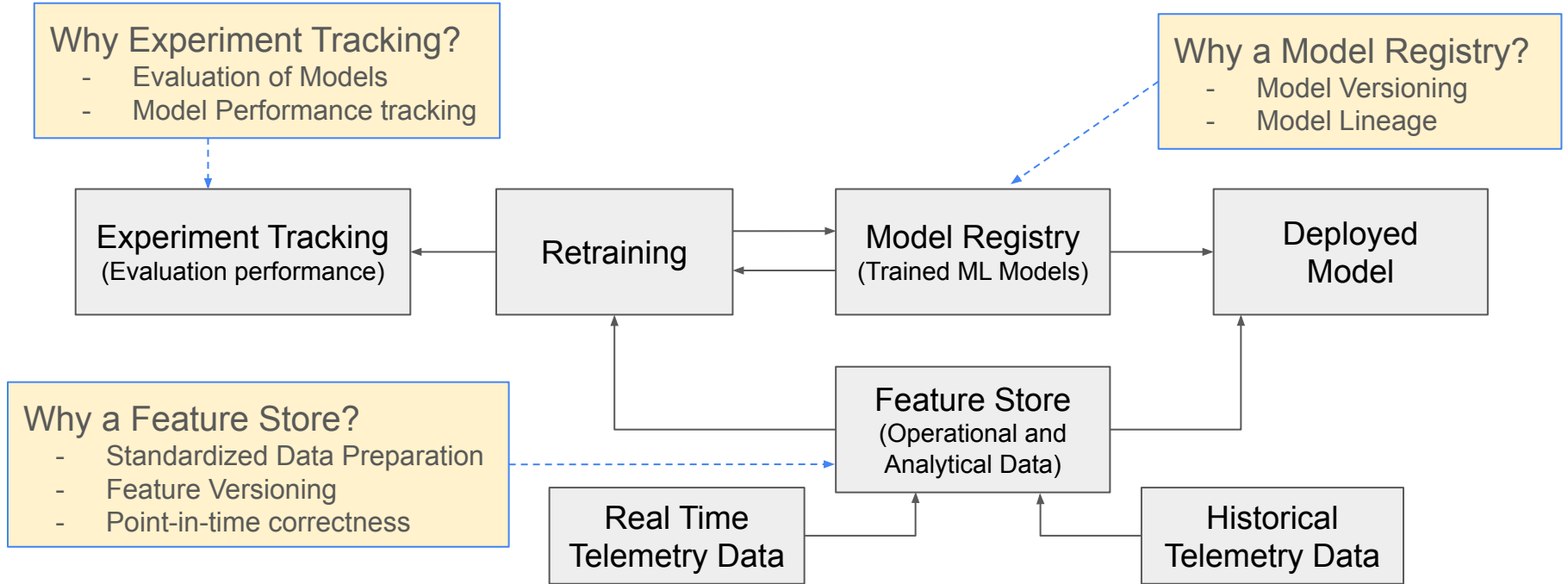


The exposure of information to the Refinement stage **might require some extra information** (e.g. found out about the need to have an explicit link to operational data only during development phase)

TODO: Explore the internals of the Refinement process to finalize the validation of the data model to be used for refinement.

Based on the validation so far, there is no need to standardize the interface between Refinement and Detection. **Any different opinion?**

Antagonist - What's coming next



Next Steps

- Finalize the validation of the Data Models
 - Continue with the integration of Swisscom data on Antagonist to check that all the details have been taken into account
 - Finalize the validation of all the models, by performing a full end-to-end test with Swisscom data
- Project Development
 - Finalize testing of the backend
 - Finalize the GUI implementation and integration
 - Finalize the implementation of the Refinement infrastructure

Thanks!

Team:

Vincenzo Riccobene

Thomas Graf

Wanting Du

Benoit Claise

Do you have any:

- feedback?
- suggestion?
- feature request?

Feel Free to get in touch!

