Natural Language Interface to Knowledge Graph

NLI2KG

Enzo FENOGLIO
Emerging Technology & Incubation – Cisco France
Research Fellow in Computer Science at UCL

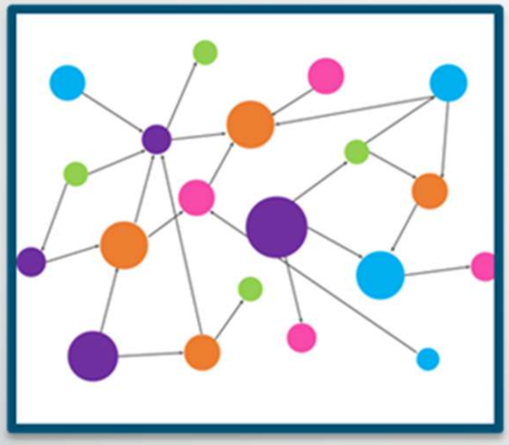
Berlin - Nov 17, 2022

The Problem

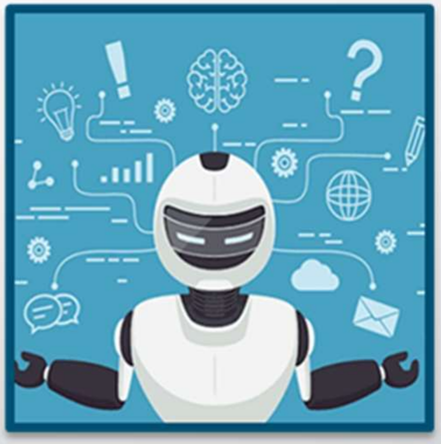
How to represent knowledge for sparse heterogeneous datasets ?

How to query for knowledge discovery w/o knowing a query language?

The Solution

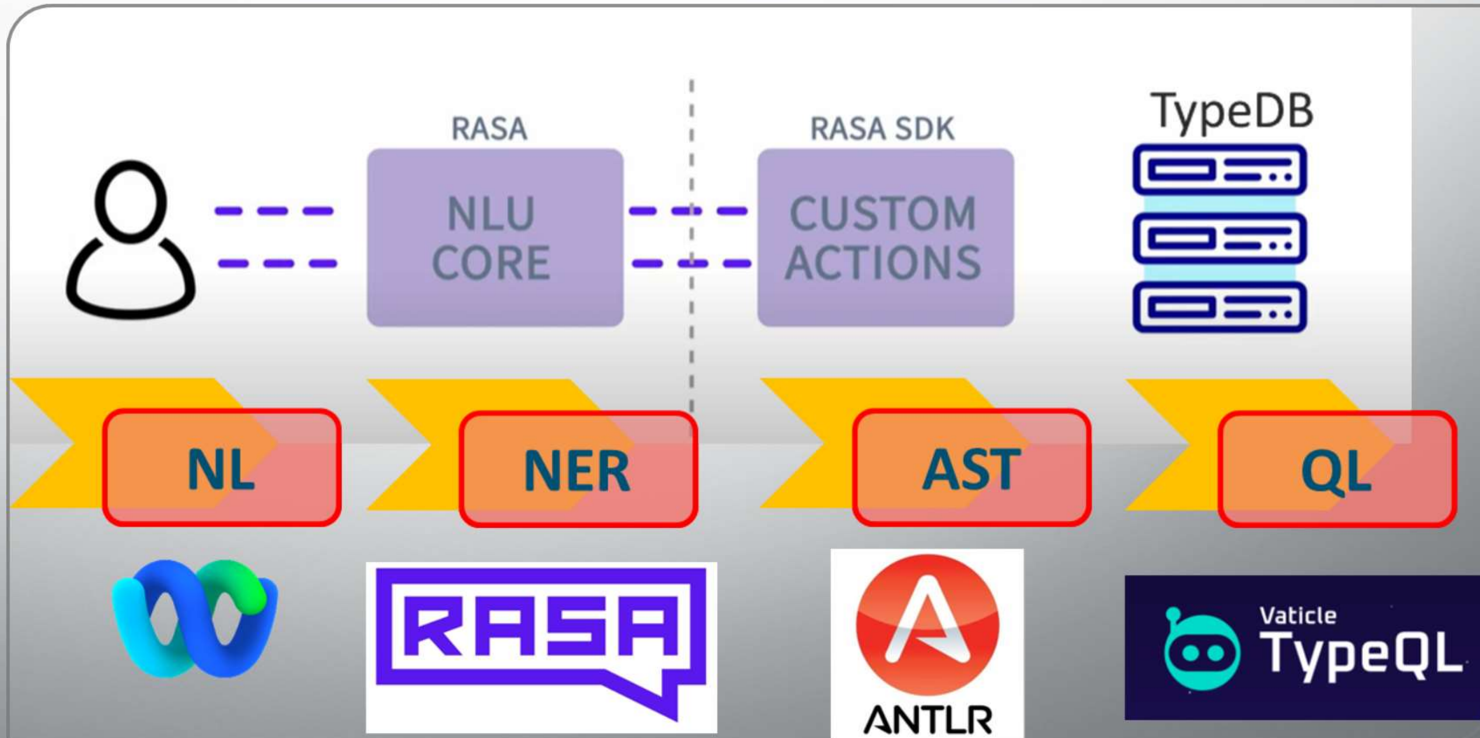


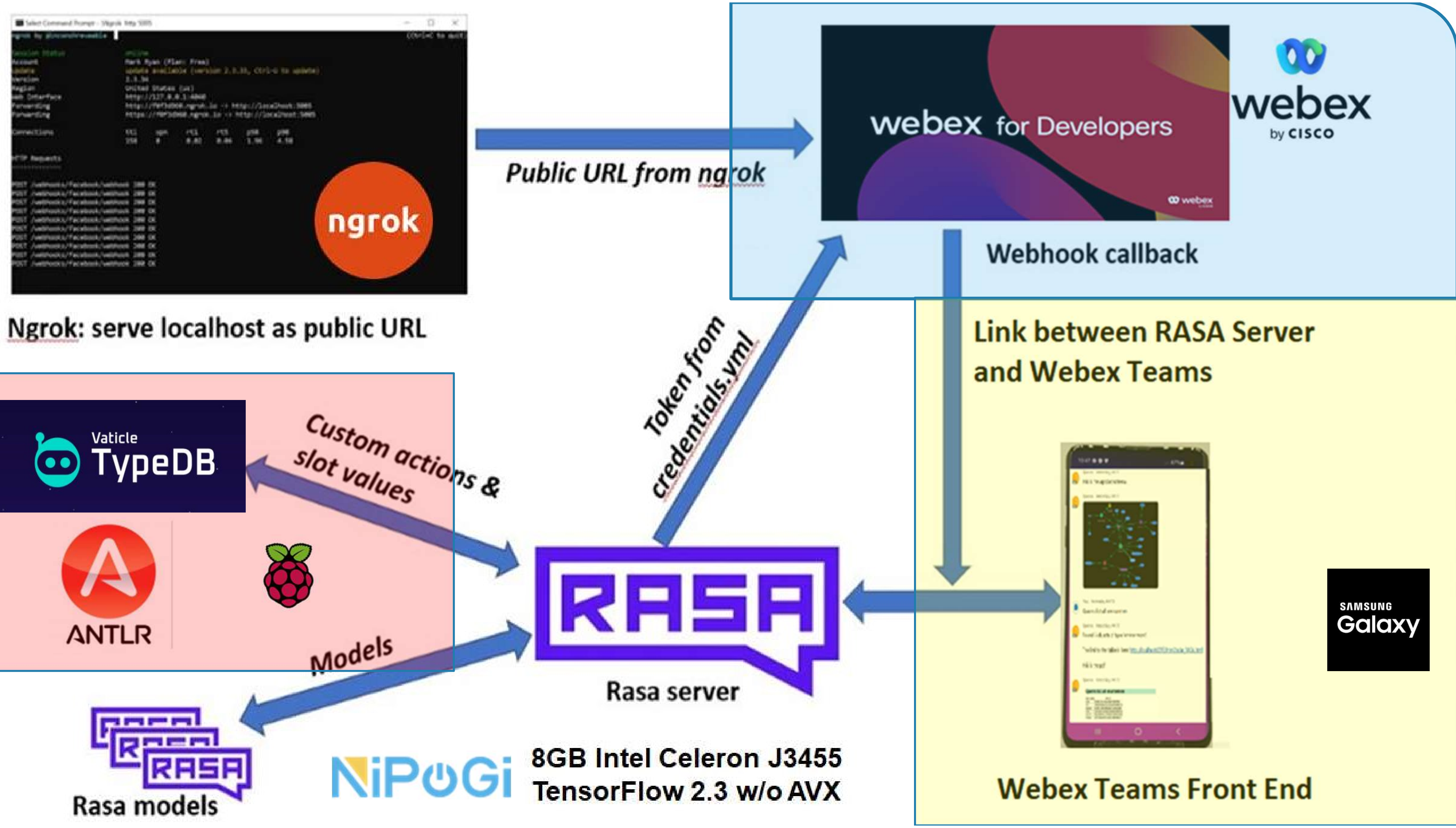
Store data + ontology
in a Knowledge Graph



Use NL to query the KG
using the built-in reasoner

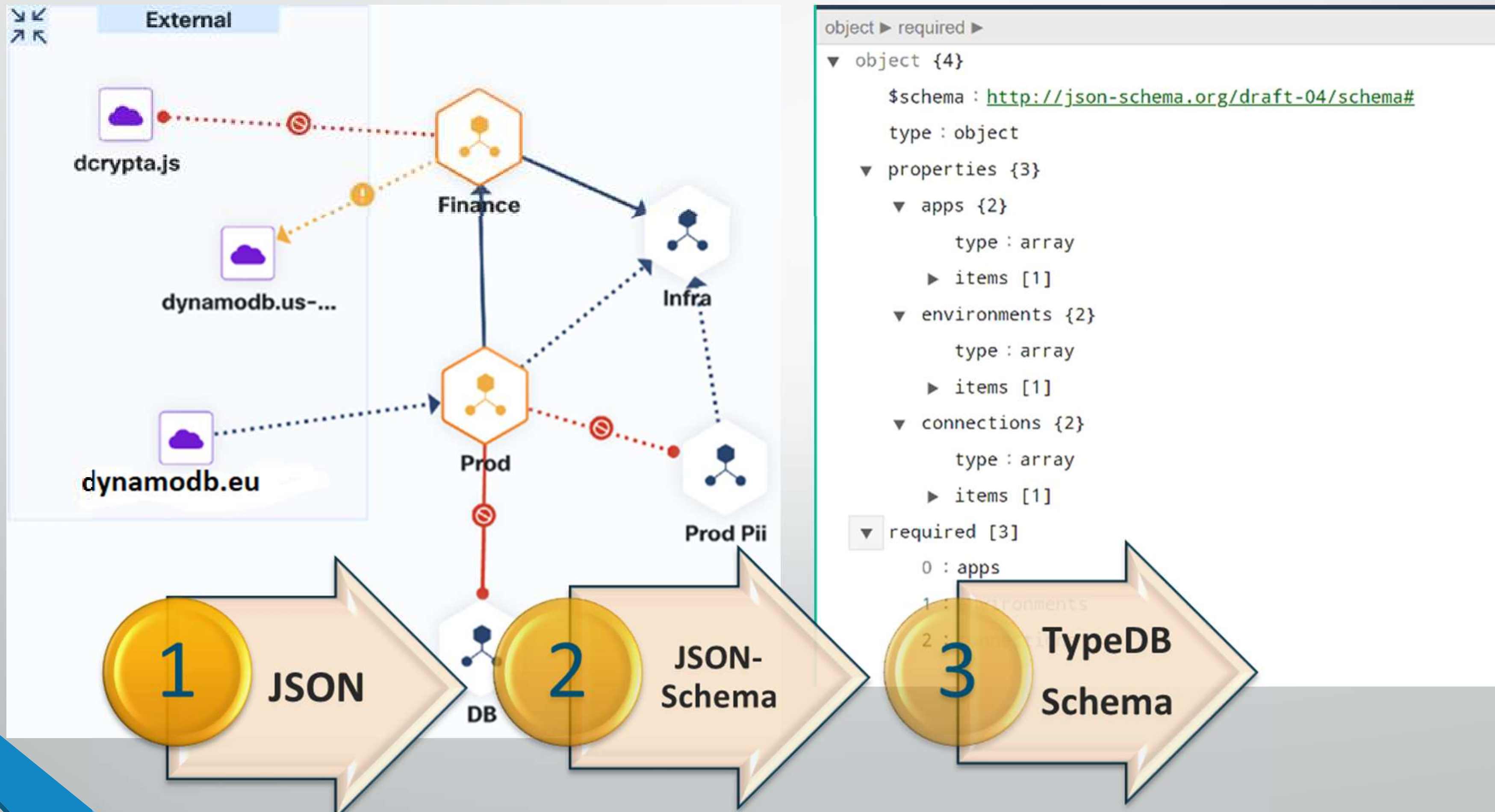
The bridge from NL to KG





ngrok
Ngrok: serve localhost as public URL
Public URL from ngrok
webex for Developers
webex
by CISCO
Webhook callback
Link between RASA Server
and Webex Teams
Token from
credentials.yml
Custom actions &
slot values
Vaticle
TypeDB
ANTLR
Models
RASA
Rasa server
Rasa models
NiPoGi
8GB Intel Celeron J3455
TensorFlow 2.3 w/o AVX
SAMSUNG
Galaxy
Webex Teams Front End

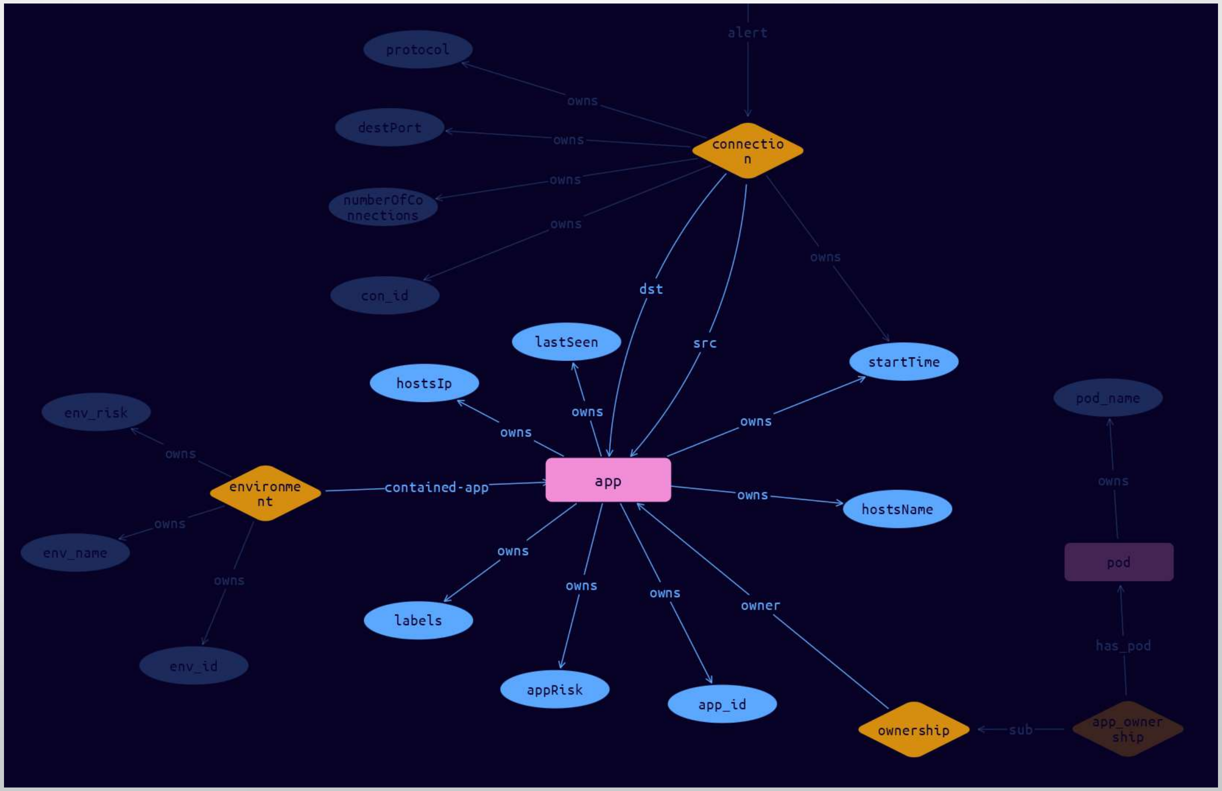
Data and Ontology



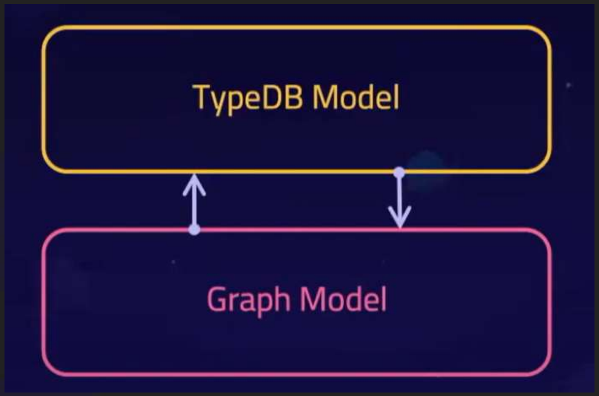
The Knowledge Graph Schema

```
app sub entity,
   owns app_id,
   owns lastSeen,
   owns startTime,
   owns appRisk,
   owns hostsIp, owns hostsName,
   owns labels,
   plays connection:src,
   plays connection:dst,
   plays environment:contained-app,
   plays app_ownership:owner;

env_risk sub attribute, value string;
env_id sub attribute, value string;
environment sub relation,
    owns env_name,
    owns env_risk,
    owns env_id,
    relates contained-app;
```



Why TypeDB ?



Open Source: Great documentation & Community support

Implements a concept level entity-relationship model

Implements hypergraphs data model with schema/logical verification

Provides a reasoning engine for reasoning at query time

RASA / Cisco Webex BOT API solution

- access token
- room ID
- webhook

```
$ typedb server
```

-to start the typedb server

```
$ ngrok http --region=eu 5005
```

-to redirect local traffic to port 5005

```
$ python getWebhook.py
```

-to create the Webhook for RASA endpoints

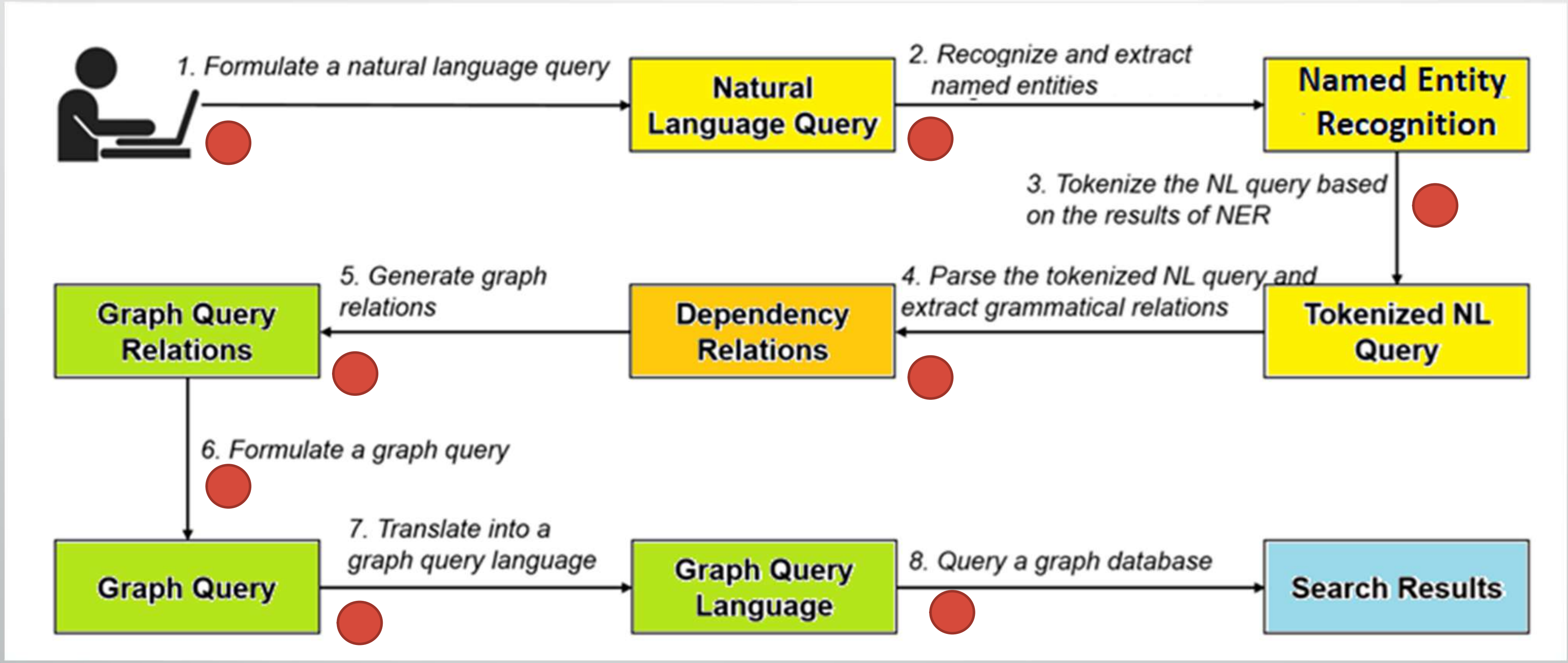
```
$ rasa run  --enable-api
```

-to start RASA server on port 5055 and enable APIs

```
$ rasa run  actions
```

-to allow communication from RASA Custom Action server to **TypeDB**

Query Translation Pipeline



The method: working details

STEP 1 (NER extraction)

- **NER** semantic parsing is done with a [RASA pipeline](#) and a simplified scheme (ontology) structure for semantic extraction in the form of a **JSON** variable:

[{“object_type”:<value>,”slot”:<value>,”role”:<value>,”value”:<value>},

{“object_type”:<value>,”attribute”:<value>,”role”:<value>}, {. . .}]

- **"object_type"** : a distinctive concept (nodes) in the graph base
- **"slot"** : a qualified attribute assigned to a concept
- **"attribute"** : a generic attribute assigned to a concept
- **"role"** : a relationship modifier such as ["from", "to", "max", "min", ...]
- **"value"** : a qualified instance of a slot

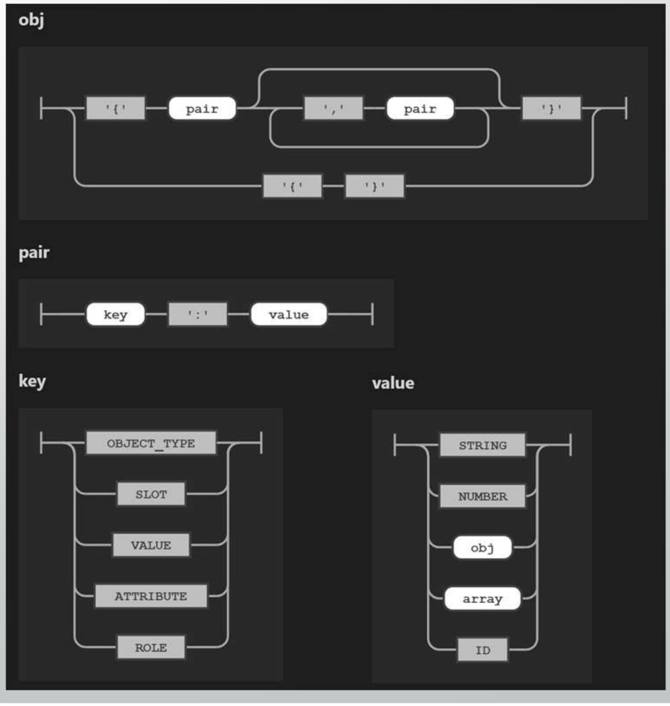
https://www.antlr.org/

STEP 2 (AST generation)

- A parse generator (**ANTLR)** is used to build an intermediate abstract syntax tree (**AST**)
- **ANTLR: AN**other **T**ool for **L**anguage **R**ecognition, is a Java-based tree parser generator that allows users to define grammar and **AST**-parsers for [specific languages](#).

STEP 3 (Query Generation)

- The lexer and parser generated from the grammar are used to translate the **JSON** array derived from the **NER** extractor into a corresponding **TypeQL** query.



RASA

Vaticle TypeDB

Vaticle TypeQL

webex by CISCO

Quorra the Bot

https://disney.fandom.com/wiki/Quorra

The Query Languages

- QLs have Complex syntax not easy to learn
- Require understanding of the knowledge base structure
- Writing query is error-prone (typos !)

Querying a Knowledge Graph is difficult …

- Telcos generate high volumes of service requests,
- BUT it is extremely difficult to find relevant content that can help Technical Assistance Center (TAC) engineers to solve customers' problems quickly.

Q1: Which is the application with more connections?

```
> match $x1 isa app; (src:$x1, dst:$anyone) isa
connection, has numberOfConnections $x3; max $x3;
```

3.4030913E7

```
>match $x1 isa app, has app_id $x2; (src:$x1,
dst:$anyone) isa connection, has numberOfConnections
34030913; (contained-app:$x1) isa environment, has
env_name $x4; get $x2, $x4;
```

{ $x2 "787a49fb-a5d4-552e-a7f3" isa app_id; $x4 "Prod" isa env_name; }

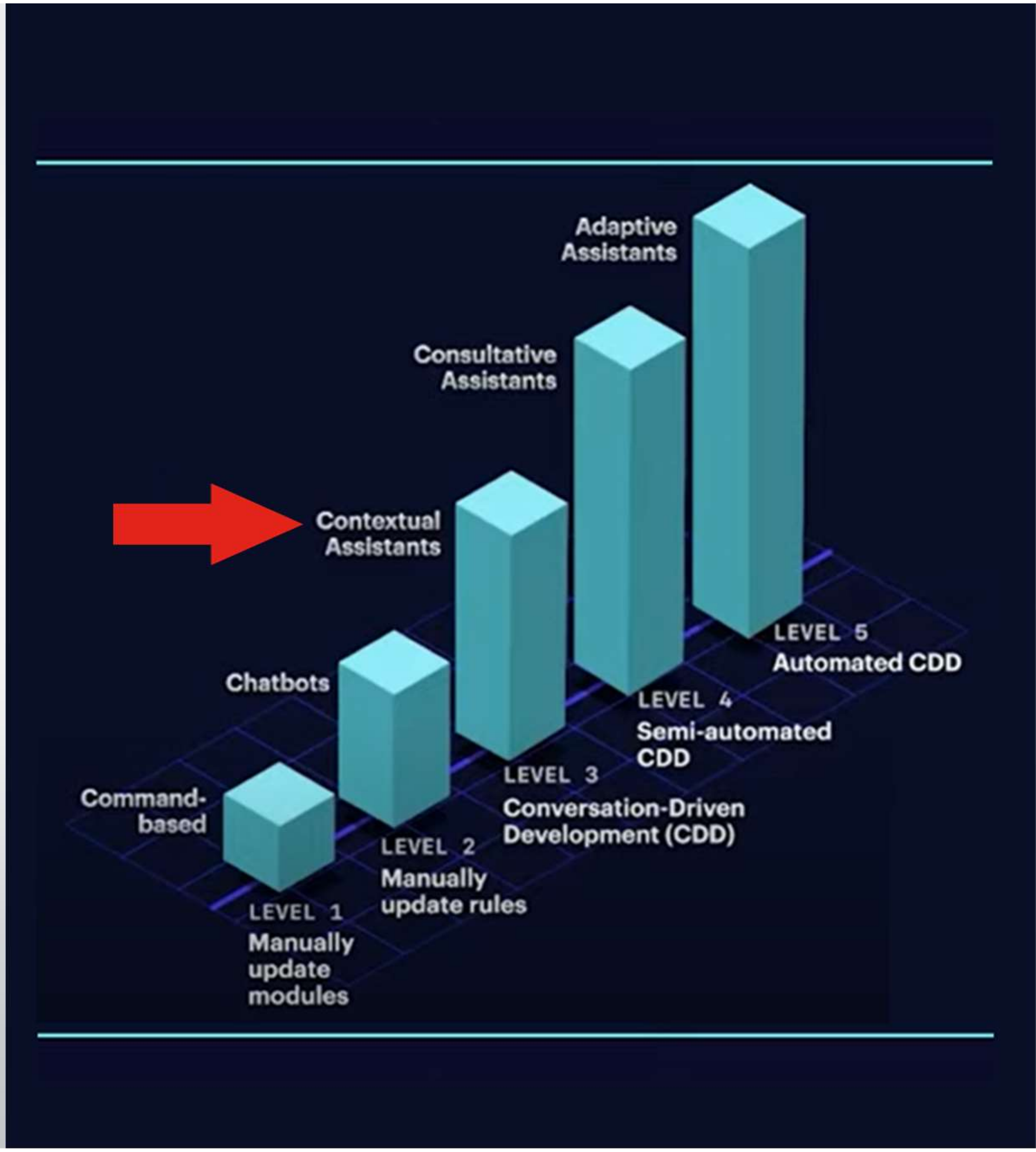
Why not use a **Conversational AI** assistant to access the **Knowledge Graph** using **Natural Language** queries ?

Motivations for a Natural Language Interface (NLI)

- It is impossible to anticipate all the things users could say…

  … BUT users tell in their own words exactly what they want.
- NLI can match natural language questions with formal queries for easy access to information stored in a knowledge base
- A KG organises and integrates data according to an ontology and applies a reasoner to derive new knowledge
- Knowledge reasoning over KGs aims to identify errors and infer new conclusions from existing data

Conversational AI Levels

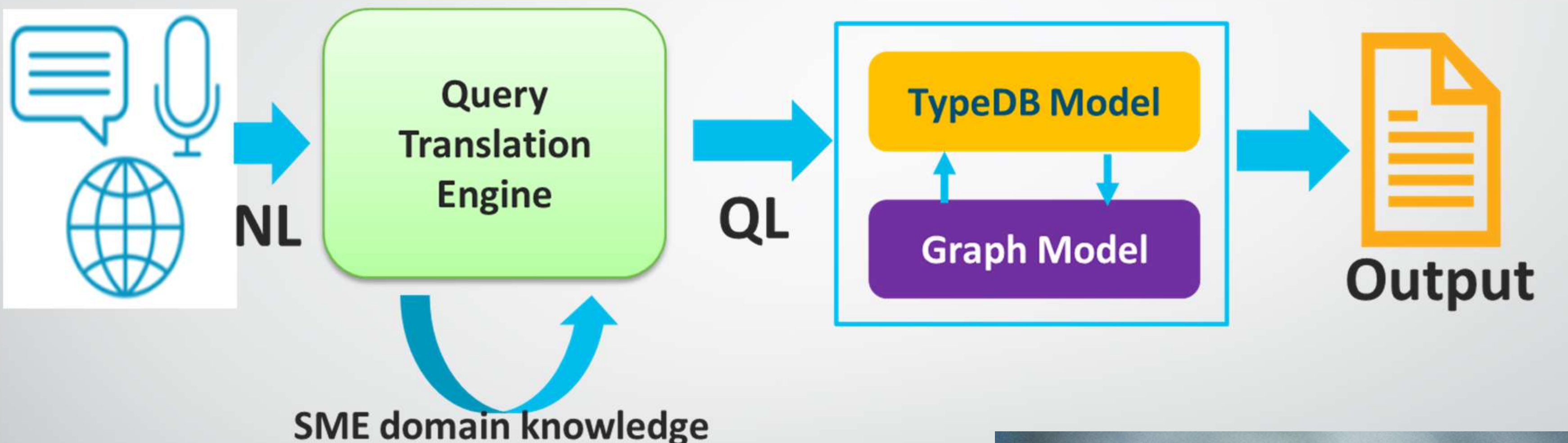


Advantages of NLI

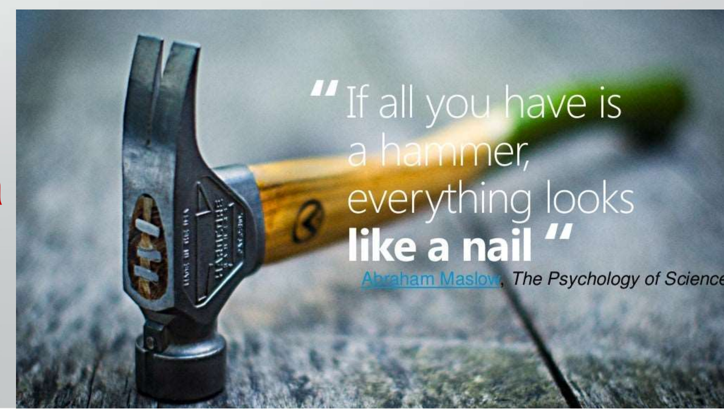
- NL queries make KG more accessible to the average user
- benefit for analysts with various backgrounds
- allow end-users to focus on other tasks and access KGs as they like

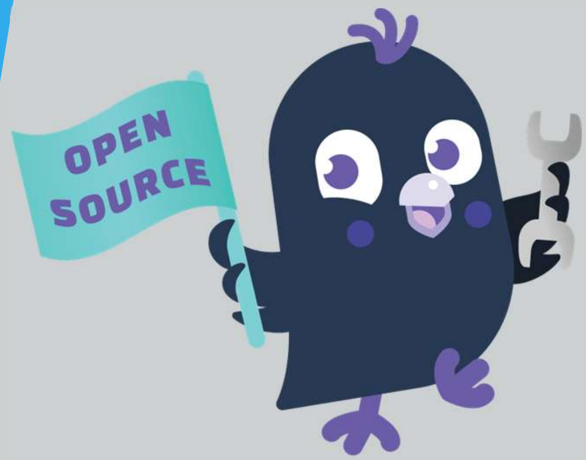
Use Case Example for Networking Applications

The Translation Pipeline from NL to QL



The translation from NL to QL is *not* a classification problem

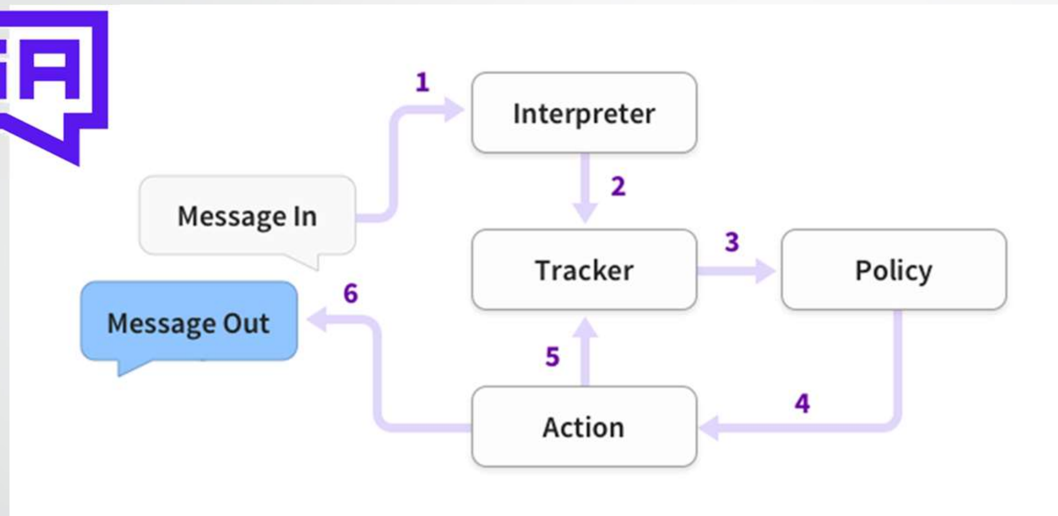




- **RASA** is a conversational AI framework for building *contextual assistants*.
- It provides E2E tools to build advanced NLI interacting with human users.
- We use RASA to train an NLU model for multi-intent classification and **Named Entity Recognition** (NER)
- **NER** is the task of identifying the entities that appear in a text in multiple contexts

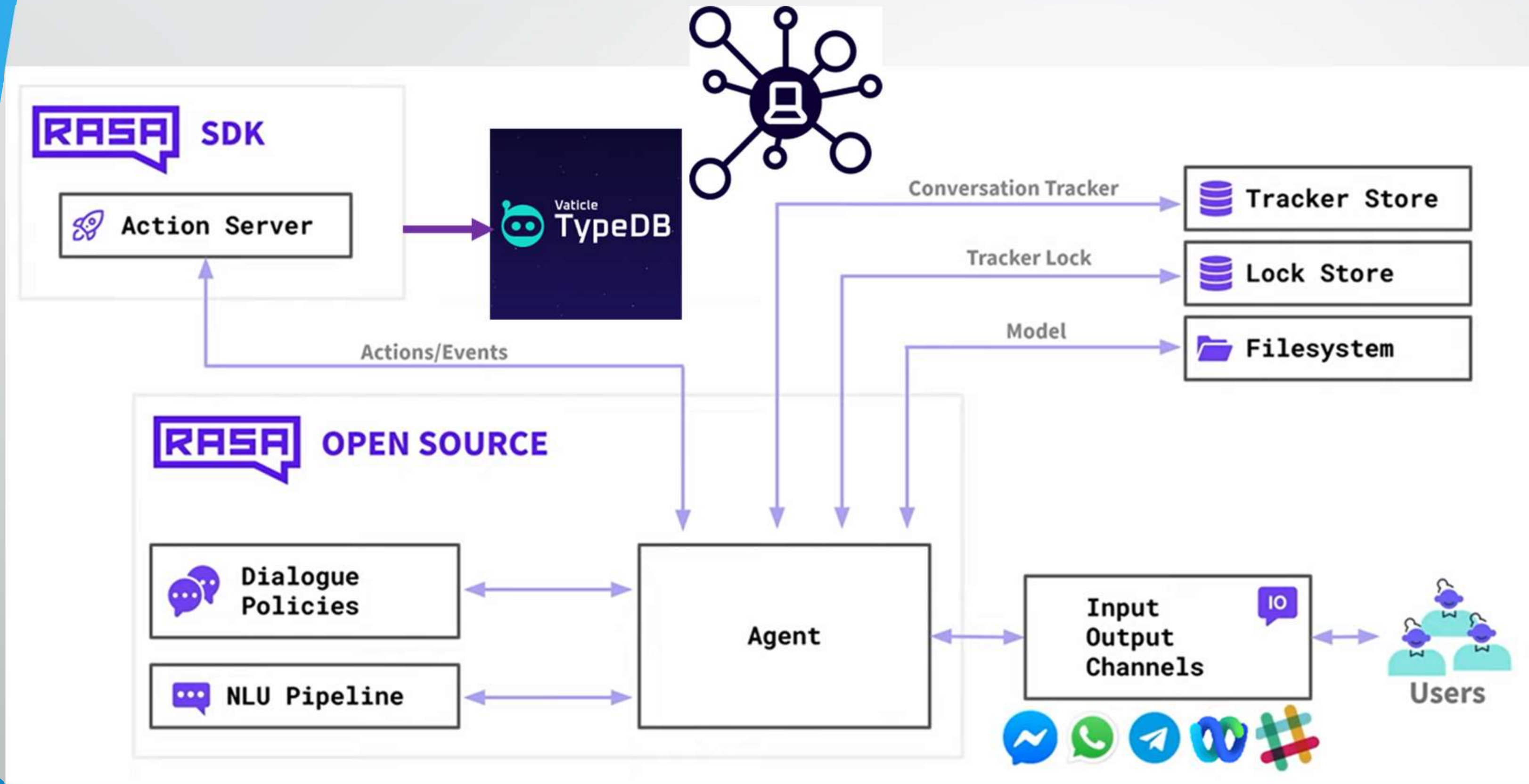
https://rasa.com

RASA Architecture




```yaml
- rule: query knowledge base
  steps:
    - intent: query_knowledge_base
    - action: action_check_typos
    - action: action_query_knowledge_base
```

RASA meets TypeDB



UC#1 : FAQs and predefined Q/As

- Show the attributes of the environment
- Display the knowledge base schema

UC#2 : Query an object and its attributes

- Show the environments with names
- Show applicatios with app risk and app ID

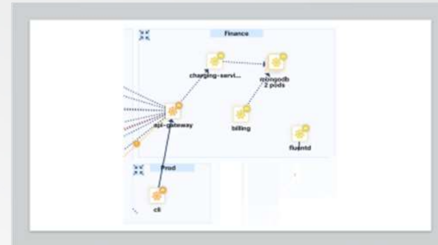
UC#3 : Query multiple objects and attributes

- Show applications in Finance with pod names
- Show links from Prod to Finance

UC#4 : Computation

- What application has more connections ?
- What application in Finance has higher risk ?

Annotated Example



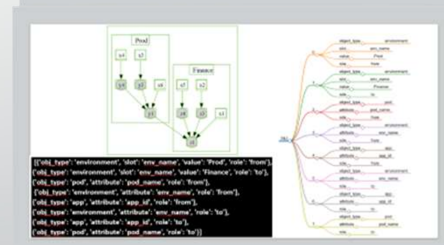
- ❑ Input Query (NL) :

show links from proad to fananze with pod names

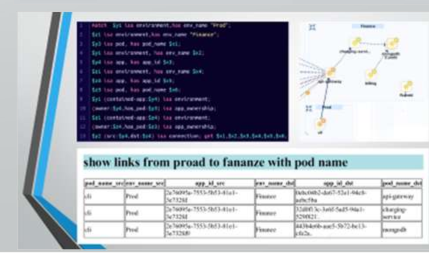
- ❑ NER extraction

- ❑ Typos correction

We use [Levenshtein Distance](#) to calculate the differences between sequences based on the [**fuzzywuzzy**](#) python package

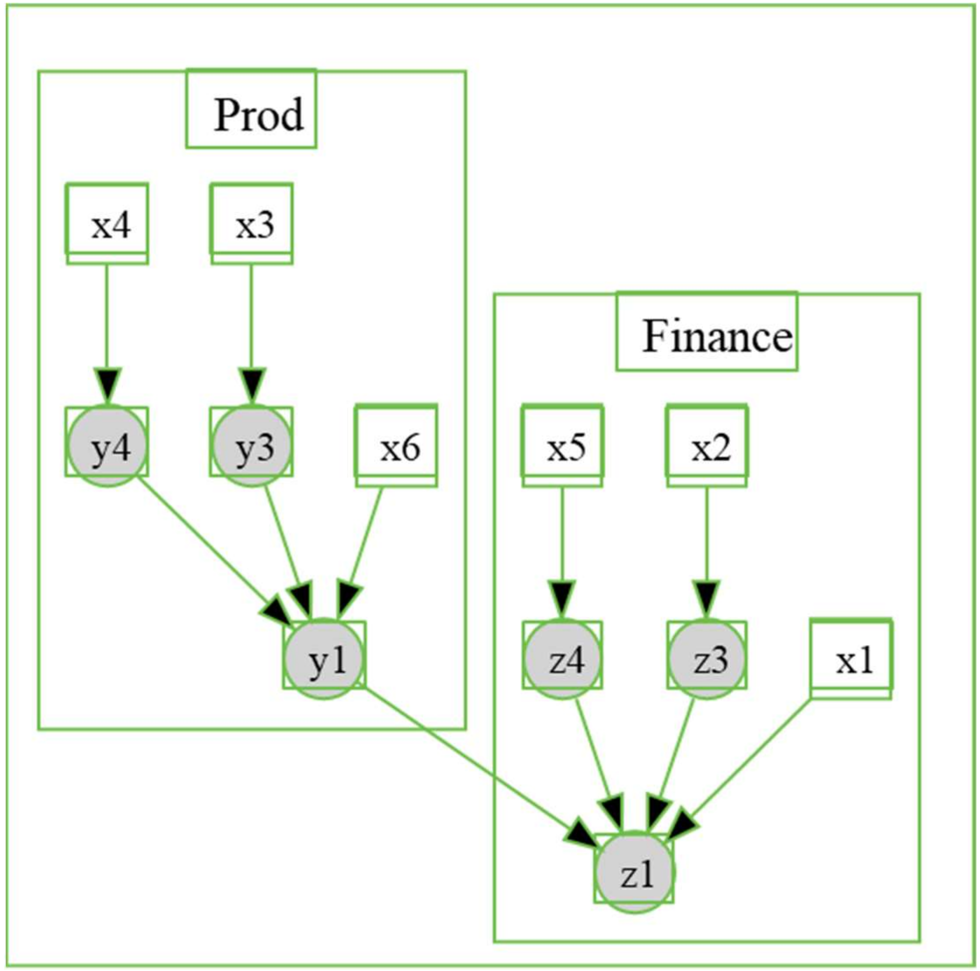


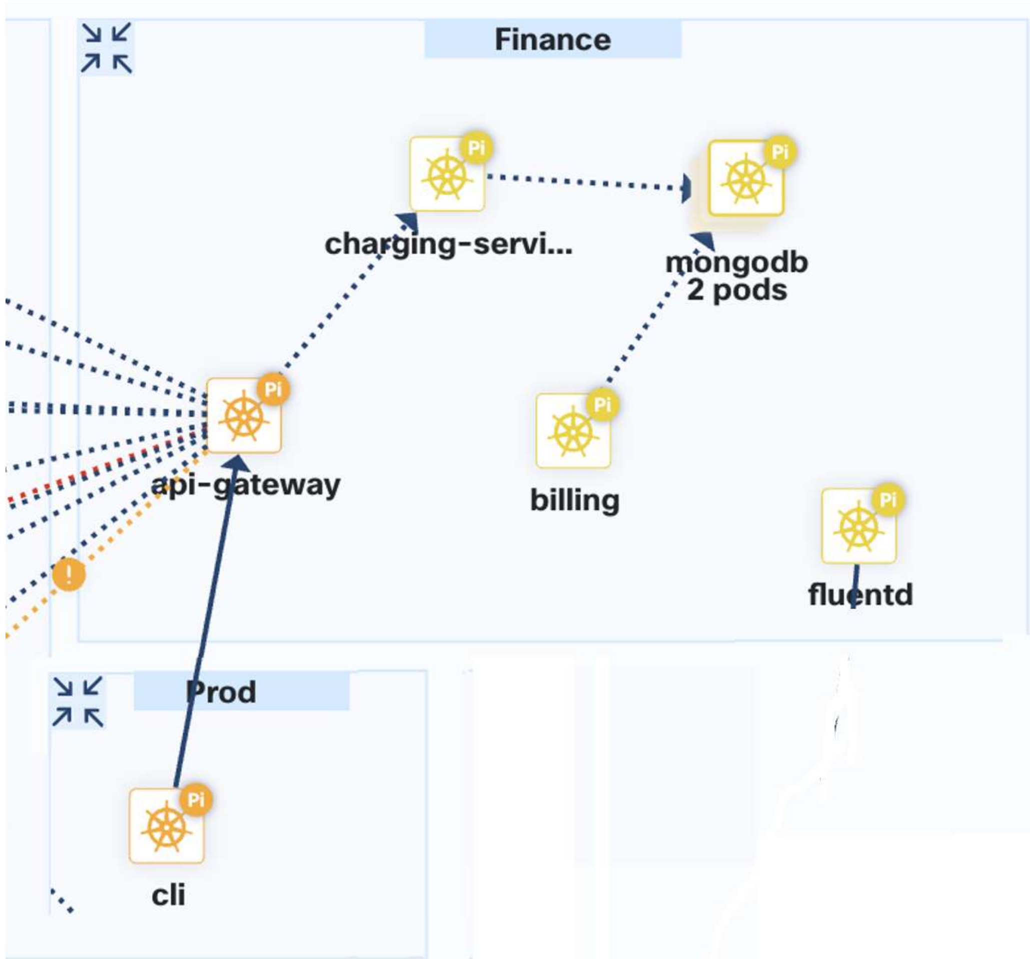
- ❑ NER integration with implicit references
- ❑ TypeQL query generation
  - ❑ Execute inferences



```
language: en

pipeline:
  - name: "WhitespaceTokenizer"            -> tokenizer using whitespaces
  - name: "RegexFeaturizer"                -> creates features for NER and intent classification
  - name: "LexicalSyntacticFeaturizer"     -> Creates lexical/syntactic features for NER
  - name: "CountVectorsFeaturizer"         -> counts whole words
  - name: "CountVectorsFeaturizer"         -> counts sub-sequence of n-gram
    analyzer: "char_wb"
    min_ngram: 1
    max_ngram: 5
  - name: "DIETClassifier"                 -> Dual Intent Entity Transformer
    entity_recognition: true
    epochs: 200
    constrain_similarities: True
  - name: "EntitySynonymMapper"            -> Maps synonymous entity values
```

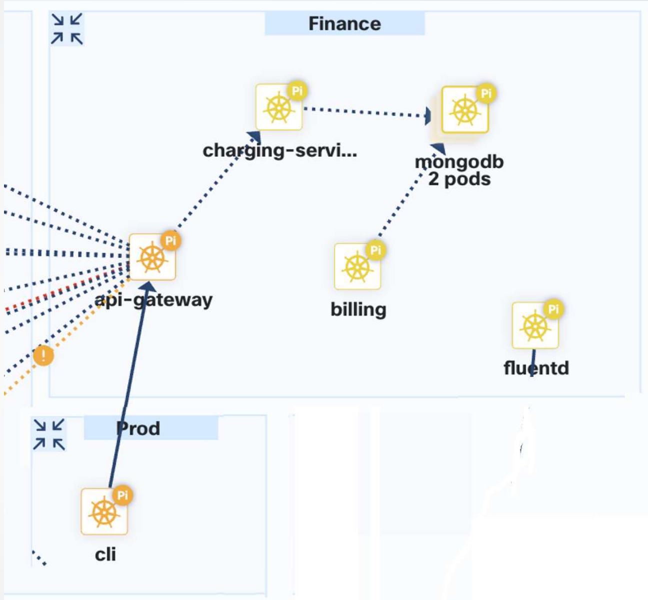

Prod
x4
x3
y4
y3
x6
y1
Finance
x5
x2
z4
z3
x1
z1



Finance
charging-servi...
mongodb
2 pods
api-gateway
billing
fluentd
Prod
cli

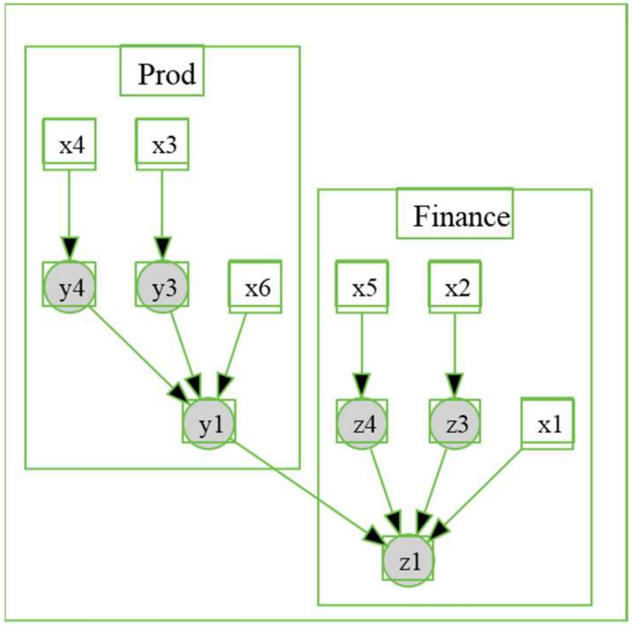
```
show links [from proad]{"entity": "env_name", "role": "from"} [to fananze]{"entity": "env_name", "role": "to"} with [pod
names]{"entity": "attribute", "value": "pod_name"}
intent: connection_data 1.00
```

```
match  $y1 isa environment,has env_name "Prod";
$z1 isa environment,has env_name "Finance";
$y3 isa pod, has pod_name $x1;
$y1 isa environment, has env_name $x2;
$y4 isa app, has app_id $x3;
$z1 isa environment, has env_name $x4;
$z4 isa app, has app_id $x5;
$z3 isa pod, has pod_name $x6;
$y1 (contained-app:$y4) isa environment;
(owner:$y4,has_pod:$y3) isa app_ownership;
$z1 (contained-app:$z4) isa environment;
(owner:$z4,has_pod:$z3) isa app_ownership;
$y2 (src:$y4,dst:$z4) isa connection; get $x1,$x2,$x3,$x4,$x5,$x6;
```

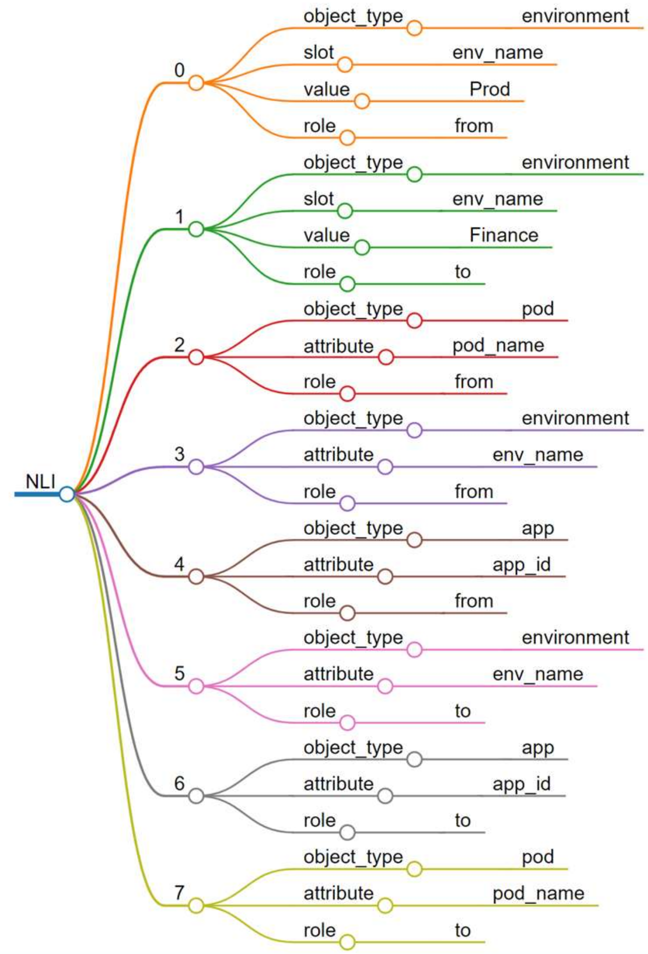


show links from proad to fananze with pod name

pod_name_src	env_name_src	app_id_src	env_name_dst	app_id_dst	pod_name_dst
cli	Prod	2e76095a-7553-5b53-81e1-3e732fd	Finance	0ebc04b2-de67-52e1-94c8-aebc5ba	api-gateway
cli	Prod	2e76095a-7553-5b53-81e1-3e732fd	Finance	32d0f13c-3e6f-5ad5-94a1-529f821	charging-service
cli	Prod	2e76095a-7553-5b53-81e1-3e732fd9	Finance	443b4e6b-aae5-5b72-bc13-cfe2a	mongodb



[{'obj_type': 'environment', 'slot': 'env_name', 'value': 'Prod', 'role': 'from'},
{'obj_type': 'environment', 'slot': 'env_name', 'value': 'Finance', 'role': 'to'},
{'obj_type': 'pod', 'attribute': 'pod_name', 'role': 'from'},
{'obj_type': 'environment', 'attribute': 'env_name', 'role': 'from'},
{'obj_type': 'app', 'attribute': 'app_id', 'role': 'from'},
{'obj_type': 'environment', 'attribute': 'env_name', 'role': 'to'},
{'obj_type': 'app', 'attribute': 'app_id', 'role': 'to'},
{'obj_type': 'pod', 'attribute': 'pod_name', 'role': 'to'}]



```
TypeDBOptions.core()
opts.infer = True
```

```
rule transitive-connections:
when {
        (src:$x, dst:$y) isa connection;
        (src:$y, dst:$z) isa connection;
      } then {
       (src:$x, dst:$z) isa connection;
      };
```

Takeaways

- allow non-technical users to query a KG with NL. All the advantages of KG reasoner w/o the complexities of a query language.
- Track information collected from users during the dialogue for dynamic dashboard presentations
- Useful for intelligent troubleshooting systems, e.g. RCA for TAC engineers
- **TypeDB** and **RASA** allow easy integration with WebexTeams REST APIs and other clients (Facebook, Telegram, Twilio, etc.)
- continuous learning strategy integrated with **GitHub**


Q&A