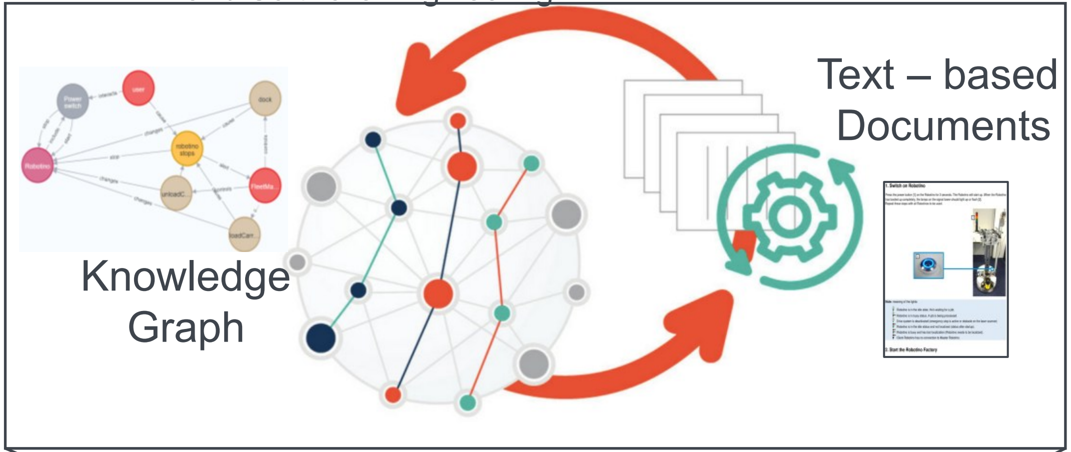
University of Stuttgart
Institute of Industrial Automation
and Software Engineering


Text – based Documents

Knowledge Graph

# Generation of knowledge graph from textual data for describing causal system behaviors on the example of an automated production facility

Name : Soheb Hanif Teli

Study Program: INFOTECH

# Agenda

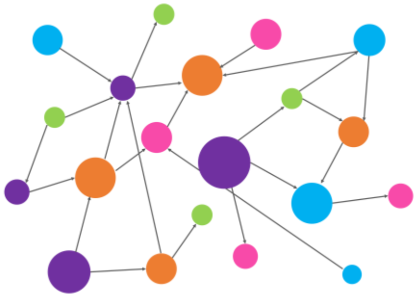
Motivation
Literature research
Methods
Results
Evaluation
System Overview
Summary and Outlook

# Motivation

## Automated generation of Knowledge Graph

Natural Texts (Manuals, articles, Etc.)

Natural Language Processing



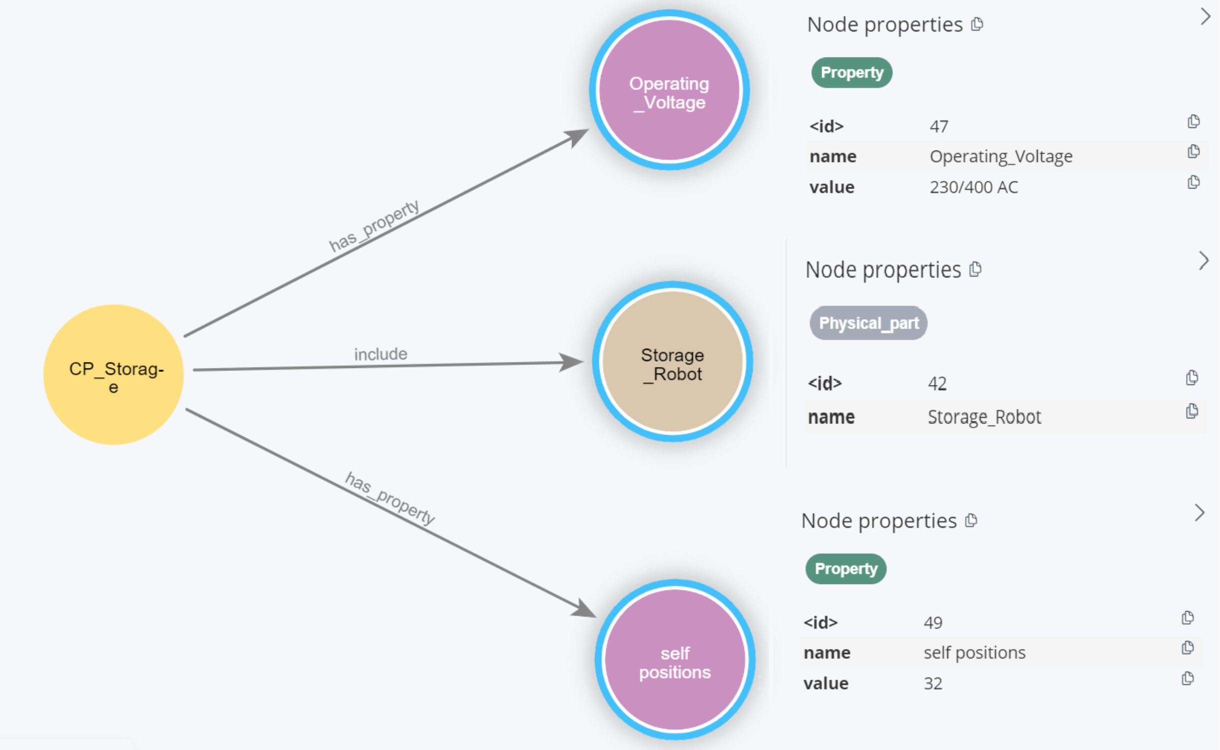
- Need for knowledge graph
- Meanwhile, minimize human effort

# Knowledge Graph

## Property Graph

For instance:

“CP-Storage has a storage robot that can be used to load or unload palettes from 32 selves and an operating voltage of 230/400 AC”



Node properties

Property

| | |
|---|---|
| <id> | 47 |
| name | Operating_Voltage |
| value | 230/400 AC |

Node properties

Physical_part

| | |
|---|---|
| <id> | 42 |
| name | Storage_Robot |

Node properties

Property

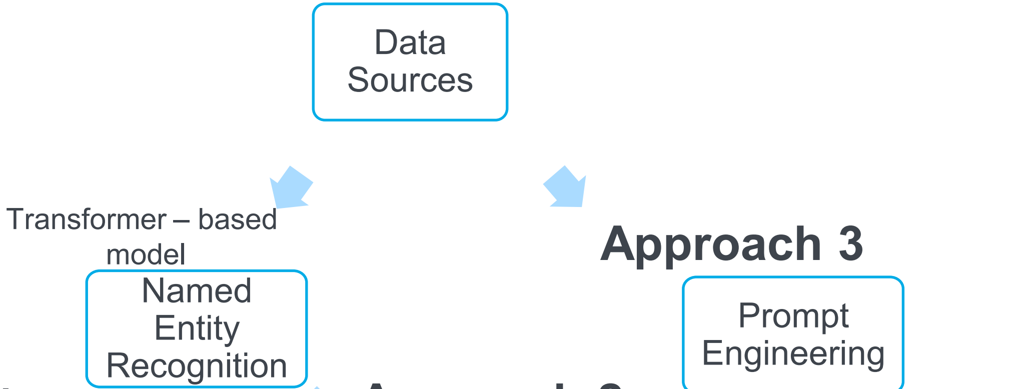
| | |
|---|---|
| <id> | 49 |
| name | self positions |
| value | 32 |

# Literature research

# Text to KG
# How could the KG be generated?

## General Overview

Fleet Manager EN 2016_12_06
QuickStart A002_FactoryApp
Robotino Factory EN 2016_12_06
Software_Doku_CP-Factory_V1.5_EN

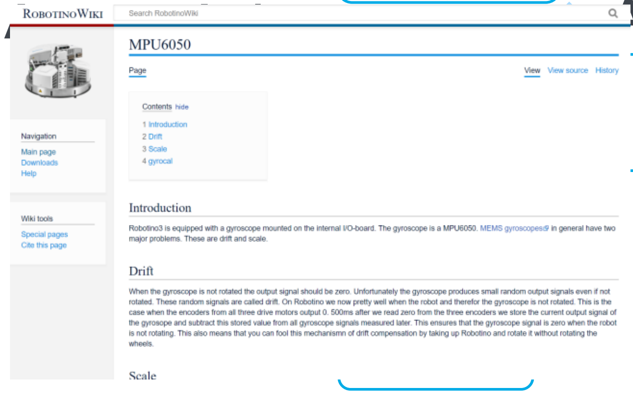


Approach 2

Relation Extraction

GPT - Model

Large Language Model



MPU6050

Contents hide

1 Introduction
2 Drift
3 Scale
4 gyrocal

Introduction

Robotino3 is equipped with a gyroscope mounted on the internal I/O-board. The gyroscope is a MPU6050. MEMS gyroscopes in general have two major problems. These are drift and scale.

Drift

When the gyroscope is not rotated the output signal should be zero. Unfortunately the gyroscope produces small random output signals even if not rotated. These random signals are called drift. On Robotino we now pretty well when the robot and therefor the gyroscope is not rotated. This is the case when the encoders from all three drive motors output 0. 500ms after we read zero from the three encoders we store the current output signal of the gyrosope and subtract this stored value from all gyroscope signals measured later. This ensures that the gyroscope signal is zero when the robot is not rotating. This also means that you can fool this mechanismn of drift compensation by taking up Robotino and rotate it without rotating the wheels.

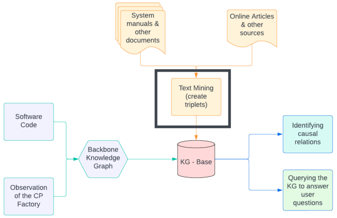
Scale

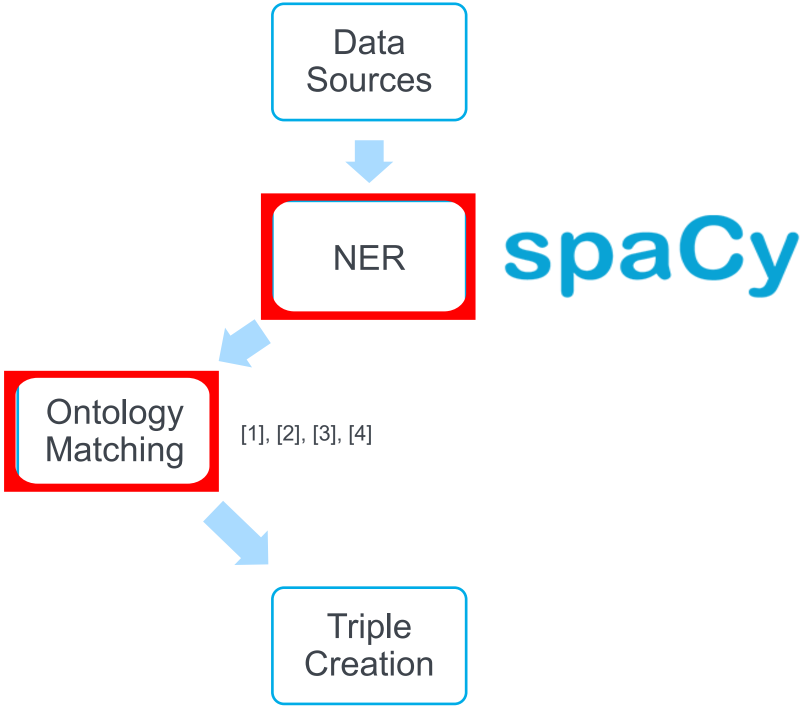
| Approach | Literature |
|---|---|
| Ontology Matching<br>Relation Extracting<br>Prompt Engineering | [1],[5],[2],[8],[6],[3],[9],[7],[4] |

# 3 Methods to generate KG in detail

# Approach (1/3)
# Text to KG

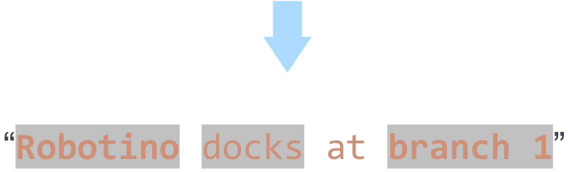
## Ontology – Based Approach

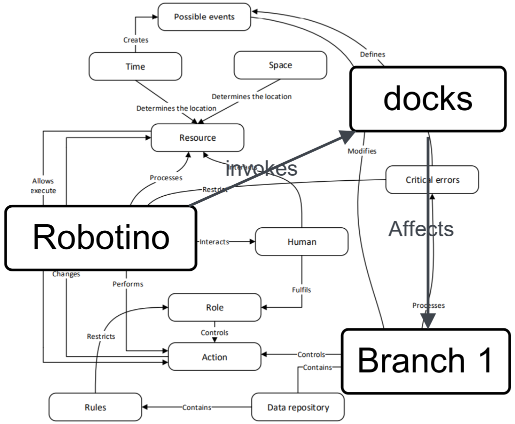

System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Software Code
Observation of the CP Factory
Backbone Knowledge Graph
KG - Base
Identifying causal relations
Querying the KG to answer user questions



[1], [2], [3], [4]

Consider the example:
“Robotino docks at branch 1”

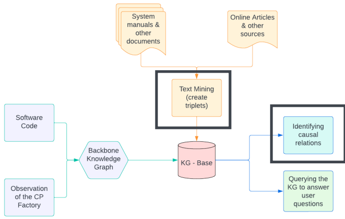




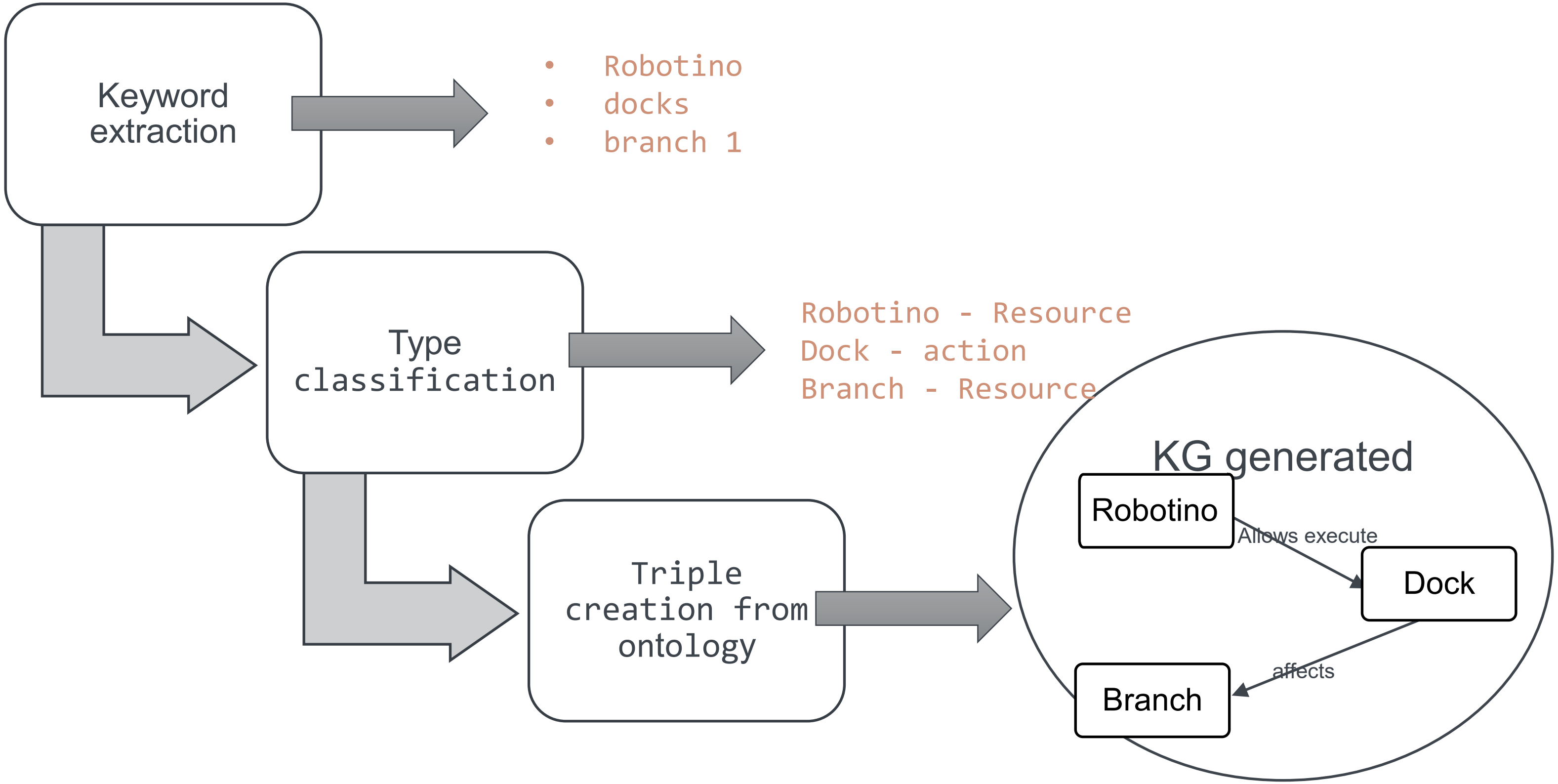
# Approach (1/3)

## Text to KG

### Ontology Matching


System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Software Code
Observation of the CP Factory
Backbone Knowledge Graph
KG - Base
Identifying causal relations
Querying the KG to answer user questions

Consider the example:
“Robotino docks at branch 1”



Keyword extraction

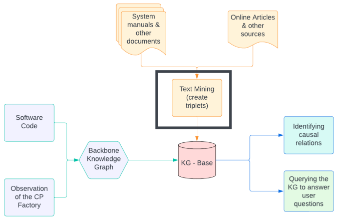
- Robotino
- docks
- branch 1

Type classification

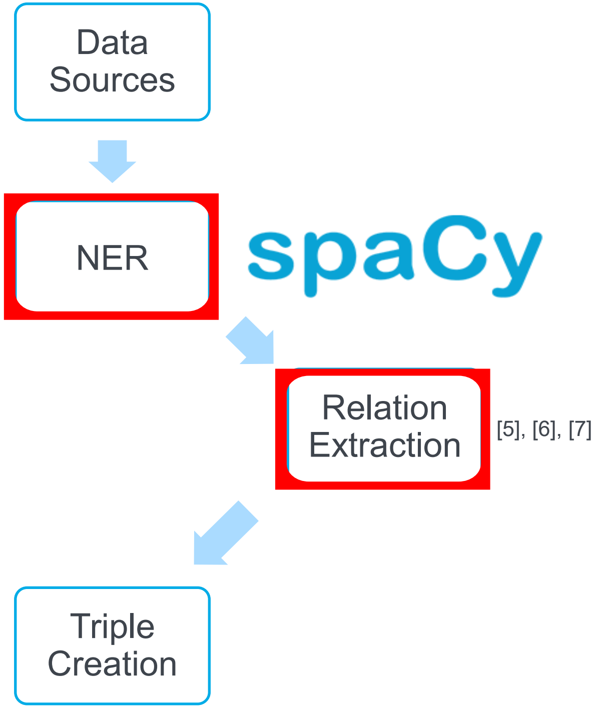
Robotino - Resource
Dock - action
Branch - Resource

Triple creation from ontology

# Approach (2/3)
# Text to KG


System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Software Code
Observation of the CP Factory
Backbone Knowledge Graph
KG - Base
Identifying causal relations
Querying the KG to answer user questions

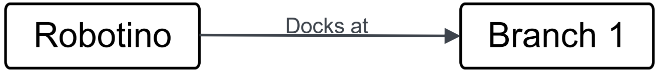
## Linguistic – Based Approach



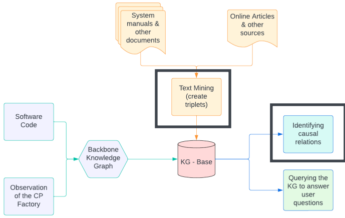
Consider the example:

“Robotino docks at branch 1”

| “Robotino | docks | at | branch 1” |
|---|---|---|---|
| Noun | Verb | | Noun |



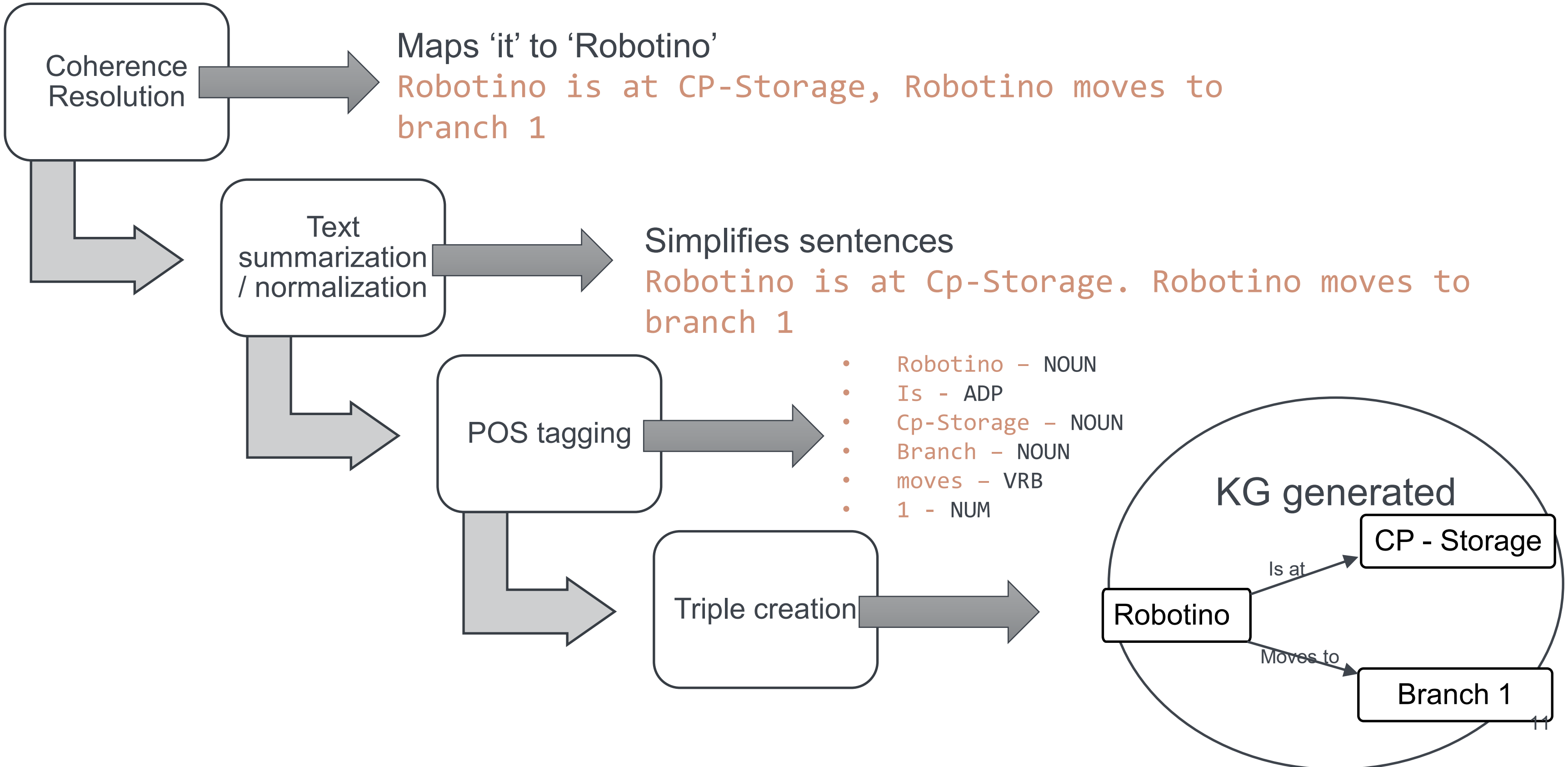
# Approach (2/3)
# Text to KG


System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Software Code
Backbone Knowledge Graph
KG - Base
Identifying causal relations
Observation of the CP Factory
Querying the KG to answer user questions

## Relation Extraction

- Consider the example:

“Robotino is at CP-Storage, it moves to branch 1”



Maps ‘it’ to ‘Robotino’

Robotino is at CP-Storage, Robotino moves to branch 1

Simplifies sentences

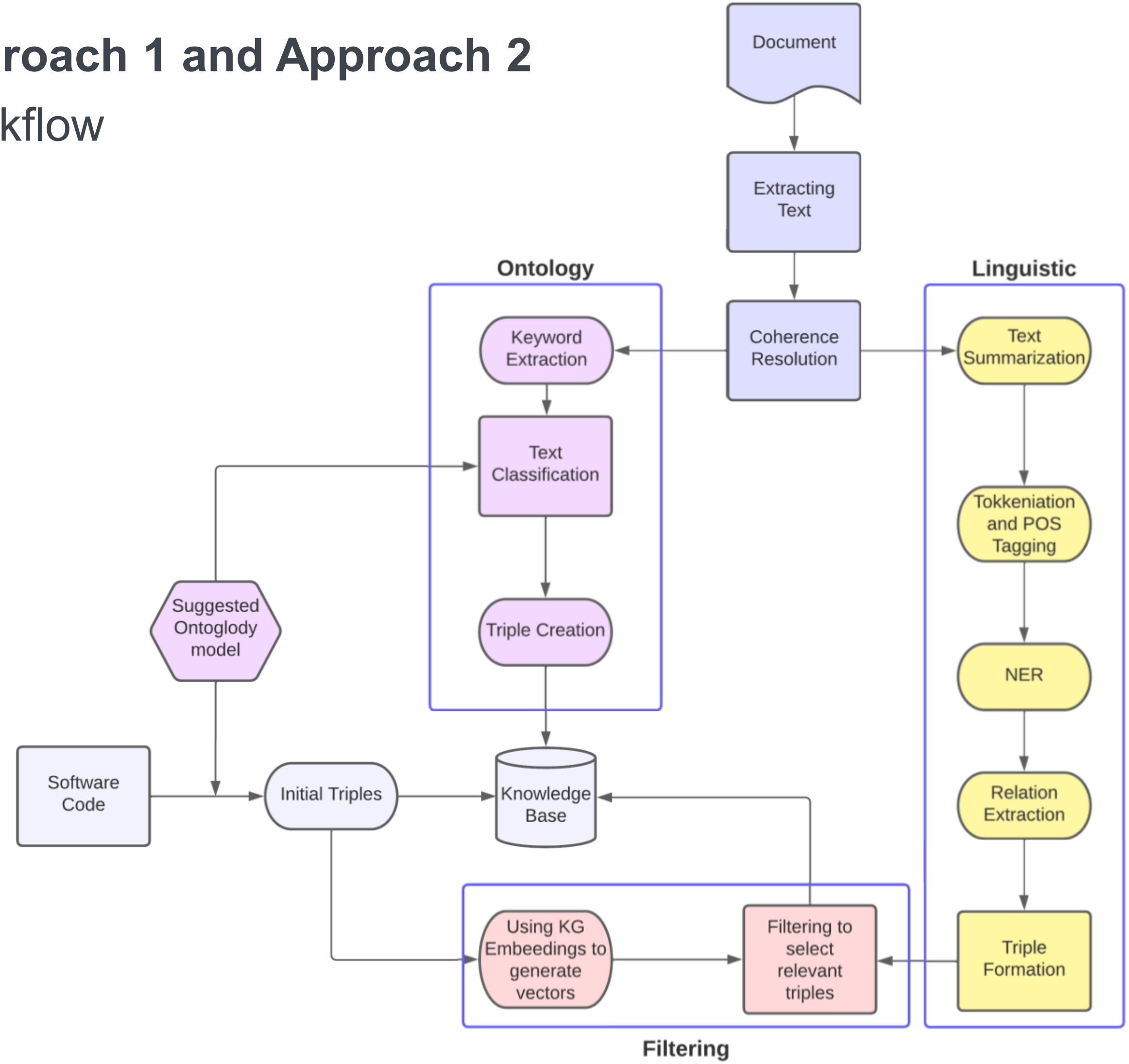
Robotino is at Cp-Storage. Robotino moves to branch 1

- Robotino – NOUN
- Is - ADP
- Cp-Storage – NOUN
- Branch – NOUN
- moves – VRB
- 1 - NUM

# Combine Approach 1 and Approach 2

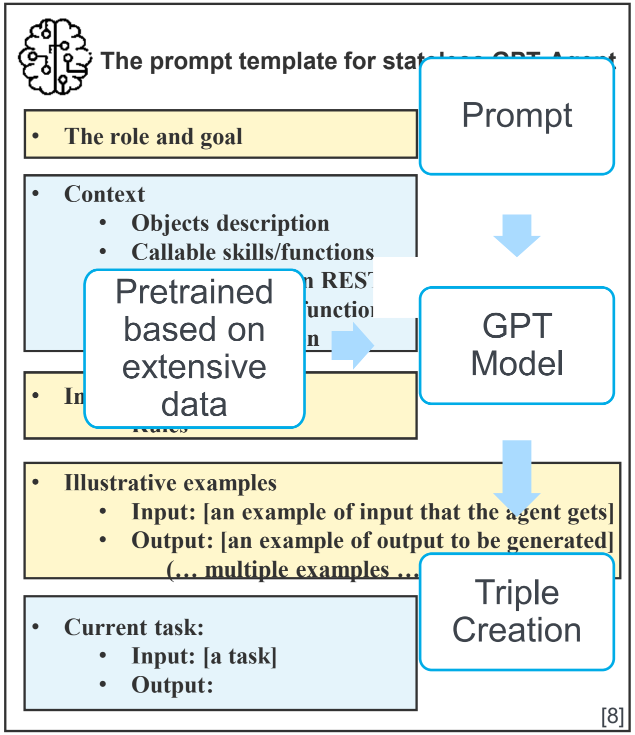
## Proposed Workflow

Details in Backup-Slides



# Approach (3/3)
# Text to KG

## Prompting using GPT model



The prompt template for stat

- The role and goal
- Context
  - Objects description
  - Callable skills/functions
- In
- Illustrative examples
  - Input: [an example of input that the agent gets]
  - Output: [an example of output to be generated]

  (… multiple examples …
- Current task:
  - Input: [a task]
  - Output:

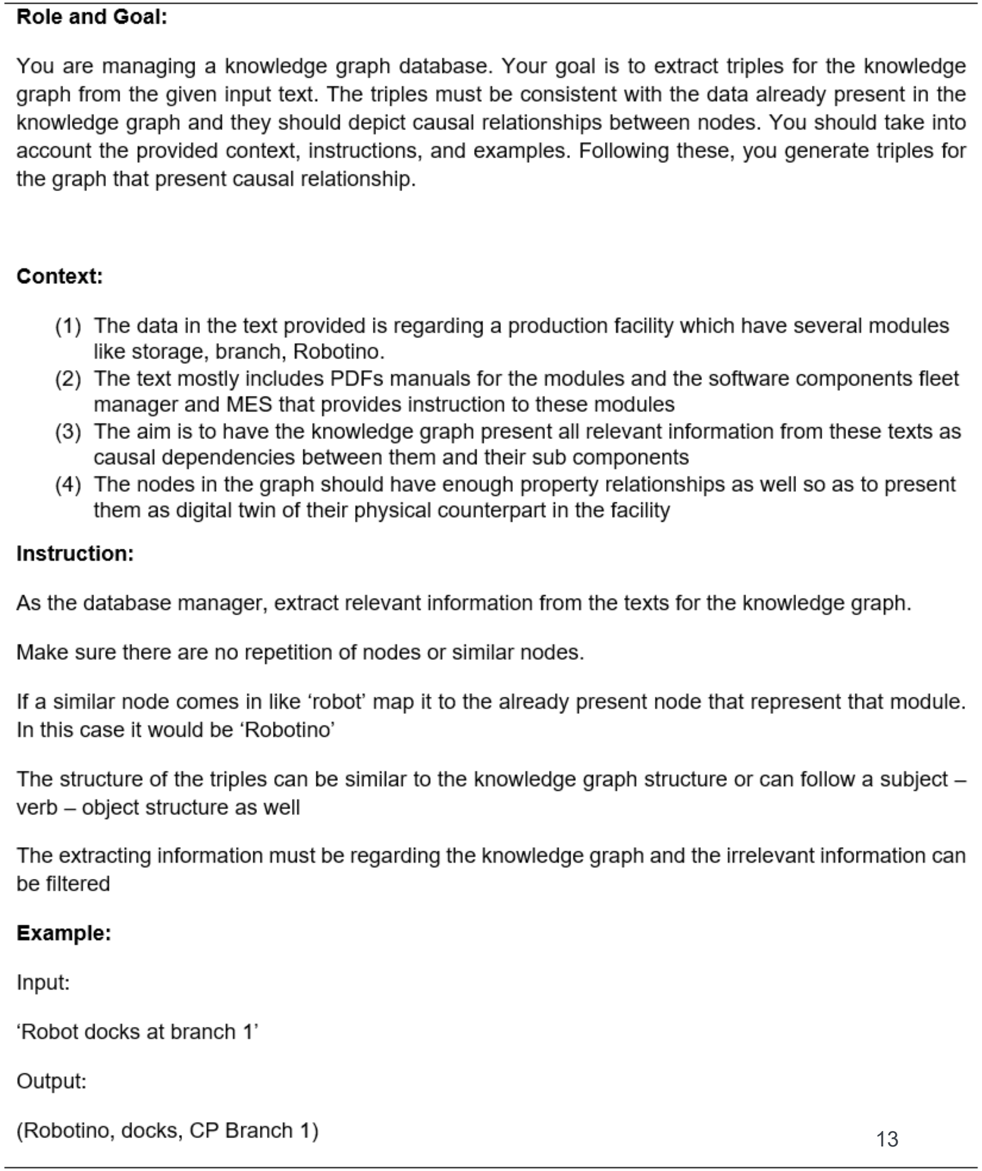
Prompt

Pretrained based on extensive data

GPT Model

Triple Creation

[8]



**Role and Goal:**

You are managing a knowledge graph database. Your goal is to extract triples for the knowledge graph from the given input text. The triples must be consistent with the data already present in the knowledge graph and they should depict causal relationships between nodes. You should take into account the provided context, instructions, and examples. Following these, you generate triples for the graph that present causal relationship.

**Context:**

(1) The data in the text provided is regarding a production facility which have several modules like storage, branch, Robotino.
(2) The text mostly includes PDFs manuals for the modules and the software components fleet manager and MES that provides instruction to these modules
(3) The aim is to have the knowledge graph present all relevant information from these texts as causal dependencies between them and their sub components
(4) The nodes in the graph should have enough property relationships as well so as to present them as digital twin of their physical counterpart in the facility

**Instruction:**

As the database manager, extract relevant information from the texts for the knowledge graph.

Make sure there are no repetition of nodes or similar nodes.

If a similar node comes in like ‘robot’ map it to the already present node that represent that module. In this case it would be ‘Robotino’

The structure of the triples can be similar to the knowledge graph structure or can follow a subject – verb – object structure as well

The extracting information must be regarding the knowledge graph and the irrelevant information can be filtered

**Example:**

Input:

‘Robot docks at branch 1’

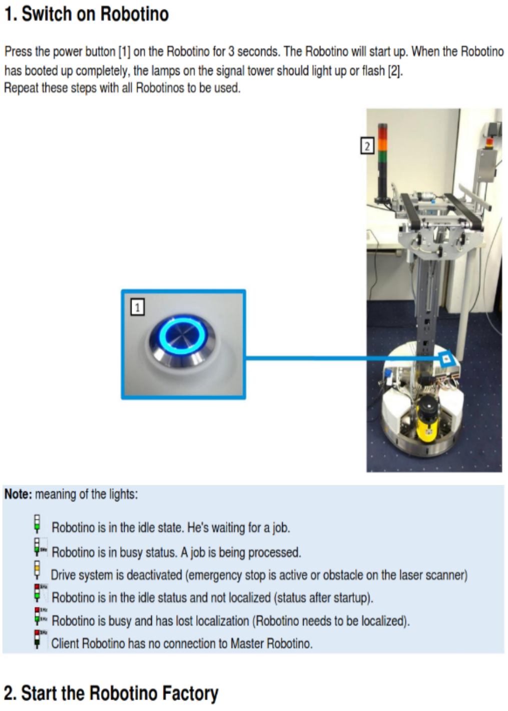
Output:

(Robotino, docks, CP Branch 1)

# Results

# Triple Creation

## Comparison of results



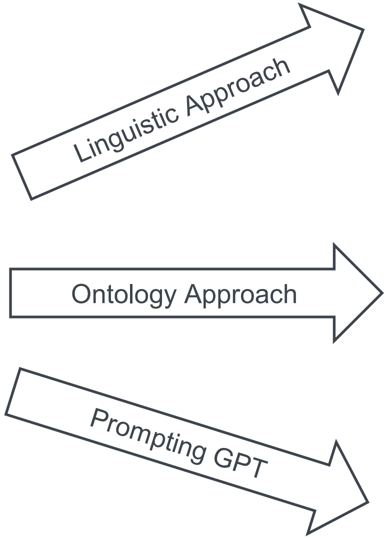
### 1. Switch on Robotino

Press the power button [1] on the Robotino for 3 seconds. The Robotino will start up. When the Robotino has booted up completely, the lamps on the signal tower should light up or flash [2].
Repeat these steps with all Robotinos to be used.

**Note:** meaning of the lights:

- Robotino is in the idle state. He's waiting for a job.
- Robotino is in busy status. A job is being processed.
- Drive system is deactivated (emergency stop is active or obstacle on the laser scanner)
- Robotino is in the idle status and not localized (status after startup).
- Robotino is busy and has lost localization (Robotino needs to be localized).
- Client Robotino has no connection to Master Robotino.

### 2. Start the Robotino Factory



Linguistic Approach

| Subject | Verb | Object |
|---|---|---|
| Robotino | Know | Initial space |
| Robotino | Has | Idle master state |
| Robotino | Is in | Idle startup |
| Step | Translated by | Laser scanner |
| Mouse click | Helps | Mapping |
| light | Indicates | status |
| Task | Assigned to | Robotino |
| User | Pressed | Manual button |

Ontology Approach

| Entity | Relation | Entity |
|---|---|---|
| position | Changes | Robotino |
| Fleet manager | Controls | Move |
| Move | Changes | Robotino |
| User | Interacts | button |

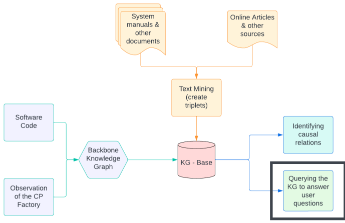
Prompting GPT

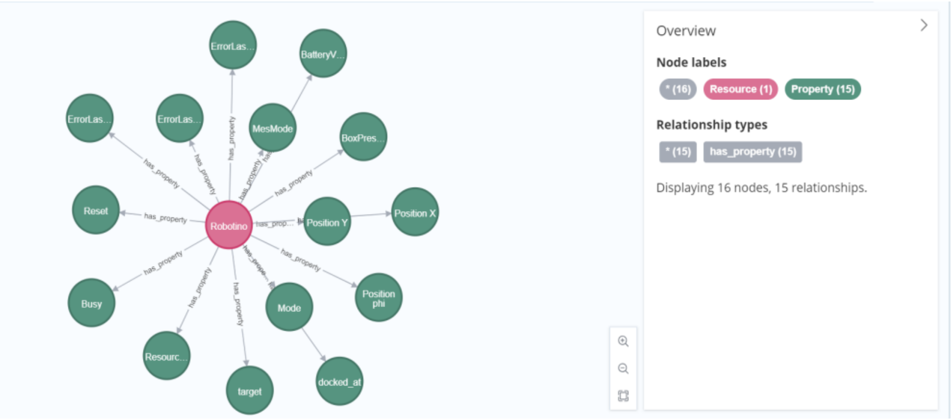
**Output:**

1. Triple: (Robotino, has, power button)
2. Triple: (Robotino, start up, lamps on signal tower)
3. Triple: (lights, indicate, status of Robotino)
4. Triple: (Robotino, use, Robotino Factory)
5. Triple: (Robotino Factory, assign, home position)
6. Triple: (multiple Robotinos, active at same time, lead to confusion)
7. Triple: (click, connection symbol, establish connection to Master Robotino)
8. Triple: (IP address, entered, pop-up screen)
9. Triple: (Master Robotino, marked, 1)
10. Triple: (Robotino, selected, left mouse click)
11. Triple: (activation of "Show laser scanner" function, helps, align Robotino more precisely)

# Use case 1: Formal Data Storage

Properties and causal relationships of a component (E.g.: Robotino)


System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Software Code
Observation of the CP Factory
Backbone Knowledge Graph
KG - Base
Identifying causal relations
Querying the KG to answer user questions



Overview

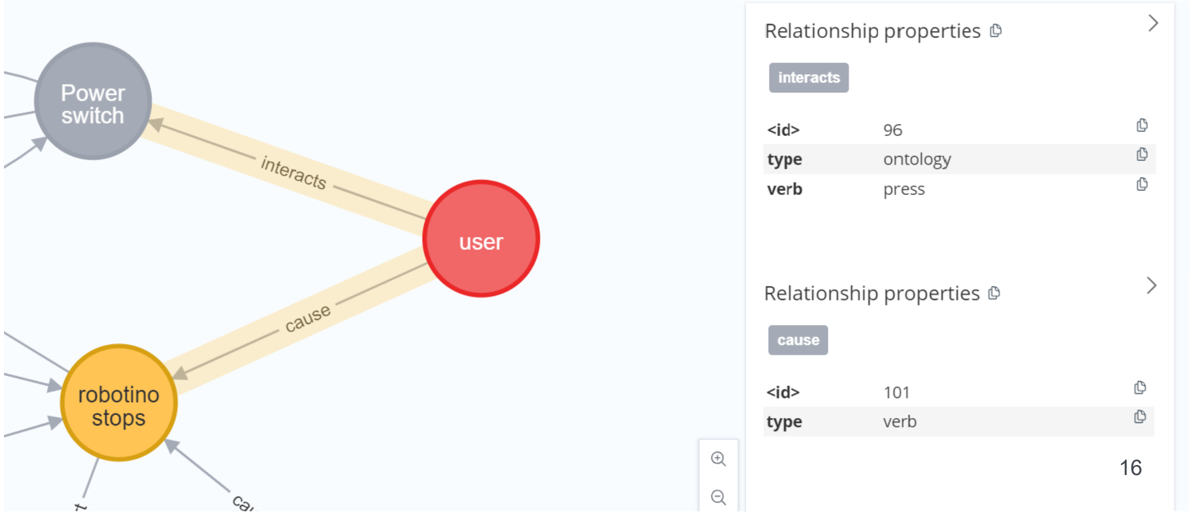
**Node labels**

* (16) Resource (1) Property (15)

**Relationship types**

* (15) has_property (15)

Displaying 16 nodes, 15 relationships.



Relationship properties

**interacts**

| | |
|---|---|
| **<id>** | 96 |
| **type** | ontology |
| **verb** | press |

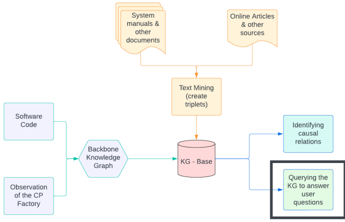
Relationship properties

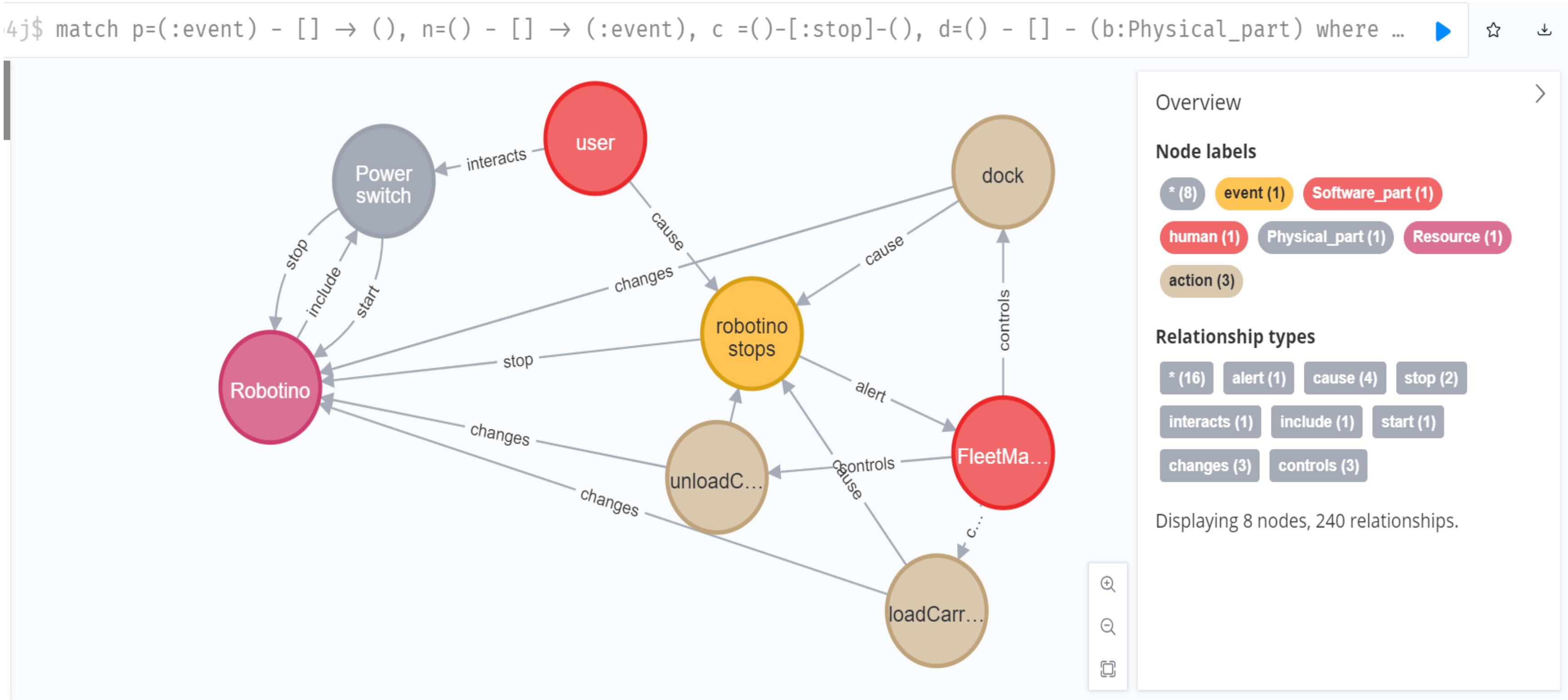
**cause**

| | |
|---|---|
| **<id>** | 101 |
| **type** | verb |

# Use case 2: Reasoning

## Querying – Robotino stops use case


System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Software Code
Observation of the CP Factory
Backbone Knowledge Graph
KG - Base
Identifying causal relations
Querying the KG to answer user questions



```
4j$ match p=(:event) - [] → (), n=() - [] → (:event), c =()-[:stop]-(), d=() - [] - (b:Physical_part) where …
```

Overview

**Node labels**

* (8) event (1) Software_part (1) human (1) Physical_part (1) Resource (1) action (3)

**Relationship types**

* (16) alert (1) cause (4) stop (2) interacts (1) include (1) start (1) changes (3) controls (3)

Displaying 8 nodes, 240 relationships.

# Evaluation

# Evaluation

Comparison of all 3 approaches (Ground truth: 44 triples)

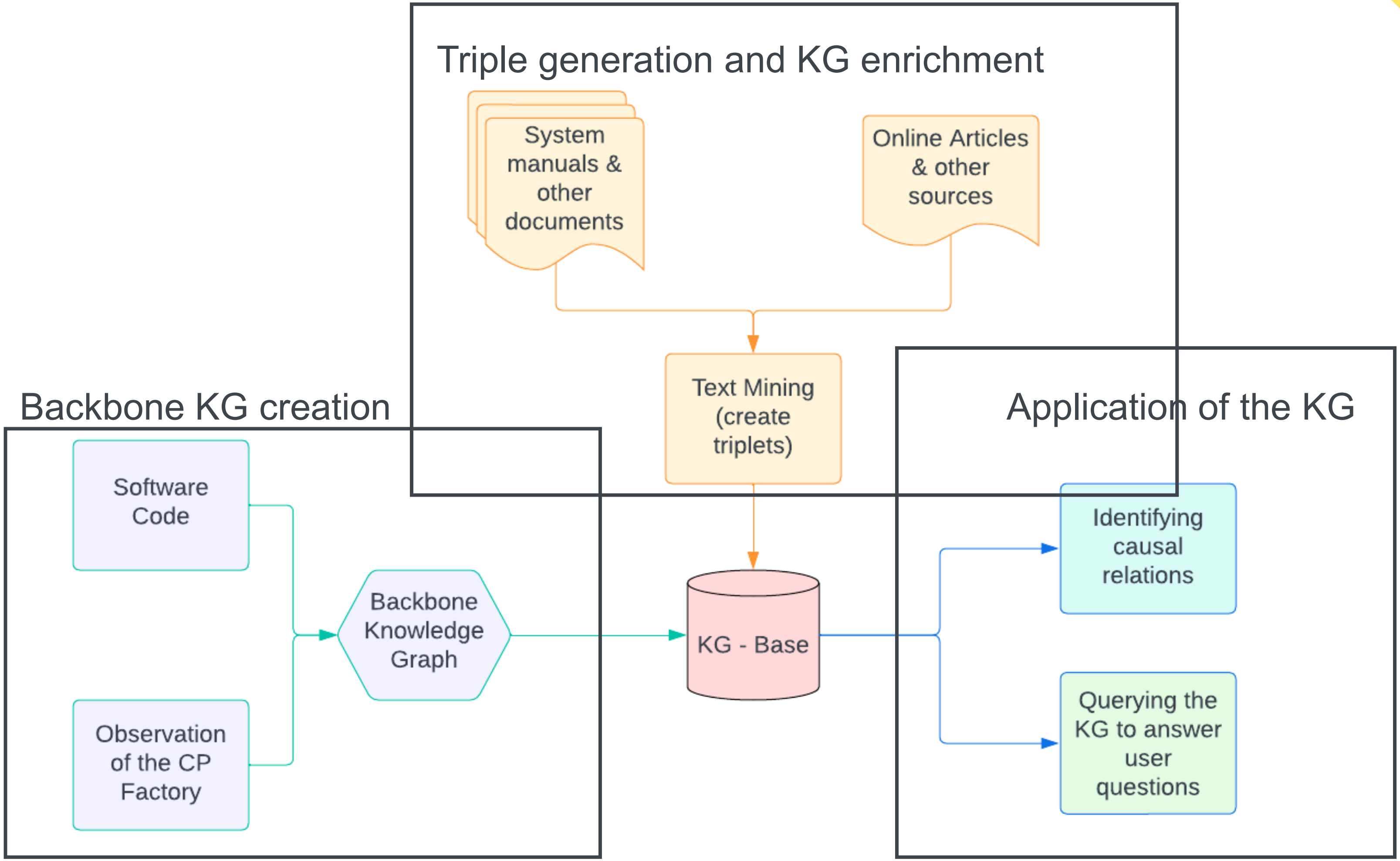
| Parameters | Ontology Matching | Relation Extraction | GPT - Prompting |
|---|---|---|---|
| Triples Extracted | 28 | 40 | 47 |
| correct triples (correctness ratio) | 20/28 | 40/40 | 43/47 |
| False Positive (wrongly extracted) | 8 | 2 | 6 |
| False Negative (failed to extract) | 24 | 6 | 3 |
| Precision | 20/28 = 0.7142 | 38/40 = 0.95 | 41/47 = 0.8723 |
| Recall | 20/44 =  0.4545 | 38/44 = 0.8636 | 41/44 = 0.9318 |
| F1 - score | 0.5521 | 0.9047 | 0.9010 |

- Ontology –based approach is limited by the entity classification
- Linguistic approach tries to extract all possible relations but works under the assumption of one triple per sentence
- GPT creates extra triples by generating text

# System Overview

# Proposed System Diagram

Details in Backup-Slides



Triple generation and KG enrichment
System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Backbone KG creation
Software Code
Observation of the CP Factory
Backbone Knowledge Graph
KG - Base
Application of the KG
Identifying causal relations
Querying the KG to answer user questions

# Summary and Outlook

- Summary
  - linguistic approach → all possible triples, but not all relevant
  - ontology based approach → more relevant results, but limited to the template KG
  - GPT Prompt-Engineering → relevant and rich results, but too creative

- A comprehensive system is developed to automatically generate KG

- Outlook
  - Add live sensor data into KG

University of Stuttgart
Institut of Industrial Automation
and Software Engineering

# Thank you!

**Soheb Hanif Teli**

| | |
|---|---|
| e-mail | st175461@stud.uni-stuttgart.de |
| phone | +49 (0) 711 685- |
| fax | +49 (0) 711 685- |

University of Stuttgart
Pfaffenwaldring 47, 70550

# Literature References

[1] Agrawal, G.; Deng, Y.; Park, J.; Liu, H.; Chen, Y.-C. Building Knowledge Graphs from Unstructured Texts: Applications and Impact Analyses in Cybersecurity Education. Information 2022, 13, 526. https://doi.org/10.3390/info13110526

[2] Voinov, Artem, and Ilya Senokosov. "Ontological models of cyber physical systems." Journal of Physics: Conference Series. Vol. 1889. No. 2. IOP Publishing, 2021.

[3] Yan Jia, Yulu Qi, Huaijun Shang, Rong Jiang, Aiping Li, “A Practical Approach to Constructing a Knowledge Graph for Cybersecurity” Engineering, Volume 4, Issue 1, 2018, Pages 53-60, ISSN 2095-8099, https://doi.org/10.1016/j.eng.2018.01.004.

[4] Abu-Salih, B. Domain-specific knowledge graphs: A survey. J. Netw. Comput. Appl. 2021, 185, 103076. [Google Scholar] [CrossRef]

[5] Kumar, A. and Dinakaran, S., “Textbook to triples: Creating knowledge graph in the form of triples from AI TextBook”, arXiv e-prints, 2021. doi:10.48550/arXiv.2111.10692.

[6] Honnibal, M. & Montani, I., 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

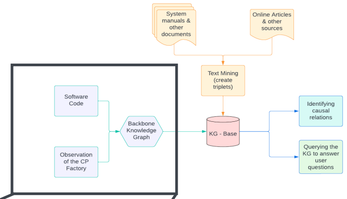
[7] Asghar, Nabiha. "Automatic extraction of causal relations from natural language texts: a comprehensive survey." arXiv preprint arXiv:1605.07895 (2016).

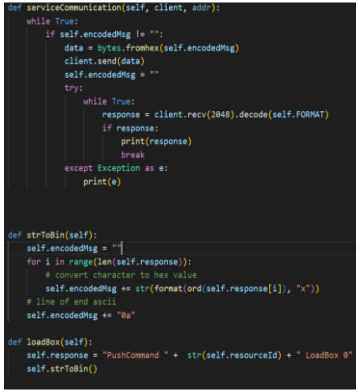
[8] Xia, Yuchen & Shenoy, Manthan & Jazdi, Nasser & Weyrich, Michael. (2023). Towards autonomous system: flexible modular production system enhanced with large language model agents.

[9] Ellen Jiang and Kristen Olson and Edwin Toh and Alejandra Molina and Aaron Michael Donsbach and Michael Terry and Carrie Jun Cai , “Prompt-based Prototyping with Large Language Models”, 2022

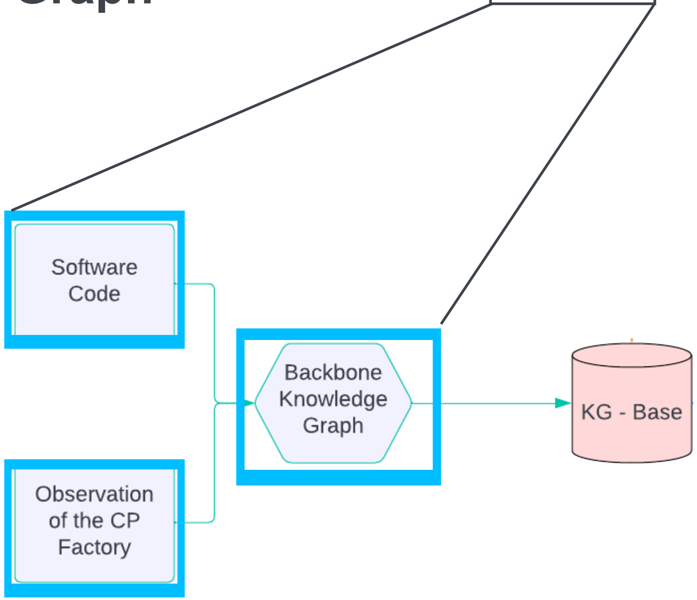
# Backups

# Functional System Design (1/3)
## Backbone Knowledge Graph


System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Software Code
Observation of the CP Factory
Backbone Knowledge Graph
KG - Base
Identifying causal relations
Querying the KG to answer user questions



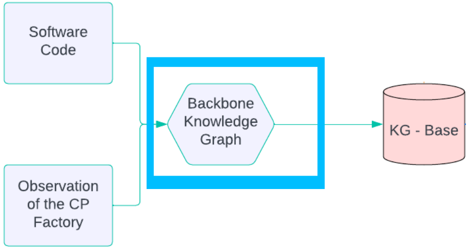
```
def serviceCommunication(self, client, addr):
    while True:
        if self.encodedMsg != "":
            data = bytes.fromhex(self.encodedMsg)
            client.send(data)
            self.encodedMsg = ""
            try:
                while True:
                    response = client.recv(2048).decode(self.FORMAT)
                    if response:
                        print(response)
                        break
            except Exception as e:
                print(e)



def strToBin(self):
    self.encodedMsg = ""
    for i in range(len(self.response)):
        # convert character to hex value
        self.encodedMsg += str(format(ord(self.response[i]), "x"))
    # line of end ascii
    self.encodedMsg += "0a"

def loadBox(self):
    self.response = "PushCommand " +  str(self.resourceId) + " LoadBox 0"
    self.strToBin()
```

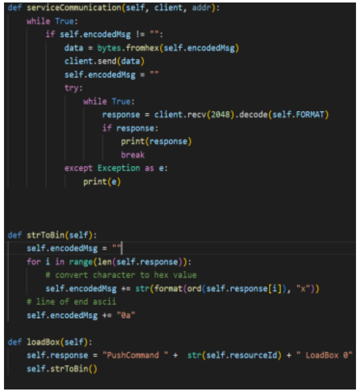


- Robotino moves to CP-Storage and docks itself there.
- The gripper in CP-Storage places the palette on the conveyor belt
- The palette is loaded on to the Robotino and then it undocks itself

# Functional System Design (1/3)

## Backbone KG - Manual

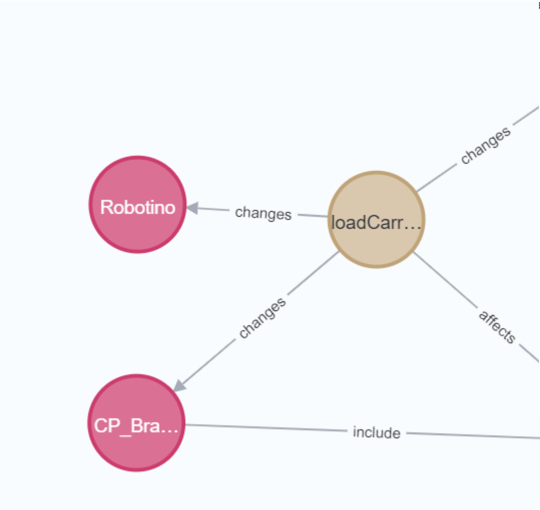


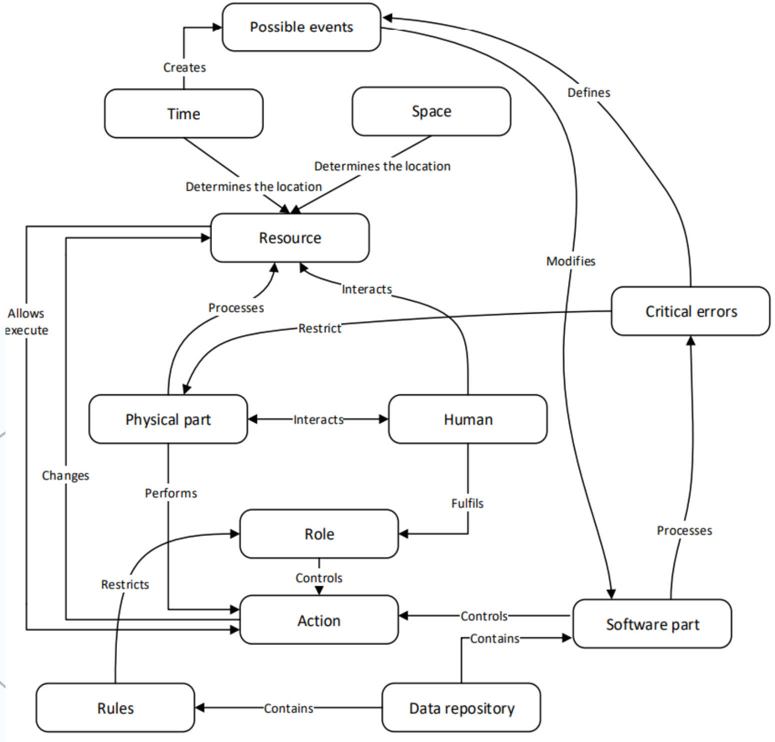


```
def serviceCommunication(self, client, addr):
    while True:
        if self.encodedMsg != "":
            data = bytes.fromhex(self.encodedMsg)
            client.send(data)
            self.encodedMsg = ""
            try:
                while True:
                    response = client.recv(2048).decode(self.FORMAT)
                    if response:
                        print(response)
                        break
            except Exception as e:
                print(e)


def strToBin(self):
    self.encodedMsg = ""
    for i in range(len(self.response)):
        # convert character to hex value
        self.encodedMsg += str(format(ord(self.response[i]), "x"))
    # line of end ascii
    self.encodedMsg += "0a"

def loadBox(self):
    self.response = "PushCommand " +  str(self.resourceId) + " LoadBox 0"
    self.strToBin()
```

Functional blocks in the code

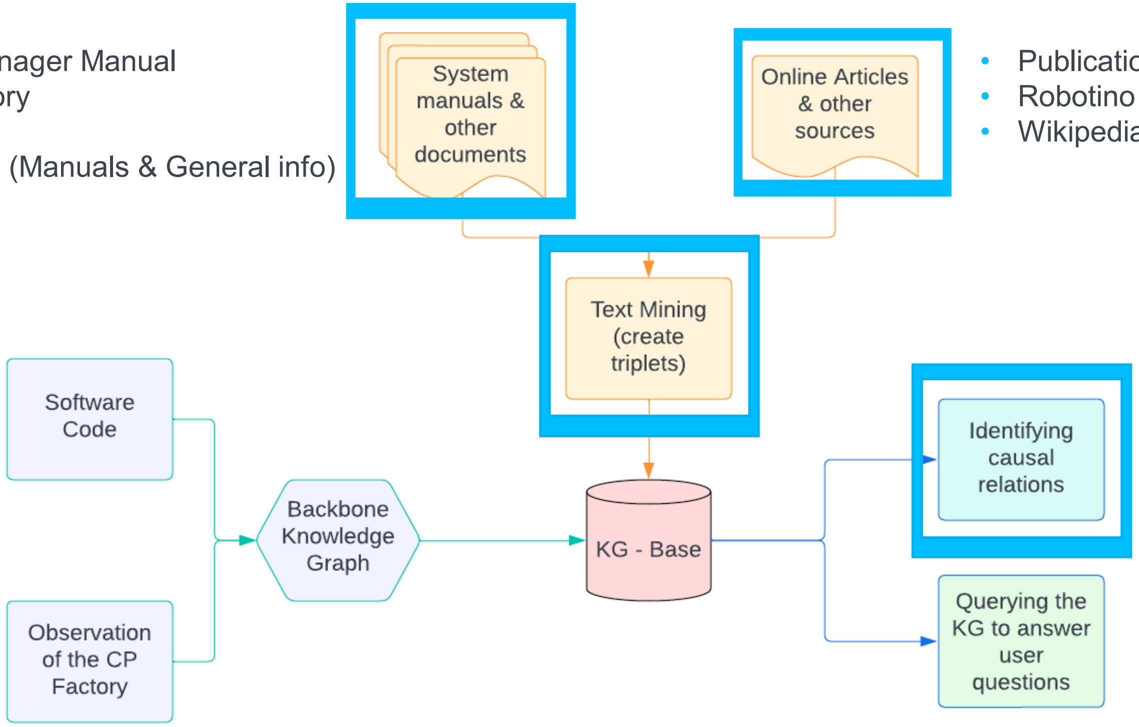
Mapped to ontology





# Functional System Design (2/3)
# Text to KG

- Fleet Manager Manual
- CP Factory
- MES
- Robotino (Manuals & General info)


System manuals & other documents
Online Articles & other sources
Text Mining (create triplets)
Software Code
Observation of the CP Factory
Backbone Knowledge Graph
KG - Base
Identifying causal relations
Querying the KG to answer user questions

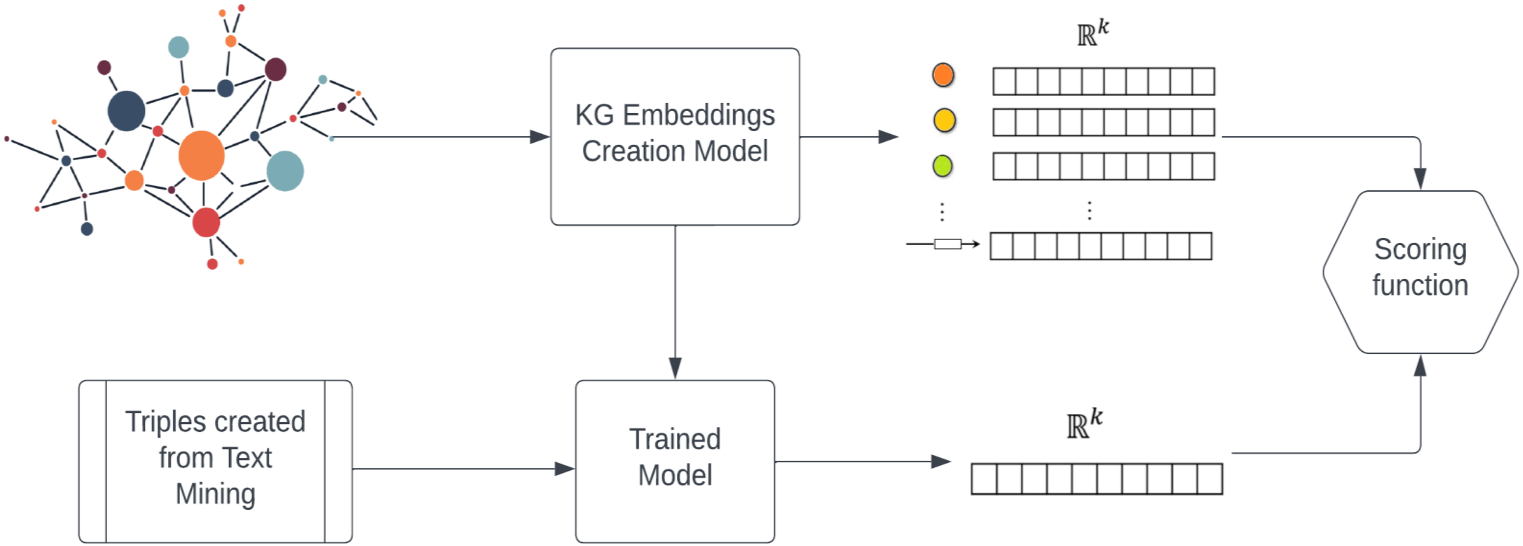
- Publications on CP factory
- Robotino wiki
- Wikipedia article

**Relation identification**:

- the verbs are lemmatized during text formatting
- A list of verbs (because, such, etc.) are identified and marked as causal

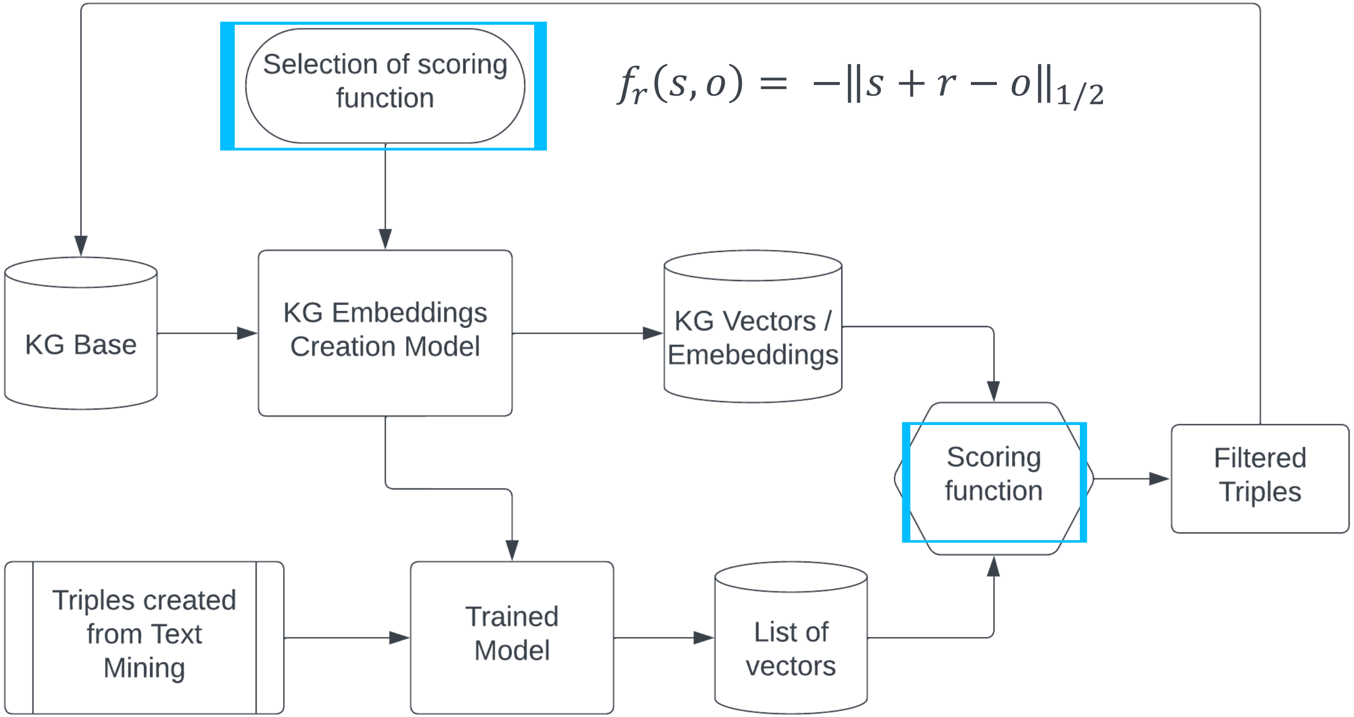
# Text to KG

## Filtering – Graph Embeddings


KG Embeddings Creation Model
$\mathbb{R}^k$
Scoring function
Triples created from Text Mining
Trained Model
$\mathbb{R}^k$

# Text to KG

## Filtering – Knowledge graph embeddings



Selection of scoring function
$f_r(s,o) = -\|s+r-o\|_{1/2}$
KG Base
KG Embeddings Creation Model
KG Vectors / Emebeddings
Scoring function
Filtered Triples
Triples created from Text Mining
Trained Model
List of vectors