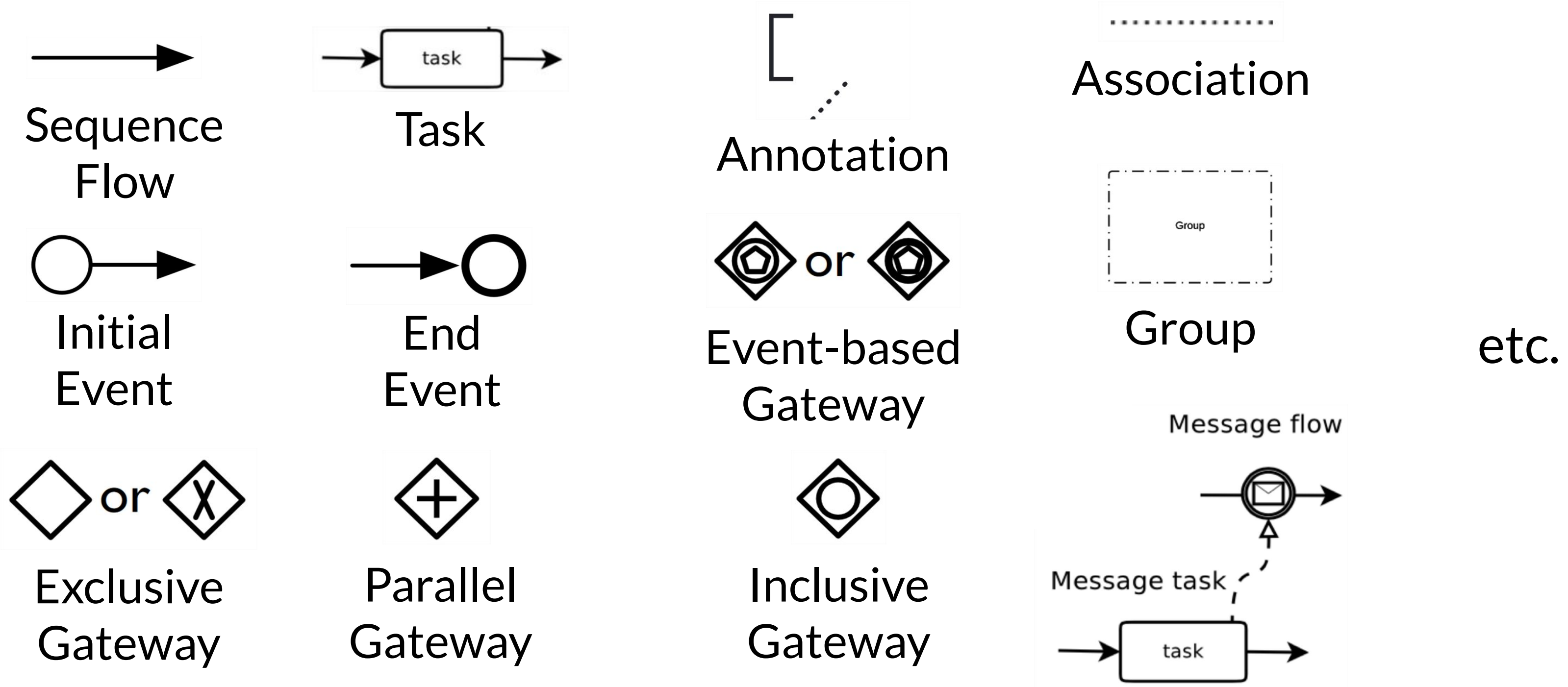
LLM-Based Generation of BPMN Workflows From Textual Descriptions

NIVON Quentin, SALAÜN Gwen

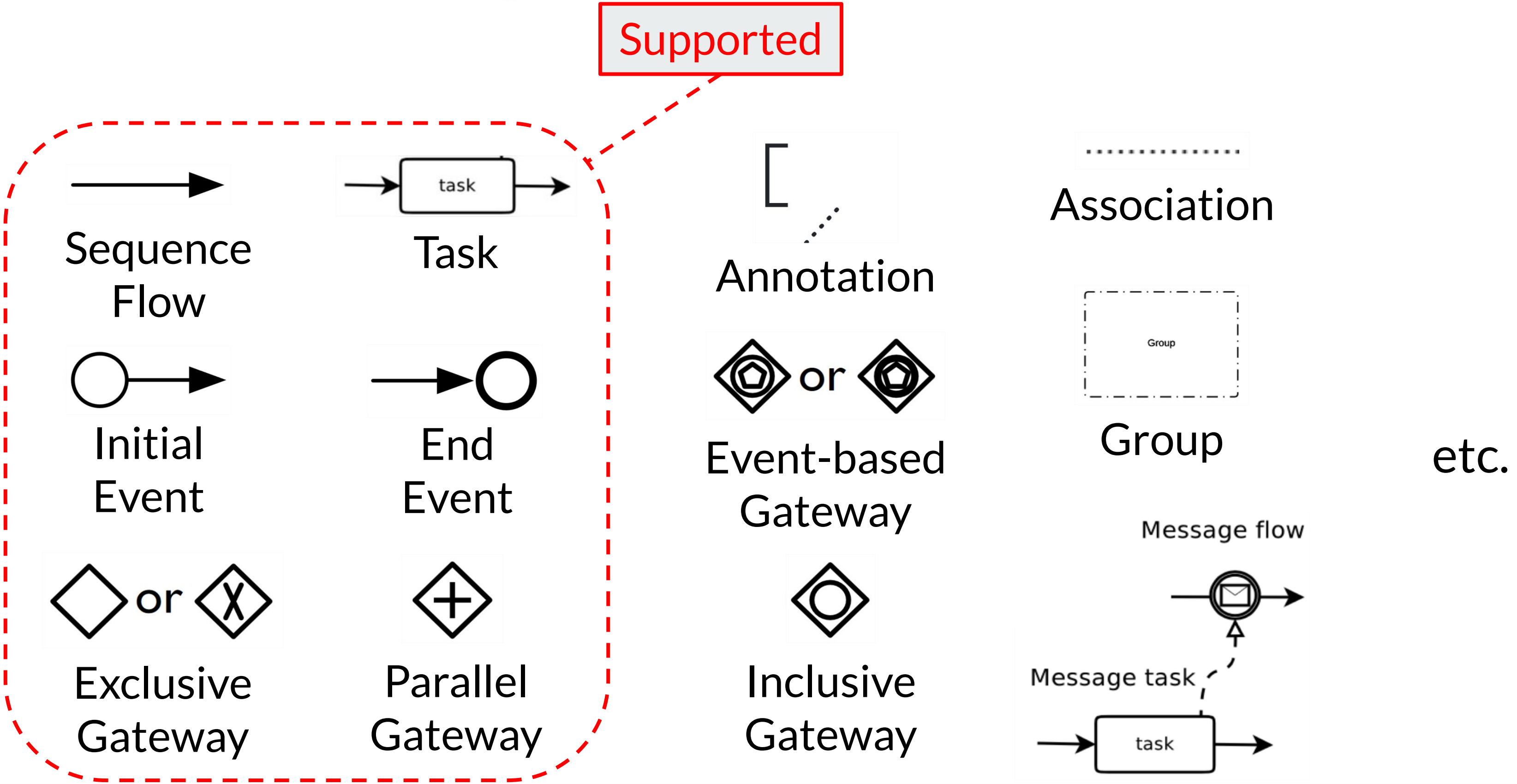
What is BPMN?

- A **workflow-based notation** created in 2004 by the Business Process Management Initiative (BPMI) and the Object Management Group (OMG).
- It aims at **representing business processes** in a way that is **understandable for both experienced and novice users.**
- An **ISO/IEC standard** since version 2.0 in 2013.

BPMN Syntax


task
Sequence Flow
Task
Annotation
Association
Group
Initial Event
End Event
or
Event-based Gateway
Group
etc.
Message flow
or
Exclusive Gateway
Parallel Gateway
Inclusive Gateway
Message task
task

BPMN Syntax



Supported
task
Sequence Flow
Task
Initial Event
End Event
or
Exclusive Gateway
Parallel Gateway
Annotation
Association
or
Event-based Gateway
Group
Group
etc.
Message flow
Message task
task
Inclusive Gateway

Context

- Companies are making use of the BPMN notation to **represent their business processes.**

- They **hire experts** to analyse and design the **most adequate BPMN process** according to their needs.

- These processes are often **syntactically/semantically incorrect**.

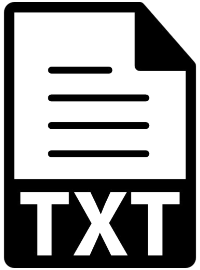
Research Questions

- What if you do not know **how to write BPMN**?

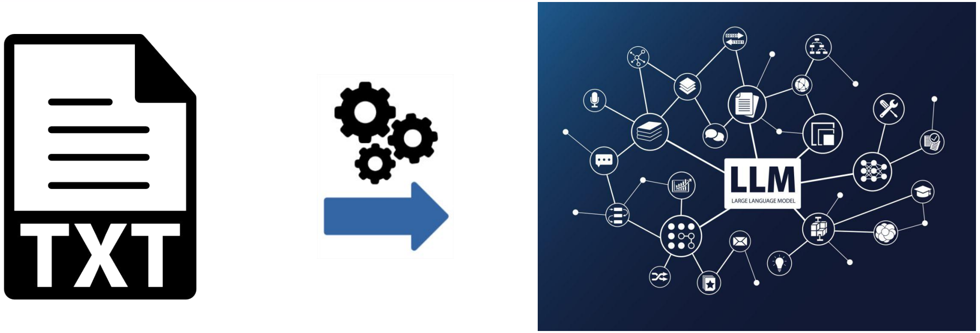
- What if you do not want to **spend time designing** your BPMN process graphically?

- How can you be sure that your BPMN process is **syntactically/semantically correct**?

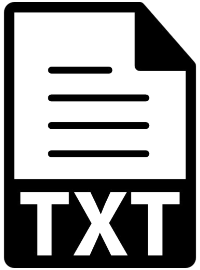
General Solution

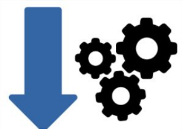


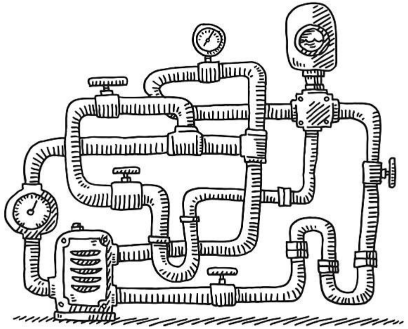
General Solution



General Solution

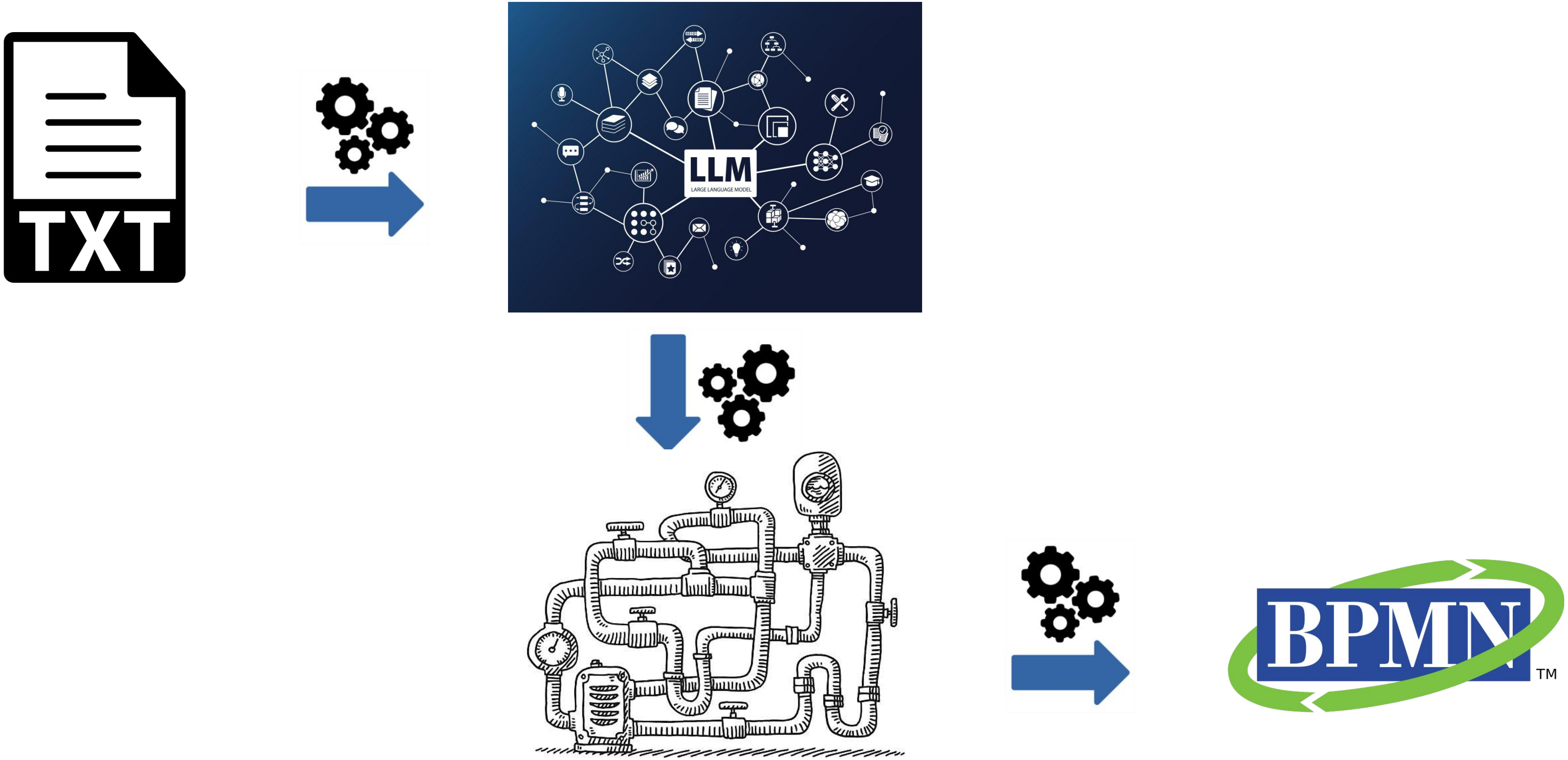






Internal Computations

General Solution



Internal Computations

Overview of our Solution

First of all, an employee CollectGoods. Then, the client PayForDelivery while the employee PrepareParcel. Finally, the company can either DeliverByCar or DeliverByDrone (depending on the distance for example)

Textual Representation of the Process

Overview of our Solution

First of all, an employee CollectGoods. Then, the client PayForDelivery while the employee PrepareParcel. Finally, the company can either DeliverByCar or DeliverByDrone (depending on the distance for example)

Textual Representation of the Process

GPT-3.5

Large Language Model (LLM)

Overview of our Solution

First of all, an employee CollectGoods. Then, the client PayForDelivery while the employee PrepareParcel. Finally, the company can either DeliverByCar or DeliverByDrone (depending on the distance for example)

Textual Representation of the Process

Large Language Model (LLM)

- CollectGoods < (PayForDelivery, PrepareParcel)
- (PayForDelivery, PrepareParcel) < (DeliverByCar, DeliverByDrone)

$$\begin{aligned} \langle \mathrm{E} \rangle &::= \mathtt{t} \mid (\langle \mathrm{E} \rangle) \mid \\ &\quad \langle \mathrm{E}_1 \rangle \langle \mathrm{op} \rangle \langle \mathrm{E}_2 \rangle \mid (\langle \mathrm{E}_1 \rangle)* \\ \langle \mathrm{op} \rangle &::= \text{`}|\text{'} \mid \text{`}\&\text{'} \mid \text{`}<\text{'} \mid \text{`};\text{'} \end{aligned}$$

Expressions Following an Internal Grammar

Overview of our Solution

First of all, an employee CollectGoods. Then, the client PayForDelivery while the employee PrepareParcel. Finally, the company can either DeliverByCar or DeliverByDrone (depending on the distance for example)

Textual Representation of the Process

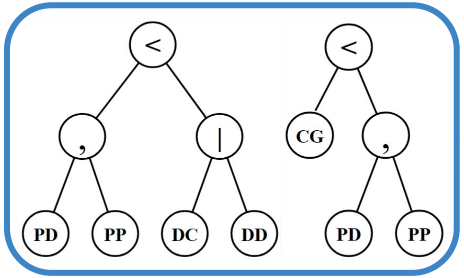
GPT-3.5

Large Language Model (LLM)

- CollectGoods < (PayForDelivery, PrepareParcel)
- (PayForDelivery, PrepareParcel) < (DeliverByCar, DeliverByDrone)

$$
\begin{aligned}
\langle \mathrm{E} \rangle ::= \quad & \texttt{t} \quad | \quad (\langle \mathrm{E} \rangle) \quad | \\
& \langle \mathrm{E}_1 \rangle \ \langle \mathrm{op} \rangle \ \langle \mathrm{E}_2 \rangle \quad | \quad (\langle \mathrm{E}_1 \rangle)* \\
\langle \mathrm{op} \rangle ::= \quad & `|\text{'} \quad | \quad `\&\text{'} \quad | \quad `<\text{'} \quad | \quad `,\text{'}
\end{aligned}
$$

Expressions Following an Internal Grammar



Abstract Syntax Trees

Overview of our Solution

First of all, an employee CollectGoods. Then, the client PayForDelivery while the employee PrepareParcel. Finally, the company can either DeliverByCar or DeliverByDrone (depending on the distance for example)

Textual Representation of the Process

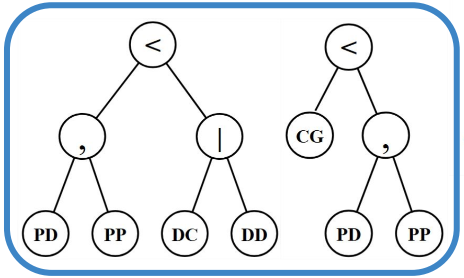
GPT-3.5

Large Language Model (LLM)

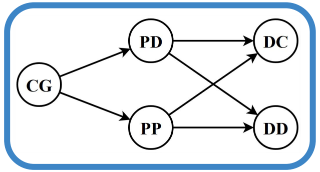
- CollectGoods < (PayForDelivery, PrepareParcel)
- (PayForDelivery, PrepareParcel) < (DeliverByCar, DeliverByDrone)

$$
\begin{aligned}
\langle \mathrm{E} \rangle \ ::= \ & \mathtt{t} \ \mid \ (\langle \mathrm{E} \rangle) \ \mid \\
& \langle \mathrm{E}_1 \rangle \ \langle \mathrm{op} \rangle \ \langle \mathrm{E}_2 \rangle \ \mid \ (\langle \mathrm{E}_1 \rangle)* \\
\langle \mathrm{op} \rangle \ ::= \ & \text{‘}|\text{’} \ \mid \ \text{‘}\&\text{’} \ \mid \ \text{‘}<\text{’} \ \mid \ \text{‘},\text{’}
\end{aligned}
$$

Expressions Following an Internal Grammar

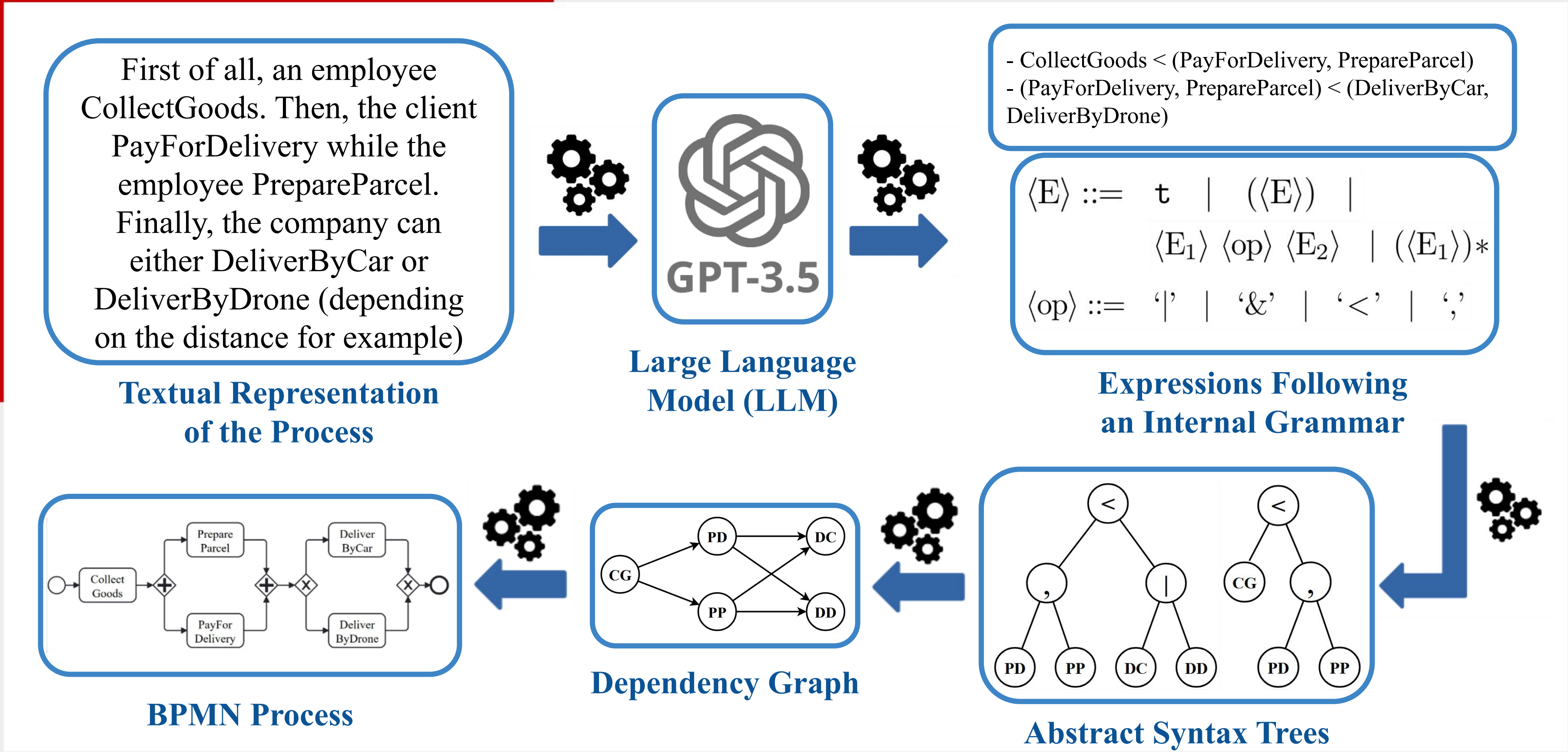


Abstract Syntax Trees



Dependency Graph

Overview of our Solution



Detailed Solution – Step 1

The user first has to write a **textual description** of the process-to-be.

First, the banker either CreateProfile (CP) for the user, or, if it is not needed, he RetrieveCustomerProfile (RCP) which triggers the system to perform the AnalyseCustomerProfile (ACP) task.
Then, the user executes the task ReceiveSupportDocuments (RSD) so that the system can start UpdateInfoRecords (UID) and perform a BackgroundVerification (BV).
If the verification finds missing or incorrect information, the system RequestAdditionalInfo (RAI) to the user, who has to ReceiveSupportDocuments (RSD) again.
Otherwise, the process ends with CreateAccount (CA).

Detailed Solution – Step 2

The textual description is then **given to a (fine-tuned) LLM** (GPT 3.5 atm).

First, the banker either CreateProfile (CP) for the user, or, if it is not
needed, he RetrieveCustomerProfile (RCP) which triggers the system to perform the AnalyseCustomerProfile (ACP) task.
Then, the user executes the task ReceiveSupportDocuments (RSD) so that the system can start UpdateInfoRecords (UID) and perform a BackgroundVerification (BV).
If the verification finds missing or incorrect information, the system RequestAdditionalInfo (RAI) to the user, who has to ReceiveSupportDocuments (RSD) again.
Otherwise, the process ends with CreateAccount (CA).

GPT - 3.5

The textual description is then **given to a (fine-tuned) LLM** (GPT 3.5 atm).

First, the banker either <u>CreateProfile</u> (CP) for the user, or, if it is not
needed, he <u>RetrieveCustomerProfile</u> (RCP) which triggers the system to perform the <u>AnalyseCustomerProfile</u> (ACP) task.
Then, the user executes the task <u>ReceiveSupportDocuments</u> (RSD) so that the system can start <u>UpdateInfoRecords</u> (UID) and perform a <u>BackgroundVerification</u> (BV).
If the verification finds missing or incorrect information, the system <u>RequestAdditionalInfo</u> (RAI) to the user, who has to <u>ReceiveSupportDocuments</u> (RSD) again.
Otherwise, the process ends with <u>CreateAccount</u> (CA).

OpenAI

GPT - 3.5

The LLM processes the description and returns a **set of expressions** following an **internal grammar**.

$$\langle \mathrm{E} \rangle ::= \quad \mathtt{t} \quad | \quad (\langle \mathrm{E} \rangle) \quad | \quad \langle \mathrm{E}_1 \rangle \ \langle \mathrm{op} \rangle \ \langle \mathrm{E}_2 \rangle \quad | \quad (\langle \mathrm{E}_1 \rangle)*$$

$$\langle \mathrm{op} \rangle ::= \quad `|' \quad | \quad `\&' \quad | \quad `<' \quad | \quad `,'$$

Detailed Solution – Step 3

Given our description, the LLM returns **three expressions**:

```
(RetrieveCustomerProfile < AnalyseCustomerProfile) | CreateProfile
```

Detailed Solution – Step 3

Given our description, the LLM returns **three expressions**:

(RetrieveCustomerProfile < AnalyseCustomerProfile) | CreateProfile

(RetrieveCustomerProfile, AnalyseCustomerProfile, CreateProfile) <
(ReceiveSupportDocuments < (UpdateInfoRecords, BackgroundVerification))

Detailed Solution – Step 3

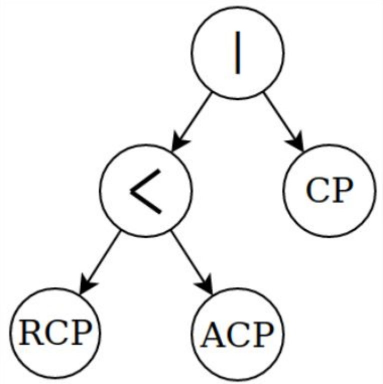
Given our description, the LLM returns **three expressions**:

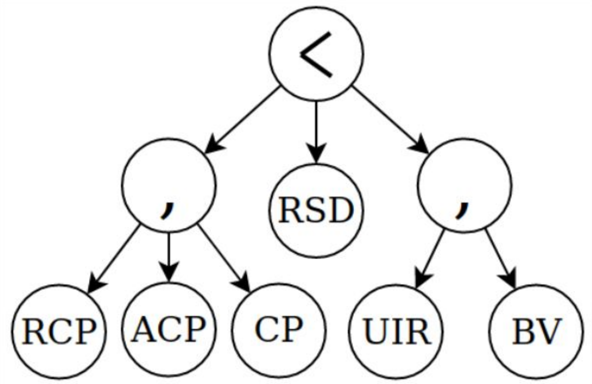
(RetrieveCustomerProfile < AnalyseCustomerProfile) | CreateProfile

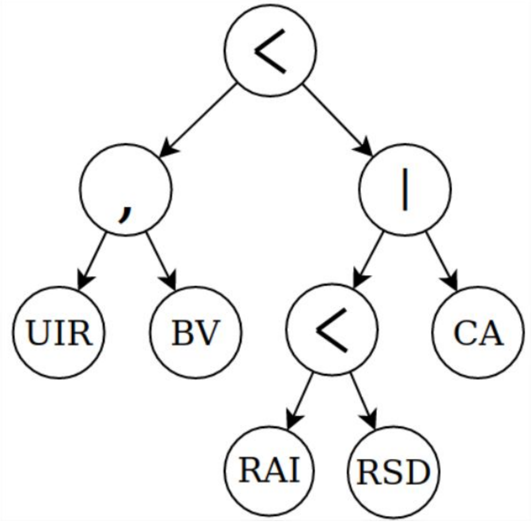
(RetrieveCustomerProfile, AnalyseCustomerProfile, CreateProfile) < (ReceiveSupportDocuments < (UpdateInfoRecords, BackgroundVerification))

(UpdateInfoRecords, BackgroundVerification) < ((RequestAdditionalInfo < ReceiveSupportDocuments) | CreateAccount)

These expressions are then **mapped to** their corresponding **abstract syntax trees (ASTs)**.

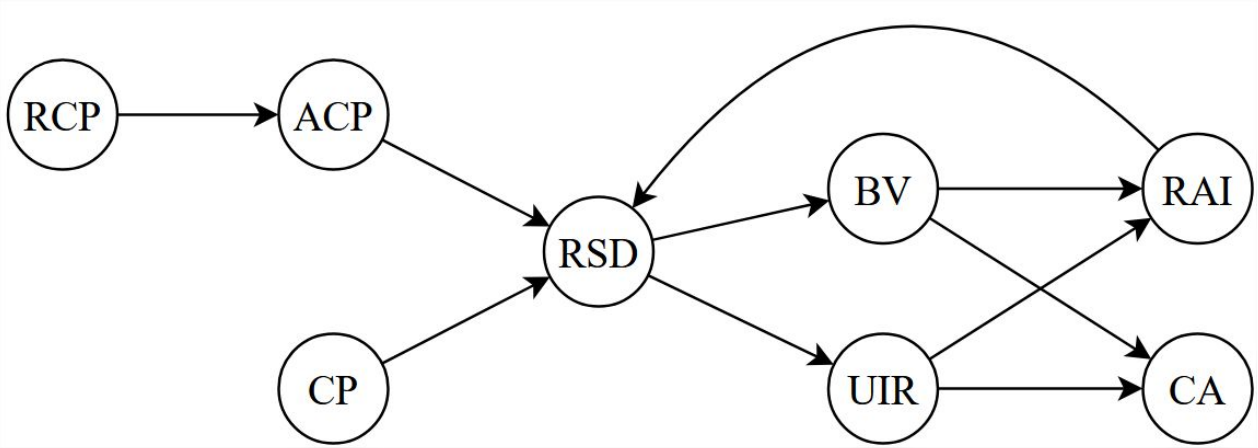




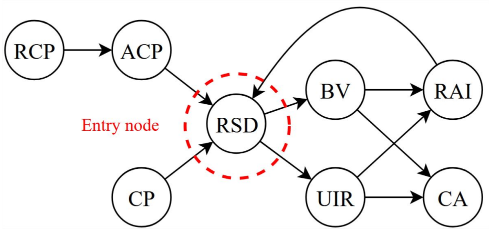


Detailed Solution – Step 5

The **sequential information** contained in the multiple ASTs is gathered to obtain a **cleaner** representation of it, called **dependency graph**.



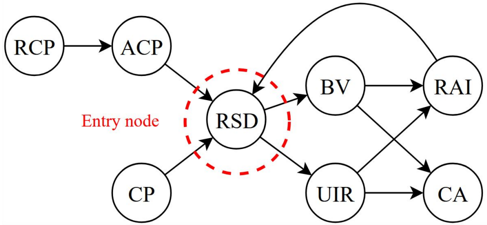
If the dependency graph **contains loops**, they are **analysed**, and all the **information needed to reconstruct** them is extracted.



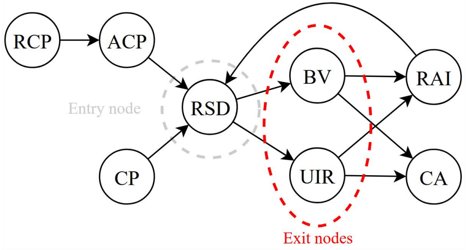
Entry node(s) computation

Detailed Solution – Step 6

If the dependency graph **contains loops**, they are **analysed**, and all the **information needed to reconstruct** them is extracted.



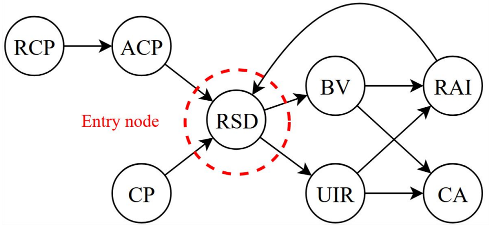
Entry node(s) computation



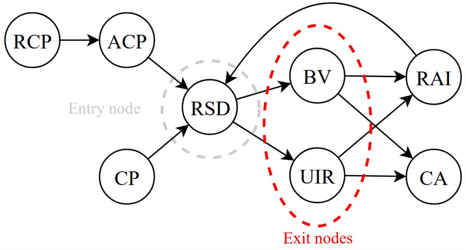
Exit node(s) computation

Detailed Solution – Step 6

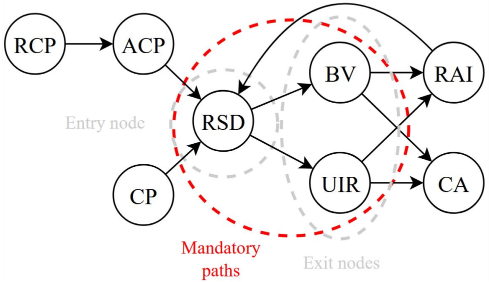
If the dependency graph **contains loops**, they are **analysed**, and all the **information needed to reconstruct** them is extracted.



Entry node(s) computation



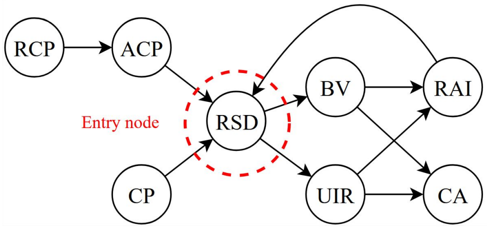
Exit node(s) computation



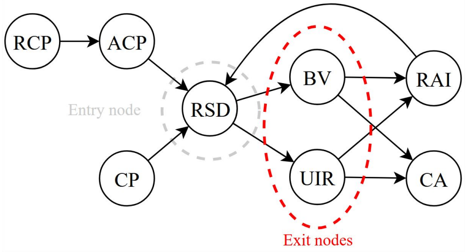
Mandatory path(s) computation

Detailed Solution – Step 6

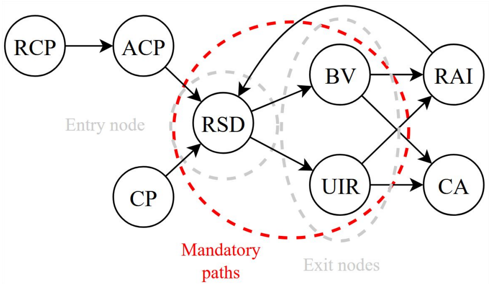
If the dependency graph **contains loops**, they are **analysed**, and all the **information needed to reconstruct** them is extracted.



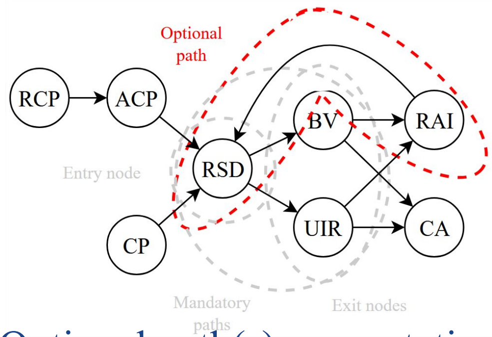
Entry node(s) computation



Exit node(s) computation



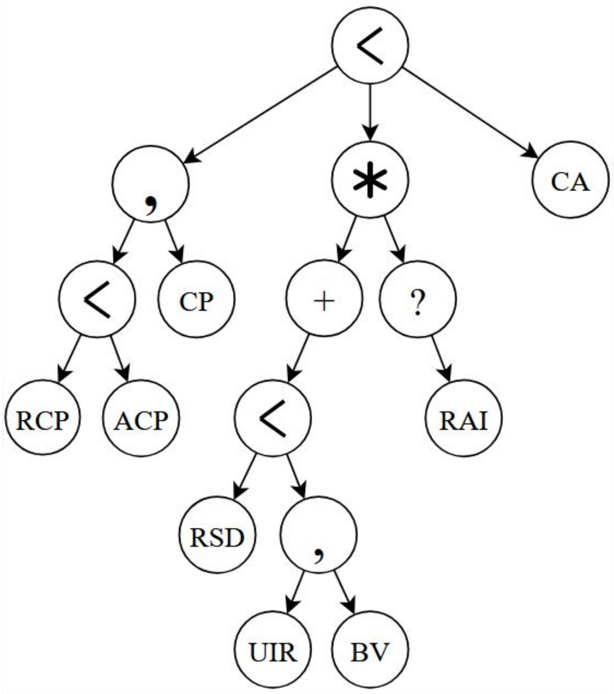
Mandatory path(s) computation



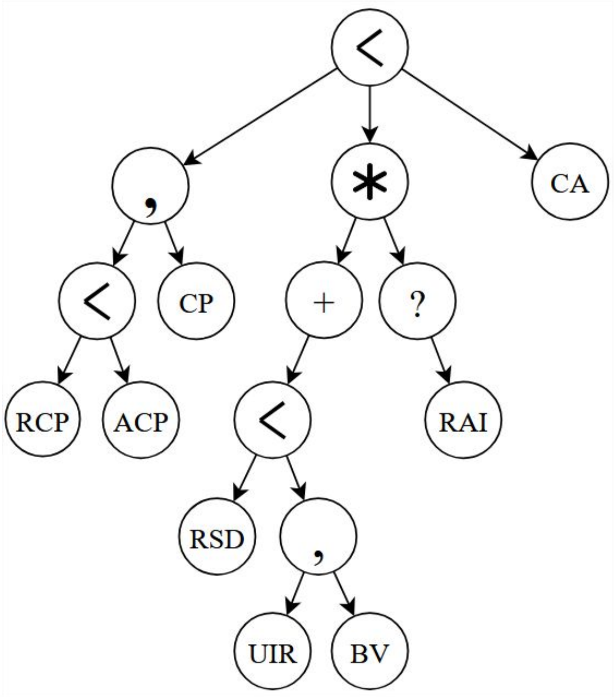
Optional path(s) computation

Detailed Solution – Step 7

Once the loops have been retrieved, the **AST corresponding to the dependency graph** is built from the dependency graph.



The **ultimate step** to obtain an AST containing all the information belonging to the original expressions consists in **inserting the choices**.

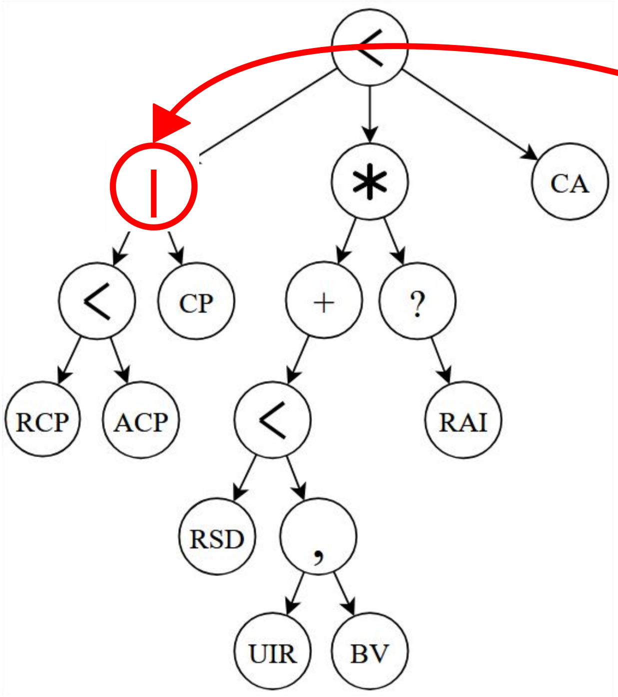


(RCP < ACP) | CP

(RAI < RSD) | CA

Detailed Solution – Step 8

The **ultimate step** to obtain an AST containing all the information belonging to the original expressions consists in **inserting the choices**.

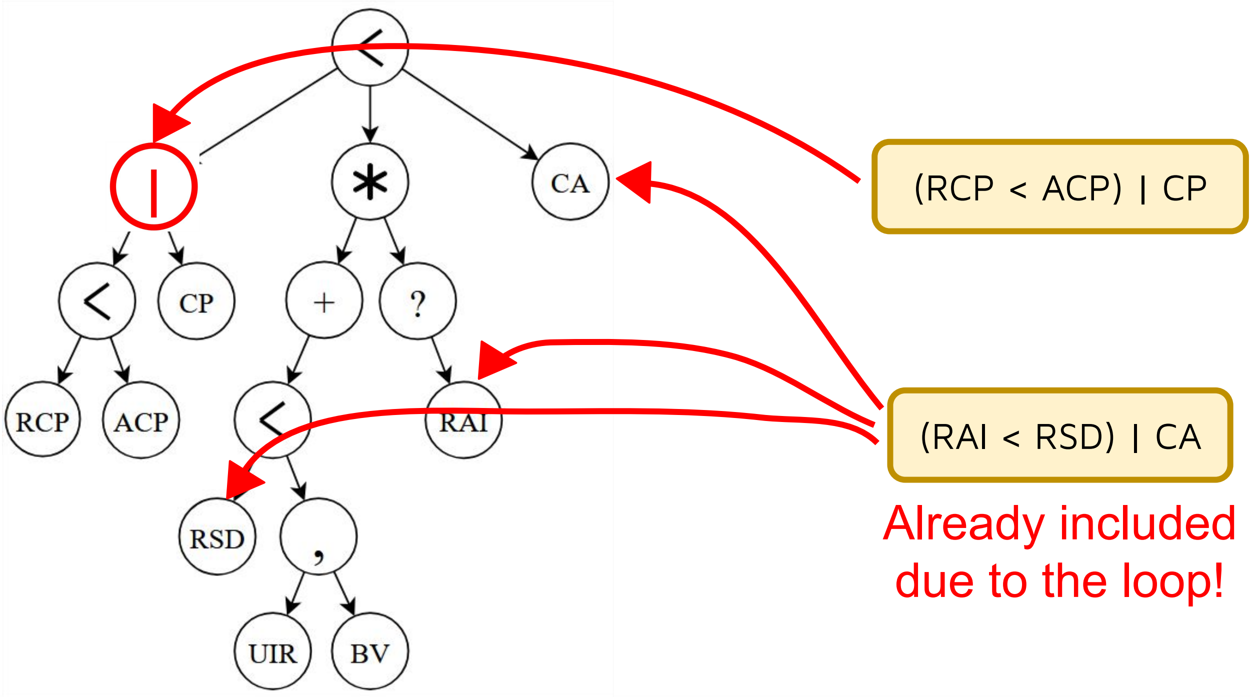


(RCP < ACP) | CP

(RAI < RSD) | CA

Detailed Solution – Step 8

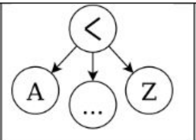
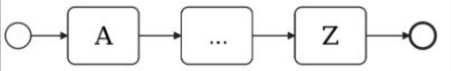
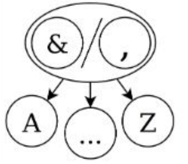
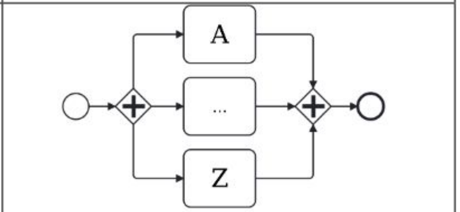
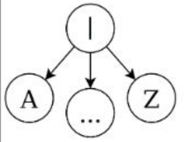
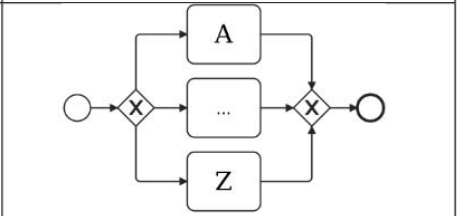
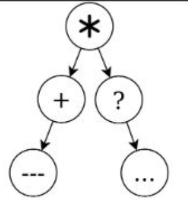
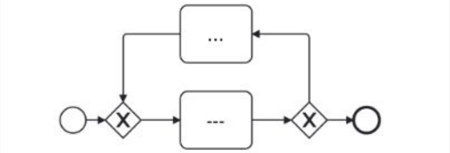
The **ultimate step** to obtain an AST containing all the information belonging to the original expressions consists in **inserting the choices**.



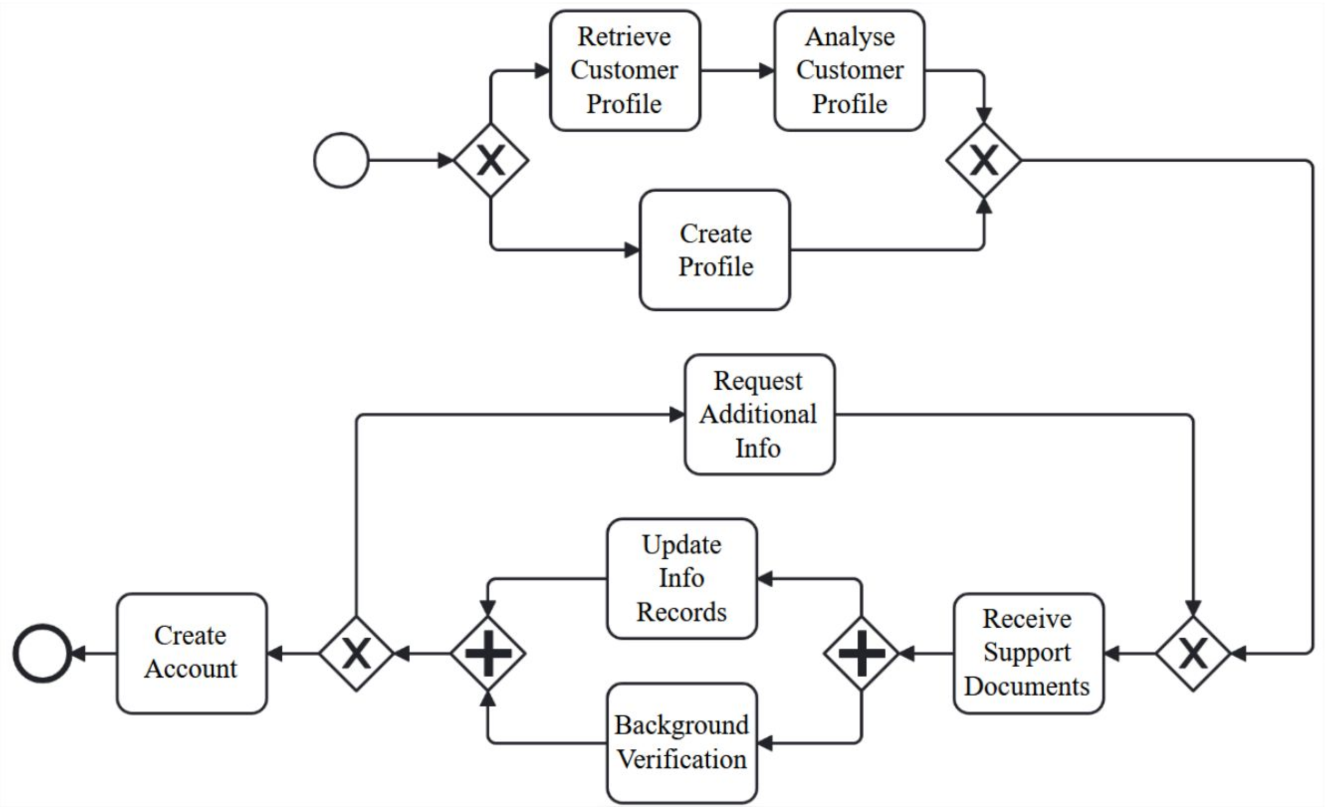
Detailed Solution – Step 9

Now that the AST contains the choices, the **BPMN process is ready to be generated**.

To do so, **patterns** are applied recursively to the merged AST, **starting from the deepest nodes** (leafs).

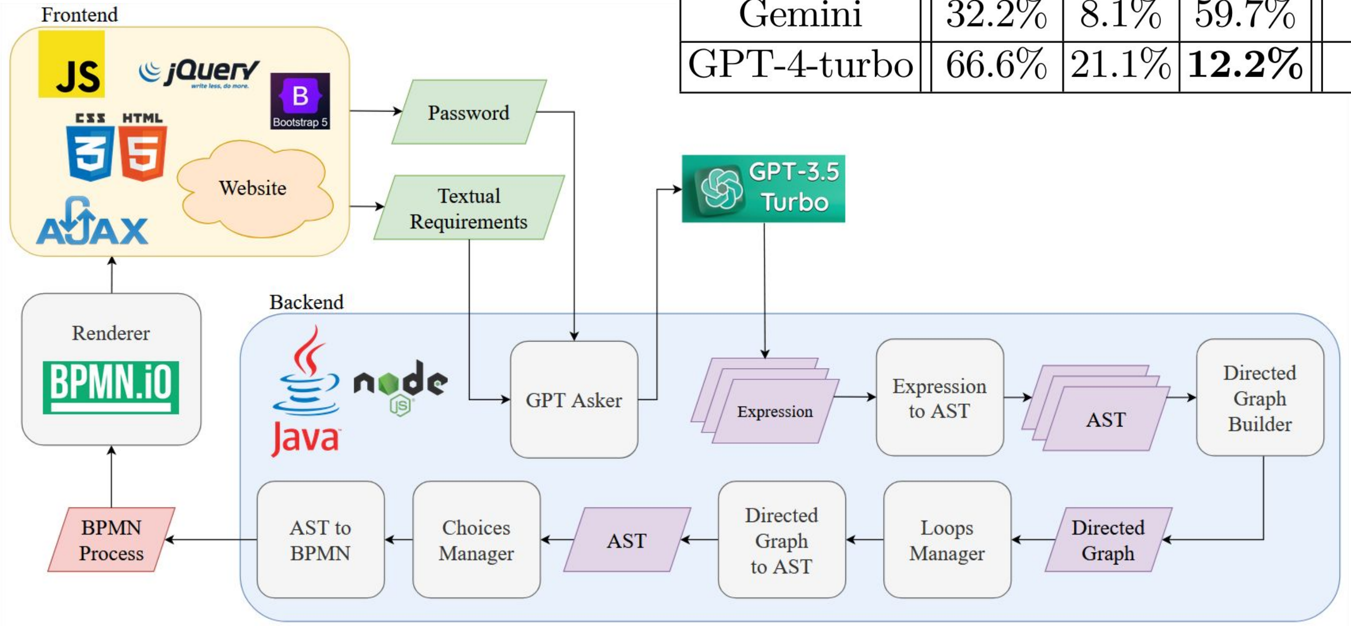
Pattern	AST	BPMN
(1) Sequential Pattern		
(2) Parallel Pattern		
(3) Choice Pattern		
(4) Loop Pattern		

Detailed Solution – Result



Tool – Presentation & Experiments

Tool/Model	✓	?	✗	Avg. Exec. Time (s)
Our tool	**78.5%**	**8%**	13.5%	**4.07**
ProMoAI	50%	8.7%	41.2%	24.7
Gemini	32.2%	8.1%	59.7%	8.32
GPT-4-turbo	66.6%	21.1%	**12.2%**	19.2



Tool – Online Version

Link: https://quentinnivon.github.io/pages/givup.html

Conclusion

In this work, we proposed a **9 steps approach** aiming at automatically designing **syntactically and semantically correct BPMN** processes from a **textual description** of the requirements.

This work offers several **perspectives**:

- Get rid of ASTs to allow more complex constructs (intricate loops/unbalanced gateways)
- Formalise the transformation operations
- Add new features (such as model checking of textual properties)
- Adapt the description to (visual) process changes

[TSE'25]

[FSE'25]

David's work