

42nd International Conference on Conceptual Modeling

ER Forum 1 – Tools & Technology

Lisbon, 7 November 2023

CONCEPTUAL MODEL INTERPRETER FOR LARGE LANGUAGE MODELS

Felix Härer

University of Fribourg, Switzerland



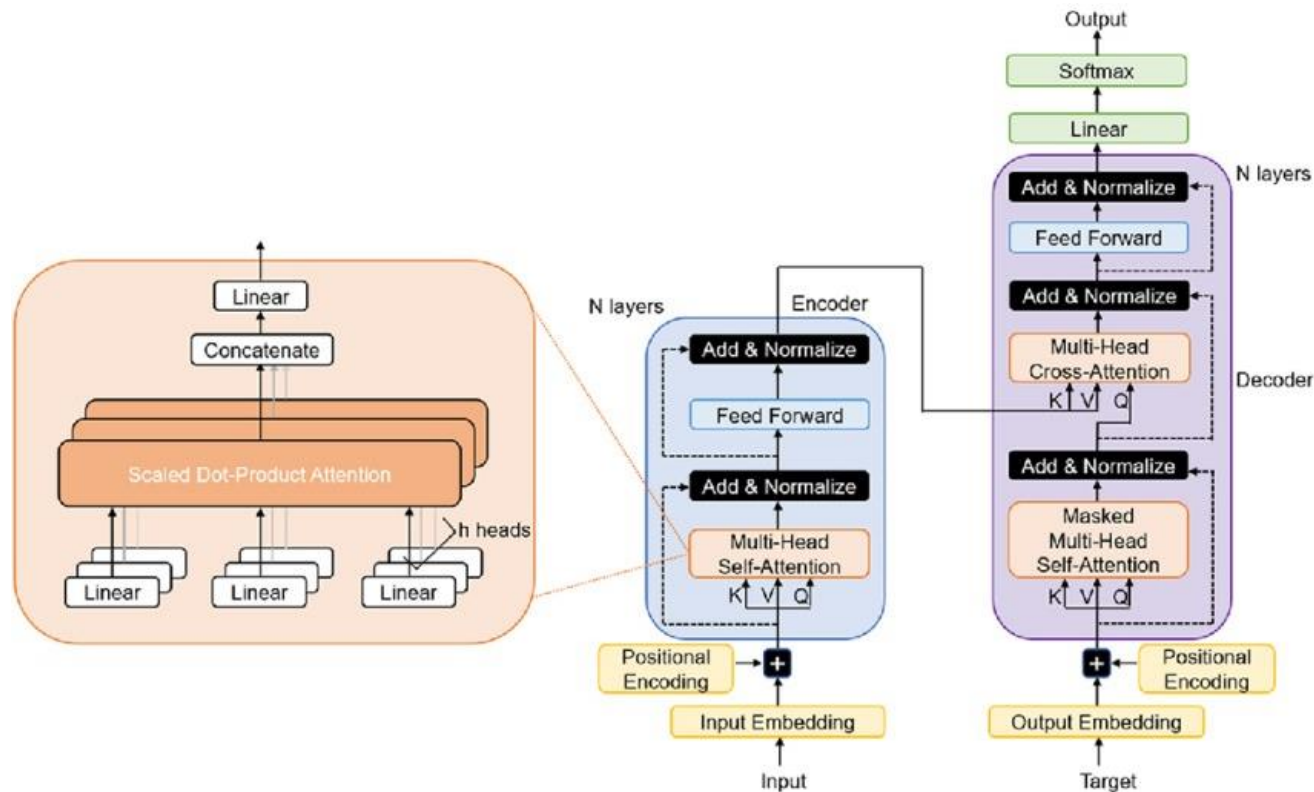
UNIVERSITÉ DE FRIBOURG
UNIVERSITÄT FREIBURG

OUTLINE

1. Evolution of LLM Applications
2. Conceptual Modeling With LLMs
 - State of the Art
 - Research Objectives and Exploratory Approach
 - Architecture and Prototype
3. Experimental Results
 - UML Conceptualization
 - Graph Generation From Data
4. Discussion and Outlook

1. EVOLUTION OF LLM APPLICATIONS

- Large Language Models (LLMs)
 - Support for language-related tasks, primarily text completion
 - Recent success of Generative Pre-trained Transformer models (GPTs)



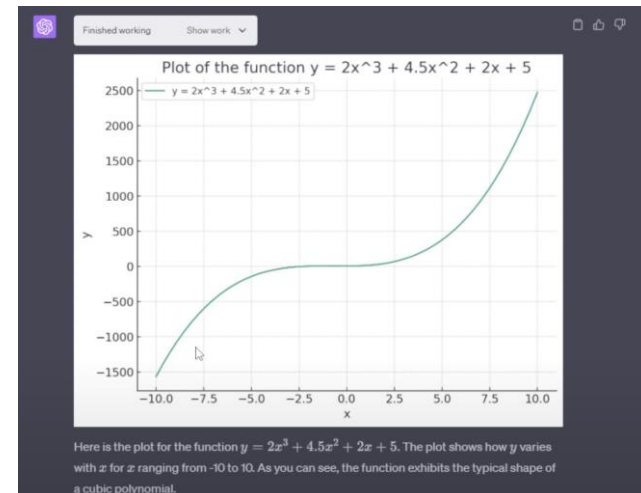
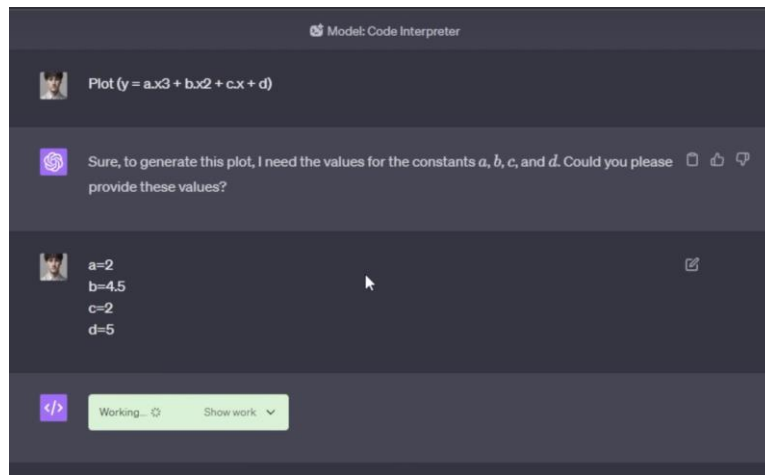
Vaswani et al. (2017): Attention is all you need.

Figure by Hu and Buehler (2023): Deep language models for interpretative and predictive materials science.

1. EVOLUTION OF LLM APPLICATIONS

■ Evolvement of Applications

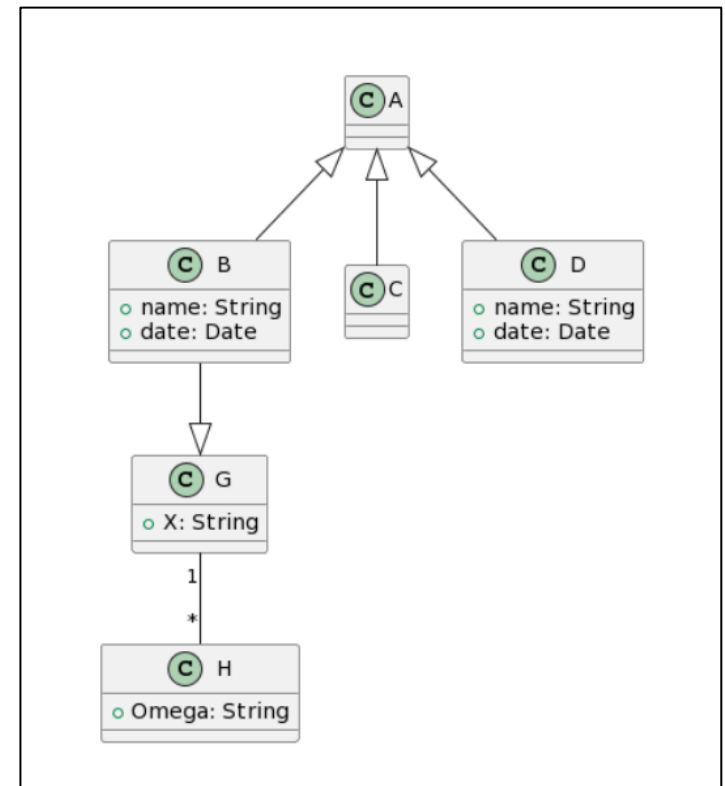
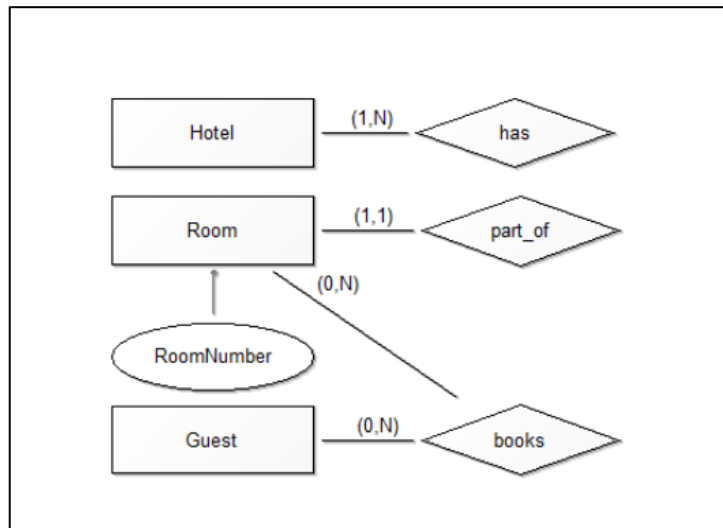
- Language-related:
Generation, summarization, question answering, translation, analysis etc.
- Multi-modality:
Audio transcription and translation, voice cloning, image description and generation, video and 3D scene generation, 3D object generation etc.
- Software:
Analysis and generation of source code and, recently, code interpretation



Images by Moritz Kremb, @moritzkremb, X

2. CONCEPTUAL MODELING WITH LLMS

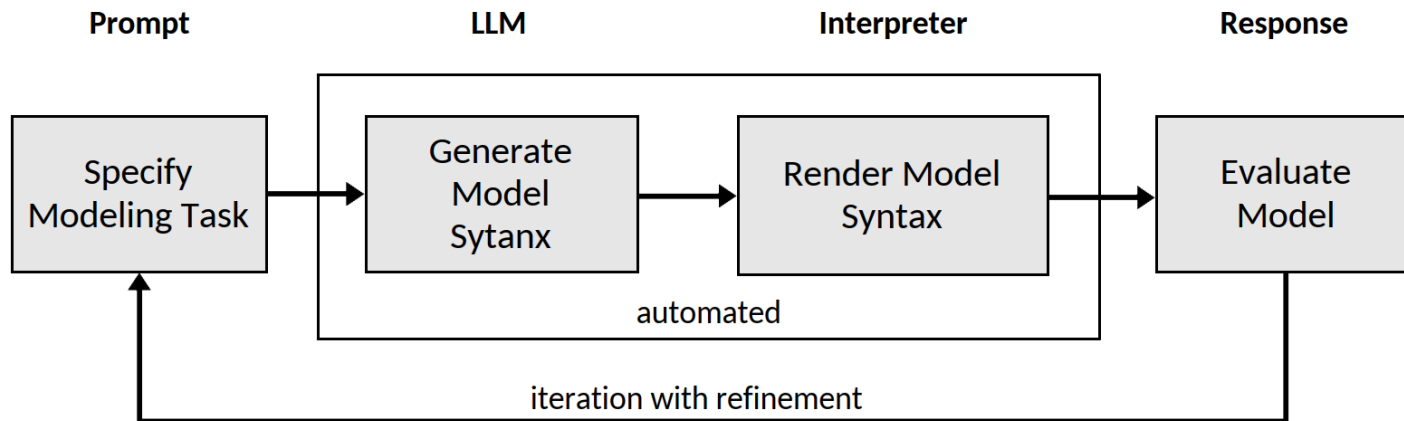
- State of the Art
 - Software models and conceptual models (e.g. *Cámara et al.*, *Burgueño et al.*) such as UML class diagrams
 - BPM applications (*Vidgof et al.*)
 - UML, ER, BPMN, domain-specific models, abstract UML generation (*Fill et al.*)



Fill et al. (2023): Conceptual Modeling and Large Language Models: Impressions From First Experiments With ChatGPT

2. CONCEPTUAL MODELING WITH LLMS

- State of the art applied for Conceptual Modeling with LLMs
 - Natural language descriptions
 - Model source code generation, e.g., for PlantUML
 - Code generation with execution with instant feedback



→ Code Interpreter for Conceptual Models

2. CONCEPTUAL MODELING WITH LLMS

→ Research Objectives

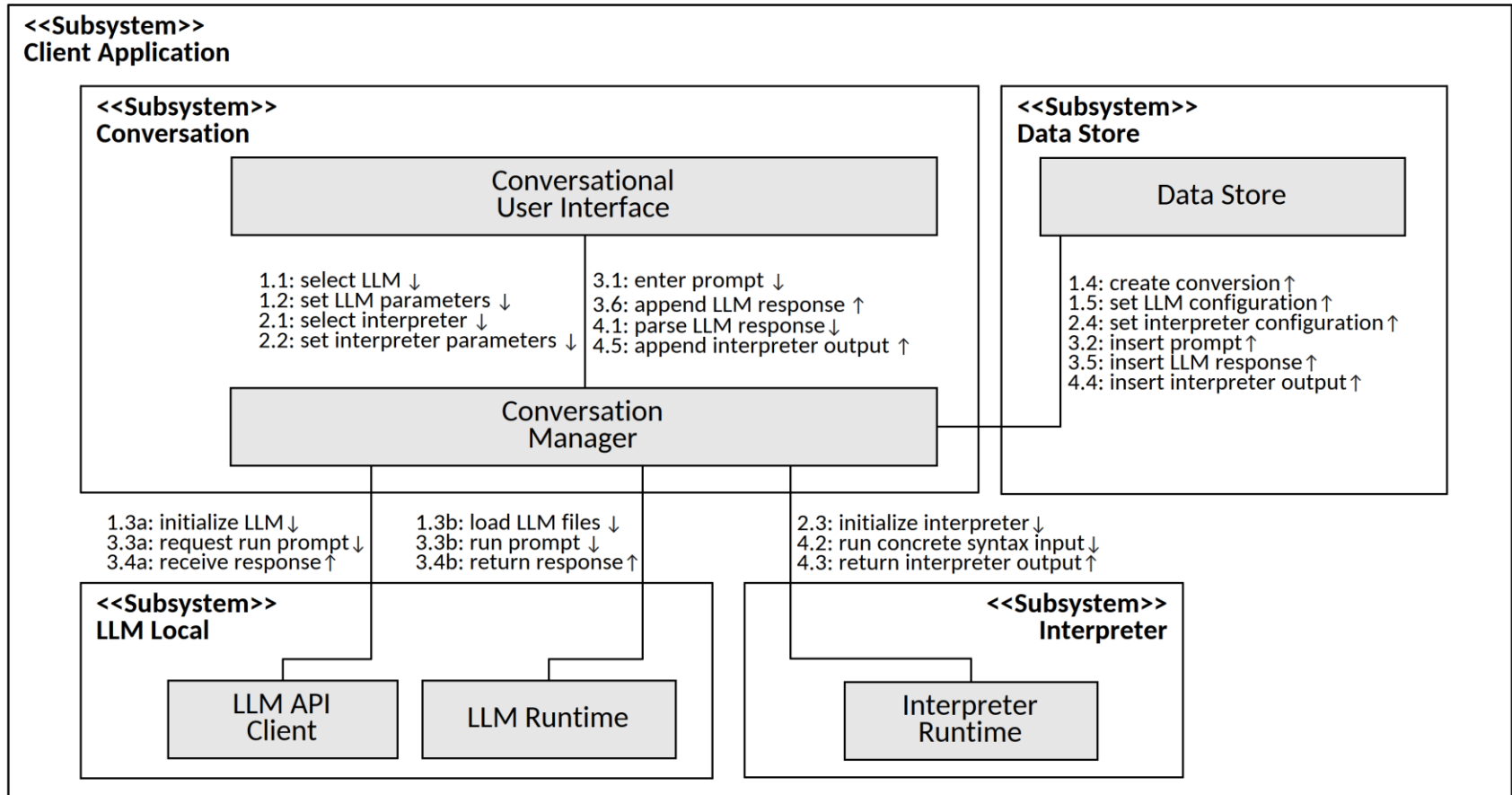
- (1) Determine whether a conversational approach with instant feedback and step-wise refinement can be realized with state-of-the-art commercial or open source LLMs and interpreters.
- (2) Determine how a possible realization could be constructed in terms of an architecture.

→ Exploratory Research Approach

- Requirement specification, construction of an architecture and prototype supporting multiple LLMs and interpreters to address objective (2)
- Experimental application of LLMs and interpreters to address objective (1), in particular GPT-4 and Llama 2 LLMs with Plantweb interpreters

2. CONCEPTUAL MODELING WITH LLMS

- Architecture Overview



UML Communication Diagram

2. CONCEPTUAL MODELING WITH LLMS

■ Prototype

LLM Settings

Model

OpenAI/gpt-4

API keys found: Replicate, OpenAI

temperature: 0.20

top_p: 0.90

frequency_penalty: 0.00


presence_penalty: 0.00



Interpreter Settings

Interpreter

Plantweb/PlantUML

 RUNNING... Stop Deploy

 Create a class diagram in PlantUML code according to the following description. An order consists of an order header and one or more order itmes. Each order has an id and a timestamp. In an order header, a customer is referenced in addition to a billing address, a shipping address and a payment method. For the customer, an id as well as first name, last name and a registration timestamp are known in addition to a billing address and potentially multiple shipping addresses.

  Running Inference: OpenAI/gpt-4 ...

Here is a PlantUML class diagram based on your description:

```
@startuml
class Order {
+id
+timestamp
+OrderHeader
+OrderItems[]
}

class OrderHeader
```

Your message

<https://github.com/fhaer/llm-cmi>

3. RESULTS – UML CONCEPTUALIZATION

- Conceptual modeling task to create a UML class diagram
- Scenario description in natural language
- Generation of PlantUML source code using
 - LLM: GPT-4 and Llama 2
 - Interpreter: Plantweb/PlantUML
- Scenario (1/2):

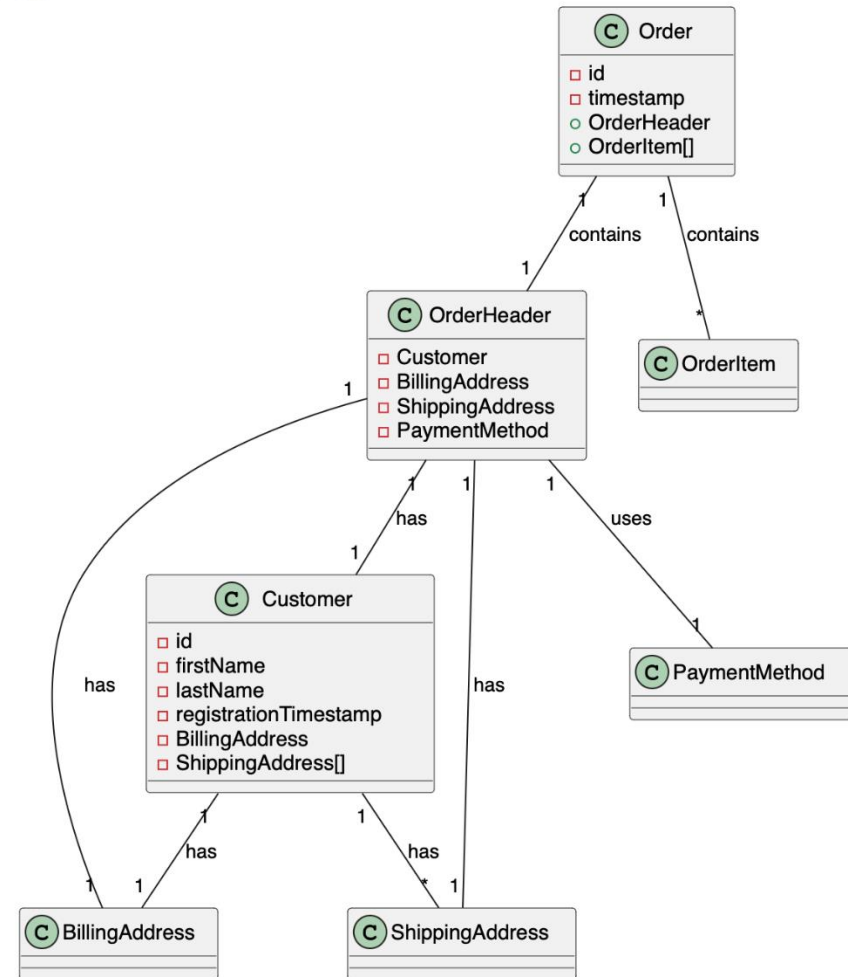


Create a class diagram in PlantUML code according to the following description. An order consists of an order header and one or more order items. Each order has an id and a timestamp. In an order header, a customer is referenced in addition to a billing address, a shipping address and a payment method. For the customer, an id as well as first name, last name and a registration timestamp are known in addition to a billing address and potentially multiple shipping addresses.

3. RESULTS – UML CONCEPTUALIZATION

- UML generation by GPT-4 (1/2)
 - Generation of syntactically correct PlantUML models
 - Classes, attributes, relationships with multiplicities
 - Overall accurate and appropriate for the scenario
 - Detailed Observations
 - Visibility inconsistent
 - Address classes not generalized
 - Capitalization inconsistent
 - Variability in results and design choices

Running interpreter: Plantweb/PlantUML ...



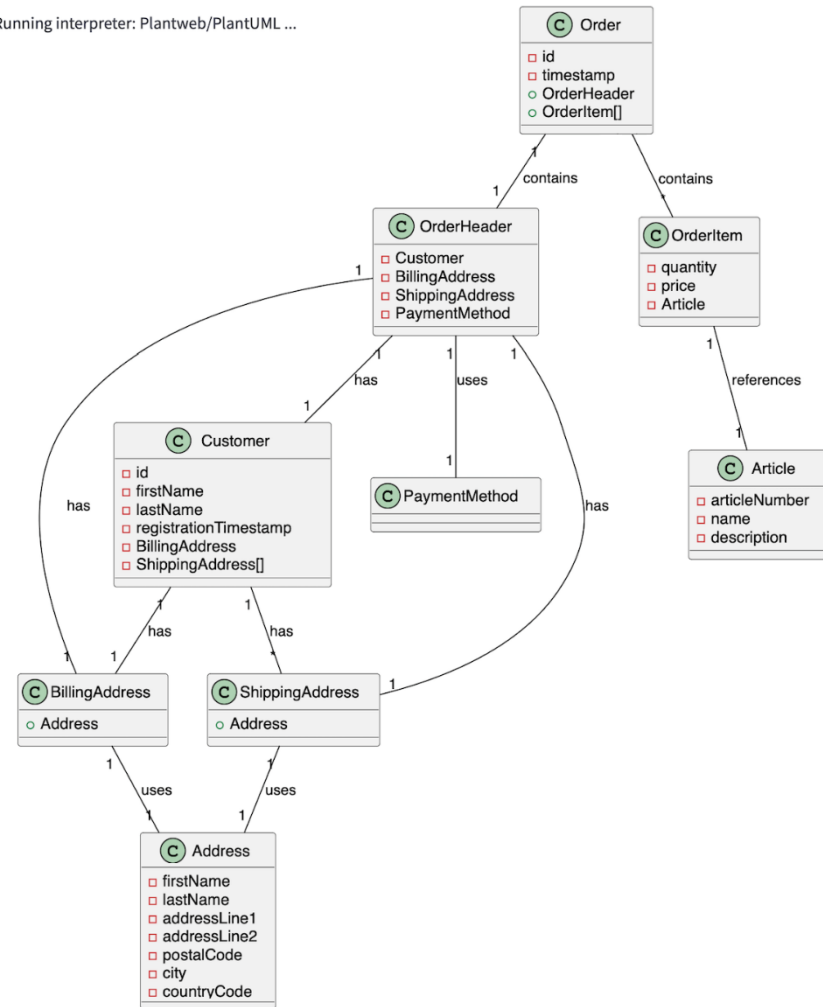
3. RESULTS – UML CONCEPTUALIZATION

UML generation by GPT-4 (2/2)

Update the diagram as follows: Addresses, both billing and shipping addresses, have a first name and last name, an address line 1 and 2, a postal code, a city and a country code. In an order item, there are a quantity, a price, and an article is referenced. Each article has an article number, a name and a description.

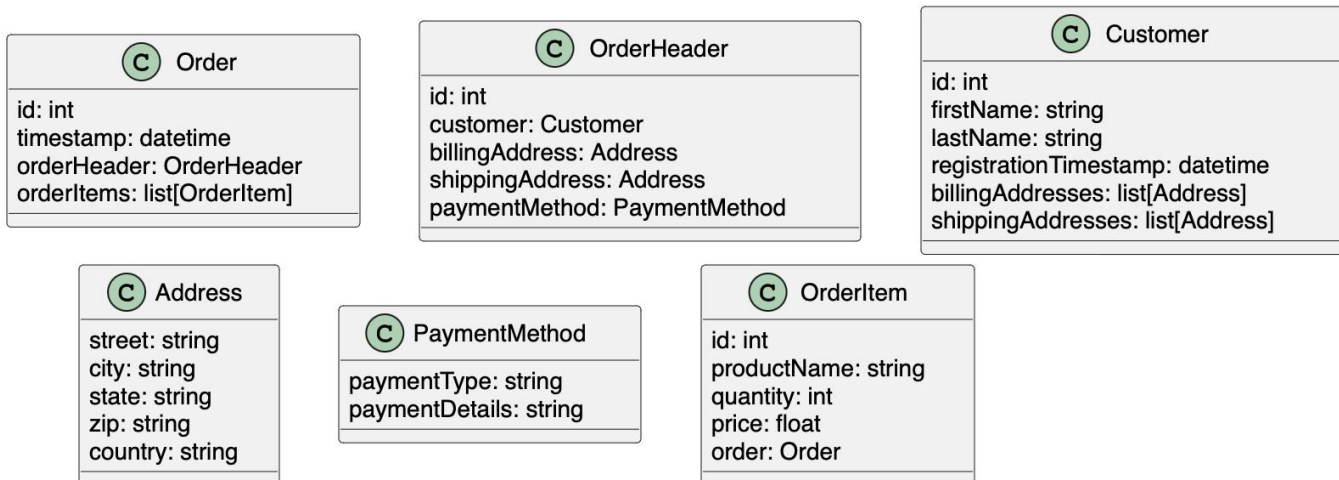
- New classes correctly added
- Attributes correctly introduced
- Introduction of an "Address" class with a composition pattern
 - Prevents the duplication of address attributes
 - Apparently favoring composition over inheritance

Running interpreter: Plantweb/PlantUML ...



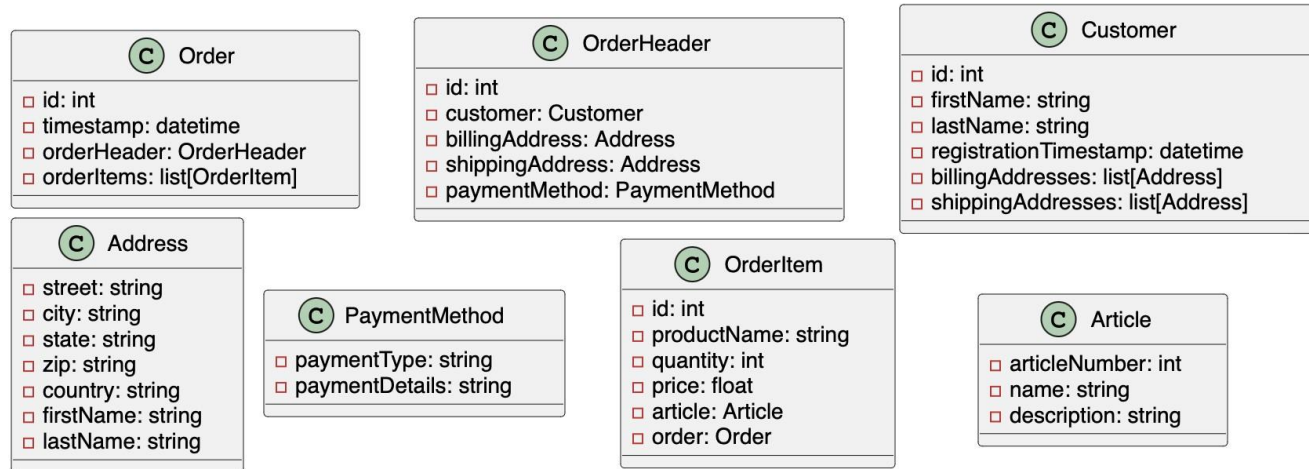
3. RESULTS – UML CONCEPTUALIZATION

- Conceptualization by Llama 2 (1/2):
 - Generation of syntactically correct PlantUML models without relationships
 - Classes and attributes with data types created
 - Requests to add relationships ignored or using incorrect syntax
 - Detailed Observations
 - Also no generalization of address classes, however, a single address class prevents duplicate attributes
 - Hallucinations of attributes



3. RESULTS – UML CONCEPTUALIZATION

- Conceptualization by Llama 2 (2/2):
 - New classes correctly added
 - Attributes introduced if not conflicting with previously hallucinated attributes



3. RESULTS – GRAPH GENERATION FROM DATA

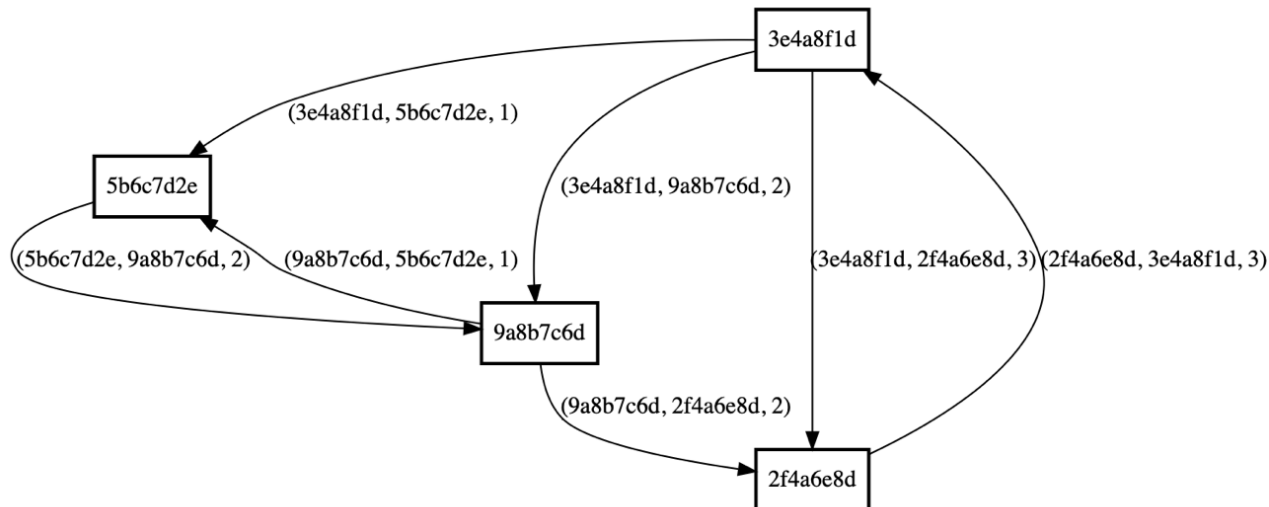
- Generation of graphs on the instance-level
- Scenario description containing exact data in a custom syntax and formatting instructions
- Generation of Graphviz source code using
 - LLM: GPT-4 and Llama 2
 - Interpreter: Graphviz
- Scenario:



Consider the following data stored in triples using the format (s,t,w) and create a directed graph in Graphviz syntax. For each triple, create a directed edge from node s to node t with weight w. Show nodes as rectangles with rounded corners and add labels numbering them N1, N2 and so on. Adjust the width of edges according to the weight. ("3e4a8f1d","5b6c7d2e",1), ("3e4a8f1d","9a8b7c6d",2), ("3e4a8f1d","2f4a6e8d",3), ("5b6c7d2e","9a8b7c6d",2), ("9a8b7c6d","5b6c7d2e",1), ("9a8b7c6d","2f4a6e8d",2), ("2f4a6e8d","3e4a8f1d",3)

3. RESULTS – GRAPH GENERATION FROM DATA

- Generation of graphs by Llama 2:
 - Generation of syntactically correct Graphviz code
 - Recognition of custom syntax
 - Correct graph, according to the data
 - Formatting without edge width and style attribute (corners)
 - Node names not numbered, superfluous data in edge labels
 - Variability regarding edge labels, no hallucinations



4. DISCUSSION AND OUTLOOK

- Application of LLMs for Conceptual Modeling using a conversational approach (Objective 1)
 - Language-based approach and comprehension of scenarios and data well suited for conceptual modeling tasks
 - Not only a new interface – natural language instructions and conceptual understanding
 - Main Challenges for LLMs, mainly Llama 2 and other open source LLMs:
 - High variability over prompts and responses
 - Parametrization
 - Unknown syntax
 - Comprehension of semantics
 - Hallucination

4. DISCUSSION AND OUTLOOK

- Construction of an architecture (Objective 2):
 - Architecture and prototype for multiple state-of-the-art LLMs and interpreters
 - Experimentation environment for LLM-related studies
 - Integration in applications, e.g., for the ad-hoc model generation
- Directions of future research
 - LLM capabilities are improving, further studies needed regarding parametrization, prompt engineering, training data sets, specialized models
 - Interpreters for modeling and domain-specific languages
 - Software integrations utilizing LLMs beyond textual instructions and interpreters beyond rendering
- Potential to lower the barrier for modeling and design activities

Thank you for your attention

felix.haerer@unifr.ch

unifr.ch/inf/digits