

# CMAG: A Framework for Conceptual Model Augmented Generative Artificial Intelligence

Hans-Georg Fill<sup>1</sup>, Felix Härer<sup>1,2</sup>, Iva Vasic<sup>1,3</sup>, Daniel Borcard<sup>1</sup>, Benedikt Reitemeyer<sup>1</sup>, Fabian Muff<sup>1</sup>, Simon Curty<sup>1</sup> and Marcel Bühlmann<sup>1</sup>

<sup>1</sup>University of Fribourg, Research Group Digitalization and Information Systems, 1700 Fribourg, Switzerland

<sup>2</sup>FHNW University of Applied Sciences and Arts, Digital Trust Competence Center, 4002 Basel, Switzerland

<sup>3</sup>Marche Polytechnic University, 60131 Ancona, Italy

## Abstract

The advent of generative artificial intelligence, and in particular large language models, has opened up new possibilities for information processing in a multitude of domains. Nevertheless, it is essential to validate their output in order to ensure its validity within the specified context. This is due to their nature as probabilistic models of language, which may lead to the generation of inaccuracies or non-existent facts commonly known as hallucinations. As a solution, we propose a framework and a prompt structure for the validation of the results of generative artificial intelligence in formats that are more human-comprehensible through the use of conceptual models. We denote this as conceptual model augmented generative artificial intelligence (CMAG). We illustrate the approach through application examples in the domains of data management, knowledge graphs and cultural heritage, and software engineering.

## Keywords

Generative Artificial Intelligence, Large Language Model, Conceptual Model, Prompt Structure

## 1. Introduction

Since the introduction of ChatGPT in November 2022, generative artificial intelligence and large language models (LLM) have caught enormous interest in many research fields. Ranging from such diverse disciplines as law, medicine, robotics, education, or computer science, many researchers started to evaluate how this technology can be successfully applied [1, 2, 3, 4, 5, 6, 7]. In contrast to previous approaches in machine learning, the fundamental new aspect of this technology is that it can be applied to a variety of tasks without having been explicitly trained for them. Rather, it has an unprecedented versatility that allows it to be used in many different areas. These models are therefore referred to as *foundation models* [8].

---


ER2024: Companion Proceedings of the 43rd International Conference on Conceptual Modeling: ER Forum, Special Topics, Posters and Demos, October 28-31, 2024, Pittsburgh, Pennsylvania, USA

✉ hans-georg.fill@unifr.ch (H. Fill); felix.haerer@unifr.ch (F. Härer); iva.vasic@unifr.ch (I. Vasic); daniel.borcard@unifr.ch (D. Borcard); benedikt.reitemeyer@unifr.ch (B. Reitemeyer); fabian.muff@unifr.ch (F. Muff); simon.curty@unifr.ch (S. Curty); marcel.buehlmann@unifr.ch (M. Bühlmann)

🌐 <https://www.unifr.ch/inf/digits> (H. Fill)

🆔 0000-0001-5076-5341 (H. Fill); 0000-0002-2768-2342 (F. Härer); 0000-0003-4081-0203 (I. Vasic); 0000-0001-7211-4793 (D. Borcard); 0009-0001-4131-1716 (B. Reitemeyer); 0000-0002-7283-6603 (F. Muff); 0000-0002-2868-9001 (S. Curty); 0009-0000-1886-207X (M. Bühlmann)

© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

Following the initial excitement about this new technology, it has however soon been observed that the interaction with such models requires some caution as they may produce false or unsubstantiated information. Although the output quality is constantly improving [9], the validation of the generated output is still a challenge [10]. This originates from the fact that these machine learning models may (i) generate output in a form that requires a lot of human effort to review, e.g. due to the length or complexity of the output, and (ii) may introduce errors such as hallucinations or confabulations in the output due to unreliable input sources or the stochastic nature of the models [11]. In addition to data enhancements during the training phase, the first issue can be resolved through the use of suitable prompting strategies that force machine learning models to generate specific output formats that can be more effortlessly validated by humans. Such prompts are patterns that the model takes as input instructions for generating the output. As demonstrated previously, the instructions provided can encompass the specification of novel languages that were not incorporated into the training of the model as well as instructions on which formats to output. This may include domain-specific languages or language profiles, as well as existing or emerging modeling languages [12].

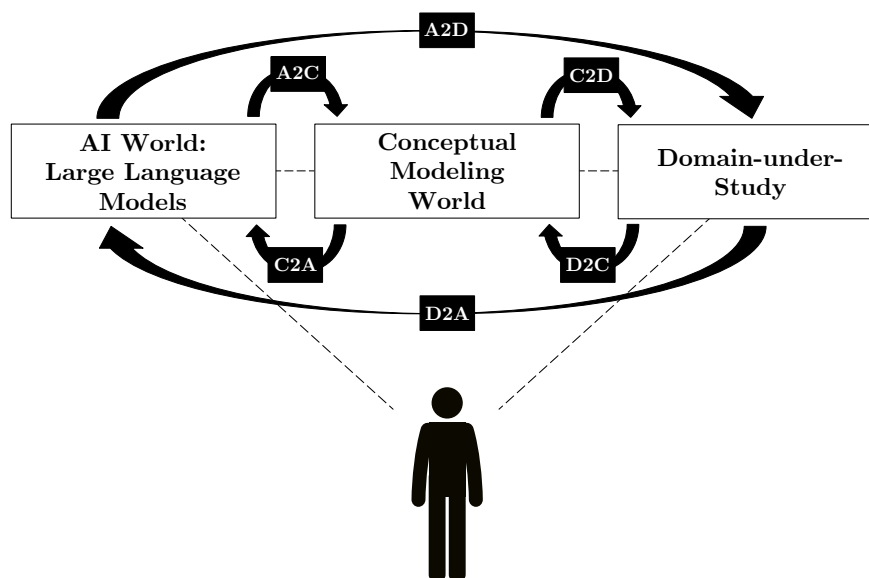
The choice of such languages depends on the availability of languages for the domain and use cases where a generative AI model is to be deployed. In our view, conceptual modeling languages constitute an optimal candidate for such formats given that they are, by their very nature, oriented towards human understanding and communication [13]. These characteristics render them ideal for the translation of information generated by an artificial intelligence, such as a large language model, into human-understandable structures that can be inspected based on pre-defined semantic concepts.

In the following we therefore propose a framework and a prompt structure for describing how to augment the output of generative AI models with conceptual models. In particular, we generalize previous proposals for generating conceptual models using large language models by introducing a prompt structure for forcing them to generate instances of conceptual models according to an existing or newly defined modeling language (metamodel) or even based on an explicitly specified meta<sup>2</sup> model. The goal is to arrive at representations in a format that can be more effortlessly inspected by humans than pure text-based output.

## 2. A Framework for Integrating Conceptual Models and Large Language Models

The framework we propose consists of three sections - see Figure 1: A. the *AI World* and in particular large language models, B. the *Conceptual Modeling World*, and C. the *Domain-under-Study*. In the traditional usage of large language models, they are employed for directly inferring information about some domain that a human individual is interested in. A domain may be for example a knowledge area, some technical system, or even code in a programming language. Therefore, information about the domain-under-study included in a prompt is sent to the LLM. This is expressed by the relation  $D2A$ . The output of the large language model then contains information on the system-under-study, which is denoted as  $A2D$ . When adding conceptual models, the interaction with the LLMs includes an intermediary step via the Conceptual Modeling World. Therefore, information about the domain-under-study is first being

translated into a conceptual model via the relation  $D2C$ . This represents the traditional activity in conceptual modeling for creating a model based on pre-defined concepts. This information is then used in a prompt directed at the LLM as denoted by the relation  $C2A$ . Based on this information the LLM generates an output that maps back to the Conceptual Modeling World, denoted as  $A2C$  where it can be interpreted by the human individual for inferring information about the domain-under-study in  $C2D$ .

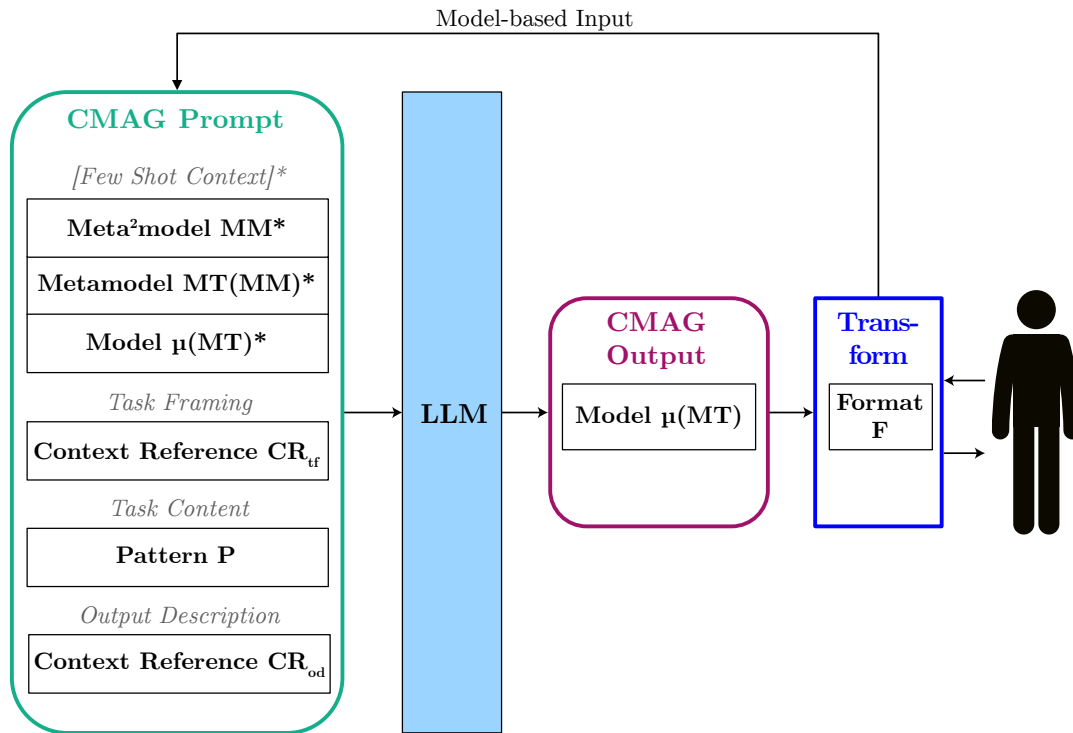


**Figure 1:** Proposed Framework for Conceptual Model Augmented Generative Artificial Intelligence (CMAG)

The main contribution of this framework is to establish conceptual models as the *central medium for interacting with generative artificial intelligence* and in particular large language models. Depending both on the data that the machine learning models have been trained on and the domain-under-study, it can be chosen from different modeling languages for this purpose, including the definition of new or emerging domain-specific modeling languages [14].

For the prompts for interacting with the generative artificial intelligence, we propose the structure shown in Figure 2. For the symbols used in the prompt we refer to a notation similar to  $FDMM^1$ , which has been introduced previously for the formal specification of modeling language constructs and includes concepts for meta models, model types, object and data types, as well as their instantiation in model instances [15, 16]. Thus, we use a classical three-tier modeling architecture with a meta<sup>2</sup> model, a metamodel, and model instances [17]. The structure consists of the following items: an optional specification of a *few shot context*, which can be supplied in the prompt to include information about the meta<sup>2</sup> model  $MM$ , the metamodel  $MT(MM)$  in the sense of  $FDMM$  model types as instantiated from the meta<sup>2</sup> model, and the model  $\mu(MT)$  as instantiated from the metamodel. Depending on the data used for the training of the generative AI, some or all of these items may be skipped. For example, as the LLM GPT 3.5/4 includes

<sup>1</sup> $FDMM$  stands for Formalism for Describing Metamodels and Models.



**Figure 2:** Proposed Architecture for Prompts based on CMAG - [\*] indicates optional parts

training data about the UML modeling language in the form of PlantUML code, it would not be necessary to include this information in the few-shot context [12, 18]. Similarly, if a meta<sup>2</sup> model is already known to an LLM, only this information can be skipped and metamodels can be directly specified.

The next part of the prompt structure is what we denote as *task framing*. Here, a reference to the previously established or known context is added to the prompt in the form of a context reference  $\mathbf{CR}_{tf}$ . It instructs the LLM to use the information from the few-shot context or of a learned context, i.e. a modeling language, for the following task. The *task content* then specifies the concrete task to be addressed by the LLM in the form of a pattern  $\mathbf{P}$ . Finally, an *output description* is added, which contains a second reference to the context  $\mathbf{CR}_{od}$  to force the LLM to create the result in the format specified by the context, e.g. as a model instance in the previously taught format.

In Prompt 1, an example for this prompt structure is shown, which has been tested using GPT-4. It contains a meta<sup>2</sup> model, a metamodel, and a model instance for the few shot context, a task framing with a reference to this context, a task content for creating a hiring process and an output description with another reference to the context.

**Prompt 1**

```
Assume the following meta2model: Class (x), Role (y), Relation (z) := (Role(a), Role(b))
Assume the following metamodel: Class (Task), Class (XOR), Role (from_sub) = (Task, XOR),
Role (to_sub) = (Task, XOR), Relation (subsequent) := (from_sub, to_sub)
Assume the following sample model: Task(Write letter), Task(Send letter) Task(Scan letter),
XOR (Medium?), from_sub(r1) = (Write letter), to_sub(r2) = (Medium?), from_sub(r3) = (Medium?),
to_sub(r4) = (Send letter), from_sub(r5) = (Medium?), to_sub(r6) = (Scan letter),
subsequent(r1,r2), subsequent(r3,r4), subsequent(r5,r6)
```

Create a model according to this metamodel for the following:

Make a proposal for a process for hiring an employee in a company.

Output the result as in the notation shown before.

Only show the code surrounded with triple backticks.

After sending a prompt according to this structure to an LLM, the CMAG output is received in the form of a model instance  $\mu(\mathbf{MT})$ , which can subsequently be processed, e.g. by a modeling tool to transform it into a target format  $\mathbf{F}$ , which is adequate for a user to understand, e.g. in the form of a visual diagram. Similarly, the user may specify or modify input in this format, e.g. by using a modeling tool implementing the metamodel, which is then transformed back into the CMAG prompt structure and sent back to the LLM for further conversations.

### 3. Application Examples

In the following we illustrate the usage of the proposed framework with further examples. These stem from the domain of data management, knowledge graphs and the application for cultural heritage, and software engineering.

#### 3.1. Data Management

Data management includes today the interaction with data in many different formats, ranging from databases, documents, and data stores using various data structures such as graphs. Generative AI may help in this domain for designing schemas, for creating instance data, or formulating queries in order to ease a multitude of tasks. Conceptual models may therefore act as an intermediary representation, in order to support tasks in data management where the accuracy of artificially generated structures is of utmost importance.

First applications of the framework utilized ChatGPT 4 with Entity Relationship (ER) models in order to generate model instances  $\mu(\mathbf{ER})$  of an ER metamodel [12]. Task Framing  $\mathbf{CR}_{\text{tf}}$  was provided by a natural-language explanation of the ER syntax, including elements, relationships, cardinalities, and basic syntax rules. The prompt then continued to describe the output  $\mathbf{CR}_{\text{od}}$  using a JSON syntax, where the root JSON object `ERdiagram` hosts lists of the contained entity and relationship types, attributes, and their relationships with the instruction to only output in this format without further explanation. Lastly, the prompt conveyed the task content  $\mathbf{P}$ , a description in terms of the domain in the style of textbook modeling tasks. Using this approach,  $\mu(\mathbf{ER})$  in JSON could be successfully obtained. While these first applications were generally

appropriate in terms of syntax and semantics, further evaluations of complex scenarios will follow towards enhancements involving few-shot contexts.

On its own, the principle of including few-shot model context has been shown already. By using  $\mu(\mathbf{ER})$  as a subsequent input, the output was a correct interpretation of the model by ChatGPT in natural language [12] and could be refined in multiple iterations [19]. In a second application, domain models in the form of UML class diagrams with PlantUML syntax have been generated [12, 19]. Here,  $\mathbf{CR}_{tf}$  did not require an explicit syntax description due to the inherent knowledge of PlantUML in the GPT 4 foundation model. In this case, the  $\mathbf{CR}_{od}$  merely instructs GPT to provide an output in PlantUML from textbook-style scenario descriptions  $\mathbf{P}$ . Generally, the main syntax of UML class diagrams could be generated with appropriate semantics using this approach, current limitations of recognizing subtle details and hallucination notwithstanding [12, 19].

In the future, similar approaches can be envisaged for further tasks in data management, e.g. for validating queries by using a visual conceptual modeling [20, 21] or for transformations between different modeling languages using visual triple graph grammars [22], just to name a few.

### 3.2. Knowledge Graphs and the Domain of Cultural Heritage

Another example is the application of the framework in the domain of knowledge graphs. It has been shown previously that LLMs can help in knowledge graph construction, as presented by Chen et al. in their AutoKG approach [23]. Knowledge graphs also constitute conceptual models in the sense that there is a dedicated meta<sup>2</sup> model, a metamodel and corresponding instances, e.g. when using the web ontology language (OWL) or RDF as a base format.

For example, in the cultural heritage field, structuring knowledge is important for organizing and preserving information. However, traditional processes, which mostly involve manual semantic annotation of data, are time-consuming and require domain-specific expertise. Consequently, LLMs can significantly assist in such a process due to their advanced generalization capabilities and capacity to perform only few-shot to zero-shot learning. The output should be validated by experts with proficiency in both domain ontology and the specific area of cultural heritage in question. Alternatively, approaches like Question-Answering can facilitate information retrieval from the constructed Knowledge Graphs through natural language input [24], offering a solution for CH scholars who may not have experience with complex ontology models.

For this domain, we previously derived examples of semantic annotations using LLMs based on an ontology. We chose the CIDOC Conceptual Reference Model (CRM) as it is a widely accepted metamodel for documenting cultural heritage artifacts and ISO standard (ISO 21127:2014) [25] and was seemingly contained in training data for LLMs such as GPT 3.5 and 4. Therefore, no particular few shot context was necessary. More specifically, an LLM was employed for constructing formal triple statements in the form of subject-predicate-object from the natural language used for painting descriptions. As shown in Prompt 2, the context reference  $\mathbf{CR}_{tf}$  is set to the CIDOC-CRM ontology. The pattern of the task content  $\mathbf{P}$  includes a painting description which was retrieved from the Getty Museum Collection<sup>2</sup>. The context reference

<sup>2</sup><https://www.getty.edu/art/collection/object/10986B>, last accessed: 25.07.2024

$\mathbf{CR}_{od}$  in the output description then refers to the triple statement structure for CIDOC-CRM ontologies. The output are then semantic annotations based on the CIDOC-CRM entities, i.e.  $\mu(\mathbf{CIDOC} - \mathbf{CRM})$ .

### Prompt 2

You are an expert in cultural heritage, you know CIDOC-CRM ontology and you will have to construct formal triple statements using only CIDOC-CRM classes and properties based on the painting description.

Painting description: '{user\_input}'.

Your response must be in the form: Domain class (value) - property - Range class (value). Values can also be the classes and not the literals. Do not explain anything else but construct the complete graph of a minimum of 10 different statements.

For a painting "Study of Clouds with a Sunset near Rome" with the description such as the following: "...Dark, swirling clouds loom over a narrow strip of land, gently punctuated by far-off trees and a city skyline. The place is Rome. The weather: an impending rainstorm. The day: a late afternoon within the last decades of the 1700s. This painting is one in a series of cloud studies—in fact the forty-eighth—that Simon Denis made to hone his observational and painting skills...", the LLM would then output the Prompt Result 1. This output can be easily checked by experts knowledgeable in CIDOC to verify whether the LLM has worked correctly.

### Prompt Result 1 (Source: *generated by GPT-4*)

E22\_Man-Made\_Object(Study of Clouds with a Sunset near Rome) - P102(has title) -  
E35\_Title(Study of Clouds with a Sunset near Rome)

E22\_Man-Made\_Object(Study of Clouds with a Sunset near Rome) - P108i(was produced by) -  
E12\_Production(Production of Study of Clouds with a Sunset near Rome)

E12\_Production(Production of Study of Clouds with a Sunset near Rome) - P14(carried out by) -  
E21\_Person(Simon Alexandre Clément Denis)

E21\_Person(Simon Alexandre Clément Denis) - P131(is identified by) -  
E82\_Actor\_Appellation(Simon Alexandre Clément Denis)

E12\_Production(Production of Study of Clouds with a Sunset near Rome) - P4(has time-span) -  
E52\_Time-Span(1786-1801)

E22\_Man-Made\_Object(Study of Clouds with a Sunset near Rome) - P45(consists of) -  
E57\_Material(Oil)

E22\_Man-Made\_Object(Study of Clouds with a Sunset near Rome) - P2(has type) -  
E55\_Type(Painting)

E22\_Man-Made\_Object(Study of Clouds with a Sunset near Rome) - P43(has dimension) -  
E54\_Dimension(Unframed: 33.8 × 39.4 cm)

E22\_Man-Made\_Object(Study of Clouds with a Sunset near Rome) - P43(has dimension) -  
E54\_Dimension(Framed: 41.4 × 49.5 × 2.5 cm)

E22\_Man-Made\_Object(Study of Clouds with a Sunset near Rome) - P55(has current location) -  
E53\_Place(J. Paul Getty Museum)

E22\_Man-Made\_Object(Study of Clouds with a Sunset near Rome) - P50(has current keeper) -  
E39\_Actor(J. Paul Getty Museum)

```
E22_Man-Made_Object(Study of Clouds with a Sunset near Rome) - P128(carries) -
E33_Linguistic_Object(Signature: "a Rome / S.n D.s / 48")

E22_Man-Made_Object(Study of Clouds with a Sunset near Rome) - P1(is identified by) -
E42_Identifier(Object Number: 2005.31)
```

The use of large language models in the domain of knowledge graphs and ontologies is currently intensively explored. Besides traditional tasks such as ontology engineering and querying [26] or semantic alignment [27], conceptual model augmented generative artificial intelligence may facilitate more effortless interaction with formal semantic schemata through their capability to translate between formal and less formal representations on the basis of languages as it has been known for a long time [28].

### 3.3. Software Engineering

In the domain of software engineering, various types of conceptual models are being used, with UML models being the most prominent ones [18]. Another type of models are C4 diagrams, which are used in industry to model IT architectures [29]. C4 diagrams are comprised of four principal levels, which collectively represent the entirety of the system under consideration, from the macro level down to the level of the implemented code. In the process of attempting to resolve software architecture issues through the use of LLM technologies, C4 diagrams may prove beneficial in evaluating the recommendations proposed by the AI. Integrating new functionalities within a complex system can prove challenging, particularly when there are numerous existing features that must be considered. The use of LLM enables the generation of diagrams that can serve as a reflection basis. The domain-specific language PlantUML can be utilized for the code-based specification of C4 diagrams as it can be extended with libraries to enhance its flexibility. Moreover, the employment of a language to generate diagrams from code offers the benefit of direct accessibility by current LLMs, which have already incorporated its constructs into their training data.

The approach we used to interact with  $\mu(\mathbf{C4})$  diagrams using LLMs according to the CMAG structure is comprised of three phases:

1. **Prompting:** Prompt the LLM to generate the PlantUML code using the CMAG prompt structure.
2. **Transformation:** Interpret the CMAG output using a PlantUML engine and export the generated diagram in the desired format (PNG, PDF, SVG, etc).
3. **Interpretation:** Analysis of the visual diagrams by senior software architects.

The prompt begins with the instructions on the general context,  $\mathbf{CR}_{\text{tf}}$  and includes instructions about the general goal of the prompt, the goal of the answer and the structure of the prompt. This section includes a short description about the desired output  $\mathbf{CR}_{\text{od}}$ . The structure of the prompt defines under which tags the sample meta<sup>2</sup> model  $\mathbf{MM}$ , the metamodel  $\mathbf{MT}(\mathbf{MM})$ , and the model  $\mu(\mathbf{MT})$  could be found. In our example we included a description of the  $\mathbf{MT}(\mathbf{C4})$  and examples of the  $\mu(\mathbf{C4})$  diagrams under a *Documentation* tag. The *Requirements* tag contains the specific goal of the current prompt ( $\mathbf{P}$ ) and can be of various forms (e.g. a feature definition, a user query, etc). The structure of the requirements can be defined in the



general context ( $\mathbf{CR}_{tf}$ ) for potentially better results. The last section, namely *Previous diagrams* consist of  $\mu(\mathbf{C4})$  diagrams that were previously generated mainly to give a broader context and keep the same direction when generating other diagram levels about the same feature. An example of such a prompt to generate a C2 level diagram is shown in prompt 3 and the result in Figure 3. This example aims to demonstrate the generation of a container diagram for a search hint suggestion feature in a web context.

### Prompt 3

Goal:

Your task is to create a PlantUML code snippet for a C4 model diagram at the C2: Container diagram level. The diagram should help an architect develop models and define requirements for new features in a software project. Instructions:

- Ensure the PlantUML code is well-structured and annotated for clarity.
- The code should be easily adaptable for specific projects and features.
- The documentation of the library used to create C4 diagrams is provided under "Documentation".
- Feature requirements are listed after "Requirements Definition".
- Reference to previous C4 diagrams is provided under "Previous Diagrams".

Documentation:

# *C4-PlantUML*

C4-PlantUML combines the benefits of PlantUML and the C4 model to describe and communicate software architectures, especially during up-front design sessions. It uses an intuitive language and is open source and platform independent.

Requirements Definition:

*Functional Requirements*

- **Accordion Navigation Structure:**  
Users should be able to navigate through meta-objects using an accordion interface with expandable and collapsible categories. This allows for efficient access to different elements of the conceptual model. Clicking on a main category expands it to reveal subcategories or meta-objects, and clicking again collapses it. The hierarchy should reflect actual relationships between meta-objects.
- **Real-Time Updates:**  
The navigator should automatically refresh to reflect any changes made to meta-objects elsewhere in the application, ensuring displayed information is always up-to-date.
- **Search Functionality:**  
Users should be able to search for specific meta-objects within the navigator, with instant results highlighting matching objects as they type.
- **Selection and Highlighting:**  
When a meta-object is selected from the navigator, it should be visually highlighted, and the navigator should display contextual details or options related to the selected meta-object.

Previous Diagrams: *C1: Context Diagram*

@startuml

!include [https://raw.githubusercontent.com/plantuml-stdlib/C4-PlantUML/master/C4\\_Context.puml](https://raw.githubusercontent.com/plantuml-stdlib/C4-PlantUML/master/C4_Context.puml)

' Define the actors in the system

Person(admin, "Administrator", "Manages the system and meta-objects")

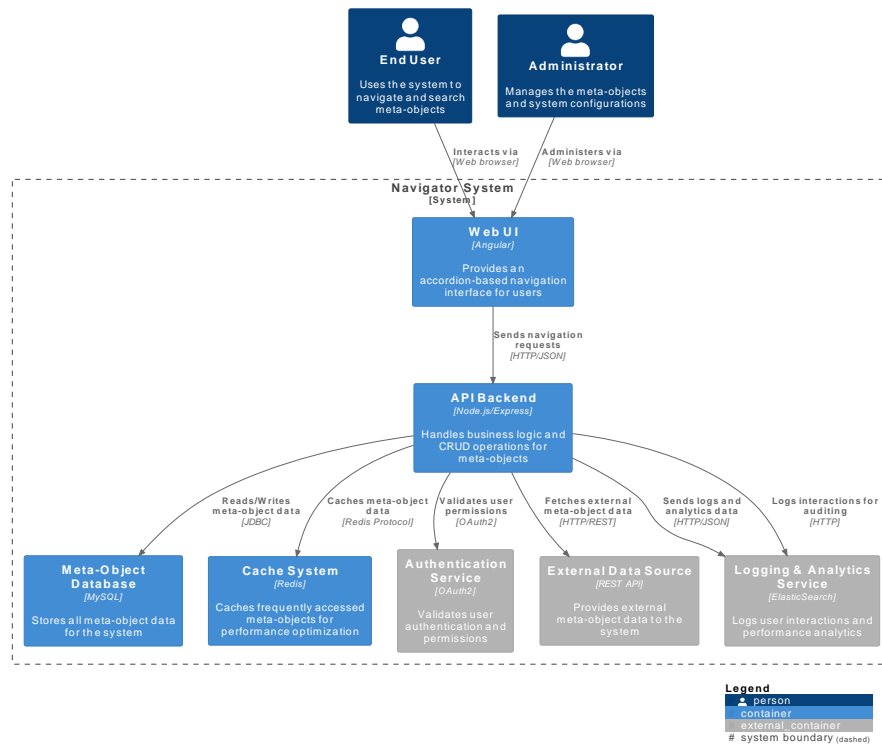
Person(user, "End User", "Accesses the accordion interface for navigation")

Person(api\_user, "API User", "Interacts with the system via API")

' Additional diagram elements

...

@enduml



**Figure 3:** Example for a C2 Diagram generated with the CMAG structure - based on PlantUML code generated with GPT-4

## 4. Related Work

The framework we presented in this paper can be seen as an operationalization of the one proposed by Maass and Storey [30]. In their paper they pair conceptual modeling with machine learning and similarly position conceptual models as a central entity for mediating between mental models, the domain, data, and machine learning models. Despite the fact that the paper appeared before the release of ChatGPT, they already envision how generative machine learning models could be considered in their framework.

In another approach by Maass et al. it is discussed how information from machine learning can be inserted into conceptual models in order to enhance the explainability of AI models [31]. Although we do not specifically focus on the explainability of the inner workings of machine learning models, our framework can be used in a similar fashion by enabling humans to better understand the output of machine learning models through mapping the results to the human sphere and thus also contributing to the better understanding of machine learning.

Reitemeyer and Fill [32] examined the ability of ChatGPT to select appropriate modeling elements from the ArchiMate modeling language strategy layer for the instantiation of a specific domain concept. The prompt incorporates both ArchiMate elements and the domain concept as knowledge graphs. The results demonstrate that ChatGPT is generally capable of selecting suitable modeling elements and specifying a natural language-based probability (such as "very

high") for instantiation. Given that the input is knowledge graph-based and the output is in JSON format, the results showed that ChatGPT may be well-suited for dedicated tasks in automatic generation of conceptual enterprise models.

Explorative research on a software tool combining LLMs and conceptual modeling demonstrated a conceptual model interpreter [19]. With a conversational user interface, the proposed architecture shows the feasibility of modeling in a conversational fashion through (1) syntax generation for one or more LLMs, (2) interpretation of the syntax in rendered diagrams, and (3) subsequent refinement in multiple iterations. In addition, the evaluation of multiple LLMs could be shown with the tool and, in first experiments, indicated correct understanding of modeling task semantics and syntactically correct diagrams for basic modeling tasks with ChatGPT 4. Limitations concerned variance in answers over time, parametrization, and hallucination, especially in Llama 2 when compared to ChatGPT 4. This paper showed that also open source LLMs may be used in the context of conceptual model augmented generative AI, which is equally covered by our proposed framework.

Muff and Fill [33] demonstrated the limitations of using ChatGPT for conceptual modeling, and more specifically metamodeling, by performing various experiments on interpreting metamodels and model instances based on complex JSON data, e.g. comprising only approx. 700KB of data<sup>3</sup>. This showed that LLMs are currently not capable of processing very complex metamodel structures. This currently presents a limitation for the proposed framework, which may however be overcome in the future with the further advancement of machine learning models and increasing input token sizes.

## 5. Conclusion and Outlook

In this paper we presented a framework for using conceptual models as the central medium for interacting with generative artificial intelligence. By reverting to existing or emerging conceptual modeling languages, large language models can thus be forced to generate output which corresponds to conceptual schemata. This enables humans to more effortlessly validate the output of such machine learning models. This will have to be further validated in experiments with LLM outputs based on CMAG and standard textual output formats. For illustrating the application, we showed examples from the area of data management, knowledge graphs and the domain of cultural heritage, and software engineering. Future work will include the further formalization of the framework and its application to additional domains in conceptual modeling. In addition, it is planned to conduct user studies for quantifying the benefits of conceptual model augmented generative AI outputs in comparison to traditional ones in the form of text or code.

## Acknowledgement

Financial support is gratefully acknowledged by the Smart Living Lab funded by the University of Fribourg, EPFL, and HEIA-FR.

---

<sup>3</sup>For details see the original data set used for testing and the results at <https://zenodo.org/records/10695823>.

## References

- [1] R. Buchmann, J. Eder, H.-G. Fill, U. Frank, D. Karagiannis, E. Laurenzi, J. Mylopoulos, D. Plexousakis, M. Y. Santos, Large language models: Expectations for semantics-driven systems engineering, *Data & Knowledge Engineering* 152 (2024) 102324. doi:<https://doi.org/10.1016/j.datak.2024.102324>.
- [2] H. Fill, J. Cabot, W. Maass, M. van Sinderen, Ai-driven software engineering - the role of conceptual modeling, *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.* 19 (2024). doi:[10.18417/EMISA.19.1](https://doi.org/10.18417/EMISA.19.1).
- [3] X. Kang, L. Qu, L. Soon, A. Trakic, T. Y. Zhuo, P. C. Emerton, G. Grant, Can chatgpt perform reasoning using the IRAC method in analyzing legal scenarios like a lawyer?, in: H. Bouamor, J. Pino, K. Bali (Eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, Singapore, December 6-10, 2023, Association for Computational Linguistics, 2023, pp. 13900–13923. doi:[10.18653/v1/2023.FINDINGS-EMNLP.929](https://doi.org/10.18653/v1/2023.FINDINGS-EMNLP.929).
- [4] S. Tripathi, R. Sukumaran, T. S. Cook, Efficient healthcare with large language models: optimizing clinical workflow and enhancing patient care, *J. Am. Medical Informatics Assoc.* 31 (2024) 1436–1440. doi:[10.1093/JAMIA/OCAD258](https://doi.org/10.1093/JAMIA/OCAD258).
- [5] S. Vemprala, R. Bonatti, A. Bucker, A. Kapoor, Chatgpt for robotics: Design principles and model abilities, *IEEE Access* 12 (2024) 55682–55696. doi:[10.1109/ACCESS.2024.3387941](https://doi.org/10.1109/ACCESS.2024.3387941).
- [6] M. Wieser, K. Schöffmann, D. Stefanics, A. Bollin, S. Pasterk, Investigating the role of chatgpt in supporting text-based programming education for students and teachers, in: J. Pellet, G. Parriaux (Eds.), *ISSEP 2023*, Lausanne, Switzerland, October 23-25, 2023, *Proceedings*, volume 14296, Springer, 2023, pp. 40–53. doi:[10.1007/978-3-031-44900-0\\_4](https://doi.org/10.1007/978-3-031-44900-0_4).
- [7] I. Vasic, H. Fill, R. Quattrini, R. Pierdicca, LLM-aided museum guide: Personalized tours based on user preferences, in: L. T. De Paolis, P. Arpaia, M. Sacco (Eds.), *Extended Reality*, Springer Nature Switzerland, Cham, 2024, pp. 249–262. doi:[10.1007/978-3-031-71710-9\\_18](https://doi.org/10.1007/978-3-031-71710-9_18).
- [8] J. Schneider, C. Meske, P. M. Kuss, Foundation models, *Bus. Inf. Syst. Eng.* 66 (2024) 221–231. doi:[10.1007/S12599-024-00851-0](https://doi.org/10.1007/S12599-024-00851-0).
- [9] M. A. K. Raiaan, M. S. H. Mukta, K. Fatema, N. M. Fahad, S. Sakib, M. M. J. Mim, J. Ahmad, M. E. Ali, S. Azam, A review on large language models: Architectures, applications, taxonomies, open issues and challenges, *IEEE Access* 12 (2024) 26839–26874. doi:[10.1109/ACCESS.2024.3365742](https://doi.org/10.1109/ACCESS.2024.3365742).
- [10] S. Farquhar, J. Kossen, L. Kuhn, Y. Gal, Detecting hallucinations in large language models using semantic entropy, *Nat.* 630 (2024) 625–630. doi:[10.1038/S41586-024-07421-0](https://doi.org/10.1038/S41586-024-07421-0).
- [11] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, A. Madotto, P. Fung, Survey of hallucination in natural language generation, *ACM Comput. Surv.* 55 (2023) 248:1–248:38. doi:[10.1145/3571730](https://doi.org/10.1145/3571730).
- [12] H. Fill, P. Fettke, J. Köpke, Conceptual modeling and large language models: Impressions from first experiments with chatgpt, *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.* 18 (2023) 3. doi:[10.18417/EMISA.18.3](https://doi.org/10.18417/EMISA.18.3).
- [13] J. Mylopoulos, A. Borgida, M. Jarke, M. Koubarakis, Telos: Representing knowledge about information systems, *ACM Trans. Inf. Syst.* 8 (1990) 325–362. doi:[10.1145/102675](https://doi.org/10.1145/102675).

- 102676.
- [14] F. Härer, H. Fill, Past trends and future prospects in conceptual modeling - A bibliometric analysis, in: G. Dobbie, U. Frank, G. Kappel, S. W. Liddle, H. C. Mayr (Eds.), *Conceptual Modeling - 39th International Conference, ER 2020, Vienna, Austria, November 3-6, 2020, Proceedings*, volume 12400 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 34–47. doi:10.1007/978-3-030-62522-1\_3.
  - [15] H. Fill, T. Redmond, D. Karagiannis, Formalizing meta models with FDMM: the ADOxx Case, in: J. Cordeiro, L. A. Maciaszek, J. Filipe (Eds.), *Enterprise Information Systems - 14th International Conference, ICEIS 2012, Wroclaw, Poland, June 28 - July 1, 2012, Revised Selected Papers*, volume 141, Springer, 2012, pp. 429–451. doi:10.1007/978-3-642-40654-6\_26.
  - [16] H. Fill, T. Redmond, D. Karagiannis, FDMM: A Formalism for Describing ADOxx Meta Models and Models, in: L. A. Maciaszek, A. Cuzzocrea, J. Cordeiro (Eds.), *ICEIS 2012 - Proceedings of the 14th International Conference on Enterprise Information Systems, Volume 3, Wroclaw, Poland, 28 June - 1 July, 2012*, SciTePress, 2012, pp. 133–144.
  - [17] H. Fill, D. Karagiannis, On the conceptualisation of modelling methods using the adoxx meta modelling platform, *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.* 8 (2013) 4–25. doi:10.18417/EMISA.8.1.1.
  - [18] J. Cámara, J. Troya, L. Burgueño, A. Vallecillo, On the assessment of generative AI in modeling tasks: an experience report with chatgpt and UML, *Softw. Syst. Model.* 22 (2023) 781–793. doi:10.1007/S10270-023-01105-5.
  - [19] F. Härer, Conceptual model interpreter for large language models, in: *Companion Proceedings of the 42nd International Conference on Conceptual Modeling: ER Forum, 7th SCME, Project Exhibitions, Posters and Demos, and Doctoral Consortium co-located with ER 2023, Lisbon, Portugal, November 06-09, 2023*, volume 3618 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2023. URL: [https://ceur-ws.org/Vol-3618/forum\\_paper\\_11.pdf](https://ceur-ws.org/Vol-3618/forum_paper_11.pdf).
  - [20] H. Jaakkola, B. Thalheim, Visual SQL ? high-quality er-based query treatment, in: M. A. Jeusfeld, O. Pastor (Eds.), *Conceptual Modeling for Novel Application Domains, ER 2003 Workshops ECOMO, IWCMQ, AOIS, and XSDM, Chicago, IL, USA, October 13, 2003, Proceedings*, volume 2814 of *Lecture Notes in Computer Science*, Springer, 2003, pp. 129–139. doi:10.1007/978-3-540-39597-3\_13.
  - [21] Q. Fu, X. Wang, Y. Li, A general-purpose visual query language for knowledge graphs with bidirectional transformations, in: K. L. Taylor, R. S. Gonçalves, F. Lécué, J. Yan (Eds.), *Proceedings of the ISWC 2020 Demos and Industry Tracks*, volume 2721, CEUR-WS.org, 2020, pp. 138–143.
  - [22] C. Ermel, F. Hermann, J. Gall, D. Binanzer, Visual modeling and analysis of EMF model transformations based on triple graph grammars, *Electron. Commun. Eur. Assoc. Softw. Sci. Technol.* 54 (2012). doi:10.14279/TUJ.ECEASST.54.771.
  - [23] B. Chen, A. L. Bertozzi, AutoKG: Efficient automated knowledge graph generation for language models, in: *2023 IEEE International Conference on Big Data (BigData)*, 2023, pp. 3117–3126. doi:10.1109/BigData59044.2023.10386454.
  - [24] N. Gounakis, M. Mountantonakis, Y. Tzitzikas, Evaluating a radius-based pipeline for question answering over cultural (CIDOC-CRM based) knowledge graphs, in: *Proceedings of the 34th ACM Conference on Hypertext and Social Media, HT '23*, Association for

- Computing Machinery, New York, NY, USA, 2023. URL: <https://doi.org/10.1145/3603163.3609067>. doi:10.1145/3603163.3609067.
- [25] C. Meghini, M. Doerr, A first-order logic expression of the CIDOC conceptual reference model, *International Journal of Metadata, Semantics and Ontologies* 13 (2018) 131. doi:10.1504/IJMSO.2018.098393.
- [26] M. J. Saeedizade, E. Blomqvist, Navigating ontology development with large language models, in: A. Meroño-Peñuela, A. Dimou, R. Troncy, O. Hartig, M. Acosta, M. Alam, H. Paulheim, P. Lisena (Eds.), *The Semantic Web - 21st International Conference, ESWC 2024*, volume 14664 of *Lecture Notes in Computer Science*, Springer, 2024, pp. 143–161. doi:10.1007/978-3-031-60626-7\_8.
- [27] R. Amini, S. S. Norouzi, P. Hitzler, R. Amini, Towards complex ontology alignment using large language models, *CoRR abs/2404.10329* (2024). doi:10.48550/ARXIV.2404.10329. arXiv:2404.10329.
- [28] M. A. Musen, Conceptual models of interactive knowledge acquisition tools, *Knowledge Acquisition* 1 (1989) 73–88.
- [29] S. Brown, *The C4 Model for Visualising Software Architecture*, Leanpub.com, 2023. URL: <https://samples.leanpub.com/visualising-software-architecture-sample.pdf>, Last accessed: 2024-10-01.
- [30] W. Maass, V. C. Storey, Pairing conceptual modeling with machine learning, *Data Knowl. Eng.* 134 (2021) 101909. doi:10.1016/J.DATK.2021.101909.
- [31] W. Maass, A. Castellanos, M. C. Tremblay, R. Lukyanenko, V. C. Storey, AI explainability: Embedding conceptual models, in: *ICIS 2022*, Association for Information Systems, 2022.
- [32] B. Reitemeyer, H. Fill, Leveraging llms in semantic mapping for knowledge graph-based automated enterprise model generation, in: H. Giese, K. Rosenthal (Eds.), *Modellierung 2024 - Workshop Proceedings*, Potsdam, Germany, March 12-15, 2024, Gesellschaft für Informatik e.V., 2024, p. 8. doi:10.18420/MODELLIERUNG2024-WS-008.
- [33] F. Muff, H. Fill, Limitations of chatgpt in conceptual modeling: Insights from experiments in metamodeling, in: H. Giese, K. Rosenthal (Eds.), *Modellierung 2024 - Workshop Proceedings*, Potsdam, Germany, March 12-15, 2024, Gesellschaft für Informatik e.V., 2024, p. 8. doi:10.18420/MODELLIERUNG2024-WS-008.