Towards Satisfaction Analysis of GRL Model Images Using Vision-Enabled LLMs

Ahmed Hassine
Department of Computer Science
Carleton University
Ottawa, Canada
ahmedhassine@cmail.carleton.ca,

Jameleddine Hassine
Department of Computer Science
Université du Québec à Montréal (UQAM)
Montréal, Canada
hassine.jameleddine@uqam.ca,

***Abstract*—Goal models, such as i*, KAOS, and GRL, are central artifacts in requirements engineering, capturing stakeholder intentions, dependencies, and system objectives. However, querying and reasoning over these models traditionally require specialized expertise and dedicated analysis tools. Recent advances in vision-enabled large language models (LLMs) create new opportunities for performing model analysis directly from graphical representations. In this paper, we propose a novel vision-based approach that leverages GPT-5.5 to perform GRL satisfaction analysis directly from diagram images through an end-to-end pipeline that automatically extracts an intermediate structured representation before performing quantitative satisfaction analysis. The approach is evaluated through four research questions addressing (1) identification of GRL constructs from images, (2) computation of satisfaction values through quantitative propagation, (3) identification of satisfaction-maximizing strategies, and (4) identification of threshold-satisfying strategies. Experiments were conducted on three GRL models of varying complexity and multiple image variants. The results indicate that GPT-5.5 is generally able to identify GRL constructs from model images, although recognition accuracy decreases as model complexity increases and image resolution decreases. Once a structurally correct model representation is recovered, GPT-5.5 correctly performs quantitative satisfaction propagation and successfully identifies optimal and threshold-satisfying strategies across all evaluated cases. To support the proposed approach, we developed *GRL Satisfaction Analyzer*, a browser-based tool that enables interactive GRL model inspection and satisfaction analysis using vision-enabled LLMs.**

***Index Terms*—Goal models, GRL, requirements engineering, large language models, vision-enabled LLMs, satisfaction analysis, jUCMNav**

I. Introduction

Goal-oriented requirements engineering (GORE) provides a structured and systematic approach for eliciting, modeling, and analyzing stakeholder intentions, system objectives, and inter-agent dependencies. Several modeling languages have been proposed in this space, including i*, KAOS, and the Goal-oriented Requirement Language (GRL) — part of the ITU-T User Requirements Notation (URN) standard [1]. Among these, GRL has gained broad adoption for capturing and reasoning about the motivations and rationales behind system design decisions.

GRL analysis techniques span qualitative, quantitative, optimization-based, compliance-oriented, and evolution-oriented reasoning methods [2]. GRL satisfaction analysis introduced qualitative and quantitative satisfaction analysis and strategy-based evaluation to compare alternative strategies and perform "what-if" analysis [3], [4]. In a related effort, Anda and Amyot [5] formalized GRL semantics using constraint satisfaction and optimization techniques, translating GRL goal models into arithmetic functions compatible with IBM CPLEX, enabling the automatic computation of optimal strategies under stakeholder and system constraints.

In practice, GRL satisfaction analysis relies on dedicated tooling such as *jUCMNav* [6], a free Eclipse-based environment for URN modeling and analysis that provides built-in support for GRL satisfaction evaluation [4]. Yet in practice, GRL models are most commonly communicated and shared as graphical diagrams. Ideally, an analyst should be able to conduct satisfaction analysis using nothing more than such a diagram, without having to re-engage with the tool environment. The recent emergence of vision-enabled large language models (LLMs), systems capable of jointly processing textual and visual inputs, opens new possibilities in this direction. If a vision-enabled LLM can faithfully parse and reason over GRL diagram images, it would become possible to perform a broad range of goal model analysis tasks directly from exported diagrams, without any intermediate model transformation or formal encoding.

In this paper, we propose a novel approach that combines multimodal reasoning with in-context domain knowledge. GPT-5.5 receives both GRL diagram images exported from *jUCMNav* and a reference document describing the GRL notation and quantitative semantics, enabling GRL satisfaction analysis without requiring any manual model transformation or access to the underlying GRL model. To the best of our knowledge, this capability has not been explored before.
The main contributions of this paper are:

- **A novel vision-based approach for GRL satisfaction analysis.** The first investigation of the capability of a vision-enabled LLM to perform GRL satisfaction analysis tasks of increasing complexity from GRL diagram images through an end-to-end workflow that automatically extracts and reasons over a structured model representation, from actor/element/link identification and satisfaction propagation to satisfaction-maximizing and threshold-satisfying strategy generation.

- **An evaluation of GPT-5.5 on GRL satisfaction analysis tasks.** An assessment of GPT-5.5 across four research questions, accounting for the inherent visual challenges of GRL diagrams, and evaluated against ground truth produced by *jUCMNav* over a set of three GRL examples.
- **GRL Satisfaction Analyzer.** A publicly available[1] browser-based prototype tool that implements the proposed approach, accompanied by a full replication package, making GRL satisfaction analysis accessible to requirements engineers without the need for formal modeling expertise or dedicated tooling.

The remainder of this paper is organized as follows. Section II introduces the GRL language and illustrates its key concepts. Section III reviews related work. Section IV presents the proposed LLM-based framework, including the research questions, methodology, the evaluation examples and metrics, and the GRL Satisfaction Analyzer prototype tool. Section V reports the experimental results. Section VI discusses the threats to validity. Section VII concludes the paper and outlines directions for future work.

II. GRL IN A NUTSHELL

In this section, we present the Goal-oriented Requirement Language (GRL) [1], illustrated through the Smart City Traffic Management model introduced in this paper.

Figure 1 shows the graphical GRL representation of the Smart City Traffic Management model, created using the *jUCMNav* [6] tool. The model contains two GRL actors: 'Traffic Management Centre" (TMC) and 'Autonomous Vehicle" (AV). Actors (illustrated as Actor) represent holders of intentions and are commonly used to model stakeholders and systems. In this example, the TMC is an institutional stakeholder, whereas the AV is a software-embedded technical agent.

The actor 'Traffic Management Centre" aims to maintain road safety across the city network, represented by the softgoal (illustrated as) *'Maintain Road Safety"*. Unlike goals (illustrated as), softgoals do not have a clear-cut satisfaction criterion. A second softgoal, *'Minimise Signal Latency"*, captures the quality concern that signal data reach consumers with minimal delay. Decomposition links (illustrated as) refine intentional elements through AND, OR, or XOR relationships. For example, *'Maintain Road Safety"* is AND-decomposed into the tasks *'Collect Sensor Data"* and *'Optimise Signal Timing"*, meaning both must be performed for the softgoal to be satisfiable. The actor also contains two leaf tasks: *'Publish Signal Data via REST API"* and *'Broadcast Signal Data via V2X Radio"*.

Tasks (illustrated as) represent activities or operationalizations used to achieve goals and softgoals. Intentional elements may be connected through contribution links (illustrated as), which capture the positive or negative impact of one element on another. A contribution link has both a qualitative type (Make (), Help (), SomePositive (), Unknown (), SomeNegative (), Hurt (), Break ()) and a quantitative weight (an integer in the range [–100,100]). In the model, *'Optimise Signal Timing"* contributes positively (SomePositive, +50) to *'Minimise Signal Latency"*, while *'Broadcast Signal Data via V2X Radio"* contributes positively (Help, +25) to the same softgoal. Conversely, *'Publish Signal Data via REST API"* contributes negatively (Hurt, –25), reflecting the network latency introduced by REST communication. The TMC actor also contains a resource (illustrated as) named *'Road Safety Regulations"*, on which the softgoal *'Maintain Road Safety"* depends internally.

The actor 'Autonomous Vehicle" contains the goal *'Navigate Safely and Efficiently"*, which, unlike a softgoal, has a clear satisfaction criterion: the vehicle must complete its journey without incident. This goal is OR-decomposed into the tasks *'Adapt Route in Real Time"* and *'Follow Predefined Safe Route"*, meaning either task can satisfy the goal. The former operationalizes dynamic navigation using signal data received from the TMC, while the latter provides a fallback when live signal data is unavailable.

The interaction between the two actors is represented by a dependency link (illustrated as) from the AV goal *'Navigate Safely and Efficiently"* to the TMC softgoal *'Minimise Signal Latency"*. This dependency captures that the AV's ability to navigate safely and efficiently depends on the TMC delivering signal data with minimal latency, enabling real-time route adaptation and reducing reliance on the predefined safe route.

Intentional elements may be assigned an importance value (an integer in the range [0,100]) to indicate their priority within an actor. In the Smart City Traffic Management model, three intentional elements carry importance values. Within the TMC actor, *'Maintain Road Safety"* is assigned the highest importance value (100), reflecting its non-negotiable priority, while *'Minimise Signal Latency"* is assigned 75, indicating a high-priority quality concern closely aligned with safety. Within the AV actor, *"Navigate Safely and Efficiently"* is assigned an importance of 100, as it represents the actor's sole objective. These values are particularly relevant for satisfiability analysis, as they establish a priority ordering when evaluating the impact of model changes on competing intentional elements.

The URN standard [1] defines GRL satisfaction analysis [3] as the process of assigning initial satisfaction values, called an *evaluation strategy*, to selected intentional elements (typically leaf elements) and propagating them through the model via different link types. Satisfaction strategies may be *qualitative*, *quantitative*, or *hybrid*. This work uses quantitative strategies, where satisfaction values range either from [–100,100] (with –100 denoting sufficiently denied and 100 sufficiently satisfied) or from [0,100] (with 0 denoting sufficiently denied and 100 sufficiently satisfied).

Figure 2 demonstrates how satisfaction values propagate through different GRL links. As shown in Fig. 2a, when a goal G is refined through an AND-decomposition, its satisfaction equals the minimum evaluation among its source elements G1 (+75), G2 (+25), and G3 (–50), resulting in a satisfaction of –50. For an OR-decomposition (Fig. 2b), satisfaction equals

[1]https://github.com/ahmedhassine05/GRL-Satisfaction-Analyzer

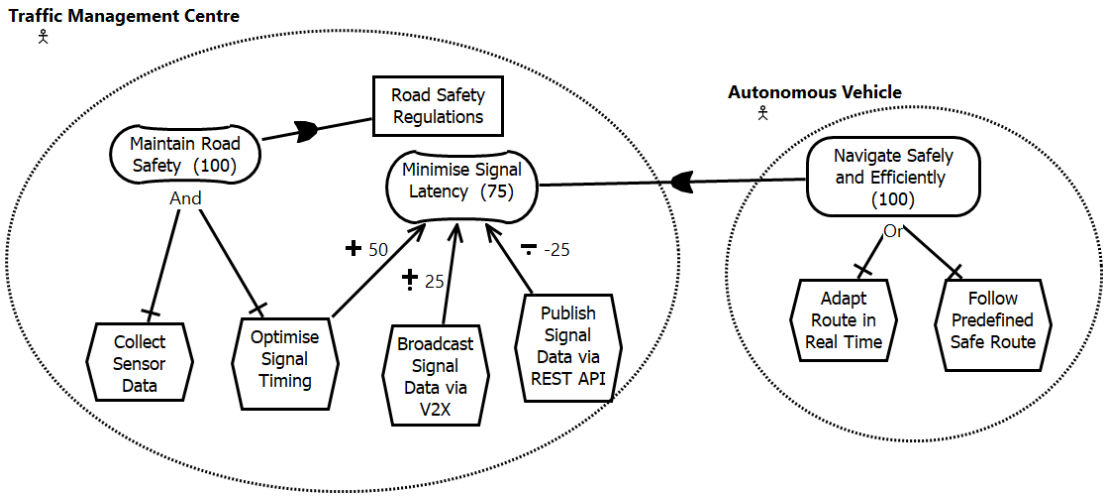
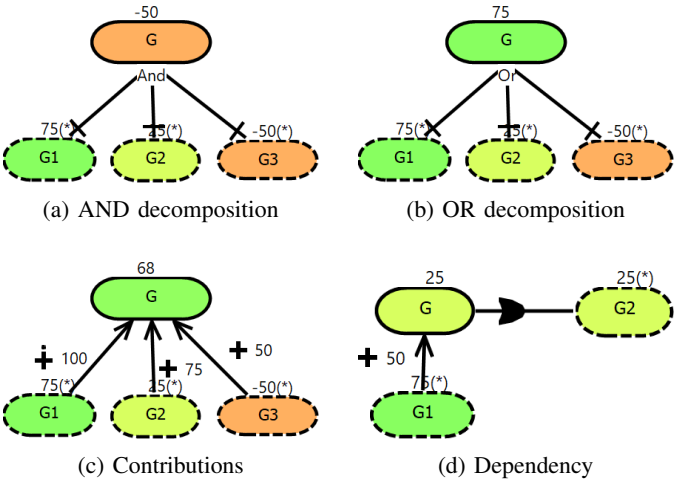


Fig. 1: Smart City Traffic Management GRL model

the maximum source evaluation; therefore, goal G evaluates to +75.



(a) AND decomposition (b) OR decomposition

(c) Contributions (d) Dependency

Fig. 2: Quantitative propagation of satisfaction values through GRL links [7]

For contribution links, the overall quantitative contribution is obtained by summing the products of each source element's evaluation and its contribution weight toward the target element. As illustrated in Fig. 2c, the satisfaction of goal G is calculated as: (75 (G1) × 100%) + (25 (G2) × 75%) + (–50 (G3) × 50%) = 68. Satisfaction values above +100 are capped at +100, while values below –100 are capped at –100. For dependency links, a source element's evaluation is bounded by the evaluation of the target element on which it depends. In Fig. 2d, given initial evaluations of +75 for G1 and +25 for G2, the satisfaction of G is computed as: min(75 × 50%, 25) = 25.

Beyond individual intentional elements, a consolidated satisfaction value can be computed for a GRL actor. The analyst determines which intentional elements are significant for the actor and assigns importance values to at least one of them. The actor's overall satisfaction is then calculated as the weighted average of the satisfaction values of its intentional elements, using their importance values as weights.

For a complete description of GRL, interested readers are referred to the URN standard [1].

III. Related Work

In this section, we review work on the use of vision LLMs for recognizing elements in requirements models represented as images, and on the application of LLMs to goal model analysis.

A. Vision LLMs for requirements model recognition

Vision LLMs (VLMs), which combine visual encoders with language models to process diagram images directly, are increasingly being explored for automated recognition of elements in requirements models. Early work by Conrardy and Cabot [8] evaluated GPT-4V, Gemini, and CogVLM for converting images of hand-drawn UML class diagrams into machine-readable models, reporting promising but imperfect results and stressing the continued need to keep a human in the loop to correct model errors. Building on this direction, Bates et al. [9] proposed a fine-tuning approach (standard and LoRA) to adapt LLaVA-based VLMs for generating executable UML code from activity and sequence diagram images, achieving BLEU and SSIM scores of 0.779 and 0.942 respectively on sequence diagrams, demonstrating feasibility but limited generalization beyond the training distribution. Similarly, Ibáñez et al. [10] assessed four VLMs (ChatGPT-4, Gemini, Amazon AI, Claude 3.5 Sonnet) on evaluating structural elements of UML class diagram images against SOLID design principles, comparing model judgments to those of expert instructors; findings revealed variable accuracy and limited reliability, corroborating the broader observation that VLMs still fall short of expert-level understanding of requirements model images.

More recently, Hassine [11] evaluated GPT-4o and GPT-4o-mini for automatically recognizing semantic elements (actors, use cases, relationships) in UML Use Case Diagram images

using a newly constructed dataset, finding that both models struggled to accurately identify and interpret key elements, often misclassifying or overlooking essential ones.

B. LLM-based goal model analysis

Large language models have recently begun to be applied to the analysis, generation, and quality improvement of goal models, opening new avenues for automating otherwise manual and error-prone tasks. Chen et al. [12] conducted the first systematic study of GPT-4 for GRL model generation, testing four prompt combinations (with/without textual syntax; with/without domain knowledge) across two case studies; while generated elements were often generic or incorrect, multi-run aggregation and iterative follow-up prompting improved output quality. Building on this foundation, Siddeshwar et al. [13] presented a technique using iterative prompt engineering to automatically generate GRL models from user stories, guiding an LLM through intentional element extraction and XML-compatible GRL generation visualisable in *jUCMNav*. Beyond model generation, Hassine [14] proposed an approach using GPT-3.5-turbo in a zero-shot setting to generate security-related traceability links between natural language requirements and goals in GRL models, implemented as a prototype tool for the Textual GRL (TGRL) format. Addressing quality issues in goal models, Alturayeif and Hassine [15] build upon a prior catalog of 17 GRL linguistic bad smells [16] by refining detection using a combination of NLP-based and LLM-based techniques and introducing automated GPT-prompt-based refactoring for 9 of them, with both detection and refactoring implemented in a tool tailored to the TGRL language.

IV. An VLM-Based Framework for GRL Satisfaction Analysis

We begin by presenting the research questions that guide this study, followed by a detailed description of the adopted methodology. We then present an overview of *GRL Satisfaction Analyzer*[2], the prototype tool developed to support our work.

The proposed approach should be viewed as an end-to-end workflow whose external input is a GRL diagram image and whose output is the requested satisfaction analysis. Internally, the approach first extracts a structured JSON representation of the GRL model from the image using the accompanying GRL reference document as in-context domain knowledge. This JSON representation is generated automatically and is never provided or edited by the user. It is subsequently reused to perform quantitative satisfaction propagation and strategy generation. Thus, the evaluation separately considers the visual extraction stage and the reasoning stage built upon the automatically extracted representation.

A. Research questions

The following research questions drive the design and evaluation of our framework:

[2]https://github.com/ahmedhassine05/GRL-Satisfaction-Analyzer

RQ1: Can GPT-5.5 VLM accurately identify GRL elements from an image of a GRL model? This RQ examines whether GPT-5.5's vision channel can faithfully identify actors, intentional elements, contribution links, and dependency links from a GRL model image exported from *jUCMNav*, and whether visual ambiguities such as overlapping links or dense layouts, where diagram elements and links are tightly packed, degrade identification accuracy.

RQ2: To what extent can GPT-5.5 VLM accurately compute GRL satisfaction values from a GRL model image and an initial strategy? This RQ examines whether GPT-5.5 can correctly propagate satisfaction values across a GRL model image starting from an initial GRL strategy, including interpreting contribution links, dependencies, decompositions, and actor-level aggregations to derive satisfaction levels of intentional elements and actors.

RQ3: Can GPT-5.5 VLM identify, from a GRL model image, the strategy that maximizes the satisfaction of a target element/actor? This RQ examines whether GPT-5.5 can determine the GRL strategy, defined as an assignment of satisfaction values to intentional elements, whose propagation through the model yields the highest satisfaction value for a specified target intentional element or actor.

RQ4: Can GPT-5.5 VLM identify threshold-satisfying strategies from a GRL model image? This RQ examines whether GPT-5.5 can determine, directly from a GRL model image, a strategy that enables a target actor or intentional element to reach or surpass a specified satisfaction threshold. When the threshold is unattainable due to the structure of the model, GPT-5.5 is expected to correctly identify this situation and report that no such strategy exists.

B. GRL examples

Different image formats may influence the performance of vision-enabled AI systems. JPEG uses lossy compression, which can introduce visual artifacts and blur fine details such as link labels and element boundaries. PNG uses lossless compression, preserving sharp edges and textual structures more faithfully. GIF, while lossless, is limited to a 256-color palette, which may reduce visual fidelity for diagrams with subtle graphical distinctions [17]. Although *jUCMNav* can also export models in BMP format, GPT-5.5 does not support BMP images; this format was therefore excluded from the study.

The study employs three GRL models of varying complexity, created using *jUCMNav* and differing in the number of actors, intentional elements, contribution links (both positive and negative), decomposition links, and dependency relationships. To assess the impact of image representation and diagram layout on recognition performance, each model was instantiated in seven variants: three image formats (PNG, JPG, and GIF), a dense layout version, two versions containing one and two link intersections respectively (e.g., Fig. 3), and a reduced-scale PNG image (50% of the original size). Model #2 is larger than Model #1 in terms of the number of actors, intentional elements, and links. Model #3 (Fig. 4)

TABLE I: Characteristics of the GRL examples. (G: Goal, S: Softgoal, T: Task, R: Resource)

#	GRL Model	Nb of actors	Nb of intentional elements (G, S, T, R)	Nb of contribution links (Positive, Negative)	Nb of decomposition links (AND, OR)	Nb of dependencies
1	Smart City Traffic System	2	(1, 2, 6, 1)	(2, 1)	(2, 2)	2
2	Electric Vehicle (adapted from [15])	3	(7, 11, 19, 0)	(17, 3)	(12, 2)	3
3	Online Retail	1	(2, 2, 3, 0)	(3, 3)	(2, 0)	0

is specifically designed to evaluate the impact of conflicting contributions on satisfaction analysis. Unlike the other two models, it contains elements that contribute both positively and negatively to different intentional elements, thereby introducing trade-offs between competing objectives. Consequently, computing the satisfaction of a target actor or intentional element requires balancing beneficial and detrimental effects, rather than propagating exclusively positive influences through the model. Table I summarizes the characteristics of each model.

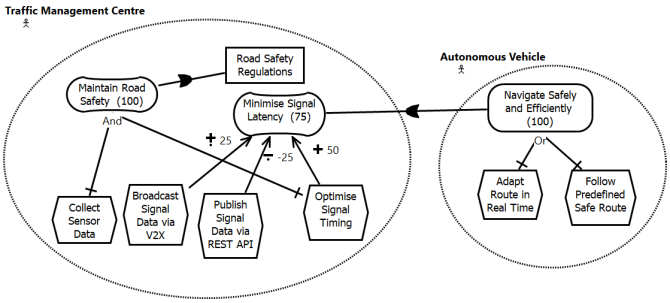


Fig. 3: Smart City Traffic with two link intersections

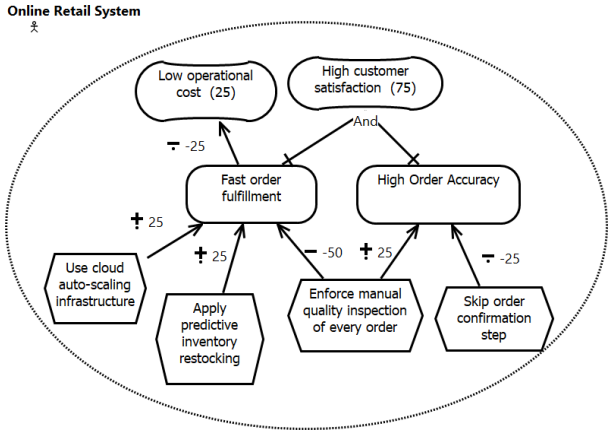


Fig. 4: Model 3: Online Retail System

C. Methodology

Figure 5 illustrates the experimental methodology adopted to answer the four research questions. The methodology is organized as a pipeline in which the output of each research question serves as the input for the subsequent one. All experiments were conducted using GPT-5.5 with image inputs corresponding to the GRL model variants described

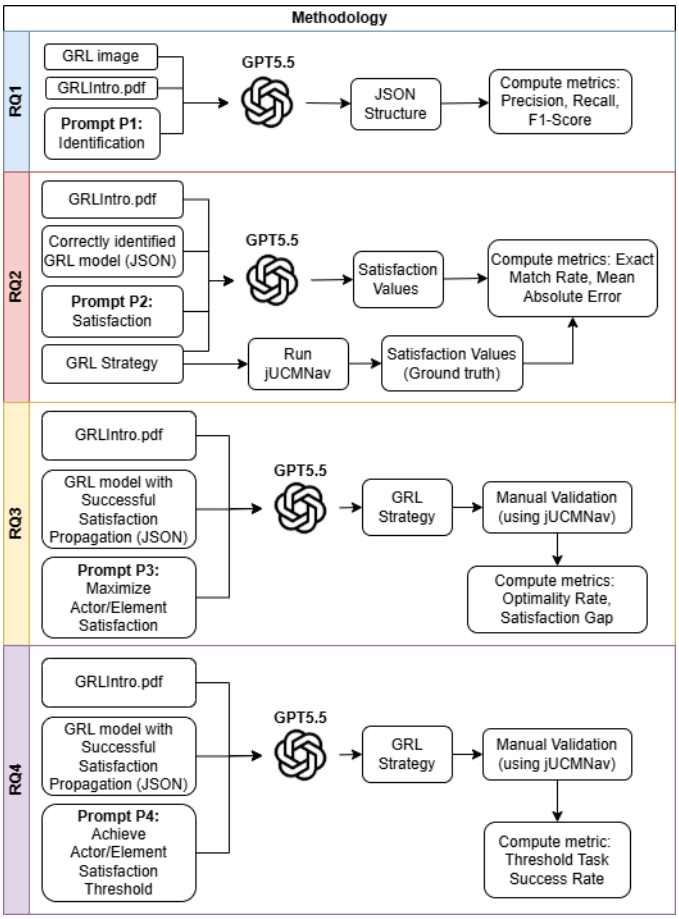


Fig. 5: Methodology

in Section IV-B. To reduce variability, each experiment was repeated 5 times using the same prompting configuration, and the reported results correspond to the average across the 5 runs.

RQ1: Identification of GRL elements. For the first research question, GPT-5.5 was provided with a GRL diagram image together with a reference document describing the GRL notation and semantics. The model was prompted to identify all actors, intentional elements, and links appearing in the diagram and to return the extracted information as a structured JSON object. Figure 6 lists the prompt used for RQ1. The generated JSON representation is used to populate the GUI of the *GRL Satisfaction Analyzer* tool, as shown in Fig. 10.

Since RQ1 asks whether GPT-5.5 can correctly identify GRL elements from a model image, we treat it as a set retrieval

Prompt P1: Identification of GRL Elements

```
You are a GRL diagram parser. Extract every
actor, intentional element, and link from this
GRL diagram image.
Refer to the attached GRL reference document for
notation, element types, and link types.
Output a single raw JSON object -- no markdown, no
prose, nothing else.
Exact schema:
{"actors":[{"id":"a1","name":"Actor Name"}],
"elements":[{"id":"e1","name":"Element Name",
"type":"Goal","actorId":"a1","importance":100,
"decompositionType":"and"}],
"links":[{"id":"l1","type":"Decomposition",
"subtype":"and","sourceId":"e2","targetId":"e1"}]}
Field rules:
- element type: exactly one of Goal SoftGoal Task
Resource Belief
- importance: integer from diagram (e.g. "(100)")
-- omit if not shown
- decompositionType: "and" or "or" -- omit if not
shown
- link type: exactly one of Decomposition
Contribution Dependency
- Decomposition: set subtype "and"/"or", omit
value
- Contribution: set value as signed integer, omit
subtype
- Dependency: omit subtype and value
- Omit any field whose value is null
```

Fig. 6: Prompt used to identify GRL actors, intentional elements, and links from a GRL diagram image (RQ1).

problem and evaluate it using precision, recall, and F1-score computed per element type (actors, intentional elements, contribution links, and dependency links). Precision measures the proportion of VLM-identified elements that are correct, while recall measures the proportion of ground-truth elements that were successfully identified. The F1-score is their harmonic mean.

RQ2: Computation of satisfaction values. Only diagrams for which all GRL constructs were correctly identified in RQ1 were retained. Minor misclassifications of intentional element types, such as a softgoal identified as a goal or vice versa, do not affect the propagation of satisfaction values and were therefore tolerated; a diagram was excluded only when structural elements (e.g., links or actors) were missing or incorrectly identified. The JSON representation automatically generated during RQ1 is then used, together with the GRL reference document and an initial satisfaction strategy, to perform quantitative satisfaction propagation. Although RQ2 operates on the extracted representation rather than directly on the image, this representation is produced automatically by the first stage of the proposed pipeline and requires no user intervention. GPT-5.5 is then asked to propagate satisfaction values through the GRL model and compute the resulting satisfaction values of all intentional elements and actors. Ground-truth values are obtained using the built-in GRL evaluation mechanism of *jUCMNav* and compared with the values produced by GPT-5.5. Figure 7 illustrates the prompt

Prompt P2: GRL Satisfaction Propagation

```
Perform GRL satisfaction value propagation.
Apply the propagation rules defined in the
attached GRL reference document.
Satisfaction values are integers in [{min};
{max}].
INPUT FORMAT:
- [SEED=x]: this element has an initial
satisfaction value of x (treat as a fixed leaf
input).
- [UNFILLED]: this element has no assigned value;
compute it from the graph or omit if unreachable.
OVERRIDE DETECTION:
After computing bottom-up, if a [SEED=x] element's
computed value (from its children) differs from
its seed x, report it in "overrides". The COMPUTED
value is authoritative.
Elements:
{element lines, each as:
id: name (type, actor: ActorName) [SEED=x or
UNFILLED]}
Links (sourceId -[type, subtype?, value?]->
targetId):
{link lines, each as:
sourceId -[Type, subtype?, value?]-> targetId}
Respond with ONLY this JSON -- no prose, no
markdown fences:
{"propagation":{"ELEM_ID":NUMBER,...},
"overrides":[{"id":"ELEM_ID",
"userValue":X,
"computedValue":Y,
"note":"reason"}],
"reasoning":"brief (max 3 sentences)"}
Rules:
- "propagation": include only elements with a
computed value
- Omit unreachable [UNFILLED] elements
- "overrides": include [SEED=x] elements whose
bottom-up result ≠ x
- All NUMBER values must be integers in [{min};
{max}]
```

Fig. 7: Prompt used for GRL satisfaction value propagation (RQ2).

used for RQ2.

To evaluate the results of RQ2, we compare VLM-computed values against ground-truth values produced by *jUCMNav* using two metrics: (1) the Exact Match Rate (EMR) measures the proportion of intentional elements and actors for which the VLM returns the correct satisfaction value, and (2) the Mean Absolute Error (MAE) quantifies the average deviation across all elements: $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|v_i - \hat{v}_i|$, where v_i is the ground-truth satisfaction value and $\hat{v}_i$ is the VLM-computed value for element i.

RQ3: Identification of satisfaction-maximizing strategies. Only diagrams for which GPT-5.5 correctly computed satisfaction values in RQ2 were retained. To evaluate this research question, the JSON representation automatically extracted during RQ1 was reused together with the GRL reference document describing the notation and quantitative semantics. This intermediate representation was generated automatically by the proposed approach and was never supplied or edited by the user. GPT-5.5 was then provided with a target

actor or intentional element. Two variants of the experiment were considered: (1) maximizing the satisfaction of a target actor and (2) maximizing the satisfaction of a target intentional element. The generated strategy was manually validated by the two authors (and verified using *jUCMNav*) to determine whether it achieved the optimal satisfaction value. Figure 8 illustrates the prompt used for RQ3.

Prompt P3: Actor/Element Satisfaction Maximization

```
Given the GRL model below, produce a satisfaction
strategy to maximize {scopeDesc}.
Apply GRL satisfaction propagation rules from the
attached reference document.
GRL Model (JSON):
{grlJson}
CONSTRAINT: Seed only leaf elements (elements with
no incoming decomposition or contribution links),
regardless of type.
All non-leaf values must be propagated upward
through the GRL links.
Respond with ONLY a raw JSON object -- no
markdown, no explanation:
{"seeds":[{"name":"...",
"type":"Goal|SoftGoal|Task|Resource|Belief",
"actor":"...",
"satisfaction":75}],
"model":[{"name":"...",
"type":"Goal|SoftGoal|Task|Resource|Belief",
"actor":"...",
"satisfaction":50}]}
Rules:
- seeds: leaf elements only
- A leaf element has no incoming decomposition or
contribution links
- In seeds, order from highest to lowest
satisfaction
- model: ALL elements in the model with their
final propagated satisfaction values
- Every element must appear in the model section
- satisfaction is integer in [{min}; {max}]
- {min} = fully denied
- {max} = fully satisfied
- In model, group by actor, then order by
satisfaction descending within each actor
- Output ONLY the JSON object, nothing else
```

Fig. 8: Prompt used to identify a satisfaction-maximizing strategy for a target actor or intentional element (RQ3).

To evaluate the results of RQ3, we use the following two metrics: (1) the Optimality Rate (OR) measures the proportion of cases in which the VLM-identified strategy achieves the same satisfaction value for the target element as the ground-truth optimal strategy, and (2) the Satisfaction Gap (SG) measures the difference between the target element's satisfaction value under the optimal strategy and under the VLM strategy: $\Delta s = s^* - \hat{s}$, where s^* is the optimal satisfaction value and $\hat{s}$ is the value achieved by the VLM strategy.

RQ4: Identification of Threshold-Satisfying Strategies. Finally, to evaluate this research question, the JSON representation automatically extracted during RQ1 was reused together with the GRL reference document describing the notation and quantitative semantics. This intermediate representation was generated automatically by the proposed approach and was never supplied or edited by the user. GPT-5.5 was then asked to identify a strategy that enables a target actor or intentional element to reach or surpass a specified satisfaction threshold. If the threshold could not be achieved given the structure and constraints of the GRL model, GPT-5.5 was expected to correctly report that the threshold was unattainable. As in RQ3, experiments were conducted for both actor-level and intentional-element-level targets. The resulting strategies were compared against reference strategies manually derived by the two authors and verified using *jUCMNav* to determine whether the specified satisfaction threshold was achieved or correctly identified as unattainable.Figure 9 illustrates the prompt used for RQ4.

Prompt P4: Threshold Satisfaction Strategy

```
Given the GRL model below, make {scopeDesc} reach
or surpass {threshold}.
If unattainable, maximize instead and set
"threshold_reached": false.
Apply GRL satisfaction propagation rules from the
attached reference document.
GRL Model (JSON):
{grlJson}
CONSTRAINT: Seed only leaf elements (elements with
no incoming decomposition or contribution links),
regardless of type.
All non-leaf values must be propagated upward
through the GRL links.
Respond with ONLY a raw JSON object -- no
markdown, no explanation:
{"threshold_reached":true,
"seeds":[{"name":"...",
"type":"Goal|SoftGoal|Task|Resource|Belief",
"actor":"...",
"satisfaction":75}],
"model":[{"name":"...",
"type":"Goal|SoftGoal|Task|Resource|Belief",
"actor":"...",
"satisfaction":50}]}
Rules:
- threshold_reached: true if {scopeDesc} reaches
≥{threshold}; false if unattainable
- When false, also include "best_achievable":
INTEGER
- seeds: leaf elements only
- A leaf element has no incoming decomposition or
contribution links
- In seeds, order from highest to lowest
satisfaction
- model: ALL elements in the model with their
final propagated satisfaction values
- Every element must appear in the model section
- satisfaction is integer in [{min}; {max}]
- {min} = fully denied
- {max} = fully satisfied
- In model, group by actor, then order by
satisfaction descending within each actor
- Output ONLY the JSON object, nothing else
```

Fig. 9: Prompt used to identify a threshold satisfaction strategy for a target actor or intentional element (RQ4).

To evaluate the results of RQ4, we use the *Threshold Task Success Rate* (TTSR), defined as the proportion of threshold-achievement tasks successfully solved by GPT-5.5, where success means either reaching a reachable threshold or

correctly identifying an unreachable threshold as unattainable (TTSR = #successful cases/#total cases).

D. Tool: GRL Satisfaction Analyzer

GRL Satisfaction Analyzer is a browser-based tool for the interactive inspection and satisfaction analysis of Goal-oriented Requirements Language (GRL) models, implemented as a single self-contained HTML file using vanilla JavaScript, CSS, and the browser-native DOM and Fetch APIs, requiring no build toolchain, backend server, or external runtime dependencies. Figure 10 illustrates the graphical user interface (GUI), which consists of the following components:

- **Configuration (top bar)**: Select LLM provider (Ollama or OpenAI), choose a model, supply the endpoint URL or API key, and verify connectivity via the PING button. Status dot confirms live connection.
- **Input**: Drag-and-drop or click-upload a GRL model diagram as an image file. Uploading a PDF file for GRL rules is strongly recommended (it is used to guide accurate identification and correct satisfaction propagation); without it, built-in fallback rules apply.
- **GRL Model browser (left panels)**: Extracted Actors, Intentional Elements, and Links populate three resizable panels. Click an actor to filter elements.
- **Sat Mode:** Assign numeric satisfaction seeds to elements (i.e., initial satisfaction values). Use "Propagate" button to invoke LLM-computed bottom-up propagation. Results appear in the Output and Satisfaction Strategy panels.
- **Side panel:** Tracks file history and exposes LLM hyperparameters (e.g., reasoning_effort, max_tokens). Note that temperature was not considered a tunable hyperparameter, since GPT-5.5 does not expose temperature control to users.
- **Query panel (right)**: Dedicated buttons run two functions: "Maximize Element/Actor Satisfaction" and "Reach Threshold" (with a numeric target), each producing a GRL strategy as output. In addition, the user may query the LLM about the GRL model by submitting free-text questions. The default satisfaction value range is [0,100], but users can optionally change it to [–100,100].

V. Results

This section presents the results of our experiments, addressing the research questions. Unless otherwise specified, GPT-5.5 was executed using its default reasoning effort setting (*medium*) and without a fixed seed value. For Models #1 and #3, the maximum output length was set to 4096 tokens, which was sufficient to accommodate the generated JSON responses. For Model #2, the limit was increased to 8192 tokens due to the larger size of the generated JSON output resulting from the greater number of intentional elements, and links present in the model.

A. Identification of GRL Elements (RQ1)

Table II reports the identification accuracy for all model variants. GPT-5.5 achieved perfect performance on Model #3 across all seven image variants. For Model #1, all observed errors corresponded exclusively to confusions between goals and softgoals, with no missing or spurious constructs. Model #2 proved substantially more challenging due to its larger size and higher structural complexity. While several errors again involved goal/softgoal confusions, a number of structural errors were also observed, including missing and spurious contribution links, an incorrectly identified decomposition link, and dependency link misclassifications.

The PNG variants achieved the highest overall accuracy, whereas the lowest performance was observed for the 50%-scale version of Model #2, which contained numerous intentional-element type confusions and several structural errors. This result suggests that reduced image size has a stronger impact on recognition accuracy than dense layouts or link intersections.

B. Propagation of GRL Satisfaction Values (RQ2)

Only GRL models that were correctly identified during RQ1 were carried forward to RQ2. Consequently, we retained only the PNG variants of the three models. The PNG versions of Models #1 and #3 were identified without errors, while the PNG version of Model #2 contained a single goal/softgoal confusion. Since goals and softgoals are propagated using the same quantitative satisfaction rules in GRL, this misclassification does not affect the computation of satisfaction values and was therefore considered acceptable for the remaining phases.

For each model, three different satisfaction strategies were evaluated. Across all nine evaluated strategies, GPT-5.5 correctly computed the propagated satisfaction values for all intentional elements and actors. The Exact Match Rate (EMR) therefore reached 100%, while the Mean Absolute Error (MAE) was 0 for all evaluated cases. These results indicate that once a GRL model is correctly extracted, GPT-5.5 is capable of accurately applying the quantitative propagation semantics.

C. Identification of Satisfaction-Maximizing Strategies (RQ3)

For each model, three maximization tasks were evaluated: one actor-level maximization task and two intentional-element maximization tasks. Across all nine evaluated tasks, GPT-5.5 generated optimal GRL strategies, achieving an Optimality Rate (OR) of 100% and a Satisfaction Gap (SG) of 0.

These results indicate that GPT-5.5 can successfully identify satisfaction-maximizing strategies for the evaluated GRL models once a correct structural representation of the model (i.e., JSON format in our case) is available.

D. Identification of Threshold-Satisfying Strategies (RQ4)

For each model, three threshold-achievement tasks were evaluated: one actor-level threshold task and two intentional-element threshold tasks. Across all nine evaluated tasks, GPT-5.5 successfully generated GRL strategies that achieved the specified satisfaction thresholds, resulting in a Threshold Task Success Rate (TTSR) of 100%.

These results suggest that GPT-5.5 can reliably identify threshold-satisfying strategies for the evaluated GRL models

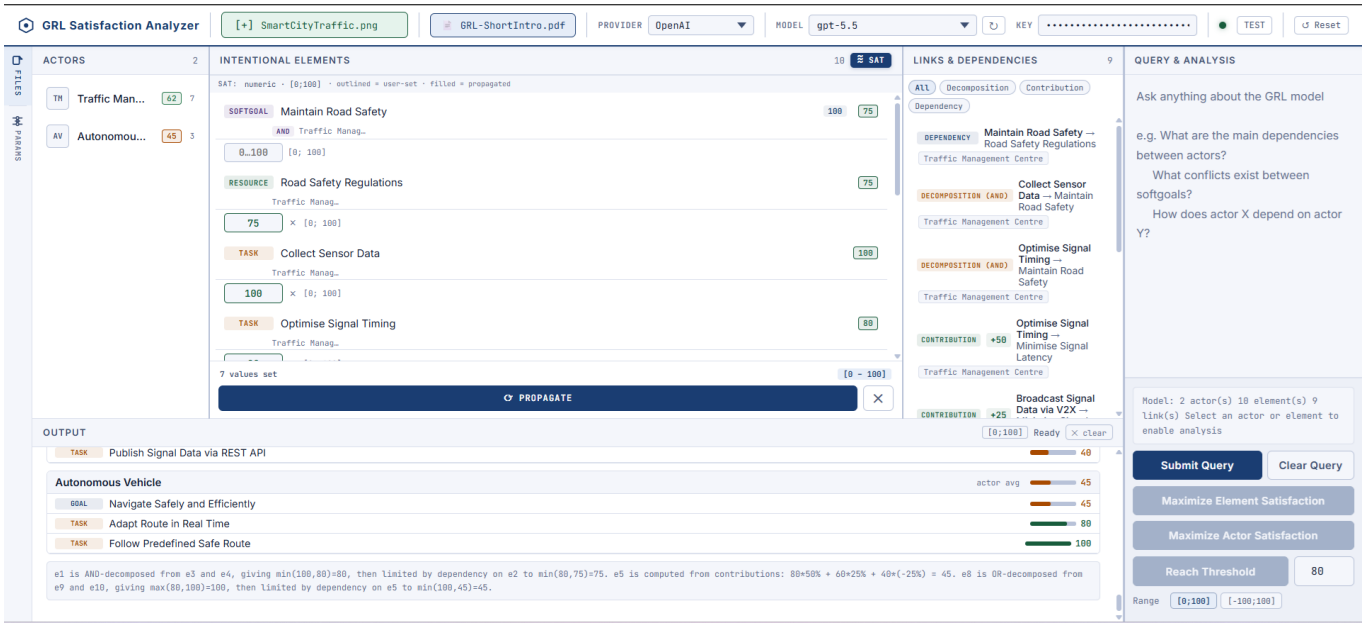


Fig. 10: GRL Satisfaction Analyzer tool GUI

TABLE II: RQ1 – Precision (P), Recall (R), and F1-score for GRL construct identification.

Variant	Model #1			Model #2			Model #3		
	P	R	F1	P	R	F1	P	R	F1
PNG	1.00	1.00	1.00	0.99	0.99	0.99	1.00	1.00	1.00
GIF	1.00	1.00	1.00	0.95	0.95	0.95	1.00	1.00	1.00
JPG	0.90	0.90	0.90	0.96	0.96	0.96	1.00	1.00	1.00
Dense	0.90	0.90	0.90	0.97	0.97	0.97	1.00	1.00	1.00
Small PNG (50%)	0.95	0.95	0.95	0.80	0.80	0.80	1.00	1.00	1.00
1 Intersection	0.95	0.95	0.95	0.96	0.96	0.96	1.00	1.00	1.00
2 Intersections	0.90	0.90	0.90	0.96	0.96	0.96	1.00	1.00	1.00

and correctly reason about satisfaction thresholds once the underlying GRL structure has been recovered.

VI. Threats to Validity

The proposed approach and the GRL models used in the evaluation are subject to several threats to validity that we classify into three categories: internal, construct, and external.

- **Internal validity:** A key threat to internal validity arises from the nondeterministic nature of large language models. Since GPT-5.5 was executed without a fixed seed value, repeated executions of the same prompt may produce different outputs. To mitigate this threat, each experiment was repeated five times using identical prompts and configurations, and the reported results correspond to the average across runs.
  Another threat concerns the correctness of the ground-truth solutions used to evaluate RQ3 and RQ4. Since jUCMNav does not natively support strategy optimization, the optimal strategies used for RQ3 and the threshold-achievement cases used for RQ4 were derived manually and independently validated by the two authors, with disagreements resolved by consensus. Nevertheless, there is no guarantee that all manually derived solutions are complete or globally optimal.
- **Construct validity:** A threat to construct validity concerns the evaluation metrics used in the study. For RQ1, precision, recall, and F1-score measure the correctness of GRL construct identification but do not fully capture the severity of different error types. For example, confusing a goal with a softgoal may have a different practical impact than missing an intentional element or a link. To partially mitigate this threat, only models for which all structural constructs (e.g., actors and links) were correctly identified were retained for RQ2–RQ4. Furthermore, goal/softgoal confusions were tolerated because these element types follow the same quantitative satisfaction propagation rules in GRL and therefore do not affect the computation of satisfaction values. Similarly, for RQ3 and RQ4, the Optimality Rate (OR), Satisfaction Gap (SG), and Threshold Task Success Rate (TTSR) provide quantitative measures of performance but may not fully capture the quality, diversity, or explainability of the generated strategies.
- **External validity:** The main threat to external validity concerns the limited number and diversity of GRL models used in the study. Although the selected models vary in

size, structure, and complexity, they cannot represent the full spectrum of GRL models encountered in practice. Larger and more complex models may exhibit visual and structural characteristics not captured in the current dataset.

Another limitation stems from the exclusive use of GRL models exported from *jUCMNav* and the restricted set of image variants considered (PNG, JPG, GIF, dense layouts, link intersections, and reduced-size images). Additional visual transformations, such as hand-drawn sketches, low-resolution screenshots, or heavily compressed images, were not evaluated.

Finally, the study was conducted exclusively using GPT-5.5. The observed results may not generalize to other vision-enabled LLMs, which may differ in visual perception, reasoning capabilities, and prompt sensitivity. Future work should therefore replicate the study using additional models and larger benchmark datasets.

VII. Conclusion and future work

This paper presented a vision-based approach that leverages GPT-5.5 to perform GRL satisfaction analysis directly from diagram images exported from jUCMNav, without requiring intermediate model transformations or access to the underlying model representation. The evaluation showed that GPT-5.5 is generally able to identify GRL constructs from model images, although recognition accuracy decreases as model complexity increases and image resolution decreases. Once a structurally correct model representation is recovered, GPT-5.5 accurately performs quantitative satisfaction propagation and successfully generates both satisfaction-maximizing and threshold-satisfying strategies. The approach is implemented in the browser-based *GRL Satisfaction Analyzer* tool.

Future work will extend the evaluation to a larger and more diverse set of GRL models and investigate additional image-based analysis tasks, such as consistency checking and change impact analysis. We also plan to explore the applicability of the approach to other goal modeling languages, including *iStar*. Another promising direction is the evaluation of additional vision-enabled LLMs to better understand the generalizability of the observed results and the impact of different visual reasoning capabilities on goal model analysis.

References

[1] ITU-T, "Recommendation Z.151 (10/18), User Requirements Notation (URN) language definition, Geneva, Switzerland," Geneva,Switzerland, 2018. [Online]. Available: http://www.itu.int/rec/T-REC-Z.151/en

[2] D. Amyot, O. Akhigbe, M. Baslyman, S. Ghanavati, M. Ghasemi, J. Hassine, L. Lessard, G. Mussbacher, K. Shen, and E. Yu, "Combining goal modelling with business process modelling two decades of experience with the user requirements notation standard," *Enterp. Model. Inf. Syst. Archit.*, vol. 17, 2022. [Online]. Available: https://doi.org/10.18417/emisa.17.2

[3] D. Amyot, S. Ghanavati, J. Horkoff, G. Mussbacher, L. Peyton, and E. S. K. Yu, "Evaluating goal models within the goal-oriented requirement language," *Int. J. Intell. Syst.*, vol. 25, no. 8, pp. 841–877, 2010. [Online]. Available: https://doi.org/10.1002/int.20433

[4] D. Amyot, A. Shamsaei, J. Kealey, E. Tremblay, A. Miga, G. Mussbacher, M. Alhaj, R. Tawhid, E. Braun, and N. Cartwright, "Towards advanced goal model analysis with jucmnav," in *Proceedings of the 2012 International Conference on Advances in Conceptual Modeling*, ser. ER'12. Berlin, Heidelberg: Springer-Verlag, 2012, p. 201–210. [Online]. Available: https://doi.org/10.1007/978-3-642-33999-8_25

[5] A. A. Anda and D. Amyot, "An optimization modeling method for adaptive systems based on goal and feature models," in *10th IEEE International Model-Driven Requirements Engineering, MoDRE@RE 2020, Zurich, Switzerland, August 31, 2020*. IEEE, 2020, pp. 11–20. [Online]. Available: https://doi.org/10.1109/MoDRE51215.2020.00008

[6] jUCMNav, "v7.0.0," https://github.com/JUCMNAV, University of Ottawa, Canada, last Accessed May 2026.

[7] J. Hassine and M. Tukur, "Measurement and classification of inter-actor dependencies in goal models," *Softw. Syst. Model.*, vol. 21, no. 6, pp. 2267–2310, 2022. [Online]. Available: https://doi.org/10.1007/s10270-021-00961-3

[8] A. D. Conrardy and J. Cabot, "From image to UML: first results of image-based UML diagram generation using llms," in *Proceedings of the STAF 2024 Workshops: AgileMDE 2024, LLM4MDE 2024, and MeSS 2024 co-located with the International Conference on Software Technologies: Applications and Foundations (STAF 2024) Enschede, The Netherlands, July 8-11, 2024*, ser. CEUR Workshop Proceedings, H. Alfraihi, F. Basciani, G. Caltais, N. Ferry, J. A. H. López, L. Iovino, R. Jongeling, S. Klikovits, S. K. Rahimi, R. Rubei, S. Y. Tehrani, J. Troya, M. Wessel, and V. Zaytsev, Eds. CEUR-WS.org, 2024, pp. 55–65. [Online]. Available: https://ceur-ws.org/Vol-3727/paper5.pdf

[9] A. Bates, R. Vavricka, S. Carleton, R. Shao, and C. Pan, "Unified modeling language code generation from diagram images using multimodal large language models," *CoRR*, vol. abs/2503.12293, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2503.12293

[10] M. Ibáñez, M. L. Barrón-Estrada, and R. Zataraín-Cabada, "Can multimodal large language models grade like an expert? A study on UML class diagram assessment accuracy," *Comput. Appl. Eng. Educ.*, vol. 33, no. 5, 2025. [Online]. Available: https://doi.org/10.1002/cae.70080

[11] J. Hassine, "Evaluating multi-modal llms for automatically recognizing semantic elements in UML use case diagram images," in *IEEE International Conference on Software Analysis, Evolution and Reengineering, SANER 2025, Montreal, QC, Canada, March 4-7, 2025*. IEEE, 2025, pp. 861–866. [Online]. Available: https://doi.org/10.1109/SANER64311.2025.00092

[12] B. Chen, K. Chen, S. Hassani, Y. Yang, D. Amyot, L. Lessard, G. Mussbacher, M. Sabetzadeh, and D. Varró, "On the use of GPT-4 for creating goal models: An exploratory study," in *31st IEEE International Requirements Engineering Conference, RE 2023 - Workshops, Hannover, Germany, September 4-5, 2023*, K. Schneider, F. Dalpiaz, and J. Horkoff, Eds. IEEE, 2023, pp. 262–271. [Online]. Available: https://doi.org/10.1109/REW57809.2023.00052

[13] V. Siddeshwar, S. A. Alwidian, and M. Makrehchi, "A comparative study of large language models for goal model extraction," in *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, MODELS Companion 2024, Linz, Austria, September 22-27, 2024*, M. Wimmer, A. Egyed, B. Combemale, and M. Chechik, Eds. ACM, 2024, pp. 253–263. [Online]. Available: https://doi.org/10.1145/3652620.3686246

[14] J. Hassine, "An llm-based approach to recover traceability links between security requirements and goal models," in *Proceedings of the 28th International Conference on Evaluation and Assessment in Software Engineering, EASE 2024, Salerno, Italy, June 18-21, 2024*. ACM, 2024, pp. 643–651. [Online]. Available: https://doi.org/10.1145/3661167.3661261

[15] N. Alturayeif and J. Hassine, "Refactoring goal-oriented models: a linguistic improvement using large language models," *Softw. Syst. Model.*, vol. 25, no. 1, pp. 51–79, 2026. [Online]. Available: https://doi.org/10.1007/s10270-024-01254-1

[16] ——, "Detection of linguistic bad smells in GRL models: An NLP approach," in *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. IEEE Press, 2023, p. 318–327. [Online]. Available: https://doi.org/10.1109/MODELS-C59198.2023.00062

[17] S. Dodge and L. Karam, "Understanding how image quality affects deep neural networks," in *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*, 2016, pp. 1–6.