Supporting Systems Modeling Using Semantic Cues for Concept Development

Natansh Vyas
Department of Science and Industry systems
University of South-Eastern Norway
Kongsberg, Norway
natansh.vyas@usn.no

Gerrit Muller
Department of Science and Industry systems
University of South-Eastern Norway
Kongsberg, Norway
gerrit.muller@gmail.com

Mo Mansouri
Department of Systems and Enterprises
Stevens Institute of Technology
Hoboken, United States
mmansour@stevens.edu

Kristin Falk
Department of Science and Industry systems
University of South-Eastern Norway
Kongsberg, Norway
kristin.falk@usn.no

Omid Razbani
Department of Science and Industry systems
University of South-Eastern Norway
Kongsberg, Norway
omid.razbani@usn.no

***Abstract*—Engineering complex systems requires integrating multiple perspectives to create a cohesive whole without overlooking critical elements. In the early stages of concept development, systems engineers often work independently, with limited time and minimal stakeholder input. This constraint increases the likelihood of missing essential system components, leading to models that lack completeness and detail due to cognitive limitations. This work presents an approach for traversing a knowledge graph to extract semantically related concepts from words used in diagrams. The graph search strategy is based on exploration and refinement activities as defined in the systems lifecycle management standard, ISO/IEC/IEEE 24748-1:2024. The weighted concepts are provided as semantic cues to aid word retrieval, optimizing the use of working memory capacity. The proposed method is implemented as a plugin for the diagramming tool Draw.io, utilizing the semantic network, ConceptNet. The plugin dynamically extracts related words and their relationship strengths, visualizing them as word clouds to enhance conceptual modeling. The approach was evaluated using 25 Systemigrams from the literature for measuring its ability to highlight critical features and reduce ambiguity in system representation. Results indicate that the approach helps identify 24.15% more critical features and provides detailed elaboration on 15.5% of them, aligning with key conclusions drawn from the Systemigrams. Additionally, findings suggest that the tool's effectiveness depends on its ability to provide domain-specific vocabulary, particularly in scenarios requiring scientific and engineering comprehension.**

***Index Terms*—systems modeling, systems engineering, concept development, semantic network, knowledge representation.**

I. Introduction

The concept of 'systems' broadly refers to thinking in terms of wholes rather than parts, where considering all aspects of products remains essential. The systems approach becomes crucial when developing safety-critical or mission-critical products with high complexity [1], [2]. Failing to account for certain system elements can lead to long-term, irreversible, and potentially catastrophic consequences. Systems theory pioneers describe 'systems thinking' as this distinct set of cognitive and strategic practices [3] of conceiving interrelationships, shifting perspectives, and considering uncertainty outside system boundaries [4]. Since systems thinking is not often an innate skill [5], the engineering development process has evolved over decades to incorporate it through systems modelling. This design approach of modelling spans a broad spectrum, ranging from soft to hard methodologies, with the general-purpose visual modelling languages like unified modelling language [6] and Systemigrams [7] as a dominant choice for conceptualizing and analysing complex engineering systems during early phases of the project.

Traditional aerospace and defense sectors have widely adopted systems modelling practices [8], [9], but organizations on lower levels of systems engineering capability often face challenges in providing the right environment for systems engineers during the conceptual design phase [10]. One significant issue is the disproportionate allocation of staffing resources and project schedules, which tend to prioritize later phases over the concept phase [11]. This imbalance limits the collaboration among internal stakeholders with domain knowledge, making valuable insights about component-level details and lessons learnt from previous projects inaccessible to the systems engineer. Low staffing levels further reduce team diversity for a constructive dialogue and collective brainstorming. This constrains the ability to generate a broad range of concepts that meet the customer needs [12], [13]. This remains especially true when the systems engineer has transitioned to a new project or industry and is still gaining domain-specific knowledge. These challenges are exacerbated by the need for systems engineers to independently develop and evaluate system concepts under tight deadlines. This increases the risk of oversights and reduces the overall effectiveness of the conceptual design process in terms of its completeness, cohesiveness, and consideration of multiple perspective.

This research was supported by the Research Council of Norway.

This is a pre-copyedited version of a contribution published in 2025 IEEE 29th International Conference on Intelligent Engineering Systems (INES) published by IEEE. The definitive authenticated version is available online at IEEE Explore: https://doi.org/10.1109/INES67149.2025.11078208

A. Cognitive mechanisms

Systems engineers often experience cognitive overload during the modeling process, which diagramming tools could help mitigate. However, despite the central role of cognition in systems engineering, research on these challenges—particularly those encountered during modeling—remains limited. Figure 1 illustrates the impact of semantic cues on cognitive processing for systems modelling, based on Atkinson–Shiffrin model [14].

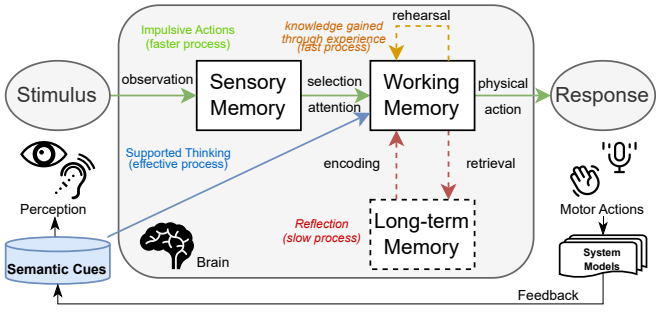

Fig. 1. The role of semantic cues in information processing based on the first-order multi-store memory model proposed by Atkinson and Shiffrin [14].

Frank's Capacity for Engineering Systems Thinking (CEST) model identifies 16 competencies essential for systems engineers [15], where they emphasize the importance of cognitive mechanisms. Sibo et al., in their review of relationship between creative thinking and cognitive load, found that while excessive cognitive load hinders creativity, strategic management can enhance performance [16]. Similarly, Redifer et al. highlighted working memory's role in regulating cognitive resources, noting that its efficiency depends on prior knowledge, task complexity, and individual traits [17].

Working memory also affects inhibition, and novice systems engineer may frequently experience cognitive blockages during their learning phase due to information overload. Conway and Engle observed the importance of working memory in tasks requiring effortful search for retrieving information from long-term memory [18], an essential capability for concept generation. Schelble et al. further highlight that generating multiple examples within a category is an important component in this search process [19]. Interestingly, Frank's cognitive competencies—such as avoiding fixation, considering multiple perspectives, and drawing analogies—closely align with established findings in cognitive science [15].

Sibo et al. classify cognitive load management strategies into three main types: external resources, environmental interventions, and self-regulation [16]. External resources, such as semantic cues, assist in retrieving relevant concepts from long-term memory while helping users evaluate them in working memory, reducing distractions and improving focus. Gwizdka's experimental study further supports this, demonstrating how search interfaces influence cognitive effort, emphasizing the role of external tools like word clouds in managing cognitive load and enhancing conceptual clarity [20].

B. Present state of practice

Systems engineers develop conceptual and graphical models that connect high-level stakeholder needs to detailed engineering specifications [21]. These models help reduce ambiguity, mitigate risks [22], and ensure feasibility throughout the development process. As shown in Figure 2, models vary in scope, formality, and abstraction depending on the development life cycle phase [23]. Early-stage models rely heavily on natural language [24], enabling broad concept exploration with reduced precision. These models are essential for effective communication, informed decision-making, and comprehensive analysis among diverse stakeholders.

Modern digital diagramming tools such as Microsoft Visio, LucidChart, Dassault MagicDraw, and IBM Rational Rhapsody have largely replaced traditional free-hand drawings, improving reusability and maintainability [25]. The widespread adoption of these tools presents an opportunity to enhance the modeling process by incorporating cognitive support mechanisms, benefiting both systems engineers and software platform providers. Modeling a system involves exploring concepts through discussions, internet research, and document analysis. While Large Language Models (LLMs) assist in retrieving information [26], they suffer from reliability issues [27], as well as concerns related to explainability, trustworthiness, security, and privacy [28]. More importantly, LLMs fail to address cognitive overload and often worsen it by generating excessive and irrelevant responses [29], simultaneously inducing overreliance and reduced critical thinking [30]. Systems engineers typically possess the necessary domain knowledge, surpassing that of LLMs, but need the right cues for extraction of the missing concepts from long-term memory.

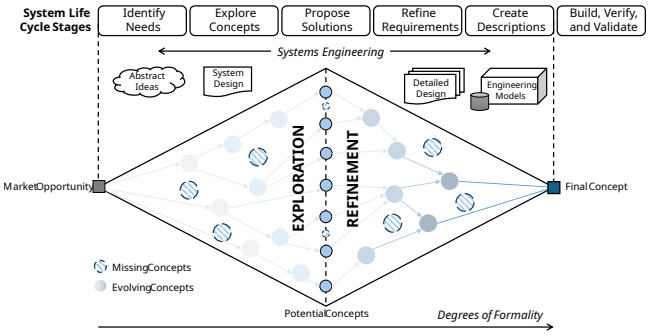

Fig. 2. Systems engineering development lifecycle [31] according to ISO/IEC/IEEE 24748-1:2024 with concept exploration and refinement.

The present work hypothesizes that identifying system constructs that are not immediately apparent or require extended cognitive processing can enhance the quality of system models in terms of their completeness and resolution of details. To achieve this, the work exemplifies a methodology for discovering concepts linked to the words and phrases used in models through a set of predefined semantic relationships.

To evaluate the approach, the work implements a prototype that extracts semantically related terms based on the proposed methodology and visualizes them as a word cloud. The vali-

dation process applies the prototype to existing Systemigrams from the literature to assess its effectiveness in identifying additional variables and uncovering internal relationships that the original Systemigram may have omitted. The results are then compared with the conclusions drawn in the respective papers to determine whether the tool could have contributed to a more comprehensive representation.

II. Methodology

Engineering design decisions demand reproducibility, interpretability, and predictability, especially in safety-critical systems [32]. Models based on distributional semantics struggle to achieve these qualities when trained on text corpora that lack system-specific context [33] or when the distributional hypothesis fails to represent multidimensional relationships like attributes [34], as opposed to broad taxonomic categories.

This work addresses these challenges by leveraging a semantic network that explicitly encodes diverse lexical relationships between concepts. This approach offers several advantages. Engineers routinely use structured information models in software and systems engineering, and many open-source knowledge bases (e.g., Cyc, DBPedia, WordNet, ConceptNet, Wikipedia) and industry standards (e.g., DNV reference structures, ISO standards) provide widely accessible structured vocabularies. Integrating these prebuilt resources enhances knowledge reuse across projects within organizations, improving both efficiency and accuracy in engineering applications.

Semantic networks interconnect concepts through relationships such as hyponymy (specific-to-general), hypernymy (general-to-specific), meronymy (part-to-whole), holonymy (whole-to-part), synonymy (similar meanings), and antonymy (opposite meanings). Among these, hypernym-hyponym and meronym-holonym relationships play a crucial role in forming subsumptive and compositional hierarchical structures, which are essential for organizing knowledge. These relationships are similar to inheritance and composition mechanisms in object-oriented programming, where objects inherit properties from general classes and are composed of various parts [35].

Figure 3 illustrates semantic entities and their relationships, structured along the axes of compositionality and specificity. Using structural isA and hasA relations, a concept moves vertically to represent whole-part relationships or horizontally to capture generic-specific associations. While these structures inherently exist in systems engineering through principles and heuristics, they lack formalization in guiding how semantic information should be extracted and applied. This work proposes a graph traversal mechanism that supports exploration and refinement based on a systems engineer's cognitive processes.

A. Concept Exploration

During the concept exploration phase, systems engineers use divergent thinking to identify high-level, generic concepts positioned higher in the hierarchy of specificity while also considering important and potentially critical details of those concepts. As Kossiakoff et al. describe, this phase ensures all possible alternative courses of action are considered before committing to development [36]. From a cognitive perspective, this process relies on working memory to manipulate relevant concepts while drawing prior knowledge about related systems, attributes, risks, and constraints from long-term memory.

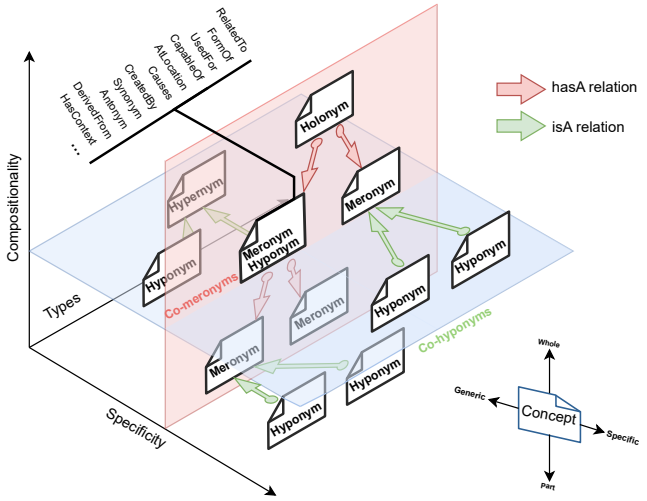

Fig. 3. The organization of the concepts in a semantic network based on their degrees of compositionality and specificity.

For example, if a stakeholder requires a cost-effective cargo transport solution, the systems engineer ideally begins by instantiating broad categories such as "merchant ships," "cargo aircraft," or "motor vehicles." These co-hyponyms under the hypernym "vehicles" help in activating stored knowledge and triggering associative retrieval of relevant subsystems and dependencies. The systems engineer may already know that "transport cost depends on fuel efficiency" and that "merchant ships have lower fuel efficiency than motor vehicles." However, to reinforce this connection, the "fuel system" must be explicitly surfaced in working memory to guide decision-making. Alternatively, the systems engineer may adopt a bottom-up approach, beginning with specific technologies such as "electric motors" or "gas turbines" rather than broad functional categories. In this case, identifying related hyponyms expands contextual understanding, connecting low-level technologies to higher-level system concepts. The efficiency of retrieval depends on semantic proximity and associative strength within the engineer's mental model, enabling the adaptation of domain knowledge to novel design challenges.

Attentional limitations in working memory can cause systems engineers to overlook critical system attributes. Certain features may be well-known in one domain but absent in another. For example, "airbags" are a safety feature in motor vehicles but are not conventionally associated with "merchant ships." However, recognizing this concept "airbag" can trigger engineers to consider analogous safety mechanisms in maritime transport. Such context-dependent memory retrieval enhances the breadth of conceptual search by prompting engineers to explore not only hierarchical relationships but also structurally similar concepts across domains. Thus, the systems engineer must maintain situational awareness

of holonyms (e.g., "transportation system"), meronyms (e.g., "vehicle"), system locations (e.g., "ports"), co-hyponyms (e.g., "oil tanker"), co-meronyms (e.g., "rail freight"), and other related concepts (e.g., "ferry").

To operationalize this process, a set of relationships and an exploration function is defined that systematically traverses a structured semantic network to retrieve concepts and strength of relations. The set of relationships $\mathcal{R}$ are defined as:

$$\mathcal{R}^k = \big(R_1^k \cup R_2^k \cup R_3^k(R_c^k) \cup R_4^k(R_c^k) \cup R_5(R_s^k) \cup \ldots \\ \ldots \cup R_8(R_s^k)\big) \ where\ k \in \mathbb{N}, k \geq 1 \quad (1)$$

where:

- R_1^k represents the holonym applied k times,
- R_2^k represents the meronym applied k times,
- R_3^k, R_4^k represent additional subsumptive relations (e.g., hyponyms, hypernyms) applied k times,
- $R_c^k = R_1^k \cup R_2^k$ and $R_s = R_c^k \cup R_3^k \cup R_4^k$,
- R_5, R_6, R_7, R_8 represent the non-hierarchical relations: polysemy, retronym, troponym, and synonym, each applied once.

The exploration function $P(W)$ for a word W is given by:

$$P(W) = \bigcup_{k=1}^{n} \Big\{(X, y) \mid X \in \mathcal{R}^k(W), X, W \in \mathcal{L} \\ y = \prod_{j=1}^{k} S(\mathcal{R}^j), S(\mathcal{R}) \in [0, 1]\Big\} \quad (2)$$

where:

- X is the set of words related to W by the relations $\mathcal{R}$ applied k times and $\mathcal{L}$ is the set of all linguistic terms,
- y is the product of the relatedness scores S of all relations between words from W to X up to k-th iteration,

The relatedness scores S between two words is determined by the number of semantic traversal steps, with each step assigned a weighted value reflecting the strength of the relationship. These weights can be either manually annotated by contributors or measured using semantic similarity methods to build a knowledge base [37] of fundamental linguistic entities.

B. Concept Refinement

Once potential concepts are identified during the exploration phase, the systems engineer must refine these concepts by introducing specific details and defining the attributes necessary for system implementation. This phase, also known as concept definition, ensures that the concepts are feasible, clear, and traceable throughout the project lifecycle, particularly during the implementation. It requires adding detailed specifications for system components using quantitative elements that will allow for verification and validation during system testing.

For example, if the selected concept is a "merchant ship", the refinement process must detail the specifications for the "engine", including "fuel capacity" and the specific "fuel types" to be used. This process involves traversing through meronymic and hyponymic relationships to precisely define the characteristics of each component. In this case, the "engine" is part of the larger system, that includes other components like "engine monitoring systems," "gearbox," "multifunction monitoring system," and "report and alarm system."

The systems engineer must also consider functions, causes, and quality attributes of the system components, ensuring that these parameters are not only defined but can also be extracted for documentation or incorporated into engineering and production drawings for implementation. For example, it might be necessary to specify that the vessel operates effectively under 40 knots of wind, and that the towing carriage must maintain an accuracy of 0.03°. Furthermore, environmental factors, such as potential obstructions near the "starboard" side at the "dockyard", must be included in the refinement to ensure safety and operational feasibility.

This process involves a systematic breakdown of high-level concepts into concrete, verifiable components, which can be expressed attributes, operational constraints, and environmental considerations for precise implementation according to the stakeholders' specifications. Based on this description, the set of relations $\mathcal{R}$ for the refinement equation can be defined as:

$$\mathcal{R}^k = \big(R_1^k \cup R_2^k(R_1^k) \cup R_3^k(R_2^k(R_1^k)) \cup R_4^k(R_d^k) \cup \ldots \\ \ldots \cup R_9(R_d^k)\big) \ where\ k \in \mathbb{N}, k \geq 1 \quad (3)$$

where:

- R_1^k represents the meronym applied k times,
- $R_2^k(R_1^k)$ represents the hyponyms of R_1^k applied k times,
- $R_3^k(R_2^k(R_1^k))$ represents the meronyms of $R_2^k(R_1^k)$ applied k times,
- $R_d^k = R_1^k \cup R_2^k(R_1^k) \cup R_3(R_2^k(R_1^k))$,
- $R_4, R_5, R_6, R_7, R_8, R_9$ represent function, cause, property, context, capability, and qualities, each applied once.

The refinement function $F(W)$ for a word W is given by:

$$F(W) = \bigcup_{k=1}^{n} \Big\{(X, y) \mid X \in \mathcal{R}^k(W), X, W \in \mathcal{L} \\ y = S(\mathcal{R}(W, X)), S(\mathcal{R}) \in [0, 1]\Big\} \quad (4)$$

where:

- X is the set of words related to W by the relations $\mathcal{R}$ applied k times and $\mathcal{L}$ is the set of all linguistic terms,
- y is the relatedness score of relations directly between words W and X,

The stop distance n for both the functions is determined using the stopping criteria for graph traversal, where the cost of further exploration outweighs the decrease in the relatedness score y, or until the concepts in the knowledge base are fully explored — whichever occurs first.

III. Validation

While the proposed approach offers potential benefits such as improved time efficiency, cognitive ease, productivity, and system quality, the present work limits the evaluation to the two specific characteristics of system model: completeness and resolution. The evaluation was conducted through a prototype implementation and secondary data analysis of 17 publications where Systemigrams were used as a method.

A. Prototype Implementation

The prototype was implemented as a plugin for Draw.io, an open-source, cross-platform diagramming tool with a web-based interface for creating UML diagrams, flowcharts, wireframes, and other visual models. Draw.io is widely used for organizing and visualizing information due to its minimal yet powerful functionality and high interoperability with various machine-readable exchange formats.

The semantic network ConceptNet, a commonsense knowledge graph built from the contributions of over 15,000 contributors, served as the knowledge base [37]. ConceptNet represents knowledge as a directed graph, where nodes correspond to concepts, and edges define semantic relationships between them. The edge weights indicate assertion strength, determined from a combination of manually assigned values and diversity of the sources. ConceptNet provides a public API that returns JavaScript Object Notation for Linking Data for queried terms.

When a user selects a word or phrase in the diagramming space, the plugin accesses the text and the selected mode (”exploration” or ”refinement”) from the Extensible Markup Language (XML) configuration in the backend. It then iteratively queries the ConceptNet API to retrieve related concepts and their associated weights. The query results are structured based on directional constraints before being displayed as a word cloud in Draw.io's user interface with the size, position, and color representing the strength, type, and connectivity of relationships between concepts. By clicking on any term, users are redirected to a thesaurus to learn about the definitions and example sentences. The interface dynamically updates based on user interactions to provide real-time access to relevant semantic associations where user can iterate based on their preference of either waterfall or agile development approach.

B. Data Collection

The methodology is validated through secondary data analysis, using system models collected from existing literature. This study focuses on a specific type of system model called Systemigrams, a soft systems modeling approach that relies on natural language for informally organizing ideas to highlight relationships and dependencies. These models are relevant to the present study, as they facilitate conceptual modeling and early-stage discussions by representing complex systems through key concepts, actors, events, patterns, and processes.

As shown in Table I, a total of 25 Systemigrams from 17 articles were analyzed, covering engineering and socio-technical development across the maritime, defense, energy, transportation, manufacturing, and education industries. The articles were selected based on the criterion that the Systemigram was explicitly provided as a figure in the publication. Each Systemigram was recreated in Draw.io, where its words and phrases were processed using the prototype. The evaluation measured two key aspects: expansion and upscaling. Expansion was assessed by counting entities identified by the prototype that were important to the study's conclusions but absent in the original Systemigram. Upscaling was evaluated by counting additional entities that could be inserted between existing nodes to improve clarity and reduce ambiguity. The counting was followed by manual curation by the authors from the list of generated concepts by the prototype based on subjective appreciation. Both the modes—exploration and refinement—were used iteratively to generate the relevant concepts. These metrics were then normalized relative to the size of each Systemigram (i.e., the total number of entities) to reveal the impact relative to the model quality.

TABLE I
SYSTEMIGRAMS BASED ON INDUSTRY COLLECTED FROM LITERATURE.

Sr. No.	Research Focus	No. of Models	Industry
1	Analysing and Simulating Manufacturing Systems with Systems Methodology [38]	1	Manufacturing
2	Communicating Strategic Intent for the Network-Enabled Challenge [39]	1	Defence
3	Resilience Study in Maritime Transportation System of Systems [40]	2	Maritime
4	Understanding Plagiarism Using Boardman's Soft Systems Methodology [41]	1	Education
5	Defining the Body of Knowledge and Curriculum to Advance Systems Engineering (BKCASE) Project [42]	1	Education
6	Problem Definition of Maritime Resilience for Homeland Security [43]	1	Maritime
7	Modeling Small Vessel Security Strategy for Developing Enterprise Resilience [44]	1	Maritime
8	Identifying Emergence in Complex Adaptive Systems [45]	6	Defence
9	Impact of Electronic Governance on Sustainable Development [46]	1	Manufacturing
10	eHealth Marketing to Millennials: A View Through A Systemigram [47]	1	Education
11	Systems Thinking for Early Validation of User Needs in the Front End of Innovation; a Case Study in Offshore SoS [48]	1	Maritime
12	Framing the Problem in a Subsea Development Projects with High-Level Business Requirements [49]	1	Maritime
13	Framing and Exploring a Test System for Product Verification in Large Defence Projects [50]	2	Defence
14	Conducting PESTEL Analysis of An Offshore Windfarm System [51]	1	Energy
15	Data Analysis for Automated Parking System [52]	2	Transportation
16	Early Validation of An Automated Parking System [53]	1	Transportation
17	Data-Driven Product Development [54]	1	Manufacturing

The expansion, average upscaling per edge, and graph sparsity of a Systemigram $\mathcal{G}$, with the set of nodes $\mathcal{N}$, and edges $\mathcal{E}$, can be formally defined as:

$$\text{expansion}(G) = \frac{n(\mathcal{N}) - n(\mathcal{N}_1)}{n(\mathcal{N})} \tag{5}$$

$$\text{upscaling}(G) = \frac{n(\mathcal{N}) - n(\mathcal{N}_2)}{n(\mathcal{N}) \times n(\mathcal{E})} \tag{6}$$

$$\text{sparsity}(G) = \frac{n(\mathcal{N})}{n(\mathcal{E})} \tag{7}$$

Where:

- $\mathcal{N}_1$ is the set of new nodes that expand the scope of the Systemigram to make it comprehensive and reliable.
- $\mathcal{N}_2$ is the set of new nodes that refine the structure by adding intermediary nodes along existing edges to ensure that it remains unambiguous during the implementation.
- $n(\mathcal{N})$ and $n(\mathcal{E})$ denote the number of nodes and edges.

IV. Results and Discussion

Applying the prototype to the Systemigrams showed that, on average, the models could incorporate 24.15% more entities than those initially included. The prototype also enhanced edge detail by adding intermediate entities, resulting in a 15.5% overall increase, with an average of 2.98% more entities per edge. However, the relative standard deviations of 15.04% and 7.18% in the expansion and upscaling criteria indicate moderate variability. This suggests that the methodology lacks the precision required for consistent system development using complete and reliable system models across all applications.

Further analysis revealed that industry trends significantly influenced the suggestion of relevant concepts. For instance, the defense industry relies heavily on abbreviations and context-specific terminology that may not be represented in the ConceptNet knowledge graph. When excluding defense industry data and reevaluating the findings, the relative standard deviation decreased to 12.38% indicating higher precision.

Additionally, the graph sparsity (i.e., the number of nodes per edge) significantly impacted the method's effectiveness. Systemigrams with fewer entities but higher connectivity proved less effective for the proposed solution, as the entities in the graph are more descriptive, consisting primarily of nouns and concepts rather than verbs. This suggests that the methodology struggles to replicate the human cognitive ability to emphasize connections instead of listing entities, a skill highlighted by Frank [15]. Consequently, expert systems engineers may find this tool less useful than novice users.

A broader observation across all systemigrams was that the tool retrieved concepts related to social and environmental aspects more effectively than those associated with technical and engineering domains. While this presents challenges for engineering development, it enables solution-oriented engineers with silo mentality to broaden their perspective and consider the 'big picture.' This tool can be valuable in addressing larger global issues such as sustainability and child labour, although it is currently less effective for engineering-specific projects.

V. Conclusion

Systems engineering, as a discipline, has traditionally focused on technical challenges while scarcely exploring the complexities of human mind. As intelligent technologies continue to evolve, investigating solutions at the intersection of engineering and cognitive processes becomes increasingly important. This work presents one such approach to systems modeling, introducing a methodology that traverses a knowledge graph inspiring from systems engineering processes. The proposed method aims to reduce cognitive burden during concept development by assisting with information retrieval and organization. A prototype implementation, validated using secondary data from literature, provided valuable insights into the approach's performance and limitations. The findings emphasize the need for integration of structured ontologies and domain-specific documentation to improve the semantic accuracy in diagramming tools for enhanced model usability.

Future research should refine the methodology to better align with user cognition, focusing on how well the tool supports problem-solving and workflow efficiency. Further validation in real-world settings with industry professionals will be essential for refining usability and functionality. Enhancements such as collaborative tagging systems and retrieval-augmented generation will help tailor suggestions to individual fields, making the tool a more adaptive and scalable resource for technical and academic applications.

References

[1] C. J. Fong, E. A. Matranga, and D. J. DeTurris, "Failure from an engineering systems perspective," in *AIAA AVIATION FORUM AND ASCEND 2024*, 2024, p. 4901.

[2] D. McNab, J. McKay, S. Shorrock, S. Luty, and P. Bowie, "Development and application of 'systems thinking'principles for quality improvement," *BMJ open quality*, vol. 9, no. 1, p. e000714, 2020.

[3] R. D. Arnold and J. P. Wade, "A definition of systems thinking: A systems approach," *Procedia computer science*, vol. 44, pp. 669–678, 2015.

[4] R. Hummelbrunner, "Systems thinking and evaluation," *Evaluation*, vol. 17, no. 4, pp. 395–403, 2011.

[5] W. Hung, "Enhancing systems-thinking skills with modelling," *British Journal of Educational Technology*, vol. 39, no. 6, pp. 1099–1120, 2008.

[6] M. Ozkaya and F. Erata, "A survey on the practical use of uml for different software architecture viewpoints," *Information and Software Technology*, vol. 121, p. 106275, 2020.

[7] B. Sauser and J. Boardman, "Systemigram modeling for contextualizing complexity in system of systems," *Modeling and simulation support for system of systems engineering applications*, pp. 273–302, 2014.

[8] A. Akundi, W. Ankobiah, O. Mondragon, and S. Luna, "Perceptions and the extent of model-based systems engineering (mbse) use–an industry survey," in *2022 IEEE International Systems Conference (SysCon)*. IEEE, 2022, pp. 1–7.

[9] J. H. Brill, "Systems engineering—a retrospective view," *Systems Engineering: the Journal of the International Council on Systems Engineering*, vol. 1, no. 4, pp. 258–266, 1998.

[10] D. Wilke, S. A. Pfeifer, R. Heitmann, H. Anacker, R. Dumitrescu, and V. Franke, "Implementation of systems engineering: A maturity-based approach," in *2022 IEEE International Symposium on Systems Engineering (ISSE)*. IEEE, 2022, pp. 1–7.

[11] F. Dong, M. Li, J. Li, Y. Yang, and Q. Wang, "Effect of staffing pattern on software project: An empirical analysis," in *2009 3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE, 2009, pp. 23–33.

[12] T. M. Egan, "Creativity in the context of team diversity: Team leader perspectives," *Advances in developing human resources*, vol. 7, no. 2, pp. 207–225, 2005.

[13] S. Kavadias and S. C. Sommer, "The effects of problem structure and team diversity on brainstorming effectiveness," *Management Science*, vol. 55, no. 12, pp. 1899–1913, 2009.

[14] R. C. Atkinson, "Human memory: A proposed system and its control processes," *The psychology of learning and motivation*, vol. 2, 1968.

[15] M. Frank, "Engineering systems thinking: Cognitive competencies of successful systems engineers," *Procedia computer science*, vol. 8, pp. 273–278, 2012.

[16] I. P. Hernandez Sibo, D. A. Gomez Celis, and S. Liou, "Exploring the landscape of cognitive load in creative thinking: a systematic literature review," *Educational Psychology Review*, vol. 36, no. 1, p. 24, 2024.

[17] J. L. Redifer, C. L. Bae, and M. DeBusk-Lane, "Implicit theories, working memory, and cognitive load: Impacts on creative thinking," *Sage Open*, vol. 9, no. 1, p. 2158244019835919, 2019.

[18] A. R. Conway and R. W. Engle, "Working memory and retrieval: a resource-dependent inhibition model." *Journal of Experimental Psychology: General*, vol. 123, no. 4, p. 354, 1994.

[19] J. L. Schelble, D. J. Therriault, and M. D. Miller, "Classifying retrieval strategies as a function of working memory," *Memory & cognition*, vol. 40, pp. 218–230, 2012.

[20] J. Gwizdka, "What a difference a tag cloud makes: effects of tasks and cognitive abilities on search results interface use." *Information Research: An International Electronic Journal*, vol. 14, no. 4, p. n4, 2009.

[21] P. Marbach, "Agile systems engineering," *The Guide to the Systems Engineering Body of Knowledge (SEBoK)*, vol. 2, 2024.

[22] A.-R. Preda, C. Mayr-Dorn, A. Mashkoor, and A. Egyed, "Supporting high-level to low-level requirements coverage reviewing with large language models," in *Proceedings of the 21st International Conference on Mining Software Repositories*, 2024, pp. 242–253.

[23] J. Carter and C. Singam, "Model based systems engineering (mbse)," *The Guide to the Systems Engineering Body of Knowledge (SEBoK)*, vol. 2, 2024.

[24] A. Zaki-Ismail, M. Osama, M. Abdelrazek, J. Grundy, and A. Ibrahim, "Requirements formality levels analysis and transformation of formal notations into semi-formal and informal notations," in *International Conference on Software Engineering and Knowledge Engineering 2021*. KSI Research Inc., 2021, pp. 303–308.

[25] M. Cherubini, G. Venolia, R. DeLine, and A. J. Ko, "Let's go to the whiteboard: how and why software developers use drawings," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2007, pp. 557–566.

[26] T. McDermott, K. Pepe, and M. Clifford, "The updated serc ai and autonomy roadmap 2023," in *INCOSE International Symposium*, vol. 34, no. 1. Wiley Online Library, 2024, pp. 1135–1148.

[27] Y. Hu, Y. Goktas, D. D. Yellamati, and C. De Tassigny, "The use and misuse of pre-trained generative large language models in reliability engineering," in *2024 Annual Reliability and Maintainability Symposium (RAMS)*. IEEE, 2024, pp. 1–7.

[28] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang, "A survey on large language model (llm) security and privacy: The good, the bad, and the ugly," *High-Confidence Computing*, p. 100211, 2024.

[29] H. Que, F. Duan, L. He, Y. Mou, W. Zhou, J. Liu, W. Rong, Z. M. Wang, J. Yang, G. Zhang *et al.*, "Hellobench: Evaluating long text generation capabilities of large language models," *arXiv preprint arXiv:2409.16191*, 2024.

[30] T. Duenas and D. Ruiz, "The risks of human overreliance on large language models for critical thinking," *Research Gate, 2024e. URL http://dx. doi. org/10.13140/RG*, vol. 2, no. 26002.06082, 2024.

[31] D. Dutil, J. Rose, W. Suryn, and B. Thimot, "Software quality engineering in the new iso standard: Iso/iec 24748-systems and software engineering—guide for life cycle management," in *Proceedings of the Third C* Conference on Computer Science and Software Engineering*, 2010, pp. 89–96.

[32] H. Kuwajima, H. Yasuoka, and T. Nakae, "Open problems in engineering and quality assurance of safety critical machine learning systems," *arXiv preprint arXiv:1812.03057*, 2018.

[33] L. Yu, E. Alégroth, P. Chatzipetrou, and T. Gorschek, "Experience with large language model applications for information retrieval from enterprise proprietary data," in *International Conference on Product-Focused Software Process Improvement*. Springer, 2024, pp. 92–107.

[34] D. Rubinstein, E. Levi, R. Schwartz, and A. Rappoport, "How well do distributional models capture different types of semantic knowledge?" in *Proceedings of the 53rd annual meeting of the Association for Computational Linguistics and the 7th international joint conference on natural language processing (Volume 2: Short papers)*, 2015, pp. 726–730.

[35] D. Ratiu and F. Deissenboeck, "How programs represent reality (and how they don't)," in *2006 13th Working Conference on Reverse Engineering*. IEEE, 2006, pp. 83–92.

[36] A. Kossiakoff, W. N. Sweet, S. J. Seymour, and S. M. Biemer, *Systems engineering principles and practice*. John Wiley & Sons, 2011, vol. 83.

[37] R. Speer, J. Chin, and C. Havasi, "Conceptnet 5.5: An open multilingual graph of general knowledge," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.

[38] J. Bulbeck and B. Clegg, "A systems methodology for analysing and simulating manufacturing systems," in *1996 IEEE International Conference on Systems, Man and Cybernetics. Information Intelligence and Systems (Cat. No. 96CH35929)*, vol. 1. IEEE, 1996, pp. 778–783.

[39] C. D. Blair, J. T. Boardman, and B. J. Sauser, "Communicating strategic intent with systemigrams: Application to the network-enabled challenge," *Systems Engineering*, vol. 10, no. 4, pp. 309–322, 2007.

[40] M. Mansouri, B. Sauser, and J. Boardman, "Applications of systems thinking for resilience study in maritime transportation system of systems," in *2009 3rd Annual IEEE Systems Conference*. IEEE, 2009, pp. 211–217.

[41] S. Sivadasan and B. Sauser, "Understanding plagiarism using boardman's soft systems methodology," in *2009 Annual Conference & Exposition*, 2009, pp. 14–1296.

[42] A. Squires, A. Pyster, B. Sauser, D. Olwell, S. Enck, D. Gelosh, and J. Anthony, "6.1. 1 applying systems thinking via systemigrams™ for defining the body of knowledge and curriculum to advance systems engineering (bkcase) project," in *INCOSE International Symposium*, vol. 20, no. 1. Wiley Online Library, 2010, pp. 739–753.

[43] B. Sauser, M. Mansouri, and M. Omer, "Using systemigrams in problem definition: A case study in maritime resilience for homeland security," *Journal of Homeland Security and Emergency Management*, vol. 8, no. 1, p. 000010220154773551773, 2011.

[44] B. Sauser, Q. Li, and J. Ramirez-Marquez, "Systemigram modeling of the small vessel security strategy for developing enterprise resilience," *Marine Technology Society Journal*, vol. 45, no. 3, pp. 88–102, 2011.

[45] T. McDermott, M. Nadolski, and L. Sheppard, "Use of systemigrams to identify emergence in complex adaptive systems," in *INCOSE International Symposium*, vol. 25, no. 1. Wiley Online Library, 2015, pp. 582–599.

[46] N. Khansari, M. Finger, A. Mostashari, and M. Mansouri, "Conceptual systemigram model: Impact of electronic governance on sustainable development," *International Journal of System of Systems Engineering*, vol. 7, no. 4, pp. 258–276, 2016.

[47] G. Prybutok, A. Harun, and V. Prybutok, "ehealth marketing to millennials: a view through a systemigram," *International Journal of Electronic Healthcare*, vol. 9, no. 4, pp. 319–338, 2017.

[48] M. Kjørstad, M. Mansouri, G. Muller, and S. Kjenner, "Systems thinking for early validation of user needs in the front end of innovation; a case study in an offshore sos," in *2019 14th Annual Conference System of Systems Engineering (SoSE)*. IEEE, 2019, pp. 382–387.

[49] S. Engen, M. Mansouri, and G. Muller, "Application of system thinking to frame the problem in a subsea development projects with high-level business requirements," in *2019 14th Annual Conference System of Systems Engineering (SoSE)*. IEEE, 2019, pp. 81–86.

[50] R. A. Haugen and M. Mansouri, "Applying systems thinking to frame and explore a test system for product verification; a case study in large defence projects," in *INCOSE International Symposium*, vol. 30, no. 1. Wiley Online Library, 2020, pp. 78–93.

[51] Y. Chen and M. Mansouri, "Systemigrams for pestel analysis of an offshore windfarm system," in *The Seventeenth International Conference on Systems (ICONS)*. IARIA XPS Press Barcelona, 2022.

[52] H. Ali, G. Muller, M. Mansouri, F. A. Salim, and K. Falk, "Supporting systems thinking application by data analysis a case study: An automated parking system," *International Journal on Advances in Intelligent Systems*, 2022.

[53] H. Ali, M. Mansouri, and G. Muller, "Applying systems thinking for early validation of a case study definition: An automated parking system," in *MODERN SYSTEMS 2022: International Conference of Modern Systems Engineering Solutions*, 2022.

[54] T. Langen, K. Falk, and M. Mansouri, "A systems thinking approach to data-driven product development," *Proceedings of the Design Society*, vol. 2, pp. 1915–1924, 2022.