Proceedings of The 6th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/2026.31609

Exploration of AI Software Generation Methods Based on Work Flow Knowledge and Domain Constraints

Yuteng Fu

San Jose, USA
yutengfu1102@gmail.com

Abstract. In response to the challenges faced by large language models in generating complex industrial software, this paper proposes a neural-symbolic collaborative generation framework (WFC-Gen) that integrates workflow knowledge and domain constraints. In the constructed EW-Bench enterprise-level benchmark test, the research results confirm that introducing structured knowledge guidance is the inevitable path to achieving high-confidence, engineering-level software automatic generation.

Keywords: AI software generation, workflow knowledge, domain constraints, neural-symbolic AI, constraint-guided decoding

1. Introduction

Currently, a new round of technological revolution is sweeping across the globe, and the field of software engineering is also at a historical turning point of transformation [1]. The emergence of artificial intelligence, especially large language models (LLMs), has greatly shortened the development cycle for general functions [2], yet when faced with industrial-grade software systems with high complexity, strict logic, and specific domain regulations, their inherent defect of "probabilistic speculation" becomes evident [3].

To the core, the main bottleneck of current AI software generation technology lies in the "semantic gap" [4]. How to effectively integrate structured workflow knowledge and formal domain constraints into the generation model, enabling it to evolve from a "random parrot" to a "smart artisan" who understands business and abides by rules, has become a key issue that both the academic community and the industrial sector urgently need to solve.

In view of this, this paper is based on the deep waters of the intelligent transformation of software engineering, and proposes a new method for AI software generation that integrates workflow knowledge and domain constraints. This research aims to explore a practical technical path for building a trustworthy and controllable next-generation intelligent software development system through this mechanism.

2. Related work

As shown in Figure 1, LLMs based on the Transformer architecture, in existing research, mostly focus on enhancing the generation effect by increasing the model parameter size, optimizing the pre-

DOI: 10.54254/2755-2721/2026.31609

training objective, or introducing the thinking chain prompt engineering [5,6]. Compared with the "emotional intuition" of large models, traditional software engineering has long been committed to establishing the "rational order" of the system through formal methods. Workflow technology, as the digital carrier of business logic, has already formed a mature representation system including Petri nets, which can precisely depict the control flow, data flow, and resource flow in business processes [7,8]. However, previous research lacks an effective mechanism to deeply integrate these highly structured explicit knowledge with generative models with strong generalization ability.

With the exponential expansion of software scale, traditional solvers face a serious combinatorial explosion problem. In recent years, some scholars have attempted to introduce constraint checking into the neural generation process, by enhancing the reinforcement learning reward function or filtering constraints in the decoding stage to improve the generation quality [9]. Nevertheless, most existing methods lack the "guidance" ability to actively intervene at the generation source.

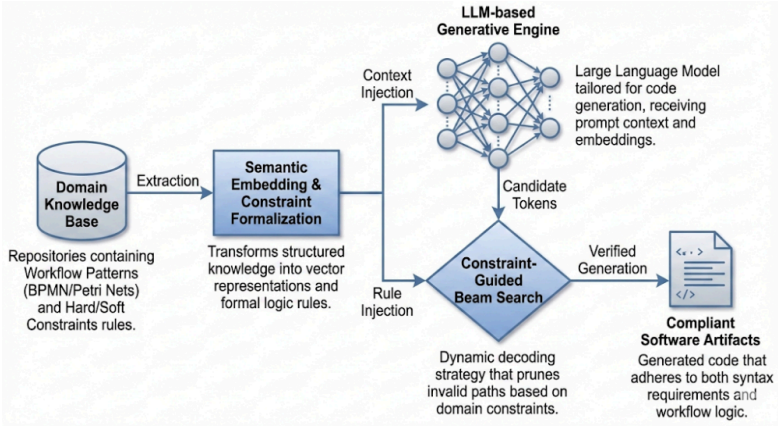

Figure 1. Core concept framework diagram of the system

3. Methodology

3.1. Formalization of workflow knowledge

To address the chronic problem of the general model's "blindness" towards the business temporal logic, the primary task is to map the unstructured requirements description into a structured graph representation. This study employs semantic-enhanced Petri nets as the carrier of workflow knowledge [10].

Define SAPN as a quadruple $W = \langle P, T, F, \Lambda \rangle$.

Here, the set of places represents the business state:

$$P = \{p_1, p_2, \ldots, p_m\} \tag{1}$$

Change set, representing specific functional functions or API calls:

$$T = \{t_1, t_2, \ldots, t_n\} \tag{2}$$

The directed arc set defines a strict causal flow logic:

$$F \subseteq (P \times T) \cup (T \times P) \tag{3}$$

The semantic mapping function is as follows:

DOI: 10.54254/2755-2721/2026.31609

$$\Lambda : P \cup T \to R^d \tag{4}$$

Using the pre-trained encoder, the discrete node symbols are mapped to high-dimensional semantic vectors v_{sem}, thereby endowing the graph structure with computable "semantic thickness".

Through this formalization process, the business logic is no longer a set of discrete text fragments, but transformed into a mathematical object with topological constraints, providing a rigid "framework" for the subsequent generation process.

3.2. Domain constraint injection mechanism

The injection of domain constraints is not a simple concatenation of rules, but a multimodal alignment process. We orthogonalize the constraint $\mathscr{C}$ into hard constraints ($\mathscr{C}$) and soft constraints ($\mathscr{C}$) [11].

The hard constraints can be expressed as:

$$\forall x \in \text{Code}, \text{Call}(x, \text{Pay}) \to \exists y \in \text{Pre}(x), \text{State}(y) = \text{Auth\_Verified} \tag{5}$$

The soft constraints fuse the workflow semantic vector v_{sem} and the constraint embedding vector v_{const} through the multi-head attention mechanism to construct a context-aware guiding vector:

$$h_{guide} = \text{Attention}(Q = W_q v_{sem}, K = W_k v_{const}, V = W_v v_{const}) \tag{6}$$

This vector will be injected as a prompt prefix into the large model, thereby pre-calibrating the generation direction in the hidden layer space.

3.3. Constraint-guided decoding algorithm

This is the "heart" of this methodology. Traditional large models use greedy search, which is prone to "logical drift". This study proposes a constraint-aware beam search algorithm, which implements dynamic intervention at each time step of decoding [12].

Let y_t be the generated Token at time t, and $\mathscr{V}$ be the vocabulary. The standard generation probability distribution is:

$$P_{LLM}(y_t y_{<t}, \mathscr{W}) \tag{7}$$

The CABS algorithm introduces a discriminant function [13]:

$$\Phi(y_t, \mathscr{C}) \tag{8}$$

When the candidate Token causes a violation of the hard constraint, this function returns $-\infty$; otherwise, it returns 0.

The logarithmic form of the corrected generation probability is as follows:

$$logP^*(y_t) = logP_{LLM}(y_t y_{<t}, h_{guide}) + \lambda \cdot \Phi(y_t, \mathscr{C}) \tag{9}$$

DOI: 10.54254/2755-2721/2026.31609

The first item is predicted by the large model based on semantic guided vectors. The second item is a hard constraint penalty term, which actually plays the role of a "logical gate". When Φ is negative infinity, the probability of the violated path collapses instantly to zero, thereby being forcibly pruned in the search tree.

The overall architecture is shown in Figure 2.

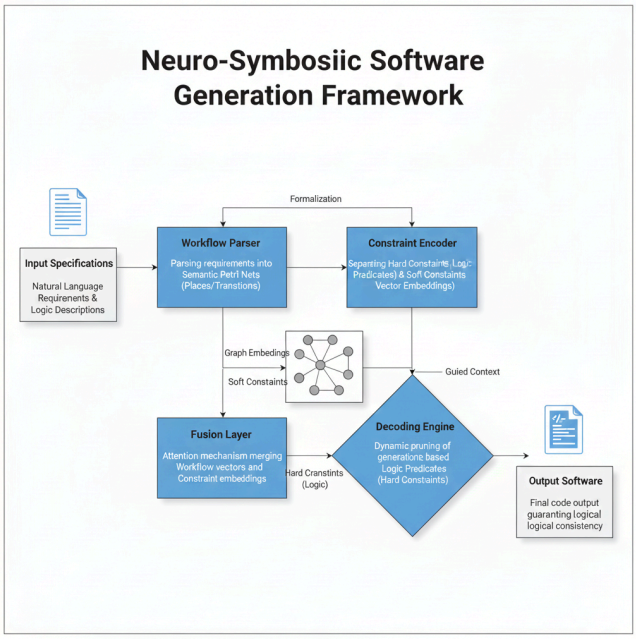

Figure 2. System flowchart

4. Implementation and experiments

4.1. Experimental setup

The original data source for the financial end adopts the BankScope bank core business dataset; for the supply chain end, the supplychainlarge dataset [14] is selected. The dataset contains 800 independent business scenarios, divided into two subsets: EW-Fin and EW-SCM.

This paper selects three representative models as the baseline for horizontal evaluation: General LLMs, GPT-4-Turbo, and CodeLlama-34B-Instruct.

The prompt engineering method introduces the thinking chain prompt to induce the model to generate code step by step, which is used to compare the differences between "explicit constraint guidance" and "implicit thinking reasoning" in this study. The retrieval enhancement method is based on the traditional RAG architecture of vector retrieval, and retrieves the Top-k similar code snippets as the context. This baseline is used to verify the superiority of "structured knowledge injection" in this study compared to "unstructured text retrieval".

Evaluation metrics are as follows:

Pass@1: The proportion of generated code passing all unit test cases. This is the basic indicator for measuring whether the code can run.

Constraint Satisfaction Rate(CSR):

$$CSR = \frac{1}{N} \sum_{i=1}^{N} I\left(Check\left(c_i, \mathscr{C}\right)\right) \tag{10}$$

DOI: 10.54254/2755-2721/2026.31609

Among them, "Check" is the verification function of the static analyzer for hard constraints. This indicator directly reflects whether the model is "conforming to the rules".

Workflow Consistency Score (WCS): By parsing the abstract syntax tree to generate the control flow graph of the code and calculating the proportion of isomorphic subgraphs of the graph of the input Petri net. The score range is [0, 100], measuring whether the code logic is "deviating".

Cyclomatic Complexity: Evaluates the maintainability of the generated code, avoiding the generation of overly complex "noodle code".

4.2. Overall performance comparison

Table 1 shows the comprehensive performance of the proposed scheme (WFC-Gen) and each baseline model on EW-Bench. This provides strong evidence that formalizing hard constraints as logical predicates and integrating them into the decoding process can effectively curb the "probabilistic illusion" of large models.

Table 1. Main performance comparison on EW-Bench

Model	Sub-Dataset	Pass@1 (%)	CSR(%)	WCS
GPT-4-Turbo	EW-Fin	68.2± 1.5	61.4± 2.1	74.5
	EW-SCM	64.7± 1.8	58.9± 2.4	70.2
CodeLlama-34B	EW-Fin	56.5± 2.2	52.3± 3.0	63.8
	EW-SCM	53.1± 2.5	48.7± 3.2	60.1
RAG-Code	EW-Fin	70.1± 1.2	71.5± 1.8	78.2
	EW-SCM	67.4± 1.4	68.2± 2.0	75.6
WFC-Gen	EW-Fin	81.3± 0.9	97.2± 0.5	95.4
	EW-SCM	78.6± 1.1	95.8± 0.8	93.7

4.3. Fine-grained constraint analysis

To further investigate the performance of the model under different types of constraints, we decomposed the CSR metric into two dimensions: "hard constraints" and "soft constraints" for in-depth comparison. In the "hard constraints" scenario, WFC-Gen achieved a significant improvement; for the "soft constraints" related to style, WFC-Gen also showed a stable improvement, but the magnitude was smaller than that of the hard constraints.

Table 2. Performance breakdown by constraint type

Constraint Category	GPT-4 CSR (%)	WFC-Gen CSR (%)	Improvement (Δ)
Hard: Sequential Logic	65.2	98.5	+33.3%
Hard: Data Atomicity	58.7	96.2	+37.5%
Soft: Naming Convention	88.4	92.1	+3.7%
Soft: Design Pattern	76.5	89.3	+12.8%

DOI: 10.54254/2755-2721/2026.31609

4.4. Ablation experiment

The ablation experiment is designed to quantify the individual contributions of each core component and confirm the minimum completeness of the system architecture. The results are shown in Table 3.

Table 3. Ablation experiment

Configuration	Pass@1 (Δ)	CSR (Δ)	Analysis Focus
Full Mode	81.3%	97.2%	Baseline complete state
w/o Constraint Guidance	73.5%	74.1%	Compliance collapse
w/o Workflow Embedding	69.8%	89.4%	Context confusion
w/o Both (Base LLM)	56.5%	52.3%	Degradation

As shown intuitively in Figure 3, constraint guidance is the decisive factor for ensuring CSR and serves as the "brake pad" of the system. Workflow embedding is the core driving force for improving Pass@1, and acts as the "navigation instrument" of the system. Both are indispensable and together form the "dual engines" of WFC-Gen for handling the generation of complex industrial software.

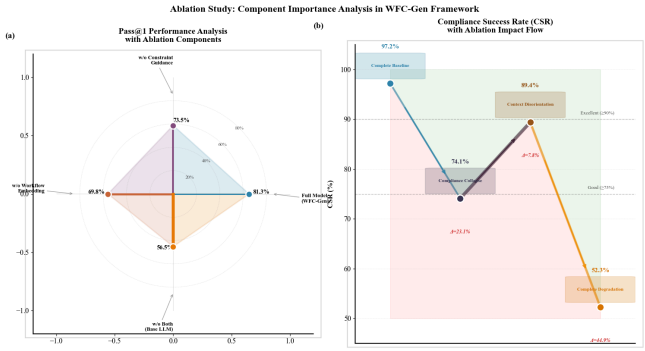

Figure 3. Ablation experiment

5. Conclusion

This paper addresses the pain points of large models in the generation of complex industrial software, and innovatively constructs a neural-symbolic collaborative generation framework. By "softly injecting" explicit business knowledge into the context and "hardly gating" implicit logical rules at the decoding end, we successfully curbed the random entropy increase of generative AI and achieved a paradigm shift from probabilistic statistical fitting to logical confirmatory reasoning. This research not only confirms the decisive role of "knowledge guidance" in enhancing software credibility, but also provides a solid practical paradigm for exploring the deep waters of artificial intelligence's transition from "assisting programming" to "autonomous engineering".

References

[1] Pezzè, M., Abrahão, S., Penzenstadler, B., Poshyvanyk, D., Roychoudhury, A., & Yue, T. (2025).A 2030 Roadmap for Software Engineering. ACM Transactions on Software Engineering andMethodology, 34(5), 1-55.

[2] Han, S., Wang, M., Zhang, J., Li, D., & Duan, J. (2024). A review of large language models: Fundamental architectures, key technological evolutions, interdisciplinary technologiesintegration, optimization and compression

DOI: 10.54254/2755-2721/2026.31609

techniques, applications, and challenges.Electronics, 13(24), 5040.
[3] Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., ... & Xie, X. (2024). A survey onevaluation of large language models. ACM transactions on intelligent systems and technology, 15(3), 1-45.
[4] Gao, C., Hu, X., Gao, S., Xia, X., & Jin, Z. (2025). The current challenges of software engineeringin the era of large language models. ACM Transactions on Software Engineering andMethodology, 34(5), 1-30.
[5] Long, S., Tan, J., Mao, B., Tang, F., Li, Y., Zhao, M., & Kato, N. (2025). A survey on intelligentnetwork operations and performance optimization based on large language models. IEEECommunications Surveys & Tutorials.
[6] Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., ... & Hu, X. (2024). Harnessing the powerof llms in practice: A survey on chatgpt and beyond. ACM Transactions on KnowledgeDiscovery from Data, 18(6), 1-32.
[7] Medina-Garcia, S., Medina-Marin, J., Montaño-Arango, O., Gonzalez-Hernandez, M., &Hernandez-Gress, E. S. (2023). A Petri net approach for business process modeling andsimulation. Applied Sciences, 13(20), 11192.
[8] Fajardo, S., Kleijn, J., Takes, F. W., & Langejans, G. H. (2022). Modelling and measuringcomplexity of traditional and ancient technologies using Petri nets. Plos one, 17(11), e0278310.
[9] Le, H., Wang, Y., Gotmare, A. D., Savarese, S., & Hoi, S. C. H. (2022). Coderl: Mastering codegeneration through pretrained models and deep reinforcement learning. Advances in NeuralInformation Processing Systems, 35, 21314-21328.
[10] Fan, Y., & Zhou, B. (2025, September). Multifeature fusion remote sensing workflowrecommendation method based on knowledge graph. In Second International Conference onRemote Sensing Technology and Survey Map** (RSTSM 2025) (Vol. 13802, pp. 122-128).SPIE.
[11] Zhan, Y., Yang, R., You, J., Huang, M., Liu, W., & Liu, X. (2025). A systematic literature reviewon incomplete multimodal learning: techniques and challenges. Systems Science & ControlEngineering, 13(1), 2467083.
[12] Qin, Z., He, Z., Prakriya, N., Cong, J., & Sun, Y. (2025, April). Dynamic-width speculative beamdecoding for llm inference. In Proceedings of the AAAI Conference on Artificial Intelligence(Vol. 39, No. 23, pp. 25056-25064).
[13] Wei, C., Koo, K. K., Tavanaei, A., & Bouyarmane, K. (2024). Confidence-aware sub-structurebeam search (cabs): Mitigating hallucination in structured data generation with large languagemodels. arXiv preprint arXiv: 2406.00069.
[14] Wang, J., Liu, G., Cheng, Y., Xu, X., & Li, Z. (2025). Leveraging internet-sourced text data forfinancial analytics in supply chain finance: A large language model-enhanced text miningworkflow. IEEE Transactions on Engineering Management.