LLM-powered Cyber-Physical System Features Extraction

Guillaume Nguyen[1,*], Xavier Devroey[1]

[1]NADI, University of Namur, rue de Bruxelles 51, Namur, 5000, Belgium

Abstract

Compliance assessment of Cyber-Physical Systems (CPSs) with legal requirements remains a complex and critical task in software engineering. Current approaches often lack traceability between source code, requirements, and legal basis. Furthermore, the abstraction gap makes the task challenging and requires mastery of the different abstraction levels. In this abstract, we propose a semi-automated method that bridges that gap by leveraging Large Language Models (LLMs) to extract features from CPS source code using a Feature Model specified in the Universal Variability Language (UVL). Our approach shows the promise of LLMs for traceable compliance verification while identifying key challenges in domain-specific expertise and legal-technical alignment.

Keywords

Verification, Requirements Engineering, Compliance, Legal, Regulatory, LLM, Feature Model, CPS

Introduction Cyber-Physical Systems (CPS) are ubiquitous; they can be found in many application domains, such as healthcare, transport, robotics, etc. Furthermore, they can be used in life-critical scenarios where a system failure can lead to human death (e.g., an automatic brake failure in an Advanced Driver Assistance System or ADAS). To ensure the safety, security, and overall quality of those products, manufacturers perform Requirements Engineering (RE) tasks to provide a list of requirements that engineers need to consider when building a new CPS [1]. Systems Under Conformity Assessment (SUCAs), such as the software products under the European Conformity (CE) marking (e.g., medical devices, ADAS, etc.), need to satisfy a set of legal requirements to be allowed on the market. However, Nguyen et al. [2] showed that such an assessment was entirely based on the provided documentation of a SUCA. When considering software products, our approach could be integrated into the approval process to help non-technical assessors when reviewing products. Thus, the assessor wouldn't need to 'trust' the coupling of the provided documentation with the actual product.

Models can be used to verify the alignment between the requirements and a system. Indeed, Yue et al. performed an SLR showing that analysis models are widely used in requirements engineering to go from text in Natural Language (NL) to an actual representation of the desired system [3]. Saleem et al show the potential strengths and limitations of using LLM for Requirement Engineering (RE) tasks and compared the performance of different LLMs against four benchmark datasets [4]. Their study showed that the performance of the tested LLM increased with the level of expert knowledge provided in the prompt. Another essential trait of RE is the *traceability* between requirements. Nowadays, multiple model-driven engineering techniques allow modeling the complexity of real-time, embedded, and cyber-physical systems, such as UML, MARTE, and SysML [5]. However, those models remain extremely complex and are used to guide such systems' development and maintenance. Indeed, using it to represent an existing, not yet modelled system could prove to be challenging.

Approach Our approach brings various fields together to leverage valuable source code information in verifying the requirements' actual implementation. We realised that using features to describe source code could alleviate the requirements verification effort by gaining a technology-agnostic level of abstraction that could still be linked to source code. Fortunately, CPS source code (i.e., software) is often embedded into small computer systems and needs to be limited in size. However, LLMs' stochastic,

BNAIC BeNeLearn 2025: The 37th Benelux Conference on Artificial Intelligence and the 34th Belgian Dutch Conference on Machine Learning, November 19, 2025, Namur, BE

*Corresponding author.

guillaume.nguyen@unamur.be (G. Nguyen); xavier.devroey@unamur.be (X. Devroey)

0000-0002-9724-6634 (G. Nguyen); 0000-0002-0831-7606 (X. Devroey)

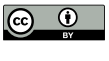

Table 1
Summary of the results from the various LLMs for extracting features from the Patient Monitor app

Company/Model	gold	pred	matched	prec.	recall	f1 score
meta-llama/Llama-4-Scout-17B-16E-Instruct	8	8	3	0,38	0,38	0,38
Qwen/Qwen3-32B	8	10	5	0,5	0,62	0,56
deepseek-ai/DeepSeek-R1-Distill-Llama-70B	8	17	4	0,24	0,5	0,32
NousResearch/Hermes-3-Llama-3.1-70B	8	11	5	0,45	0,62	0,53
google/gemma-3-27b-it	8	12	3	0,25	0,38	0,3
meta-llama/Llama-3.3-70B-Instruct	8	10	5	0,5	0,62	0,56
mistralai/Mistral-Small-24B-Instruct-2501	8	11	4	0,36	0,5	0,42
mistralai/Mixtral-8x7B-Instruct-v0.1	8	9	4	0,44	0,5	0,47
Average	8	11	4,125	0,39	0,515	0,4425

non-deterministic, and prompt-dependent characteristics lead to similar feature identification and generation characteristics [6]. To correct this problem, we can easily play on various parameters such as the *temperature*, the *top-k sampling*, and the *top-p sampling* [7]. Here, we want the most deterministic outcome, so we will set the *temperature* to 0 and use a domain-specific feature model to prompt the LLM to map the features rather than generating them.

Preliminary evaluation Following previous work in assessing the compliance of a medical device as an Android application [8] and by extending the approach to any types of static resource (source code, etc.) from various programming languages [9], we decided to test the potential of our approach by analysing eight open-source medical devices. For example, the **ESP8266 Patient Monitor**[1] (2022) enables pulse and temperature monitoring via ESP8266 and Arduino modules, with data transmission to an IoT platform (Thingspeak) [10]. For this device, we created an 8-feature gold standard by reviewing the project's documentation and completed it by manually analysing the code. Then, we compared the performance of 8 popular LLMs on HuggingFace as shown in Table 1.

The prompt was embedded with a medical-device-specific Feature Model, the computer code was enhanced with numbers at the beginning of the lines, the instruction to use features from the model before generating a new one, and we provided the expected JSON structure of the response. We computed a Jaccard index for establishing if a model correctly predicted a feature, and since all features were decomposed at three levels (e.g., <Sensors; Heart Rate; Optical Pulse Sensor>), we used a threshold of .66 for validating a feature after solving a Hungarian matrix to ensure a 1:1 matching between the predicted features and the gold features. Since we aim for an exact feature identification and provide the context to the LLM, we compared the precise matching of the features. As we can see, while LLMs manage to find some features, they still hallucinate in most cases. When looking more carefully at the generated features, we noticed that while our effort to reduce indeterminism might have helped, it was not enough to force the LLM first to use the features from the provided Feature Model. For example, we had < *Output; LED; Heart Rate* > in the gold standard and < *Actuators; LED; Heart Rate Indicator* > for the prediction from Gemma 3, which technically points to the same feature while not expressed as in the provided Feature Model.

Future Work We intend to fine-tune a model with datasets from the variability community for mapping source code to features and revise our prompting strategy. We also intend to push our evaluation further by analysing more devices and building a more complete gold standard for evaluation.

Acknowledgments

This research was supported by the CyberExcellence by DigitalWallonia project (No. 2110186), funded by the Public Service of Wallonia (SPW Recherche).

[1] https://electronicsworkshops.com/2022/11/14/patient-health-monitoring-using-esp8266-arduino/

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] E. A. Lee, S. A. Seshia, Introduction to embedded systems, The MIT Press, 2 ed., MIT Press, London, England, 2016.

[2] G. Nguyen, M. Knockaert, M. Lognoul, X. Devroey, Towards comprehensive legislative requirements for cyber physical systems testing in the european union (2024). arXiv:2412.04132.

[3] T. Yue, L. C. Briand, Y. Labiche, A systematic review of transformation approaches between user requirements and analysis models, Requirements Engineering 16 (2010) 75–99. URL: http://dx.doi.org/10.1007/s00766-010-0111-y. doi:10.1007/s00766-010-0111-y.

[4] S. Saleem, M. N. Asim, L. V. Elst, A. Dengel, Generative language models potential for requirement engineering applications: insights into current strengths and limitations, Complex & Intelligent Systems 11 (2025). URL: http://dx.doi.org/10.1007/s40747-024-01707-6. doi:10.1007/s40747-024-01707-6.

[5] V. Opranescu, A. D. Ionita, Review of cyber-physical systems modeling with uml, sysml, and marte, IEEE Access 13 (2025) 47132–47145. URL: http://dx.doi.org/10.1109/ACCESS.2025.3551117. doi:10.1109/access.2025.3551117.

[6] M. Acher, J. G. Duarte, J.-M. Jézéquel, On programming variability with large language model-based assistant, in: Proceedings of the 27th ACM International Systems and Software Product Line Conference - Volume A, SPLC '23, ACM, 2023, p. 8–14. URL: http://dx.doi.org/10.1145/3579027.3608972. doi:10.1145/3579027.3608972.

[7] D. Jurafsky, J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, 3rd ed., 2025. URL: https://web.stanford.edu/~jurafsky/slp3/, online manuscript released January 12, 2025.

[8] G. Nguyen, X. Devroey, A3S3 - Automated Android Audit of Safety and Security Signals, Springer Nature Switzerland, 2025, p. 205–212. URL: http://dx.doi.org/10.1007/978-3-031-94590-8_25. doi:10.1007/978-3-031-94590-8_25.

[9] G. Nguyen, A. Sacré, A. Simonofski, X. Devroey, Ponderarium, a place for cyber physical system conformity assessment, in: Proceedings of the 59th Hawaii International Conference on System Sciences (HICSS), University of Hawai'i at Manoa, Waikoloa, HI, USA, 2026, p. 10. Forthcoming.

[10] J. A. J. Alsayaydeh, M. F. bin Yusof, M. Z. B. A. Halim, M. N. S. Zainudin, S. G. Herawan, Patient health monitoring system development using esp8266 and arduino with iot platform, International Journal of Advanced Computer Science and Applications 14 (2023). URL: http://dx.doi.org/10.14569/IJACSA.2023.0140467. doi:10.14569/IJACSA.2023.0140467.