

Towards LLM-Enhanced Product Line Scoping

Alexander Felfernig[†], Damian Garber[†], Viet-Man Le[†], Sebastian Lubos[†] and Thi Ngoc Trang Tran[†]

Institute of Software Engineering and AI, Graz University of Technology, Graz, Austria

Abstract

The idea of product line scoping is to identify the set of features and configurations that a product line should include, i.e., offer for configuration purposes. In this context, a major scoping task is to find a balance between commercial relevance and technical feasibility. Traditional product line scoping approaches rely on formal feature models and require a manual analysis which can be quite time-consuming. In this paper, we sketch how Large Language Models (LLMs) can be applied to support product line scoping tasks with a natural language interaction based scoping process. Using a working example from the smarthome domain, we sketch how LLMs can be applied to evaluate different feature model alternatives. We discuss open research challenges regarding the integration of LLMs with product line scoping.

Keywords

Product Line Scoping, Feature Models, Configuration

1. Introduction

Configurable products and services such as smarthomes, cars, and software systems have a high variability in terms of which components can be combined with each other [1, 2, 3]. To be able to handle variability in an efficient fashion, product line (PL) approaches have been widely adopted [4]. The idea of product lines is to allow a systematic reuse of shared assets which enables the reduction of development costs, reduced time to market, and higher product quality. Product line scoping is at the heart of PL engineering [5, 6, 7, 8, 9] – it is the process of defining which features and constraints should be included in a product line, i.e., which features and corresponding constraints should be part of a feature model. High-quality scoping decisions are crucial since they directly have an influence on the feasibility and commercial success of a product line.

Determining an optimal scope for a PL is a challenging task. This includes the evaluation of market trends, the balancing of potentially contradicting stakeholder requirements, and also ensuring the technical feasibility of the offered feature model configurations. A typical PL scoping process is based on workshops with experts. Related scoping decisions can be suboptimal due to a limited market and domain knowledge and – as a consequence – product lines have the risk of being under- or over-restricted. The underlying group decision task makes product line scoping a task directly related to requirements prioritization [10, 11] and group recommender systems [12, 13, 14].

Developments in the context of large language models (LLMs) [15] have created the potential to improve a variety of PL related tasks [16, 17, 18]. For example, in the context of software development, LLMs can be applied for re-engineering purposes allowing an LLM-based creation of PLs on the basis of artifacts such as UML diagrams, state charts, and Java programs [19]. Furthermore, LLMs have shown to be applicable in the context of new feature identification from different textual sources such as appstore evaluations [20]. Finally, LLMs have also shown to be applicable for model generation tasks, more precisely, the generation of feature models out of textual domain descriptions [21]. In the context of

ConfWS'25: 27th International Workshop on Configuration, Oct 25–26, 2025, Bologna, Italy

[†]These authors contributed equally.

✉ alexander.felfernig@tugraz.at (A. Felfernig); damian.garber@tugraz.at (D. Garber); v.m.le@tugraz.at (V. Le); sebastian.lubos@tugraz.at (S. Lubos); trang.tran@tugraz.at (T.N. T. Tran)

ORCID 0000-0003-0108-3146 (A. Felfernig); 0009-0005-0993-0911 (D. Garber); 0000-0001-5778-975X (V. Le); 0000-0002-5024-3786 (S. Lubos); 0000-0002-3550-8352 (T.N. T. Tran)



© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

PL scoping, LLMs have the potential to support engineers in their tasks of analyzing trade-offs and identifying commercially promising variability concepts.

In this paper, we propose the idea of applying LLMs to pro-actively support different tasks in product line scoping. This includes the aspects of estimating model optimality (in terms of market potential, alignment with customer preferences, and cost efficiency) and technical feasibility of the offered features by taking into account the existing product development resources.

We want to emphasize that these tasks are also relevant beyond software product lines, for example, in the context of designing and configuring complex products such as cars and smarthome systems. As a basis for our discussions, we introduce a simplified working example from the domain of smarthome systems. With the introduced feature model, we sketch scenarios where LLM-supported product line scoping can provide help in estimating commercial relevance and technical feasibility.

The basic idea sketched in this paper is to exploit LLMs for pro-actively supporting group decision processes in product line scoping, i.e., we envision a scenario based on human-AI collaboration where LLMs provide additional insights (not covered by experts), indicate new alternatives, and explain the consequences of specific decisions [22].

The major contributions of this paper are: first, we introduce the idea of exploiting LLMs for supporting product line scoping processes. Second, we sketch our ideas on the basis of a working example from the domain of smarthomes. Finally, we discuss related open research issues.

The remainder of this paper is organized as follows. In Section 2, we present a feature model from the smarthome domain. In Section 3, we analyze different scenarios in which LLMs can be employed to support product line scoping. Section 4 discusses open research issues. With Section 5, we conclude the paper.

2. Working Example: SmartHome Feature Model

In the following, we introduce a simplified feature model from the smarthome domain which will be used as a working example throughout the paper. Smarthome systems include a diverse set of features/functionalities including security, lighting, and climate control. Figure 1 depicts a feature model of a simplified smarthome product line. The root feature `SmartHomeSystem` includes three basic subfeatures which are `Security`, `Lighting`, and `ClimateControl`. Each of those features has further subfeatures (either optional or mandatory ones).

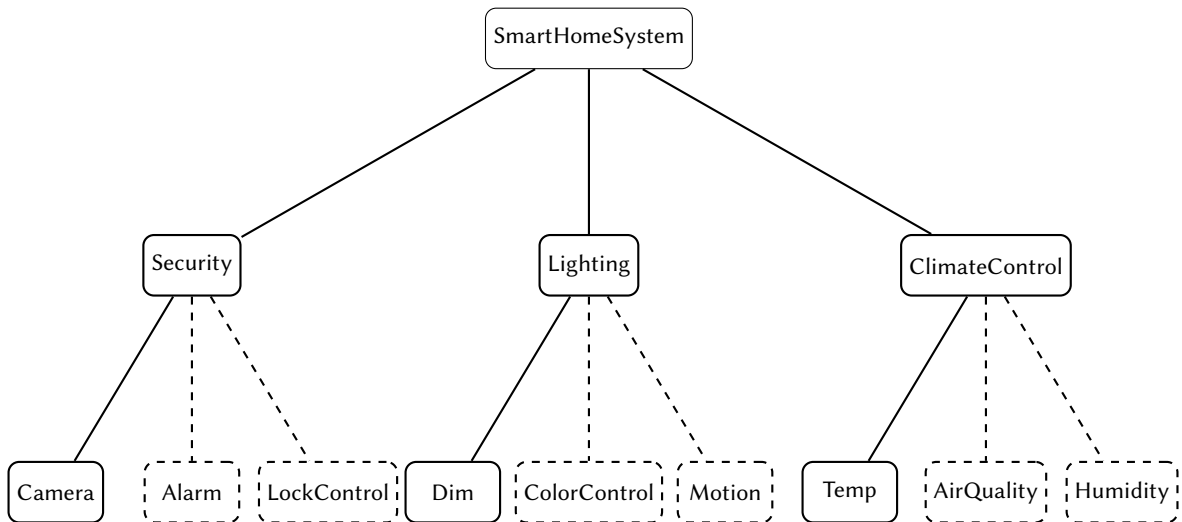


Figure 1: Feature model of a smarthome product line (dashed lines represent optional features, the other mandatory ones.)

In the feature model of Figure 1, the feature `SmartHomeSystem` is regarded as mandatory (due to the fact that we are not interested in empty configurations). The first-level child features `Security`,

Lighting, and ClimateControl are represented as *mandatory* which means that each smarthome configuration must include (in one way or another) each of those subfeatures (core features). Importantly, within the scope of a product line scoping process, this model can be regarded as flexible, i.e., features can be deleted or adapted and additional features (and also constraints) can be included. Such adaptations can be triggered by new insights from market analyses as well as insights directly related to the technical feasibility of allowed configurations.

Features at the second level can either be mandatory or optional – the features Camera, Dimming(Dim), and Temperature(Temp) are regarded as mandatory, since they represent basic equipment to be included in every smarthome configuration. The remaining features of the model are regarded as optional, for example, the feature ColorControl can be offered to a customer but does not have to be included in every configuration. Note that further modeling concepts can be used for representing feature model properties. For a detailed discussion of feature model knowledge representations, we refer to [1, 23].

Beyond hierarchical relationships such as mandatory and optional features, feature models often include cross-tree constraints that express dependencies between features. Such constraints further restrict the configuration space. Related example constraints in the smarthome domain could be Alarm requires MotionSensor (i.e., $Alarm \rightarrow MotionSensor$) and Alarm excludes ColorControl (i.e., $\neg(Alarm \wedge ColorControl)$). On the basis of such a variability model, users can perform different analysis operations (representing individual queries on the feature model). Examples of such queries are: *What are the minimum features required for a basic smarthome system with climate control?* or *Which feature combinations are most relevant for urban apartment customers?* A related LLM-based assistant has the potential to provide explanations why specific features should be included in the feature model.

3. Leveraging Large Language Models for Product Line Scoping

Large language models (LLMs) have a deep contextual understanding and vast commonsense knowledge which makes them applicable in assisting complex decision-making. In the following, we analyze in which way LLMs can be used to analyse the optimality of a product line (in terms of market relevance and technical feasibility). Product line scoping includes the task of identifying which features or feature combinations are commercially relevant and technically feasible. In contrast to often manual scoping operations on the basis of feature models, LLMs can augment and partly automate scoping operations by supporting analysis operations, feasibility checks, and related commercial insights.

In such scenarios, LLMs can be applied to answer scoping questions such as *Does a smarthome system including Security with Alarm and LockControl but excluding Lighting make commercial sense?* In this example, an LLM can infer potential consequences of omitting the Lighting feature. Furthermore, the related market acceptance could be estimated on the basis of knowledge about typical customer preferences, market trends in the smarthome domain, and technical background knowledge about the feasibility of such configurations. To some extent, LLMs can also take over reasoning/inference tasks such as assuring that constraints integrated in the feature model do not induce an inconsistency.

Since LLMs are capable of processing natural language, they can be applied for developing conversational interfaces that support, for example, product line scoping processes. On the basis of such interfaces, users (members of the product line scoping team) can express complex queries without necessarily being able to understand the formal semantics of feature models. Furthermore, product line scoping is not necessarily based on feature models but can also be based on a textual definition of a product line (a blueprint-based representation).

Examples of complex user queries are the following: a product manager might ask *What is the most commercially attractive combination of security features for urban apartments?* or *Suggest configurations that maximize energy efficiency while keeping costs low.*, or *Create a feature model that supports the previously mentioned configurations.*

Such a query-based interaction in product line scoping is based on the following LLM-related capabilities. On the basis of available product domain knowledge, LLMs can identify when specific

feature model parts are potentially triggering technical infeasibility. On the basis of information from product reviews, market reports, and other (potentially external) information sources in the training data, LLMs can estimate market potentials and the market relevance of specific features.

LLMs are able to generate human-readable explanations of the provided feedback/explanations which can also be tailored to the users' background knowledge [24]. For example, technical argumentations can be provided to users with the corresponding technical background. This also helps to create transparency and decision confidence for users part of the product line scoping team. Furthermore, product line scoping can be supported in an interactive fashion, i.e., alternative feature model implementations can be explored and users receive immediate feedback on the implications of their scoping decisions.

4. Open Research Issues

In the context of applying LLMs for supporting product line scoping processes, there exist a couple of open research issues which will be discussed in the following.

Reliability of LLM Feedback. LLM feedback/assessment quality regarding product line optimality and feasibility needs to be assured. Since LLMs do not have formal reasoning capabilities, hallucinations and inconsistencies can occur (e.g., an LLM could generate feature models or parts thereof which are inconsistent, i.e., do not allow the generation of a configuration). In this context, hybrid approaches need to be developed which allow a combination of LLMs with formal consistency checking (e.g., on the basis of constraint solvers or SAT solvers) [18].

LLM Updates. LLMs are based on domain-specific knowledge which experiences frequent updates. Since product line scoping has to depend on up-to-date market trend information and information about technological advances, and regulatory changes, efficient methods are needed that are able to continuously update the used LLMs and include information from Web search in result presentations. Furthermore, methods need to be developed that help to explain LLM feedback in terms of explaining the knowledge sources responsible for the given LLM feedback. This will help to support the aspects of transparency and trust which are crucial in the context of making high-involvement decisions for complex products and services.

Scalability of Inference Services. Since variability models can become quite large, the corresponding analysis and inference tasks require significant computational resources. Consequently, there is a need for inference services within reasonable runtime performance.

Dialog Management. Product line scoping is a complex (often group-based) decision task. This requires guidance in terms of proposing appropriate decision strategies (and decision processes) to be used for completing a decision task and also in terms of informing the user in an understandable fashion about the next steps to be completed to achieve the overall goal of identifying an optimal variability model of a product line. In this context, natural language interaction can be quite intuitive for users. However, communication has to be personalized, i.e., each user should receive system feedback and explanations in an understandable fashion.

Sustainability Aspects. Technical feasibility (T) and market relevance (M) are regarded as important decision criteria in the context of product line scoping. However, an important additional aspect to be taken into account are sustainability criteria (S) as defined by the United Nations Sustainability Development Goals (SDGs).¹ In this context, T, M, and C goals can be regarded as basic input of an optimization problem with the goal to identify optimal solutions.

Evaluation Metrics. Evaluation metrics need to be developed that help to evaluate the outcomes of LLM-enhanced product line scoping processes. Specifically, the inclusion of outdated knowledge and LLM hallucinations needs to be avoided. Related results need to be compared with the outcome of baseline processes without the support of LLM features. Example metrics include aspects such as

¹<https://sdgs.un.org/goals>

commercial impact, technical feasibility, satisfaction of the customer community, and longterm positive sustainability effects. Finally, the LLM output also needs to be evaluated with regard to potential biases, for example, manipulating a group decision into a specific direction.

Acceptance of Group Decision Support. For different reasons, group decision support tools often suffer from limited user acceptance [25]. On the one hand, such tools often require user feedback in terms of specifying explicit preferences which is not appreciated in complex scenarios such as product line scoping. On the other hand, there are issues related to aspects such as decision manipulation and limited preparedness to share his/her preferences. An important open issue in the context is find better ways of providing user support leading to more tool support acceptance as it is the case now.

5. Conclusions

In this paper, we have introduced the basic idea of exploiting large language models (LLMs) to the support decision processes in product line scoping for complex products and services. On the basis of a working example from the domain of smarthomes, we have sketched how variability modeling can be combined with LLMs with the goal to increase the quality of product line scoping. This way, stakeholders can be supported and guided in complex decision tasks in a more efficient fashion.

However, there are a couple of open research issues including for example, the aspects of LLM feedback reliability and explainability of the LLM output. Our next step will be a more detailed analysis of the commercial needs of LLM-supported product line scoping. The corresponding results will be the major features of our envisioned tool for supporting LLM-enhanced product line scoping.

Declaration on Generative AI

The authors used ChatGPT for language refinement and improving readability. All AI-generated suggestions were carefully reviewed and edited by the authors, who take full responsibility for the content of this publication.

Acknowledgments

The work presented in this paper has been developed within the research project GENRE (Generative AI for Requirements Engineering) funded by the Austrian Research Promotion Agency under the project number 915086.

References

- [1] A. Felfernig, A. Falkner, D. Benavides, *Feature Models: AI-Driven Design, Analysis and Applications*, Springer, 2024. doi:10.1007/978-3-031-61874-1.
- [2] A. Felfernig, L. Hotz, C. Bagley, J. Tiihonen, *Knowledge-based Configuration: From Research to Business Cases*, 1 ed., Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2014.
- [3] A. Popescu, S. Polat-Erdeniz, A. Felfernig, M. Uta, M. Atas, V.-M. Le, K. Pils, M. Enzelsberger, T. N. T. Tran, An overview of machine learning techniques in constraint solving, *J Intell Inf Syst* 58 (2022) 91–118. doi:10.1007/s10844-021-00666-5.
- [4] A. Metzger, K. Pohl, Software product line engineering and variability management: achievements and challenges, in: *Future of Software Engineering Proceedings, FOSE 2014*, ACM, New York, NY, USA, 2014, pp. 70–84. doi:10.1145/2593882.2593888.
- [5] J.-M. deBaud, K. Schmid, A systematic approach to derive the scope of software product lines, in: *International Conference on Software Engineering*, 1999, pp. 34–43. doi:10.1145/302405.302409.

- [6] I. John, J. Knodel, T. Lehner, D. Muthig, A Practical Guide to Product Line Scoping, in: 10th International on Software Product Line Conference, SPLC '06, IEEE Computer Society, USA, 2006, pp. 3–12.
- [7] L. Marchezan, E. Rodrigues, W. K. G. Assunção, M. Bernardino, F. P. Basso, J. a. Carbonell, Software product line scoping: a systematic literature review, in: 26th ACM International Systems and Software Product Line Conference, SPLC '22, ACM, New York, NY, USA, 2022, p. 256. doi:10.1145/3546932.3547012.
- [8] M. C. C. Ojeda, J. A. H. Alegría, F. J. A. Rodriguez, An exploratory study for scoping software product lines in a collaborative way, in: 11th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE '18, ACM, New York, NY, USA, 2018, p. 17–20. doi:10.1145/3195836.3195852.
- [9] K. Schmid, A comprehensive product line scoping approach and its validation, in: 24th International Conference on Software Engineering, ICSE '02, ACM, New York, NY, USA, 2002, pp. 593–603. doi:10.1145/581339.581415.
- [10] S. Lubos, A. Felfernig, D. Garber, V.-M. Le, M. Henrich, R. Willfort, J. Fuchs, Towards Group Decision Support with LLM-based Meeting Analysis, in: 33rd ACM Conference on User Modeling, Adaptation and Personalization, UMAP Adjunct '25, ACM, New York, NY, USA, 2025, pp. 331–335. doi:10.1145/3708319.3733646.
- [11] R. Samer, M. Stettinger, A. Felfernig, Group Recommender User Interfaces for Improving Requirements Prioritization, in: 28th ACM Conference on User Modeling, Adaptation and Personalization, UMAP '20, ACM, New York, NY, USA, 2020, pp. 221–229. doi:10.1145/3340631.3394851.
- [12] A. Felfernig, L. Boratto, M. Stettinger, M. Tkalcic, Group Recommender Systems, Springer, 2024.
- [13] V.-M. Le, T. N. T. Tran, A. Felfernig, Consistency-based integration of multi-stakeholder recommender systems with feature model configuration, in: 26th ACM International Systems and Software Product Line Conference, SPLC '22, ACM, New York, NY, USA, 2022, pp. 178–182. doi:10.1145/3503229.3547050.
- [14] J. Masthoff, A. Delić, Group recommender systems: Beyond preference aggregation, in: F. Ricci, L. Rokach, B. Shapira (Eds.), Recommender Systems Handbook, Springer US, New York, NY, 2022, pp. 381–420. doi:10.1007/978-1-0716-2197-4_10.
- [15] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, A. Mian, A Comprehensive Overview of Large Language Models, ACM Trans. Intell. Syst. Technol. (2025). doi:10.1145/3744746.
- [16] M. Acher, J. G. Duarte, J.-M. Jézéquel, On programming variability with large language model-based assistant, in: 27th ACM International Systems and Software Product Line Conference, SPLC '23, ACM, New York, NY, USA, 2023, pp. 8–14. doi:10.1145/3579027.3608972.
- [17] S. Greiner, K. Schmid, T. Berger, S. Krieter, K. Meixner, Generative AI And Software Variability - A Research Vision, in: 18th International Working Conference on Variability Modelling of Software-Intensive Systems, VaMoS '24, ACM, New York, NY, USA, 2024, pp. 71–76. doi:10.1145/3634713.3634722.
- [18] L. Hotz, C. Bähnisch, S. Lubos, A. Felfernig, A. Haag, J. Twiefel, Exploiting Large Language Models for the Automated Generation of Constraint Satisfaction Problems, in: É. Vareilles, C. Grosso, J. M. Horcas, A. Felfernig (Eds.), ConfWS 2024, volume 3812 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2024, pp. 91–100.
- [19] M. Acher, J. Martinez, Generative AI for Reengineering Variants into Software Product Lines: An Experience Report, in: 27th ACM International Systems and Software Product Line Conference, SPLC '23, ACM, New York, NY, USA, 2023, pp. 57–66. doi:10.1145/3579028.3609016.
- [20] J. Wei, A.-L. Courbis, T. Lambolais, B. Xu, P. L. Bernard, G. Dray, W. Maalej, Getting Inspiration for Feature Elicitation: App Store- vs. LLM-based Approach, in: ASE 2024, ASE '24, ACM, New York, NY, USA, 2024, p. 857–869. doi:10.1145/3691620.3695591.
- [21] J. A. Galindo, A. J. Dominguez, J. White, D. Benavides, Large Language Models to generate meaningful feature model instances, in: 27th ACM International Systems and Software Product Line Conference, SPLC '23, ACM, New York, NY, USA, 2023, pp. 15–26. doi:10.1145/3579027.

- [22] S. Lubos, M. Gartner, A. Felfernig, R. Willfort, Leveraging LLMs to Explain the Consequences of Recommendations, in: 33rd ACM Conference on User Modeling, Adaptation and Personalization, UMAP '25, ACM, New York, NY, USA, 2025, pp. 318–322. doi:10.1145/3699682.3728328.
- [23] D. Benavides, S. Segura, A. Ruiz-Cortes, Automated analysis of feature models 20 years later: A literature review, *Information Systems* 35 (2010) 615–636.
- [24] S. Lubos, T. N. T. Tran, A. Felfernig, S. Polat Erdeniz, V.-M. Le, LLM-generated Explanations for Recommender Systems, in: 32nd ACM Conference on User Modeling, Adaptation and Personalization, UMAP Adjunct '24, ACM, New York, NY, USA, 2024, pp. 276–285. doi:10.1145/3631700.3665185.
- [25] M. C. C. Ojeda, F. A. Rodriguez, C. A. Collazos, Identifying Collaborative Aspects During Software Product Lines Scoping, in: 23rd International Systems and Software Product Line Conference, SPLC '19, ACM, New York, NY, USA, 2019, pp. 98–105. doi:10.1145/3307630.3342420.