**Paper Sections**

1. Introduction
2. Motivating Examples
3. Checking PL Refinement
4. Evaluation
5. Related Work
6. Conclusions
7. **Introduction**

**What is a Software Product Line?**

Why use Software Product Lines?

What are the advantages?

However, we have to add new features, correct bugs

SPL Evolution without tool support

There are some strategies to check SPL refinement [], however they are unsound …

To help to solve this problem, in this work we propose ….

As a basis for the implementations, we use the formal definition of SPL refinement []

A software product line (SPL) is a set of related software products that are generated from reusable assets. Products are related in the sense that they share common functionality. This kind of reuse targeted at a specific set of products can bring significant productivity and time to market improvements [1, 2]. The goal of SPLs is to enable a controlled reuse and evolution of features common to multiple products. However, SPL evolution can be quite challenging. Changing requirements will require adaptation of a product line’s features and, hence, affect deployed products as well as products still under development.

In general, successful evolution of SPLs requires comprehension of the impact of changes. For example, if a product has a bug, it is necessary to identify all deployed and under development products that are affected. Also, if a feature is added or removed it is necessary to assess the impact on all products in order to determine whether an already deployed product can be maintained using the current product line or if an older version of the product line needs to be used. Furthermore, in this context we have to deal not only with assets but also with artifacts, like feature models [4] and configuration knowledge [3], that enable product generation, and they should all be changed consistently. Being more specific, when improving the design of the SPL’s artifacts, it might be important to make sure that the associated changes do not affect the behavior of the existing SPL products. This notion of safe evolution is translated by a formal definition of SPL refinement [5, 6], which guarantees that the observable behavior of products in the original SPL is preserved by corresponding products in the evolved SPL.

References

[1] K. Pohl, G. B¨ockle, and F. van der Linden. Software Product Line Engineering: Foundations, Principles and Techniques. Springer, 2005.

[2] F. van der Linden, K. Schmid, and E. Rommes. Software Product Lines in Action: the Best Industrial Practice in Product Line Engineering. Springer, 2007.

[3] K. Czarnecki and U. Eisenecker. Generative programming: methods, tools, and applications. Addison-Wesley, 2000.

[4] K. Kang, S. Cohen, J. Hess, W. Novak, and A. S. Peterson. Featureoriented domain analysis (FODA) feasibility study. Technical Report CMU/SEI-90-TR-21, SEI CMU, 1990.