S2T2-Configurator: Interactive Support for Configuration of Large Feature Models

Goetz Botterweck and Andreas Pleuss

Lero-The Irish Software Engineering Research Centre, Univ. of Limerick, Ireland
{goetz.botterweck, andreas.pleuss}@lero.ie

Abstract. S2T2-Configurator is a visual tool for configuration of feature models. In this tool paper, we focus on interactive techniques that support the configuration of large and complex models.

1 Motivation

Even though the fundamental practices in Product Line Engineering (PLE) are well known and applied in practice [3], it is still a challenge to create and handle product lines of realistic size and complexity. Besides organizational challenges (e.g., how to transition to a PLE approach), a major inhibitor to product line adoption is the complexity of the underlying artifacts [7].

A common way to represent product lines (PL) are feature models [5, 4], which describe configuration options like optional features or alternatives to choose from. By configuring a feature model, a user can specify one particular product to be derived from the product line. However, due to the number of features and dependencies between them, it is difficult to understand the model as a whole and the consequences of particular configuration decisions during the configuration process.

A potential solution to mitigate this situation are visualization techniques [12, 11], which have been shown to reduce cognitive complexity [2].

In earlier work [1, 9] we introduced S2T2 Configurator1 a tool that demonstrates the use of visualization techniques for common PLE tasks. In this paper we present interactive techniques for the handling of large and complex models, e.g., contextual filters, views for different stakeholders, search-and-highlight, and automated collapse/expand.

2 Interactive support for product configuration

Main structure and layout modes. S2T2 provides various layout styles to show the main hierarchy of the feature model and additional dependencies between

---

1 Supported, in part, by Science Foundation Ireland grant 10/CE/I1855. We are grateful to Mikoláš Janota, Denny Schneeweiss, and Elisabeth Engel who contributed significantly to the concepts and implementation of the S2T2 Configurator.

1 S2T2 = SPL of SPL Tools and Techniques
features (cross-tree constraints). In Figure 1 we show the “vertical explorer” layout. Other display modes are, e.g., optimized to show the mapping between features and implementation components or to show the effect of configuration decisions on product attributes (see “feature flow maps” in [9]).

Configuration state. Whenever the user decides that a feature should be selected or eliminated (✔ = selected, ✗ = eliminated), the tool calculates the consequences of that decision, i.e., it tests all potential next operations for satisfiability of the model. Consequences are automatically applied to the model (i.e., automatically setting features as selected or deselected if the model would become unsatisfiable otherwise) indicated by gray colored icons. To simplify the understanding of the automatically calculated values, the tool provides an explain function, which on request marks all user decisions and constraints that caused the particular value. This graphical representation is based on a proof generated by the reasoning engine.

Progress indicators. In product configuration all available variability must be resolved until no decision is left open. Hence, in large models it is useful to mark areas that still need decisions. We use color shadings to indicate the configuration progress of subtrees and features. Since these colors can be processed pre-attentively (i.e., without consciously thinking about it), this allows to immediately spot unconfigured areas even in very large models.

Filtering of large models. To support the handling of large models, S2T2 allows to focus on a subset of the model. In order to define what is relevant, users select those features that they are interested in and then activate filtering heuristics (or combinations thereof) that show (i) all directly related nodes, (ii) entire subtrees below, or (iii) all ancestors to indicate relative location in the model.

Furthermore, we provide means to pre-define a list of such views and then step through them in the configuration process. This can be used in several application scenarios: For instance, different “views” can be defined for varying stakeholders and aspects. As an example consider the feature model shown in Figure 2 which shows different aspects of a car, like CustomerFeatures, SoftwareComponents, Casing, and Electronics. We can then treat the configuration as a guided process, handling one aspect after another and thus reducing complexity. For instance, by selecting CustomerFeatures and then using the function “Focus on subtrees” the user gets the filtered view marked with 8. After defining similar views for the features marked with 1 to 7 the users can
concentrate on one aspects of the model at a time and switch back and forth between these views - similar to a “wizard” interaction pattern.

A similar function allows to focus on a set of features and all related nodes. For instance, by selecting GasUsageEstimation and using “Focus on related features, users get the filtered view marked with dashed line. This supports the users in complex configuration decisions, where they have to focus on one feature and consider all consequences in other parts of the model.

The parameters of these views can be adapted during configuration, e.g., the users can add additional features to the set of focused nodes or they can toggle the display of related nodes, which are linked via feature dependencies.

3 Current work

To explore its practical use S2T2 Configurator has been applied to feature models from various domains (online shops, embedded systems, component based software architectures, software evolution). The largest model used so far for performance tests was a generated test model with 1114 features and 278 cross-tree constraints. The largest models used for configuration by users contained 227 features and 41 cross-tree constraints. In current work, we are evaluating the effects of the suggested interaction patterns on task execution time, error rates, and subjective user preference in a systematic user study.
References


Appendix: Demonstration Plan

- Software architecture, frameworks (Eclipse EMF, Prefuse visualization)
- Changes in the model as the configuration progresses, reasoning engine, consequences, explain, automatic completion
- Configuration with large models
- Progress indicators with color schemes
- Focus: filter view, focus on related features, focus on subtrees, show ancestors, usages of model views
- Feature models with product attributes, visualization with feature flow maps
- Integration with other tools, conversion of DSLs with model transformations.