Visualization Techniques for Application in Interactive Product Configuration

Andreas Pleuss
Lero
Limerick, Ireland
andreas.pleuss@lero.ie

Rick Rabiser
Christian Doppler Lab. for
Automated Software Eng.
JKU Linz, Austria
rick.rabiser@jku.at

Goetz Botterweck
Lero
Limerick, Ireland
goetz.botterweck@lero.ie

ABSTRACT

In product line engineering (PLE) a major challenge is the complexity of artifacts that have to be handled. In real-world product lines, variability models can become large and complex comprising thousands of elements with hundreds of non-trivial dependencies. Visual and interactive techniques aim to reduce the (cognitive) complexity and support the user during challenging PLE tasks like product configuration. There are many visualization techniques described in the literature – e.g., in Software Visualization – and some isolated techniques have been applied in PLE tools. Nevertheless, the full potential of visualization in the context of PLE has not been exploited so far. This paper provides an overview of (1) available visualization techniques and criteria to judge their benefits and drawbacks for product configuration, (2) which have been applied in product configuration in PLE, and (3) which could be beneficial to support product configuration. We propose a research agenda for future work in visual and interactive PLE techniques.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management—*Software configuration management*; D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User Interfaces*

General Terms

Design, Management

Keywords

Product line engineering, software visualization, product configuration

1. INTRODUCTION

While many basic concepts of PLE are well understood, it is still a challenge to create, use and evolve product lines in projects of realistic size [10, 37, 26]. Besides organizational challenges – such as how to introduce product lines into an organization – a major inhibitor for the adoption of product line approaches is the complexity of the underlying artifacts, in particular their variability.

The key idea of PLE is that the investments required to build up the product line are outweighed by the benefits of being able to quickly derive customized products [10, 26]. The size and complexity of product lines and the models that document commonalities and variability of reusable assets [6, 30] however can diminish the potential benefits of applying a PLE approach [10, 26]. For real-world product lines such models can comprise thousands of elements with hundreds of (often non-trivial) dependencies [7, 12]. For example, the feature models of the Linux kernel and the embedded operating system eCos comprise over 6000 features (Linux) and over 1200 features (eCos) respectively [7].

It has been shown that visualization can help to reduce (cognitive) complexity [8]. Many visualization techniques have been described in the literature (e.g., in Software Visualization [8]) and some isolated techniques – mostly trees and tables – have been applied in PLE [13, 26, 35]. Nevertheless, their full potential for PLE has not been exploited so far. In this paper *we focus on visualization techniques in product configuration* in PLE. We however believe that this work can be the basis of further work on applying 'new' visualization techniques in PLE in general.

In this paper we (1) collect available visualization techniques from the literature and provide criteria to grade their potential value for product configuration (Section 2). (2) We give examples of visualization techniques that have already been applied in a PLE context (in product configuration). This analysis (Section 3) is based on our own experience in PLE and systematic literature reviews conducted earlier [11, 26, 34]. (3) We identify additional visualization techniques that have potential to be applied and could provide benefit in product configuration (Section 4). (4) We discuss how these techniques can be applied and evaluated in practice by considering aspects like user tasks, tailoring of visualizations, and interactive tool support and propose a resulting research agenda in Section 5.

2. VISUALIZATION TECHNIQUES

We collected visualization techniques from various sources, based on a (non-systematic) literature review, including the "periodic table of visualization methods" by Lengler and Eppler [19], the collection by Friedman [15] and textbooks by Card*et al.* [8] and Ware [38]. We also considered surveys like the one by Herman *et al.* [17] and comparisons based on

SPLC'11 August 21-26, 2011, Munich, Germany

empirical data like those for tree visualization techniques by Kobsa [18] and by Barlow and Neville [3].

The choice of a visualization technique depends on the underlying data and the tasks to be supported [32, 39]. In product configuration in PLE, the data to be visualized is the variability (defined in variability models) with the aim to support product configuration. Variability models can be considered as graphs with nodes (e.g., features [6] or decisions [30]) and different kinds of edges (relationships among features or among decisions) between them. The relationships can define a hierarchy between the nodes (e.g., a tree hierarchy in feature diagrams) or connect arbitrary nodes (e.g., cross-tree constraints in feature models). In addition, there can be further information relevant for making decisions in configuration like attribute values associated with product features (e.g., costs, weight).

The strengths and weaknesses of a visualization technique depend on which of these aspects it visualizes and how much emphasis it gives to an aspect. For instance, a tree map [21] visualizes a tree hierarchy among elements together with numerical values but does not support representing additional dependencies between questions. It emphasizes the influence of numerical values while the hierarchical structure is toned down compared to simpler tree visualizations.

Table 1 provides an overview of each visualization technique discussed throughout this paper and rates their support for the aspects of variability models identified above, i.e., hierarchies, cross-cutting dependencies, and attribute values. Further criteria are additional support for the configuration task itself in terms of a configuration workflow (e.g., step-by-step approach, ordering of nodes), and adequacy for complex models. The techniques are grouped by the dominant aspect they support. The second last column lists the specific (potential) benefits and drawbacks of each technique while the last column provides literature references for techniques which have already been used in PLE.

3. VISUALIZATION TECHNIQUES USED IN PRODUCT CONFIGURATION

We identified sample cases where the described visualizations are already applied in a product line context. More specifically, we identified clustering, decision trees, tree maps, cone trees, tables, flow maps, and UML diagrams as visualization techniques that have been applied in or at least investigated in product configuration in PLE. All techniques have their benefits and drawbacks and thus 'the' optimal technique cannot be identified and will probably never exist.

3.1 Clustering (Feature Trees)

Clustering is a visualization technique that presents elements that belong together due to some criterion close to each other, e.g., in a graph or tree with a special layout. The principle of clustering is used in feature modeling [6] to establish a (tree) hierarchy of features. Clustering plays a role in product configuration in so far feature trees present hierarchically arranged features to users. Users can traverse the feature tree and select features to derive products. Clustering is an essential concept of feature modeling in PLE and affects product configuration based on feature models.

Benefits: Clustering is very useful to create and visualize hierarchies.

Drawbacks: It is hard to focus on more than one criterion at a time, e.g., to arrange features according to diverse attributes. Also, cross-tree dependencies frequently break the nicely clustered hierarchy.

3.2 Decision Trees

Decision trees are a visualization technique that presents a number of choices to users as a tree to reflect a certain order of decisions to be made. Decision modeling in PLE [30] focuses on capturing variability as a number of user choices. Product configuration tools that are based on decision models typically present decisions in a certain order, however, not necessarily as a tree [14] – some approaches also allow to present them as flat lists or tables [25, 29]. From an abstract point of view, feature trees can also be seen as decision trees, however, an order of making choices is usually not explicitly defined but rather implied by constraints. Decision trees also are an essential concept in PLE to visualize variability in product configuration.

Benefits: Decision trees are very useful to present users with a number of choices in a particular order or hierarchy.

Drawbacks: Variability models – be it feature models or decision models – typically are graphs and not trees (cross-cutting dependencies). Also, an order of user choices is often hard to explicitly define as there might be more than one order possible and/or desired.

3.3 Tree Maps

The tree map visualization method is a space efficient representation of hierarchical structures. It maps hierarchical data to a rectangular 2D display and partitions the display into a collection of bounding rectangles each representing a node. The primary advantage of tree map visualizations is that 100% of the designated display space is used. While there are research prototypes that represent product line feature models as tree maps [21], this technique is not widely used in PLE (product configuration) so far.

Benefits: Tree maps are very space efficient and good to present hierarchy.

Drawbacks: Tree maps do not support representing cross-cutting dependencies. The hierarchical structure is not as much emphasized as in 'simpler' tree visualizations.

3.4 Cone Trees

Cone trees are a 3D visualization technique to represent hierarchical information. Cone trees distribute the elements of tree-like structures in a virtual 3D room where roots of subtrees are placed in the apex of cones that link them to their child nodes which are placed around their cone bases. Trinidad *et al.* [36] have investigated the use of cone trees for feature diagram visualization in PLE. This technique is also not widely used in PLE so far.

Benefits: 3D looks very fancy and elaborate.

Drawbacks: Most users are not used to 3D-visualizations and input devices are not well-suited for working with 3D-models. Also, like all tree representations, cross-cutting dependencies are a problem.

3.5 Tables

Tables are a central concept in PLE (e.g., in scoping [4, 28]) and are also widely used in product configuration. Many tools that support product configuration in PLE represent variability in tables and/or allow to compare variations of

Table 1: Visualization techniques for product configuration in PLE and rating of how well they support diverse aspects of variability models relevant for visualization (++ very good support; + good support; – weak support; – – no support). We grouped techniques whose dominant aspects are (A) representing hierarchy, (B) visualizing cross-cutting dependencies, (C) visualizing attributes, (D) visualizing the configuration workflow, and (E) not easily assignable to (A)-(D).

Dominant Aspect	Technique	Hierarchy	Cross-Cutting Dependencies	Attribute Values	Configuration Workflows	Adequacy for complex models	(Potential) Benefits (+) and Drawbacks (–)	Used in PLE Product Configuration?
(A)	Clustering (Feature Trees)	++	–	–	–	+	(+/–) focuses on one attribute; (–) cross-tree dependencies break nicely clustered hierarchy; Section 3.1	[6]
(A)	Decision Trees	+	–	–	+	+	(+) choices in a particular order or hierarchy; (–) order/hierarchy is hard to define explicitly; Section 3.2	[14, 25]
(A)	Tree Maps	++	– –	+	–	+	(+) very space efficient; (+) displays attributes; (–) no support for representing additional, cross-cutting dependencies and workflows; Section 3.3	[21]
(A)	Cone Trees	+	–	–	–	+	(+) fancy and elaborate; (–) most users not used to 3D-visualizations; (–) cross-cutting dependencies are a problem; Section 3.4	[36]
(C)	Tables	–	–	++	+	–	(+) good to show attributes; (+) good to communicate with sales/management; (+) good for comparing configurations; (–) hard to visualize many (types of) dependencies; Section 3.5	[25, 27]
(D)	Flow Maps	+	– –	–	+	–	(+) nice to show the flow of data; (–) become confusing for large and complex models quickly; (–) do not support representing cross-cutting dependencies; Section 3.6	[31]
(E)	UML Diagrams	+	+	++	–	– –	(+) compliance with standards; (–) hard to perceive by non-software engineers; (–) not initially developed to support variability representation; Section 3.7	[2, 16]
(A)	Hyperbolic Trees Space Trees EncCon Trees	++	–	–	–	++	(+) good for really large models; (–) visualizing cross-cutting dependencies problematic; (–) not good to present (feature) attributes; Section 4.1, Section 4.2, Section 4.3	n
(A)	Cheops	++	–	+	+	++	(+) good for really large models; (+) allows displaying attributes for selected 'triangles'; (+) also supports configuration workflows; (–) no cross-cutting dependencies; Section 4.4	n
(B)	Venn Diagrams	+	+	+	–	– –	(+) good to visualize cross-cutting dependencies; (–) large and complex models -> many circles and overlaps -> confusing; Section 4.5	n
(B)	Semantic Networks	+	+	+	+	–	(+) might help making configuration decisions; allows to visualize cross-cutting dependencies; (–) yet another type of dependency might confuse the user; Section 4.6	n
(C)	Elastic Lists	–	–	++	+	–	(+) focuses on visualizing important attributes of e.g. features; (–) dependencies are not taken into account; Section 4.7	n
(D)	Flow Charts	+	–	–	+	–	(+) 'find a way' through the decision space in product configuration; (–) hard to (also) visualize dependencies not related with the order of making decisions; Section 4.8	n
(D)	Cause-Effect Chains	+	–	–	+	–	(+) help to 'find a way' through the decision space in product configuration and explains complex dependencies; (–) cross-cutting dependencies are still a problem (here: cross-cutting the chain); Section 4.9	n
(E)	Data Maps	+	– –	–	+	–	(+) software-intensive systems can quite easily be presented in a schematic manner that users can understand; (–) not so useful for technical details; (–) no cross-cutting dependencies and non-functional attributes of features; Section 4.10	n

products side by side. For instance, [25, 27] follow a decision-oriented variability modeling approach [30] and use tables to present decisions and their attributes.

Benefits: Tables allow showing features (or decisions) in relation with arbitrary attributes (e.g., description, price). Tables are widely used by sales and marketing people as well as management and are thus very useful to communicate variability to such stakeholders. Also, tables allow comparing multiple configurations with each other.

Drawbacks: While it is possible to visualize relationships within tables, for large and complex models with multiple types of dependencies tables quickly become confusing.

3.6 Flow Maps

The flow map visualization technique aims to show the flow of data graphically, e.g., a tree can be represented as a flow map where the thickness of edges denotes the amount of data. Flow maps have recently been applied to represent feature trees to support product configuration [31]. However, this technique is also not widely used in PLE so far.

Benefits: Flow maps are nice to show the flow of data, e.g., in product configuration.

Drawbacks: Flow maps get confusing for large and complex models quickly. Also, they do not support representing cross-cutting dependencies.

3.7 UML Diagrams

The visual representation of UML class diagrams and similar structural diagrams (summarized as "Entity Relationship Diagrams" in [19]) is important in UML-based PLE approaches [2, 16] where UML models are adapted to document and represent variability, for example, using the UML stereotype mechanism. While these approaches document variability using UML, product configuration is usually not supported and requires additional representations. For example, the approach by Atkinson *et al.* [2] uses decision tables. Product configuration might however still be supported directly in the diagrams.

Benefits: In industry, compliance with standards is very important. Using a standard like UML would foster standardization also in product configuration.

Drawbacks: UML diagrams are hard to perceive by non-software engineers and have not initially been developed to support representing variability.

4. VISUALIZATION TECHNIQUES TO BE INVESTIGATED FURTHER

In product configuration variability is to a large extent represented as a tree or table of features or decisions. While other techniques (UML diagrams, tree map, flow map, cone tree) have been used or at least investigated, these have not gained much attention. It might make sense to further investigate the use of tree maps and flow maps, however, there are several other visualization techniques that have not been tried out at all so far and that could be useful to support product configuration. We have had a look at a large number of techniques (cf. Section 2). Here we discuss for which of these we think it would make sense to investigate them further (cf. Table 1) and make some proposals of how they could be used to support product configuration. For some selected techniques we provide sketches of how we think they might be used in product configuration. We use a simple and easy to understand fictitious mobile phone example for this purpose. The variability in this example originates from different screen options, connectivity options and input device options for a mobile phone. We provide links to further information for techniques we do not present sketches for.

4.1 Hyperbolic Trees

A hyperbolic tree addresses an important challenge of representing hierarchical data as a tree: visual clutter. The number of nodes per level of a tree can grow rather fast leading to a space problem. A hyperbolic tree employs hyperbolic space: it places a node far enough from its parent and gives the node almost the same amount of space as its parent for laying out its own children. A hyperbolic tree is usually presented as a disk (a circle), where the root of the tree is in the middle and the branches are arranged towards the border of the disk. A special form of a hyperbolic tree is the H3Viewer [20], which is a 3D spherical representation of a hyperbolic tree.

Potential Benefits: In product configuration, a hyperbolic tree could be used to present large variability models for which a standard tree layout would lead to space problems.

Potential Drawbacks: Hyperbolic trees do not help with the problem of visualizing cross-cutting dependencies and are also not well-suited to present (feature) attributes.

Further Information: `hypertree.woot.com.ar`; [20].

4.2 Space Trees

Like hyperbolic trees, space trees are an approach to overcome the space problem of conventional tree visualizations. They support dynamic rescaling of branches to fit the screen. In tools rescaling is visualized by animated transitions and camera movement has been optimized. Collapsed branches are represented by triangular preview icons where the shading, height, and width of the triangle indicate the number of nodes, the depth, and the average width of the node.

Potential Benefits: Same as for hyperbolic trees but additional support for large models using interactive techniques.

Potential Drawbacks: Same as for hyperbolic trees.

Further Information:
`www.cs.umd.edu/hcil/spacetree/`; [24].

4.3 EncCon Trees

Similar to hyperbolic trees and space trees EncCon ("Enclosure" and "Connection") trees support representing very large trees. It combines the good support of conventional tree visualizations for hierarchy ("connection") with efficient space usage like in tree maps ("enclosure"). Child nodes are placed around parent nodes using a circular, space-filling division method.

Potential Benefits: Same as for hyperbolic trees and space trees but additional support for large models using space-filling algorithm.

Potential Drawbacks: Same as for hyperbolic trees and space trees.

Further Information: [22].

4.4 Cheops

Cheops is an approach that was designed to support representing and browsing huge, complex information hierarchies. The Cheops approach maintains context within a huge hierarchy represented as a tree of triangular-shaped nodes (very space efficient), while simultaneously providing easy access

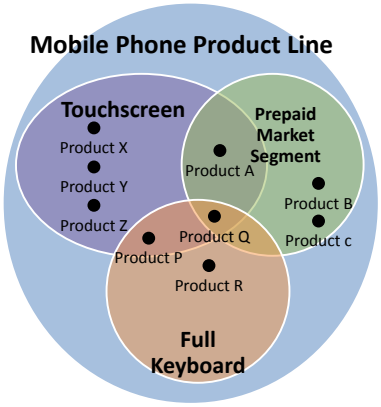

Figure 1: Example of using a Venn Diagram to support product configuration.

to details (for selected nodes). It would thus be very beneficial to support product configuration: the tree of triangles would provide an overview that allows to navigate through the decision-space and selecting a triangle would allow resolving variability and/or displaying relevant attributes.

Potential Benefits: same as for other trees but additionally supports configuration workflow and displaying attributes.

Potential Drawbacks: cross-cutting dependencies cannot easily be visualized.

Further Information:
`http://maparent.ca/~maparent/paper.html`; [5].

4.5 Venn Diagrams

A venn diagram shows commonalities and variabilities of sets of elements using overlapping circles. Such a representation could also be applied to support product configuration in PLE. Features or groups of features (or even whole products) could be presented as circles. Features/Products that have something in common could then visually overlap each other (cf. Fig. 1). Selecting a particular circle would allow to configure the feature/product being aware of overlaps due to constraints and dependencies.

Potential Benefits: Might help to better visualize cross-cutting dependencies, however, it might make sense to use this technique only in combination with other techniques.

Potential Drawbacks: Large and complex models will result in many circles and overlaps which might be quite confusing.

4.6 Semantic Networks

A semantic network shows the relations of terms with regard to their semantics in a network-like graph representation. For product configuration one could imagine to present features or decisions as a network taking their semantic relations into account (cf. Fig. 2). For example, semantically, two features might be related while there is no explicit dependency modeled among them. Users might find it helpful to be presented with a relation that shows that the two features have something to do with each other.

Potential Benefits: Might help making configuration decisions. Allows to visualize cross-cutting dependencies.

Potential Drawbacks: Showing yet another type of dependency among features or decisions might confuse the users more than showing semantics helps them.

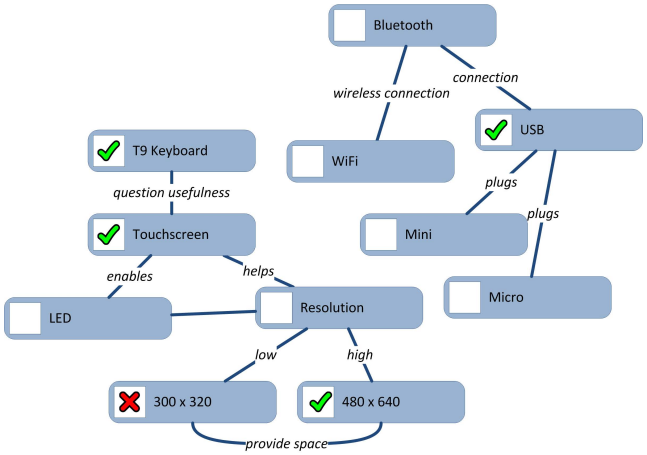

Figure 2: Example of using a Semantic Network to support product configuration.

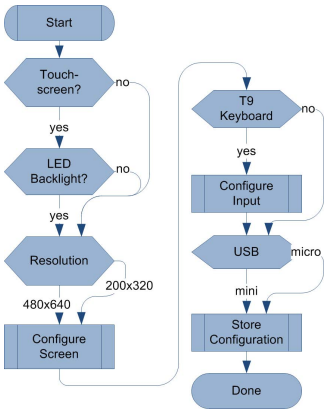

Figure 3: Example of using a Flow Chart to support product configuration.

4.7 Elastic Lists

Elastic lists visualize relative proportions (weights) of metadata by size and visualize the "characteristicness" of a metadata weight by brightness. This concept could be used in product configuration in PLE to visualize the attributes of features or decisions (the metadata) in lists of different sizes created according to the importance of the attributes. For example, price could be more important than location for a particular model.

Potential Benefits: Focuses on visualizing important attributes of e.g. features.

Potential Drawbacks: Dependencies (in general and especially cross-cutting ones) are not taken into account.

Further Information:
`http://moritz.stefaner.eu/projects/elastic-lists/`

4.8 Flow Charts

A flow chart shows a sequence of steps together with possible options. In product configuration the "flow of selecting features" or the "flow of making decisions" could be represented as a flow chart. This way, the product configuration process would be presented as a number of options in a certain order with forks and joins (cf. Fig. 3). Approaches like

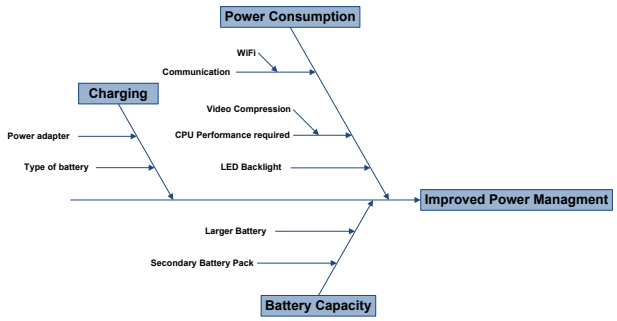

Figure 4: Example of using a Cause-effect chain to support product configuration.

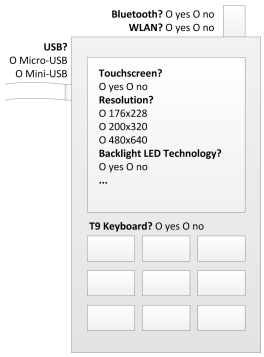

Figure 5: Example of using a Data Map to support product configuration.

the staged configuration approach proposed by Czarnecki *et al.* [9] could also benefit from a flow chart representation.

Potential Benefits: For large and complex models, flow charts might help to 'find a way' through the decision space in product configuration.

Potential Drawbacks: It might be hard to (also) visualize dependencies not related with the order of making configuration decisions, especially cross-cutting dependencies.

4.9 Cause-effect Chains

Cause-effect chains represent causes and effects related with each other through arrows pointing from cause to effect. In product configuration selecting a feature or making a decision often has many effects (e.g., requires constraints, inclusion of assets, etc.). Representing the effects of a currently selected feature or decision option using a cause-effect chain might thus make sense (cf. Fig. 4). Some research in this direction (however not focused on also visualizing the chains) has already been conducted by Rosa *et al.* [27].

Potential Benefits: Like flow charts, for large and complex models, cause-effect chains may help to 'find a way' through the decision space in product configuration, especially regarding explaining complex dependencies to users.

Potential Drawbacks: Cross-cutting dependencies are still a problem (here they cross-cut the chain).

4.10 Data Maps

Data maps represent information within another representation that is easily understandable. For example, crime rates can be presented by coloring areas on a map of a country or city. For product configuration this concept could actually be quite useful. Think of a representation of the system to be configured that is easily understood by users, for example, an architecture diagram for architects, a schematic drawing of the system for end-users, a GUI prototype, etc. Within this representation, variability could be shown arranged in relevant areas affected by the features/decisions. For instance, a user choice related with the screen of the mobile phone could be shown directly on the screen of the mobile phone currently being configured (cf. Fig. 5).

Potential Benefits: Especially in software-intensive systems, such an approach could be quite useful, for example, cars or industrial plants can be presented in a schematic manner that domain experts and end-users can understand.

Potential Drawbacks: However, this technique might not be so useful for technical details. Furthermore, visualizing cross-cutting dependencies can be hard in data maps.

5. DISCUSSION

The list of visualization techniques presented in this paper provides a base for research on product configuration in PLE and eventually for building more efficient product configuration tools that help users to cope with the complexity of variability models. However, the simple application of a visualization technique is not sufficient. Instead, visualizations have to be tailored to the user's tasks and the underlying data and have to be accompanied with appropriate interactive techniques. This section discusses these three aspects and their consequences for our goals. We finally propose an agenda for future work.

5.1 User Tasks

According to the basic principles of human-computer interaction, the usability of interactive tools relies on the precise tailoring of the tools to specific users and their specific tasks [33]. Consequently, visualization tools must be tailored to user tasks [38] and be evaluated by real users with real-world tasks [23].

The area of information visualization describes the general tasks to be supported by a visualization tool like searching, browsing, or comparison of data (see Section 5.3). However, some authors argue that "knowledge-making" activities, like learning or decision making, require more sophisticated support [1]. For example, support for visualizing complex queries on the data (e.g., statistical functions) or very domain-specific visualizations (e.g., specific to a stakeholder or company) is required. On the other hand, it has been argued that "good data speaks for itself" (see [1]). Indeed, information which is not too task-specific can be beneficial for the user to get important information one was not specifically looking for [38]. Also, a more general visualization which suits the needs of multiple stakeholders is beneficial to have a common basis for communication [38]. Finally, development effort is another argument to create tools with a certain level of generality.

According to our literature review, the user tasks for product configuration in PLE have not been much investigated yet, at least not on a level of granularity required for building tools. A first step is provided in [21] where four (rather abstract) tasks based on informal feedback from industry are listed: 1) *Retrieving the list of components for building a product*, 2) *Collect information about feature realization*, 3) *Retrieving the architectural structures of the feature set*,

and 4) *Comparing the architecture*.

We did not assume specific tasks in this paper and discussed visualizations according to aspects of variability models relevant for visualization (e.g., hierarchy, configuration workflow, etc.). However, there is the need for a systematic evaluation of the detailed tasks to be supported on a level of granularity appropriate for building tools.

5.2 Tailoring Visualizations

For each of the visualization techniques summarized in Table 1, a tool can be created. However, the identified techniques need to be tailored to the user's tasks and it might be useful to combine several of them depending on what is visualized.

A basic mantra in information visualization is *"Overview first, zoom and filter, then details-on-demand"* [32]. Overview refers to visualization techniques as discussed in this paper to show an overview on (some aspects of) the whole variability model. Here, data attributes are encoded by visual attributes like color, shape or orientation [38]. The appropriate encoding depends on the data type [39]. In case of variability models, the names of nodes (e.g., feature names) are usually important which means that a visualization must allow to provide sufficient space for readable text labels. Additional information can be added using additional visual attributes. As visualizing configuration decisions is usually not directly intended by most visualization techniques introduced in Section 4, it has to be added, e.g., using the color or the shape of nodes. Additional numerical attributes can be visualized, for instance, using color or size of nodes.

Detailed information – not required in the overview – can be represented as "details on demand", e.g., by a pop-up window or by overlaying the visualization with extra information for a selected element. There is often no need to see an overview of all cross-tree dependencies in a variability model; usually the user is interested only in the dependencies of the currently selected element. Consequently, a visualization could be simplified by showing cross-tree constraints on demand only.

5.3 Interaction

The efficiency of a visualization is strongly influenced by its interactive tool support [18]. This includes appropriate techniques to support Shneiderman's[32] mantra from above, i.e., filtering, zooming, and request of details on demand. Yi *et al.* [40] list seven interaction tasks: 1) Select, 2) Explore, 3) Reconfigure, 4) Encode, 5) Abstract/Elaborate, 6) Filter, and 7) Connect. These standard tasks must be systematically considered when creating tools. As shown in [18], well-implemented and easy to use standard functionality can significantly increase the performance of a tool. For instance, as textual labels are important in variability models, a simple textual search field can be very efficient to search for particular nodes and can be even more efficient with intelligent behavior like proposal of related terms.

5.4 Research Agenda

In summary, we propose the following research agenda:

1. Analyze the tasks of product configuration in PLE in detail and identify a common level of granularity for tool support.
2. Identify data to be shown as overview and data to be shown as details on demand (take the diverse visualization techniques described in this paper and their benefits and drawbacks into account).
3. Identify standard tasks and select tasks relevant for the concrete problem domain or area of interest.
4. Design and implement a tool, or, if possible, rather extend an existing tool.
5. Evaluate resulting tool through user tests with large-scale real-world product lines and draw conclusions about the feasibility of the visualization techniques.

6. CONCLUSIONS

In this paper we argued that (1) complexity is a limiting factor in the successful adoption of PLE, hence, (2) better techniques for handling large and complex product line models (i.e., in product configuration) are required and (3) visualization techniques are a potential solution for these challenges. We presented several sources for existing visualization techniques, e.g., well-known in the Software or Information Visualization communities. We selected a sub-set of these techniques and showed which are already used in product configuration in PLE and for which techniques it could make sense to further investigate them. We discussed the remaining research challenges – in particular regarding user tasks, tailoring visualizations, and interaction – and proposed next steps. In future work we plan to investigate selected techniques (and combinations of techniques) further, especially with large, real-world models like those from [7].

7. ACKNOWLEDGEMENTS

The authors would like to thank Mathias Schubanz for his useful input. This work was supported, in part, by Science Foundation Ireland grant 03/CE2/I303_1 to Lero – the Irish Software Engineering Research Centre, `http://www.lero.ie/`, by the Christian Doppler Forschungsgesellschaft, Austria, and Siemens VAI Metals Technologies.

8. REFERENCES

[1] R. A. Amar and J. T. Stasko. Knowledge precepts for design and evaluation of information visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 11:432–442, 2005.

[2] C. Atkinson, J. Bayer, C. Bunse, E. Kamsties, O. Laitenberger, R. Laqua, D. Muthig, B. Paech, J. Wüst, and J. Zettel. *Component-Based Product Line Engineering with UML*. Addison-Wesley, 2002.

[3] T. Barlow and P. Neville. A comparison of 2-d visualizations of hierarchies. In *INFOVIS*, pages 131–138. IEEE CS, 2001.

[4] J. Bayer, O. Flege, P. Knauber, R. Laqua, D. Muthig, K. Schmid, T. Widen, and J.-M. DeBaud. Pulse: a methodology to develop software product lines. In *SSR at ICSE 1999*, pages 122–131. ACM, 1999.

[5] L. Beaudoin, M.-A. Parent, and L. C. Vroomen. Cheops: a compact explorer for complex hierarchies. In *Visualization*, pages 87–ff. IEEE CS, 1996.

[6] D. Benavides, S. Segura, and A. Ruiz-Cortés. Automated analysis of feature models 20 years later. *Information Systems*, 35(6):615–636, 2010.

[7] T. Berger, S. She, R. Lotufo, A. Wasowski, and K. Czarnecki. Variability modeling in the real: A perspective from the operating systems domain. In *25th International Conference on Automated Software Engineering*, pages 73–82, Antwerp, Belgium, 2010. ACM.

[8] S. K. Card, J. D. Mackinlay, and B. Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, 1999.

[9] K. Czarnecki, S. Helson, and U. Eisenecker. Staged configuration using feature models. In *SPLC*, pages 266–283. Springer, 2004.

[10] S. Deelstra, M. Sinnema, and J. Bosch. Product derivation in software product families: a case study. *Journal of Systems and Software*, 74(2):173–194, 2005.

[11] D. Dhungana, P. Grünbacher, and R. Rabiser. The DOPLER meta-tool for decision-oriented variability modeling: A multiple case study. *Automated Software Engineering*, 18(1):77–114, 2011.

[12] D. Dhungana, P. Grünbacher, R. Rabiser, and T. Neumayer. Structuring the modeling space and supporting evolution in software product line engineering. *Journal of Systems and Software*, 83(7):1108–1122, 2010.

[13] O. Djebbi, C. Salinesi, and G. Fanmuy. Industry survey of product lines management tools: Requirements, qualities and open issues. In *RE*, pages 301–306. IEEE CS, 2007.

[14] ESI Spain and IKV++ Technologies AG Germany. MASTER: Model-driven architecture instrumentation, enhancement and refinement. Technical report, MASTER-2002-D1.1-V1-PUBLIC, 2002.

[15] V. Friedman. Data visualization: Modern approaches. Website, 2007. `http://www.smashingmagazine.com/2007/08/02/data-visualization-modern-approaches/`.

[16] H. Gomaa. *Designing Software Product Lines with UML*. Addison-Wesley, 2005.

[17] I. Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, 2000.

[18] A. Kobsa. User experiments with tree visualization systems. In *InfoVis*, pages 9–16. IEEE CS, 2004.

[19] R. Lengler and M. Eppler. Towards a periodic table of visualization methods for management. In *GVE*. ACTA Press, 2007.

[20] T. Munzner. Drawing large graphs with h3viewer and site manager. In *Graph Drawing*, pages 384–393. Springer, 1998.

[21] D. Nestor, L. O'Malley, P. Healy, A. Quigley, and S. Thiel. Visualisation techniques to support derivation tasks in software product line development. In *CASCON*, pages 315–325. ACM, 2007.

[22] Q. V. Nguyen and M. L. Huang. Enccon: an approach to constructing interactive visualization of large hierarchical data. *Information Visualization*, 4:1–21, 2005.

[23] C. Plaisant. The challenge of information visualization evaluation. In *AVI*, pages 109–116. ACM, 2004.

[24] C. Plaisant, J. Grosjean, and B. B. Bederson. Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation. In *InfoVis*, pages 57–64. IEEE CS, 2002.

[25] R. Rabiser, D. Dhungana, W. Heider, and P. Grünbacher. Flexibility and end-user support in model-based product line tools. In *Euromicro SEAA*, pages 508–511. IEEE CS, 2009.

[26] R. Rabiser, P. Grünbacher, and D. Dhungana. Requirements for product derivation support: Results from a systematic literature review and an expert survey. *Information and Software Technology*, 52(3):324–346, 2010.

[27] M. Rosa, W. van der Aalst, M. Dumas, and A. te Hofstede. Questionnaire-based variability modeling for system configuration. *Software and System Modeling*, 8(2):251–274, 2009.

[28] K. Schmid. A comprehensive product line scoping approach and its validation. In *ICSE*, pages 593–603. ACM, 2002.

[29] K. Schmid and I. John. A customizable approach to full-life cycle variability management. *Science of Computer Programming, Special Issue on Variability Management*, 53(3):259–284, 2004.

[30] K. Schmid, R. Rabiser, and P. Grünbacher. A comparison of decision modeling approaches in product lines. In *VaMoS*, pages 119–126. ACM, 2011.

[31] D. Schneeweiss and G. Botterweck. Using flow maps to visualize product attributes during feature configuration. In *VISPLE*, pages 219–228. Lancaster Univ., 2010.

[32] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *VL*, pages 336–343. IEEE Computer Society, 1996.

[33] B. Shneiderman and C. Plaisant. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison Wesley, 4 edition, 2004.

[34] M. Sinnema and S. Deelstra. Classifying variability modeling techniques. *Information and Software Technology*, 49(7):717–739, 2006.

[35] M. Torres, U. Kulesza, M. Sousa, T. Batista, L. Teixeira, P. Borba, E. Cirilo, C. Lucena, R. Braga, and P. Masiero. Assessment of product derivation tools in the evolution of software product lines: an empirical study. In *FOSD*, pages 10–17. ACM, 2010.

[36] P. Trinidad, A. Ruiz-Cortes, D. Benavides, and S. Segura. Three-dimensional feature diagrams visualization. In *ViSPLE*, pages 295–302. Lero, 2008.

[37] F. van der Linden, K. Schmid, and E. Rommes. *Software Product Lines in Action - The Best Industrial Practice in Product Line Engineering*. Springer, 2007.

[38] C. Ware. *Information Visualization*. Morgan Kaufmann, 2nd edition, 2004.

[39] S. Wehrend and C. Lewis. A problem-oriented classification of visualization techniques. In *VIS*, pages 139–143. IEEE CS, 1990.

[40] J. S. Yi, Y. a. Kang, J. Stasko, and J. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13:1224–1231, 2007.