Four-Step Automated Analysis of Feature Models
Student Conference on Software Engineering and Database Systems

Reimar Schröter

June 29, 2012
Software Product Lines (SPL)

**SPL:** Similar programs on a common code base

**Feature Models:** Hierarchical arrangement for commonalities and differences of SPL products
Software Product Lines (SPL)

**SPL:** Similar programs on a common code base

**Feature Models:** Hierarchical arrangement for commonalities and differences of SPL products

```
class List {
  mandatory feature Base;
  feature Debug;
  feature Synchronize;
  feature Sort {
    oneof {
      feature MergeSort;
      feature QuickSort;
    }
  }
}
```

```
constraint Synchronize -> MergeSort;
```

VELVET, see Rosenmüller et al. [2011]
Automated Analysis, see Benavides et al. [2010]

Necessity

- Crosstree constrains, i.e.: List $\rightarrow$ MergeSort
- Real world feature models are very huge, see Batory et al. [2006]

Void Feature Model: SPL represents no product
Automated Analysis, see Benavides et al. [2010]

Necessity

- Crosstree constrains, i.e.: List -> MergeSort
- Real world feature models are very huge, see Batory et al. [2006]

Void Feature Model: SPL represents no product
Core Features (CF): feature is present in all products
Automated Analysis, see Benavides et al. [2010]

Necessity

- Crosstree constrains, i.e.: List -> MergeSort
- Real world feature models are very huge, see Batory et al. [2006]

Void Feature Model: SPL represents no product

Core Features (CF): feature is present in all products

False-Optional Feature: feature is non mandatory, but is CF
Automated Analysis, see Benavides et al. [2010]

**Necessity**

- Crosstree constrains, i.e.: List $\rightarrow$ MergeSort
- Real world feature models are very huge, see Batory et al. [2006]

![Feature Model Diagram]

**Void Feature Model:** SPL represents no product

**Core Features (CF):** feature is present in all products

**False-Optional Feature:** feature is non mandatory, but is CF

**Dead Feature:** feature is not presented in any product
Automated Analysis - Two-Step Approach

Concept List

\[
\begin{align*}
\text{concept List}\{ & \quad // l \\
& \quad \ldots \\
& \quad \text{feature Debug}; // d \\
& \quad \text{feature Sort}; // s \\
& \quad \ldots \\
\}
\end{align*}
\]

Translation in propositional formulas

\[
\begin{align*}
& l \land \ldots \land (d \rightarrow l) \land (s \rightarrow l) \\
& \land \ldots
\end{align*}
\]

Execution of Analysis

Results

But what happens, if errors are already in the input feature models? Core Features?
Automated Analysis - Two-Step Approach

But what happens, if errors are already in the input feature models? ➞ Core Features?

Translation in propositional formulas

Results
Examples of Possible Problems

(E1): *Duplicated group definition*

→ Which group is used for analyses?

```
concept List{
  feature Sort{
    oneof{
      somewhere{
        // ...     
      }
    }
  }
}
```
Examples of Possible Problems

\( (E1): \) *Duplicated group definition*

\[ \rightarrow \text{Which group is used for analyses?} \]

\[ \text{concept List}\{ \]
\[ \text{mandatory feature Synchronize;} \]
\[ \text{feature Debug;} \]
\[ \text{feature Synchronize;} \]
\[ \text{// ...} \]
\[ \} \]

\( (E2): \) *Duplicated feature name*

\[ \rightarrow \text{Which feature \textit{Synchronize} (optional or mandatory) is used.} \]

\[ \text{concept List}\{ \]
\[ \text{feature Sort}\{ \]
\[ \text{oneof}\{ \]
\[ \text{someof}\{ \]
\[ \text{// ...} \]
\[ \} \} \} \]
Examples of Possible Problems

**(E1): Duplicated group definition**

→ Which group is used for analyses?

```plaintext
concept List{
  feature Sort{
    oneof{
      someone{
        // . . .
      }
    }
  }
}
```

**(E2): Duplicated feature name**

→ Which feature `Synchronize` (optional or mandatory) is used.

```plaintext
concept List{
  mandatory feature Synchronize;
  feature Debug;
  feature Synchronize;
  // . . .
}
```

**(E3): Void feature model**

→ What happens, if it is not detected before the automated analysis.

```plaintext
concept List{
  // . . .
  feature Sort{
    oneof{
      feature MergeSort;
      feature QuickSort;
    }
  }
  constraint Sort . MergeSort;
  constraint Sort . QuickSort;
}
```
Tool Evaluation

**S.P.L.O.T.**, see Mendonca et al. [2009]
- Set of tools for editing, configuring, analyzing of feature models

**FeatureIDE**, see Thüm et al. [2012]
- Supports creating and editing of feature models
Tool Evaluation

**S.P.L.O.T.**, see Mendonca et al. [2009]
- Set of tools for editing, configuring, analyzing of feature models

**FeatureIDE**, see Thüm et al. [2012]
- Supports creating and editing of feature models

<table>
<thead>
<tr>
<th></th>
<th><strong>S.P.L.O.T.</strong></th>
<th><strong>FeatureIDE</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>(E1)</td>
<td>X no error message, no results</td>
<td>X contradictory results</td>
</tr>
<tr>
<td>(E2)</td>
<td>X contradictory results</td>
<td>✓ process canceled</td>
</tr>
<tr>
<td>(E3)</td>
<td>✓</td>
<td>X root marked as dead or false optional</td>
</tr>
</tbody>
</table>
### Tool Evaluation

**S.P.L.O.T.**, see Mendonca et al. [2009]
- Set of tools for editing, configuring, analyzing of feature models

**FeatureIDE**, see Thüm et al. [2012]
- Supports creating and editing of feature models

<table>
<thead>
<tr>
<th></th>
<th>S.P.L.O.T.</th>
<th>FeatureIDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>(E1)</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>no error message, no results</td>
<td>contradictory results</td>
</tr>
<tr>
<td>(E2)</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>contradictory results</td>
<td>process canceled</td>
</tr>
<tr>
<td>(E3)</td>
<td>✓</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>root marked as dead or false optional</td>
<td></td>
</tr>
</tbody>
</table>

⇒ Two-step approach is insufficient
Four-Step Approach for Automated Analysis

Model-Parsing

Analyses

Satisfiability Check

Syntactical Checks

Because of limited expressiveness of grammars

Typos in names, incorrect keyword usage

(E1)

Is feature model a valid model?
Pre-condition of further analysis

(E3)

Analyses, such as core features can be executed

(E2)
Four-Step Approach for Automated Analysis

Model-Parsing

- Parse textual feature model
- Detect typos, incorrect keyword usage \( \rightarrow \) (E1)

Syntactical Checks

Because of limited expressiveness of grammars
- Typos in names, incorrect keyword usage

Satisfiability Check

Is feature model a void model?
- Pre-condition of further analysis

Analyses

Analyses, such as core features can be executed
Four-Step Approach for Automated Analysis

Model-Parsing
- Parse textual feature model
- Detect typos, incorrect keyword usage ➞ (E1)

Syntactical Checks
- Because of limited expressiveness of grammars
- Typos in names, incorrect keyword usage ➞ (E2)
Four-Step Approach for Automated Analysis

Model-Parsing

- Parse textual feature model
- Detect typos, incorrect keyword usage $\Rightarrow$ (E1)

Syntactical Checks

- Because of limited expressiveness of grammars
- Typos in names, incorrect keyword usage $\Rightarrow$ (E2)

Satisfiability Check

- Is feature model a void model?
- Pre-condition of further analysis $\Rightarrow$ (E3)
Four-Step Approach for Automated Analysis

Model-Parsing
- Parse textual feature model
  ➔ Detect typos, incorrect keyword usage ➔ (E1)

Syntactical Checks
- Because of limited expressiveness of grammars
  ➔ Typos in names, incorrect keyword usage ➔ (E2)

Satisfiability Check
- Is feature model a void model?
  ➔ Pre-condition of further analysis ➔ (E3)

Analyses
- Analyses, such as core features can be executed
Summary and Conclusion

- Automated Analysis for Feature Models are very important
  - Existing tool support with two-step approach ➔ insufficient (see S.P.L.O.T., FeatureIDE)
Summary and Conclusion

- Automated Analysis for Feature Models are very important
  - Existing tool support with two-step approach → insufficient (see S.P.L.O.T., FeatureIDE)

Introduction of the four-step approach

- Prevents undesired results
- Increases probability of feature model correctness
Thank you for your attention!
Questions?

