Cardinality-Based Feature Models and Their Staged Configuration

Krzysztof Czarnecki, Simon Helsen
University of Waterloo

Ulrich W. Eisenecker
University of Applied Sciences Kaiserslautern, Zweibrücken
Overview

Background
• Cardinality-Based Feature Modeling
• Staged Configuration & Specialization
• Related work
• Conclusions and future work
Feature Modeling

• Capturing common and variable features of a product line
• Introduced in Feature-Oriented Domain Analysis (FODA)
• Several uses in product-line engineering
  – Product-line scoping and analysis
  – Identifying architectural variation points
  – Product derivation
  – …
Feature Diagram in FODA

Rationale:
Manual more fuel efficient

Composition rule:
Air conditioning requires Horsepower > 150.
Another Notation From CE00
Combinations & Normalization

• Optional or mandatory features in a group
  – Top-down interpretation
• Lead to redundant representations
• Normalization

Normalized diagrams
Overview

• Background

Cardinality-Based Feature Modeling

• Staged Configuration & Specialization

• Related work

• Conclusions and future work
Cardinality-Based Feature Modeling

• Superset of the previous notations
  – Feature cardinalities
  – Group cardinalities
  – Feature attributes
  – Feature diagram references

• Imposes a special constraint
  – Grouped features have no feature cardinalities
Example
Semantics of Feature Diagrams

- A feature diagram denotes a set of all valid configurations
- A configuration is a *structured* set of features constructed according to the informal interpretation rules

```
(f, {(f1, ∅), (f3, ∅), (f3, {(f4, ∅), (f4, ∅)})})
```
Formal Semantics

• We give a formal semantics by translation to context-free grammars

• Two-level approach
  – Function
    • Takes an abstract-syntax representation of feature diagram
    • Yields a context-free grammar
  – Each sentence produced by the grammar is a string representing a valid configuration
    • E.g., (f, {((f1, ∅), (f3, ∅), (f3, {((f4, ∅), (f4, ∅))})})

• See technical report for detail
Sample UI for Configuration

- Feature Model
  - securityProfile
    - passwordPolicy
      - expiration
        - inDays (Integer)
        - never
    - chars <2-4>
      - lowerCase
      - upperCase
      - digit
      - specialChar
    - [0..*] permissionSet (String)
  - fileO
    - unrestricted
    - restricted
      - [0..*] filepath (String)
        - ref: permission
  - fileDialog
    - open
    - close
  - environmentVariables
    - ref: permission
- Configuration
  - securityProfile
    - passwordPolicy
      - expiration
        - inDays ('30' : Integer)
        - never
    - chars <2-4>
      - lowerCase
      - upperCase
      - digit
      - specialChar
    - [0..*] permissionSet (String)
  - permissionSet ('Internet' : String)
    - fileO
      - unrestricted
      - restricted
        - [0..*] filepath (String)
        - ref: permission
    - fileDialog
      - open
      - close
    - environmentVariables
      - permission
Overview

• Background
• Cardinality-Based Feature Modeling
  ➔ Staged Configuration & Specialization
• Related work
• Conclusions and future work
Staged Configuration

• Decisions taken in stages
  – Time, e.g.,
    • Phases in product lifecycle
  – Roles, e.g.,
    • Component integrator, deployer, administrator, maintainer
  – Context/target, e.g.,
    • Multiple deployment contexts

• Arises different in practical contexts
  – Software supply chains, e.g.,
    • Component supplier pre-configures its product for different system integrators
    • Further configuration by each system integrator for different end-products
  – Optimizations, e.g.,
    • Component optimized based on configuration decisions at compile-time
    • Further optimization based on configuration decisions at deployment time
  – Policy standards, e.g.,
    • Enterprise-wide security policy specialized by individual departments
    • Further specialization for each computer within a department
Staged Configuration Using Specialization

- Specialization transforms a new feature diagram into a new one denoting a subset of configurations of the original diagram

```
(f, {(f2, ∅), (f4, ∅)})
```
Specialization Steps 1-4

1. Refine feature cardinality
   - \( f \) \[1..5\] \( f_1 \)

2. Remove grouped feature \( f_3 \)
   - \( f \) \( f_1 \) \( f_2 \) \( f_3 \)

3. Select grouped feature \( f_3 \)
   - \( f \) \( f_1 \) \( f_2 \) \( f_3 \)
Specialization Steps 2-7

Assign attribute value

Clone solitary feature

Unfold feature diagram reference
Sample Specialization UI
Current Work: Multilevel Configuration

• Each stage has a separate feature model
• Each role performs configuration (not specialization) within a stage
• Manual configuration of one stage automatically specializes the feature model of the next stage
Staged Configuration

• Different roles will have different perspectives on the variability

Product-line perspective  System configuration perspective

certTax  taxGateway  cyberSource

certTax  taxGateway

taxExpress  calculationMethod
certTax  serviceld(String)
cyberSource  reportTaxBreakdown
certTax  totalPrice

certTax  certTax

taxExpress  serviceld(String)

certTax  totalPrice

certTax  certTax

taxExpress  serviceld(String)

certTax  totalPrice

Just annotating features with binding times is not enough!
Current Work: Multilevel Configuration

Stage 0
- Level 0 configuration
  - Manual configuration by stage-0 role

Stage 1
- Level-0 configuration
  - Automatic specialization based on level-0 configuration
  - Specialized level-1 feature model

Stage 2
- Level-1 configuration
  - Manual configuration by stage-1 role
  - Automatic specialization based on level-1 configuration
  - Specialized level-2 feature model
  - Manual configuration by stage-2 role
  - Level-2 configuration
Overview

- Background
- Cardinality-Based Feature Modeling
- Staged Configuration & Specialization
- Related work
- Conclusions and future work
Related Work

• Cardinality-Based Feature Modeling
  – Group cardinalities - Riebisch et al 2002
  – Feature cardinalities and attributes - Czarnecki et al 2002
  – Feature diagram references - Bednasch 2002

• Formal semantics
  – Bontemps et al 2004 – SPLC04 Workshop on Derivation

• Existing tools
  – Academic: AmiEddi, CaptainFeature, FeaturePlugin, …
  – Commercial: Pure::Variants, GEARs
  – Alternative: just use XML – Pasetti et al 2004
Overview

• Background
• Cardinality-Based Feature Modeling
• Staged Configuration & Specialization
• Related work

→ Conclusions and future work
Conclusions & Future Work

• Cardinality-based feature modeling balances expressiveness and simplicity
• Staged configuration supports more complex configuration scenarios (e.g., software supply chains)
• Current and future work
  – Synchronization between models, specializations and configurations
  – Staged configuration of feature models with constraints
  – FeaturePlugin release by OOPSLA’04
    • Michal Antkiewicz
Questions…