

# Efficient Variability Treatment Based on XML

---

Martin Becker  
System Software Group  
University of Kaiserslautern, Germany  
[mbecker@informatik.uni-kl.de](mailto:mbecker@informatik.uni-kl.de)



# Background

- SFB 501:
  - „Development of large systems with generic methods“
  - Domains: building automation, automobile
  - Projects:
    - Composition of customized run-time platforms
    - Application engineering with core-systems

## ⇒ Family (Variability) Engineering

- Premises:
  - Static variability
  - Reuse-based
  - Restricted domains (limited variability, accepted principles, domain knowledge)

# Objectives

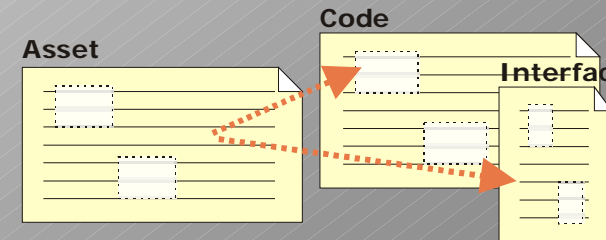
- Problems:
  - Software reuse → similarity → *Variability*
  - Softness of software → *Evolution*
  - Evolution of variability is expensive and error prone:
    - Interdependencies, implicit evolution, unforeseen changes
    - Lacking/implicit former design knowledge/assumptions
    - Impact analysis is hard
  - Variability resolved on the code-level
- Approach:
  - Uniform treatment of variability → *XML*
  - Explicit treatment of variability → *Variability Model*
  - Evolution support → *Analysis, Modification, Assimilation*
- Premises:
  - Consistency
  - Efficiency

# Variability in XML Documents

- Why XML?
  - Clear separation of *content*, *structure* and *representation*
  - Increasing pervasion/relevance of XML-based documents in SE
  - Linking/inclusion support (XLink)
  - Addressing of document fragments (XPath/XQuery)
  - Extensibility
- Existing Techniques
  - Entities: only simple parameterisation
  - XSLT: complicated (parameterisation, identification of VP)
- VML (Variability Markup Language)
  - Features:
    - Clear identification of VPs
    - Simple parameterisation (profile, macros)
    - Selection of prefabricated solutions
    - (Partial) inclusion of documents
    - Scripting support (generators)
    - Preview, partial resolution

# Evolution Support

- Analysis:
  - Origins, interdependencies, binding times, complexity of variabilities → *Variability Model*
  - Explicit VPs: *extension*,  $VP \Leftrightarrow \text{variabilities}$  → *special markup*
  - Uniform resolution specification → *VML*
  - Compact instance representation → *Profile*
  - Augmented derivation → *Repository*
- Modification:
  - Variability-oriented document organizations: *separation* of content and representation
  - View-based editing
- Assimilation:
  - Merging modified instances: *XML-Diff*
  - Compact representation of modifications → *Delta*



# Conclusion and Further Work

- *Efficient Variability Treatment based on XML*
- Approach:
  - Exploit XML-related techniques for efficient variability treatment
  - Explicit variability modeling
  - Overall variability treatment
- Results:
  - ✓ Variability Model
  - ✓ VML-based embedded operating system (OSEK-compliant)
- Further work:
  - Further investigate evolution support
  - Consolidate approaches
  - (Advance tool support)

# Thank You

