Reconciling Feature Modeling: A Feature Modeling Metamodel

Valentino Vranić

www.fiit.stuba.sk/~vranic, vranic@fiit.stuba.sk

Institute of Informatics and Software Engineering

Faculty of Informatics and Information Techologies

Slovak University of Technology

Net.ObjectDays 2004 Contributed Paper Presentation

Introduction

- Dealing with variability is an important issue
- Feature modeling enables to capture variability in a domain
- Used widely in domain engineering and product line approaches
- The problem of different feature modeling notations

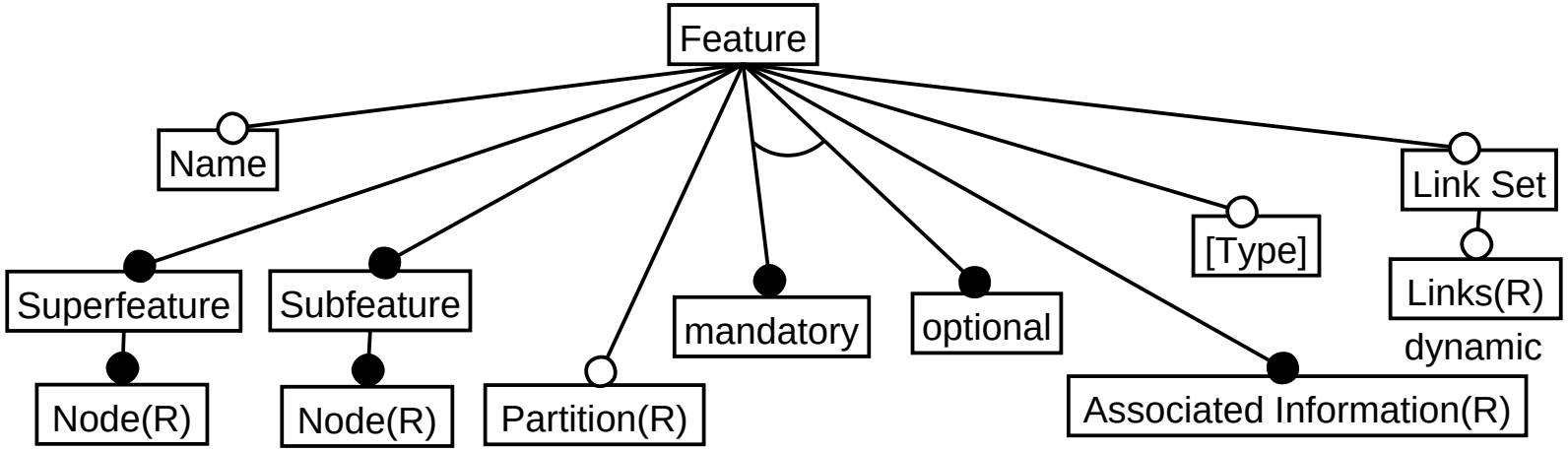
Presentation Overview

- Feature modeling
- Feature modeling metamodel
- Feature modeling for multi-paradigm design
  - Concept instantiation
  - Parameterization in feature models
  - Cardinality in feature models
- Summary and further research

Feature Modeling

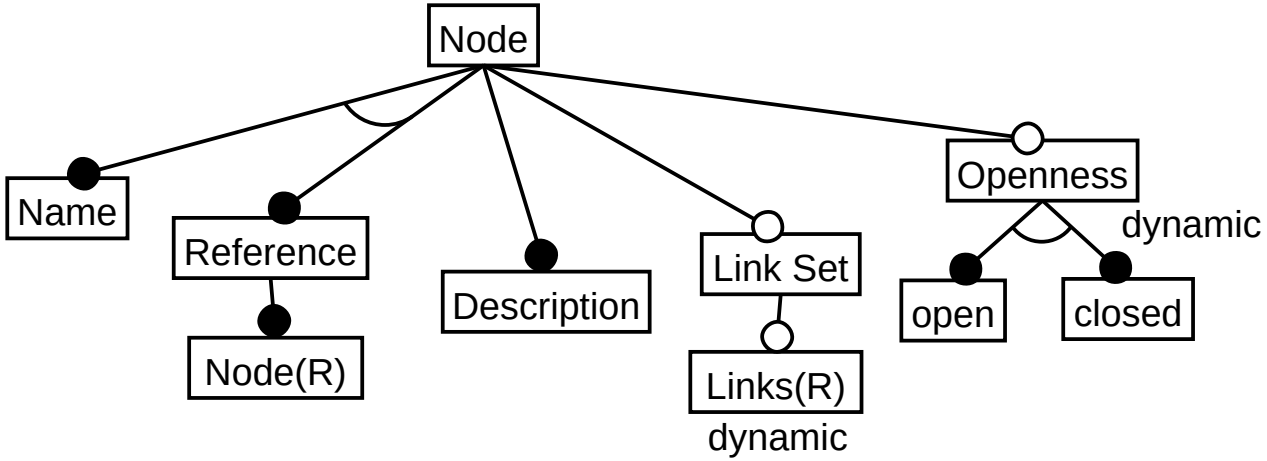
- Captures feature interdependencies and variability
- Feature model: a set of feature diagrams plus further information
- Based on the notions of domain, concept, and feature
  - Features: common and variable
  - Concept instances: concept specializations
- Different notations being used, such as FODA, ODM, Czarnecki-Eisenecker, and feature modeling for multi-paradigm design

Feature



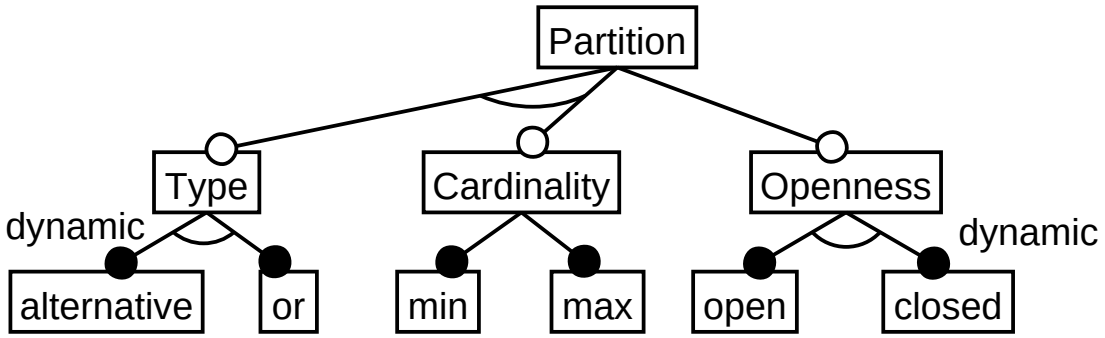
- A relationship between two feature diagram nodes
- A feature is mandatory (filled circle ended edges) or optional (empty circle ended edges)
- May be linked to other modeling artifacts
- In some approaches (FODA, FORM) features are being named and classified

Node



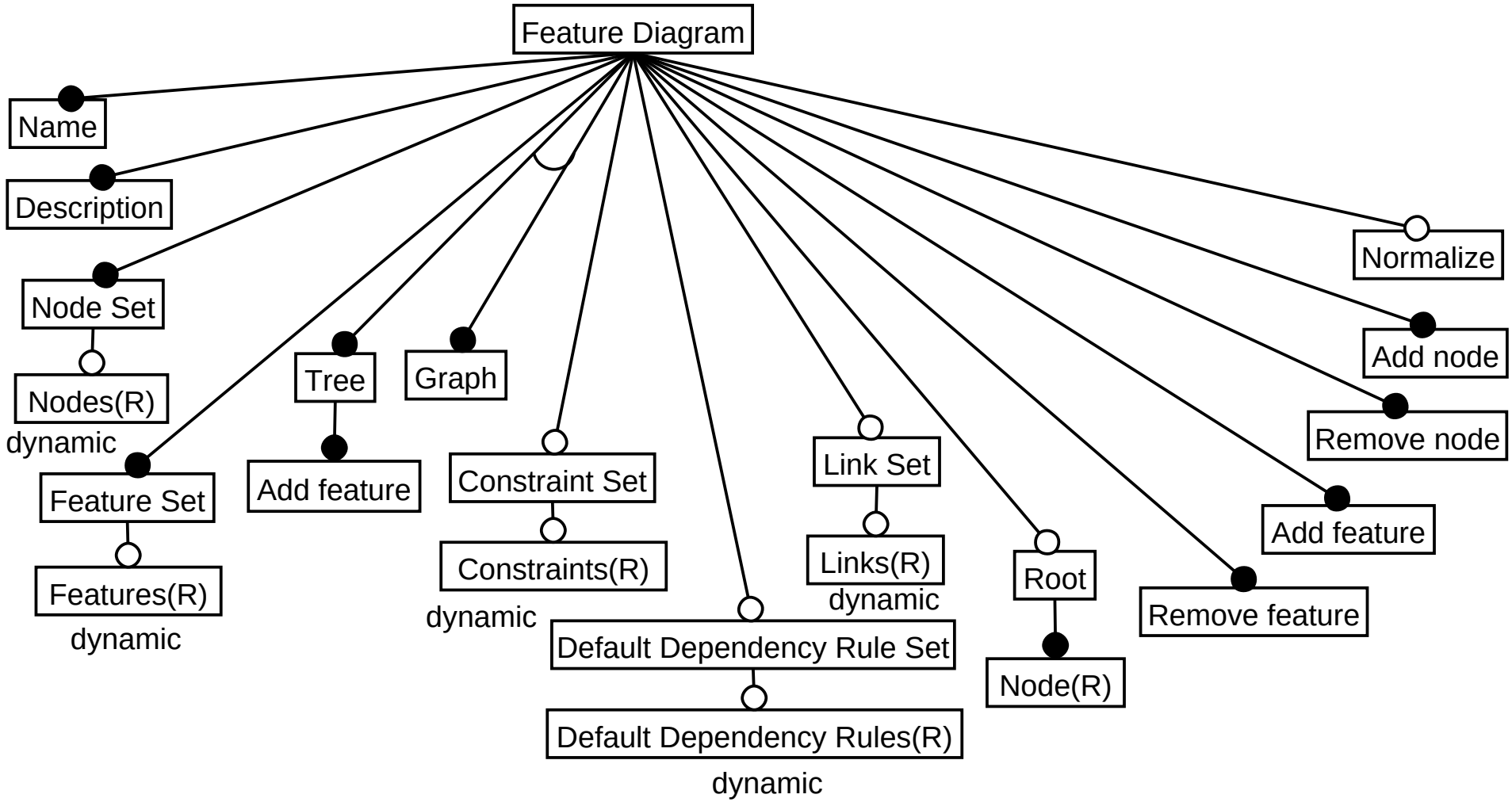
- Represents a concept in a general sense
- A node is either named or a reference to another node (concept reference Ⓡ; appears as (R) in diagrams)
- May be open (its new variable subfeatures are expected)
- May also be linked to other modeling artifacts

Partition



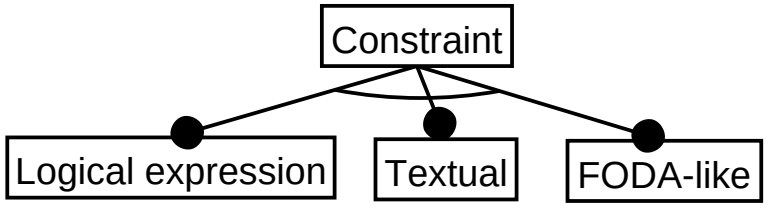
- Denoted by an arc in diagrams
- Presumed meaning is alternativity (original FODA meaning)
- Some approaches (Czarnecki-Eisenecker notation and related) support or-features (filled arc)
- Cardinality (updated Czarnecki-Eisenecker notation) allows for the greatest flexibility

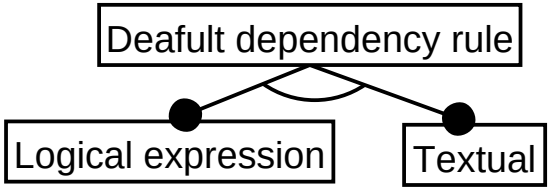
Feature Diagram



Constraint: $\neg$*Root.Node.Reference*

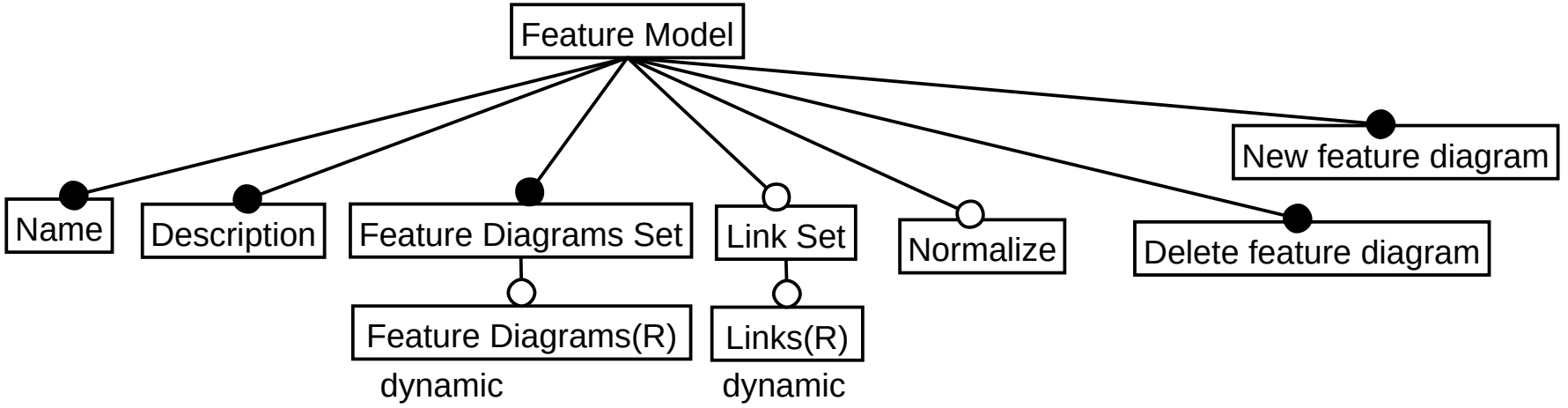
Constraint and DDR



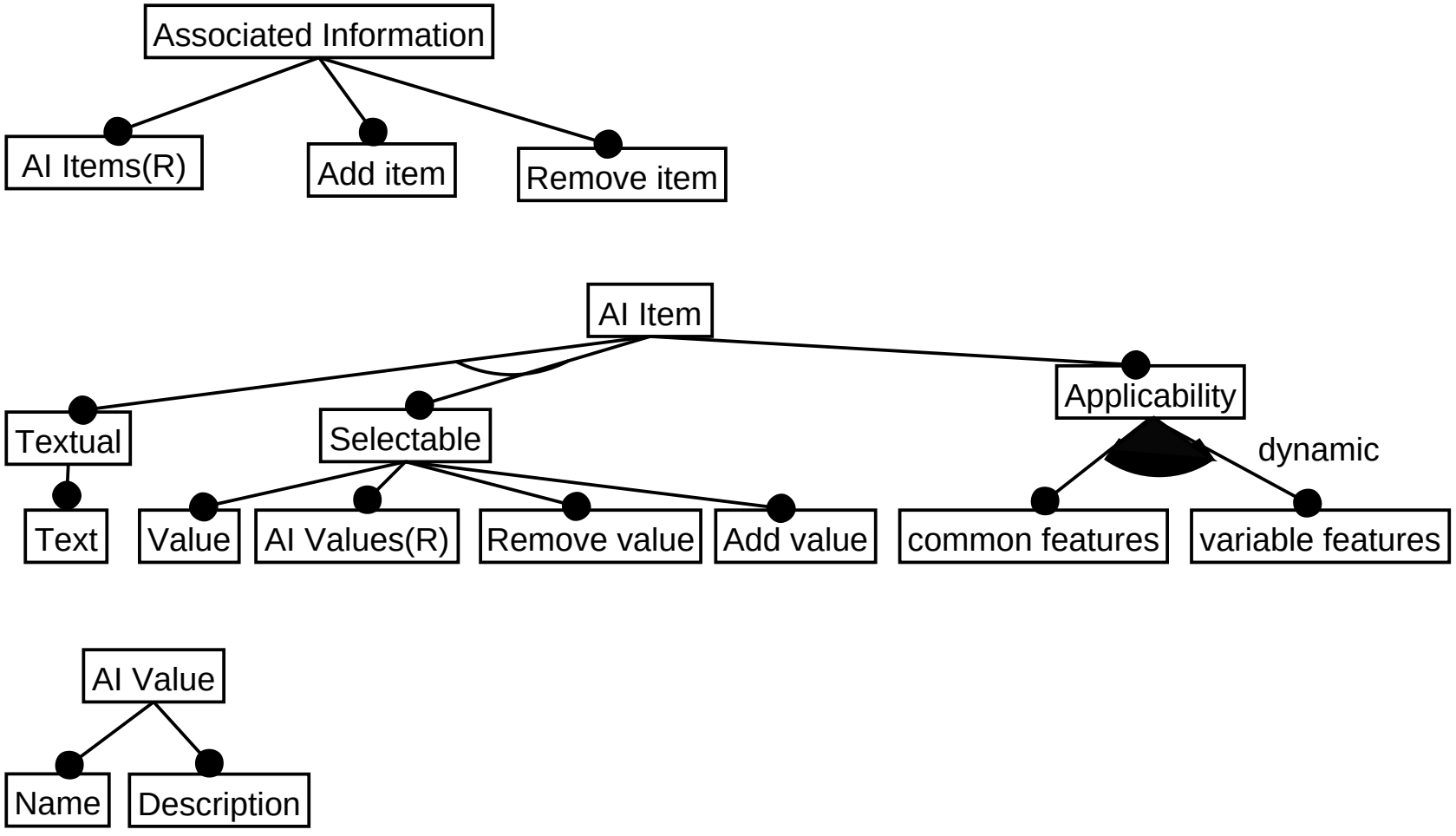


- Constraints
  - Mutual exclusions and requirements among features
- Default dependency rules
  - Which features should appear together by default

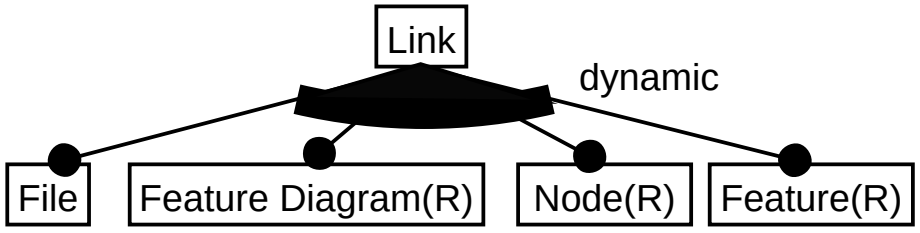
Feature Model



Associated Information



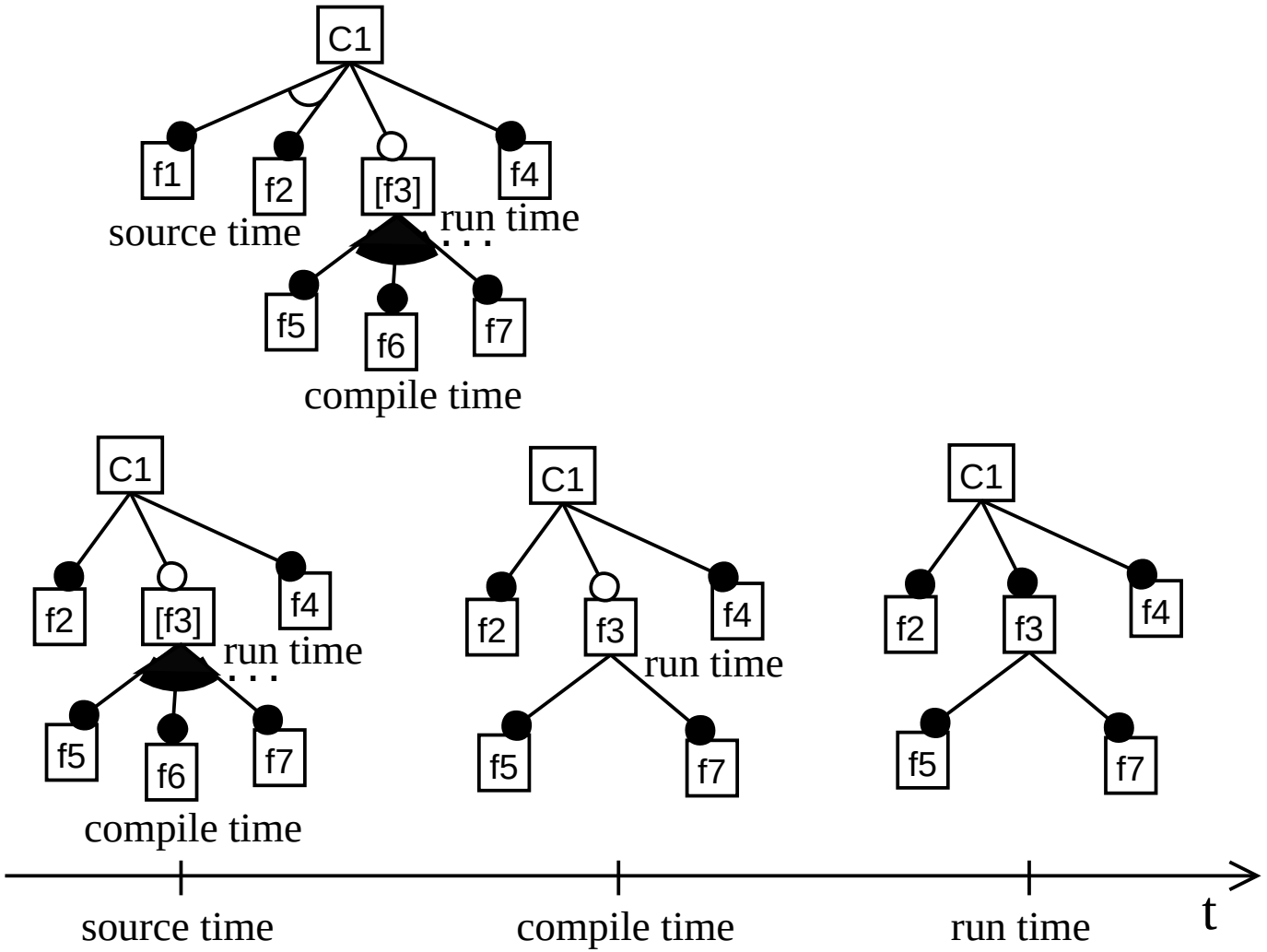
Link



Constraint: $Node \veebar Feature$

- Connecting a feature model or its parts to
  - Its own diagrams, nodes, and features
  - Other feature models or models in general

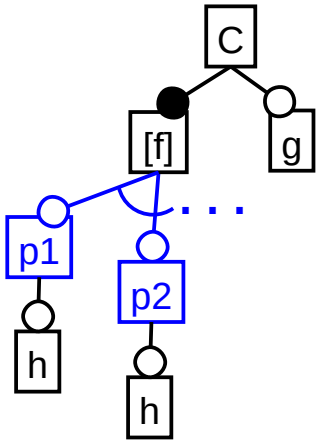
Concept Instantiation



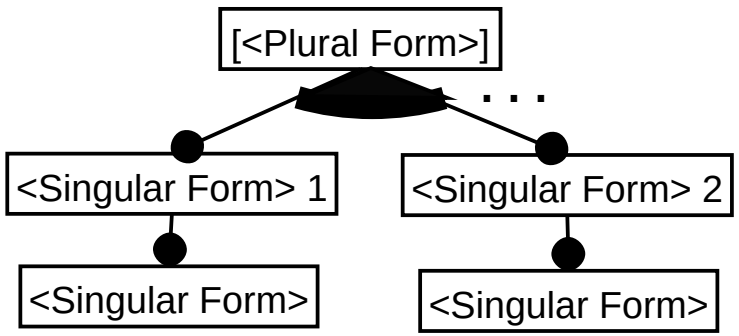
Parameterization in Feature Models

- Parameterized feature and concept names

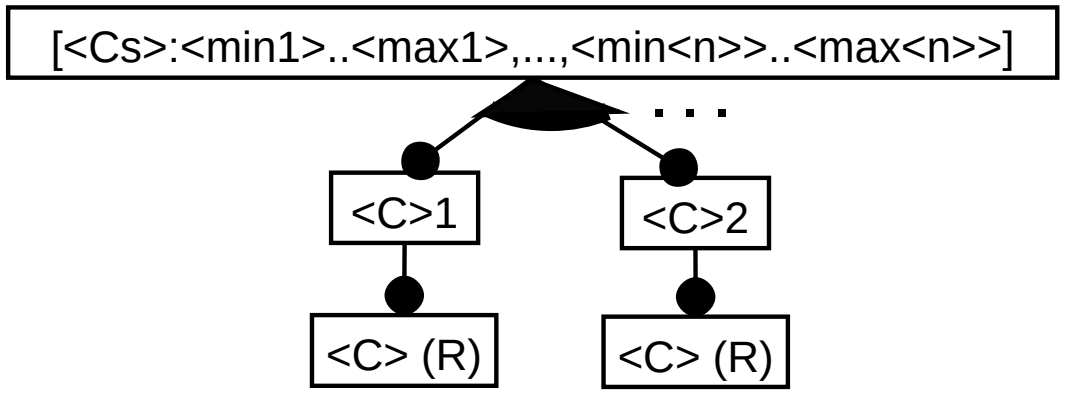
  Constraint: $\forall <i> \in N\ p<i>.h \underline{\vee} g$

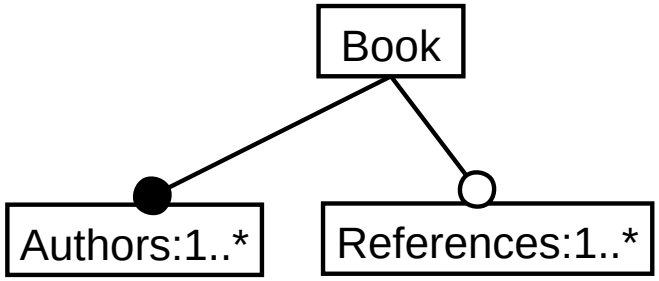


- Parameterized concepts



Cardinality in Feature Models





$$\bigvee_{<i>=1}^{<n>} ((\text{max}<i> \neq * \Rightarrow \bigvee_{<j>=<min<i>>}^{<max<i>>-<min<i>>+1} \bigwedge_{k=1}^{i} <C><k>) \wedge$$

$$\wedge (\text{max}<i> = * \Rightarrow \bigwedge_{k=1}^{<min<i>>} <C><k>))$$

Summary

- Feature modeling metamodel
  - Based on various approaches to feature modeling
  - May be used for further reasoning and as a basis for feature modeling tools
- Feature modeling for multi-paradigm design
  - An integrative approach feature modeling
  - Main improvements:
    - Concept instantiation with respect to instantiation time
    - Parameterization in feature models
    - Constraints and default dependency rules as logical expressions
    - Concept references

Further Research

- Enhancing parameterization with respect to binding time/mode
  - Binding time/mode as a parameter
  - Would improve concept reusability
- Expressing feature diagrams fully in the form of constraints
  - Represented by logical expressions
  - Primary constraints should be defined (for visualization purposes)