Binding Time Based Concept Instantiation in Feature Modeling

9th International Conference on Software Reuse — ICSR 2006
Turin, Italy

Valentino Vranić and Miloslav Šípka

Institute of Informatics and Software Engineering
Faculty of Informatics and Information Technology
Slovak University of Technology in Bratislava

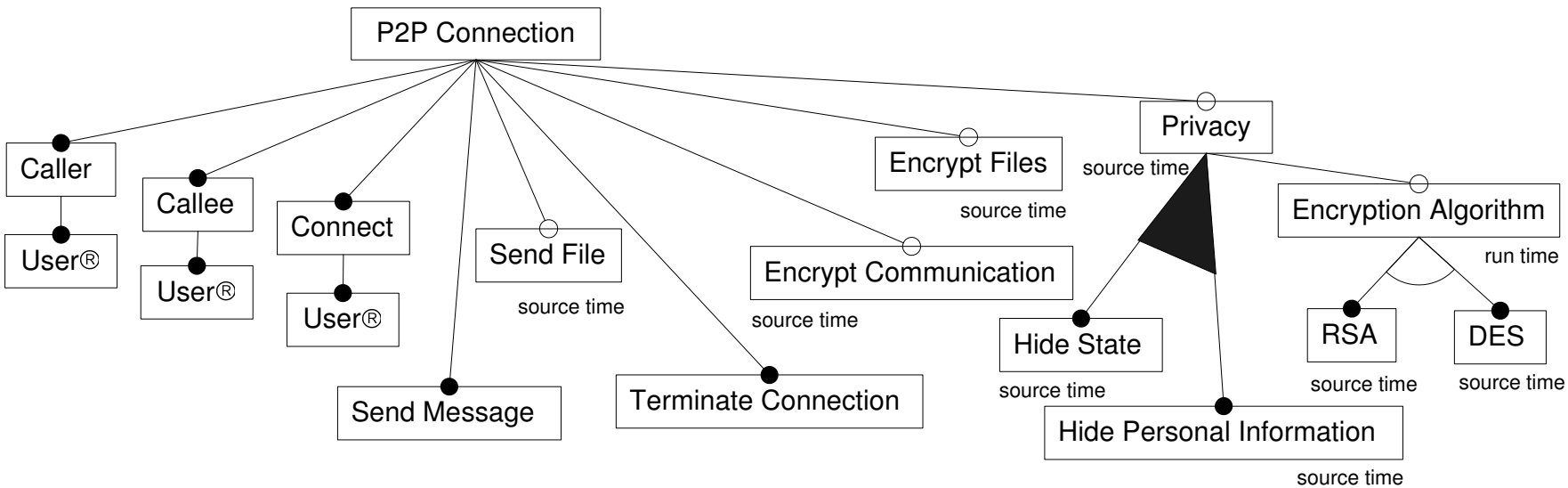
June 13, 2006

Overview

Feature Model

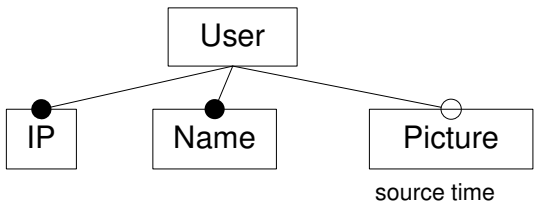
- Concepts expressed by their features
- A feature is an important property of a concept
- Common and variable features
- Focus on configurability
- A specific configuration of a system or its part are represented by concept instances
- Czarnecki-Eisenecker notation

An Example: Peer-to-Peer Chat Protocol



Concept References

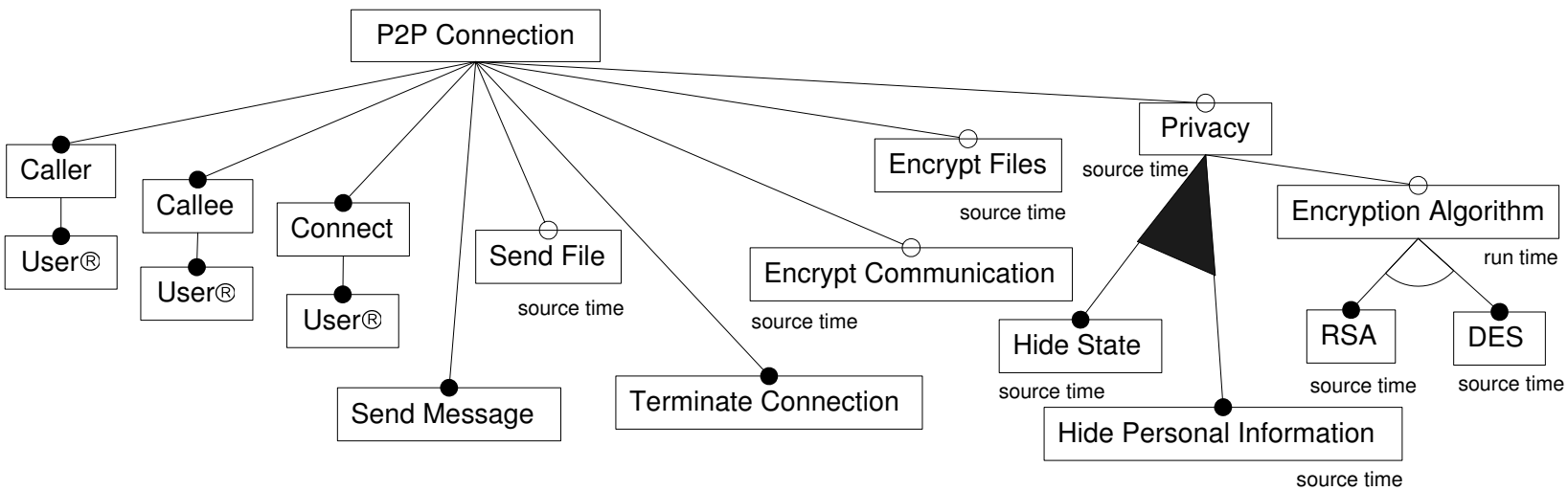
- A concept reference is equivalent with the repetition of the whole feature diagram of the referenced concept
- Peer-to-peer chat protocol references the User concept



Constraints

- Main constraints are expressed by feature diagrams
- Additional constraints — in form of predicate logic expressions
- Additional constraints in the peer-to-peer chat protocol:

$$\textit{Encrypt Communication} \Rightarrow \textit{Encryption Algorithm}$$
$$\textit{Encrypt Files} \Rightarrow \textit{Encryption Algorithm}$$
$$\textit{Encrypt Files} \Rightarrow \textit{Send File}$$



Feature Binding Time

- Binding time describes *when* a variable feature is to be *bound*, i.e. selected to become a mandatory part of a concept instance
- Mandatory features are considered to be bound
- Usual binding times include source time, compile time, link time, and run time (depends on the solution domain)

Concept Instance Definition

An instance I of the concept C at time t is a concept derived from C by selecting its features which includes the C's concept node and in which each feature f whose parent is included in I obeys the following conditions:

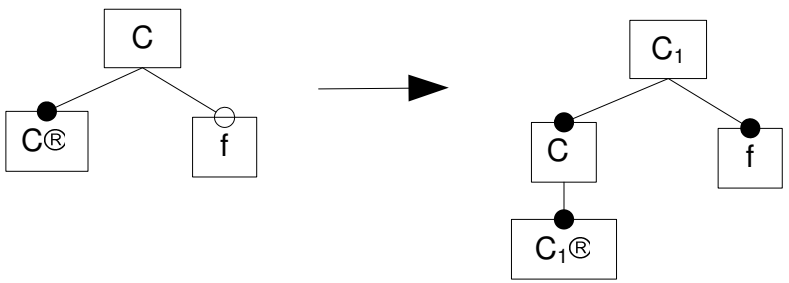
1. If f is a mandatory feature, f is included in I.
2. If f is a variable feature whose binding time is earlier than or equal to t, f is included in I or excluded from it according to the constraints of the feature diagram and additional constraints associated with it. If included, the feature becomes mandatory for I.
3. If f is a variable features whose binding time is later than t, f may be included in I as a variable feature or excluded from it, or the constraints (both feature diagram and additional ones) on f may be made more rigid as long as the set of concept instances available at later instantiation times is preserved or reduced.

Concept Instantiation in Time

- A concept instance is represented by a feature model derived from the feature model of the concept
- This model contains only the features included in the concept instance
- These features may be bound (becoming mandatory) or unbound (they stay variable)
- Mandatory features and features bound in previous instantiations are bound

Instantiation of Concepts with Circular References

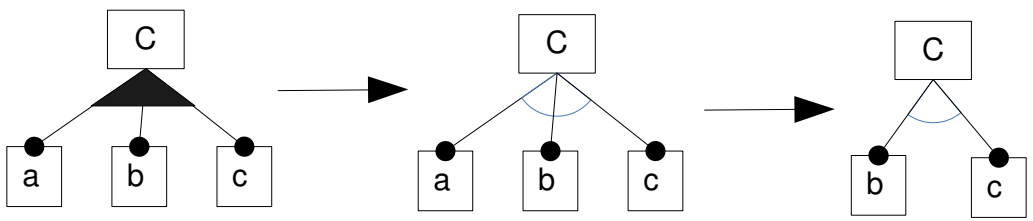
- Circular concept references are a consequence of circular dependencies in the domain
- They may be unfolded prior to instantiation
- The process of unfolding is ended by declaring that the remaining concept reference is a reference of an existing concept instance
- An example: a concept with a direct circular concept reference



- The circularity remains and may be revoked at instantiation times that follow
- Variable circular concept references may be simply removed

Reducing the Set of Concept Instances

- The constraints on unbound features may be made more rigid as long as the set of possible concept instances is preserved or reduced
- An example: a group of or-features may be transformed into a group of alternative features
- Another example: one of three alternative features may be excluded at source time, but both remaining two features must be kept



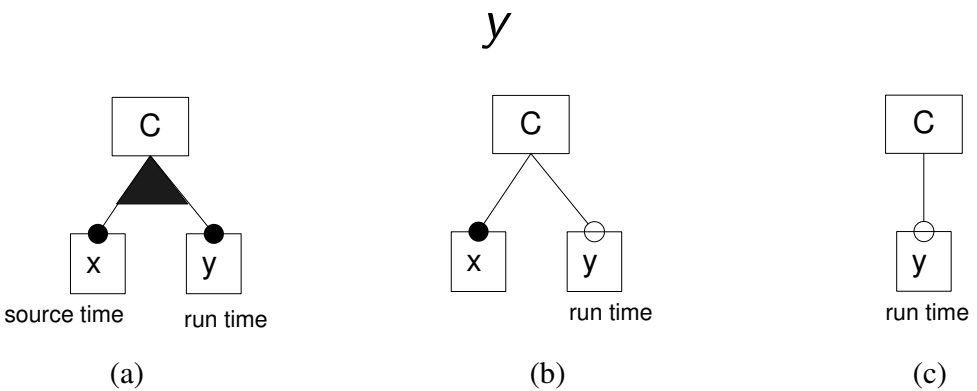
(a) (b) (c)

Concept Instance Validation

- A concept instance is valid if its features satisfy the constraints
- Validation is performed according to the concept instance definition
- A constraint may be evaluated only if all the features it refers to are bound
- Some logical expressions may be evaluated without the need to know the values of all their variables
- Additional constraints that refer to the features whose binding time is not later than the instantiation time can be safely removed from the model
- All other constraints have to be postponed for further instantiation

Evaluating Constraints with Unbound Features

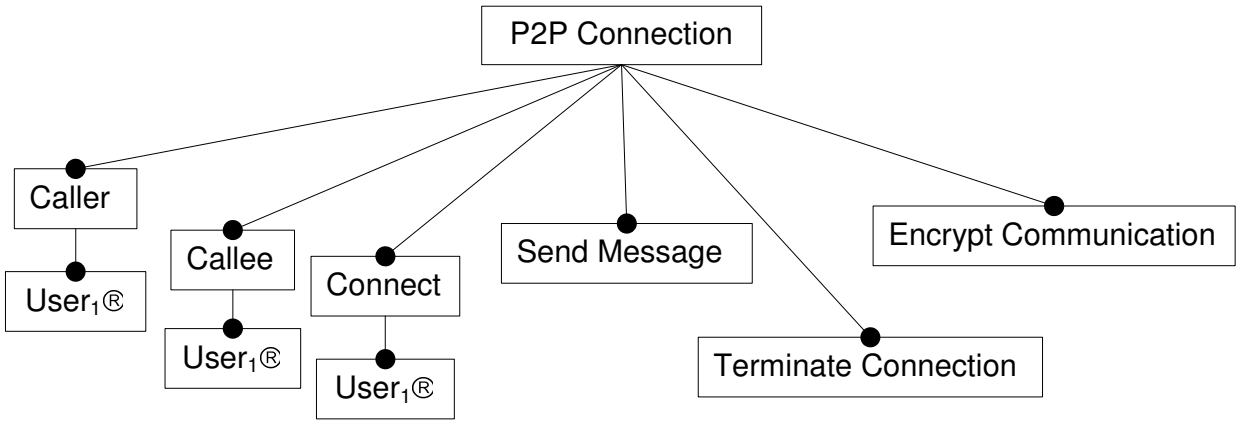
- Evaluating constraints on features whose binding time is later than the instantiation time
- An example:
  - If we bind x, the or-group constraint will be satisfied regardless of the binding of y
  - This constraint can be omitted by which y will become optional
  - Also, x may be omitted by which y must become optional
  - But it has to be assured it will finally be bound by adding a trivial constraint:



Dead-End Concept Instances

- Instances valid at a given instantiation time, but that can't be instantiated further
- For example, a source-time instance of the *P2P Connection* concept breaks the additional constraint at run time:

$$Encrypt\ Communication \Rightarrow Encryption\ Algorithm$$



Conclusions and Further Work (1)

- Concept instances are understood as concepts and represented by full-fledged feature models
- Concept instantiation taking into account feature binding time
- Support for circular references
- Concept instantiation with respect to binding time can be useful in
  - Providing a better control over the specific product configuration in product lines
  - Checking for dead-end instances
  - Creating specialized versions of frameworks (a source time instantiation)

Conclusions and Further Work (2)

- A prototype tool available at `http://fiit.stuba.sk/~vranic/fm/` (and another one coming soon)
- Further work:
  - Constraints that incorporate unbound features
  - Application to cardinality-based feature models