The Development of a Computer Application that Identifies Reusable Components through Formal Specifications

Francisco Moreira Couto
Technical University of Lisbon
Contents

• The Problem
• Proposed Solution
• Computer Application
  – Classification
  – Retrieval
  – Extensions
• Conclusions
The Problem

- Software Reuse
- Process of Reuse
  - Component Classification, Retrieval and Adaptation
- Types of Formal Reuse
  - Isomorphic
  - Compositional
Proposed Solution

• Formal Specifications
• Classification through Single Sort Algebraic Specification
• Retrieval through Category Theory
• Why Category Theory?
Computer Application

- Implementation of a Computer Application that Automates the Process of Reuse
Classification

• Translate an Algebraic Specification into a Categorical Representation
• Pre-category and Post-categories
• Objects are propositions and arrows are their logic implication
• Modifications made to the proposed methods:
  – Quantifiers representation
  – Semantics of equality
  – Represent implicit information
  – Normal forms
Classification (sample)

Component: Counter
Sort: \{0, 1, 2\}
Variables: State
Invariant: false \implies true

Method: Increment
Interface: !FinalState
Requires: State=0 \lor State=1
Ensures: State=0 \implies true \implies State=1,
true \implies !FinalState=1
State=1 \implies true \implies State=2,
true \implies !FinalState=2
Retrieval

• Isomorphic Matching
  – Between the user method and one library method
  – Identify all the functors by progressive construction
  – And/Or Graphs
  – Complexity
  – Heuristics
Retrieval (cont. 1)

• Compositional Matching
  – Between the user method and a sequence of library methods
  – Links in the sequence by progressive construction
  – Category matching by Isomorphic Matching
Extensions

- Representing quantifiers without any limitations
- Implications in the pre-conditions
- Sort members could be infinite
Conclusions

• The classification process and the isomorphic matching had already been implemented (in ML)

• Future work:
  – Implementation of the compositional matching
  – Complexity analysis
  – Implementation of the extensions
  – Develop new extensions (predicate arity)
  – Applying the application to a case of study