INTENSIONAL VIEWS AND RELATIONS

Andy Kellens
Kim Mens
Frédéric Pluquet
Roel Wuyts
Source-code View
Collection of source-code entities
**Extensional vs. Intensional**

**Extensional View**
- CheckingAccount
- SavingsAccount
- YouthAccount

**Intensional View**

Evaluating:
- "All subclasses of BankAccount"
- `if subclass(?entity,[BankAccount])`

**Accounts**
ALTERNATIVE VIEWS

"All subclasses of BankAccount"

if subclass(?entity,[BankAccount])

"All classes in package BankAccount"

if classInPackage(?entity,BankAccount)

Exclude: BankAccount

Extensional Consistency
Intensional Relation

Accounts

Bankcards

Persistence

Methods

Account changing

Methods

call

implement

implement

reference

Intensional Relation
All account changing methods must call a persistence method.

∀ x ∈ "Account changing methods"  
∃ y ∈ "Persistence Methods"  
  x calls y

Q₁, Q₂ ∈ \{∀, ∃, ∀, ∃!\, \ldots\}  
V₁, V₂ ∈ Views  
r = predicate over source-code entities
Case Study

• SmallWiki
• 15 views
• 15 nesting views
• 17 relations
• Document initial version
• Apply to small evolution (1 month)
• Apply to large evolution (+/- 1 year)

GOAL:
Support co-evolution of documentation and implementation
Methodology

- Manual code inspection
- Initial views/relations
- Check conformance:
  - Detected small “bugs”
  - Refine documentation
- Iterative process
Applying to later versions

• Check views/relations to new version of code
• Inspect conflicts
  • Conflicts because of restructuring the code
• Update documentation when needed
Lessons learned

• Visualization
• Understanding (Evolution)
• Query languages
• Other
Visualization
Understanding (evolution)

- Start: little knowledge about SmallWiki
- Incremental documentation:
  - building views/relations in small steps
  - increased insights into the internals of the system
- Checking conformance: insights into evolution of the system

Incremental approach leads to better understanding of the system
namesClasses := SmallWiki.VisitorOutput allSubclasses inject: Set new into:
  [:tas :nextclass |
  tas
    addAll: ((Soul.MLI current methodsInClass: nextclass)
      select: [:method | 'accept*:' match: method selector]
      collect: [:method | method selector copyFrom: 7 to: method selector size - 1]);
  yourself].
namesClasses inject: OrderedCollection new into:
  [:tas :next |
  value |
  (value := (Soul.MLI current classesFor: next asSymbol)
    detect: [:class | SmallWiki allClasses includes: class]
    ifNone: [nil] ) isNil ifFalse: [tas add: value].
  tas]
Other

- Completeness
- Deviations
- Dynamic vs. Static
- Do we need logic?
Future work

• Mining:
  • Views (FCA)
  • Relations (Brute-force technique)
• Document dynamic structure
• Support for assessing impact of evolution
• Aspect Documentation
QUESTIONS