Co-evolving Code and Design with Intensional Views

A Case Study

Kim Mens (UCL)
Andy Kellens (VUB)
Frédéric Pluquet & Roel Wuyts (ULB)


#&@!§ ...
I STILL HAVE TO
PREPARE MY
PRESENTATION FOR
ESUG 2005 !!!

Problem : Context

- *Intensional Views (and Relations)*
- have been proposed as a way to support
  - program understanding
  - software maintenance
  - co-evolution of source code and design
- by “actively” documenting high-level structural regularities in source code

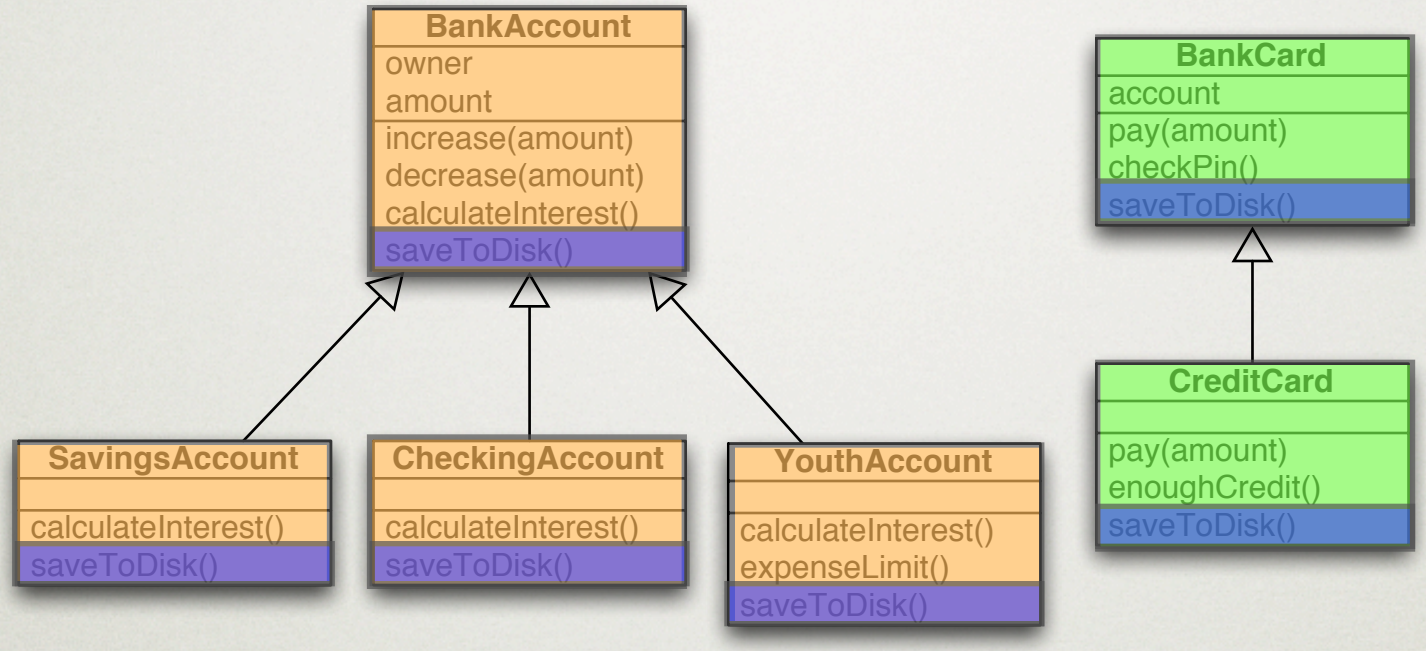
Problem : Goal

- Verify claims by conducting case study :
  - Document initial version
  - Apply to small evolution (1 month)
  - Apply to large evolution (+/- 1 year)
- Draw conclusions about *IntensiVE*
  - w.r.t. software understanding, maintenance and (co-)evolution

Intensional Views (and Relations)

- Motivating example
- Extensional vs. intensional views
- Alternative intensions
- Relations between views

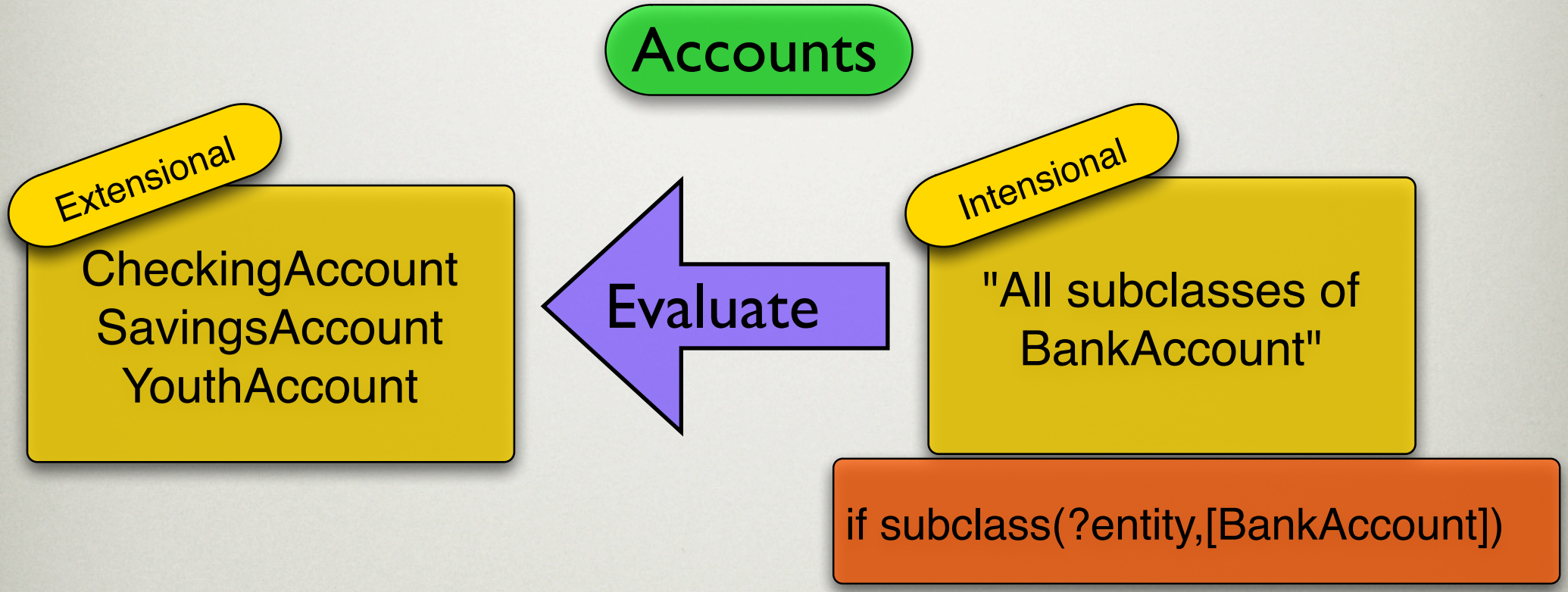
Motivating Example



Source-code View
Collection of source-code entities

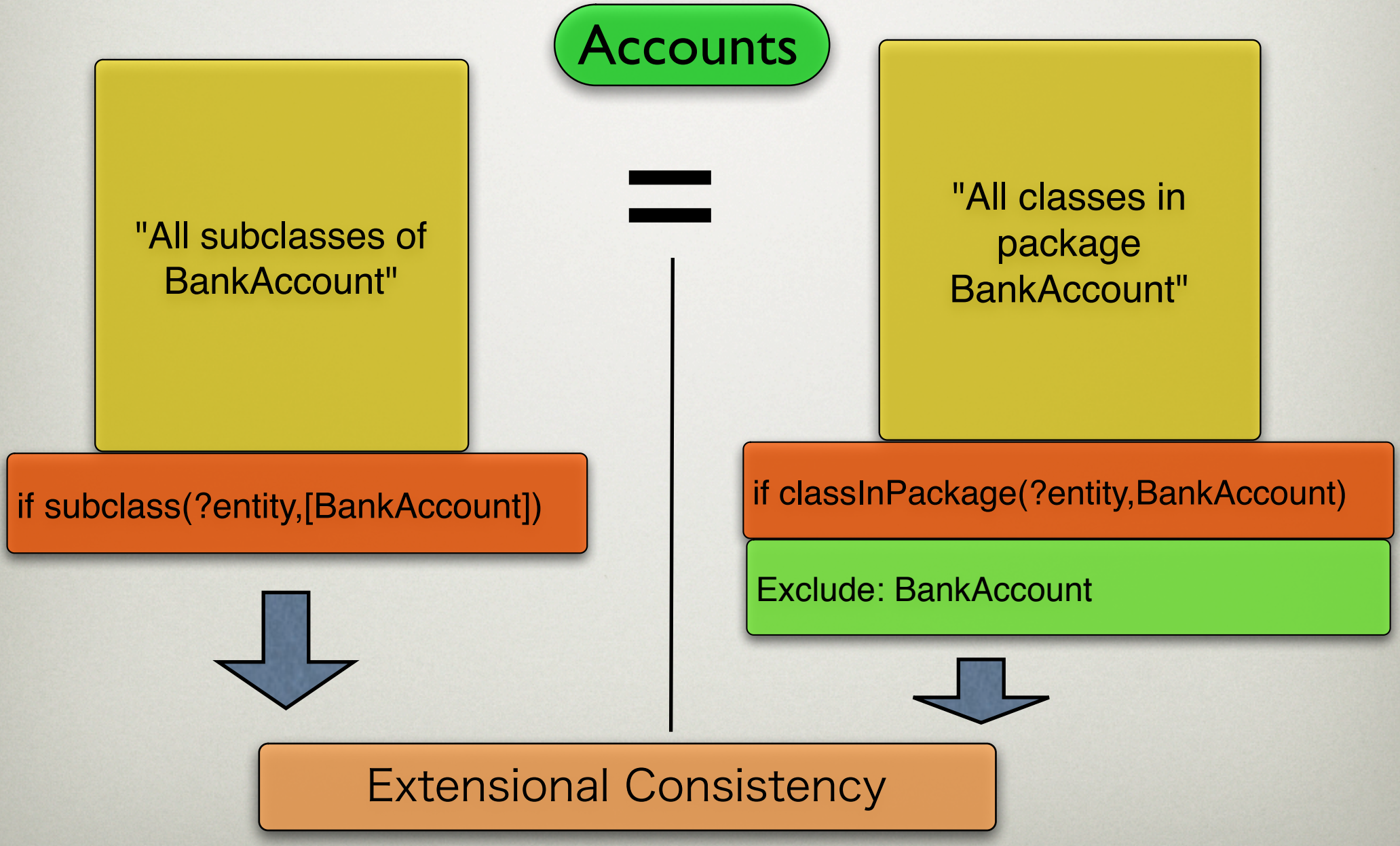
Extensional vs. Intensional Views

Problem
Intensional views
IntensiVE
The case
Case study
Lessons learned
Future work



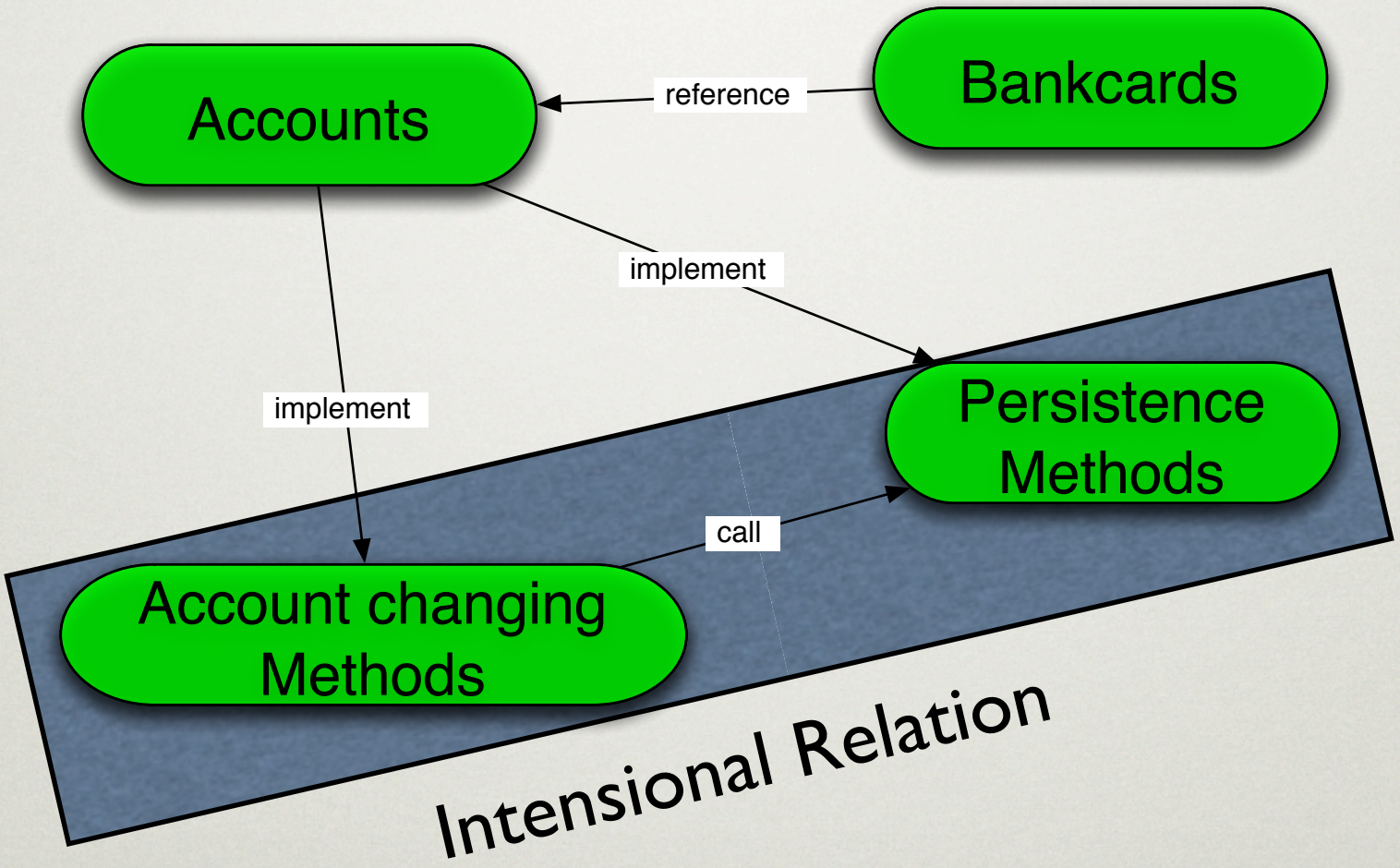
Alternative Intensions

Problem
Intensional views
IntensiVE
The case
Case study
Lessons learned
Future work



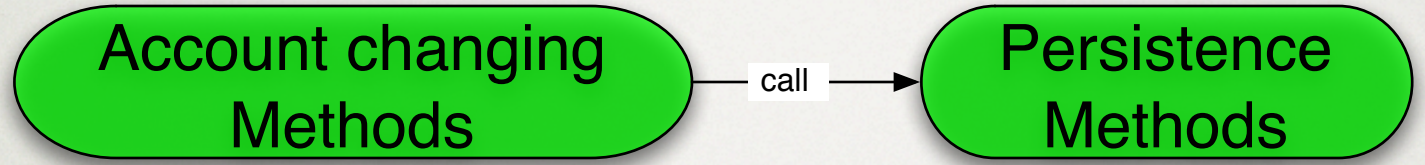
Intensional Relations

Problem
Intensional views
IntensiVE
The case
Case study
Lessons learned
Future work



Intensional Relations

Problem
Intensional views
IntensiVE
The case
Case study
Lessons learned
Future work



All account changing methods must call a persistence method

$\forall$ x $\in$ "Account changing methods"
$\exists$ y $\in$ "Persistence Methods"
x calls y

$Q_1 x \in V_1; Q_2 y \in V2: x\ r\ y$

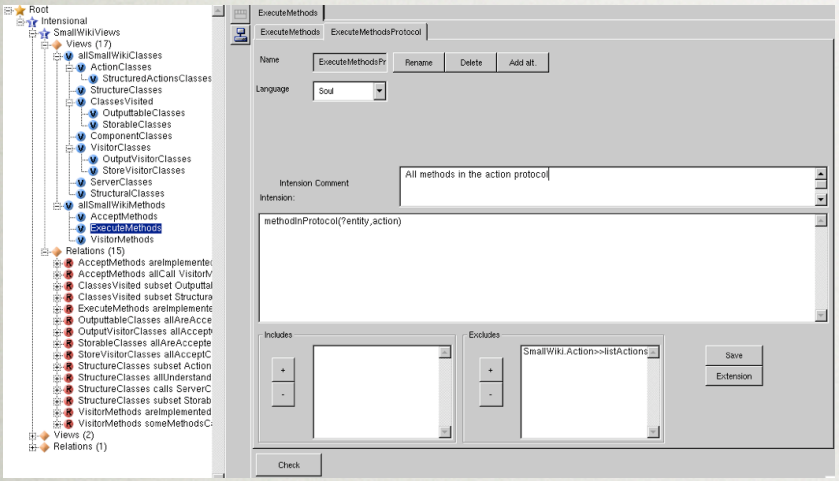
$Q_1, Q_2 \in \{\forall, \exists, \nexists, \exists !, \ldots\}$

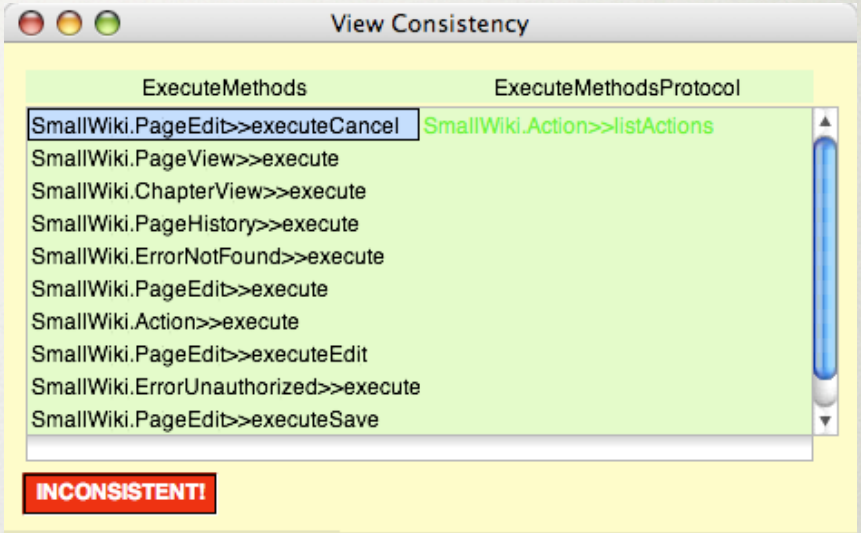
$V_1, V_2 \in$ Views

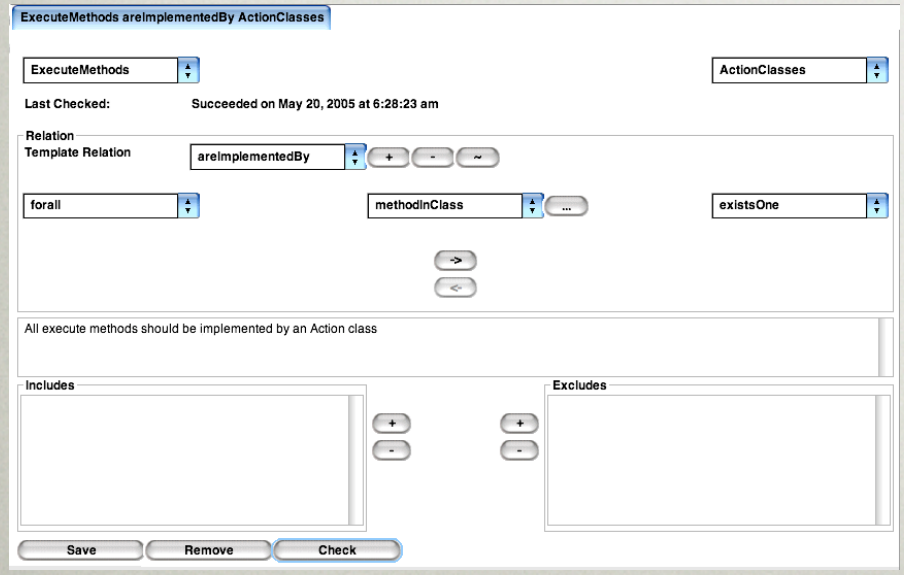
r = predicate over source-code entities

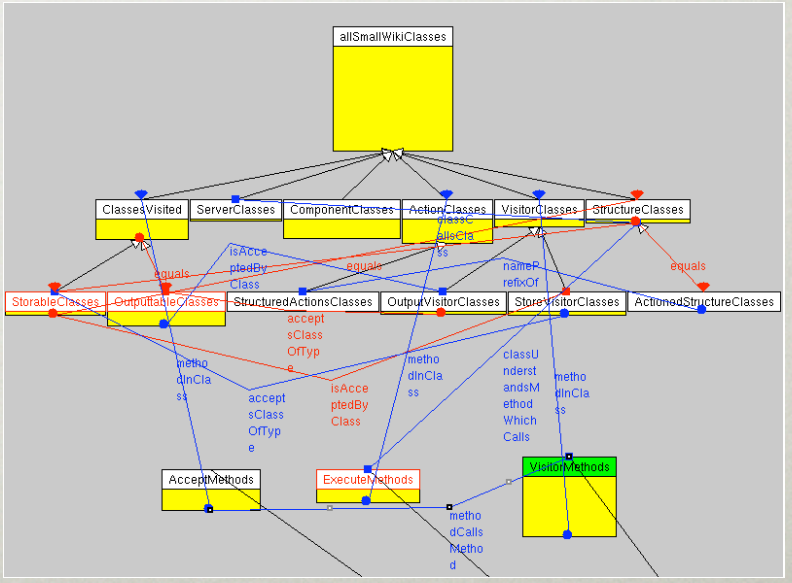
The Intensional View Environment

Problem
Intensional views
IntensiVE
The case
Case study
Lessons learned
Future work





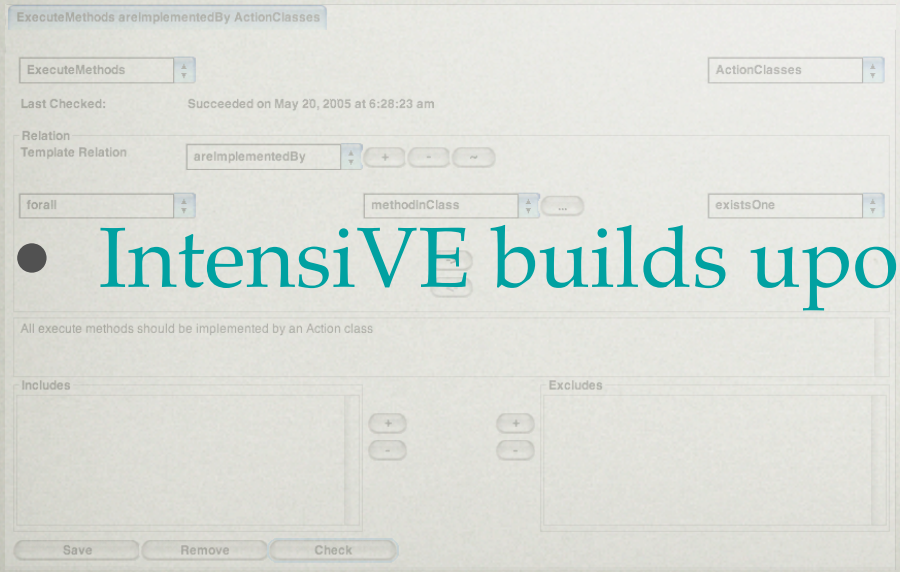




The Intensional View Environment

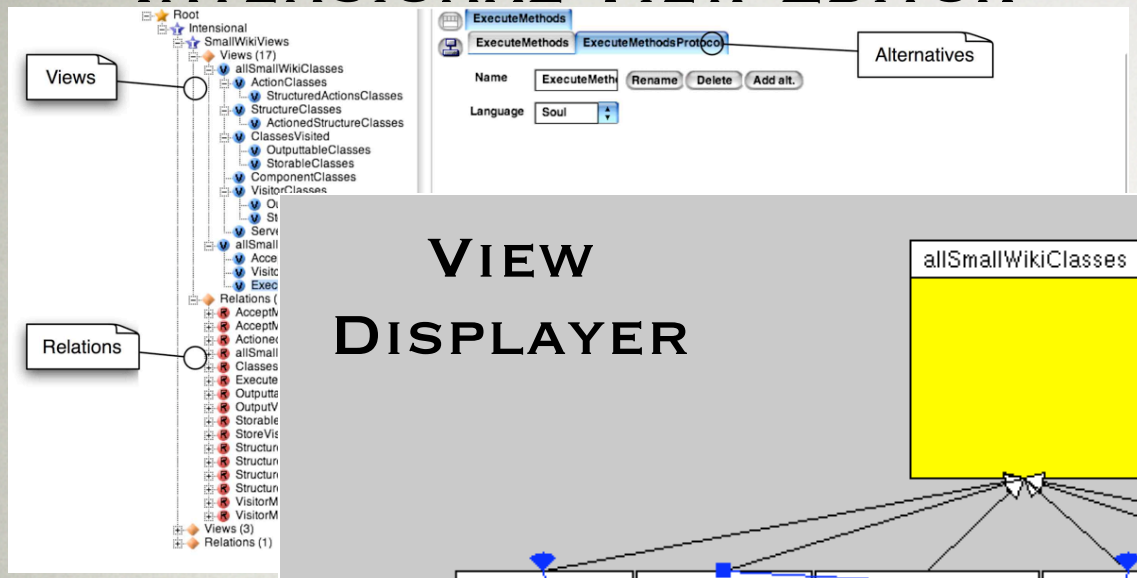
Problem
Intensional views
IntensiVE
The case
Case study
Lessons learned
Future work

- Seamlessly integrated with Visualworks 7.3
- IntensiVE comprises:
  - *Intensional View Editor*
  - *View Consistency Checker*
  - *Relation Editor*
  - *Relation Checker*
  - *View Displayer*

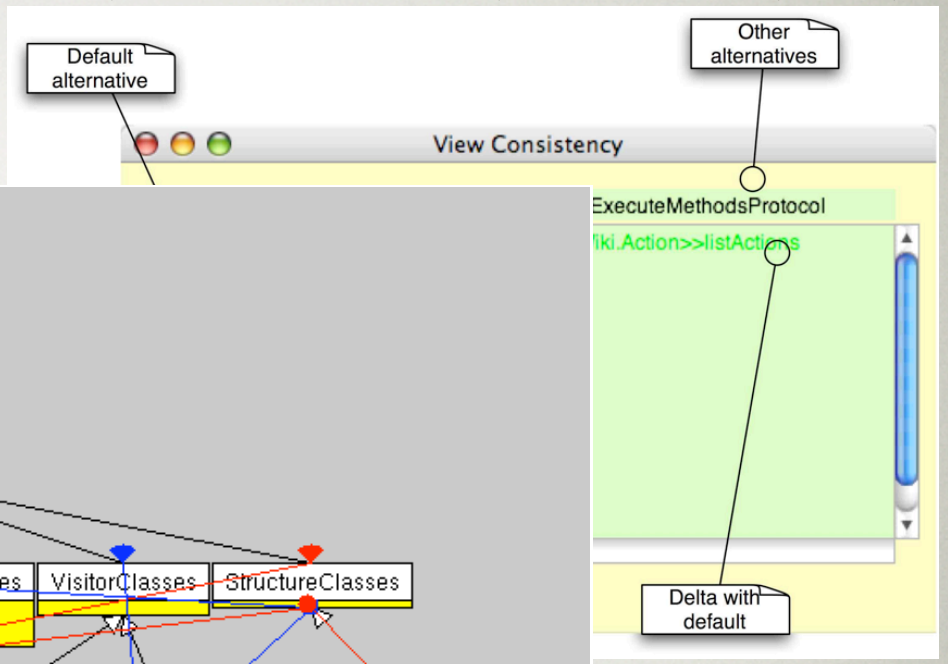


- IntensiVE builds upon:
  - *StarBrowser*
  - *SOUL + NewLicor*
  - *MOOSE + CodeCrawler*

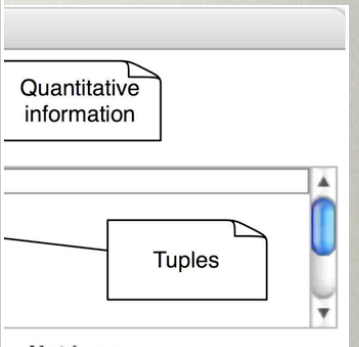
Intensional View Editor

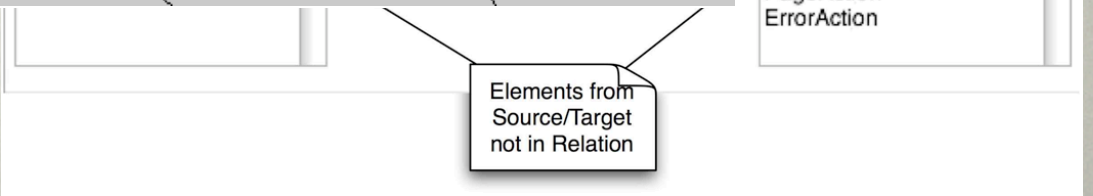


View Consistency Checker



View Displayer





Relation Editor

Relation Checker

The Case : SmallWiki

GOAL::
Support:co-evolution:of:
documentation:and:
implementation

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

Methodology

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

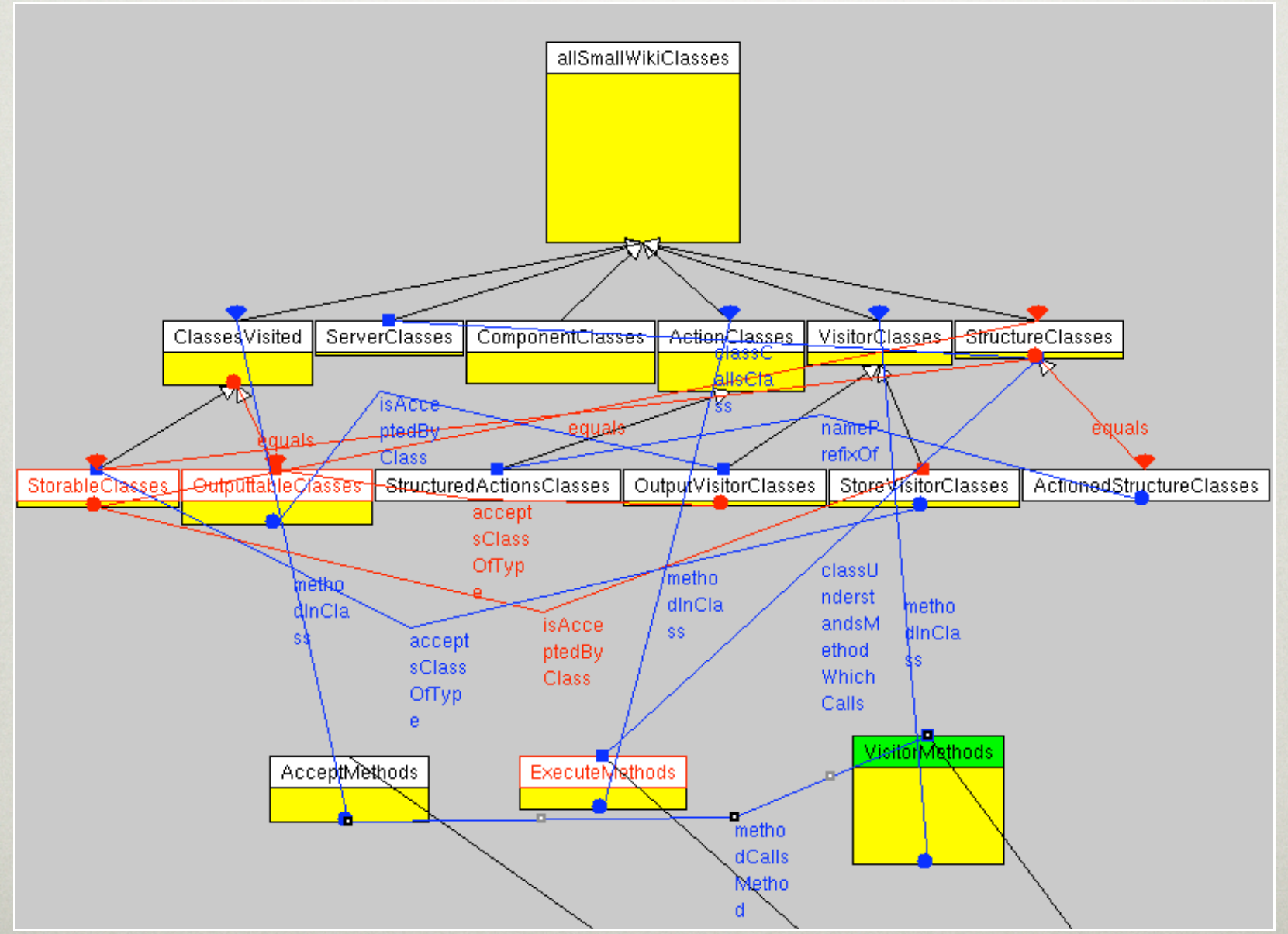
Applying to later versions

- 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

Query language

```
| namesClasses |
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]
```

```
classInHierarchyOf(?c, [SmallWiki.VisitorOutput]),
methodWithNameInClass(?,?s, ?c),
[?s = (#accept, ?entity name, ':') asSymbol]
```

Other

- Coverage
- Deviations
  - some evolution conflicts due to obsolete deviations
- Dynamic vs. Static
- Do we need logic?
- Scaleability

Future work

- Mining:
  - Views (FCA + ILP?)
  - Relations (Brute-force technique)
- Document dynamic structure
- Support for documenting versions
- Aspect Documentation

Wanna learn more ?

- Contact :
  - Kim Mens (km@info.ucl.ac.be)
  - Andy Kellens (akellens@vub.ac.be)
  - Frédéric Pluquet
  - Roel Wuyts
- Publications :
  - TOOLS1999, SEKE2002, ICSM2003, ESUG2005, ICSM2005 (2)

Wanna learn more?

✍ *Innovation Awards* (tonight)
Andy Kellens
(☑ vote for us ☻)

☞ *Technical track* (tomorrow)
Frédéric Pluquet
Tools for Intensional Views

☟ Or check our **Website**

www.intensional.be