Patching Patches

Reinout Stevens

resteven@vub.ac.be

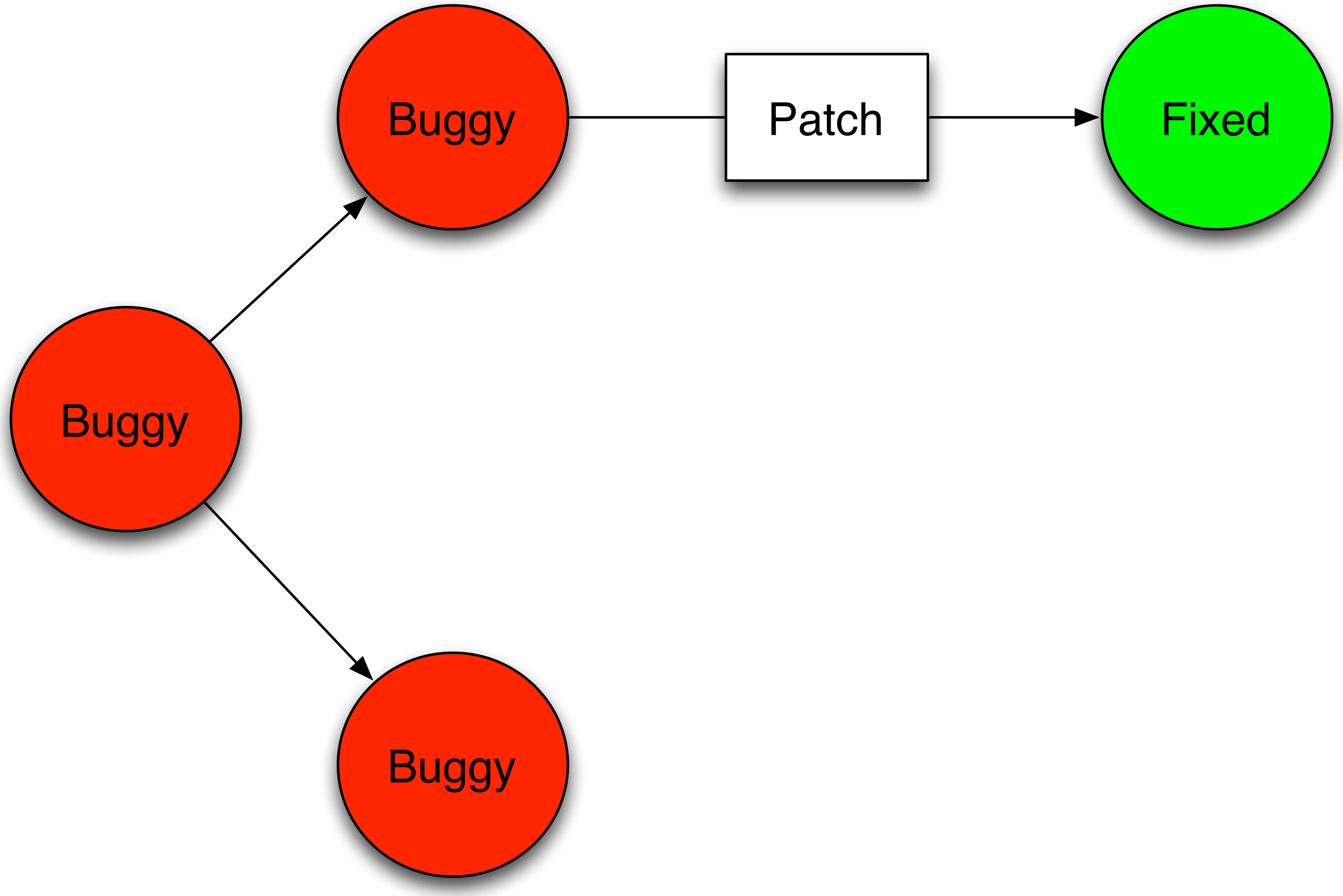
@ReinoutStevens

http://soft.vub.ac.be/~resteven/slides/13benevol.pdf

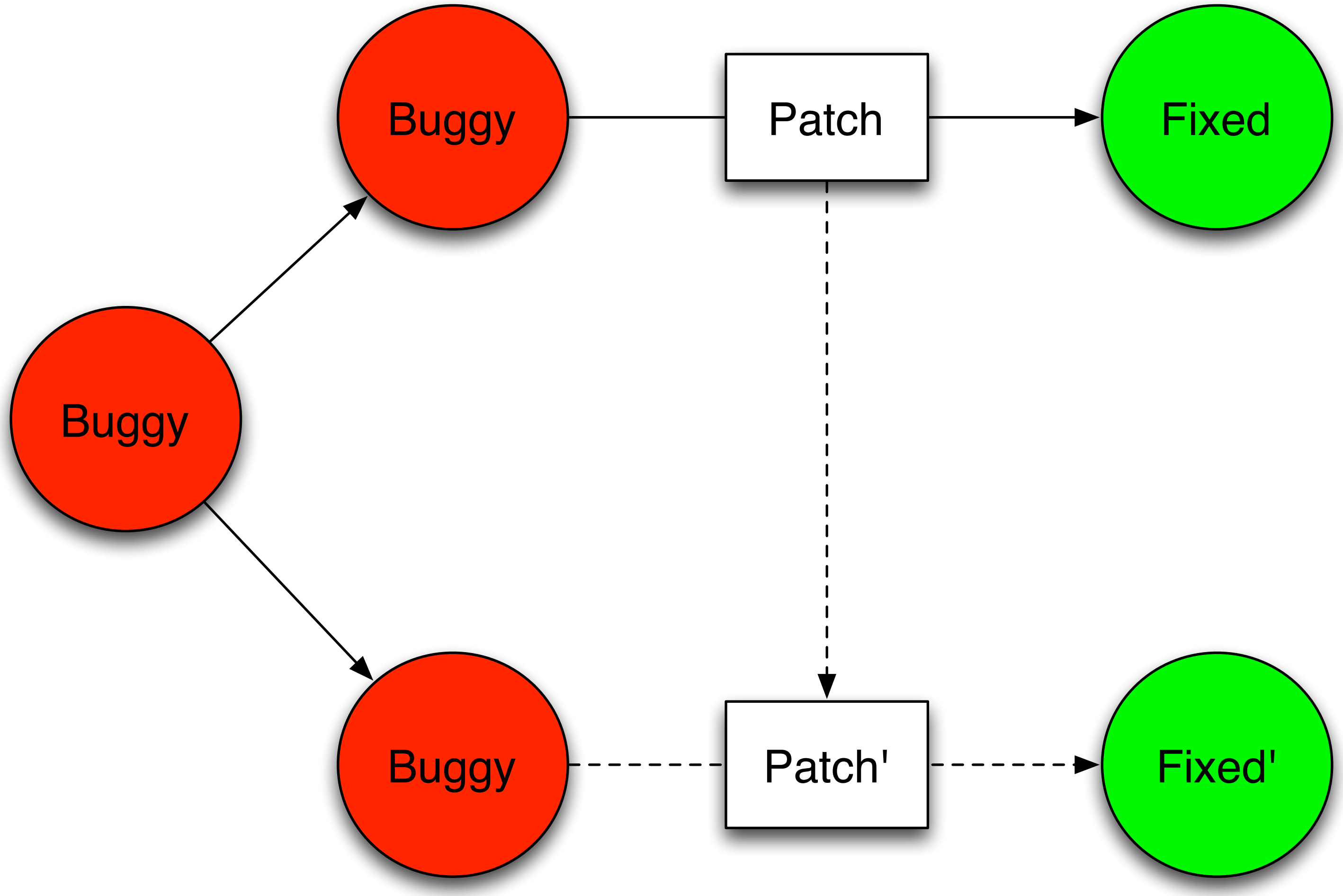
Software Languages.Lab

Vrije Universiteit Brussel

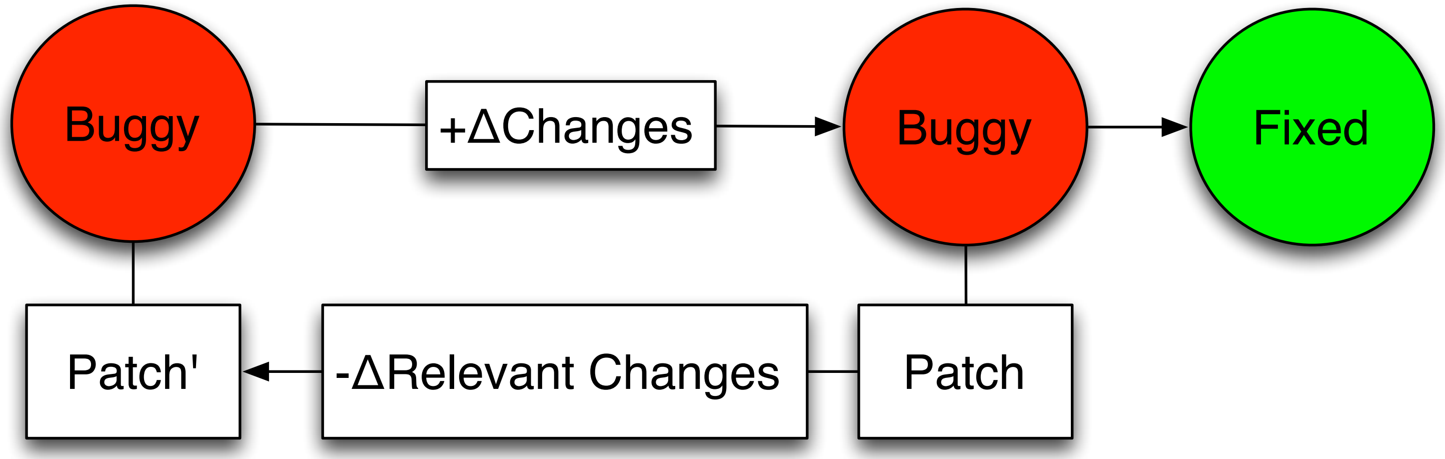
Context

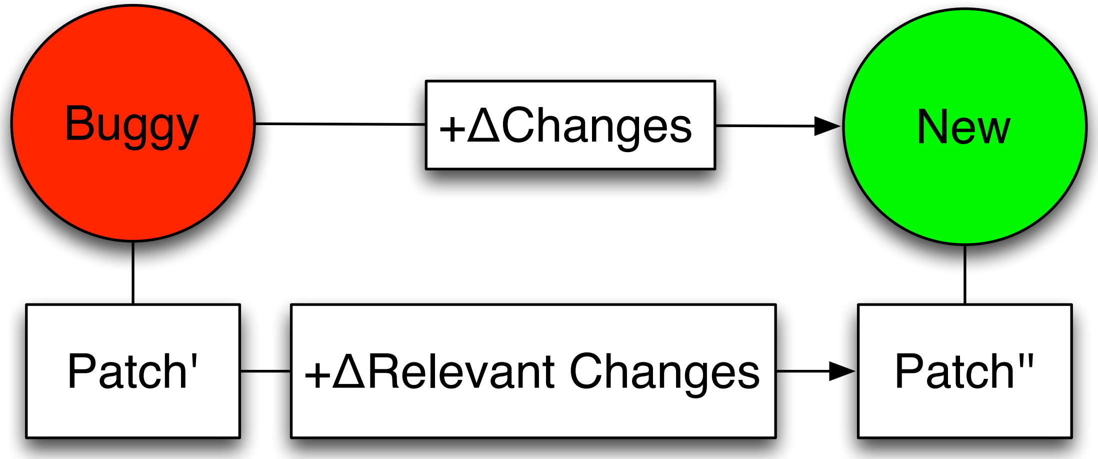


Context



Idea

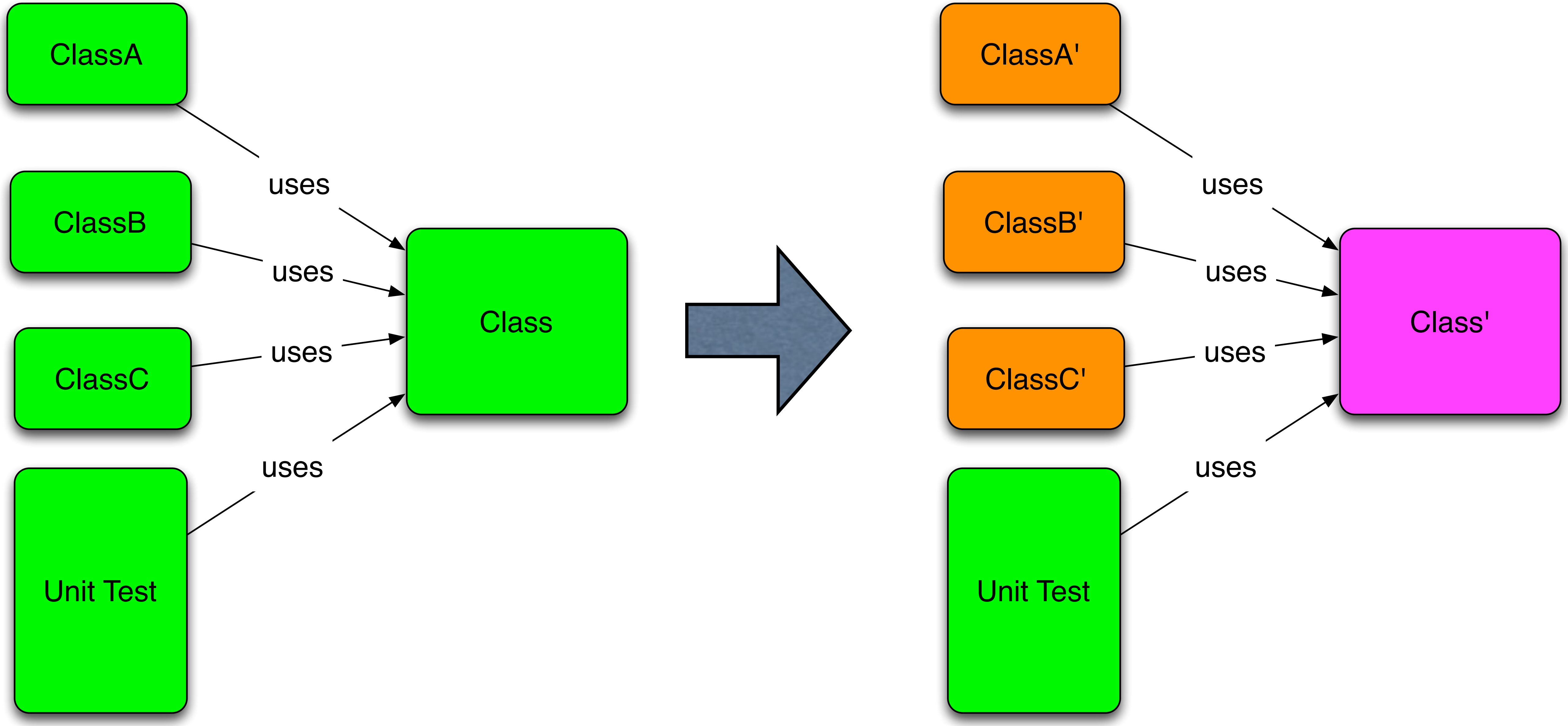

Buggy
+ΔChanges
Buggy
Fixed
Patch'
-ΔRelevant Changes
Patch


Buggy
+ΔChanges
New
Patch'
+ΔRelevant Changes
Patch''

Patching Unit Tests

Reinout Stevens
resteven@vub.ac.be
@ReinoutStevens
http://soft.vub.ac.be/~resteven/slides/13benevol.pdf

Context


ClassA
ClassB
ClassC
Unit Test
uses
uses
uses
uses
Class
ClassA'
ClassB'
ClassC'
Unit Test
uses
uses
uses
uses
Class'

Overview

- Ekeko
- ChangeDistiller
- Constraint Programming
- Mixing it all together

Ekeko

collection of all substitutions for ?s and ?e

```
(run* [?s ?e]
  (ast :ReturnStatement ?s)
  (has :expression ?s ?e)
  (ast :NullLiteral ?e))
```

such that the following Ekeko relations hold:

ast/2 holds for :ReturnStatement, ?s

has/3 holds for :expression, ?s, and ?e

ast/2 holds for :NullLiteral, ?e

?e is the value of the property named :expression of ASTNode ?s

```
([#<ReturnStatement return null;
  #<NullLiteral null>]
 ...
 [#<ReturnStatement return null;
  #<NullLiteral null>])
```

https://github.com/cderoove/damp.ekeko

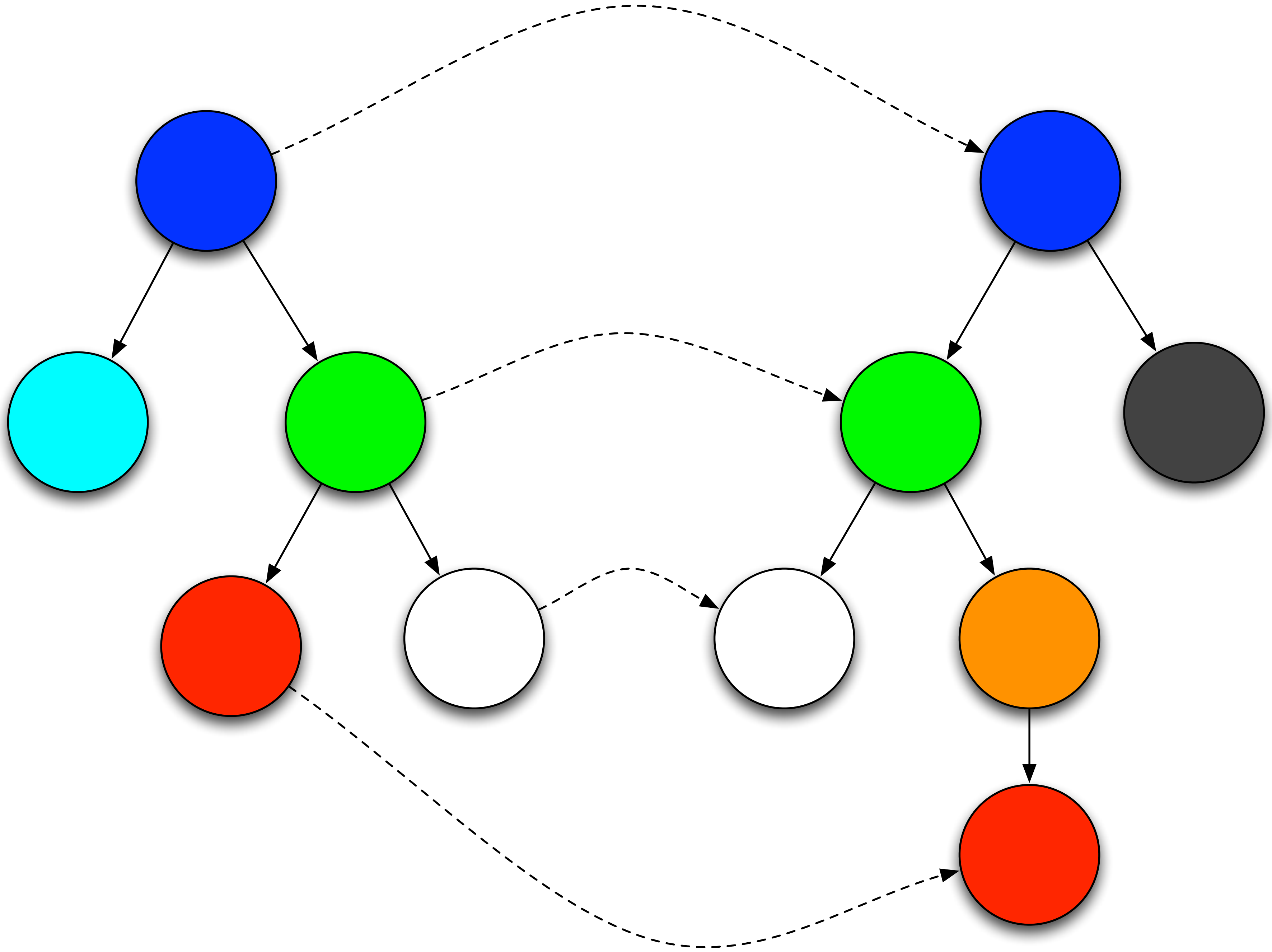
ChangeDistiller

- Takes 2 ASTs and a matching as input
- Outputs minimal list of edit operations

Beat Fluri and Harald C. Gall.
Classifying Change Types for Qualifying Change Couplings. , ICPC 2006

Beat Fluri, Michael Würsch, Martin Pinzger, and Harald C. Gall.
Change Distilling: Tree Differencing for Fine-Grained Source Code Change Extraction. TSE 2007

AST Matching



ChangeDistiller

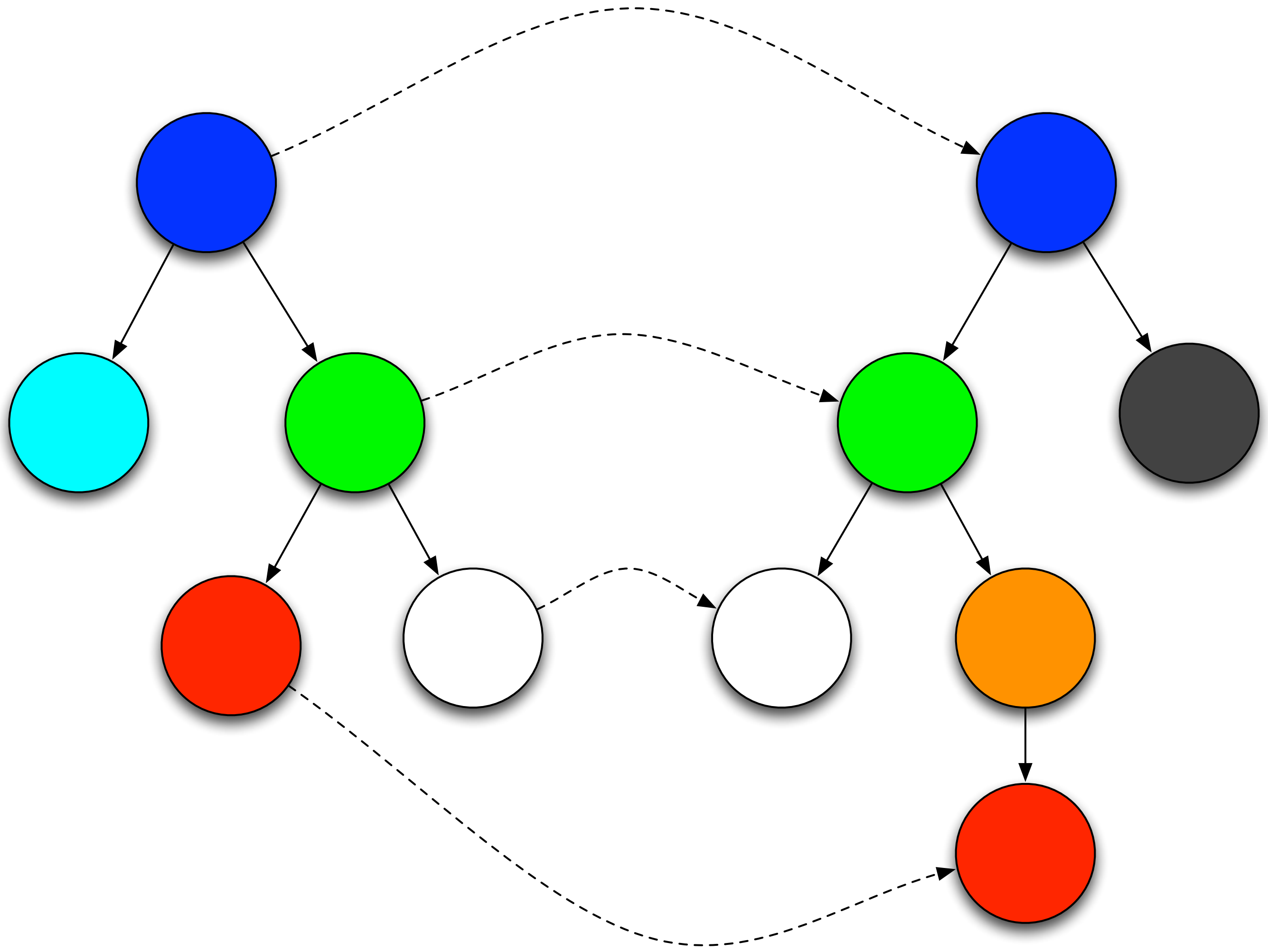
#<Insert 🟠 >

#<Delete 🔵 >

#<Insert ⚫ >

#<Move ⚪ >

#<... >



Constraint Programming

- Logic variables are assigned a finite domain
- Define operations on these domains
  - Intersection, member, ...

https://github.com/calvis/cKanren/tree/master/cKanren

Constraint Programming

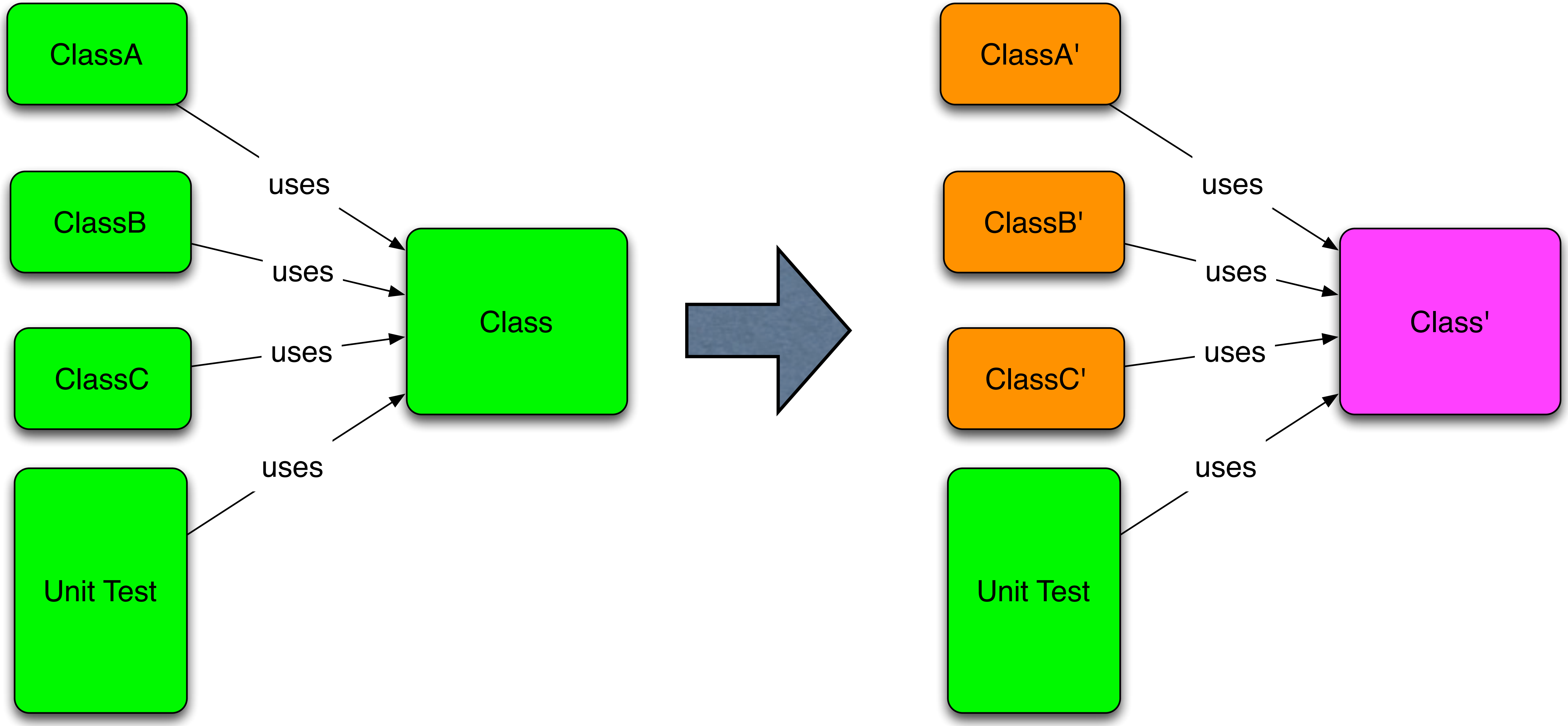
```
(run* [?x ?y]
  (dom ?x (interval 0 100))
  (dom ?y (interval 50 200))
  (>= ?x ?y))
```

```
([50 50] [51 50] [52 50] [51 51] [53 50] [52 51] [52
52] [54 50] [53 51] [53 52] ...)
```

Constraint Programming’

```
(run* [?method ?return ?value]
  (astc :MethodDeclaration ?method)
  (childc ?method ?return)
  (astc :ReturnStatement ?return)
  (hasc :expression ?return ?value)
  (astc :NullLiteral ?value)
  (ground-ast an-ast ?method ?return ?value))
```

Reminder


ClassA
ClassB
ClassC
Unit Test
uses
uses
uses
uses
Class
ClassA'
ClassB'
ClassC'
Unit Test
uses
uses
uses
uses
Class'

Approach

- Distill AST Operations
- Extract Template Patch
- Apply Patch on Unit Test

Template Patch

- AST Nodes + Operations
- Constraints on those Nodes
  - Provide context

Template Patch’

```java
package testing;

public class TestClass {
 public int foo(){
    int res = 0;
    for(int i = 0; i < 10; ++i){
       res += i;
    }
    return res;
 }
}
```

```java
package testing;

public class TestClass {
 public int bar(){
    int result = 0;
    for(int i = 0; i < 10; ++i){
       result += i;
    }
    return result;
 }
}
```

Template Patch’

```
package testing;

public class TestClass {
 public int foo(){
    int res = 0;
    for(int i = 0; i < 10; ++i){
       res += i;
    }
    return res;
 }
}
```

```
package testing;

public class TestClass {
 public int bar(){
    int result = 0;
    for(int i = 0; i < 10; ++i){
       result += i;
    }
    return result;
 }
}
```

```
#<Update MethodDecl foo => bar>
#<Update FieldDecl res => result>
#<Update Assignment res => result>
```

Constraints:

```
(childc #<MethodDecl foo> #<ClassDecl TestClass>
(childc #<ForStatement ...> #<MethodDecl foo>)
```

...

Generalization

- Matching algorithm from ChangeDistiller
- Ensure patch does not become applicable on other parts of the program
- Remove too specific constraints
- Replace concrete variables with logical ones

https://github.com/ReinoutStevens/

@ReinoutStevens

resteven@vub.ac.be

http://soft.vub.ac.be/~resteven/slides/13benevol.pdf