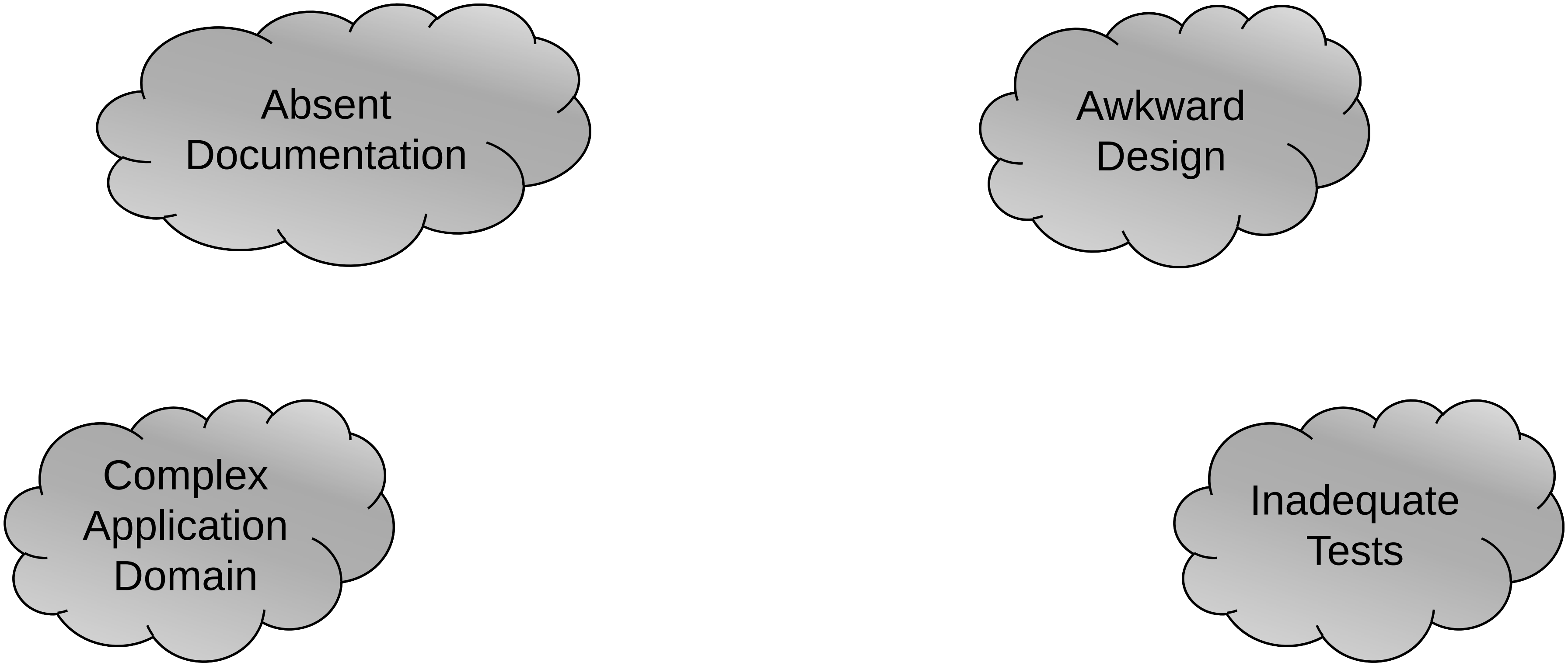
Tracking Code Clones in Evolving Software

Ekwa Duala-Ekoko
Martin Robillard

McGill

School of Computer Science
Montréal, Canada

Obstacles to Software Maintenance



Absent Documentation

Awkward Design

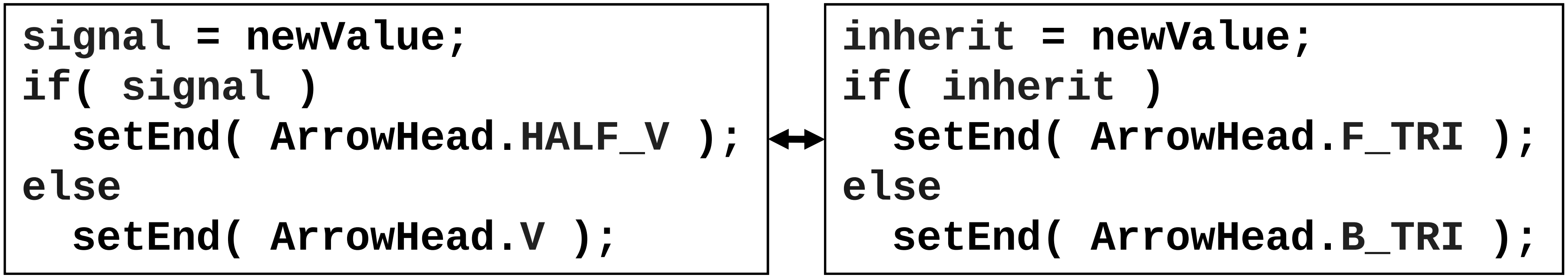
Complex Application Domain

Inadequate Tests

CODE CLONES

Code Clones

“A code portion in source files that is identical or similar to another.” [Kamiya et al., 2002]



```
signal = newValue;
if( signal )
  setEnd( ArrowHead.HALF_V );
else
  setEnd( ArrowHead.V );
```

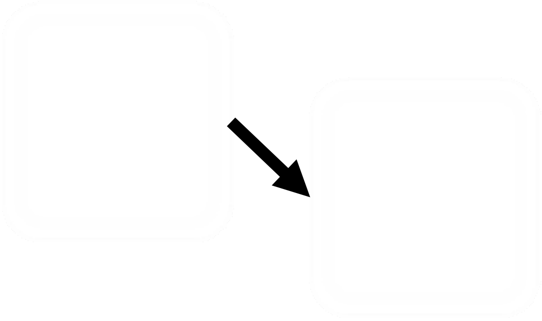
```
inherit = newValue;
if( inherit )
  setEnd( ArrowHead.F_TRI );
else
  setEnd( ArrowHead.B_TRI );
```

Change coupling

- Clone regions must be changed together
- Oversights in consistent changes lead to regression faults
- A resolved bug seems to reappear when cloned siblings are executed
- [Aversano et al., 2007], [Geiger et. al., 2006], [Jiang et al., 2007]

How are Clones Introduced?

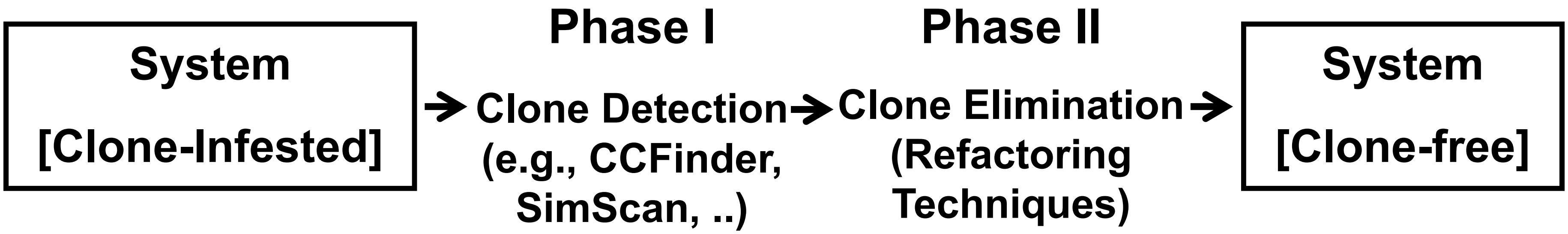
- Copy and Paste
- Programming by Example
- Use of Code Generators
- Difficult Modularization
- Architecture and Deployment Issues



Hard to completely avoid…

- 20% in XWindows [Baker, 1995]
- 13% in 400kLOC control system [Baxter et al., 1998]

Existing Solutions



System

[Clone-Infested]

➔

Phase I

Clone Detection

(e.g., CCFinder, SimScan, ..)

➔

Phase II

Clone Elimination

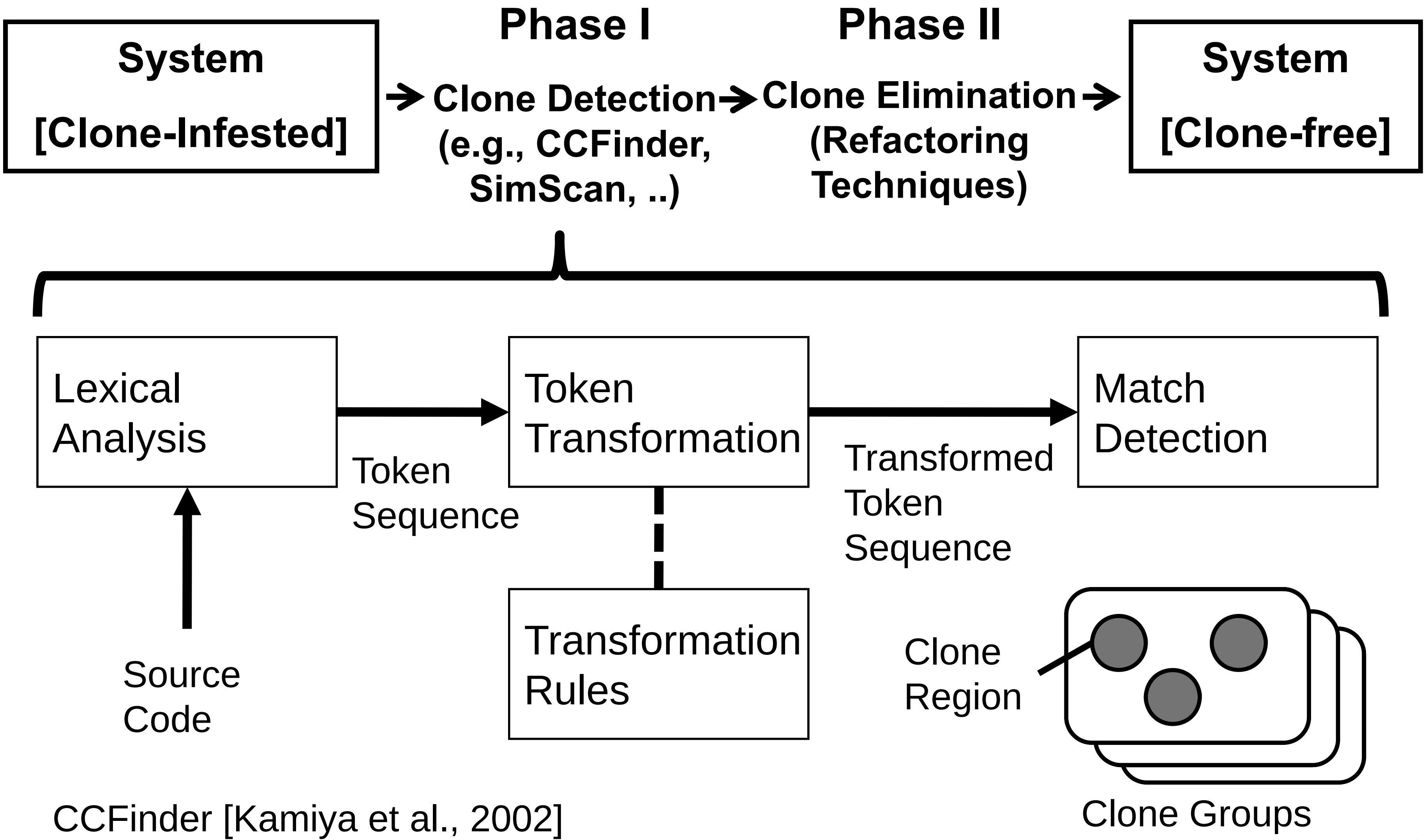
(Refactoring Techniques)

➔

System

[Clone-free]

Existing Solutions



System [Clone-Infested] → Phase I: Clone Detection (e.g., CCFinder, SimScan, ..) → Phase II: Clone Elimination (Refactoring Techniques) → System [Clone-free]

Source Code → Lexical Analysis → Token Sequence → Token Transformation → Transformed Token Sequence → Match Detection

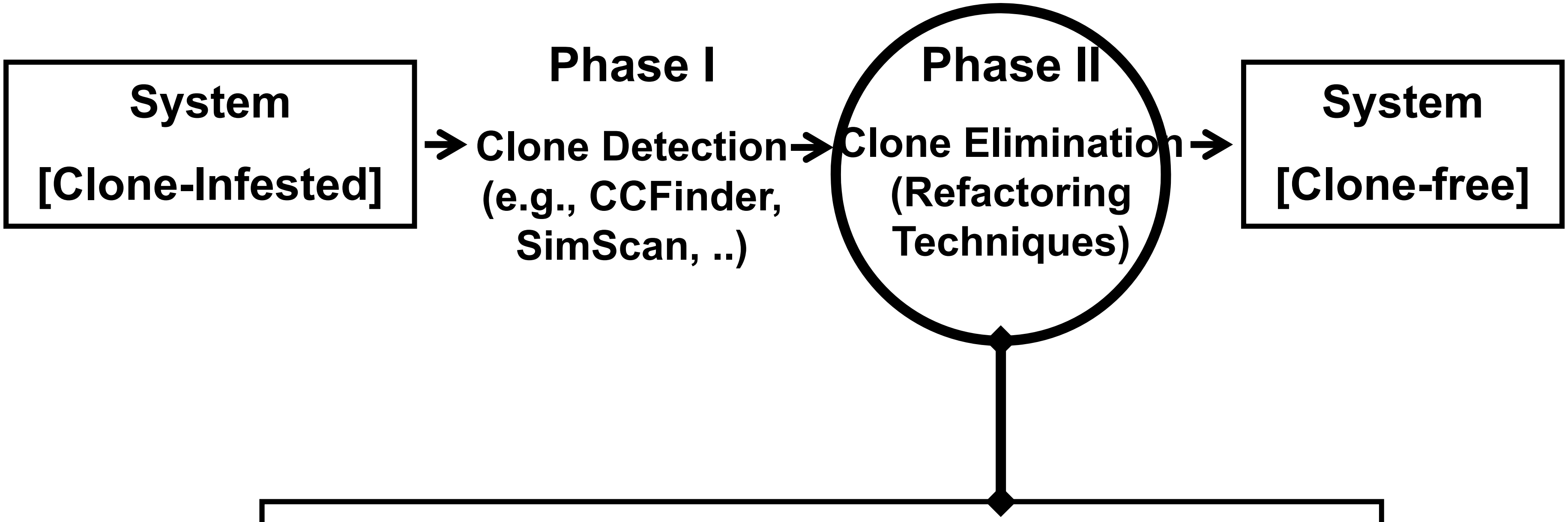
Transformation Rules

Clone Region

Clone Groups

CCFinder [Kamiya et al., 2002]

Existing Solutions



System

[Clone-Infested]

Phase I

Clone Detection (e.g., CCFinder, SimScan, ..)

Phase II

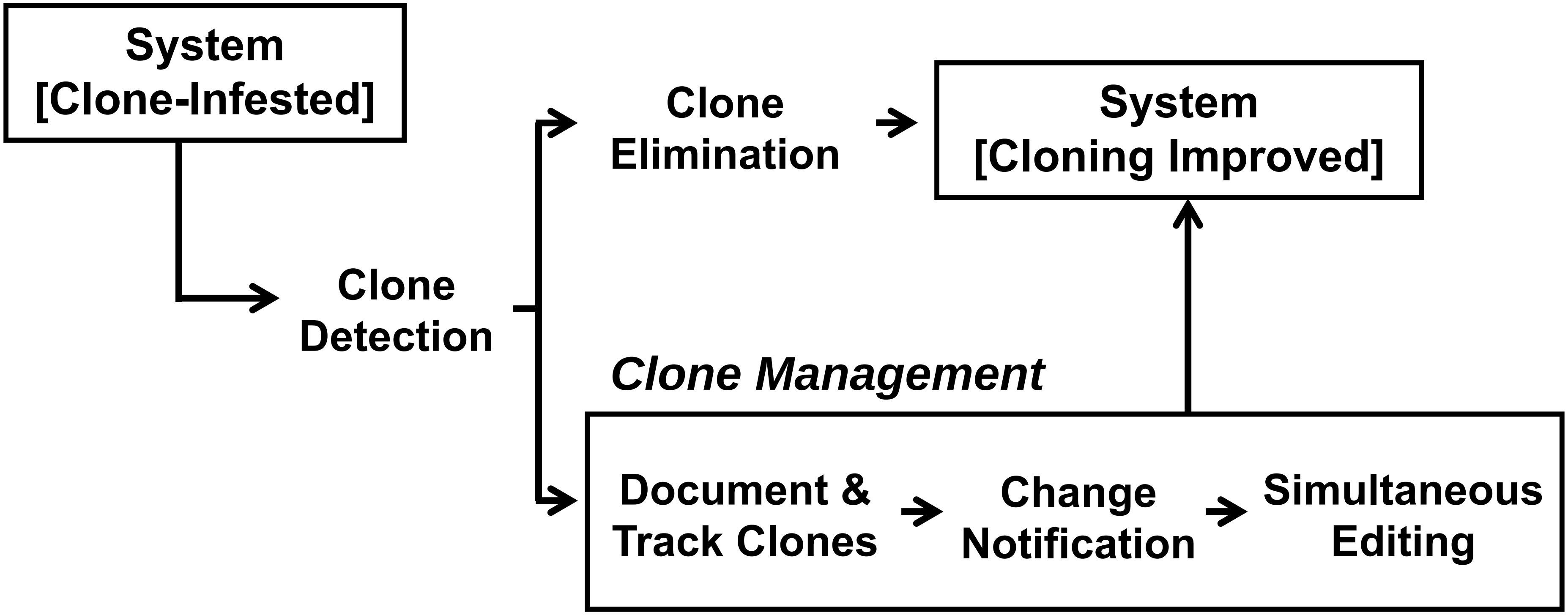
Clone Elimination (Refactoring Techniques)

System

[Clone-free]

- Some code clones are difficult/impossible to refactor
  [49%-64%, 2 open source projects, Kim et. al]
- Refactoring is not always cost-effective/beneficial
  [48%-72% disappeared within 8 check-ins, Kim et. al]
- Clones can be useful
  [e.g., for system stability, Kasper and Godfrey, 2006]

Proposed Approach



System
[Clone-Infested]

Clone
Detection

Clone
Elimination

System
[Cloning Improved]

Clone Management

Document &
Track Clones

Change
Notification

Simultaneous
Editing

✗ **Complete Clone Detection**

(54kLOC ≈ 4 hours)

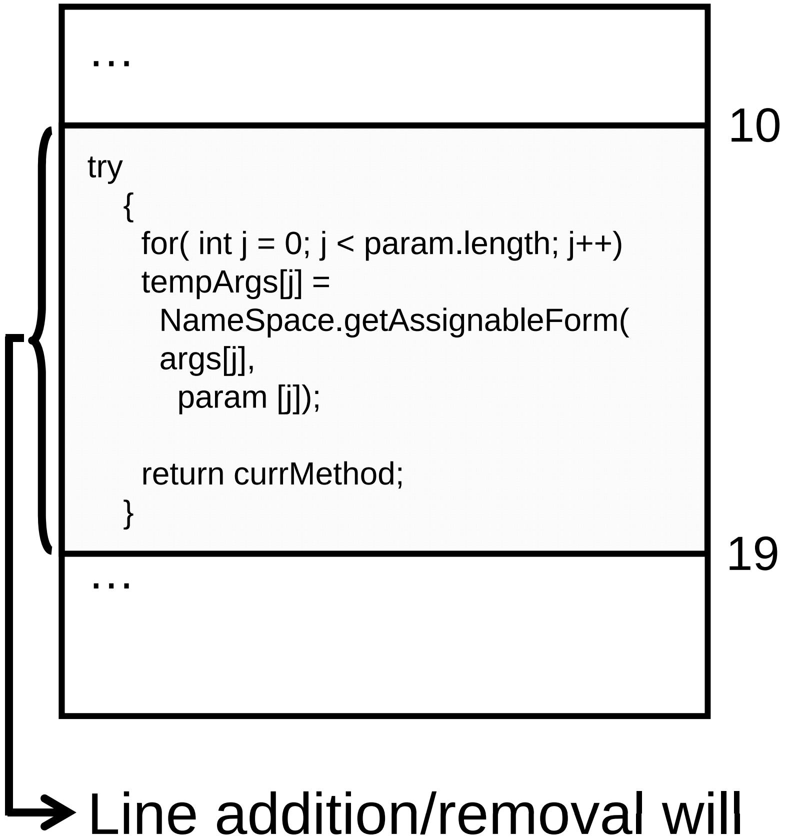
Outline

1. Clone documentation model and tool
2. Advanced tool-based features
   - Simultaneous editing
   - Incremental clone detection
3. Evaluation of the technique

Describing Clone Regions (CRs)

File/lines based descriptions

bsh.Reflect.java [10,19]



```
…
try
   {
     for( int j = 0; j < param.length; j++)
     tempArgs[j] =
       NameSpace.getAssignableForm(
       args[j],
         param [j]);

     return currMethod;
   }
…
```

Line addition/removal will invalidate descriptions

Looking for a better way…

Question:
What characteristics can uniquely and robustly identify a clone region (CR)?

Methodology:
Manually inspect ~600 CRs (in 4 different Java systems)

Answer (observations):

1. CR are constrained within the boundaries of code blocks
2. Some structural elements are unique at a given nesting level

Clone Region Descriptors (CRDs)

Clone Region Descriptors

```
public class DeleteAction
{
  …
  public void run(int x)
  {
    …
    for(int i=x;i<map.size(); i++)
    {
              A
    }
    …
  }
}
```

Description of Region A

(simplified)

<file>=	**DeleteAction.java**
<class>=	**DeleteAction**
<method>=	**run(int)**
<block type>=	**for**
<anchor>=	**“i<map.size()”**

Clone Region Descriptors

```
public class DeleteAction
{
…
   public void run(int x){
   …
     try{

              B

    }catch(IOException e){…
    }catch(Exception e){…}
   …
   }
…
}
```

Description of Region B

(simplified)

DeleteAction.java

DeleteAction

run(int)

try

IOException, Exception

Clone Region Descriptors: Conflicts

```
public class DeleteAction
{
  …
  public void run(int x)
  {
    …
    for(int i=x;i<map.size(); i++)
    {
              A
    }
    …
    for(int i=x;i<map.size(); i++)
    {
              B
    }

  }
}
```

- Basic CRDs are not always unique
- Non-trivial differences generally exist in the logic implemented by each block
- CRD capture these differences in a **corroboration metric**

Current corroboration metric:

fan-out, **cyclomatic complexity, decision density**

CRD Model

```
<CRD>     ::= <file> <class> <CM> [<method>]
<method> ::= <signature> <CM> <block>*
<block>    ::= <btype> <anchor> <CM>
<btype>    ::= ’for’ | ’while’ | ’do’ | ’if’ | ‘else’ |
                  ’switch’ | ’try’ | ’catch’ | ‘finally’
```

Block types and anchors

for **(all loops)**	**if/else**	**switch**	**try/** **finally**	**catch**
Termination condition	Branching predicate	Switch expression	List of exceptions thrown	Type of exception caught

Clone Region Descriptors

```
public class DeleteAction
{
  …
  public void run(int x)
  {
    …
    for(int i=x;i<map.size(); i++)
    {
                      A
    }
    …
  }
}
```

Description of Region A

(with corroboration metrics)

<file> = DeleteAction.java

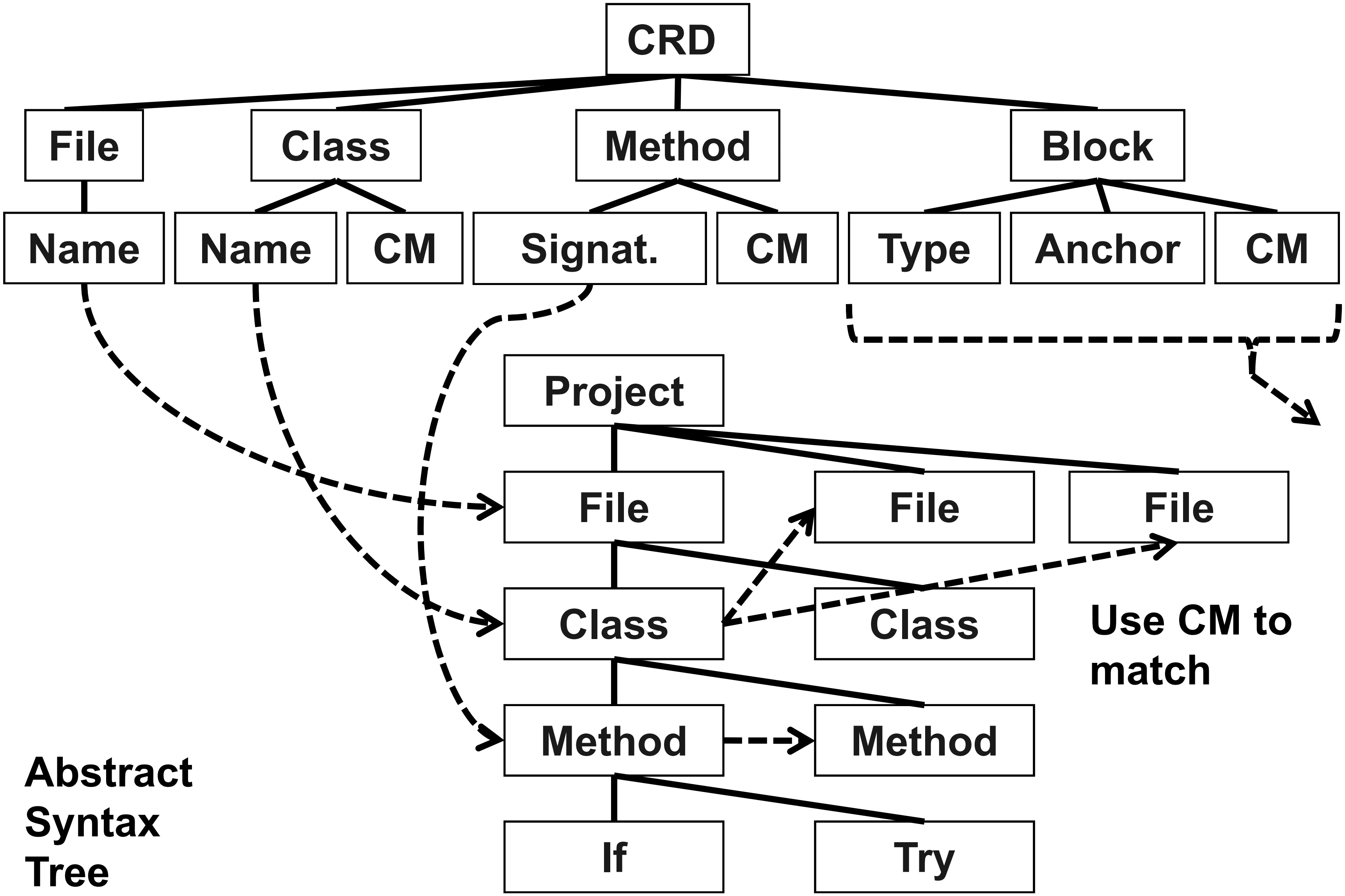
<class>,<CM> = DeleteAction, **21**

<method>,<CM> = run(int), **11**

<btype>,<CM> = for, **6**

<anchor> = “i<map.size()”

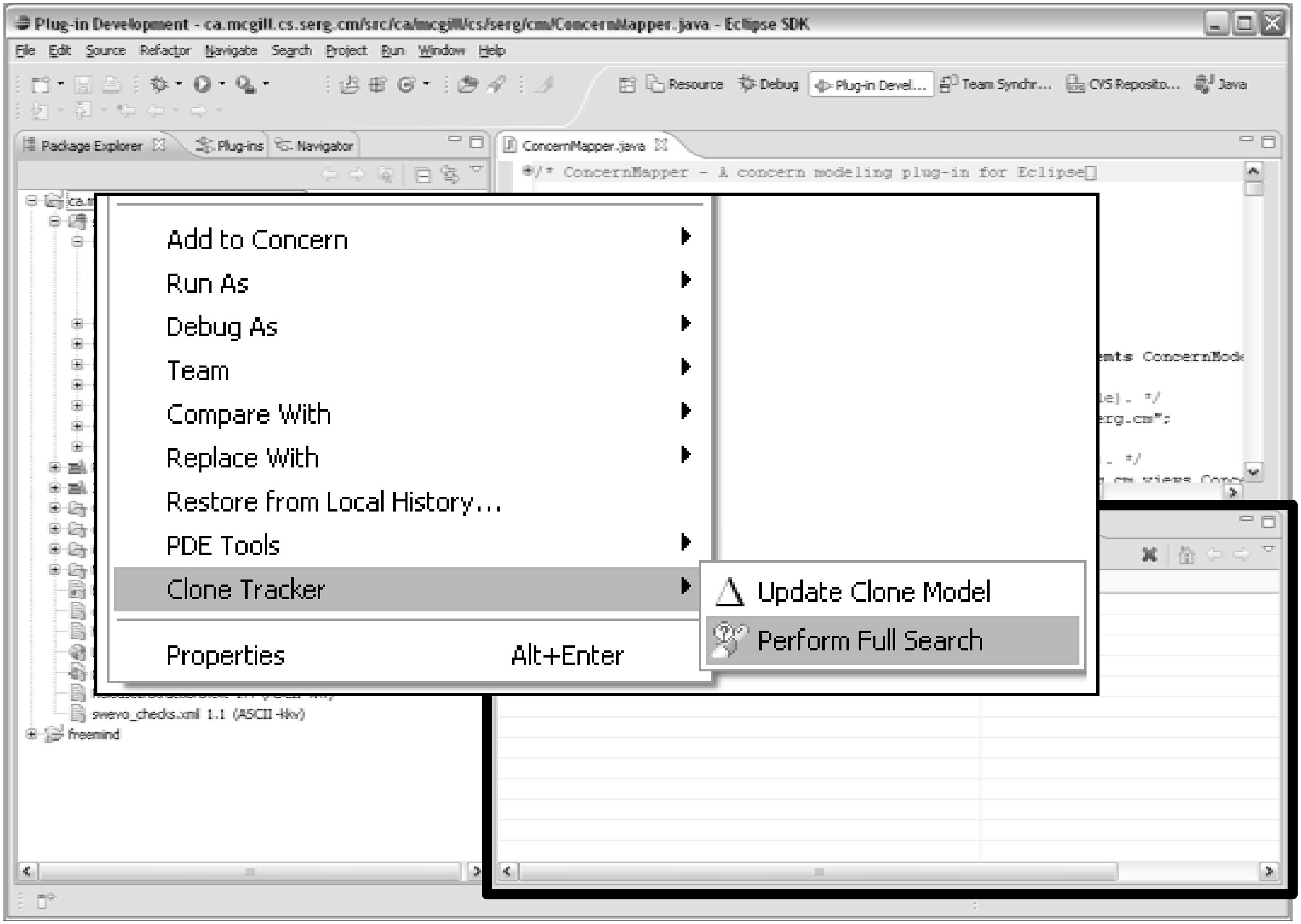
Clone Region Lookup Algorithm



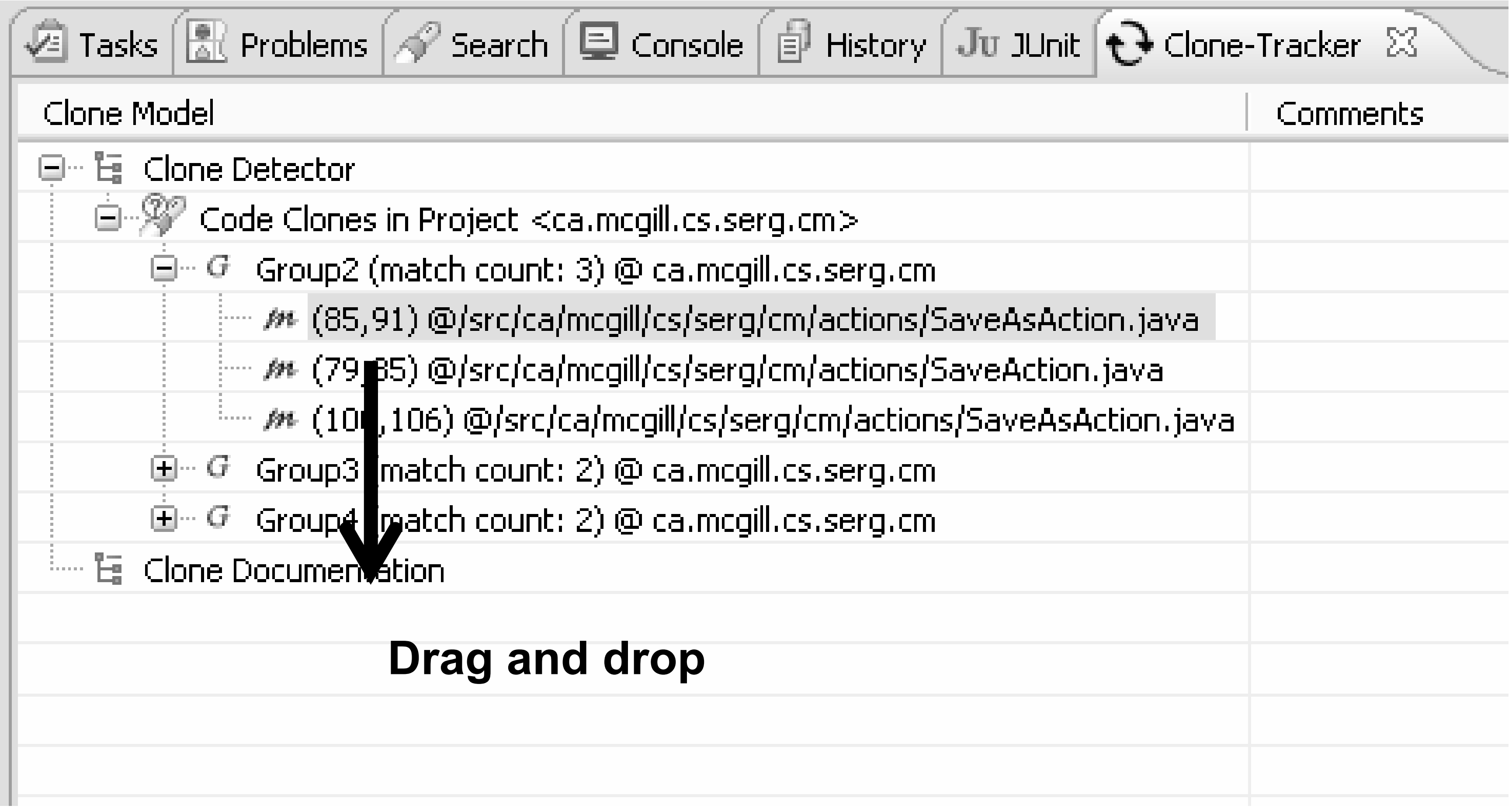
Use CM to match

Abstract Syntax Tree

The CloneTracker Eclipse Plug-in


Plug-in Development - ca.mcgill.cs.serg.cm/src/ca/mcgill/cs/serg/cm/ConcernMapper.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Resource
Debug
Plug-in Devel...
Team Synchr...
CVS Reposito...
Java
Package Explorer
Plug-ins
Navigator
ConcernMapper.java
/* ConcernMapper - A concern modeling plug-in for Eclipse
Add to Concern
Run As
Debug As
Team
Compare With
Replace With
Restore from Local History...
PDE Tools
Clone Tracker
Properties
Alt+Enter
Update Clone Model
Perform Full Search
swevo_checks.xml 1.1 (ASCII -kkv)
freemind

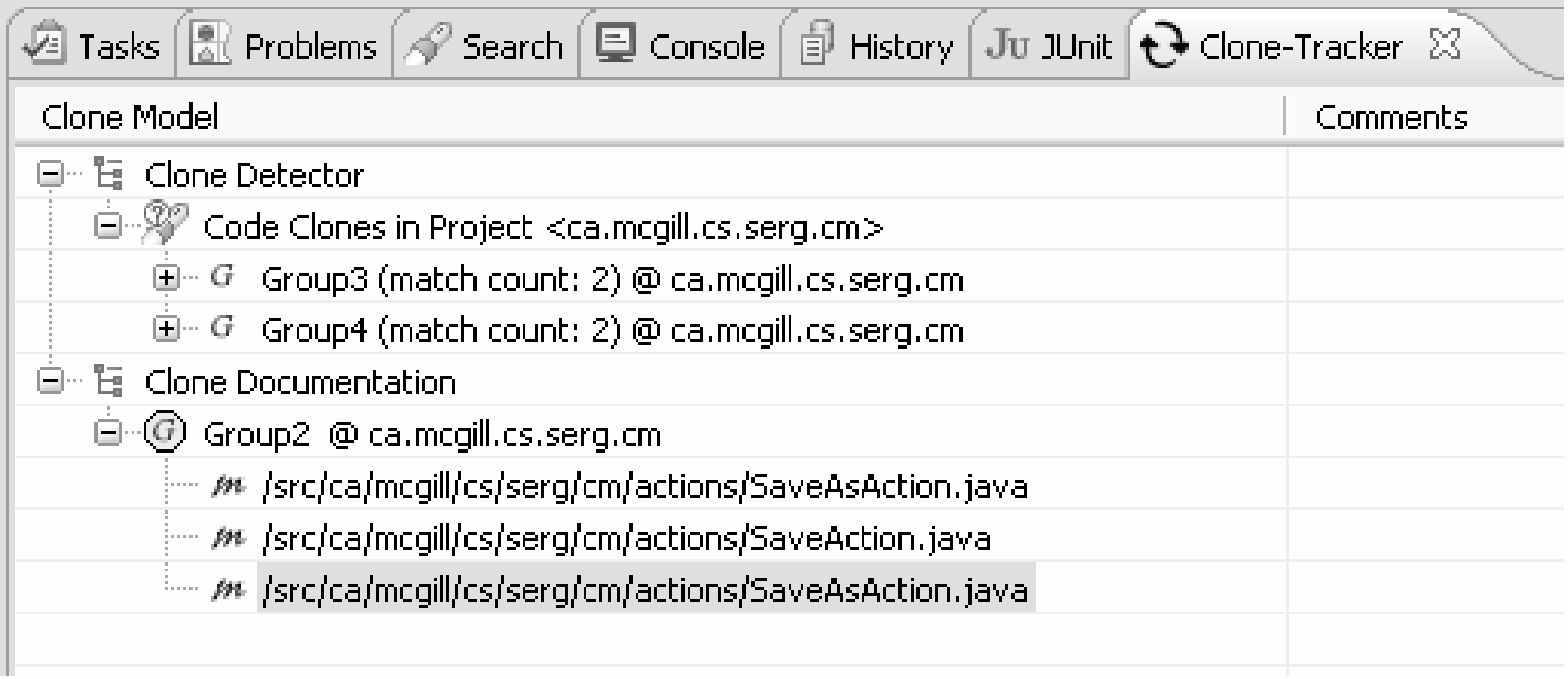
The CloneTracker View



```
Clone Detector
  Code Clones in Project <ca.mcgill.cs.serg.cm>
    Group2 (match count: 3) @ ca.mcgill.cs.serg.cm
      (85,91) @/src/ca/mcgill/cs/serg/cm/actions/SaveAsAction.java
      (79,85) @/src/ca/mcgill/cs/serg/cm/actions/SaveAction.java
      (100,106) @/src/ca/mcgill/cs/serg/cm/actions/SaveAsAction.java
    Group3 (match count: 2) @ ca.mcgill.cs.serg.cm
    Group4 (match count: 2) @ ca.mcgill.cs.serg.cm
Clone Documentation
```

Drag and drop

Documenting Clones

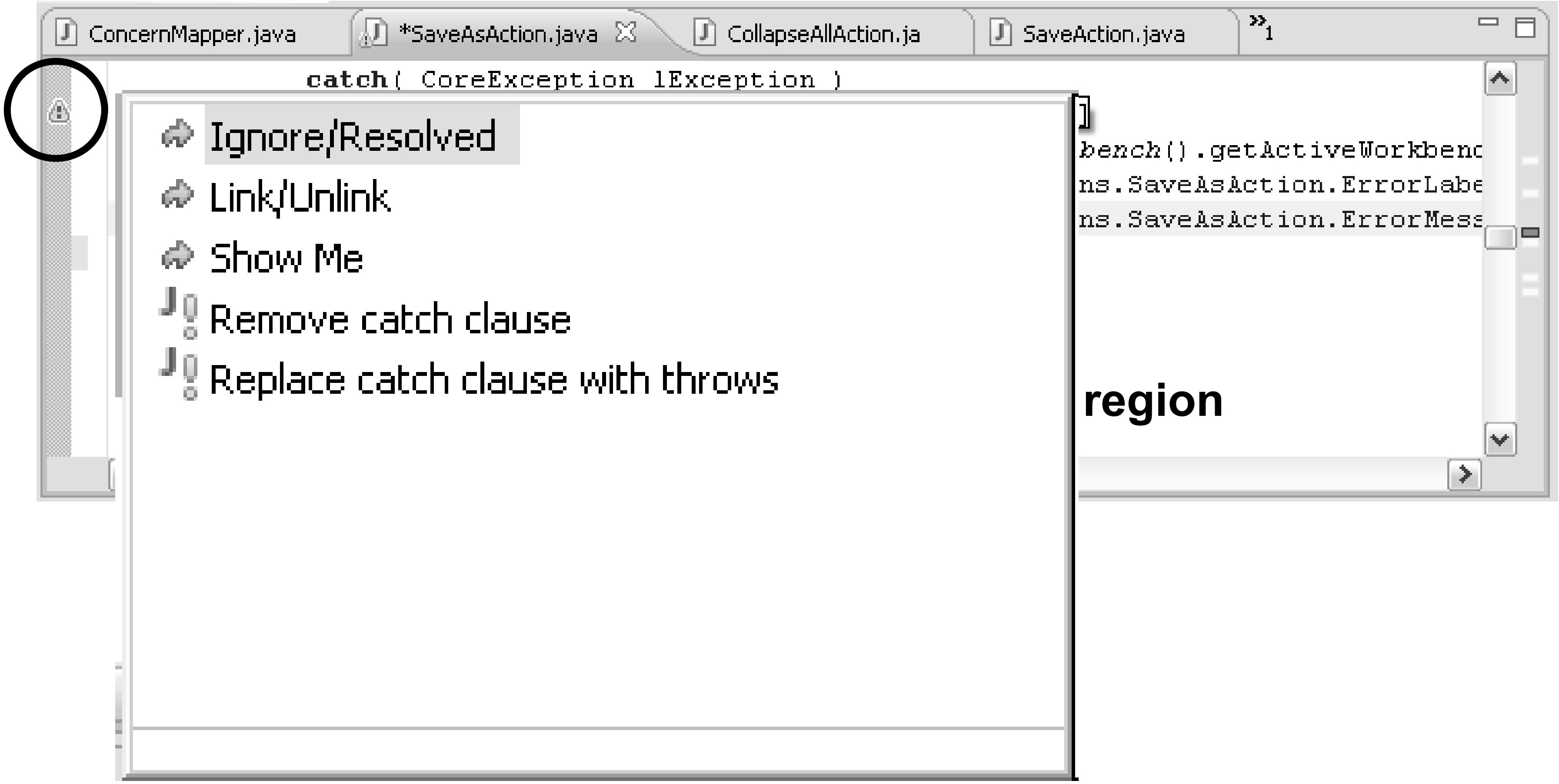


Clone Model | Comments

- Clone Detector
  - Code Clones in Project <ca.mcgill.cs.serg.cm>
    - Group3 (match count: 2) @ ca.mcgill.cs.serg.cm
    - Group4 (match count: 2) @ ca.mcgill.cs.serg.cm
- Clone Documentation
  - Group2 @ ca.mcgill.cs.serg.cm
    - /src/ca/mcgill/cs/serg/cm/actions/SaveAsAction.java
    - /src/ca/mcgill/cs/serg/cm/actions/SaveAction.java
    - /src/ca/mcgill/cs/serg/cm/actions/SaveAsAction.java

Editing Clone Regions

Code region is part of a clone group…



```
catch( CoreException lException )
]
bench().getActiveWorkbenc
ns.SaveAsAction.ErrorLabe
ns.SaveAsAction.ErrorMess
```

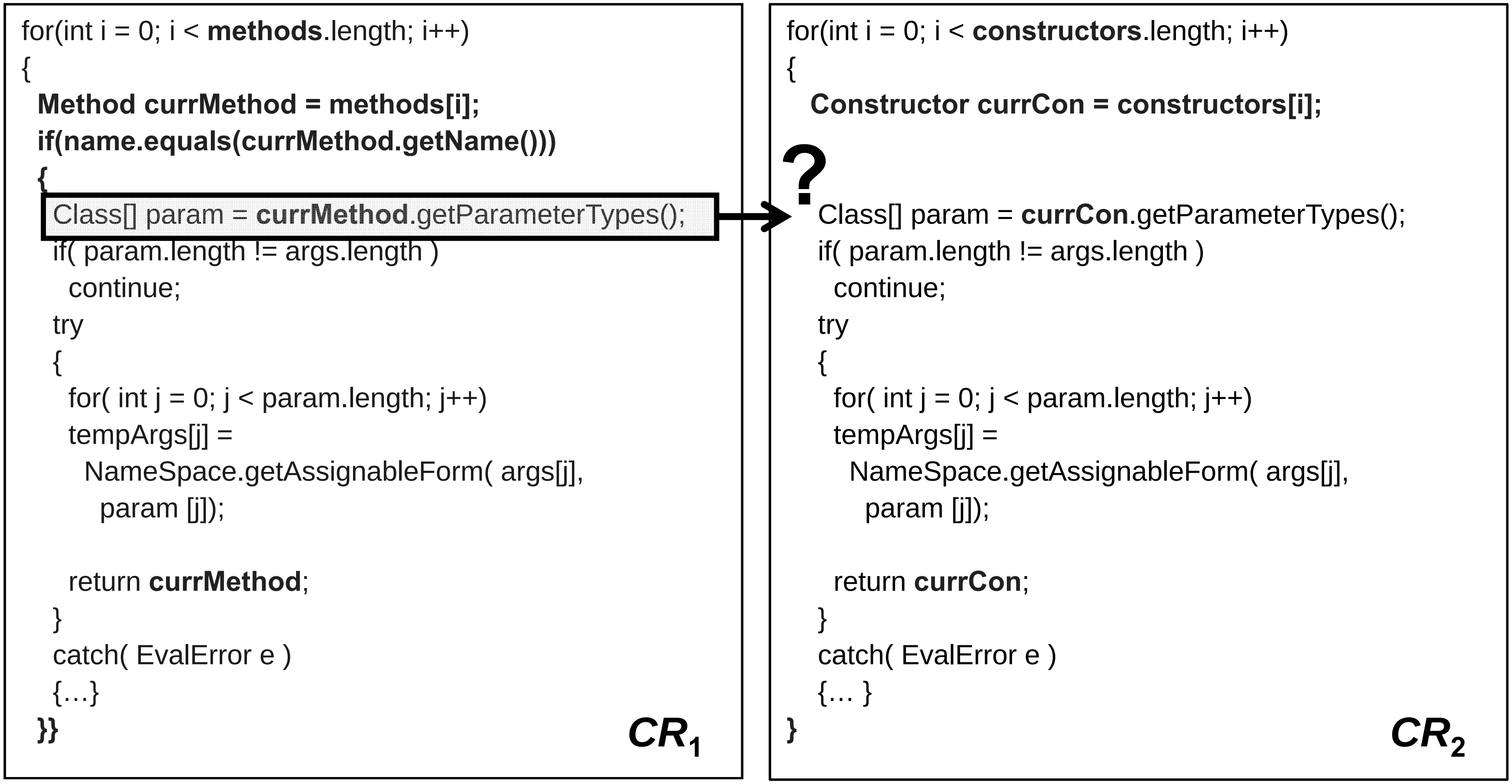
region

Outline

1. Clone documentation model and tool
2. Advanced tool-based features
   - ▪ Simultaneous editing
   - ▪ Incremental clone detection
3. Evaluation of the technique

Support for Simultaneous Editing

We used **Levenshtein Distance** (LD) to identify the line L_s in clone region CR_1 that corresponds to L_t in CR_2



```
for(int i = 0; i < methods.length; i++)
{
  Method currMethod = methods[i];
  if(name.equals(currMethod.getName()))
  {
    Class[] param = currMethod.getParameterTypes();
    if( param.length != args.length )
      continue;
    try
    {
      for( int j = 0; j < param.length; j++)
      tempArgs[j] =
        NameSpace.getAssignableForm( args[j],
          param [j]);

      return currMethod;
    }
    catch( EvalError e )
    {…}
  }}
```

CR_1

?

```
for(int i = 0; i < constructors.length; i++)
{
  Constructor currCon = constructors[i];


    Class[] param = currCon.getParameterTypes();
    if( param.length != args.length )
      continue;
    try
    {
      for( int j = 0; j < param.length; j++)
      tempArgs[j] =
        NameSpace.getAssignableForm( args[j],
          param [j]);

      return currCon;
    }
    catch( EvalError e )
    {… }
}
```

CR_2

Support for Simultaneous Editing

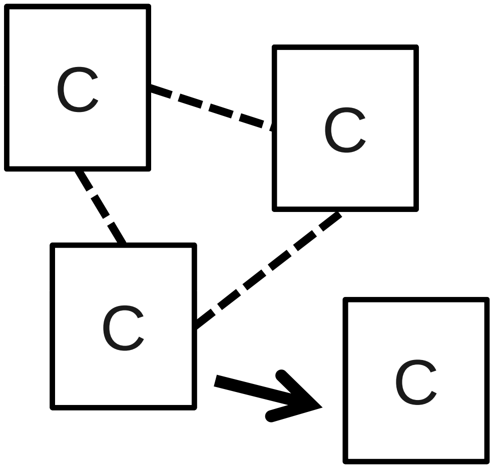
- Similarity between L_s & L_t:

$$\alpha = 1 - \frac{LD(L_t, L_s)}{\max\left(\left|L_t\right|, \left|L_s\right|\right)}$$

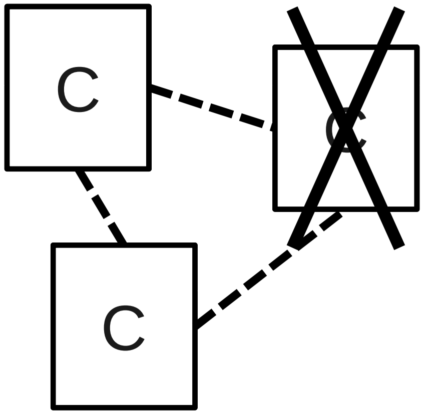
- Maintain a list of lines with **α** > ***sim*** $\boldsymbol{sim_{th}}$ (similarity threshold)
- For conflicts: Use previous and/or next line in CR to resolve conflicts

Incremental Clone Detection

Modifications following a CRD specification may invalidate the state of the documented clone relationship



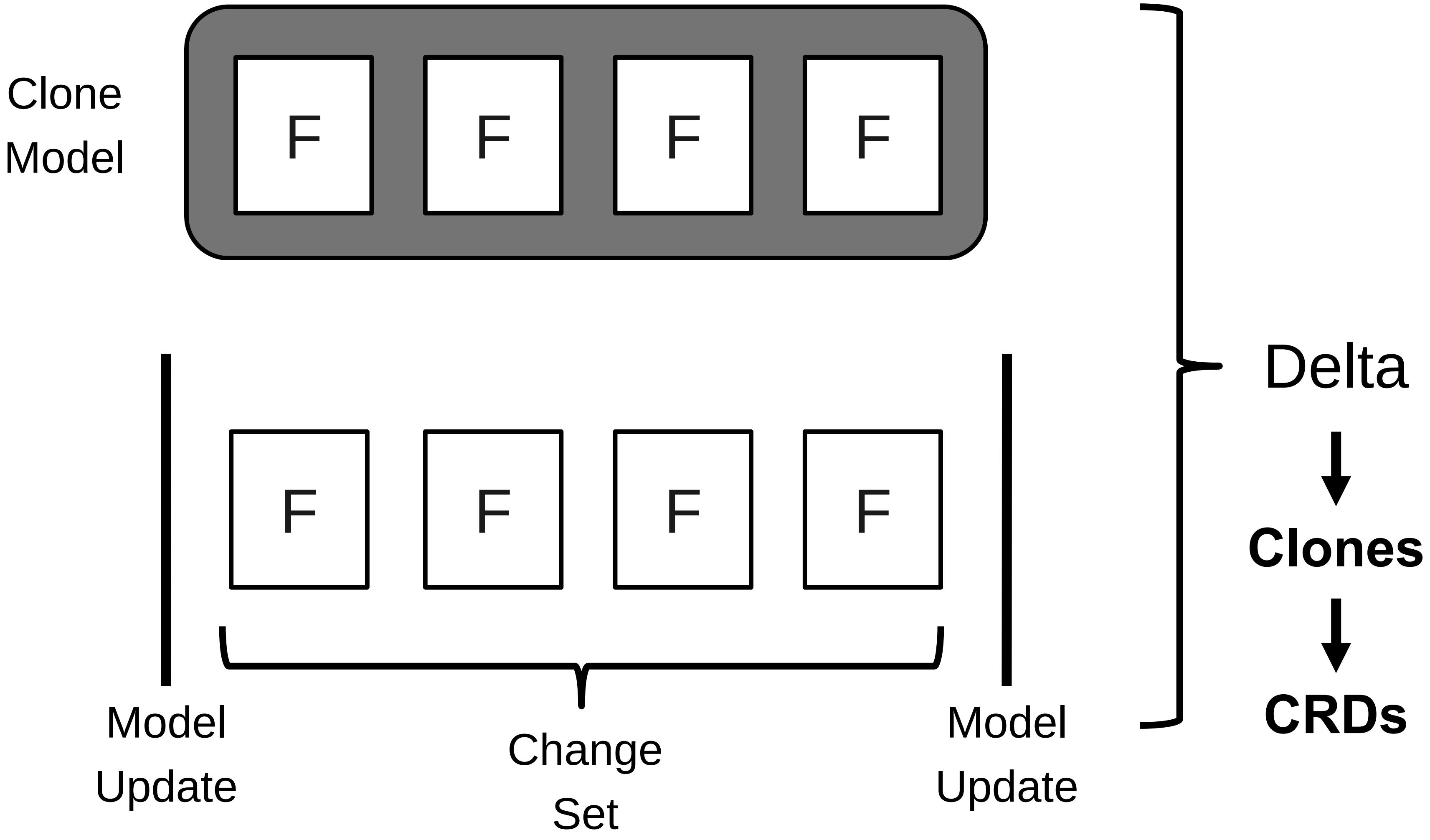
Copy and paste



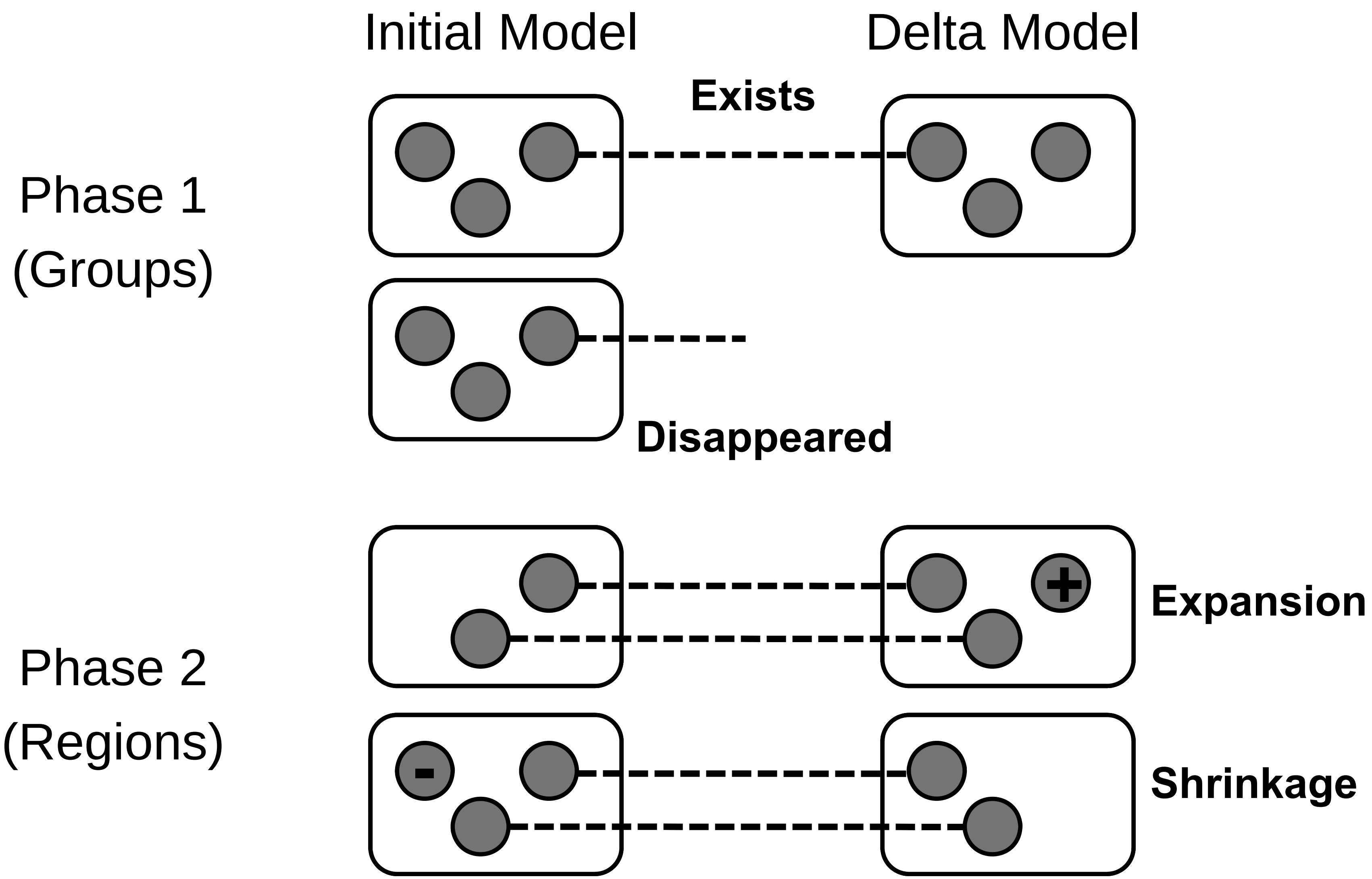
Elimination

Solution: Perform clone detection only on a subset of the code

Development Session for Model Update



Comparing CRDs



Initial Model

Delta Model

Phase 1
(Groups)

Exists

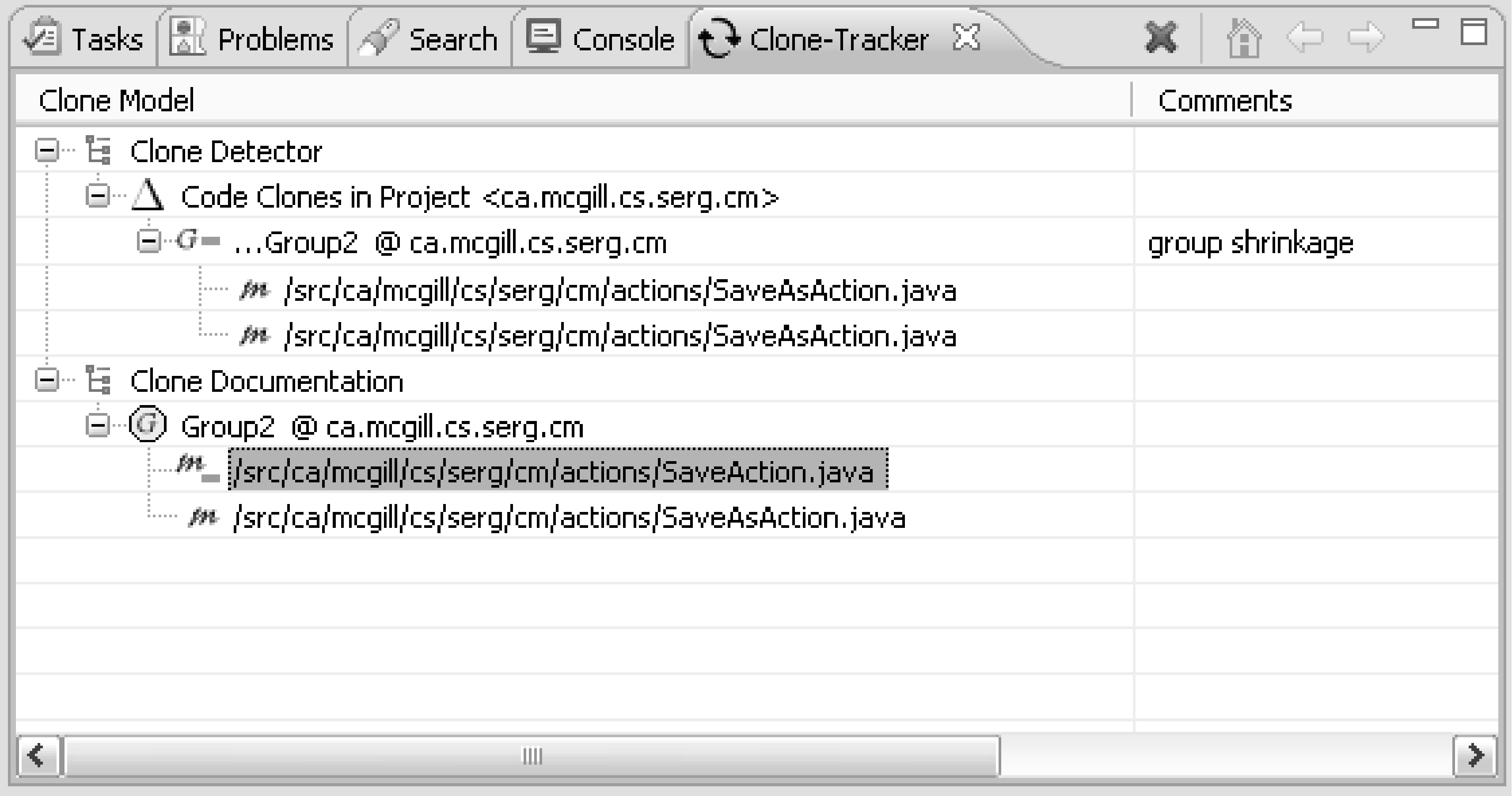
Disappeared

Phase 2
(Regions)

Expansion

Shrinkage

Reporting Model Updates



Clone Model	Comments
Clone Detector	
Code Clones in Project <ca.mcgill.cs.serg.cm>	
...Group2 @ ca.mcgill.cs.serg.cm	group shrinkage
/src/ca/mcgill/cs/serg/cm/actions/SaveAsAction.java	
/src/ca/mcgill/cs/serg/cm/actions/SaveAsAction.java	
Clone Documentation	
Group2 @ ca.mcgill.cs.serg.cm	
/src/ca/mcgill/cs/serg/cm/actions/SaveAction.java	
/src/ca/mcgill/cs/serg/cm/actions/SaveAsAction.java	

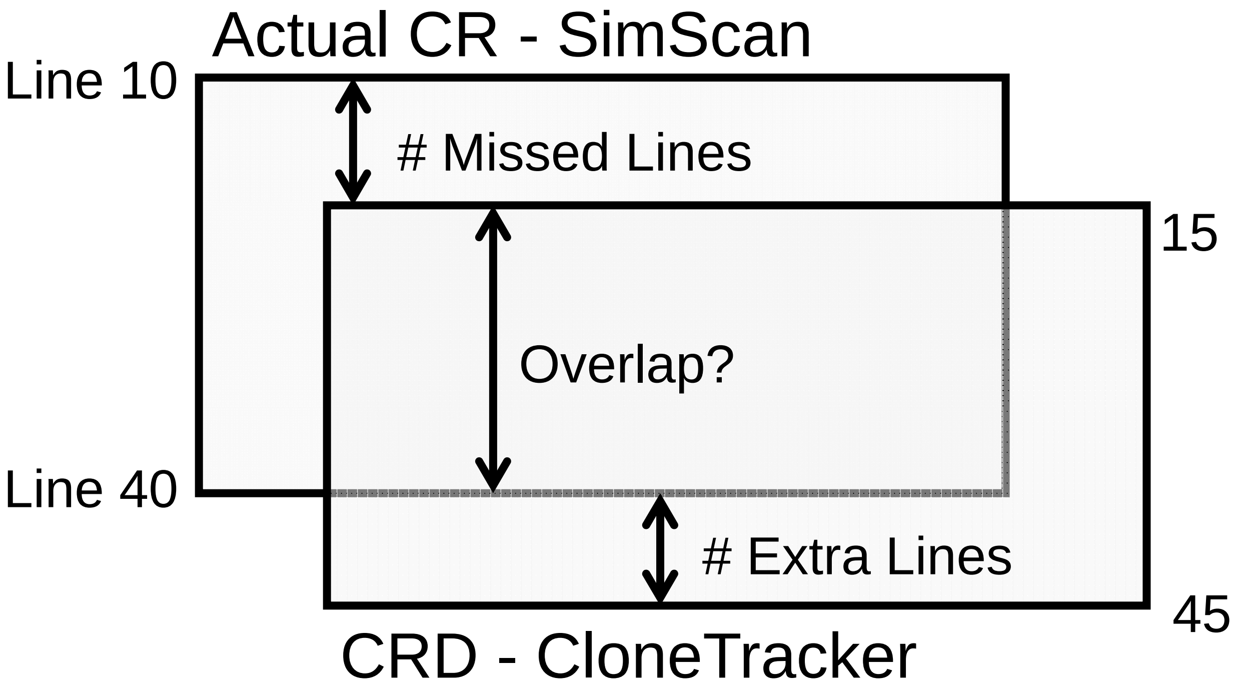
Outline

1. Clone documentation model and tool
2. Advanced tool-based features
   - Simultaneous editing
   - Incremental clone detection
3. Evaluation of the technique

Precision of CRDs

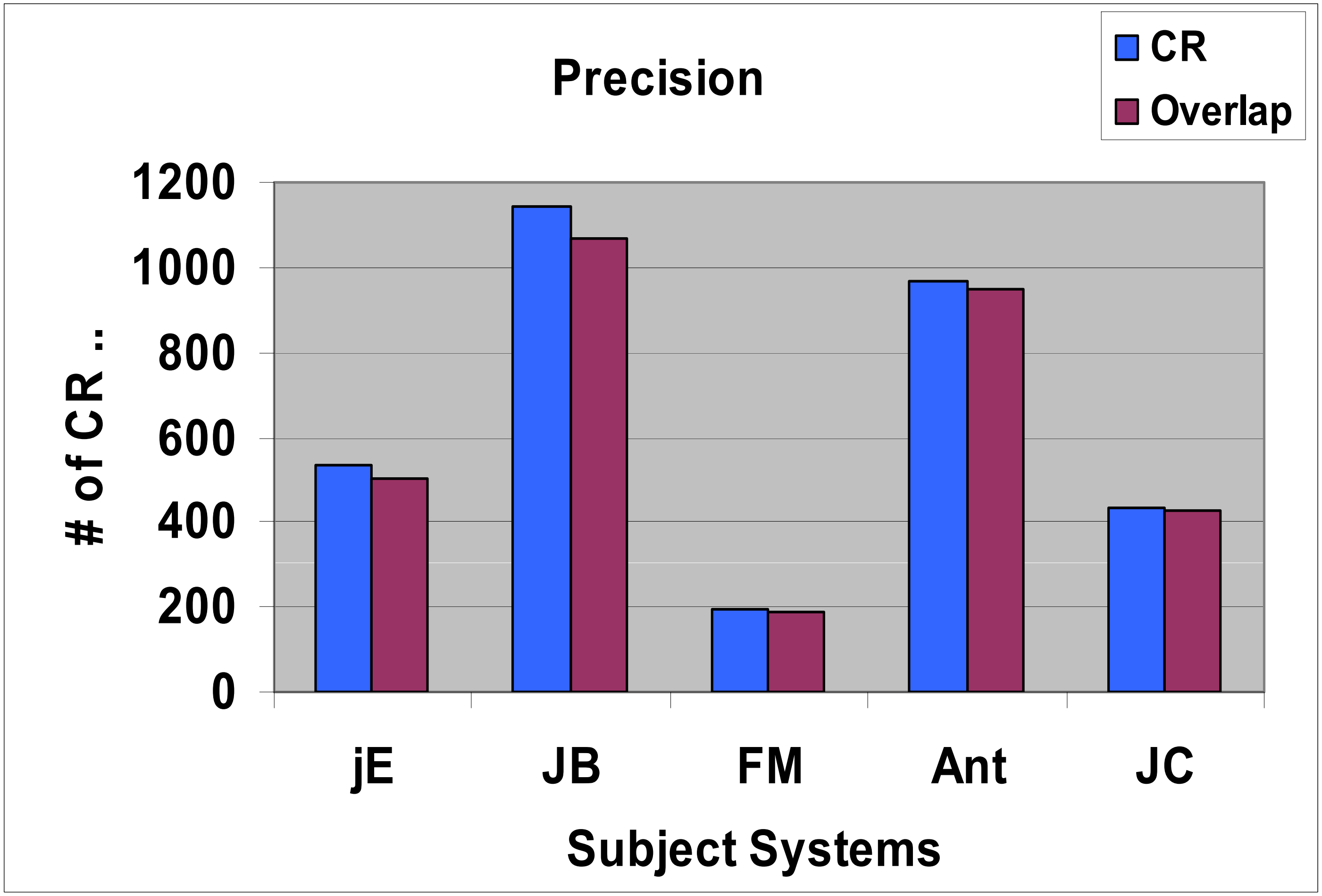
- Do CRDs accurately represent clone regions?

Precision Metrics



- Subject systems:
  jEdit, **JBossAOP**, **Ant**, **FreeMind**, **JCommander**.

Precision of CRDs



Avg. CR	24.6
Avg. ML	1.8
Avg. EL	3.5

Total # of CR: **3,275**

Total Overlap: **96%**

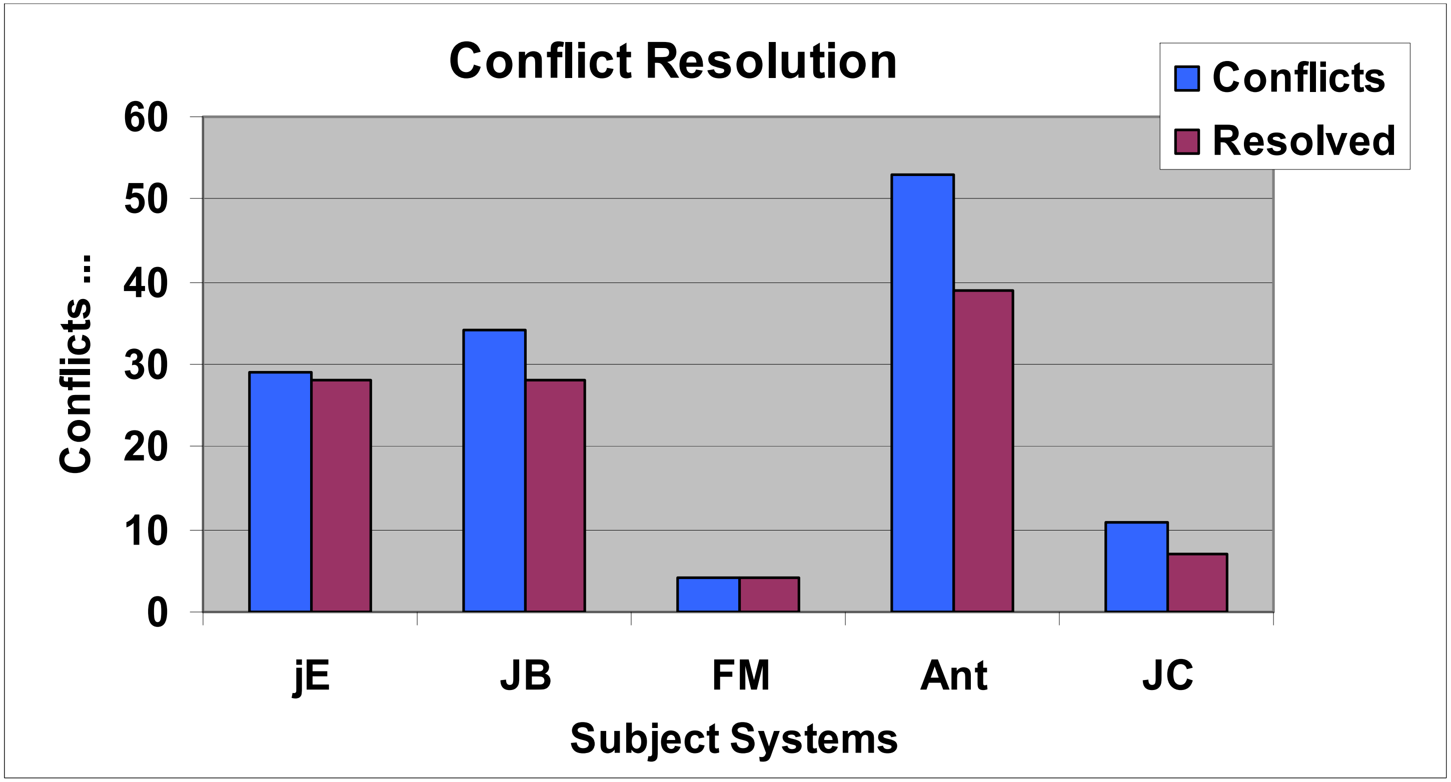
Why the Differences?

```
/**
 * Converts a String in the format "value;value;value"
 * to a List with the values (as strings)
 */
public static List stringToList(String string)
{
    StringTokenizer tok = new StringTokenizer(string, ";");
    List list = new LinkedList();
    while (tok.hasMoreTokens()) {
        list.add(tok.nextToken());
    }
    return list;
}
```

SimScan Region

CRD Borders

Precision of CRDs: Conflicts



Total Conflicts: 131 / 3275 = **4%** of clone regions

Total Resolved: 106 / 131 = **81%** of conflicts

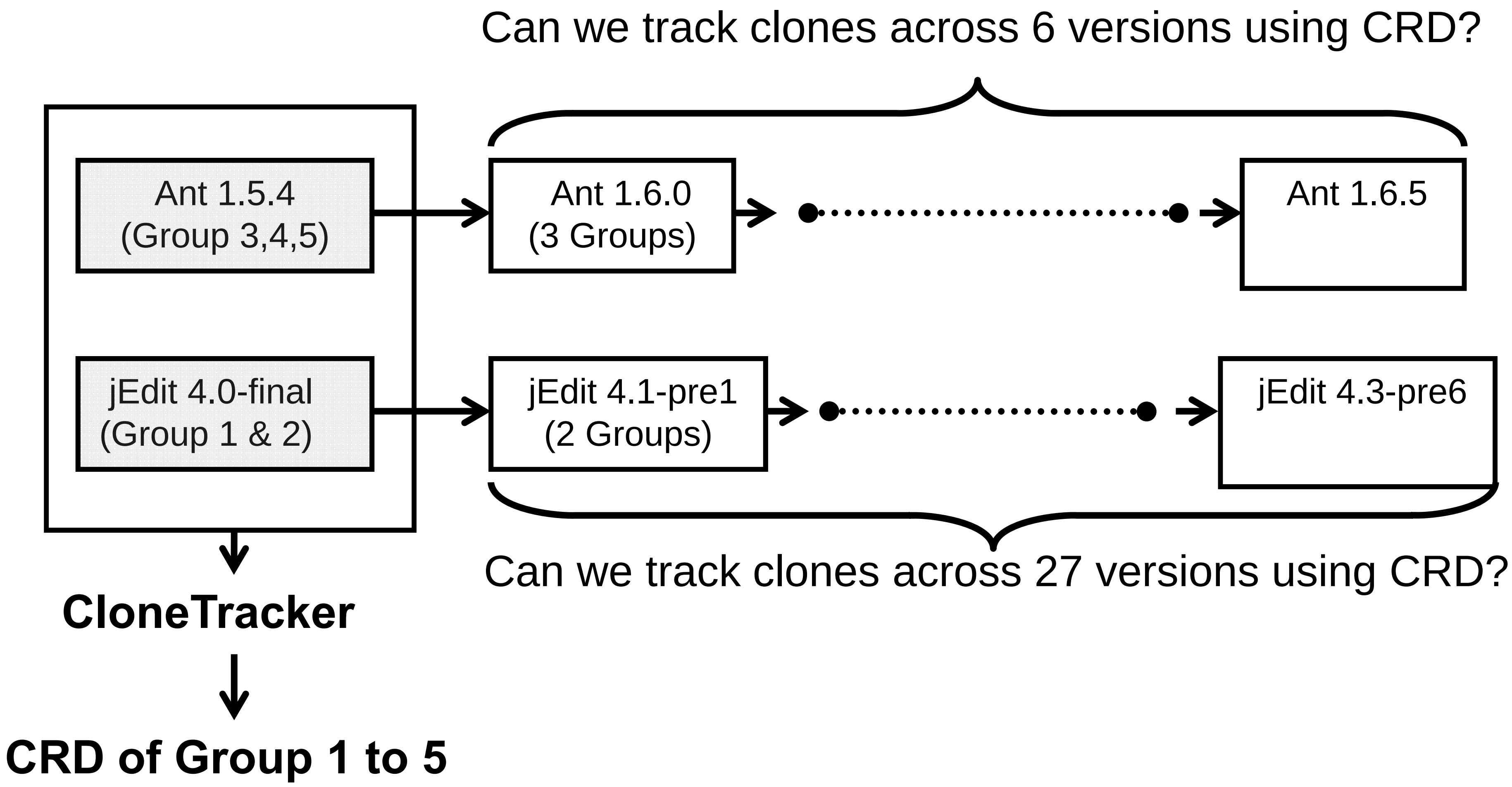
Unresolved: 25 / 3275 = **0.8%** of clone regions

Improvements Based on the Study

- A detailed analysis of the missed regions in Ant revealed:
  - Suboptimal representation of the else and finally blocks -> new block types
  - Implementation issues -> fixed
  - Unresolved conflicts -> new corroboration metric
- Rerunning all the experiments (except for Ant)
  - Conflict resolution increased from 82% to 86%
  - Overall precision increased from 97% to 99%

Case Study: Tracking Code Clones

Can CRDs track clones across subsequent versions?



Can we track clones across 6 versions using CRD?

Ant 1.5.4 (Group 3,4,5)

Ant 1.6.0 (3 Groups)

Ant 1.6.5

jEdit 4.0-final (Group 1 & 2)

jEdit 4.1-pre1 (2 Groups)

jEdit 4.3-pre6

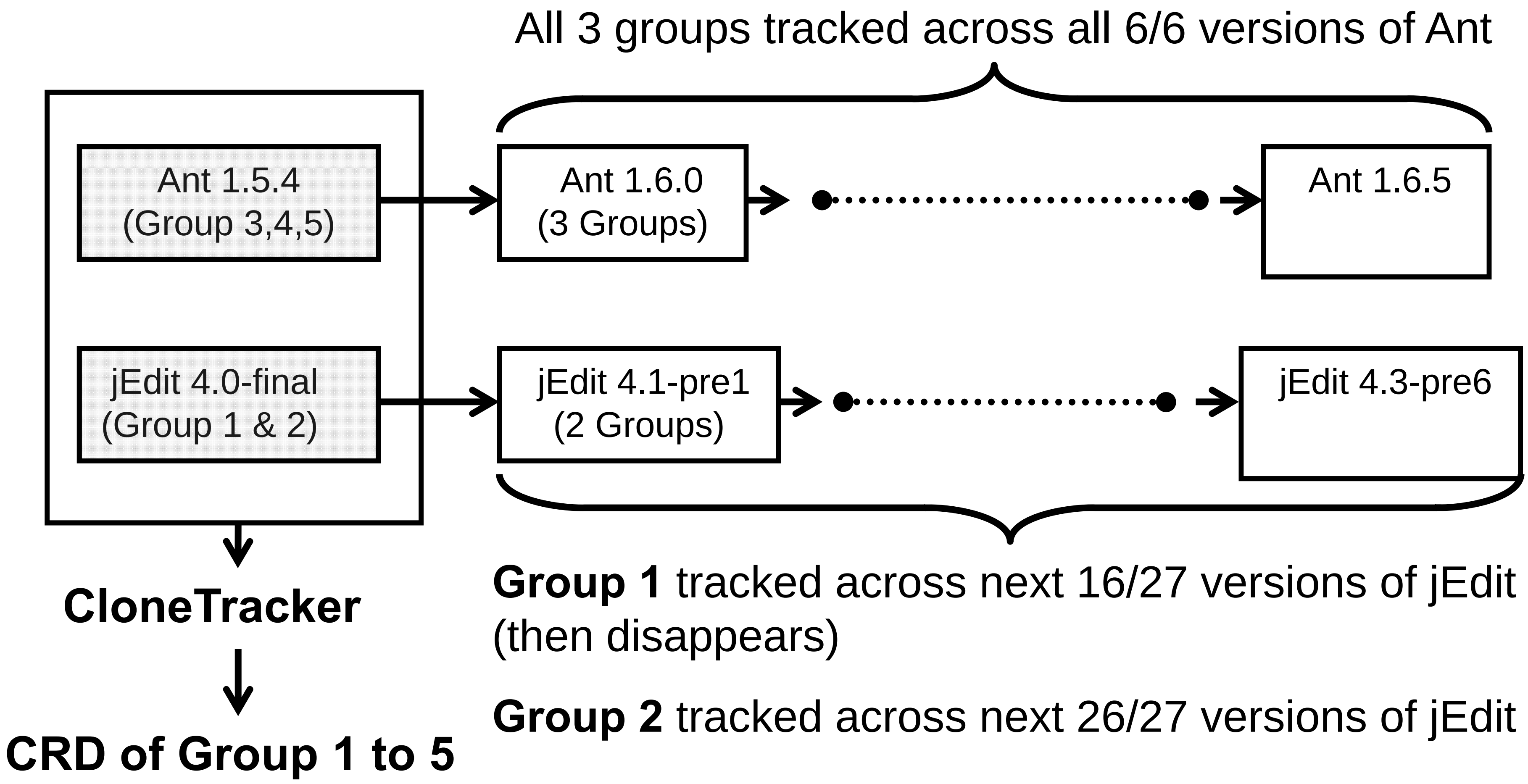
Can we track clones across 27 versions using CRD?

CloneTracker

CRD of Group 1 to 5

Case Study: Tracking Code Clones

Can CRDs track clones across subsequent versions?



All 3 groups tracked across all 6/6 versions of Ant

Ant 1.5.4 (Group 3,4,5)

jEdit 4.0-final (Group 1 & 2)

Ant 1.6.0 (3 Groups)

Ant 1.6.5

jEdit 4.1-pre1 (2 Groups)

jEdit 4.3-pre6

CloneTracker

CRD of Group 1 to 5

Group 1 tracked across next 16/27 versions of jEdit (then disappears)

Group 2 tracked across next 26/27 versions of jEdit

Conclusion

- Refactoring of code clones is not always feasible or cost-effective
- We proposed ***Clone Region Descriptors:***
  - Document and keep track of clone groups across different versions of a system
  - Be notified upon changes that affect a documented clone group
  - Support simultaneous modification to clone regions
  - Support incremental clone detection

Conclusion

- Our simple technique could accurately represent the vast majority of clones detected by Simscan on 5 open-source programs
- See our demo at ICSE 2008!
- http://www.cs.mcgill.ca/~swevo/clonetracker