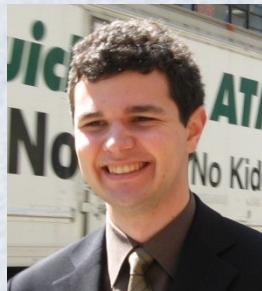


Recommending Adaptive Changes for Framework Evolution



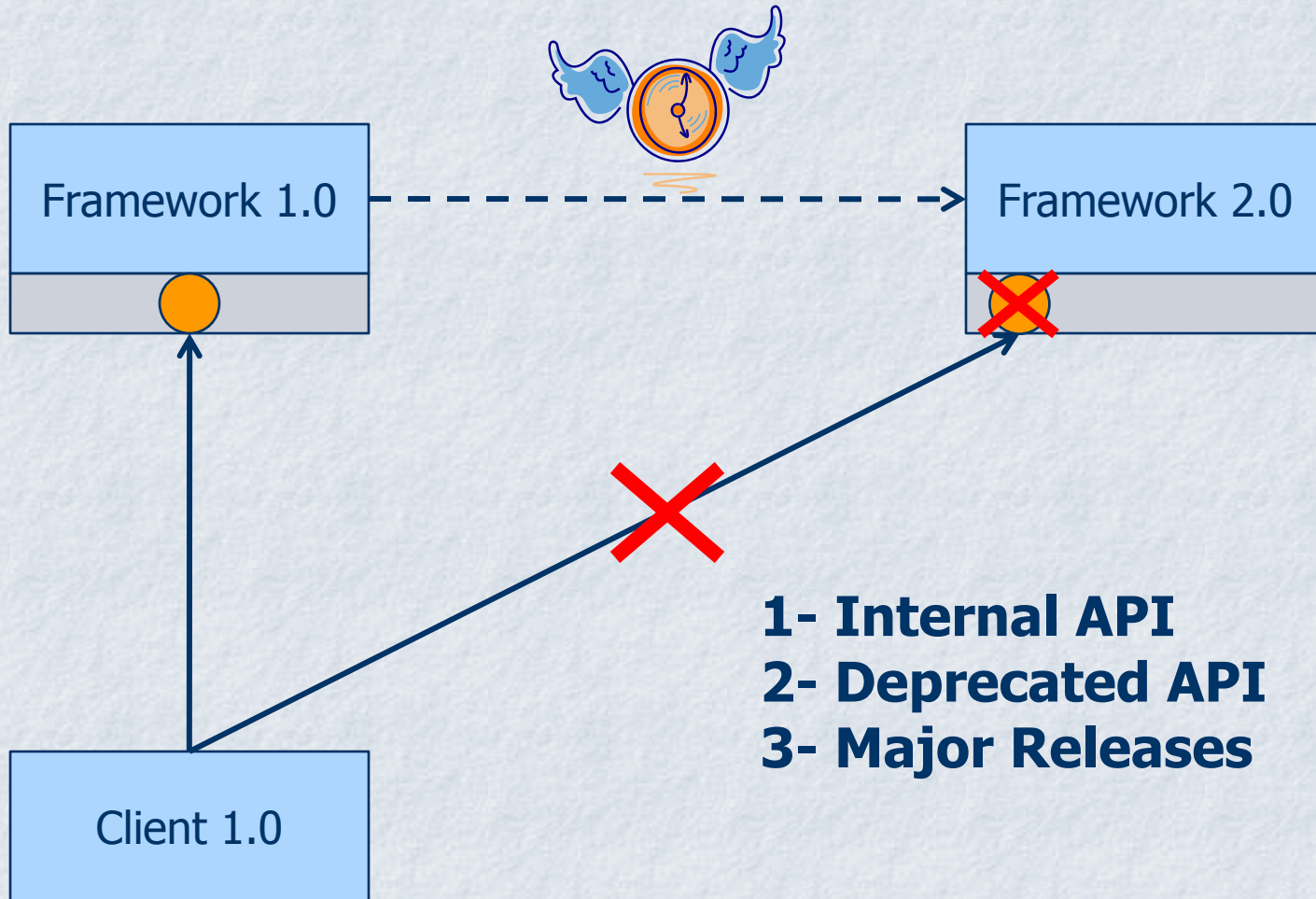
Barthélémy Dagenais



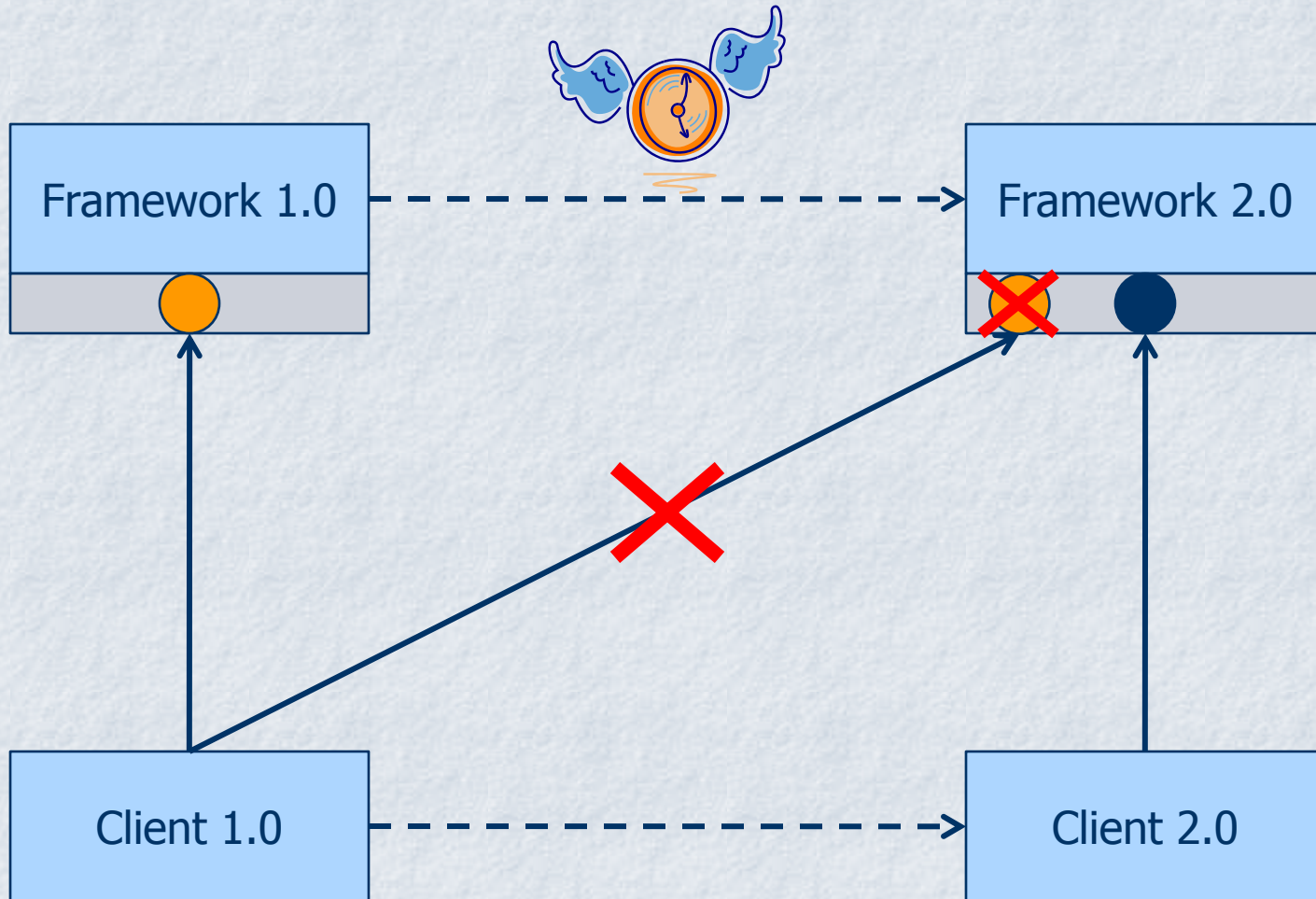
Martin Robillard

McGill University

Framework Evolution



Framework Evolution

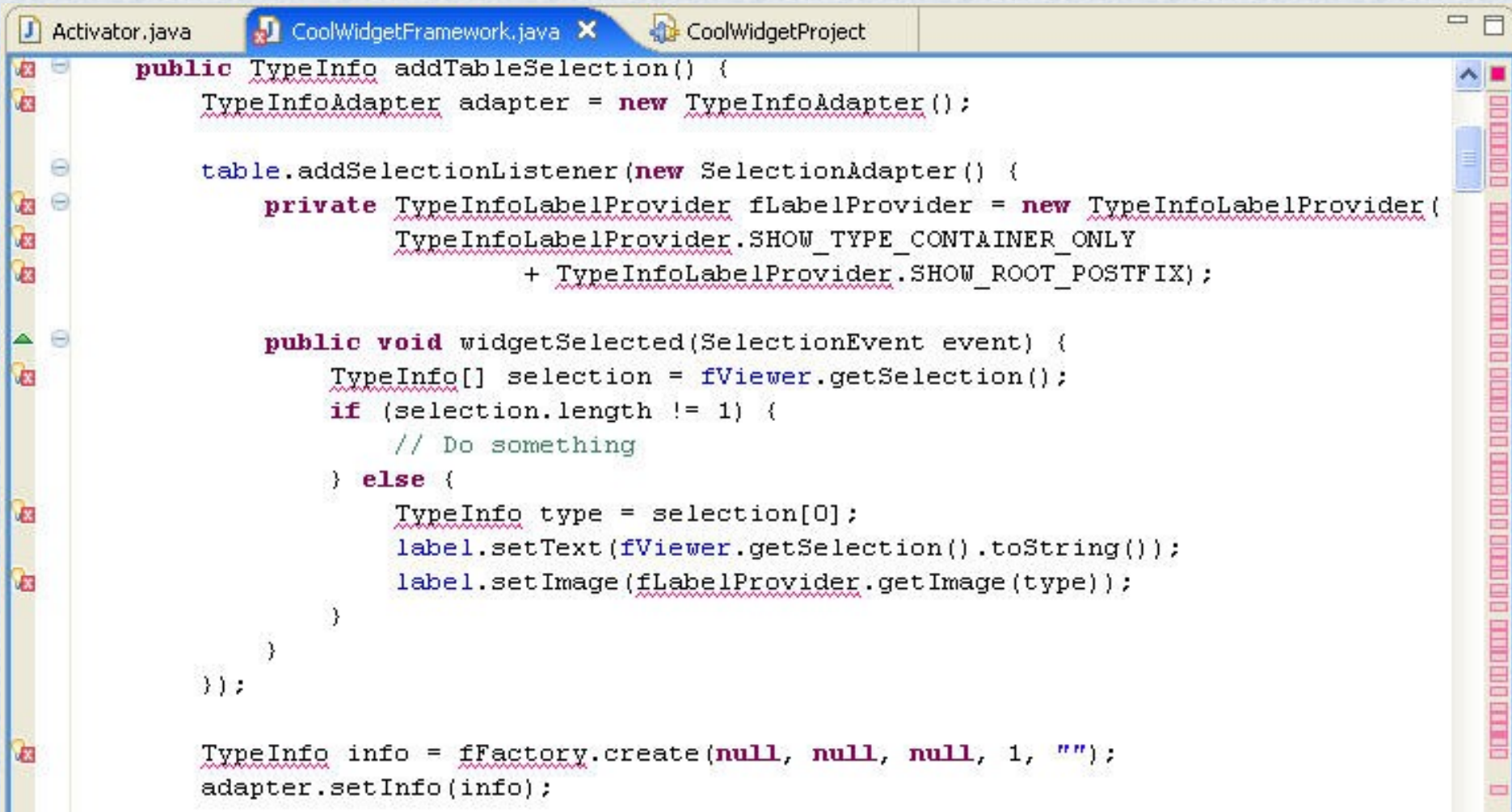


Framework Evolution

```
public TypeInfo addTableSelection() {  
    TypeInfoAdapter adapter = new TypeInfoAdapter();  
  
    table.addSelectionListener(new SelectionAdapter() {  
        private TypeInfoLabelProvider fLabelProvider = new TypeInfoLabelProvider(  
            TypeInfoLabelProvider.SHOW_TYPE_CONTAINER_ONLY  
            + TypeInfoLabelProvider.SHOW_ROOT_POSTFIX);  
  
        public void widgetSelected(SelectionEvent event) {  
            TypeInfo[] selection = fViewer.getSelection();  
            if (selection.length != 1) {  
                // Do something  
            } else {  
                TypeInfo type = selection[0];  
                label.setText(fViewer.getSelection().toString());  
                label.setImage(fLabelProvider.getImage(type));  
            }  
        }  
    });  
  
    TypeInfo info = fFactory.create(null, null, null, 1, "");  
    adapter.setInfo(info);  
}
```

Framework 3.2

Framework Evolution



```
Activator.java CoolWidgetFramework.java X CoolWidgetProject

public TypeInfo addTableSelection() {
    TypeInfoAdapter adapter = new TypeInfoAdapter();

    table.addSelectionListener(new SelectionAdapter() {
        private TypeInfoLabelProvider fLabelProvider = new TypeInfoLabelProvider(
            TypeInfoLabelProvider.SHOW_TYPE_CONTAINER_ONLY
            + TypeInfoLabelProvider.SHOW_ROOT_POSTFIX);

        public void widgetSelected(SelectionEvent event) {
            TypeInfo[] selection = fViewer.getSelection();
            if (selection.length != 1) {
                // Do something
            } else {
                TypeInfo type = selection[0];
                label.setText(fViewer.getSelection().toString());
                label.setImage(fLabelProvider.getImage(type));
            }
        }
    });

    TypeInfo info = fFactory.create(null, null, null, 1, "");
    adapter.setInfo(info);
}
```

Framework 3.3

SemDiff

Framework 1.0

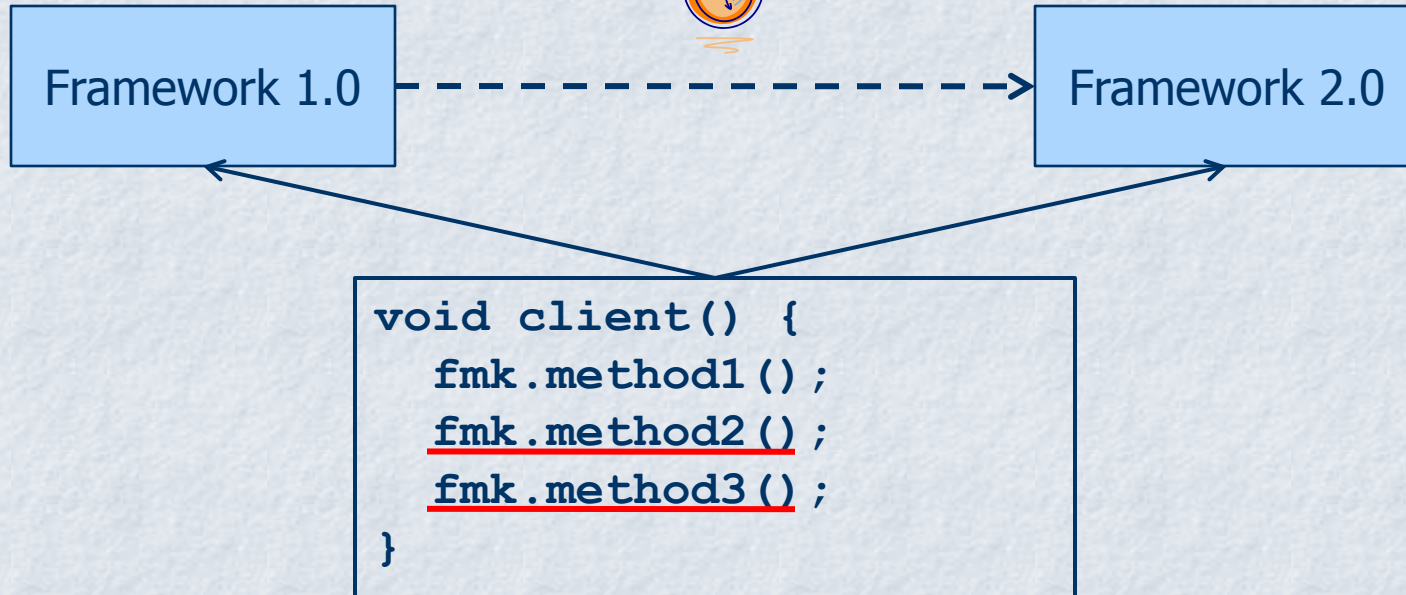
```
void client() {  
    fmk.method1 ();  
    fmk.method2 ();  
    fmk.method3 ();  
}
```

SemDiff

Query: fmk.method2 ()

Response: fmk.m2a ()	100%
Util.mABC ()	75%
fmk.mXYZ ()	50%

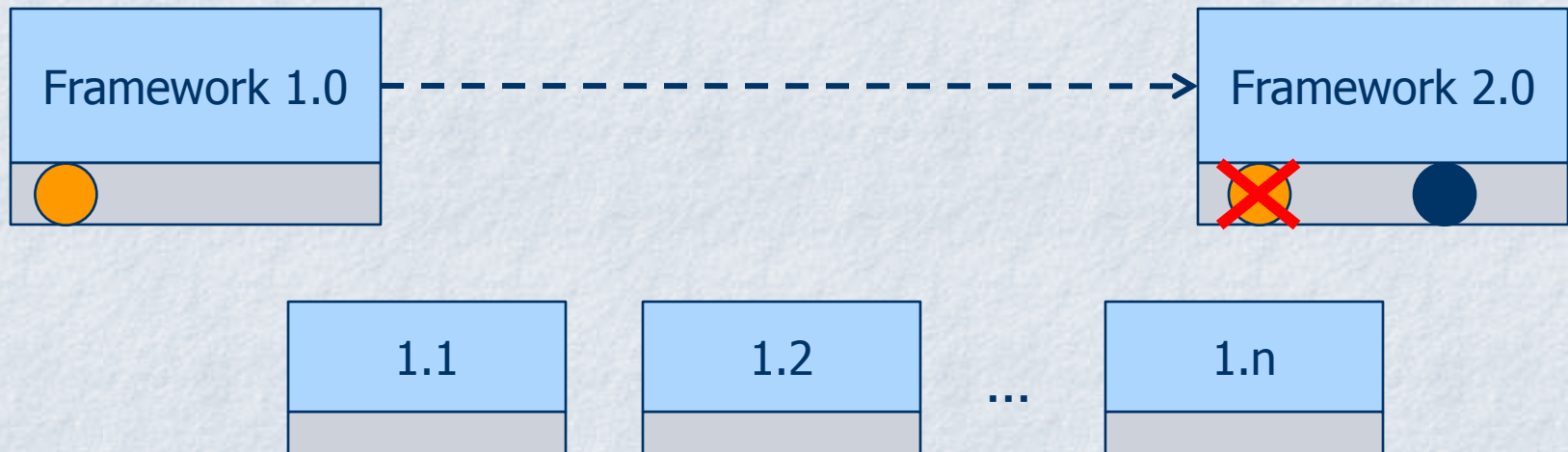
SemDiff



SemDiff

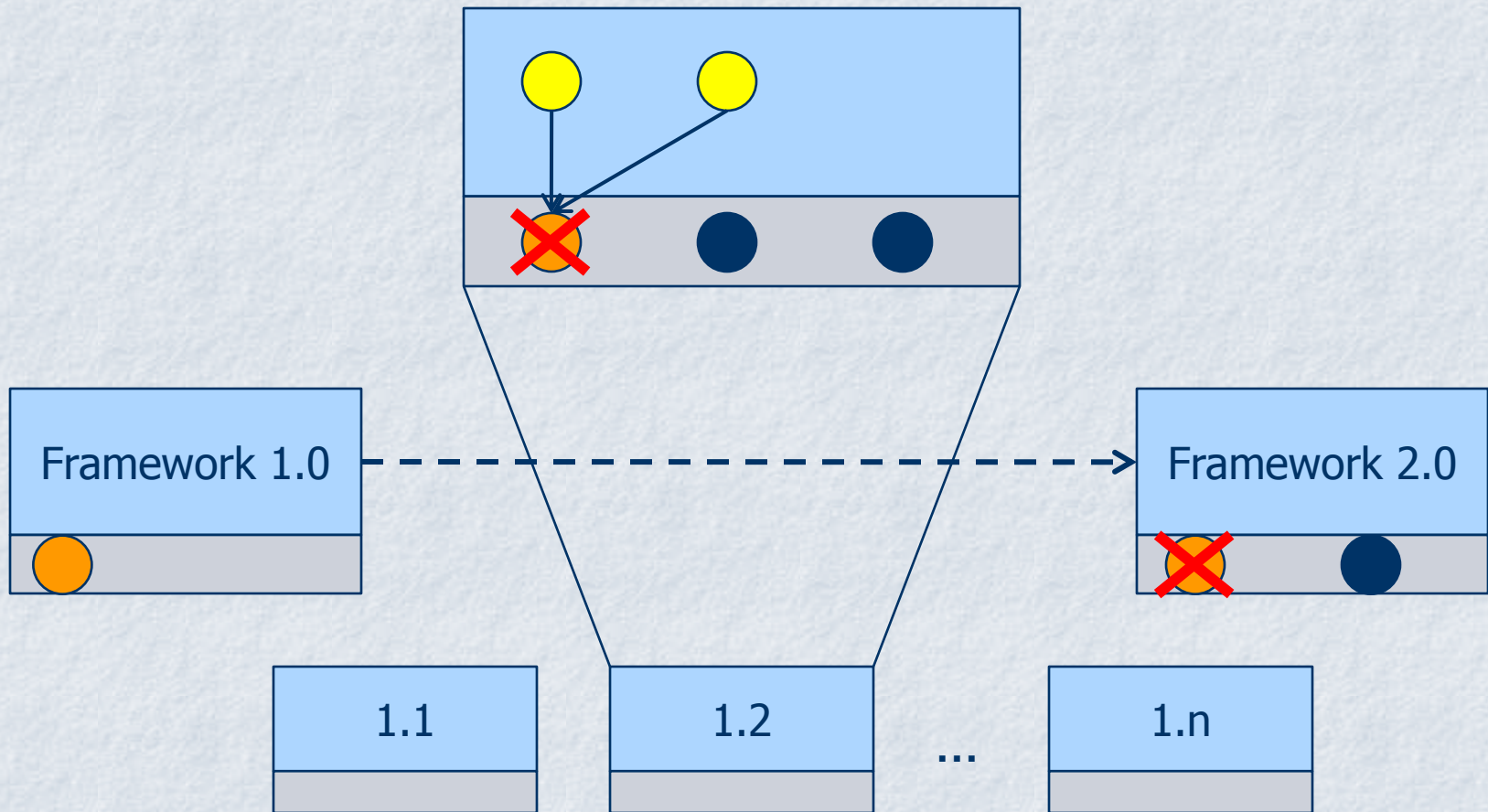
Query:	<code>fmk.method2()</code>	
Response:	<code>fmk.m2a()</code>	100%
	<code>Util.mABC()</code>	75%
	<code>fmk.mXYZ()</code>	50%

Hypothesis



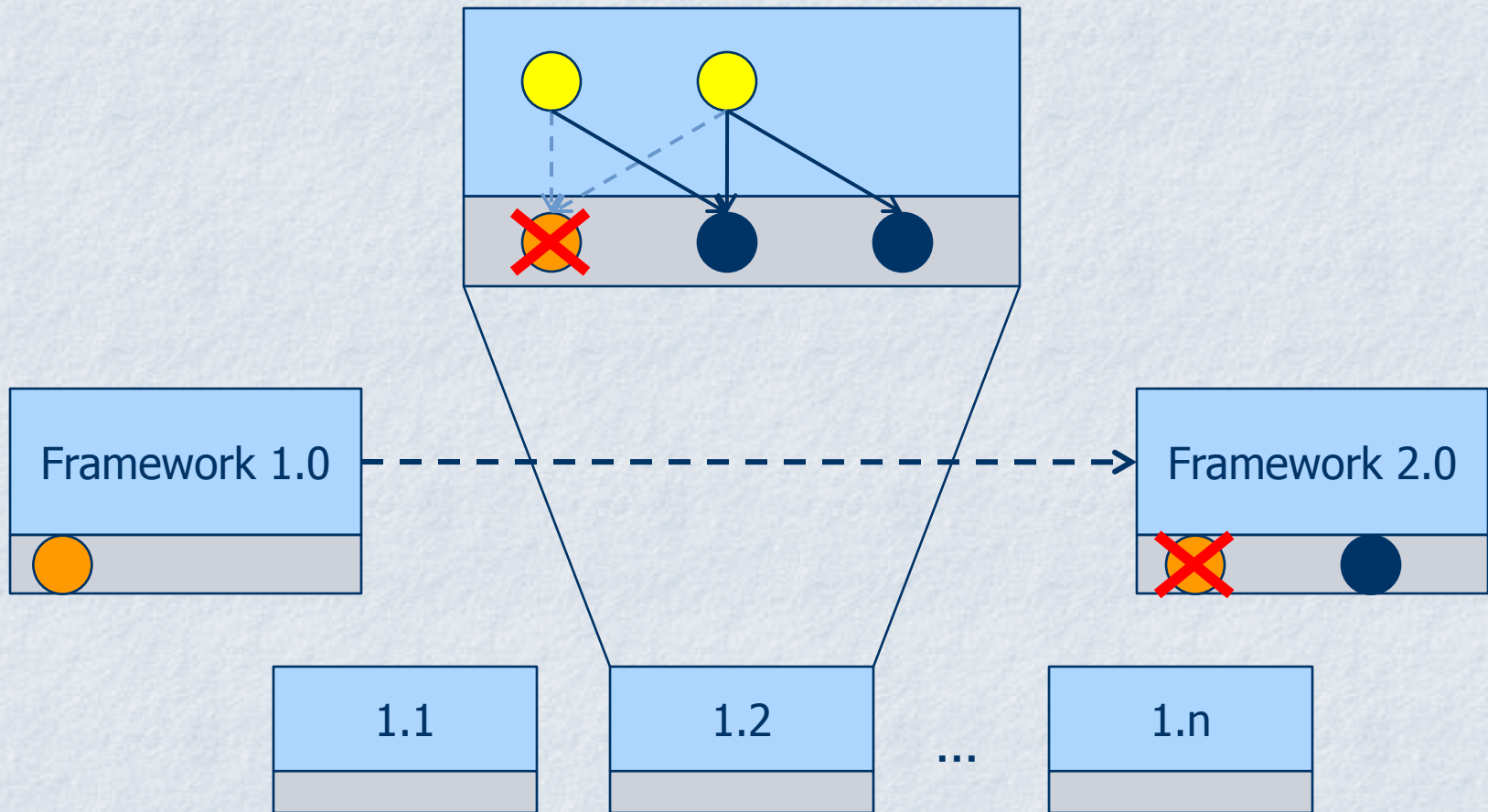
Hypothesis

Calls to broken methods are replaced **locally** in the **same change set**.



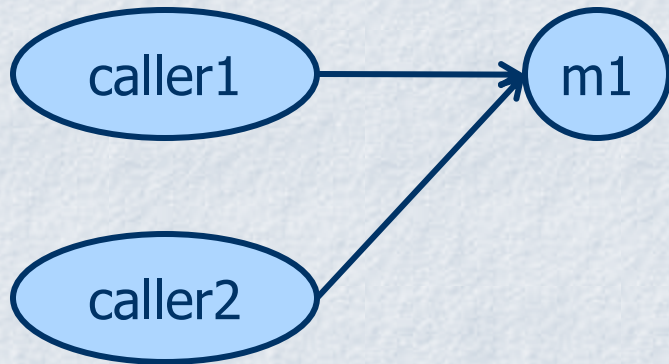
Hypothesis

Calls to broken methods are replaced **locally** in the **same change set**.

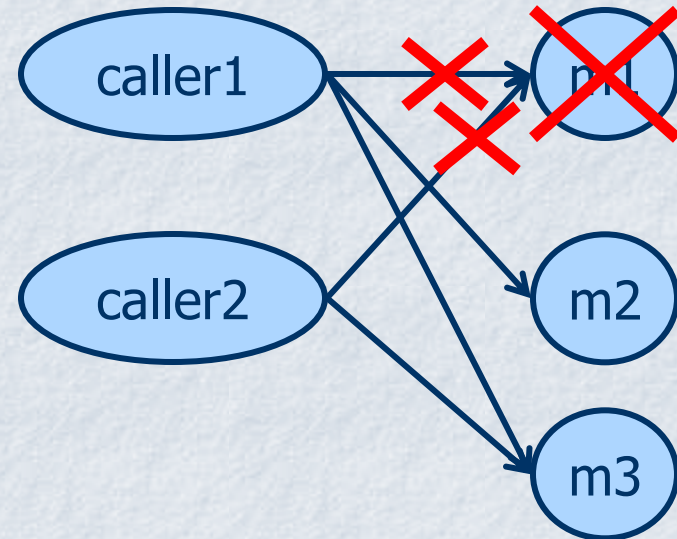


Method Calls Evolution

SemDiff Request : m1



version 1.1



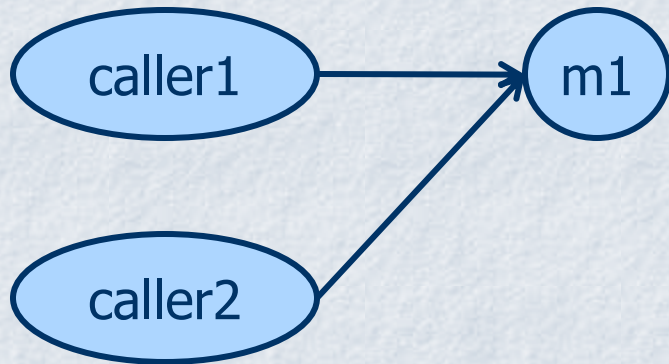
version 1.2

Recommendations:

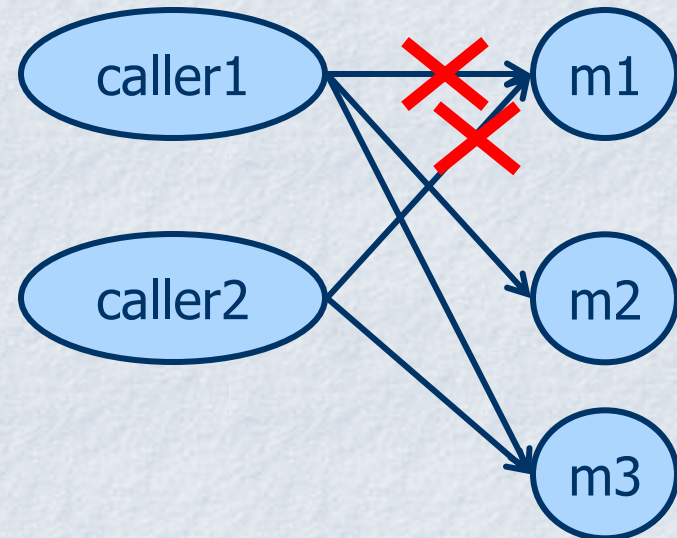
- 1) m3
- 2) m2

Method Calls Evolution

SemDiff Request : m1



version 1.1

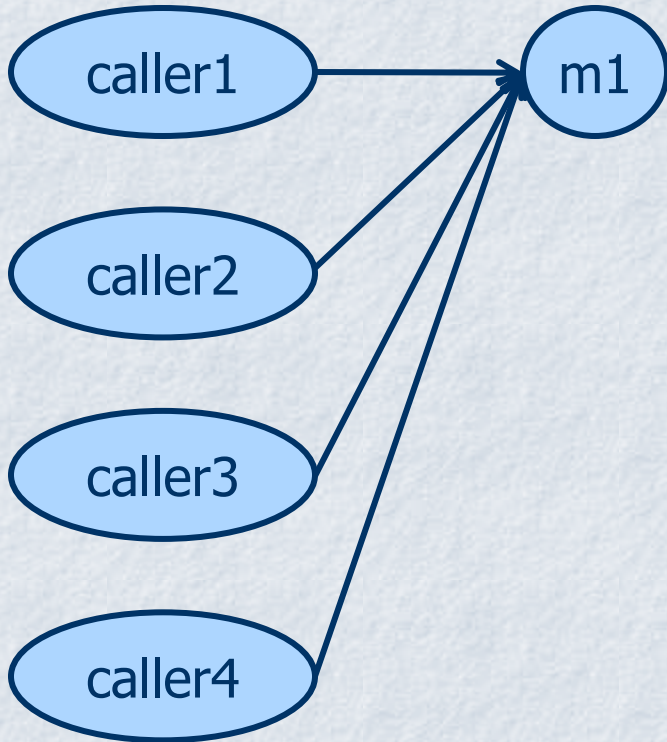


version 1.2

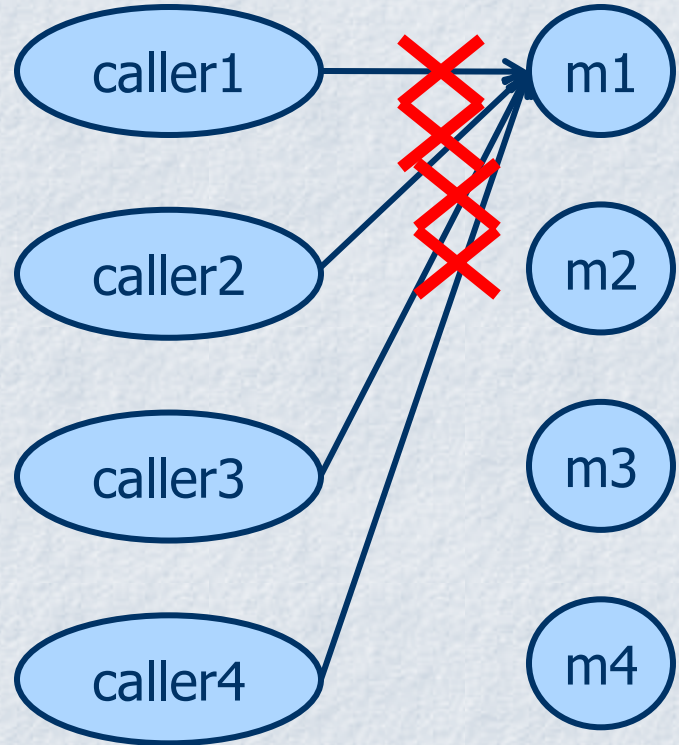
Recommendations:

- 1) m3
- 2) m2

Confidence Value



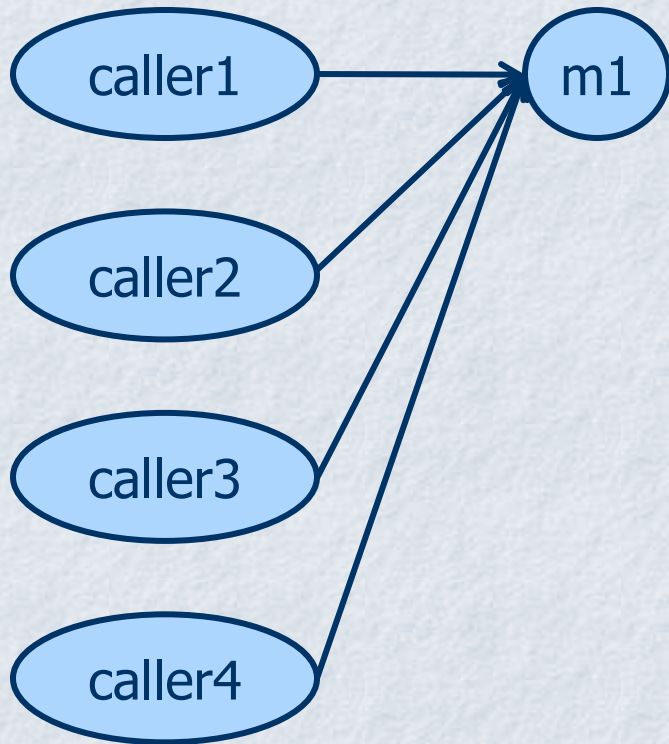
version 1.1



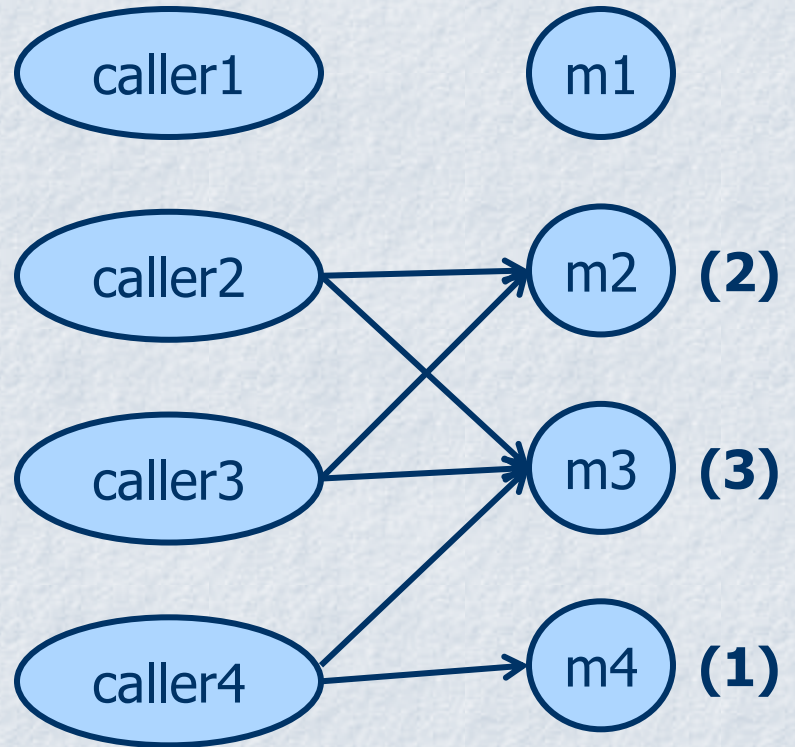
version 1.2

Recommendations:
m3: 100% (3/3)
m2: 66% (2/3)
m4: 33% (1/3)

Confidence Value



version 1.1



version 1.2

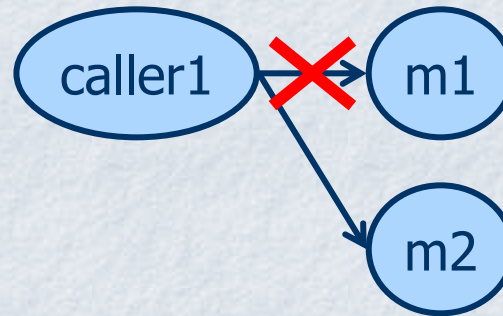
Recommendations:
m3: 100% (3/3)
m2: 66% (2/3)
m4: 33% (1/3)

Change Chains

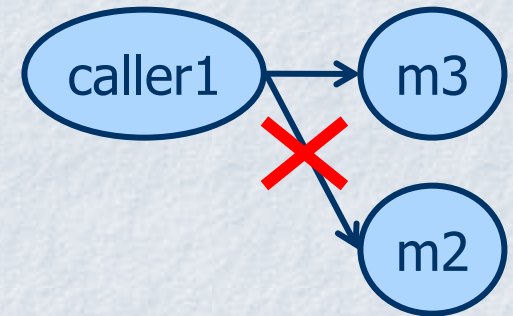
SemDiff Request : m1



version 1.1



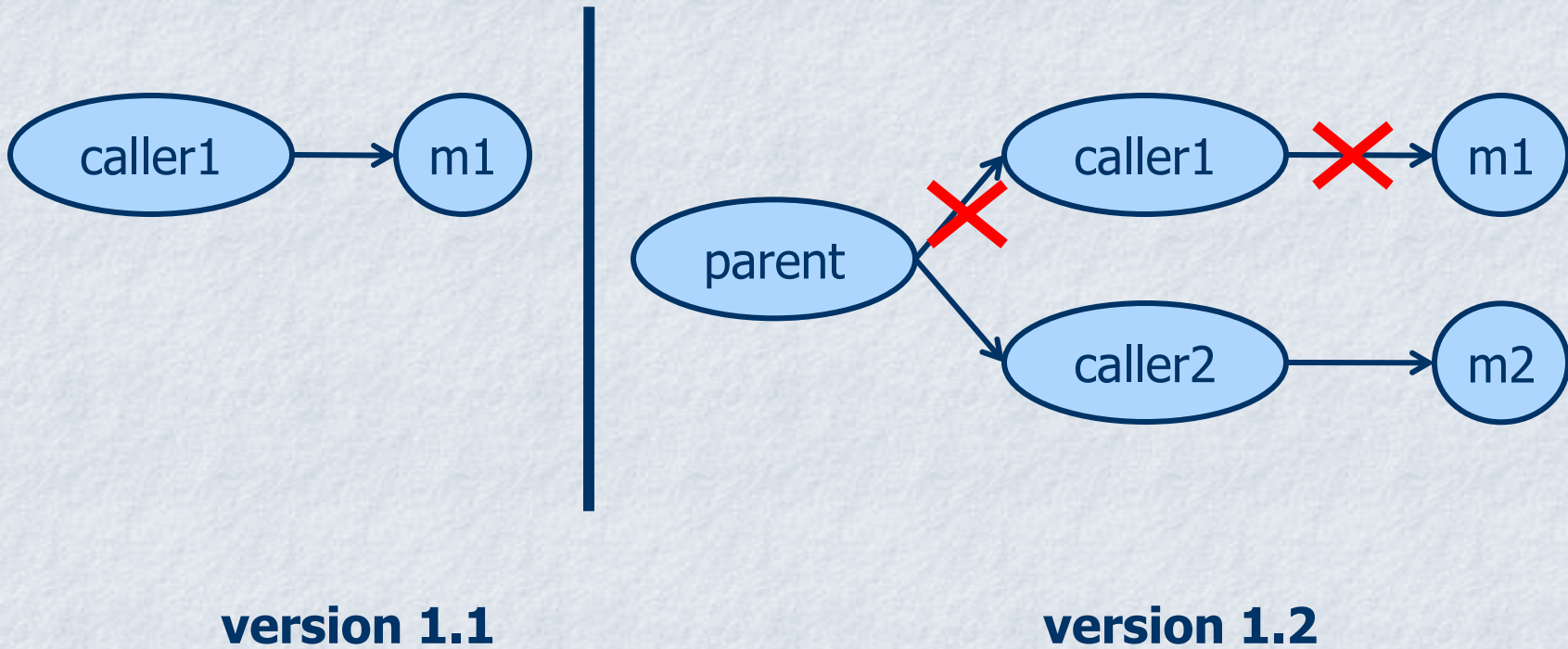
version 1.2



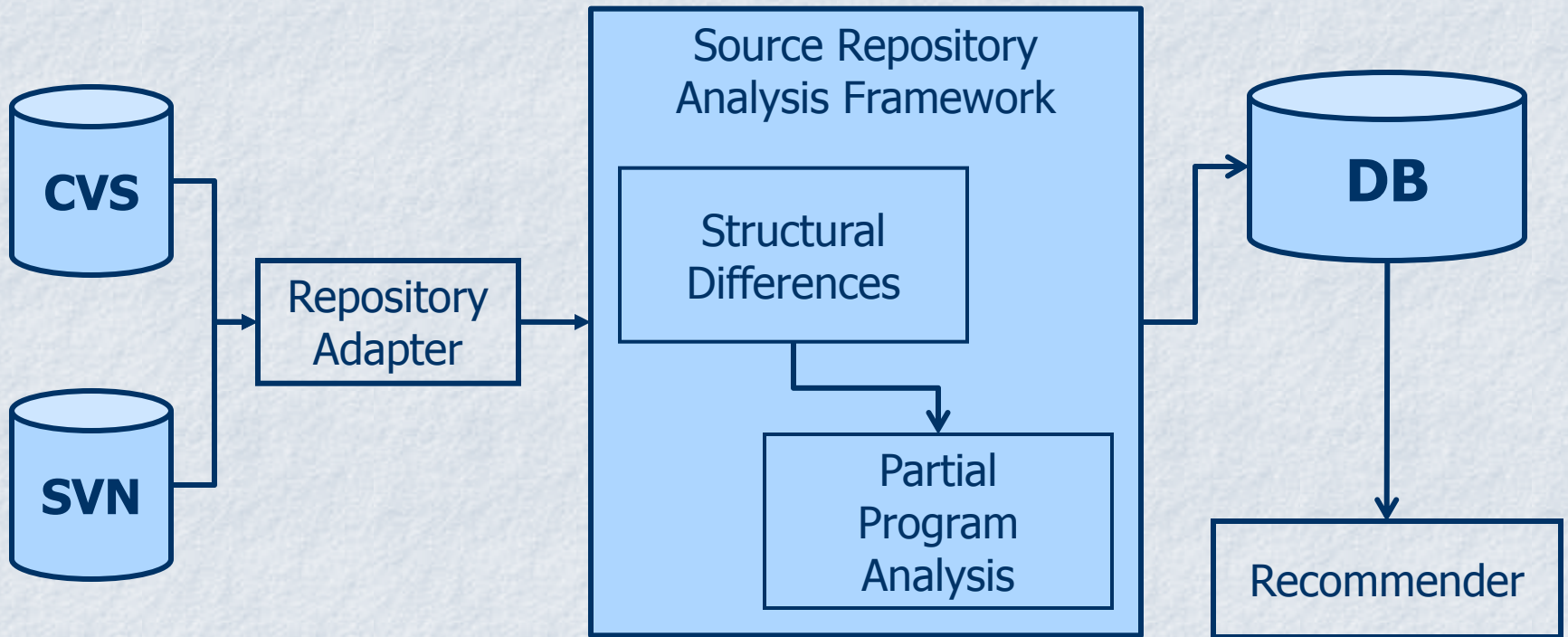
version 1.3

Caller Stability

SemDiff Request : m1

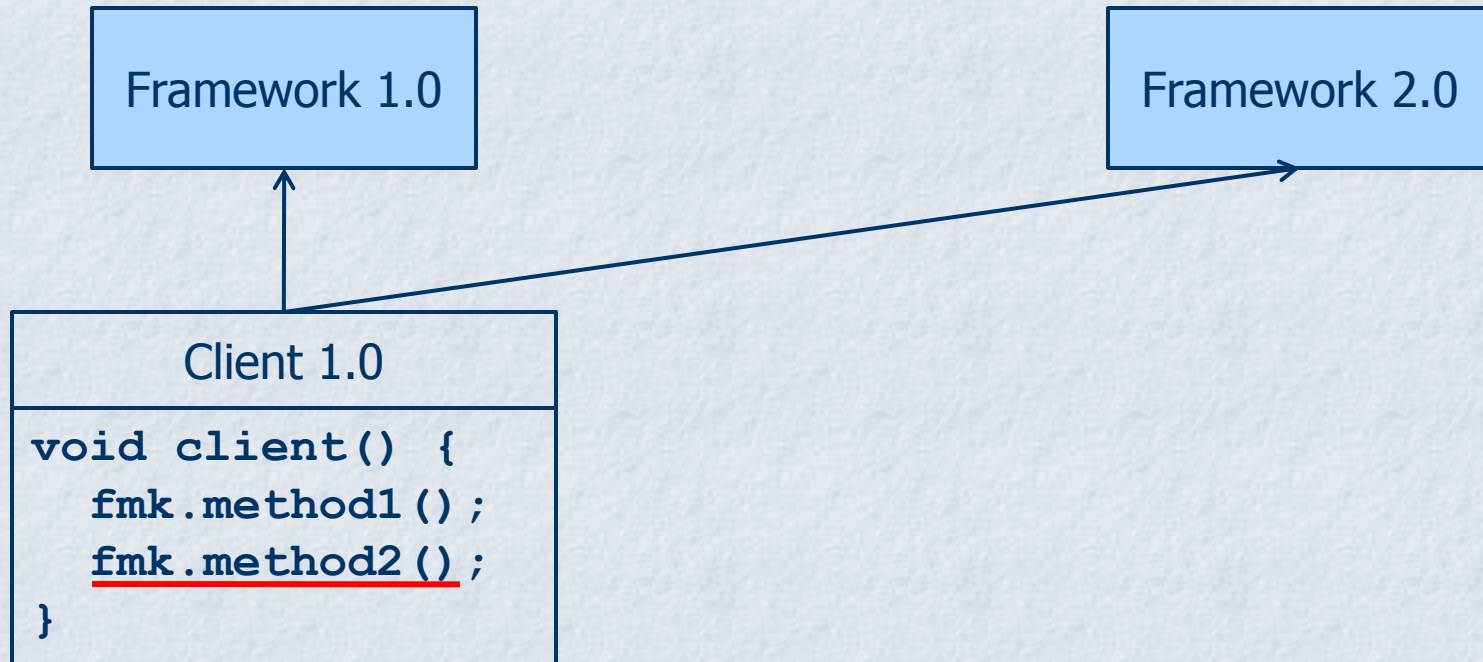


SemDiff Architecture



Data flow

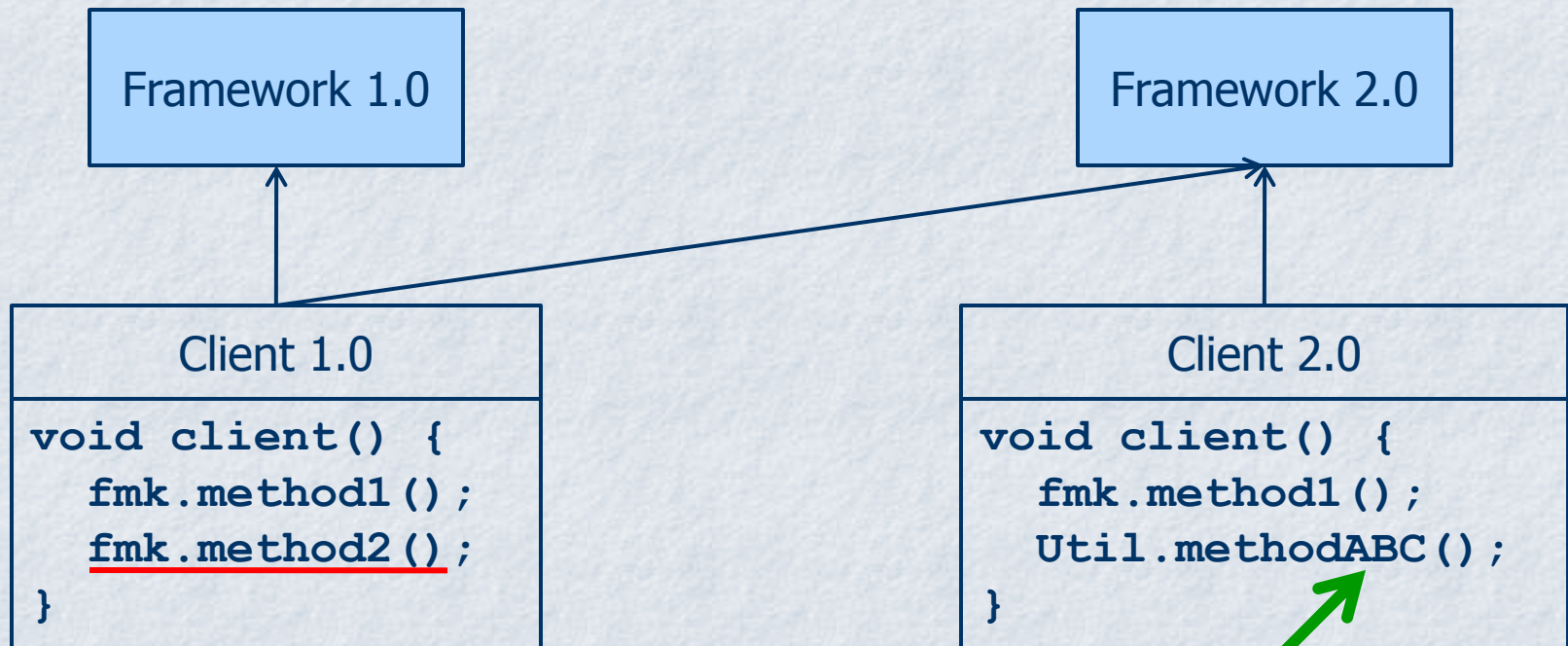
Evaluation Strategy



SemDiff

Query:	fmk.method2 ()	
Response:	fmk.method2a ()	100%
	Util.methodABC ()	75%
	fmk.methodXYZ ()	50%
	...	

Evaluation Strategy

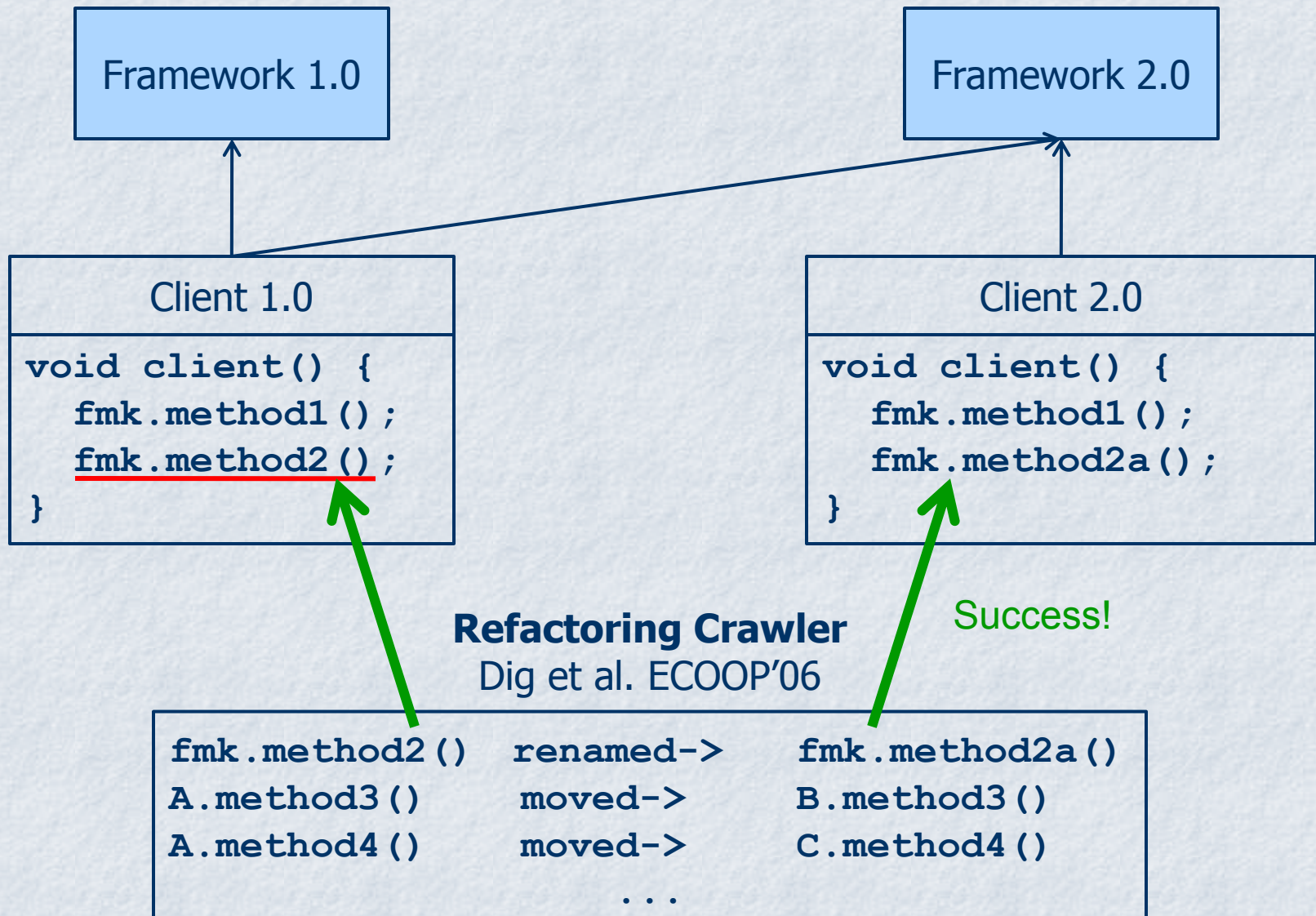


SemDiff

Query:	<code>fmk.method2()</code>	
Response:	<code>fmk.method2a()</code>	100%
	<code>Util.methodABC()</code>	75%
	<code>fmk.methodXYZ()</code>	50%
	...	

Success!

Comparing with Refactoring Crawler



Evaluation

Client	Eclipse JDT 3.1	Eclipse JDT 3.3
Mylyn	0.5	2.0
JBoss IDE	1.1	1.5
jdt.debug.ui	3.1	3.3

Eclipse JDT 3.1  Eclipse JDT 3.3
January 2005 10127 Transactions CVS July 2007

Results

Client	Errors	Scope	SemDiff	RC
Mylyn	13	8	8	0
Jboss IDE	21	15	15	0
jdt.debug.ui	28	14	10	6
Total	62	37	33	6

Results

Client	Errors	Scope	SemDiff	RC
Mylyn	13	8	8	0
Jboss IDE	21	15	15	0
jdt.debug.ui	28	14	10	6
Total	62	37	33	6

Precision

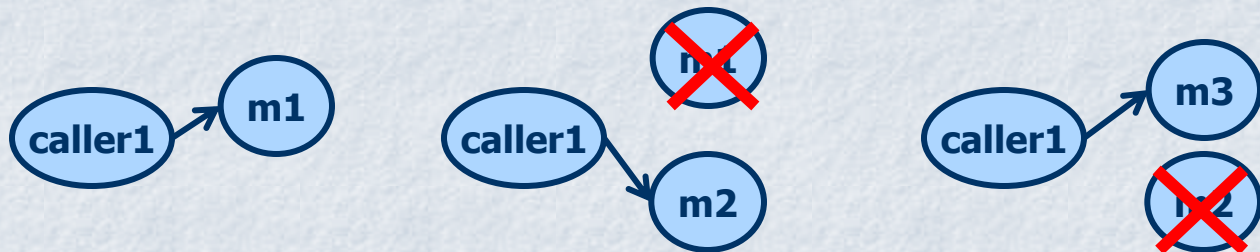
Top 3: **89%**

Top 1: **81%**

frameworkV31.myMethod() → frameworkV33.myMethod2()

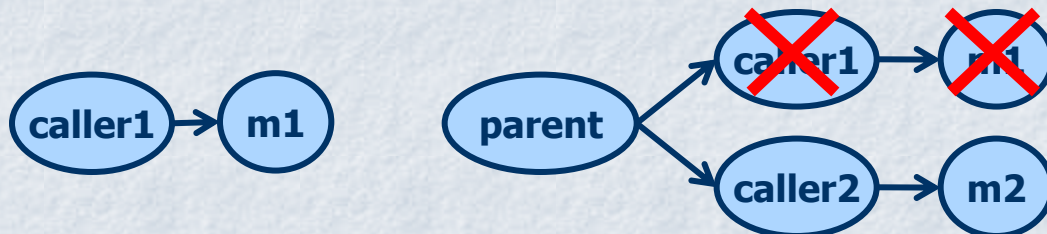
Change

Chains: **2**



Caller

Stability: **2**



Results

Client	Errors	Scope	SemDiff	RC
Mylyn	13	8	8	0
Jboss IDE	21	15	15	0
jdt.debug.ui	28	14	10	6
Total	62	37	33	6

of recommendations: **7.1**

100% Recommendations: **1.6**

method1()	– 100%
method2()	– 100%
method3()	– 75%
method4()	– 50%
method5()	– 25%
method6()	– 10%
method7()	– 10%

Related Work

Migration Path

e.g., Refactoring Scripts – Eclipse 3.3

Catchup! – Henkel et al., ICSE'05

Framework Usage

e.g., Strathcona – Holmes et al., FSE'05

PARSEWeb – Thummalapenta et al., ASE'07

Change Detection

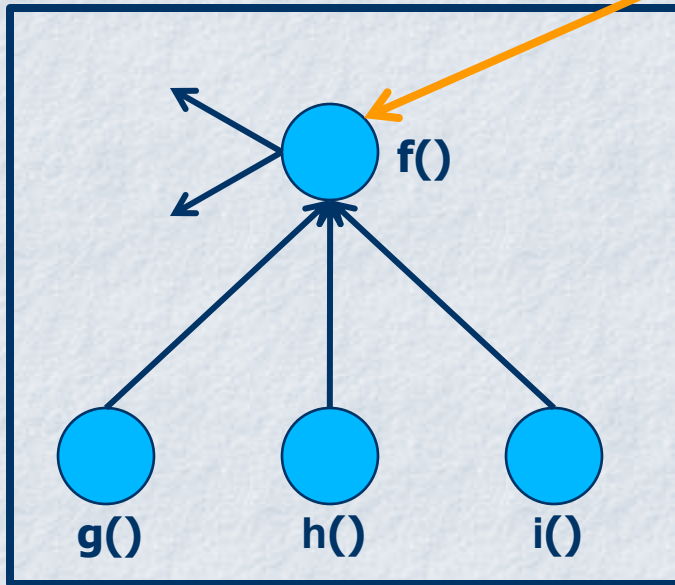
e.g., Origin Analysis – Zou et al., WCRE'03

Refactoring Crawler – Dig et al., ECOOP'06

Structural Changes – Kim et al., ICSE'07

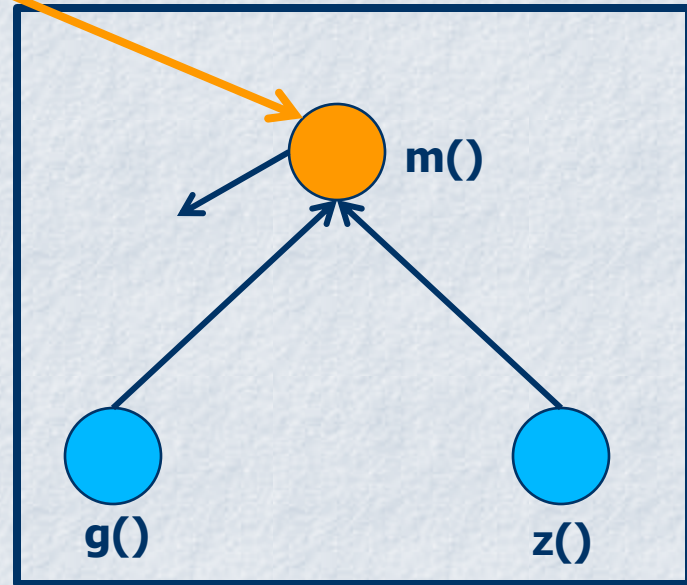
Change Detection - Limitations

$f() =? m()$



Framework

Version 1

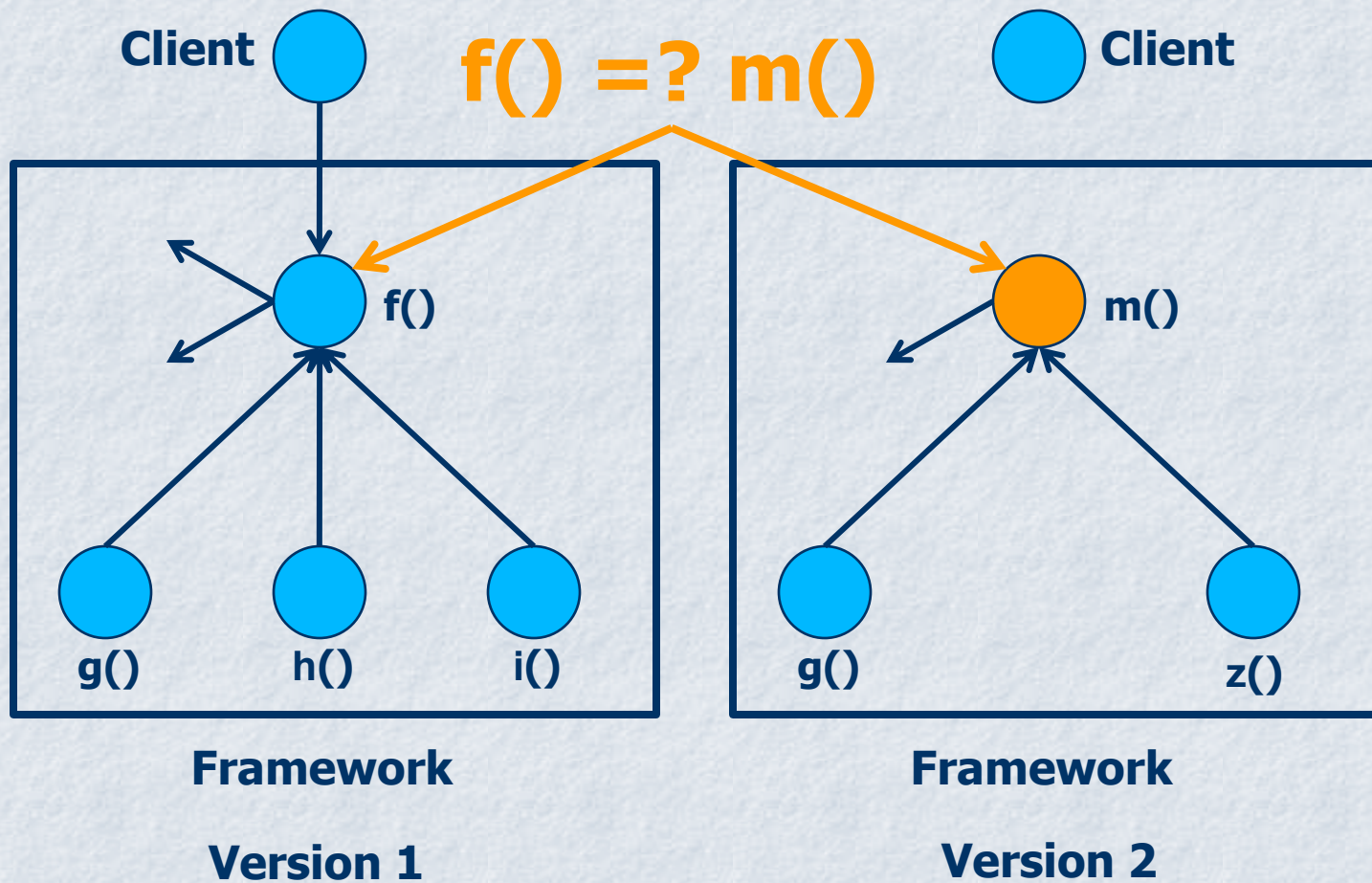


Framework

Version 2

Too many changes : refactoring **not detected!**

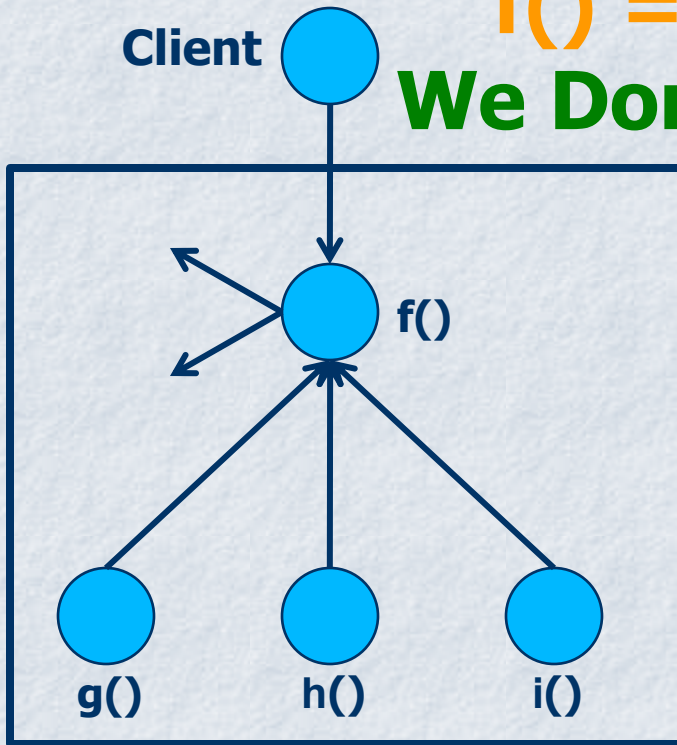
Change Detection - Limitations



Too many changes : refactoring **not detected!**

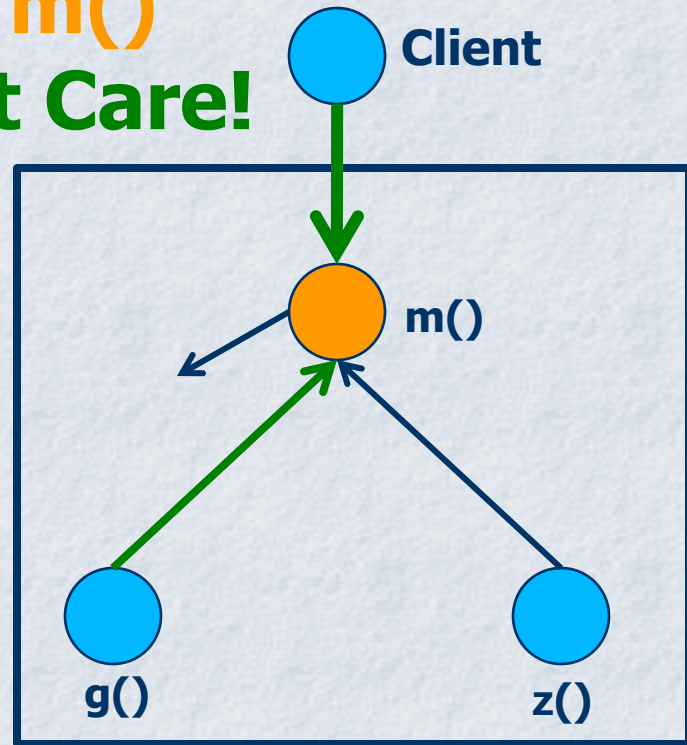
Using method calls

f() =? m()
We Don't Care!



Framework

Version 1

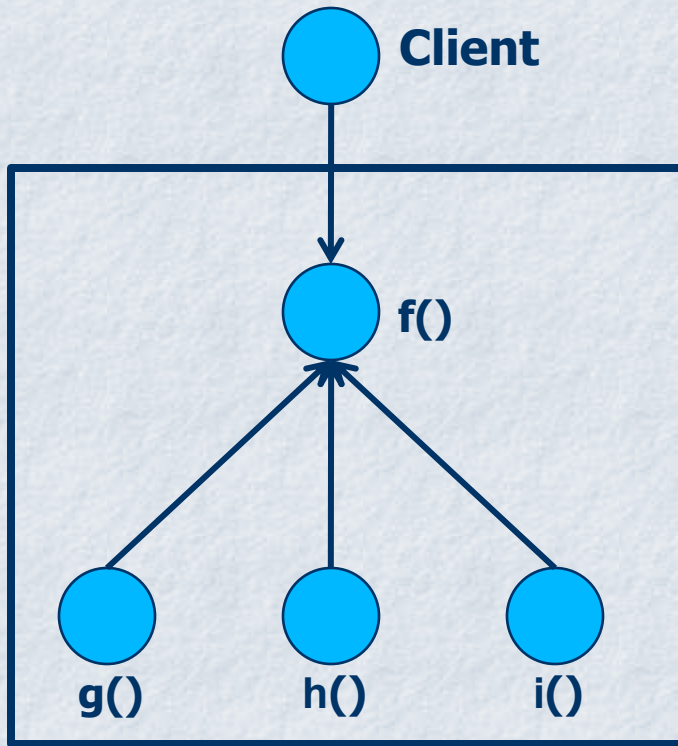


Framework

Version 2

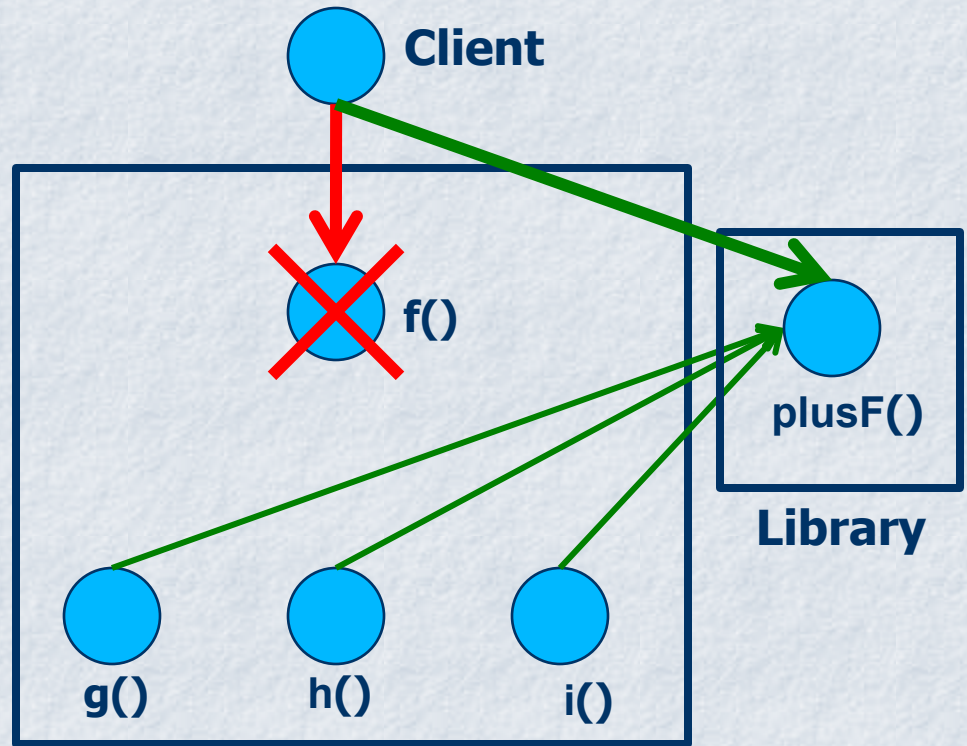
Callers not impacted by similarity

Importing Methods



Framework

Version 1



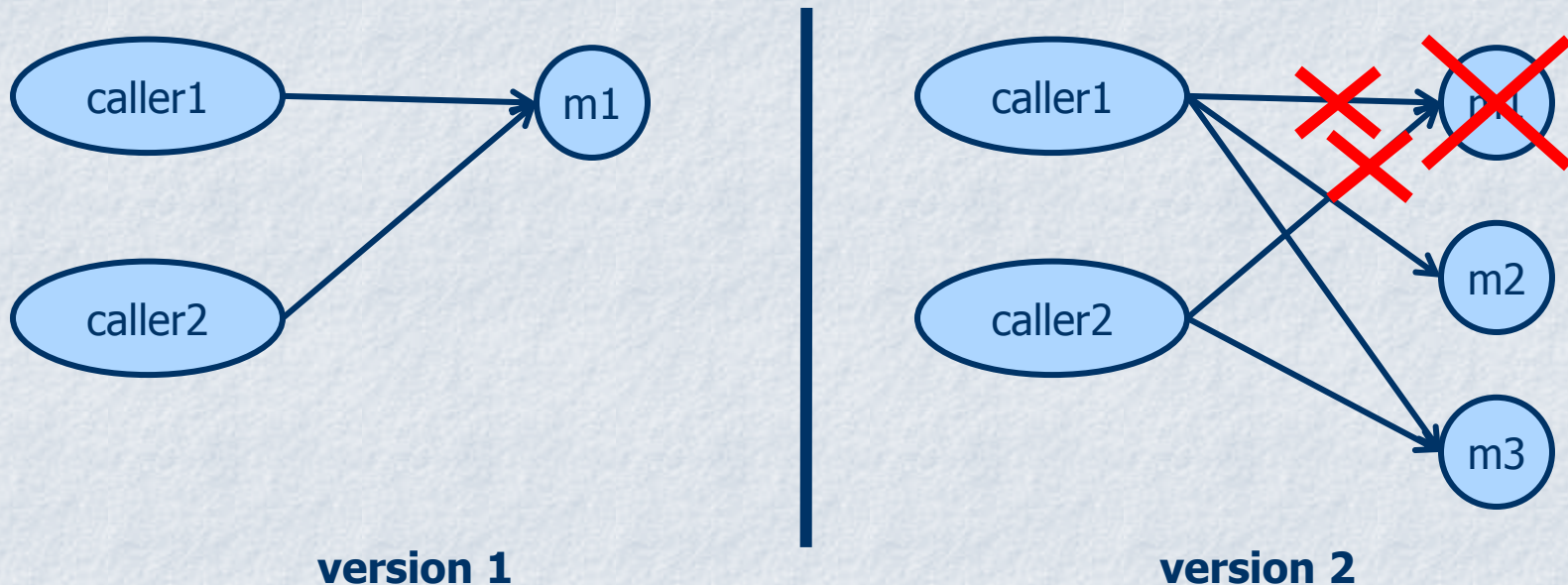
Framework

Version 2

Callers do not care about *similarity* nor *location*

Conclusion

SemDiff analyzes *method call changes* in *version histories* to recommend *adaptive changes* for framework evolution.



Recommendations

m3: 100%

m2: 50%