

# Supporting Software Integration Activities with Fine-grained Code Changes

Martín Dias

**Advisor:** Stéphane Ducasse

**Co-Advisor:** Damien Cassou

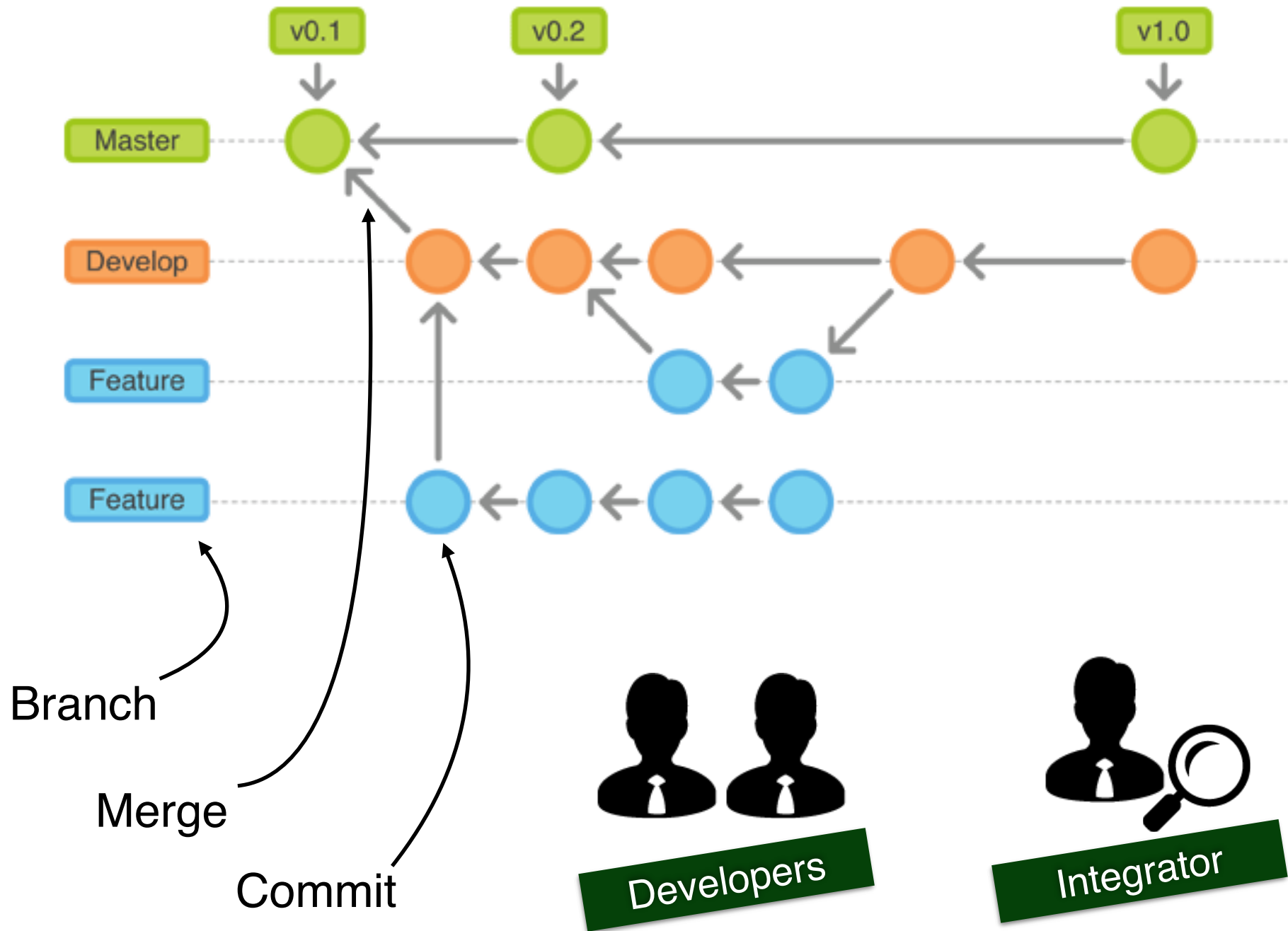
**Research team:** RMoD - Inria Lille

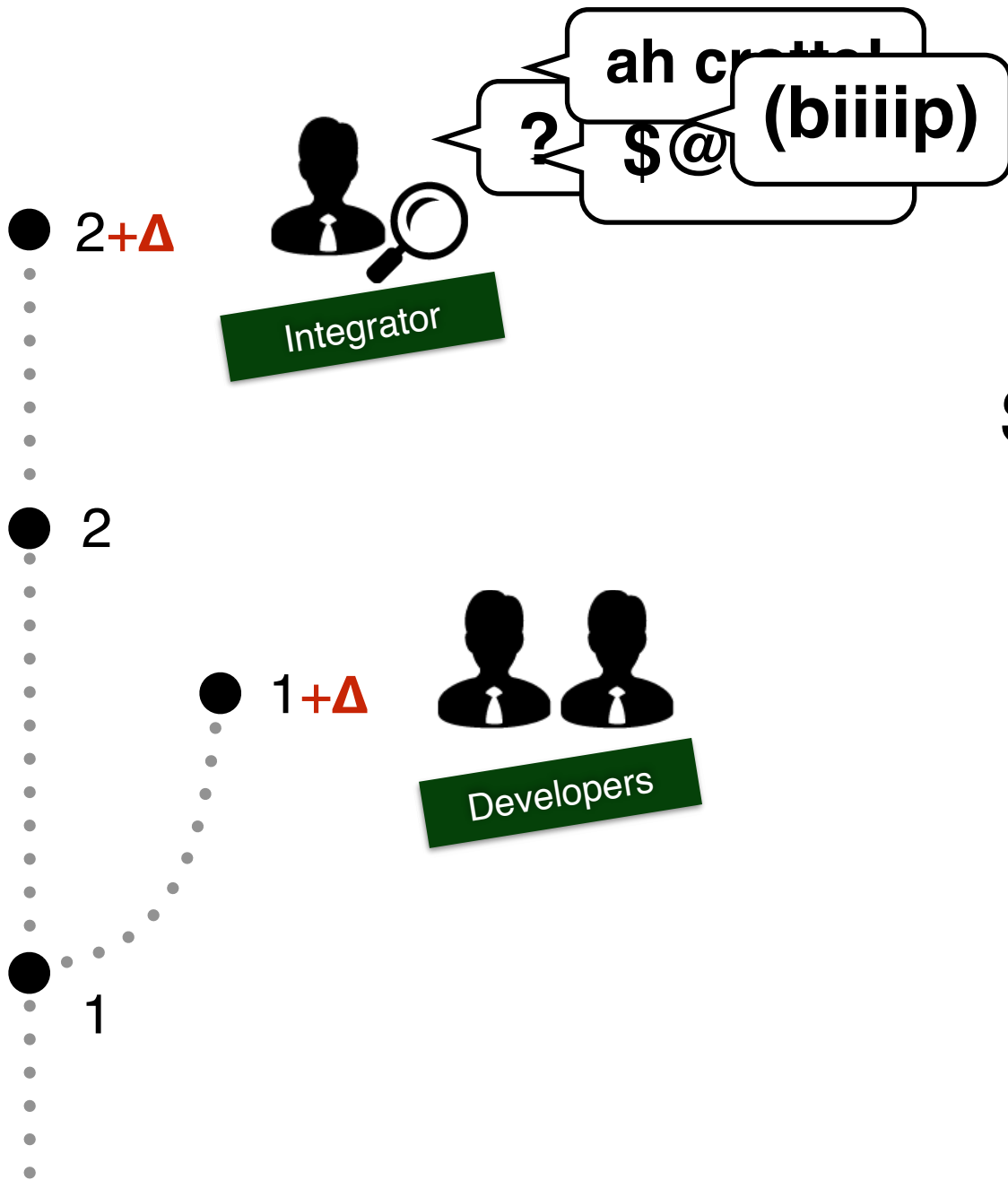
**Begin:** November 30th, 2012

**End:** November 29th, 2015

**Fundings:** INRIA doctoral grant

# Software Integration





Text-based  
Semantical Conflicts

Impacts

Cherry-picking

- Who is the owner of this changed code?

Authorship

- Which entities (e.g. classes, methods) have been changed?

Code structure

- What is the intention of this commit?

Change intention

- What bug fixes also affected the entities impacted by this change?

Bug tracking

- Does this commit depend on previous ones?

Change sequence

# Do Tools Support Code Integration? A Survey

**Martín Dias (1), Stéphane Ducasse (1), Damien Cassou (1),** Verónica Uquillas-Gómez (2)

**(1): RMoD Inria Lille–Nord Europe, University of Lille – CRIStAL, France**

**(2):** Norizzk.com, Belgium

(Under submission to Journal of Technology)

RQ

What questions do integrators ask?

- ➔ Open call in 3 development mailing-lists
- ➔ Literature survey

RQ

What is the **importance** and **tools support** of each question?

- ➔ Survey experts (42 integrators)

## Questionnaires INRIA

### SOFTWARE INTEGRATION SURVEY

0%  100%

#### Authorship/Ownership

These questions are related to the owner of the original code, and author of the commit.

Please rank each question below.

**(A1):** The word "importance" refers to the support to the integration task that the answer of that question provides.

**(A2):** Indicates the coverage of your tools for answering the question.

	(A1) What is the importance of this question?				(A2) Do your tools answer this question?			No answer
	Nothing	Little	Moderate	Extreme	No	Partially	Yes	
"Who is the author of this changed code?"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
"Who was the previous owner of the changed code?"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
"Has my own code been changed?"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
"What is the general quality of the change committer?"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
"How many people have contributed to this group of commits?"	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

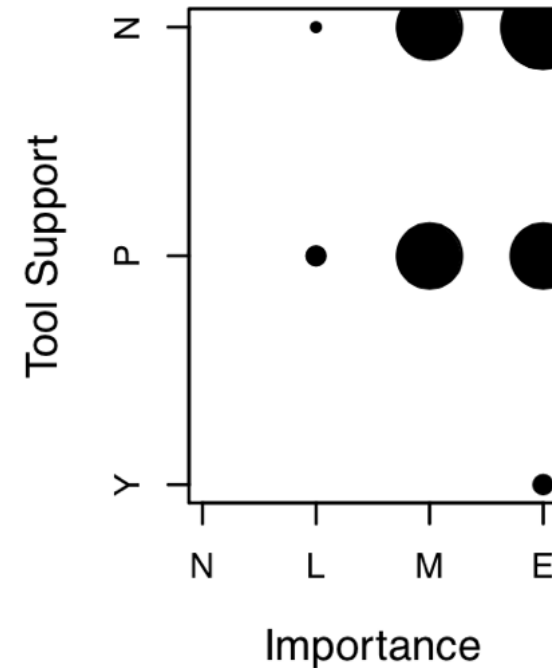
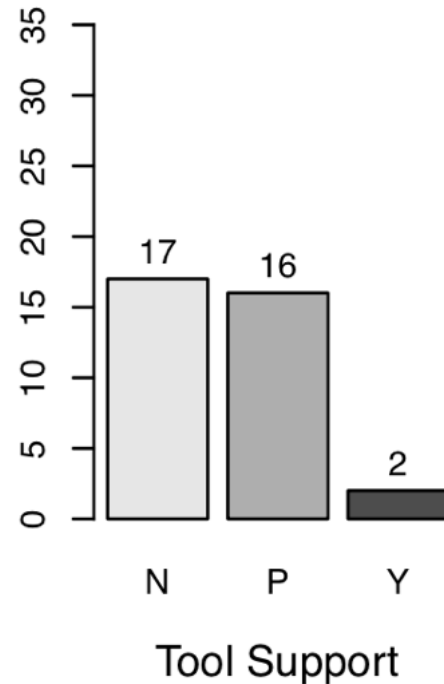
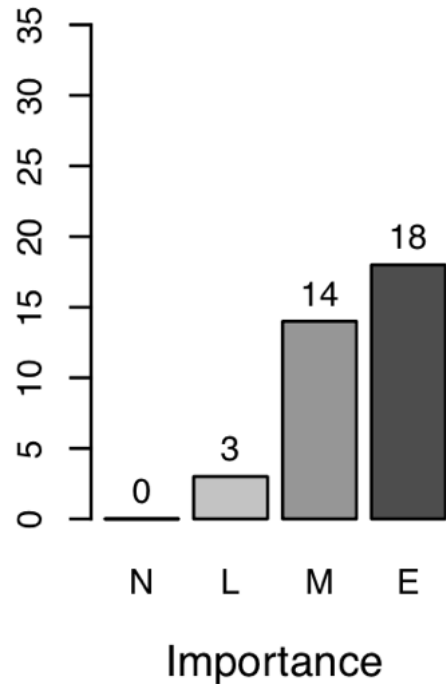
← Previous   Next →

Resume later

Exit and clear survey

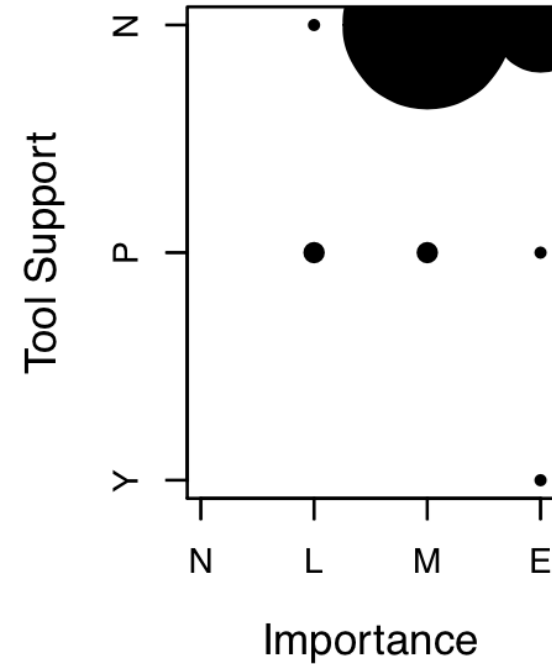
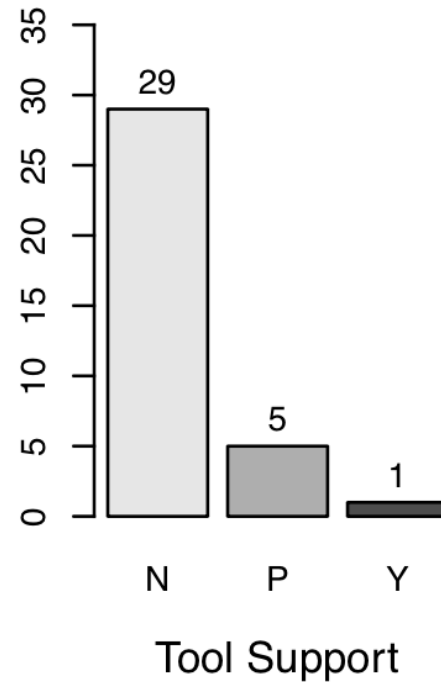
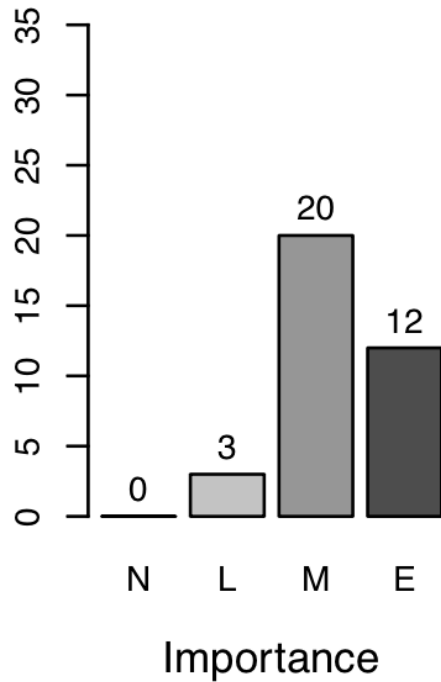


# Impact (ripple effects) ?



(Q25) If I apply the commit, what are the parts of my current system that it affect? What are the users (classes/methods/functions) potentially impacted by this change in the destination branch/fork?)

# Tangled changes?



(Q10) Do all the changes within the commit belong together? (Can we split the commit?)

# Most **important** questions without **tool support**

Understanding Change Impact

Understanding Change Dependencies  
when Cherrypicking

Understanding Change Scattering

# Untangling Fine-Grained Code Changes

**Martín Dias** (1), Alberto Bacchelli (2), Georgios Gousios (3), **Damien Cassou** (1), **Stéphane Ducasse** (1)

(1): **RMoD Inria Lille–Nord Europe, University of Lille — CRISAL, France**

(2): SORCERERS @ Software Engineering Research Group, Delft University of Technology, The Netherlands

(3): Digital Security Group, Radboud Universiteit Nijmegen, The Netherlands

(SANER'15)

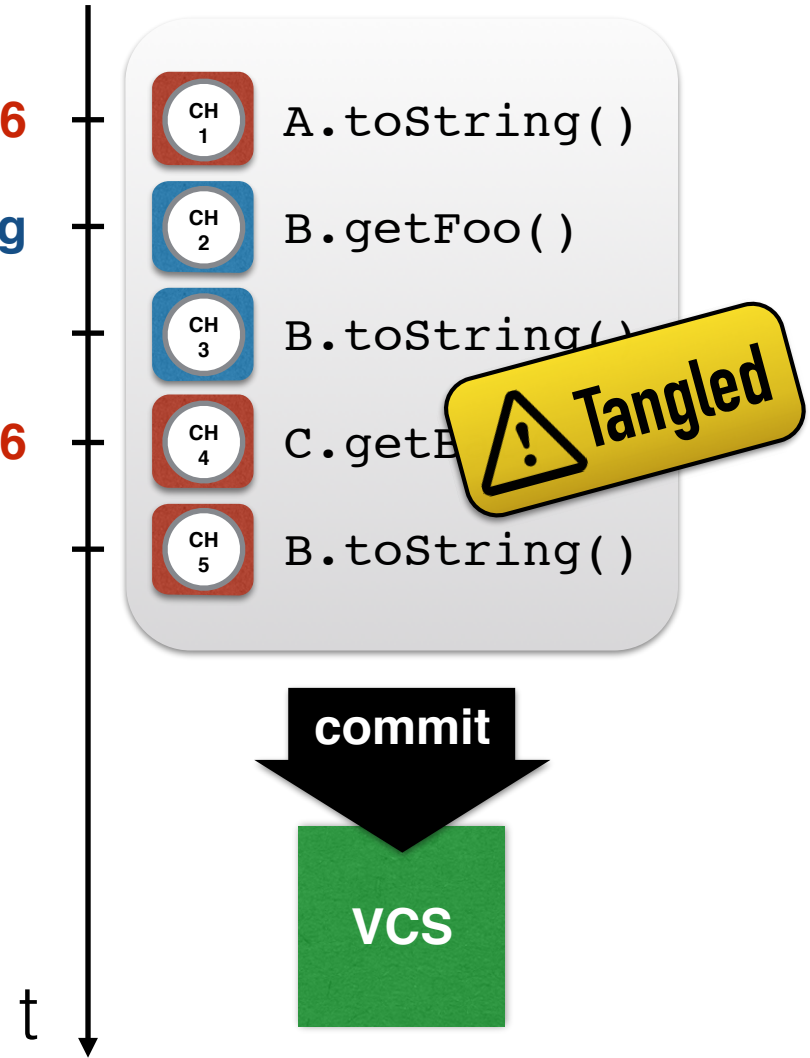
# Development



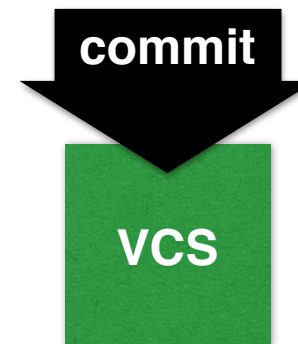
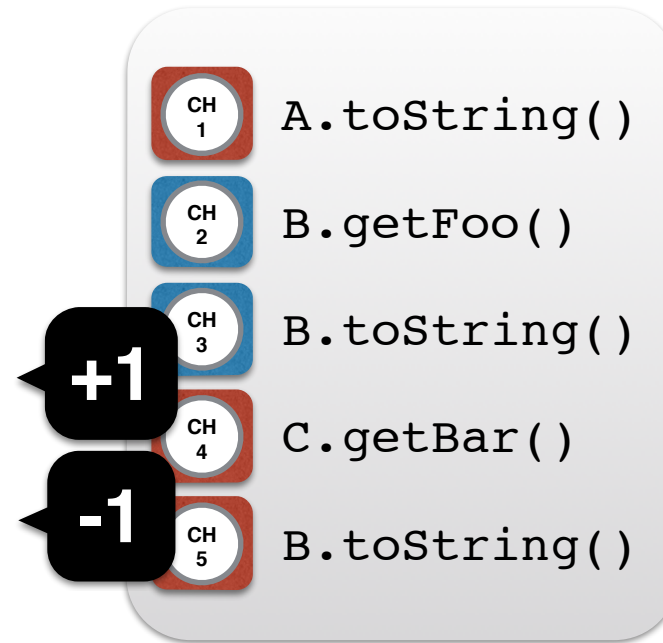
Feature #6

Fix Bug

Feature #6



# Integration



# Integration

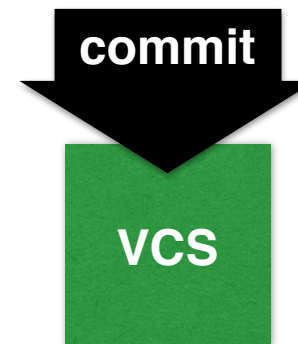


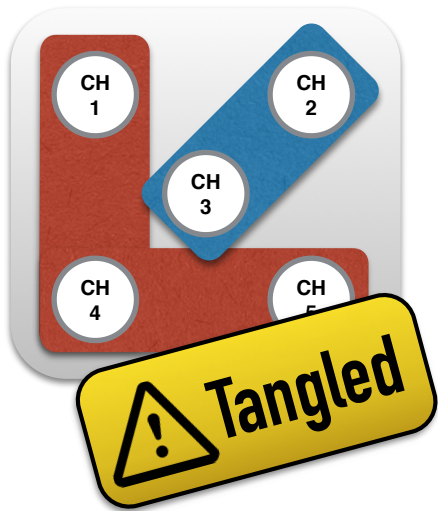
**+1**

- CH 2 B.getFoo()
- CH 3 B.toString()

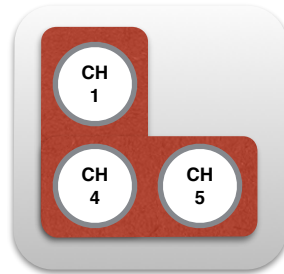
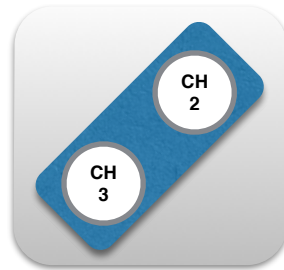
**-1**

- CH 1 A.toString()
- CH 4 C.getBar()
- CH 5 B.toString()





untangler  
tool



commit

commit





# Herzig and Zeller (MSR 2013)

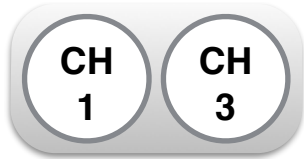
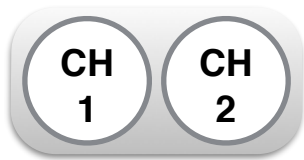
**VCS** repositories of 6 **Java** projects

Tangled commits: **20%**

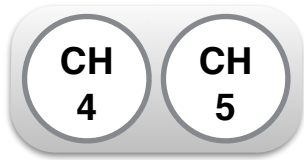
**Untangling algorithm** using **features** of code changes

Features of code changes?

Pair



...



Call Graph  
Distance



...



File  
Distance



...



...

...

...

...

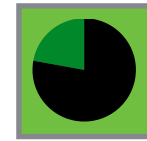
...



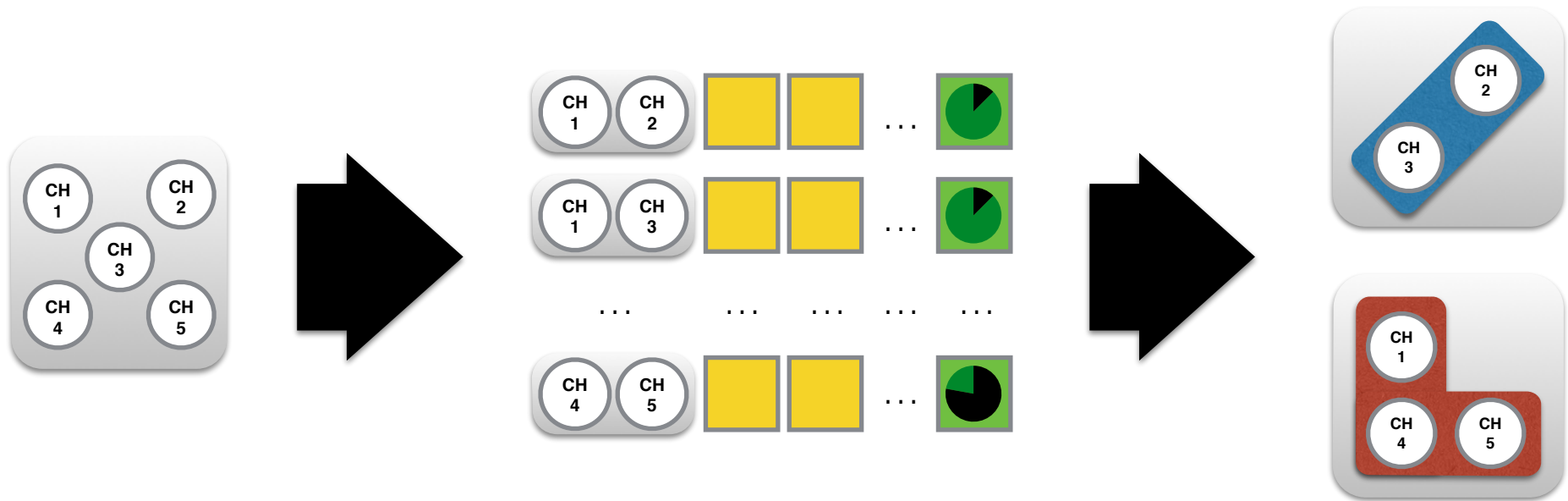
Linear  
Regression



...



# Herzig and Zeller (MSR 2013)

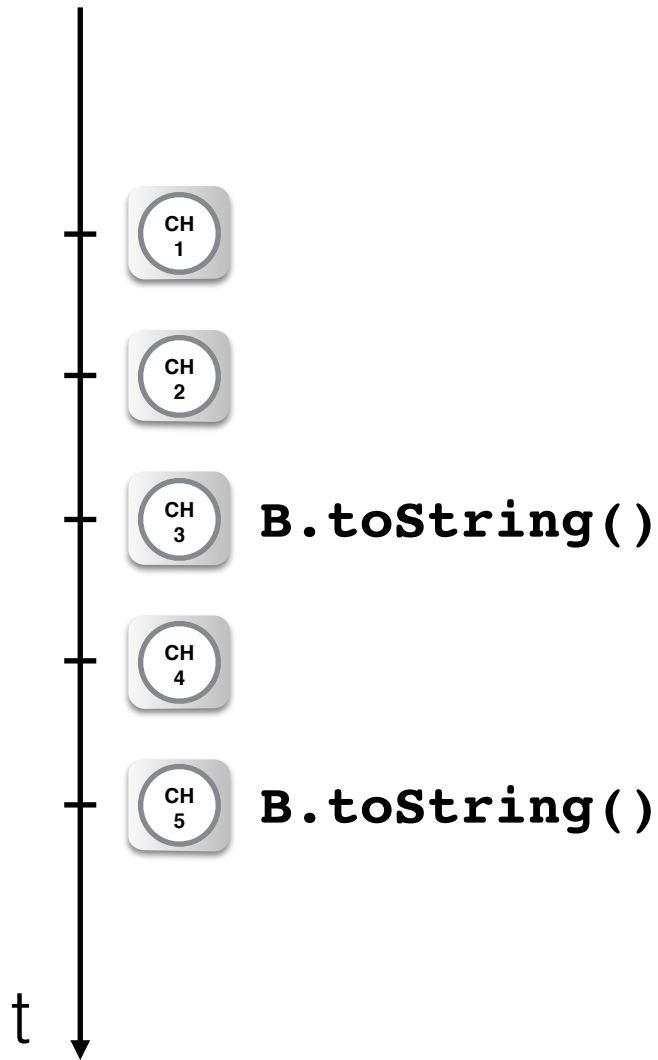


# Limitations

**dynamically**-typed languages

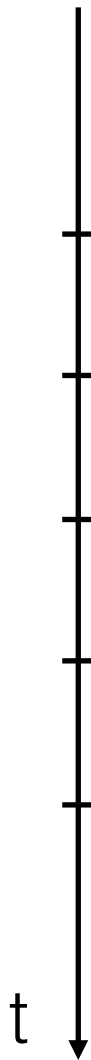


**light** static analysis









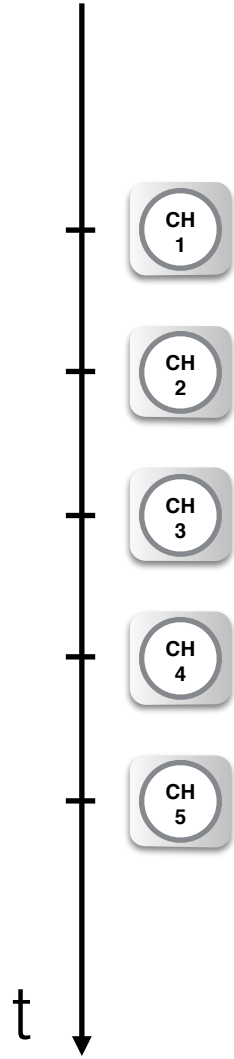
`B.toString()`

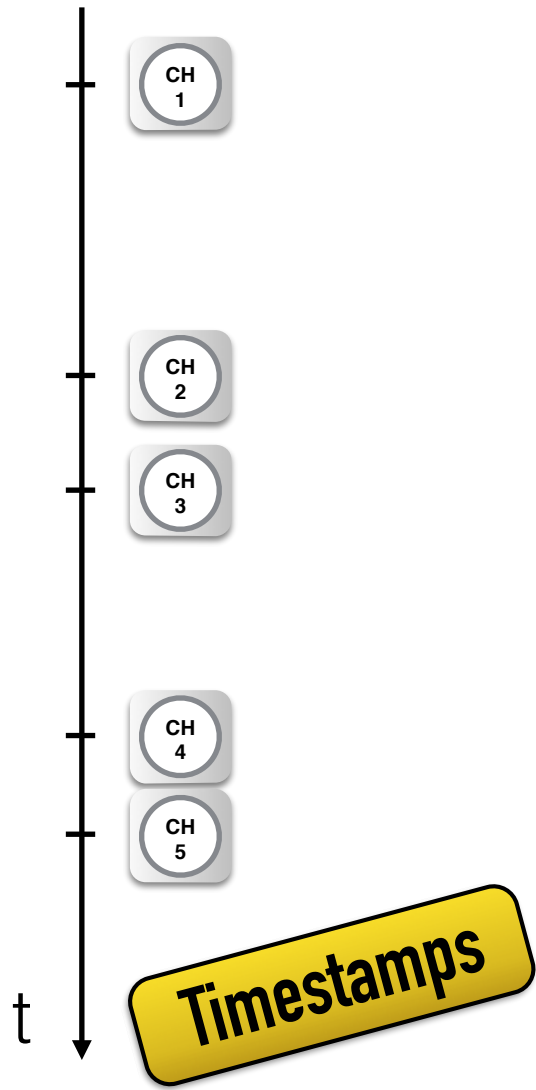
`B.toString()`

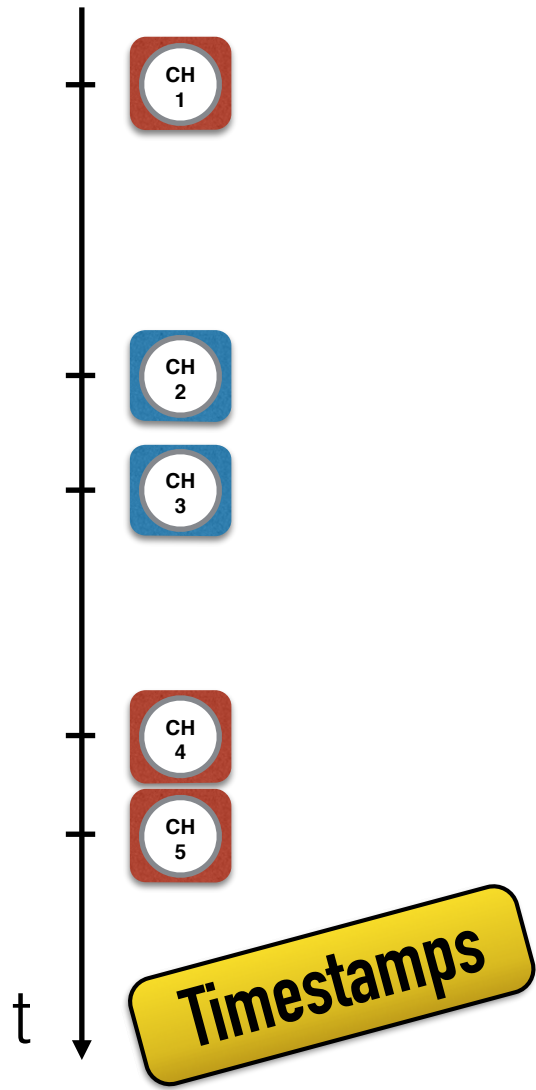


“We found that **37%** of code changes are **shadowed** by other changes, and are not stored in VCS.”

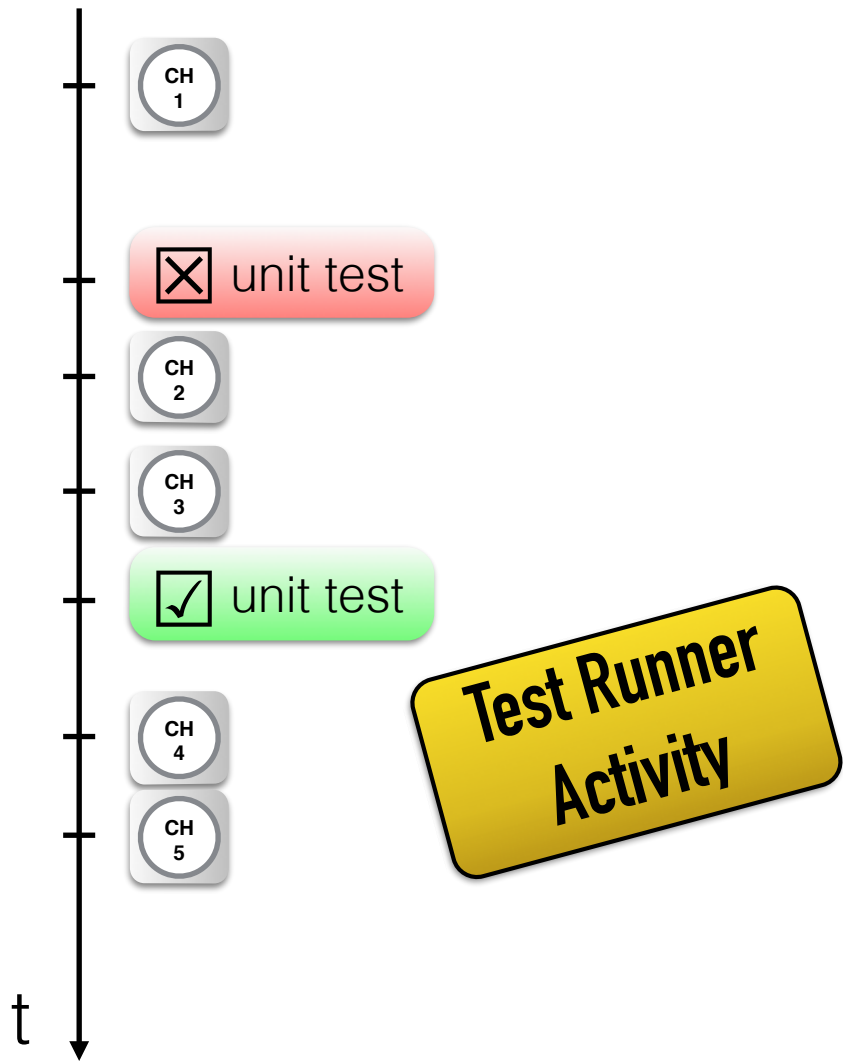
Negara et al. (ECOOP'12)

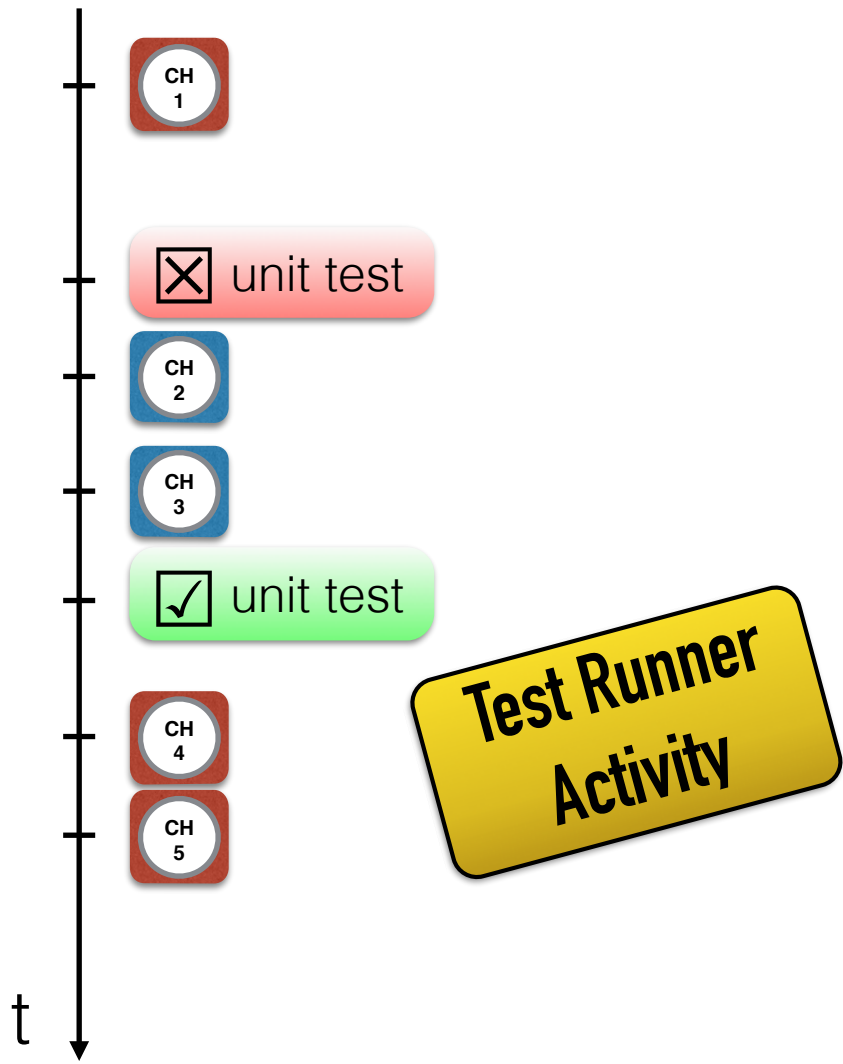






VCSs don't have this information



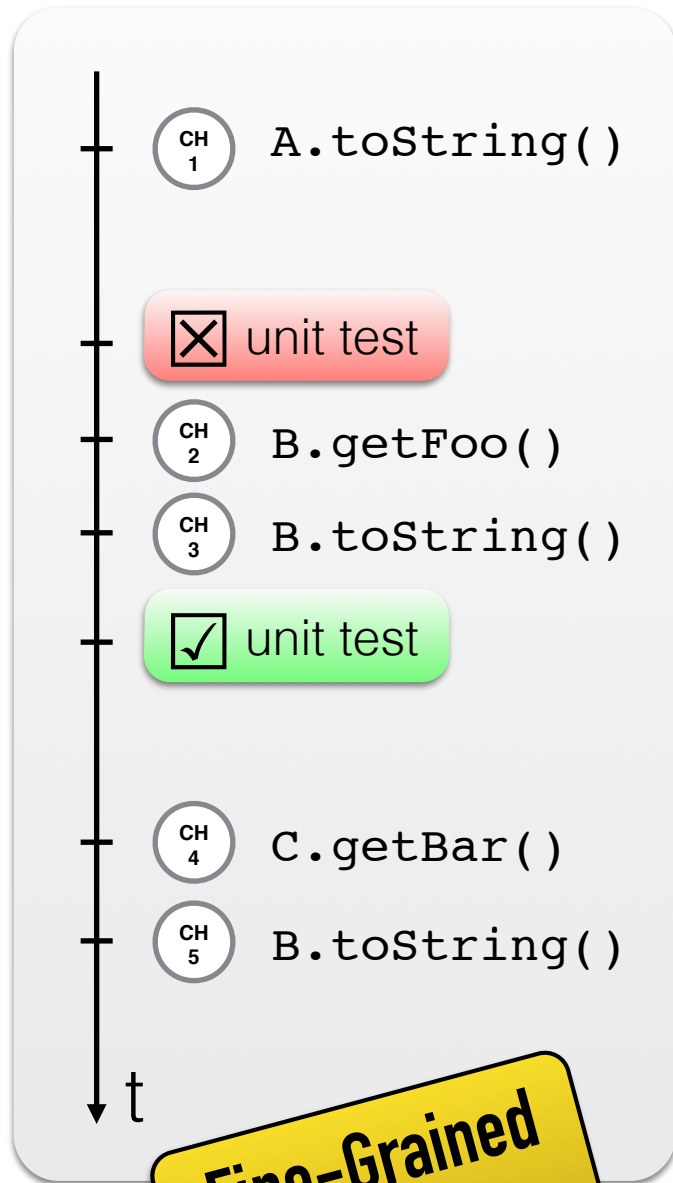




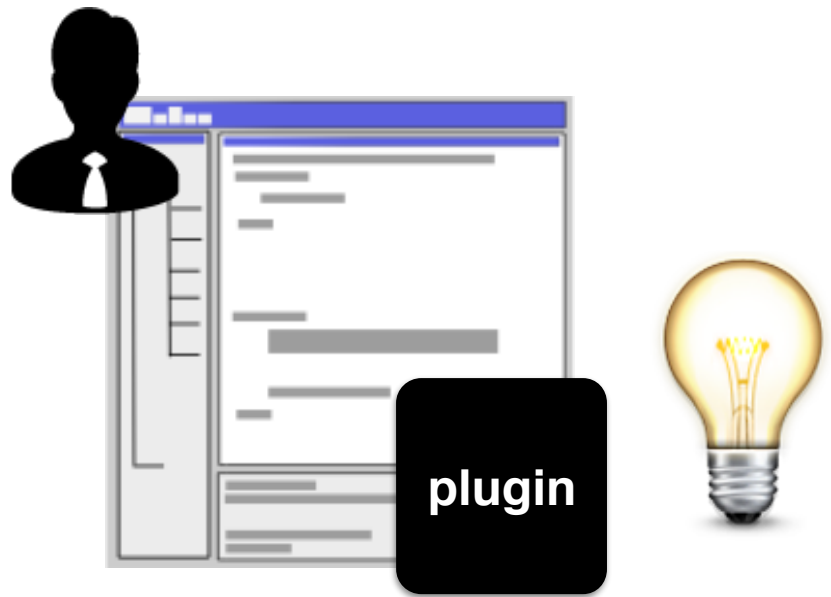
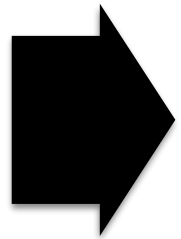
Overcoming such limitations

# Epicea

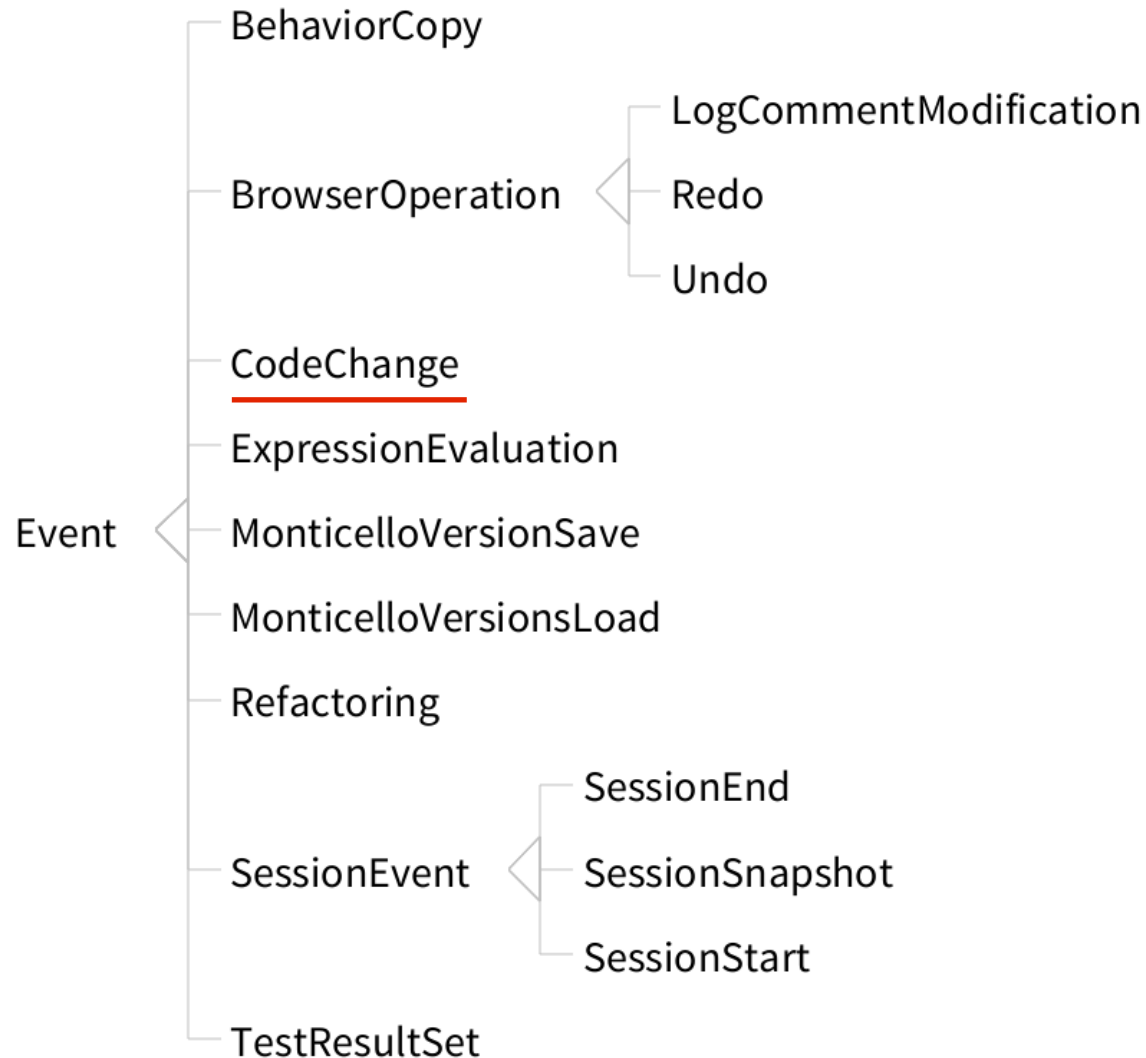
fine-grained  
**code changes & IDE events** logging



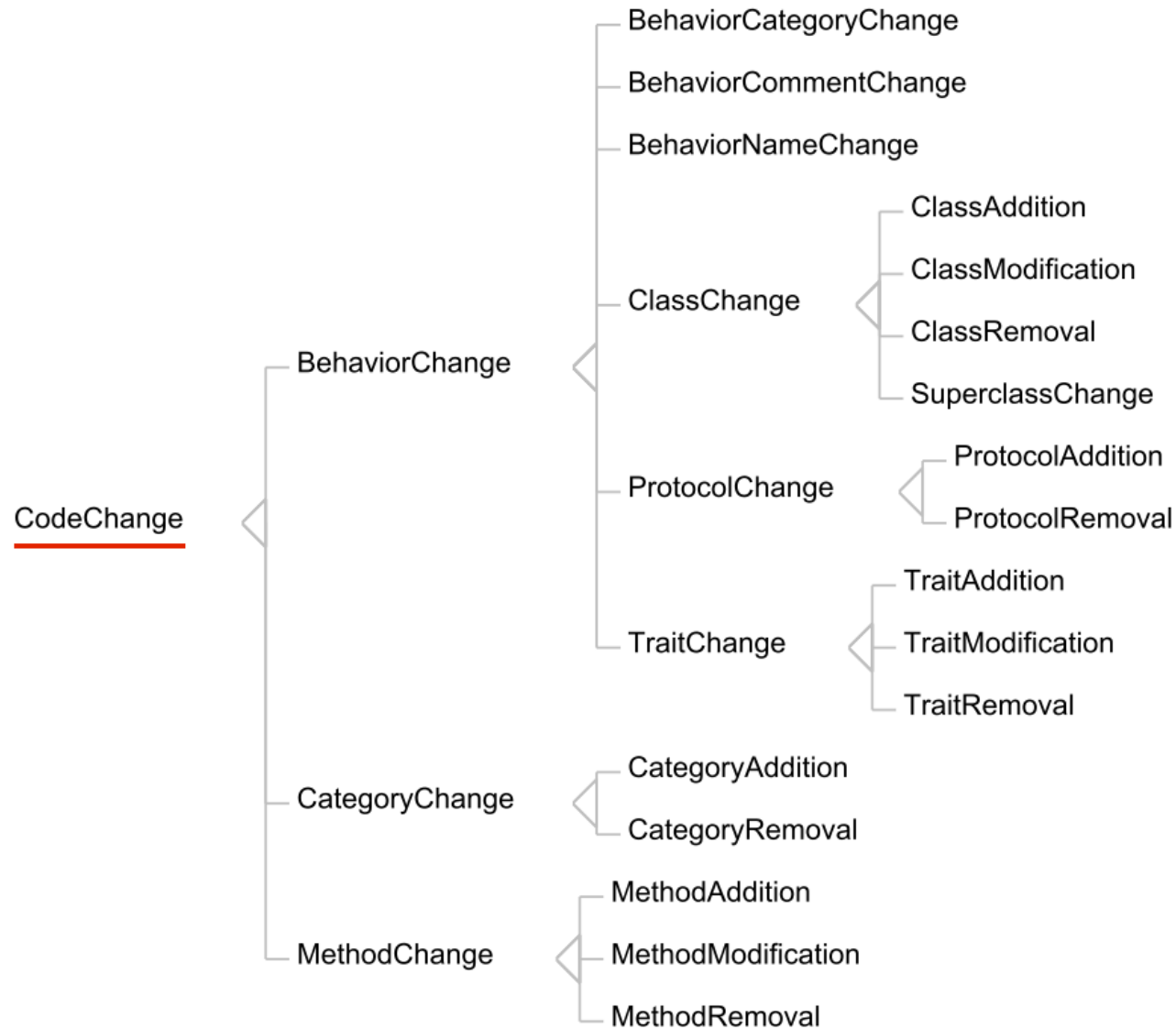
**Fine-Grained  
Code Changes**



# Epicea Model: Events



# Epicea Model: Code Changes



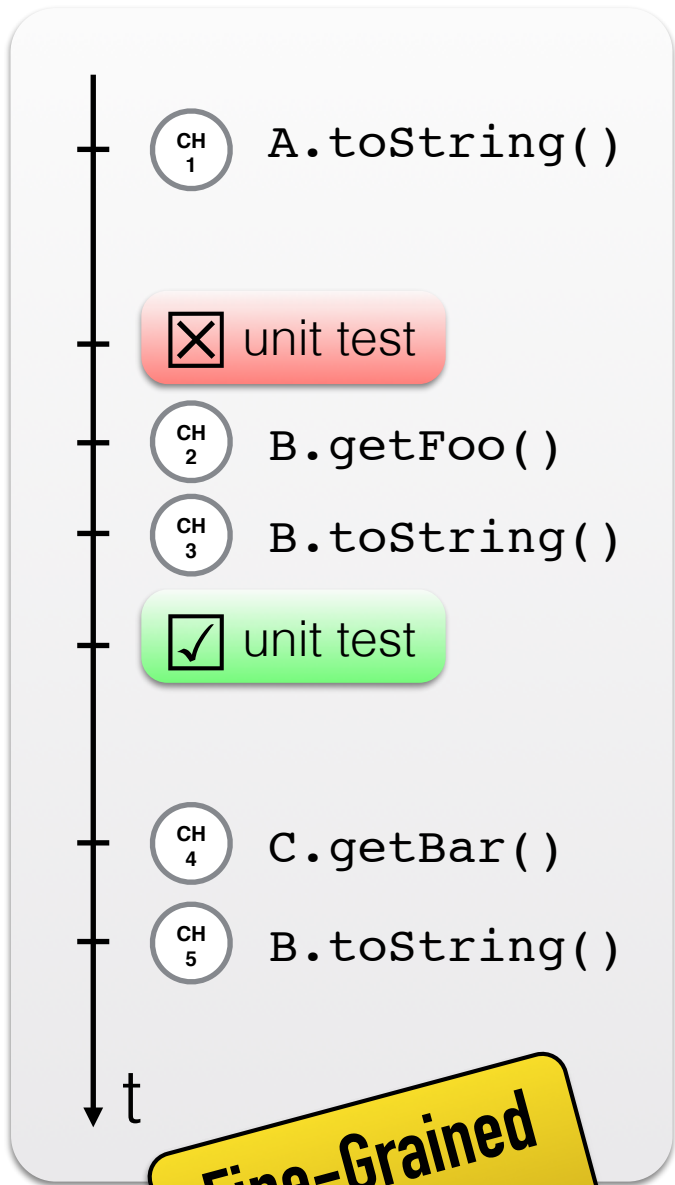
# Epicea Log Browser

The screenshot shows the Epicea Log Browser application window. The title bar reads "Epicea Log". The file path is `/Users/tinchodias/Library/Preferences/pharo/epicea-3.3/Ergo-1_1g37iyv.ombu`. The log content includes:

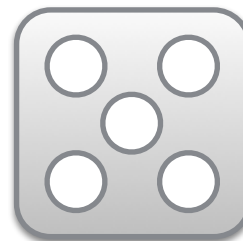
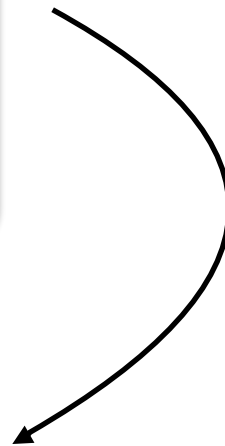
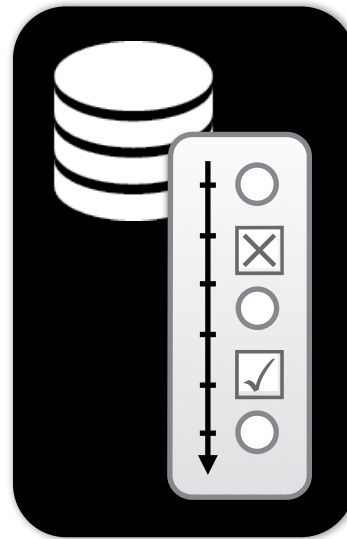
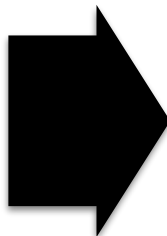
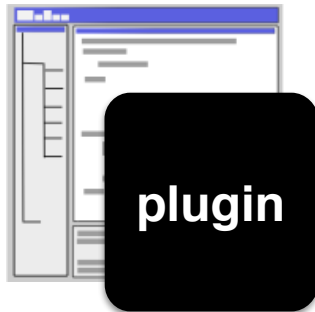
- Snapshot: `/Users/tinchodias/pharo/Epicea-code/ttt17/Ergo-1.image` 18/4/2014 18:01
- ErCollectionValuator » value:variable: 18/4/2014 18:01
- ErCollectionValuatorTest » test01Includes [error] 18/4/2014 18:01
- ErBasicTest rename: #ErCollectionValuatorTest 18/4/2014 18:01
- ErBasicTest ---> ErCollectionValuatorTest 18/4/2014 18:01
- ErBasicTest » setUp 18/4/2014 18:00
- ErCollectionValuator 18/4/2014 18:00
- DoIt: 'ErValuator subclass: #ErCollectionValuator inst...etc...' 18/4/2014 18:00
- ErIncludes » value: 18/4/2014 17:59 (highlighted)
- ErIncludes 18/4/2014 17:59
- ErIncludes » value: 18/4/2014 17:59
- ErIncludes class » value: 18/4/2014 17:58
- ErIncludes class » value: 18/4/2014 17:58
- ErBasicTest » test01Includes [error] 18/4/2014 17:58
- ErValuator » value:variable: 18/4/2014 17:57

At the bottom, there is a comparison view with two tabs: "Event" and "Filters". The "Event" tab shows two side-by-side event details:

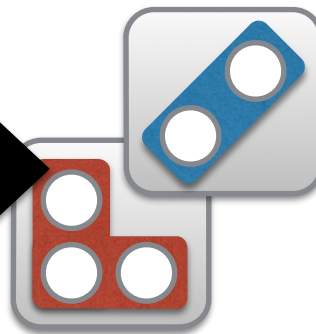
Event 1	Event 2
"protocol: #accessing"	"protocol: #accessing"
value: anInteger	value: anObject
self shouldBelImplemented.	value := anObject



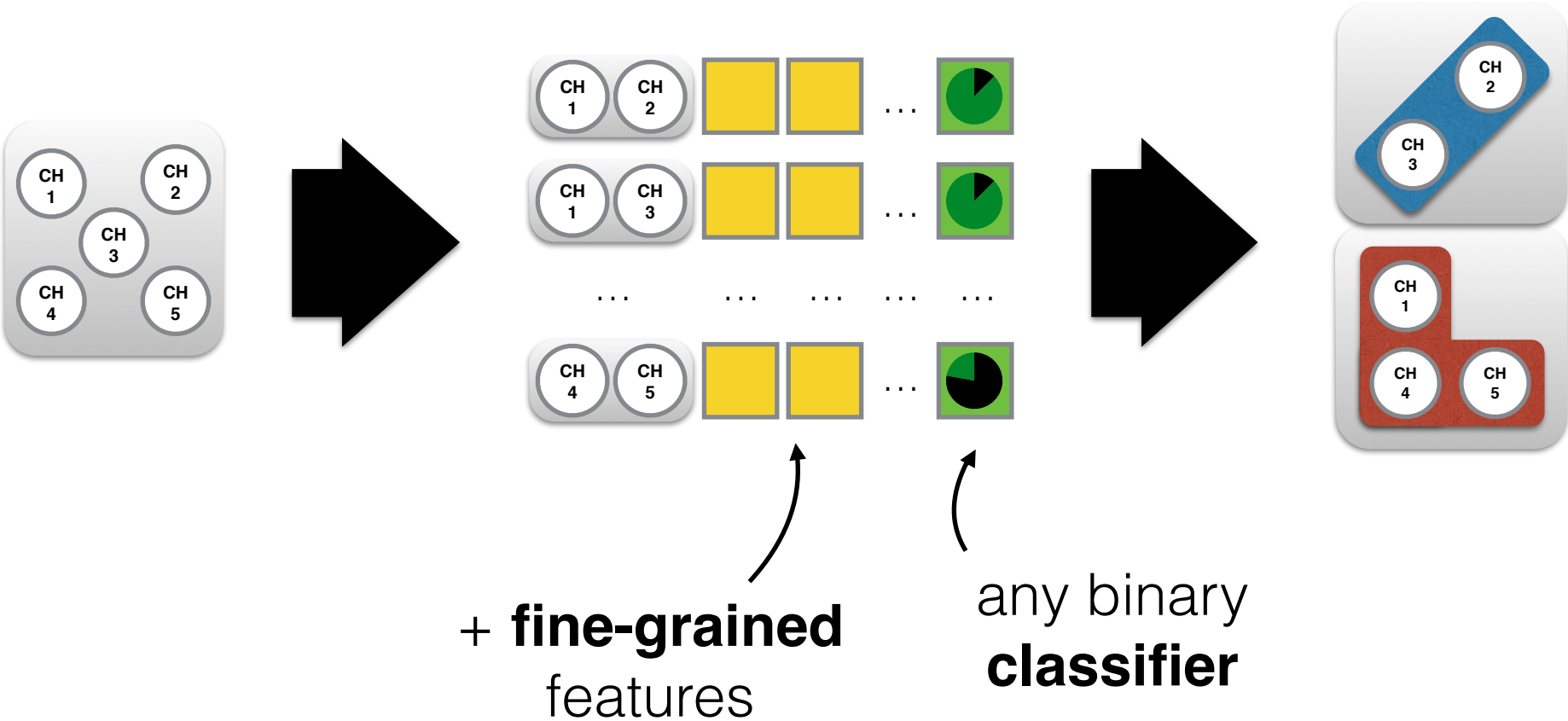
**Fine-Grained  
Code Changes**



untangler  
tool

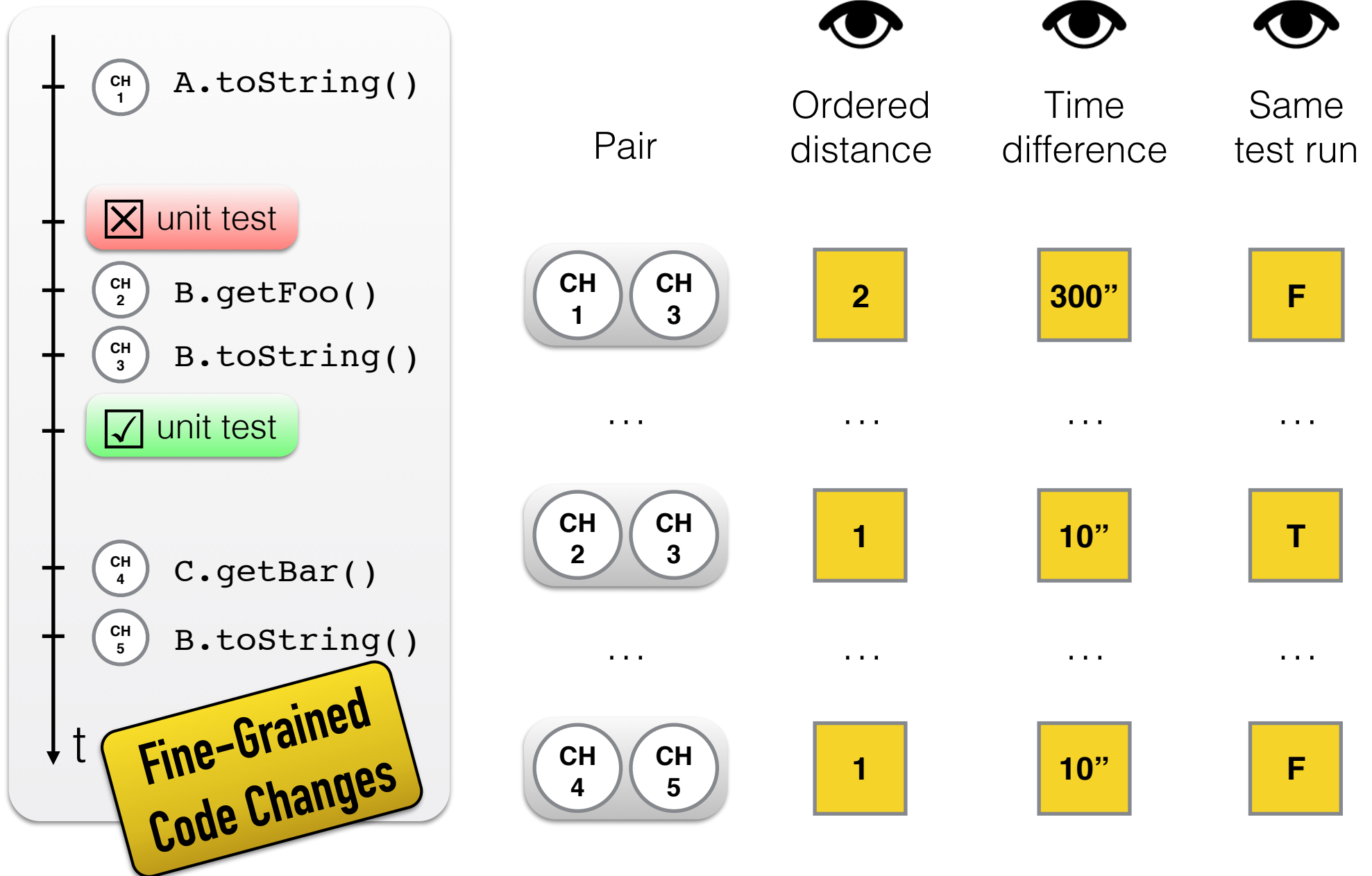


# Epicea Untangler





# Epicea Untangler: Features



# Epicea Untangler: Features

## Fine-grained Code Change Analysis

- ordered distance
- timestamp difference
- same test run

## Static Code Analysis

- same class
- same package
- same method name
- # shared variable accesses
- # shared method calls
- # shared variable accesses in delta
- # shared method calls in delta
- # variable accesses
- reciprocal method calls
- both cosmetic changes

# Epicea Untangler: Classifiers

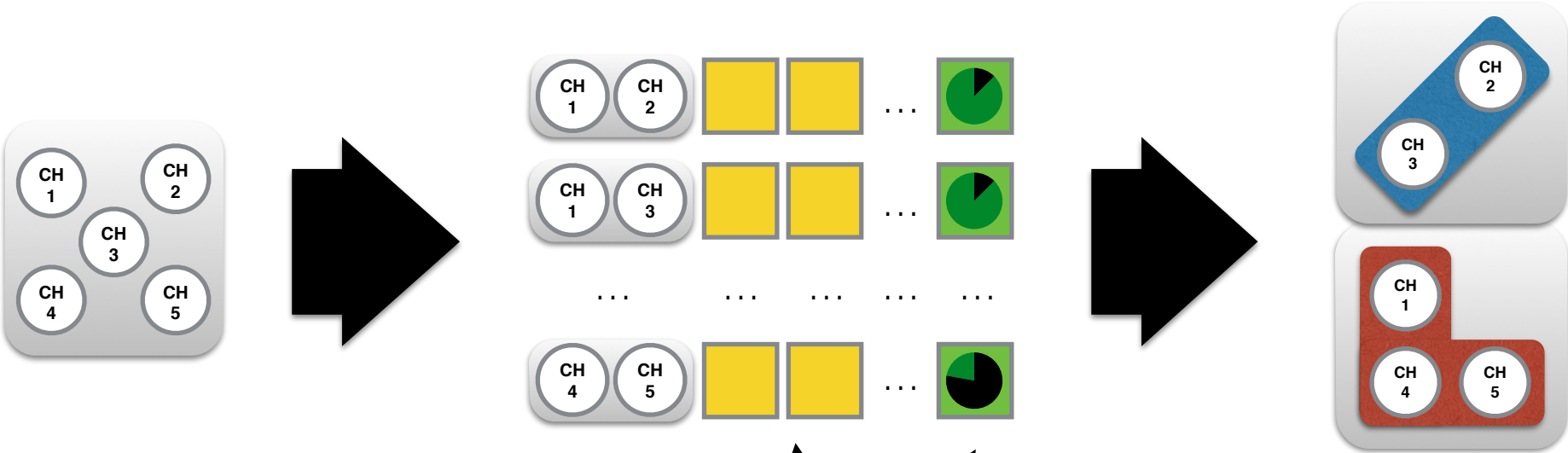
**different assumptions** on  
underlying data and model

■ binary logistic regression

■ naïve bayes

■ random forests

# Epicea Untangler



+ **fine-grained**  
features

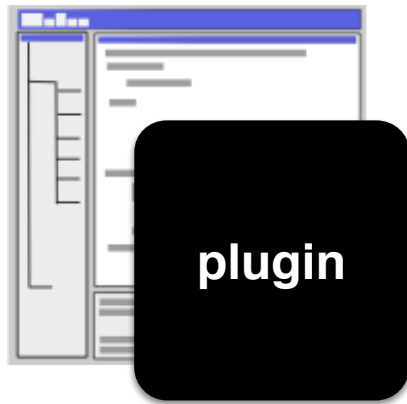
binary  
**classifier**

**RQ**

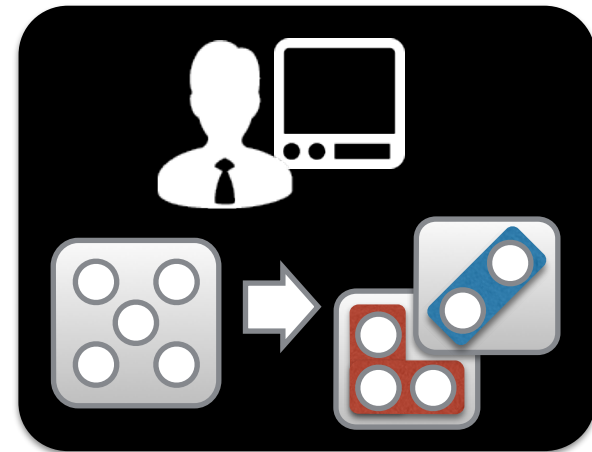
Which features are **dominant**?

**RQ**

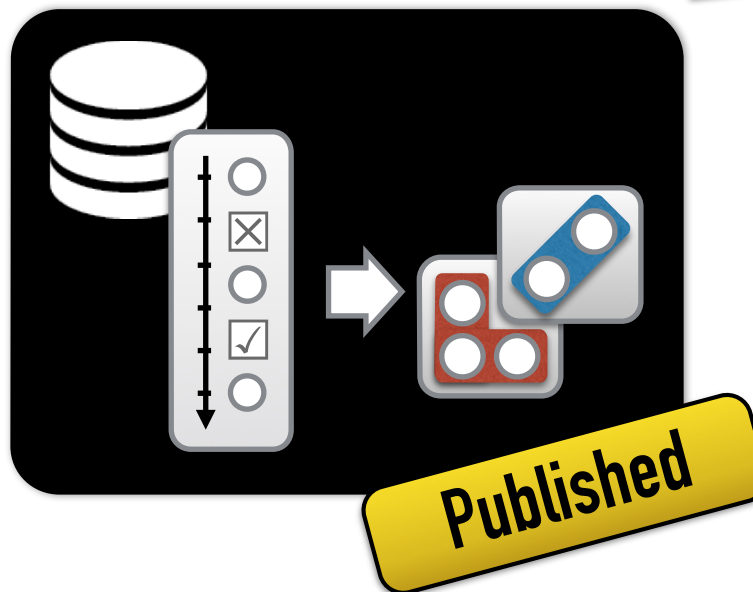
Most **effective**  
classifier?



Manual Untangling



 x 2  
4 months



<http://dx.doi.org/10.6084/m9.figshare.1241571>

Epicea Task Clusterer

Task name	New feature: Add Limit Time
+ AlarmClockTest » test04AdjustLimitTime	+ AlarmClockTest » test03LimitTime 23:14
AlarmClock » initialize 23:22	+ AlarmClock » limitTime: 23:14
+ AlarmClock » defaultLimitTime 23:23	AlarmClock 23:14
AlarmClock » defaultLimitTime 23:23	AlarmClock » limitTime: 23:14
	+ AlarmClock » finished 23:14
	AlarmClock » finished 23:14
	AlarmClockTest » test03LimitTime 23:14
	AlarmClock » finished 23:15

+ AlarmClockTest » test04AdjustLimitTime 23:21  
AlarmClock » initialize 23:22  
+ AlarmClock » defaultLimitTime 23:23  
AlarmClock » defaultLimitTime 23:23

□ Completed      ■ Completed

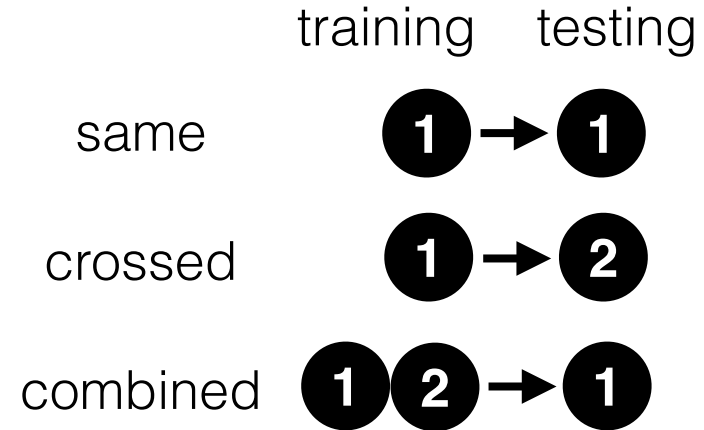
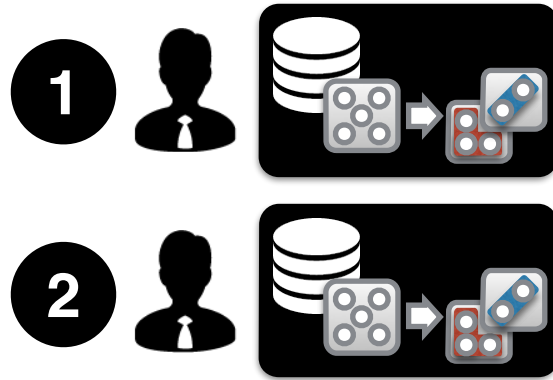
**"protocol: #accessing"**  
**"protocol: #initialization"**

defaultLimitTime  
^ 25 minutes

Done



# Most effective classifier?



	AUC	ACC	PREC	REC	F.MEASURE	G.MEAN
■ binary logistic regression	0.92	0.68	0.43	<b>0.96</b>	0.60	0.76
■ naïve bayes	0.88	0.65	0.41	0.94	0.57	0.73
■ random forests	<b>0.99</b>	<b>0.96</b>	<b>0.96</b>	0.88	<b>0.92</b>	<b>0.93</b>



# Which features are dominant?

**dominant**



- time difference
- ordered distance
- same class

**Simple features!**

	AUC	ACC	PREC	REC	F.MEASURE	G.MEAN
binary logistic regression	0.92	0.68	0.43	<b>0.96</b>	0.60	0.76
naïve bayes	0.88	0.65	0.41	0.94	0.57	0.73
random forests	<b>0.99</b>	<b>0.96</b>	<b>0.96</b>	0.88	<b>0.92</b>	<b>0.93</b>
random forests w/ <b>dominant</b>	0.98	0.95	0.96	0.82	0.88	0.90






# Which features are dominant?

**dominant**



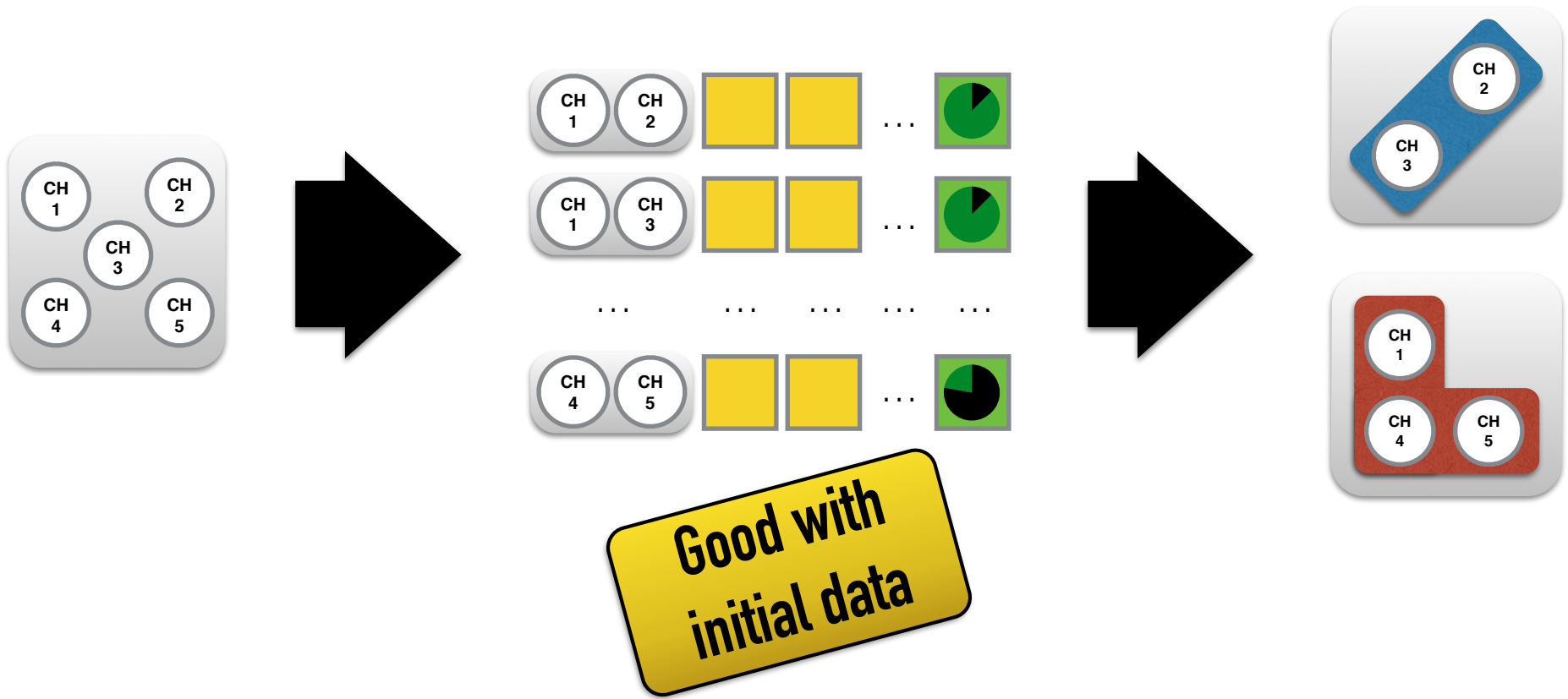
- time difference
- ordered distance
- same class

**Simple features!**

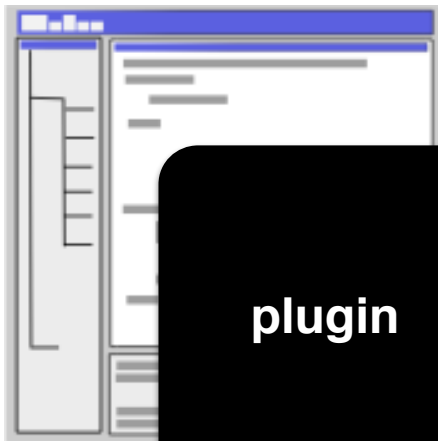
 x 800  97% of accuracy

 x 200  95% of accuracy

**Only 2 days of work!**

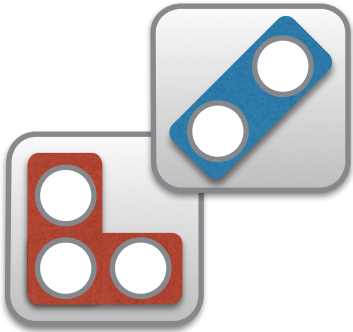
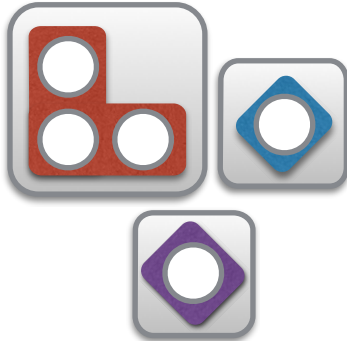


**RQ** Is it effective with **new data from real users?**



**plugin**

**Epicea  
Untangler**



**manual  
sorting**

**6**  


**2 weeks**



## Is it effective with new data?

$$\text{Success rate} = \frac{\# \text{ successfully clustered changes}}{\# \text{ changes}}$$

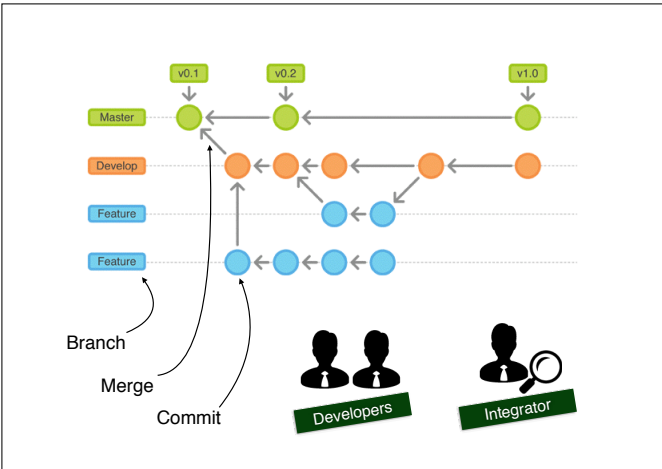
➔ Median success rate: **91%**

➔ Qualitative feedback:

- “It works good in many cases, especially for not so big change sets”
- “It was a bit painful to check everything”

**Conclusion**

# Supporting Software Integration Activities with Fine-grained Code Changes



Questionnaires INRIA  
SOFTWARE INTEGRATION SURVEY

0% 100%

**Authorship/Ownership**  
These questions are related to the owner of the original code, and author of the commit.

Please rank each question below.  
(A1): The word "Importance" refers to the support to the integration task that the answer of that question provides.  
(A2): Indicates the coverage of your tools for answering the question.

	(A1) What is the importance of this question?				(A2) Do your tools answer this question?			
	Nothing	Little	Moderate	Extreme	No	Partially	Yes	No answer
"Who is the author of this changed code?"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
"Who was the previous owner of the changed code?"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
"Has my own code been changed?"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
"What is the general quality of the change committer?"	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>
"How many people have contributed to this group of commits?"	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>

Resume later | Exit and clear survey | Previous | Next

