

# MacroRecorder: Recording and Replaying Source Code Transformations

---

Gustavo SANTOS



# Software Evolution

---

- Software is in constant evolution to remain useful [Leh1980]
- Evolution is composed of changes
  - Performed in distinct moments in time
  - By many developers
- Developers need to reason about code changes [Hat2011]

# Software Evolution

---

- Systematic Code Changes
- In Eclipse 2.1 → 3.0, for example:

**move class C** to a package 'ui.ide'  
in the initializer of C, **add invocation** to method 'setActionId'

Applied **22** times

- Related to:
  - Bug Fixes [Ngu2010]
  - API Updates [Ray2012]
  - Improve the organization

# Transformation Patterns

---

- Total of eleven patterns in real software systems

Transformation patterns	Number of operators	Pattern occurrences
Eclipse (first)	4	26
Eclipse (second)	1	(70)72
JHotDraw	5	9
MyWebMarket	5	7
PackageManager (first)	5	66
PackageManager (second)	9	19
PackageManager (third)	4	64
PackageManager (fourth)	2	7
PetitDelphi	2	(15)19
PetitSQL	4	6
VerveineJ	2	3

Transformation Patterns are frequent

# Transformation Patterns

---

- In JHotDraw, some operators were not applied
- In other systems, the pattern was not applied at once

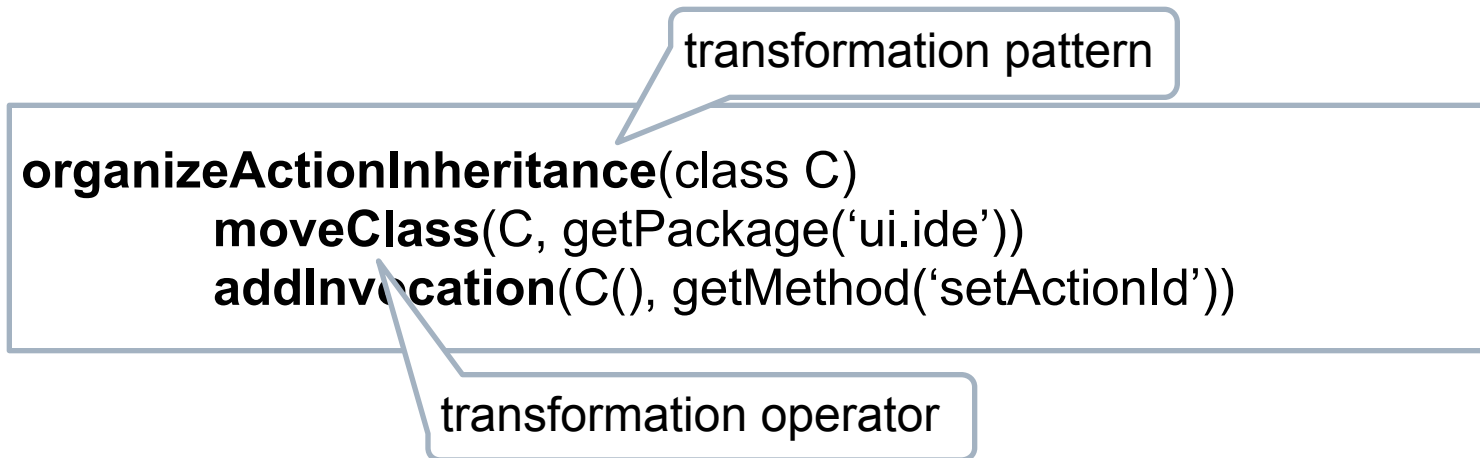
System	#Rev.	Date	Occurrences
Eclipse (second)	3.0	06/25/04, 12:08	70
	3.1	06/27/05, 14:35	71
	3.2	06/29/06, 19:05	72
	3.3	06/25/07, 15:00	72
	3.7	06/13/11, 17:36	72
PetitDelphi	210	11/19/14, 14:52	15
	211	11/19/14, 18:56	17
	212	11/26/14, 18:17	18
	213	12/03/14, 18:23	18
	214	12/22/14, 15:55	19

Transformation Patterns are complex

# Transformation Pattern

---

- Sequences of **repetitive** transformations that are applied to **similar** code entities



- Operators can be atomic or aggregated

# Systematic Code Changes

---

- Performing these sequences manually can be:
  - Complex
  - Tedious
  - Error-prone
- Examples in real world systems [San2015]
- Automation is needed

# Problem

---

- Generate customizable, composite, and abstract transformations
- Related Work in Automated Code Transformation

	Application	Destination of Changes
Sydit [Men2011]	Bug fixes	methods only
Lase [Men2013]	Bug fixes	methods only
Critics [Zha2015]	Bug fixes	customizable



# Solution

---

**move class** StoreAction to a package 'ui.ide'  
in StoreAction(), **add invocation** to 'setActionId'

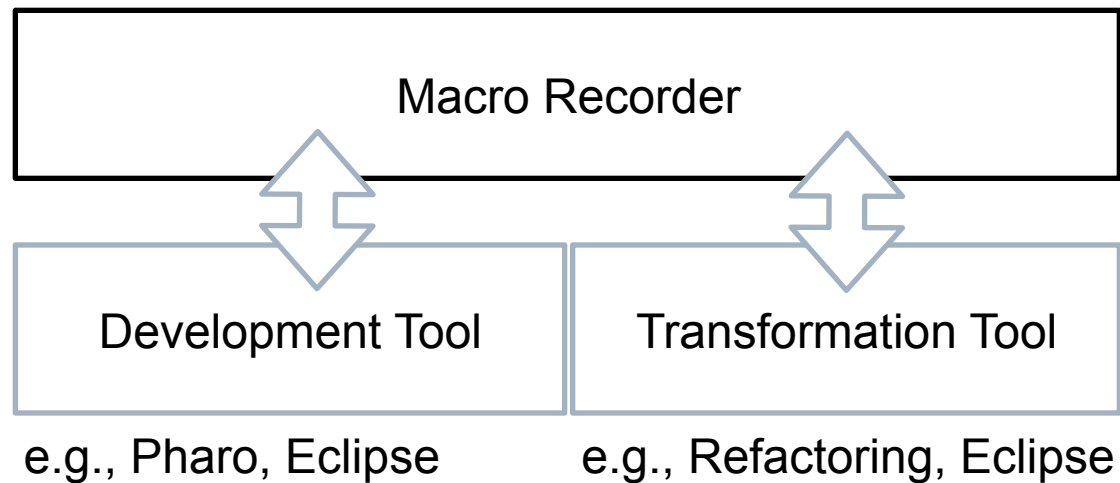
- What if the developer could...
  - Perform the changes manually **once**
  - Generalize the performed changes
  - Replay the changes in other locations

**execute it** for all class C that **extends** eclipse.Action

# Approach

---

- MacroRecorder
  - For each recorded event in the development tool, generate an equivalent transformation



# Illustrating Example

---

**remove method** block() in class Parser  
**remove class** BlockNode

- Enable recording
- Then perform the change manually
- Stop recording

# Illustrating Example - Record

The screenshot shows a window titled "New Transformation Pattern (MRTransformationPattern)". It is divided into two main panes: "Operators List" on the left and "Variables List" on the right. Below these panes is a "Changed Code" section with a "Pretty print" checkbox and a "Name:" field.

**Operators List:**

- PDASTDelphiParser » block 22:09
- PDASTBlockNode 22:09

**Variables List:**

- @method1 #block
- @class1 PDASTDelphiParser
- @className1 #PDASTBlockNode

**Changed Code:**

```
"protocol: #'as yet unclassified'"  
  
block  
  ^ super block ==> [ :parsingItems | PDASTBlockNode new ]
```

Callouts indicate the following changes:

- performed changes:** Points to the "Operators List" pane.
- changed code:** Points to the "Changed Code" section.
- changed entities:** Points to the "Variables List" pane.

# Illustrating Example - Generalize

New Transformation Pattern (MRTransformationPattern)

Operators List

- PDASTDelphiParser » block 22:09
- PDASTBlockNode 22:09

Variables List

- @method1 #block
- @class1 PDASTDelphiParser
- @className1 #'#PDASTBlockNode'

add a new value

Define new variable value

Please define the new value as a text expression (e.g., the name of a class/method). The value is only changed if the expression is valid

"PDAST" + @method1.name().asCamelCase() + "Node"

OK Cancel

Changed Code

"protocol: #'as yet unclassified'"

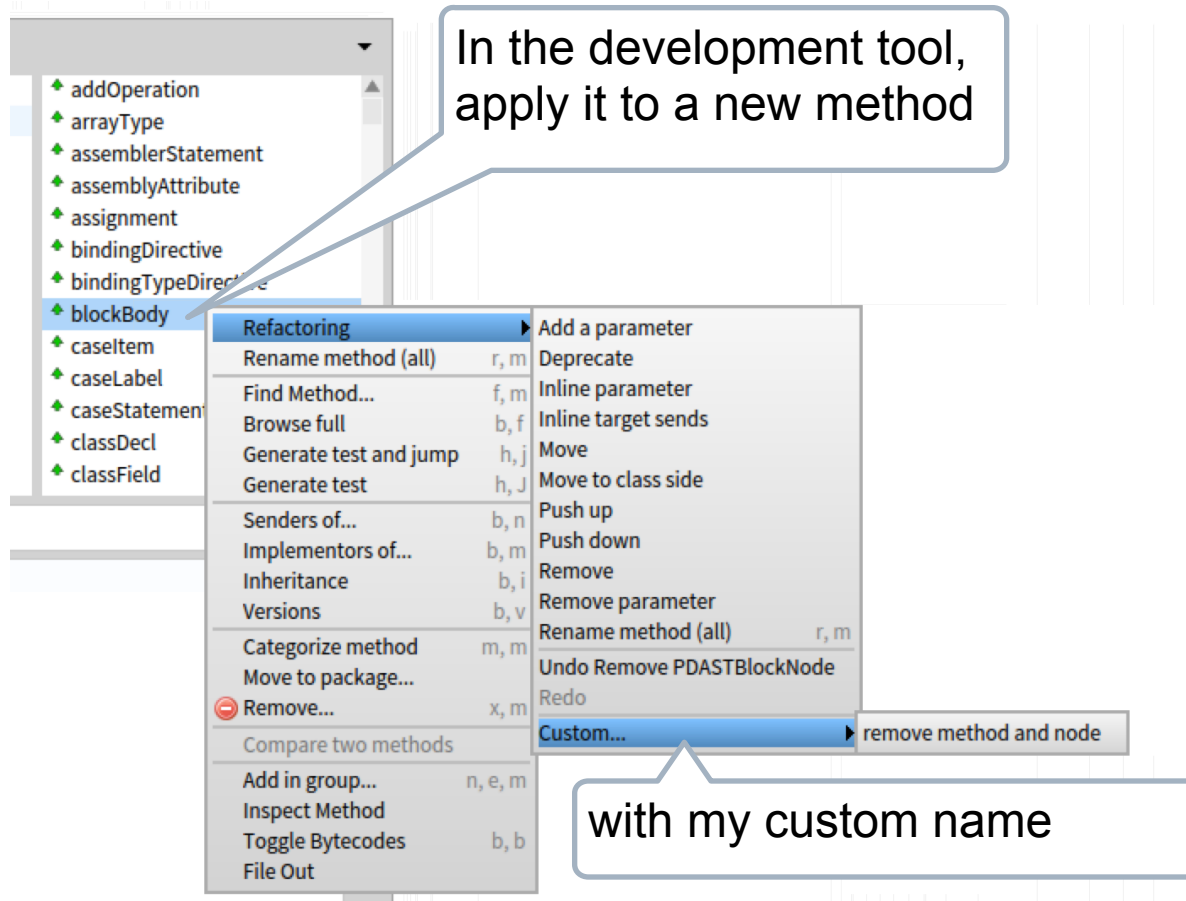
block

^ super block ==> [: parsingItems | PDASTBlockNode new ]

Name: Save

add a new name, then save

# Illustrating Example - Replay



# Illustrating Example - Replay

The screenshot shows a dialog window titled "remove method and node (MRTransformationPattern)". It contains two panes: "Operators List" and "Variables List".

- Operators List:** Contains two entries: "PDASTDelphiParser » block" (timestamp 22:09) and "PDASTBlockNode" (timestamp 22:09).
- Variables List:** Contains three entries: "@method1 #blockBody", "@class1 PDASTDelphiParser", and "@className1 \"PDASTDelphiParser\"".

A callout bubble points to the "@method1 #blockBody" entry in the Variables List, containing the text: "with the selected method".

A "Changes Browser" dialog is overlaid on the main dialog. It has a list of changes with checkboxes:

- Remove PDASTDelphiParser>>#blockBody
- Remove PDASTBlockBodyNode

Below the list, the text "Remove PDASTDelphiParser>>#blockBody" is displayed in green. At the bottom of the "Changes Browser" are "Ok" and "Cancel" buttons.

A callout bubble points to the "Changes Browser" with the text: "Inspect the changes before applying, then execute".

At the bottom of the main dialog, there is a "Name:" field containing "remove method and node" and a "Replay!" button. A callout bubble points to the "Replay!" button with the text: "Replay".

# Status

---

- Prototype tool
- Proof-of-concept study on Smalltalk systems
  - covered 92% of new code locations
  - 76% of the resulting code is behavior-equivalent to the developer's
  - 79% of the resulting code is similar to developer's manual edition
  - Submitted paper on SCAM



# Future Work

---

- Use MacroRecorder on the patterns we found before
  - Submitted paper to SCAM
- Improve the automation
  - Applying the changes to all entities in the system (matching a condition)
  - Generalize the parameters automatically



<http://smalltalkhub.com/#!/~GustavoSantos/MacroRecorder>

---