

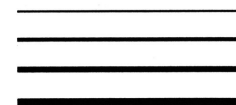
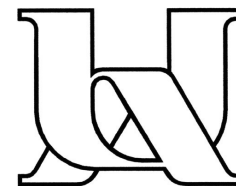


Evaluating the impact of refactorings on OO metrics

Bart Du Bois

Lab On REengineering

Dept. Wiskunde-Informatica



UNIVERSITEIT
ANTWERPEN



The problem

- There are no guidelines on how to apply refactorings in order to improve maintainability.
 - How to estimate maintainability?
 - How do refactorings affect these estimates?



Solution Approach

- Formalize metrics as maintainability indicators
- Formalize refactorings wrt metric impact

Δ Metric	Metric1	...	MetricN
Refactoring1	+		-
...			
RefactoringN	?		0

+ : [0,+inf]

0 : [0,0]

- : [-inf,0]

? : [-inf,+inf]



Formalism

- Abstract Syntax Tree T
 - Nodes
 - ✓ System, Package, Class, Attribute, Method, Local Variable, Parameter, Expression
 - Edges
 - ✓ Membership

- AST extensions ET
 - Superimposed edges
 - ✓ Inheritance, method call, access, update, type



Using the formalism

➤ Selection

- Nodes: $\text{nodeType}(T(\text{node}))$
- Edges:
 - ✓ Incoming: $\text{ET}(\text{subtree}, \text{edgeType})_{\text{inc}}$
 - ✓ Outgoing: $\text{ET}(\text{subtree}, \text{edgeType})_{\text{out}}$
 - ✓ Internal: $\text{ET}(\text{subtree}, \text{edgeType})_{\text{int}}$
 - ✓ External: $\text{ET}(\text{subtree}, \text{edgeType})_{\text{ext}}$
- Cardinality: $\#\text{Set}$
- Source(edgeSet), target(edgeSet)



Describing refactorings w formalism

```

01 public package LAN {
02   public class Machine {
03     public String name;
04     public Machine nextMachine;
05     public void accept(Packet p) {
06       System.out.println(name
07         + " is accepting "
08         + nextMachine.name);
09       this.send(p); }
10   protected void send(Packet p) {
11     System.out.println(name
12       + " is sending "
13       + nextMachine.name);
14     this.nextMachine.accept(p); }
15   }
16   public class Packet {
17     public String contents;
18     public Machine originator;
19     public Machine addressee;
20   }
21   public class PrintServer
22     extends Machine {
23     public void print(Packet p) {
24       System.out.println(p.contents); }
25     public void accept(Packet p) {
26       if(p.addressee == this)
27         this.print(p);
28       else super.accept(p); }
29   }
30   public class Workstation
31     extends Machine {
32     public void originate(Packet p) {
33       p.originator = this;
34       this.send(p); }
35     public void accept(Packet p) {
36       if(p.originator == this)
37         System.err.println("no dest");
38       else super.accept(p); }
39   }
40 }

```

type	$\Delta T(c)$	type ϵ	$\#ET(Machine, \epsilon)_{int}$	$\#ET(Machine, \epsilon)_{inc}$	$\#ET(Machine, \epsilon)_{out}$
M	2	i		2	0
A	3	t	1	2	5
P	2	c			8
L	2	a	11		2
E	24	u	0	0	0

**Impact table
for
Extract Method**

type	$\Delta T(c_i)$	ϵ	$\Delta ET(c_i, \epsilon)_{int}$
M	1	t	
A	0	c	1
P	$\#target(T(setE) \xrightarrow{[au]} T(m_1))$	a	$\#target(T(setE) \xrightarrow{a} T(m_1))$
L	0	u	$\#target(T(setE) \xrightarrow{u} T(m_1))$
E	$1 + \#target(T(setE) \xrightarrow{a} T(m_1)) + \#target(T(setE) \xrightarrow{u} T(m_1))$		



Describing metrics w formalism

➤ Primitive metrics:

– Based on model elements

- ✓ $\text{NOM}(\text{class}) = \#M(T(\text{class}))$
- ✓ $\text{NOC}(\text{class}) = \#ET(\text{class}, i)_{inc}$
- ✓ $\text{instantiatedTypes}(\text{class}) = \text{target}(ET(\text{class}, t)_{out})$
- ✓ $\text{calledClasses}(\text{class}) = \text{target}(ET(\text{class}, cm)_{out})$

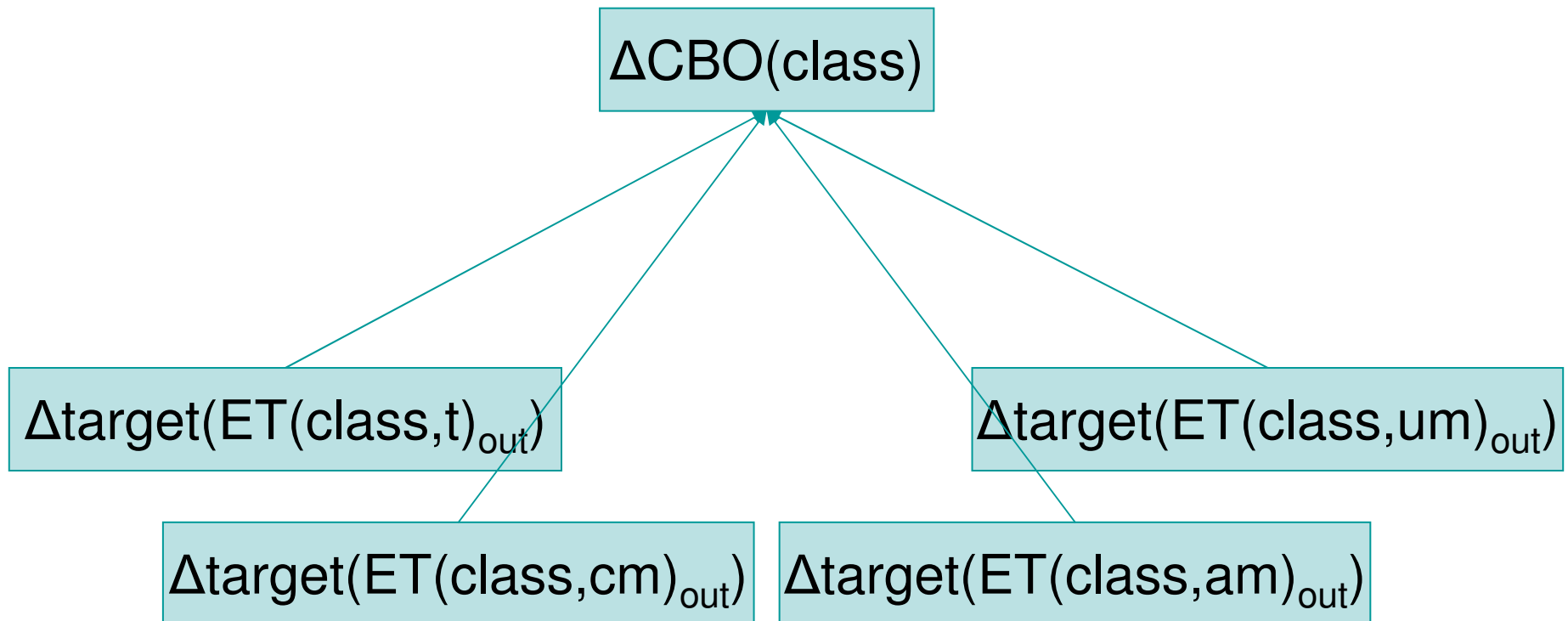
➤ Composite metrics:

– Based on primitive metrics

- ✓ $\text{CBO}(\text{class}) = \#\text{target}(ET(\text{class}, [t|cm|am|um])_{out})$



Constructive impact calculation

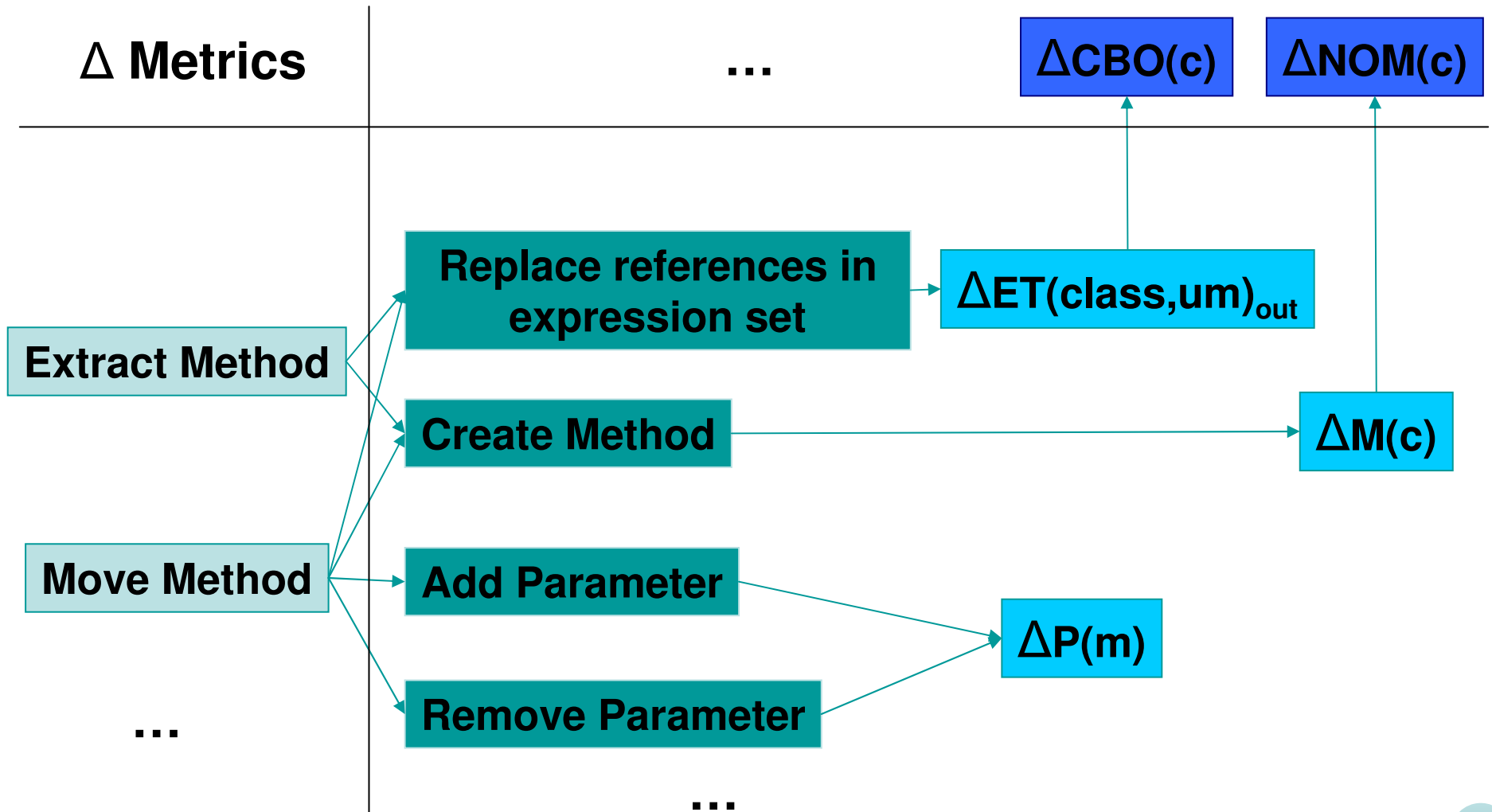


➤ Shifts problem to...

- Mapping of impact on primitive metrics
- Composition of impacts on primitive metrics



Constructive impact model





Work breakdown

- Identify primitive refactoring operations
 - Study potential reuse in Fowler refactorings
 - Should not be “behavior” preserving
- Identify primitive metrics
 - Study potential reuse in metric formalizations
- Calculate impact of primitive refactorings on primitive metrics
- Study composition rules