2IS55 Software Evolution

Repository mining

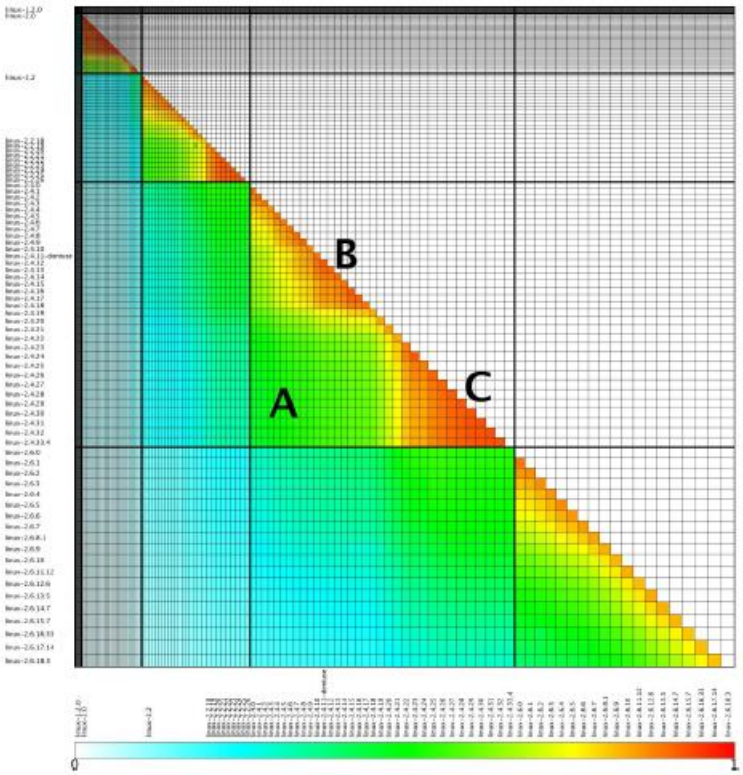
Alexander Serebrenik

Technische Universiteit Eindhoven University of Technology

Where innovation starts

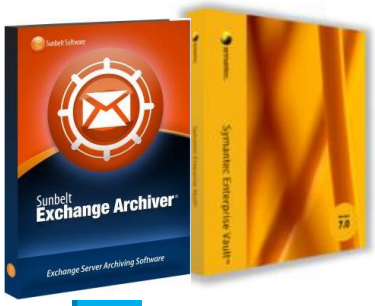
Assignments: Reminder

- **Assignment 2**
  - **Architecture Reconstruction**
  - **Deadline: Saturday**
    - **Ask for extension if needed**
- **Assignment 3**
  - **Individual**
  - **Code duplication**
  - **Replication study of a scientific paper**



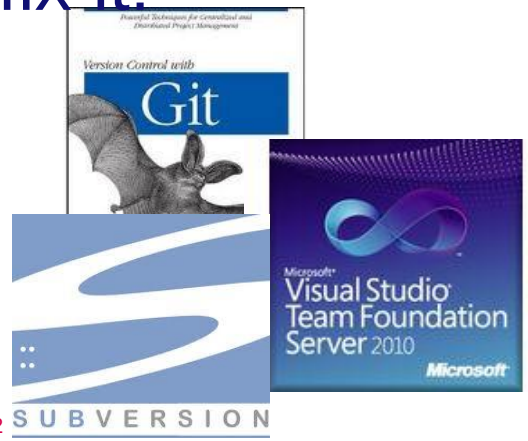
It is all about communication…

Test #14352
fails sometimes



The error should be
somewhere here…
What does this code do?

I know how to fix it!



Tools record information

Software repositories: serving software engineers

Technische Universiteit
Eindhoven
University of Technology

How can the repositories serve you?

- Is the documentation up-to-date?
- How fast are the bugs resolved?
- Who is responsible for
  - Bugs
  - Overtly complex code
  - Code guidelines violations?
- What parts are covered by tests?

and many more…

Software repositories

- Mail archives
- Version control systems
  - CVS, Subversion, Git, Mercurial, …
- Bug trackers
  - Bugzilla, JIRA
- Q&A sites
  - StackOverflow
- Forges
  - Traditional: SourceForge
  - Social: Github

Mail archives

- **Record communication between the developers:**

Sebastian answers Jernej

Re: [Gimp-developer] Trouble compiling plugin in windows

- *From*: Sebastian Kraft <mail sebastiankraft net>
- *To*: gimp-developer-list gnome org
- *Subject*: Re: [Gimp-developer] Trouble compiling plugin in windows
- *Date*: Sun, 17 Feb 2013 20:05:52 +0100

```
Am 17.02.2013 18:43, schrieb Jernej SimonÄiÄ:
| On Sunday, February 17, 2013, 18:19:44, Sebastian Kraft wrote:
```

- **Sender, receiver, Cc, Bcc**
  - **Names and addresses**
- **Date, time and time zone**
- **Re**
- **Subject**
- **Contents**

Mail archives: much more!

- **Record communication between the developers:**

Sebastian answers Jernej

Received: from localhost (localhost.localdomain [**127.0.0.1**])
by restaurant.gnome.org (Postfix) with ESMTP id D18897699B
for <gimp-developer-list@gnome.org>;
Sun, 17 Feb 2013 19:06:07 +0000 (UTC)
Received: from restaurant.gnome.org ([**127.0.0.1**])
by localhost (restaurant.gnome.org [**127.0.0.1**]) (amavisd-new, port 10024)
with ESMTP id s9Pfq7s4t8+E for <gimp-developer-list@gnome.org>;
Sun, 17 Feb 2013 19:05:57 +0000 (UTC)
Received: from mo-p00-ob.rzone.de (mo-p00-ob.rzone.de [**81.169.146.160**])
by restaurant.gnome.org (Postfix) with ESMTP id 0C43076967
for <gimp-developer-list@gnome.org>;
Sun, 17 Feb 2013 19:05:56 +0000 (UTC)
Received: from [**192.168.1.5**] (e182084073.adsl.alicedsl.de [**85.182.84.73**])
by smtp.strato.de (jored mo10) (RZmta 31.17 DYNA|AUTH)
with ESMTPA id 9003d1p1HItiFn for <gimp-developer-list@gnome.org>;
Sun, 17 Feb 2013 20:05:54 +0100 (CET)

Mail archives: much more!

- **Record communication between the developers:**

Sebastian answers Jernej

GIMP mail server 127.0.0.1

127.0.0.1	Local	
127.0.0.1		
81.169.146.160	Berlin	Germany
85.182.84.73	Hamburg	
192.168.1.5	Local	

Sebastian

Sebastian Kraft

Wissenschaftlicher Mltarbeiter bei Helmut-Schmidt-University

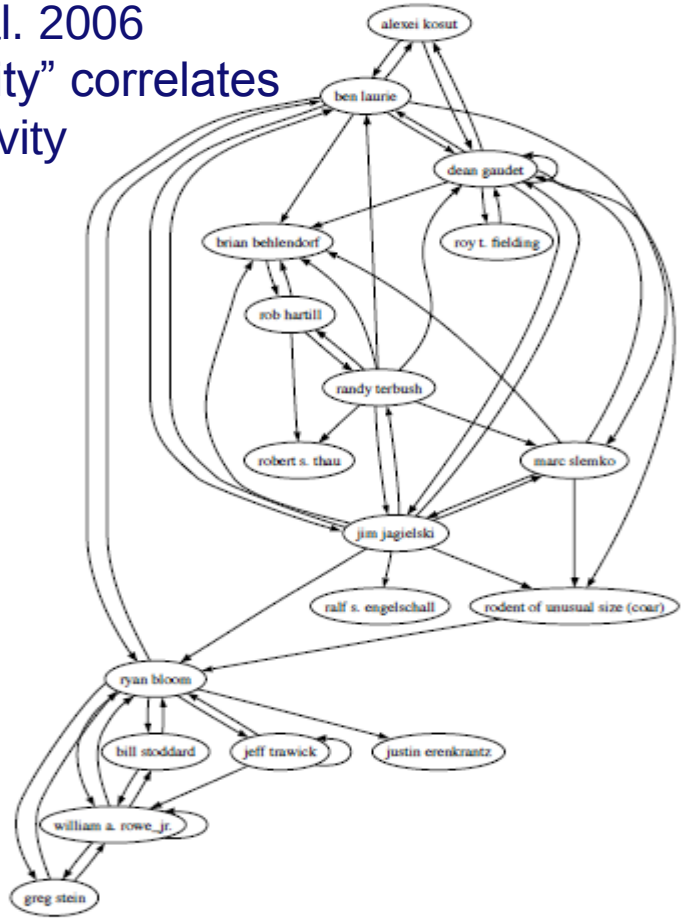
Hamburg Area, Germany | Computer Software

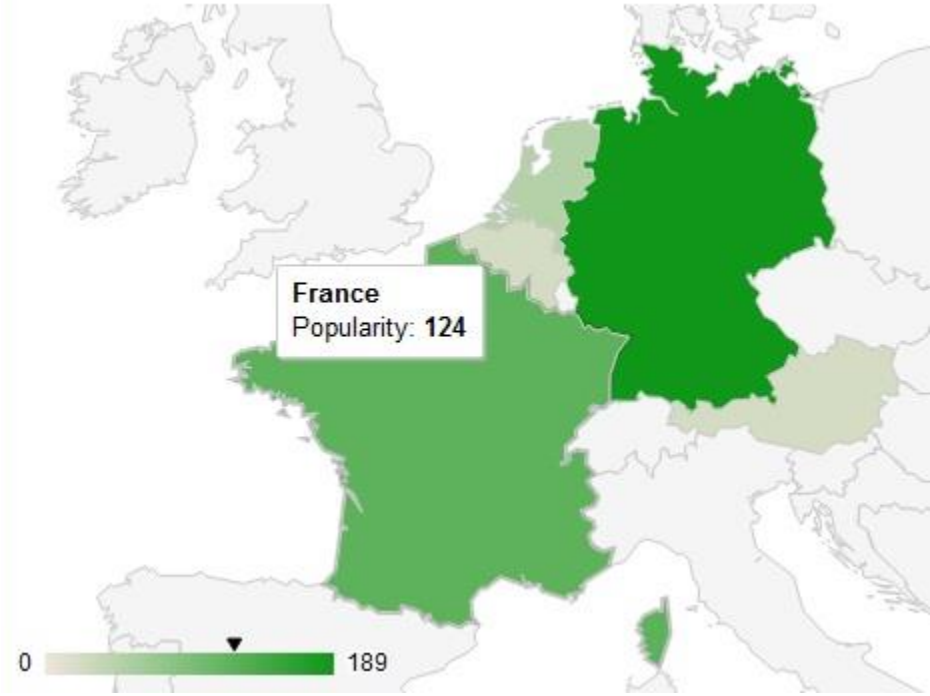
Current	Helmut-Schmidt-University
Previous	Helmut-Schmidt-University, zplane.development, Hochschule für Musik und Theater, Hamburg
Education	TU Hamburg-Harburg

Mail archives: How can we use this information?

- **Record communication between the developers**
- **Structure of the community**

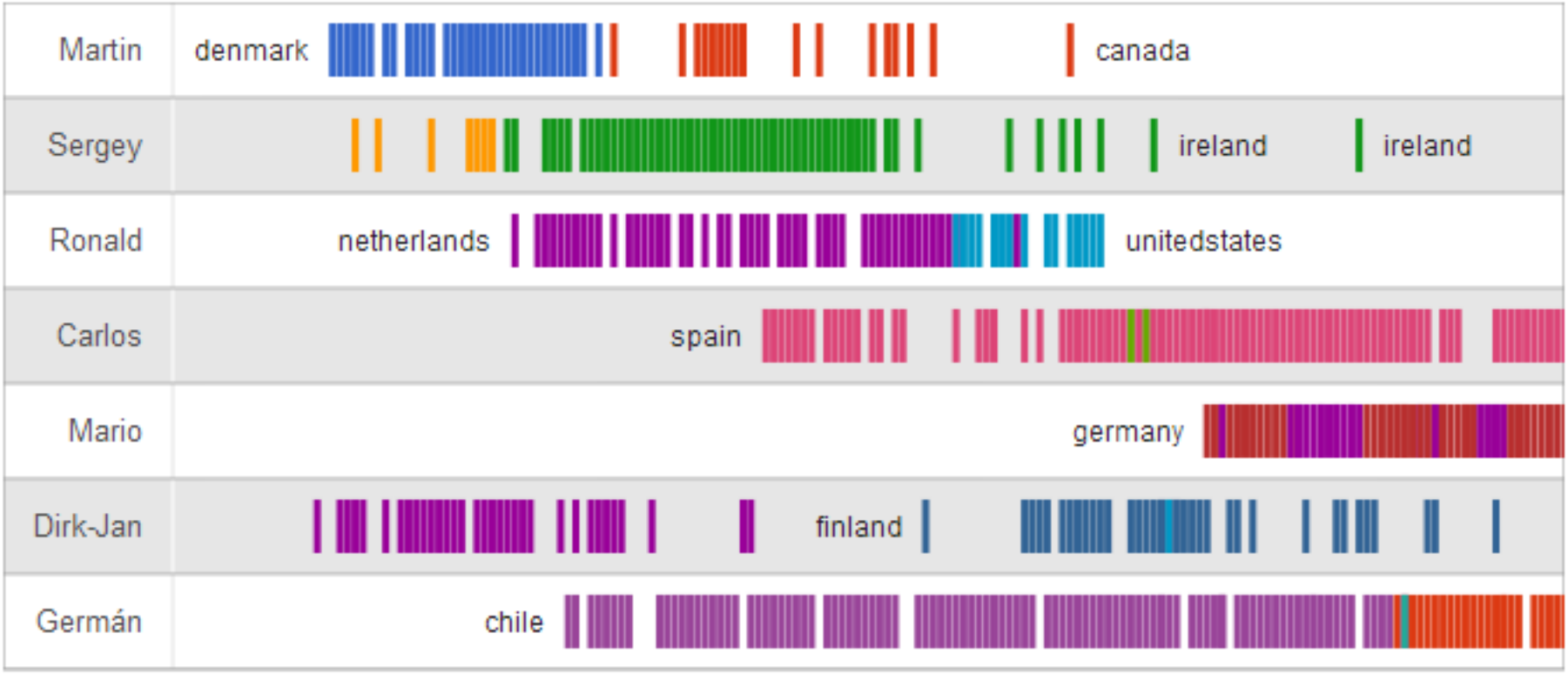
Bird et al. 2006
"Centrality" correlates with activity





Tang et al. 2009
GTK+ mailing list participants in Western Europe

GNOME migration [Kouters 2014]



Bugz

- How
  - Ho being
    res
  - Do aller
    on
  - Do ore
    bu
  - …a

Bug 18063 - Gcc doesn't check overflowed size of structure

Status: NEW

Product: gcc
Component: c
Version: 3.4.2

Importance: P3 minor
Target Milestone: ---
Assigned To: Not yet assigned to anyone

URL:
Keywords: diagnostic

Depends on:
Blocks:
Show dependency tree / graph

Attachments

Add an attachment (proposed patch, testcase, etc.)

Note
You need to log in before you can comment on or make changes to this bug.

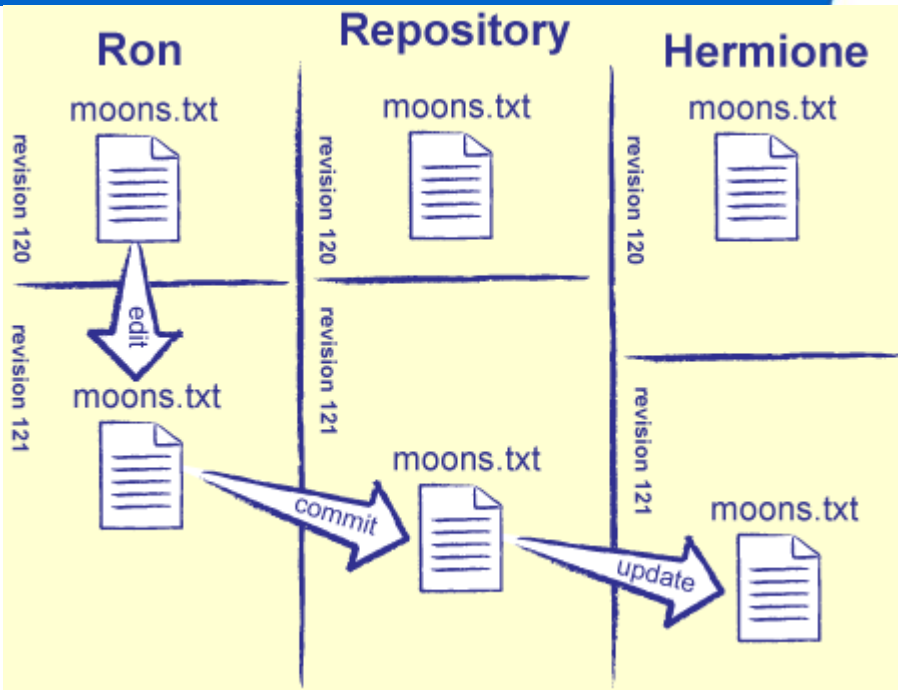
mikulas 2004-10-19 17:20:41 UTC Description

```
The following code compiles and runs, but shouldn't, because the size of
structure a overflows size_t type. Overflowed size is checked for arrays, for
global and local variables, but not for structures.

struct a {
```

Version control in a nutshell

- **System used to reliably reproduce a specific revision of software over time**

- **Terms:**
  - **mainline or trunk**
  - **branch**
  - **tag**
  - **merge**



http://svn.software-carpentry.org/swc/3.0/version/edit-update-cycle.png

Version control examples: CVS

```
RCS file: /cvs/src/sys/arch/sparc/stand/boot/boot.c,v
Working file: boot.c
head: 1.6
branch:
locks: strict
access list:
symbolic names:
   OPENBSD_4_7: 1.6.0.26
        [...]
keyword substitution: kv
total revisions: 12;   selected revisions: 1
description:
----------------------------
revision 1.3
date: 2002/03/14 01:26:44;  author: millert;  state: Exp;  lines: +5 -5
First round of __P removal in sys
=============================================================================
```

- **Information recorded per file**
  - **who, when, changes, branch, tags, message**

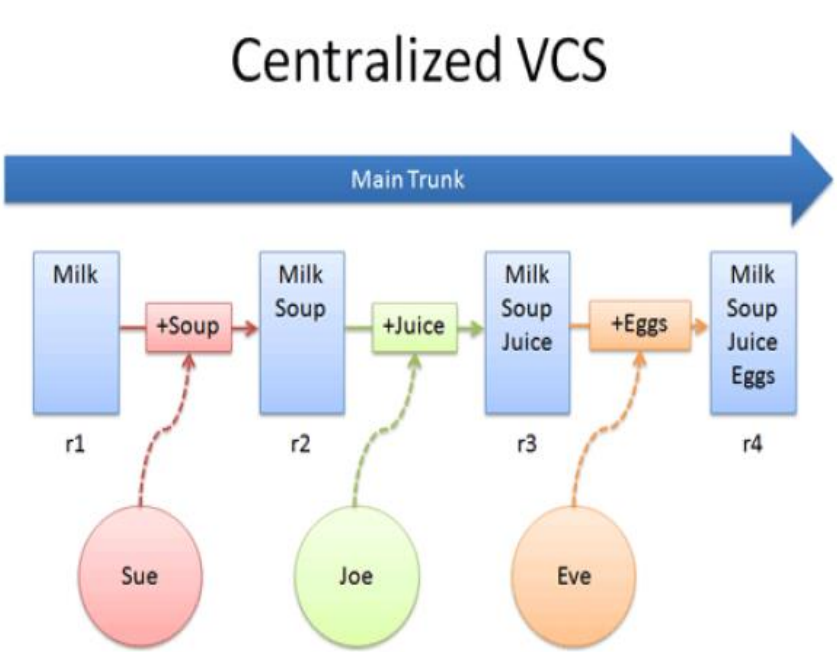
Version control examples: Subversion (SVN)

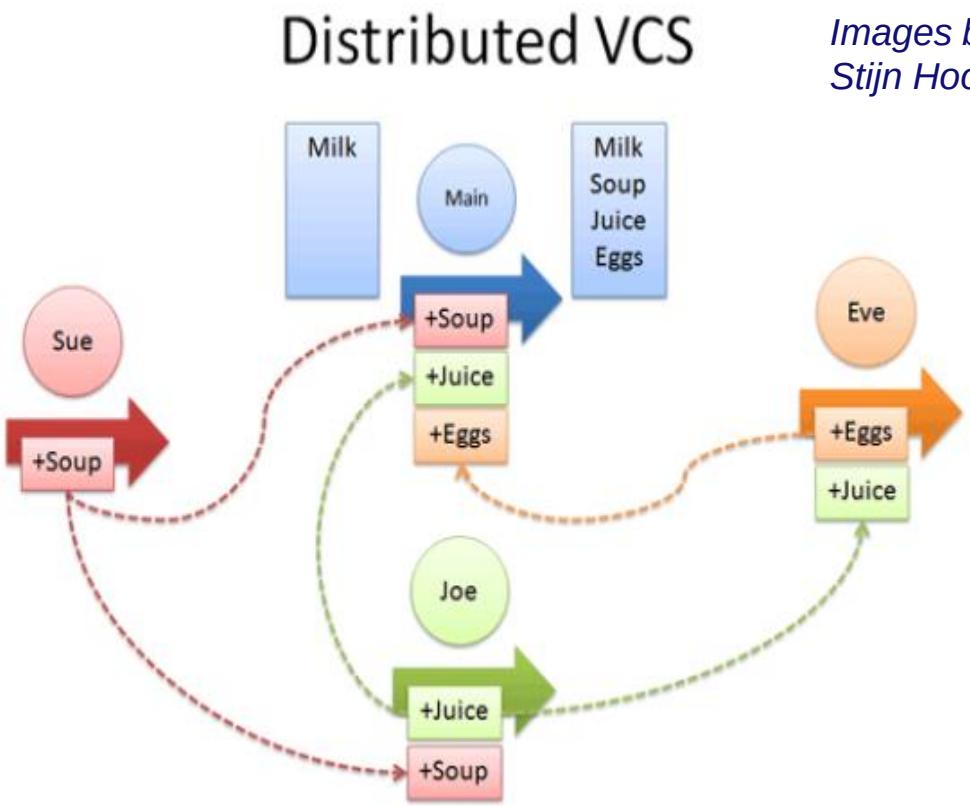
```
------------------------------------------------------------------------
r7524 | wieger | 2010-03-18 11:38:17 +0100 (Thu, 18 Mar 2010) | 1 line
Changed paths:
   M /trunk/tools/pbespareqelm/pbespareqelm.cpp

- Fixed typo in option string.
```

- **Information recorded per commit**
  - **who, when, files, changes, branch, tags, message**

- **What are advantages/disadvantages of recording information per file and per commit?**

So far: Centralized VCS

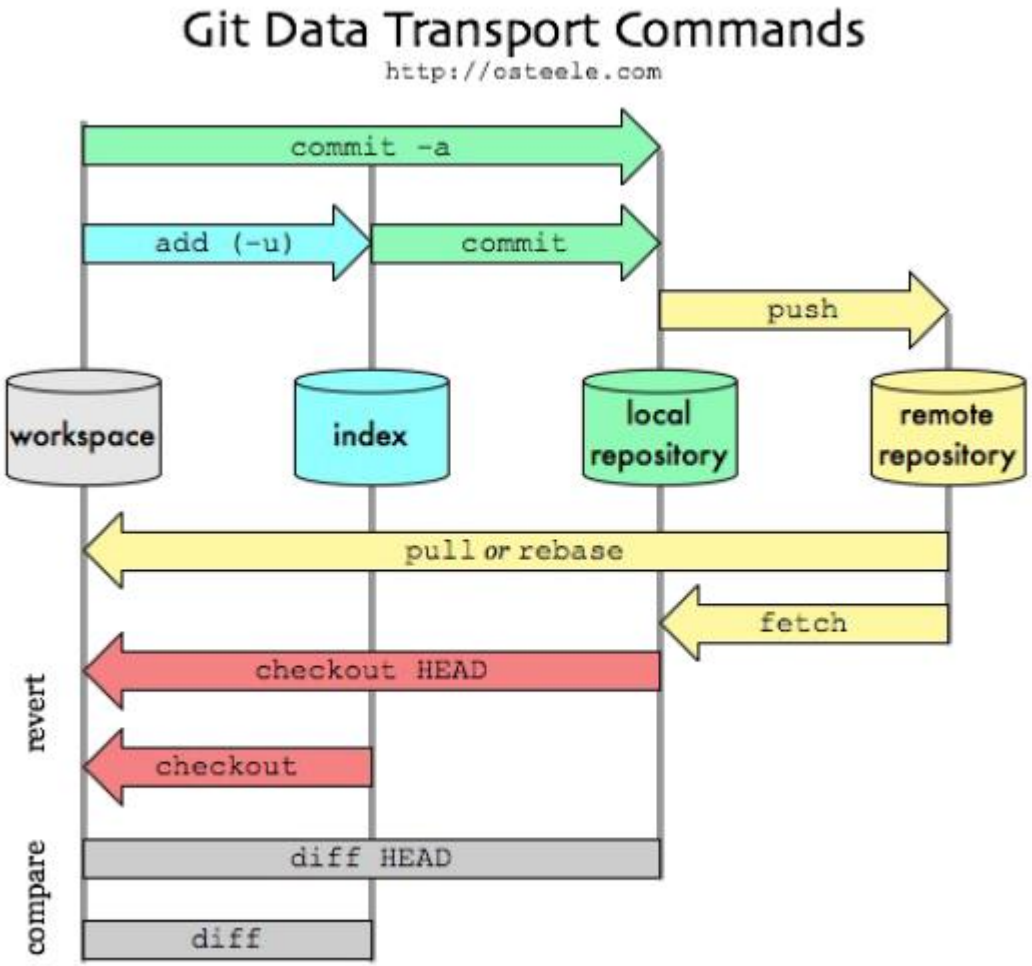




Images by Stijn Hoop

- **Local repositories**
  - **Better for subproject delegation**
  - **Fast**
  - **No need to register**

Distributed version control example: Git



Distributed version control example: Git

```
commit ef36c9a3b2828f5a11feda9e4d2708bf3a4a7a52
Author: Eric Anholt <eric@anholt.net>
Date:   Wed Mar 17 12:46:21 2010 -0700

    intel: Install the header file in the libdrm/ directory.

    Suggested-by: Rémi Cardona <remi@gentoo.org>
    Signed-off-by: Eric Anholt <eric@anholt.net>
```

- **Distinguishes between different kinds of humans**
  - **authors, committers, signed-off-by, …**
- **Much more branches than in centralized VCS**
  - **include in the analysis**
- **Popular projects: Linux Kernel, Android, …**

Version control systems

- Centralized vs. distributed
- File versioning (CVS) vs. product versioning

- Record at least
  - File name, file/product version, time stamp, committer
  - Commit message
  - Changed lines

    $\Rightarrow$ Program differencing

Program differencing: Remote cousin of cloning

- **Formally**
  - **Input: Two programs**
  - **Output:**
    - **Differences between the two programs**
    - **Unchanged code fragments in the old version and their corresponding locations in the new**
- **Similar to clone detection**
  - **Comparison of lines, tokens, trees and graphs**

Diff: Longest common subsequence

- Program: sequence of lines
- Object of comparison: line

- Comparison:
  - 1:1
  - lines are identical
  - matched pairs cannot overlap

- Technique: longest common subsequence
  - Minimal number of additions/deletions steps
  - Dynamic programming

Longest common subsequence

- **Programs X (n lines), Y (m lines)**
- **Data structure C[0..n,0..m]**
- **Init: C[r,0]=0, C[0,c]=0 for any r and c**

```
p0 mA (){
p1   if (pred_a) {
p2       foo()
p3   }
p4 }
```

X

```
c0 mA (){
c1   if (pred_a0) {
c2       if (pred_a) {
c3           foo()
c4       }
c5   }
c6 }
```

Y

C		c 0	c 1	c 2	c 3	c 4	c 5	c 6
	0	0	0	0	0	0	0	0
p0	0							
p1	0							
p2	0							
p3	0							
p4	0							

Longest common subsequence

- **For every r and every c**
  - **If X[r]=Y[c] then C[r,c]=C[r-1,c-1]+1**
  - **Else C[r,c]=max(C[r,c-1],C[r-1,c])**

p0 mA (){
p1 if (pred_a) {
p2 foo()
p3 }
p4 }
 	
X
	c0 mA (){
	c1 if (pred_a0) {
	c2 if (pred_a) {
	c3 foo()
	c4 }
	c5 }
	c6 }
	
	Y

C		c 0	c 1	c 2	c 3	c 4	c 5	c 6
	0	0	0	0	0	0	0	0
p0	0	**1**	1	1	1	1	1	1
p1	0	1	1	**2**	2	2	2	2
p2	0	1	1	2	**3**	3	3	3
p3	0	1	1	2	3	**4**	**4**	**4**
p4	0	1	1	2	3	**4**	**5**	**5**

Longest common subsequence

- **Start with r=n and c=m**
- **backTrace(r,c)**
  - **If r=0 or c=0 then “”**
  - **If X[r]=Y[c] then backTrace(r-1,c-1)+X[r]**
  - **Else**
    - **If C[r,c-1] > C[r-1,c] then backTrace(r,c-1) else backTrace(r-1,c)**

p0 mA (){
p1 if (pred_a) {
p2 foo()
p3 }
p4 }
 	
X
	c0 mA (){
	c1 if (pred_a0) {
	c2 if (pred_a) {
	c3 foo()
	c4 }
	c5 }
	c6 }
	
	Y

C		c 0	c 1	c 2	c 3	c 4	c 5	c 6
	0	0	0	0	0	0	0	0
p0	0	**1**	1	1	1	1	1	1
p1	0	1	1	**2**	2	2	2	2
p2	0	1	1	2	**3**	3	3	3
p3	0	1	1	2	3	**4**	**4**	**4**
p4	0	1	1	2	3	**4**	**5**	**5**

Longest common subsequence

- **Start with r=n and c=m**
- **backTrace(r,c)**
  - **If r=0 or c=0 then “”**
  - **If X[r]=Y[c] then backTrace(r-1,c-1)+X[r]**
  - **Else**
    - **If C[r,c-1] > C[r-1,c] then backTrace(r,c-1) else backTrace(r-1,c)**

X	Y
p0 mA (){	c0 mA (){
p1 if (pred_a) {	c1 if (pred_a0) {
p2 foo()	c2 if (pred_a) {
p3 }	c3 foo()
p4 }	c4 }
	c5 }
	c6 }

C		c 0	c 1	c 2	c 3	c 4	c 5	c 6
	0	0	0	0	0	0	0	0
p0	0	**1**	1	1	1	1	1	1
p1	0	1	1	**2**	2	2	2	2
p2	0	1	1	2	**3**	3	3	3
p3	0	1	1	2	3	**4**	**4**	**4**
p4	0	1	1	2	3	**4**	**5**	**5**

Diff: Summarizing

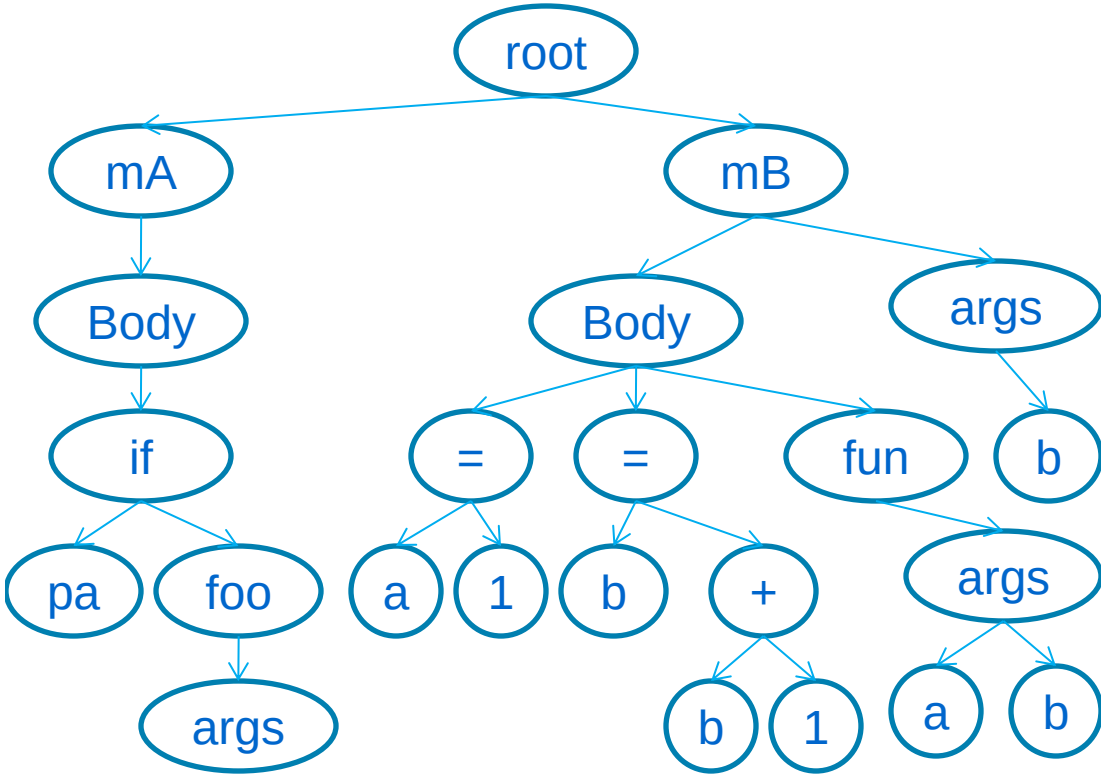
- **Comparison:**
  - **1:1, identical lines, non-overlapping pairs**
- **Technique: longest common subsequence**
- **What kind of code modifications will diff miss?**
  - **Copy & paste: apple ⇒ applple**
    - **1:1 is violated**
  - **Move: apple ⇒ aplep**

More than lines: AST Diff [Yang 1992]

- **Construct ASTs for the input programs**

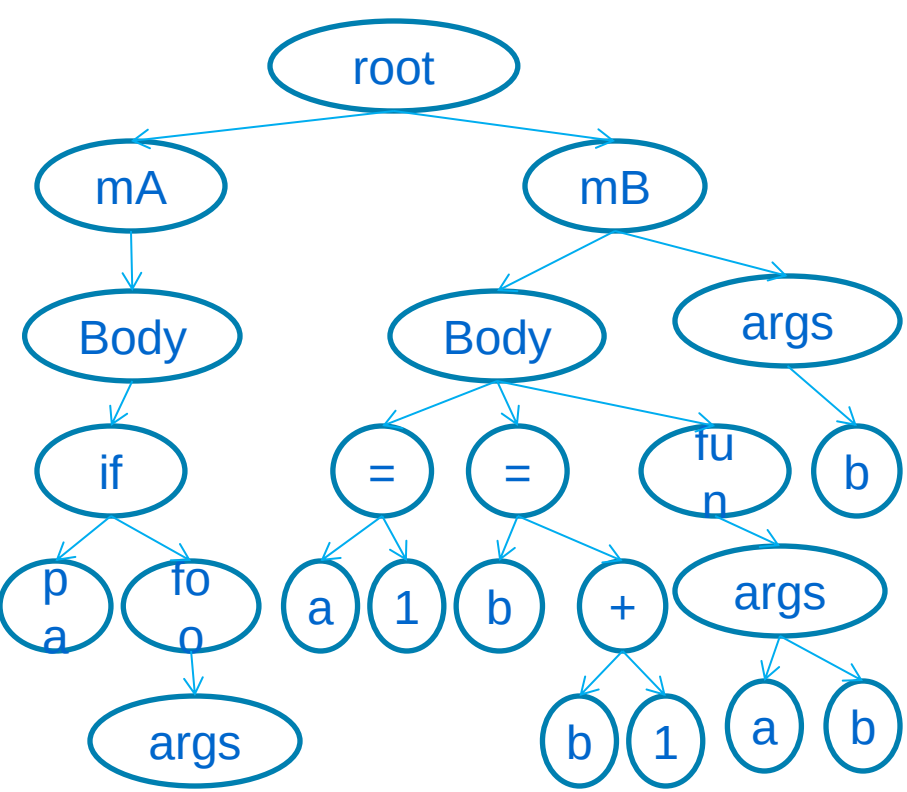
```
p0 mA (){
p1   if (pa) {
p2       foo()
p3   }
p4 }
p5 mB (b) {
p6   a = 1
p7   b = b+1
p8   fun(a,b)
p9 }
```

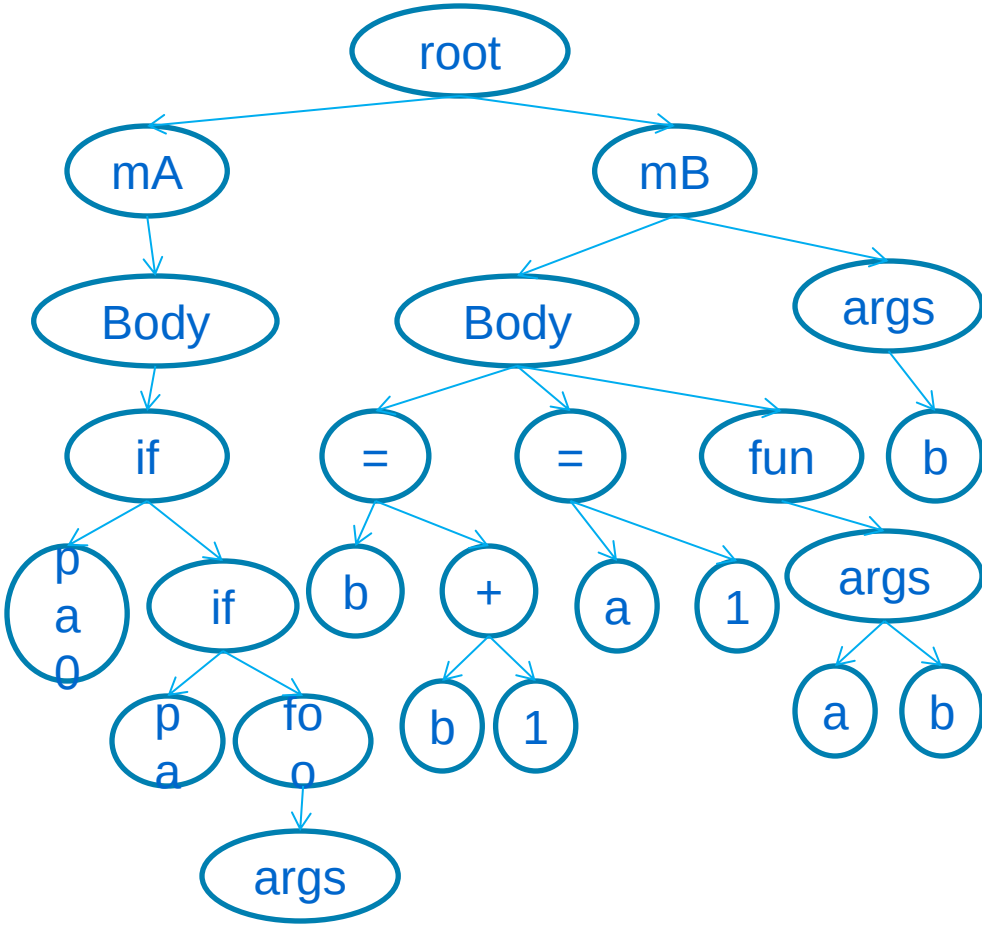
x



More than lines: AST Diff [Yang 1992]

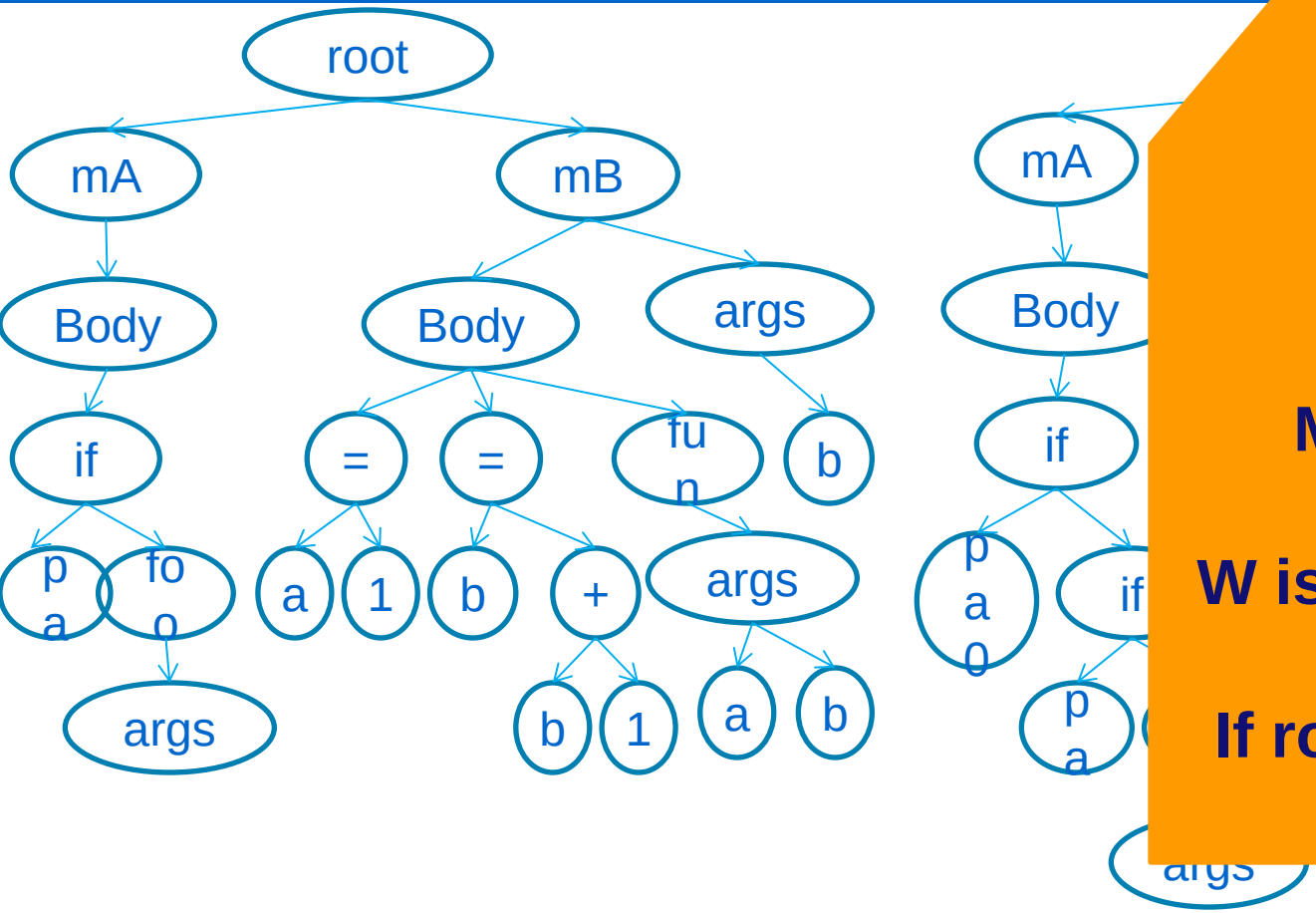
- **Recursive algo pairwise subtree comparison**





- **n – first level subtrees in X**
- **m – first level subtrees in Y**
- **Array: M[0..n, 0..m]**

More than lines: AST Diff [Yang 1992]



M[i,j] = max(M[i,j-1], M[i-1,j], M[i-1,j-1]+W[i,j])

W is the recursive call

If root symbols differ return 0

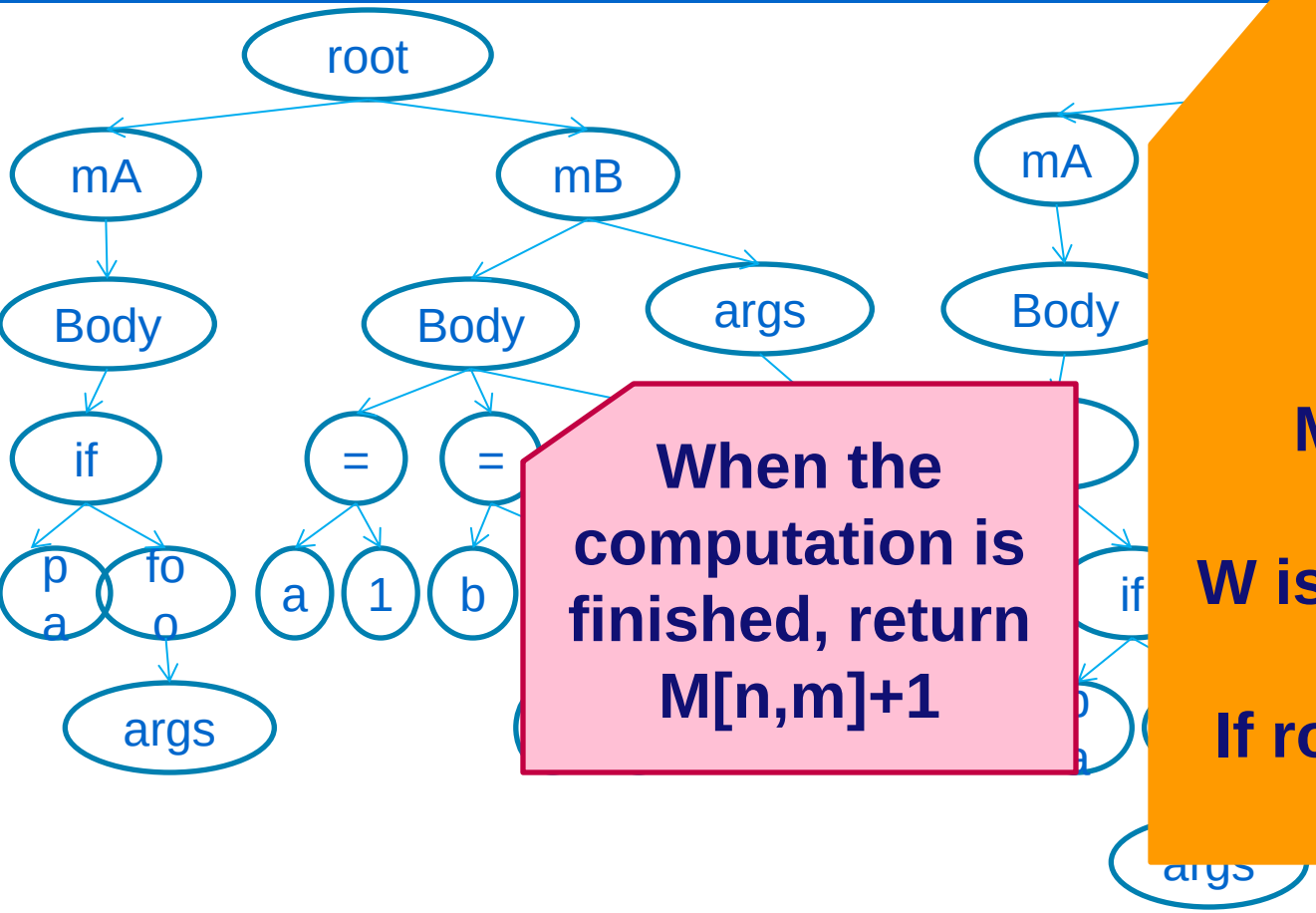
M	0	mA	mB
0	0	0	0
mA	0		
mB	0		

M	0	Body
0	0	0
Body	0	

M	0	if
0	0	0
if	0	

M	0	pa0	if
0	0	0	0
pa	0		
foo	0		

More than lines: AST Diff [Yang 1992]



When the computation is finished, return M[n,m]+1

M[i,j] = max(M[i,j-1], M[i-1,j], M[i-1,j-1]+W[i,j])

W is the recursive call

If root symbols differ return 0

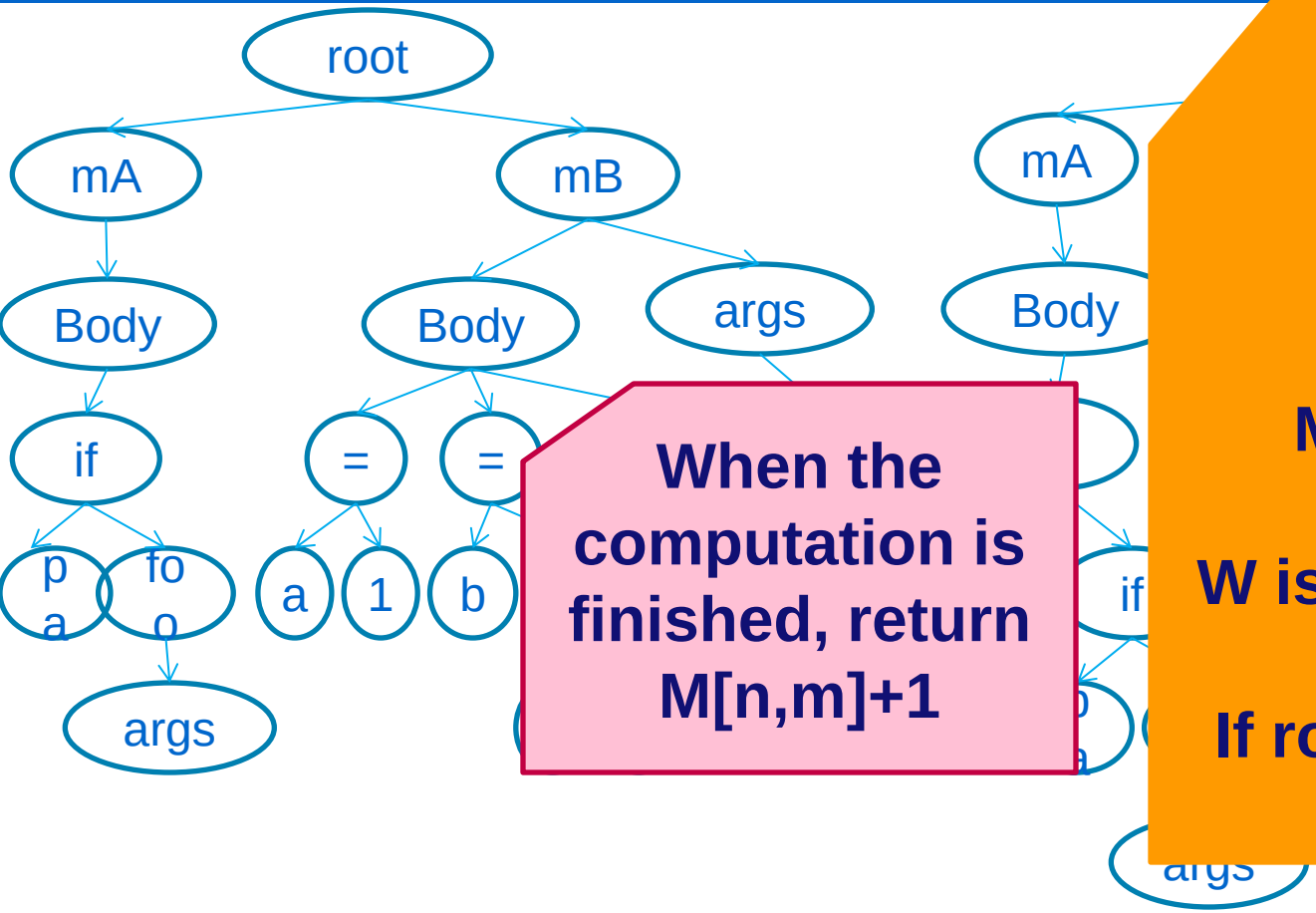
M	0	mA	mB
0	0	0	0
mA	0		
mB	0		

M	0	Body
0	0	0
Body	0	

M	0	if
0	0	0
if	0	1

M	0	pa0	if
0	0	0	0
pa	0	0	0
foo	0	0	0

More than lines: AST Diff [Yang 1992]



When the computation is finished, return M[n,m]+1

M[i,j] = max(M[i,j-1], M[i-1,j], M[i-1,j-1]+W[i,j])

W is the recursive call

If root symbols differ return 0

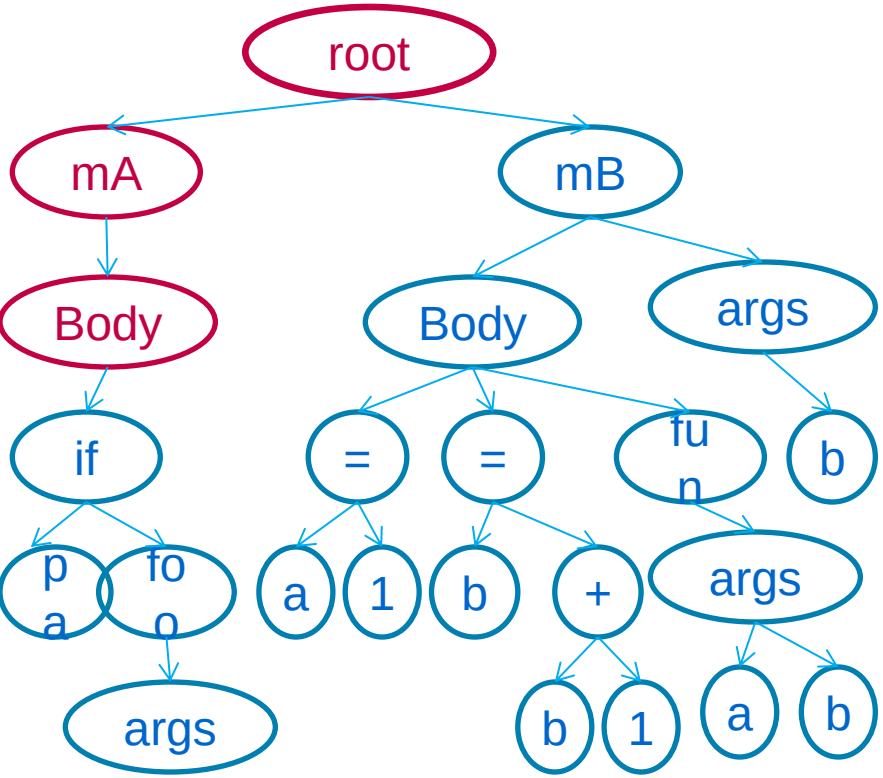
M	0	mA	mB
0	0	0	0
mA	0	3	
mB	0		

M	0	Body
0	0	0
Body	0	2

M	0	if
0	0	0
if	0	1

M	0	pa0	if
0	0	0	0
pa	0	0	0
foo	0	0	0

More than lines: AST Diff [Yang 1992]




```
p0 mA (){
p1   if (pa) {
p2       foo()
p3   }
p4 }
p5 mB (b) {
p6   a = 1
p7   b = b+1
p8   fun(a,b)
p9 }
```

X

```
c0 mA (){
c1   if (pa0) {
c2       if (pa) {
c3           foo()
c4       }
c5   }
c6 }
c7 mB (b) {
c8     b = b+1
c9     a = 1
c10   fun(a,b)
c11 }
```

Y

M	0	mA	mB
0	0	0	0
mA	0	**3**	
mB	0		

Continuing the process

X	Y
p0 mA (){	c0 mA (){
p1 if (pa) {	c1 if (pa0) {
p2 foo()	c2 if (pa) {
p3 }	c3 foo()
p4 }	c4 }
p5 mB (b) {	c5 }
p6 a = 1	c6 }
p7 b = b+1	c7 mB (b) {
p8 fun(a,b)	c8 b = b+1
p9 }	c9 a = 1
	c10 fun(a,b)
	c11 }

- **Advantages**
  - **Respect the parent-child relations**
  - **Ignore the order between siblings**
- **Disadvantages**
  - **Sensitive to tree level changes ("if (pa)")**
  - **Ignore dependencies such as data flow, etc**

- **Can be adapted to OO-specific dataflow (inheritance, exceptions): JDiff**

Changes never come alone

- Search and replace
- Check-in comment: “Common methods go in an abstract class. Easier to extend/maintain/fix”

- Change: a rule rather than a set application results
  - Rules can have exceptions

- Idea [Kim and Notkin 2009]
  - Observe differences between subsequent versions
  - Formalize them as facts
  - Discover rules (à la data mining)
  - Record exceptions

Structural changes [Kim and Notkin 2009]

- **Program ⇒ collection of facts**

v_1

```
class Bus {
    void start(Key c) {
        c.on = false; }
}

class Key {
    boolean on = false;
    void chk (Key c) {…}
    void out() {…}
}
```

FB_0

```
type("Bus")
method("Bus.start","start","Bus")
access("Key.on","Bus.start")
type("Key")
field("Key.on", "on", "Key")
method("Key.chk","chk","Key")
method("Key.out","out","Key")
```

v_2

```
class Bus {
    void start(Key c) {
        log(); }
}

class Key {
    boolean on = false;
    void chk (Key c) {…}
    void output() {…}
}
```

FB_n

```
type("Bus")
method("Bus.start","start","Bus")
calls("Bus.start","log")
type("Key")
field("Key.on", "on", "Key")
method("Key.chk","chk","Key")
method("Key.output","output","Key")
```

Structural changes [Kim and Notkin 2009]

- **Calculate the differences**

type(“Bus”)
method(“Bus.start”,”start”,”Bus”)
access(“Key.on”,”Bus.start”)
type(“Key”)
field(“Key.on”, “on”, “Key”)
method(“Key.chk”,”chk”,”Key”)
method(“Key.out”,”out”,”Key”)

type(“Bus”)
method(“Bus.start”,”start”,”Bus”)
calls(”Bus.start”,”log”)
type(“Key”)
field(“Key.on”, “on”, “Key”)
method(“Key.chk”,”chk”,”Key”)
method(“Key.output”,”output”,”Key”)

ΔFB

deleted_access(“Key.on”,“Bus.start”)
added_calls(”Bus.start”,”log”)
~~deleted_method(“Key.out”,”out”,”Key”)~~
~~added_method(“Key.output”,”output”,”Key”)~~

Deleting Key.out and adding Key.output can be identified as renaming and removed.

Learn the rules

- **Rule inference should depend on ΔFB**
  - **Is this enough? Why?**
  - **Context information is lost!**
    - **We need FB_0 and FB_n**
    - **To distinguish between the facts: past_ and current_**

past_type(“Bus”)
past_method(“Bus.start”,”start”,”Bus”)
past_access(“Key.on”,”Bus.start”)
past_type(“Key”)
past_field(“Key.on”, “on”, “Key”)
past_method(“Key.chk”,”chk”,”Key”)
past_method(“Key.out”,”out”,”Key”)

current_type(“Bus”)
current_method(“Bus.start”,”start”,”Bus”)
current_calls(”Bus.start”,”log”)
current_type(“Key”)
current_field(“Key.on”, “on”, “Key”)
current_method(“Key.chk”,”chk”,”Key”)
current_method(“Key.output”,”output”,”Key”)

deleted_access(“Key.on”,“Bus.start”)
added_calls(”Bus.start”,”log”)

Rules

- **Datalog style**
- **Restrictions:**
  - **Antecedent: one type of facts**
  - **Consequent: only deleted_ or added_**

past_*	⇒	deleted_*	feature or dependency removal
past_*	⇒	added_*	consistent clone update
current_*	⇒	added_*	feature addition
deleted_*	⇒	added_*	API migration
added_*	⇒	deleted_*	

How are the rules learned?

Algorithm 1: LSdiff Rule Inference Algorithm

Input: FB_o, FB_n, ΔFB, m, a, k, and β
Output: L and U

```
/* Initialize R, a set of ungrounded rules; L,
   a set of learned rules; and U, a set of
   facts in ΔFB that are not covered by L.  */
R := ∅, L := ∅, U := ΔFB;
U := applyDefaultWinnowingRules (ΔFB, FB_o,
FB_n);     /* reduce ΔFB with default winnowing
rules.  */
R := createInitialRules (m);     /* create rules
with an empty antecedent by enumerating all
possible consequents.  */
```

- **Winnowing rules remove trivial dependencies**
  - **"if a class is deleted, so are all its methods"**
- **Ungrounded – with variables**

U

deleted_access("Key.on","Bus.start")
added_calls("Bus.start","log")

R

deleted_access(f,m)
added_calls(m1,m2)

How are the rules learned?

Algorithm 1: LSdiff Rule Inference Algorithm

Input: FB_o, FB_n, ΔFB, m, a, k, and β
Output: L and U

```
/* Initialize R, a set of ungrounded rules; L,
   a set of learned rules; and U, a set of
   facts in ΔFB that are not covered by L.  */
R := ∅, L := ∅, U := ΔFB;
U := applyDefaultWinnowingRules (ΔFB, FB_o,
FB_n);     /* reduce ΔFB with default winnowing
rules.  */
R := createInitialRules (m);     /* create rules
with an empty antecedent by enumerating all
possible consequents.  */
foreach i = 1 ... k do
    R := extendUngroundedRules (R) ;     /* extend
    all ungrounded rules in R by adding all
    possible literals to their antecedent.  */
```

- **Winnowing rules remove trivial dependencies**
  - **"if a class is deleted, so are all its methods"**
- **Ungrounded – with variables**

R

deleted_access(f,m) ⇐
  past_method(m, mclass, mshort)
deleted_access(f,m) ⇐
  past_field(f, fshort, fclass)
deleted_access(f,m) ⇐
  past_access(f,m)

How are the rules learned?

Algorithm 1: LSdiff Rule Inference Algorithm

```
Input: FB_o, FB_n, ΔFB, m, a, k, and β
Output: L and U
/* Initialize R, a set of ungrounded rules; L,
   a set of learned rules; and U, a set of
   facts in ΔFB that are not covered by L.  */
R := ∅, L := ∅, U := ΔFB;
U := applyDefaultWinnowingRules (ΔFB, FB_o,
FB_n);     /* reduce ΔFB with default winnowing
rules.  */
R := createInitialRules (m);     /* create rules
with an empty antecedent by enumerating all
possible consequents.  */
foreach i = 1 ... k do
    R := extendUngroundedRules (R) ;     /* extend
    all ungrounded rules in R by adding all
    possible literals to their antecedent.  */
    foreach r ∈ R do
        G := createPartiallyGroundedRules (r) ;
        /* try all possible constant
        substitutions for r's variable.  */
```

R

deleted_access(…) ⇐ past_method(…)
deleted_access (…) ⇐ past_field(…)
deleted_access (“Key.on”,”Bus.start”) ⇐
past_access (“Key.on”,”Bus.start”)
deleted_access (“Key.on”,”Key.chk”) ⇐
past_access (“Key.on”,”Key.chk”)
deleted_access (“Key.on”,”Key.out”) ⇐
past_access (“Key.on”,”Key.out”)
deleted_access (“Key.on”,m) ⇐
past_access (“Key.on”,m)
deleted_access (f,”Bus.start”) ⇐
past_access (f,”Bus.start”)
deleted_access (f,”Key.chk”) ⇐
past_access (f,”Key.chk”)
deleted_access (f,”Key.out”) ⇐
past_access (f,”Key.out”)
deleted_access(f,m) ⇐
past_access(f,m)

How are the rules learned?

Algorithm 1: LSdiff Rule Inference Algorithm

```
Input: FB_o, FB_n, ΔFB, m, a, k, and β
Output: L and U
/* Initialize R, a set of ungrounded rules; L,
   a set of learned rules; and U, a set of
   facts in ΔFB that are not covered by L.  */
R := ∅, L := ∅, U := ΔFB;
U := applyDefaultWinnowingRules (ΔFB, FB_o,
FB_n);    /* reduce ΔFB with default winnowing
rules.  */
R := createInitialRules (m);    /* create rules
with an empty antecedent by enumerating all
possible consequents.  */
foreach i = 1 ... k do
    R := extendUngroundedRules (R) ;    /* extend
    all ungrounded rules in R by adding all
    possible literals to their antecedent.  */
    foreach r ∈ R do
        G := createPartiallyGroundedRules (r) ;
        /* try all possible constant
        substitutions for r's variable.  */
        foreach g in G do
            if isValid (g) then
                L :=L ∪ {g};
                U :=U − {g.matches};
            end
        end
    end
    R :=selectRules (R, β) ; /* select the best β
    rules in R  */
end
```

- **Validity**
  - **≥ m matching facts**
  - **Match/(Match+NonMatch) ≥ a**

R (examples)

deleted_access (“Key.on”,”Bus.start”) ⇐ past_access (“Key.on”,”Bus.start”)

One fact, 100% precision

deleted_access (“Key.on”,”Key.chk”) ⇐ past_access (“Key.on”,”Key.chk”)

No matching facts

deleted_access (“Key.on”,m) ⇐ past_access (“Key.on”,m)

One fact, 100% precision

deleted_access (f,”Bus.start”) ⇐ past_access (f,”Bus.start”)

One fact, 100% precision

deleted_access(f,m) ⇐ past_access(f,m)

One fact, 100% precision

How are the rules learned?

Algorithm 1: LSdiff Rule Inference Algorithm

```
Input: FB_o, FB_n, ΔFB, m, a, k, and β
Output: L and U
/* Initialize R, a set of ungrounded rules; L,
   a set of learned rules; and U, a set of
   facts in ΔFB that are not covered by L.  */
R := ∅, L := ∅, U := ΔFB;
U := applyDefaultWinnowingRules (ΔFB, FB_o,
FB_n);     /* reduce ΔFB with default winnowing
rules.  */
R := createInitialRules (m);     /* create rules
with an empty antecedent by enumerating all
possible consequents.  */
foreach i = 1 ... k do
    R := extendUngroundedRules (R) ;     /* extend
    all ungrounded rules in R by adding all
    possible literals to their antecedent.  */
    foreach r ∈ R do
        G := createPartiallyGroundedRules (r) ;
        /* try all possible constant
        substitutions for r's variable.  */
        foreach g in G do
            if isValid (g) then
                L :=L ∪ {g};
                U :=U − {g.matches};
            end
        end
    end
    R :=selectRules (R, β) ; /* select the best β
    rules in R  */
end
```

- **For more realistic examples**
  - **Thresholds do matter**
  - **More general = less accurate**
- **At the next step more antecedents are added, e.g.,**
  - **past_method(m,"start",t) ∧ past_subtype("Car",t) ⇒ added_calls(m,"Key.chk")**
- **β controls the number of rules kept for extension at the next iteration**

Evaluation

- **On average, 75% of facts in ΔFB are covered by inferred rules**
  - **~75% of structural differences are systematic change.**

- **Concise:**
  - **Textual diff: 997 lines, 16 files**
  - **LSdiff: 7 rules and 27 facts**

How did the users experience the tool?

- **Positive**
  - **“You can’t infer the intent of a programmer, but this is pretty close.”**

- **Negative**
  - **“This looks great for big architectural changes, but I wonder what it would give you if you had lots of random changes.”**
  - **“This will look for relationships that do not exist.”**

What about “random” changes?

- **eROSE [Zimmermann, Weißgerber, Diehl, Zeller ‘04]**

amazon.com

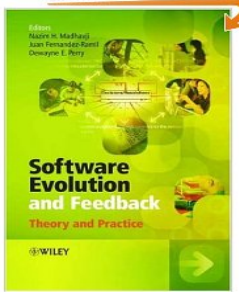
Hello. Sign in to get personalized recommendations. New customer? Start here.

Your Amazon.com | Today's Deals | Gifts & Wish Lists | Gift Cards

Shop All Departments | Search Books | GO

Books | Advanced Search | Browse Subjects | New Releases | Bestsellers | The New York Times® Bestsellers | Libros

Click to LOOK INSIDE!



Share your own customer images

Search inside this book

Software Evolution and Feedback: Theory and Practice (Hardcover)

~ Nazim H. Madhavji (Editor), Juan Fernandez-Ramil (Editor), Dewayne Perry (Editor)

Key Phrases: ripple effect measure, feedforward capability, development effort method, International Conference, New York, Imperial College (more...)

No customer reviews yet. Be the first.

Price: $130.00 & this item ships for **FREE with Super Saver Shipping**. Details

In Stock.

Ships from and sold by **Amazon.com**. Gift-wrap available.

Only 1 left in stock--order soon (more on the way).

Want it delivered Tuesday, March 16? Order it in the next 3 hours and 24 minutes, and choose **One-Day Shipping** at checkout. Details

30 new from $41.90 **10 used** from $41.92

Frequently Bought Together

Customers buy this book with Principles of Program Analysis by Flemming Nielson

+

Price For Both: $192.95

Add both to Cart | Add both to Wish List

Show availability and shipping details

Customers Who Bought This Item Also Bought

Software Evolution by Tom Mens

$80.97

Developers who modified this function also modified…

Java - Eclipse Platform

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

CompareEditorContributor.jav
CompareMessages.java
CompareNavigator.java
ComparePreferencePage.java
import declarations
NEW_PREFERENCE
OPEN_STRUCTURE_C
PREF_SAVE_ALL_EDIT
PREFIX
SHOW_MORE_INFO
SHOW_PSEUDO_CON
SYNCHRONIZE_SCRO
fCheckboxes
fCheckboxListener
fCompareConfiguratio
fKeys
fOverlayStore
fPreferenceChangeLis
fPreviewViewer
ComparePreferencePa
addCheckBox(Compos
createContents(Comp
createGeneralPage(Co
createPreviewer(Comp
createTextComparePa
dispose()
init(IWorkbench)

*ComparePreferencePage.java

```
public final OverlayPreferenceStore.OverlayKey[] fKeys= new OverlayPreferenceStore.OverlayK
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, OPEN_STRUCTURE_COM
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, SYNCHRONIZE_SCROLL
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, SHOW_PSEUDO_CONFLI
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, INITIALLY_SHOW_ANC
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, SHOW_MORE_INFO),
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, IGNORE_WHITESPACE)
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, PREF_SAVE_ALL_EDIT

    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, NEW_PREFERENCE),

    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.STRING, AbstractTextEditor.
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, AbstractTextEditor
    //new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, USE_SPLINES),
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, USE_SINGLE_LINE),
    //new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, USE_RESOLVE_UI),
};

public static void initDefaults(IPreferenceStore store) {
    store.setDefault(OPEN_STRUCTURE_COMPARE, true);
    store.setDefault(SYNCHRONIZE_SCROLLING, true);
    store.setDefault(SHOW_PSEUDO_CONFLICTS, false);
    store.setDefault(INITIALLY_SHOW_ANCESTOR_PANE, false);
    store.setDefault(SHOW_MORE_INFO, false);
    store.setDefault(IGNORE_WHITESPACE, false);
    store.setDefault(PREF_SAVE_ALL_EDITORS, false);
    //store.setDefault(USE_SPLINES, false);
    store.setDefault(USE_SINGLE_LINE, true);
```

A) The user inserts a new preference into the field fKeys[]

B) ROSE suggests locations for further changes, e.g. the function initDefaults()

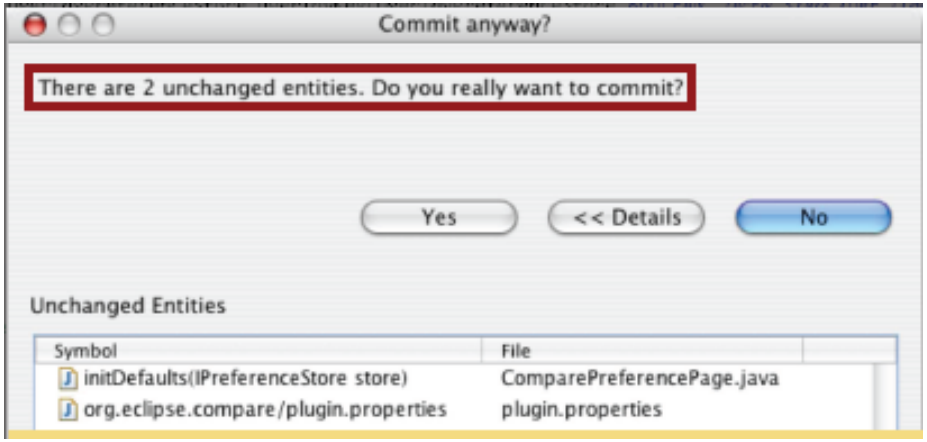
ROSE keeps a list of association rules

Related Changes

Symbol	File	Support	Confidence
initDefaults(IPreferenceStore store)	ComparePreferencePage.java	8	1.0
org.eclipse.compare/plugin.properties	plugin.properties	7	0.875
org.eclipse.compare/buildnotes_compare.html	buildnotes_compare.html	6	0.75
TextMergeViewer(Composite parent, int style, CompareConfiguration configuration)	TextMergeViewer.java	6	0.75
propertyChange(PropertyChangeEvent event)	TextMergeViewer.java	6	0.75
createGeneralPage(Composite parent)	ComparePreferencePage.java	5	0.625
createTextComparePage(Composite parent)	ComparePreferencePage.java	5	0.625
handleDispose(DisposeEvent event)	TextMergeViewer.java	4	0.5

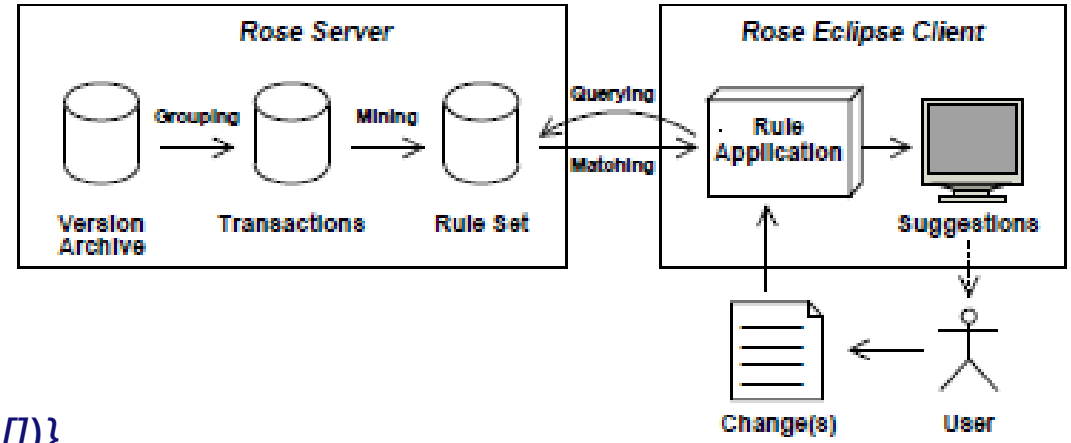
Tasks Console Related Changes

eROSE



- **ROSE alerts for incomplete changes**

How?



Association rules:

{(Comp.java, *field, fKeys[])}*

⇒ { (Comp.java, *method, initDefaults()),*

(plug.properties, *file, plug.properties) }*

Experimental evaluation

- **Recall: 0.15**
  - **suggestion included 15% of all changes that were carried out**
- **Precision: 0.26**
  - **26% of all recommendations were correct**
- **Likelyhood:**
  - **70% of all transactions, topmost three suggestions contain a changed entity.**
- **EROSE learns quickly**
  - **within 30 days**
- **Extensive evaluation**

Conclusions

- Repositories
  - Version control
    - File/commit-level change, centralized/distributed
  - Mail archives, bug trackers
- Differencing
  - Two approaches to identification of related differences:
    - Both based on data mining/rule learning
    - Interesting ideas, not always impressive results
      - A lot of improvement is possible!