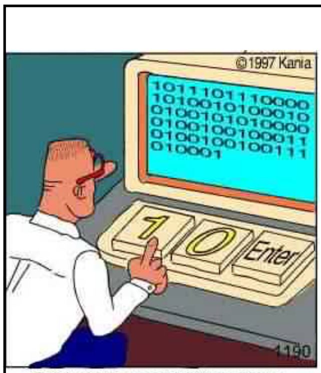


Real programmers code in binary.

Software Visualization

Visualizing the Evolution of Software Systems

Visualizing the Evolution of Software Systems

- Software Evolution
- Examples
  - SeeSoft
  - Evolution Matrix
  - Time Wheel
  - VRCE
  - VRCS
  - Revision Towers
  - GEVOL

SeeSoft, Evolution Matrix, Time Wheel: Metrics

VRCE, VRCS, Revision Towers, GEVOL: Software Archives

Typical question of a project manager

- What subsystems are very big ?
- What subsystems grow very fast?
- Where is the center of the current development, where are most developers working ?
- Are developers mostly fixing bugs or extending the program?
- Which subsystems contained most bugs?
- When were major versions released?

➔ Track changes and collect data during the development process

Metrics

- Size of moduls
- Runtime of program
- Number of Changes
- Number of Bugfixes
- Number of programmers that did a change
- Depth of nested blocks
- Type of error
- ...

Metrics and Program Code

Zeile	Metric
1	12
2	23
3	12
4	12
5	12
6	11
7	10
8	9
9	8
10	12
11	23
12	12
13	12
14	12
15	11
16	10
17	9
18	8
19	12
20	99
21	12
22	12
23	12
24	11
25	10
26	9
27	8

```
try {
    Configuration gcfg;

    Object graph  = cfg.findSingle("graph");

    // 2. Configure the protocol graph for this host...

    if (graph == null)
            throw new configException("Host must have graph attribute: "+cfg);
    else if (graph instanceof String) {
            throw new configException(„File has been deprecated; please use find instead");
    } else
            gcfg = (Configuration)graph;

    super.config(gcfg);

    ProtocolSession IP = SessionForName ("ip");

    // 3. Finally, configure the specified interface set for this host.
   for (Enumeration ifaces = cfg.find("interface");
              ifaces.hasMoreElements();) {
            Configuration ncfg = (Configuration)ifaces.nextElement();
            idrange ids = new idrange(); ids.config(ncfg);
```

Visualizing Software Systems; Marla J. Baker, Stephen G. Eick
Software Visualization in the Large; Thomas Ball, Stephen G. Eick

Requirements of a Representation for Program Code and related Metrics

- Provide an overview
- Fit on the screen!
- Moduls with one million lines of code must be displayed

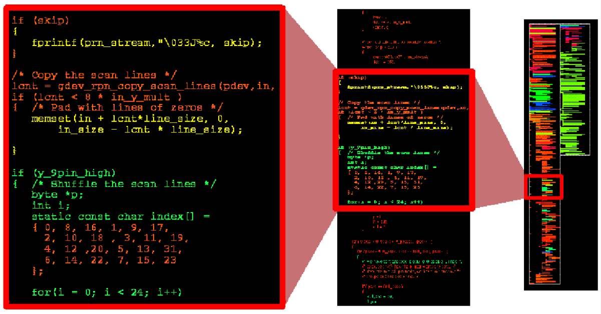
→ Use screen real estate economically

SeeSoft combines

- Colorcoding
- Textual Representation
- Line Representation
  - Each line of text is represented by a colored line of pixels
- Pixel Representation
  - Each line of text is represented by a one (or a few) pixel.
  - Order of pixels:
    - According to order of the lines in the text
    - According to color
- File Summary Representation
  - Every file is represented by a box. There are four different sizes (quartiles).
- Hierarchical Representation
  - Essentially tree maps

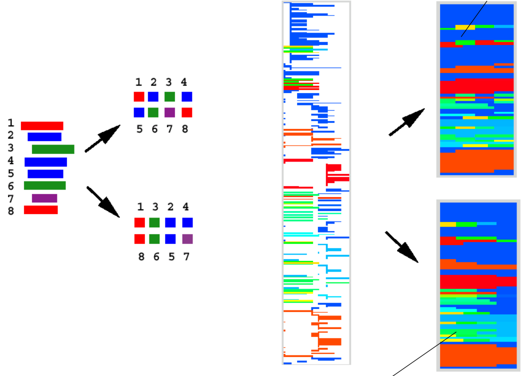
proportional

Line Representation



```
if (skip)
{
   fprintf(prn_stream,"\033J%c, skip);
}

/* Copy the scan lines */
lcnt = gdev_rpn_copy_scan_lines(pdev,in,
if (lcnt < 8 * in_y_mult )
{  /* Pad with lines of zeros */
   memset(in + lcnt*line_size, 0,
        in_size - lcnt * line_size);

}

if (y_9pin_high)
{  /* Shuffle the scan lines */
   byte *p;
   int i;
   static const char index[] =
   { 0, 8, 16, 1, 9, 17,
     2, 10, 18 , 3, 11, 19,
     4, 12 ,20, 5, 13, 31,
     6, 14, 22, 7, 15, 23
   };

   for(i = 0; i < 24; i++)
```

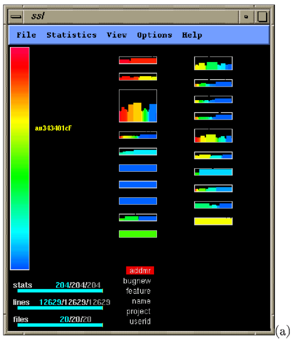
Pixel Representation



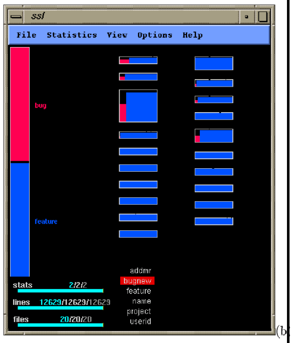
Textual order

Pixels within a line are ordered by color

File Summary: Statistics

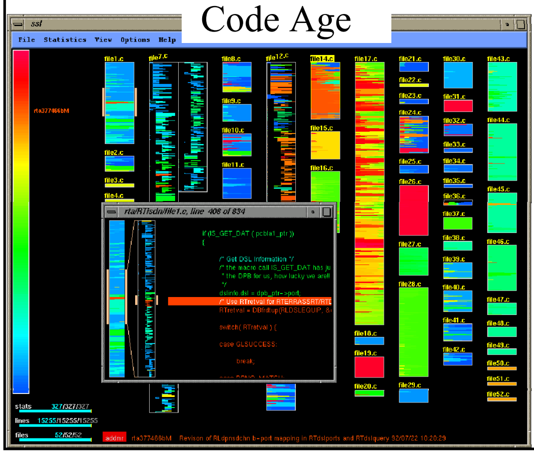


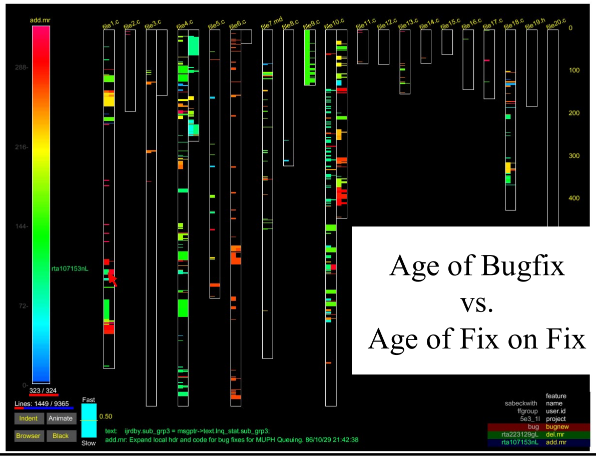
Code Age



Bug / Feature Distribution

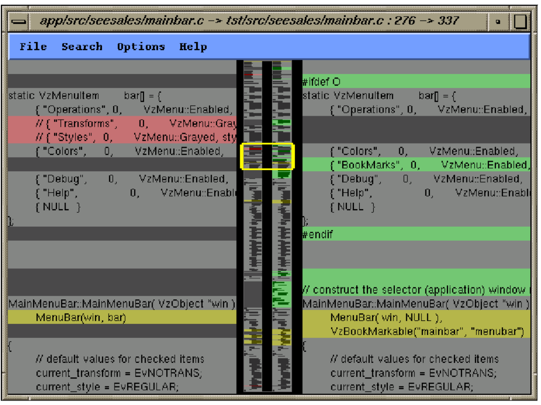
Code Age





Age of Bugfix
vs.
Age of Fix on Fix

Version Differences

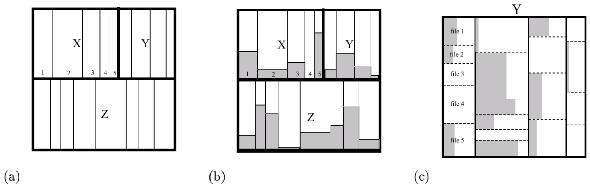

app/src/seesales/mainbar.c -> tst/src/seesales/mainbar.c : 276 -> 337
File Search Options Help

```
static VzMenuItem       bar[] = {
        { "Operations", 0,      VzMenu::Enabled,
        // { "Transforms",      0,      VzMenu::Gray
        // { "Styles",  0,      VzMenu::Grayed, sty
        { "Colors",     0,      VzMenu::Enabled,

        { "Debug",      0,      VzMenu::Enabled,
        { "Help",               0,      VzMenu::Enabl
        { NULL  }
};



MainMenuBar::MainMenuBar( VzObject *win )
        MenuBar(win, bar)

{
        // default values for checked items
        current_transform = EvNOTRANS;
        current_style = EvREGULAR;
```

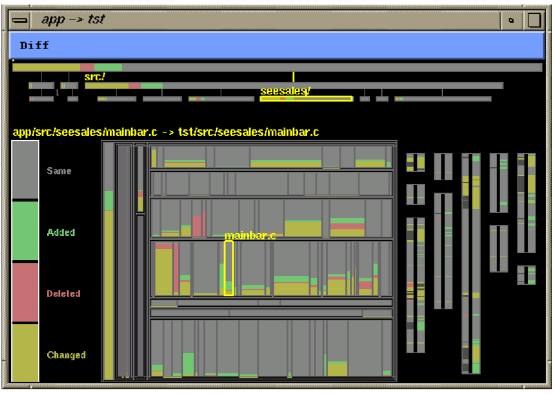
```
#ifdef O
static VzMenuItem       bar[] = {
        { "Operations", 0,      VzMenu::Enabled,


        { "Colors",     0,      VzMenu::Enabled,
        { "BookMarks",  0,      VzMenu::Enabled,
        { "Debug",      0,      VzMenu::Enabled,
        { "Help",               0,      VzMenu::Enabl
        { NULL  }
};
#endif


// construct the selector (application) window
MainMenuBar::MainMenuBar( VzObject *win )
        MenuBar( win, NULL ),
        VzBookMarkable("mainbar", "menubar")
{
        // default values for checked items
        current_transform = EvNOTRANS;
        current_style = EvREGULAR;
```

(a)

Hierarchical Representation

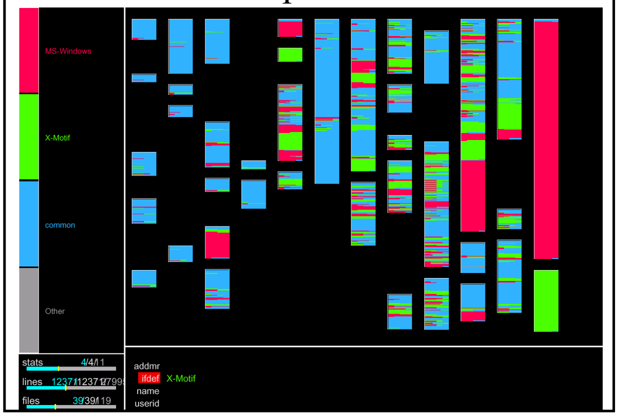


Version Differences of a Directory

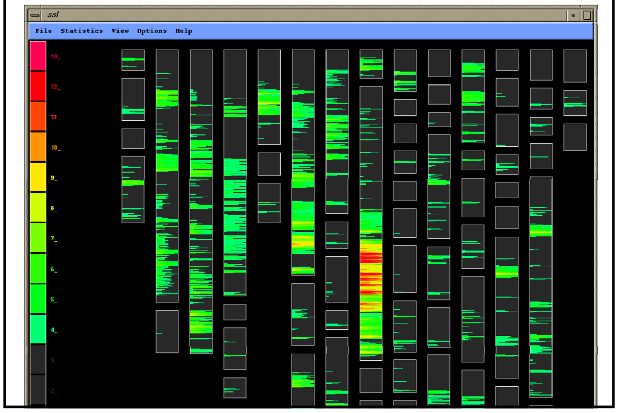


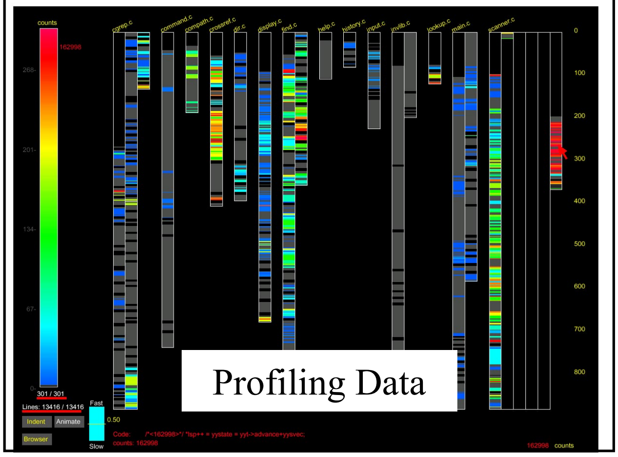
(b)

Version-specific Code



Depth of nested blocks





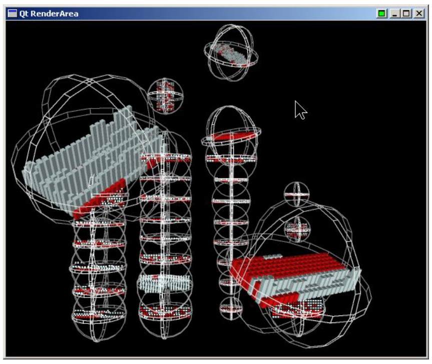
Code: /*<162998>*/ *lsp++ = yystate = yyt->advance+yysvec;
counts: 162998

Profiling Data

Animation

- Representations of the same aspects of different development states of the systems can be combined into an animation.
- Animations visualize the evolution of the system

Addding a third dimension: sv3D



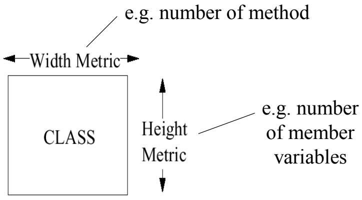
Under development by Adrian Marcus, Kent State University

The Evolution Matrix

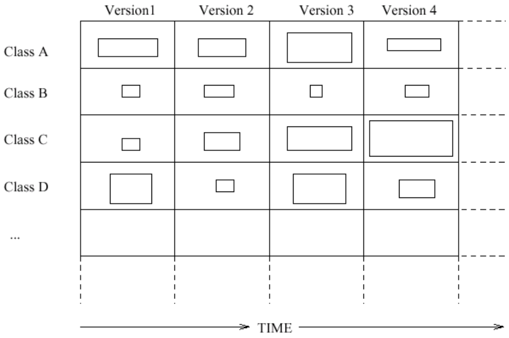
- Visualization of the Evolution of an Object-Oriented System, i.e. a set of classes.

The Evolution Matrix: Recovering Software Evolution using Software Visualization Techniques; Michele Lanza

Representation of a Class



Representation of the Evolution of several Classes



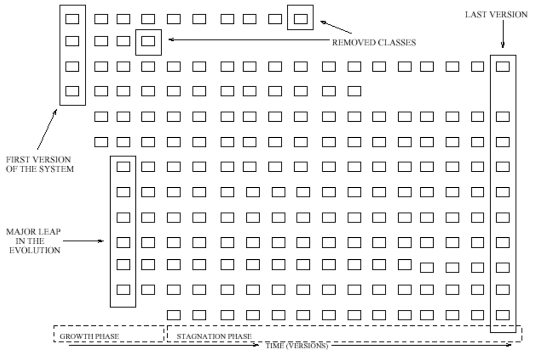
Observations about the whole System

- System Size
  - Adding and Removing Classes

⬇

  - Phases of Growth, Stagnation and Shrinking

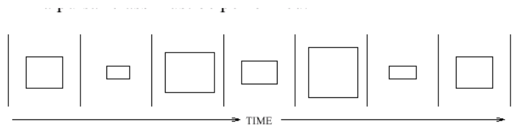
Properties of the System



Categorizing single Classes

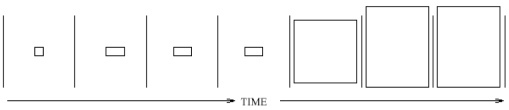
- Pulsar
- Supernova
- White Dwarf
- Red Gigant
- Dayfly
- Persistent
- Stagnant

Pulsar



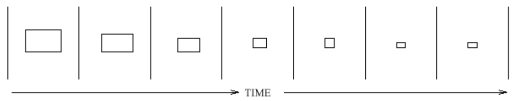
- Extending the functionality leads to increase of size
- Restructuring decreases size of the class
- This class is in the center of the development

Supernova



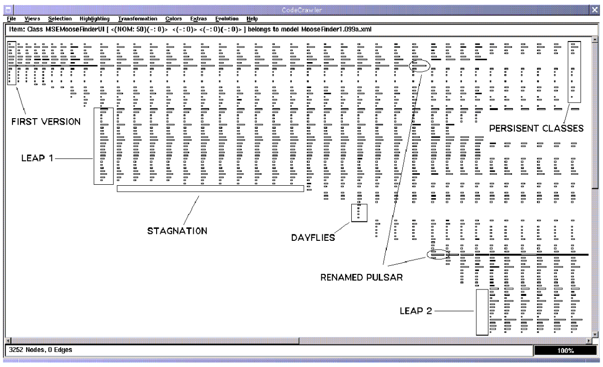
- sudden increase of size
- can be the consequence of a refactoring of the system
- pure data class, e.g. defines lots of constants, has a simple structure
- class was defined before, but implementation was just added
- can be a sign for problems with the design

White Dwarf



- Class might be obsolete

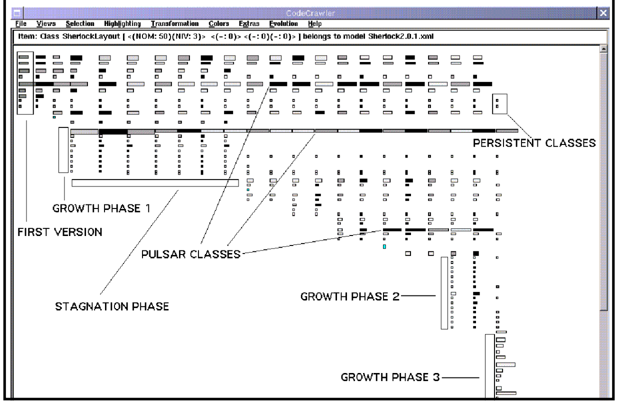
Example



Item: Class MSEMooseFinderUI [<(NOM: 50)(-: 0)> <(-: 0)> <(-: 0)(-: 0)>] belongs to model MooseFinder1.099a.xml

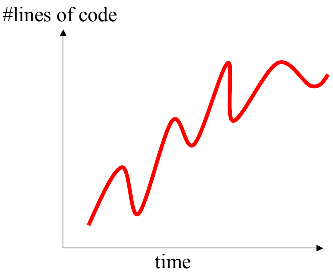
3252 Nodes, 0 Edges

Example

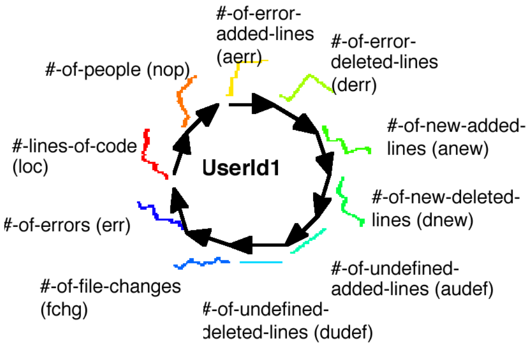


Item: Class SherlockLayout [<(NOM: 50)(NIV: 3)> <(-: 0)> <(-: 0)(-: 0)>] belongs to model Sherlock2.0.1.xml

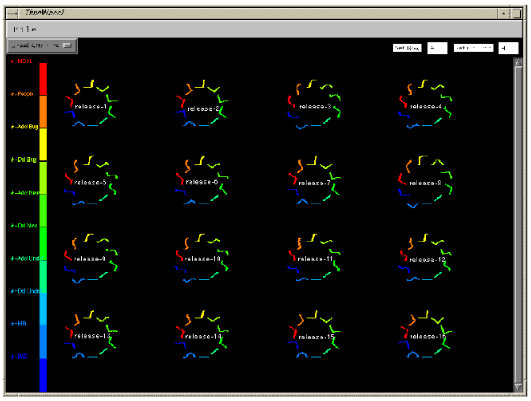
Skalar metrics over time



Time Wheel


#-of-error-added-lines (aerr)
#-of-error-deleted-lines (derr)
#-of-people (nop)
#-of-new-added-lines (anew)
#-lines-of-code (loc)
UserId1
#-of-new-deleted-lines (dnew)
#-of-errors (err)
#-of-undefined-added-lines (audef)
#-of-file-changes (fchg)
#-of-undefined-deleted-lines (dudef)

Time Wheel



Evolution of Software Systems

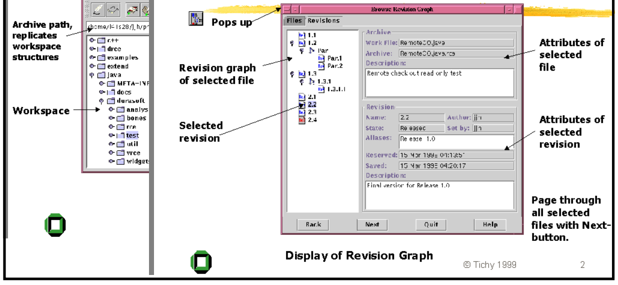
- Keeping track of versions and changes

→Configuration Management Systems

  - E.g. RCS and CVS

- The software archive contains the history of the system
- Other tools keep track of additional information, e.g. bug databases

VRCE: The Visual Revision Control Engine



Archive path, replicates workspace structures

Workspace

Pops up

Revision graph of selected file

Selected revision

Attributes of selected file

Attributes of selected revision

Page through all selected files with Next-button.

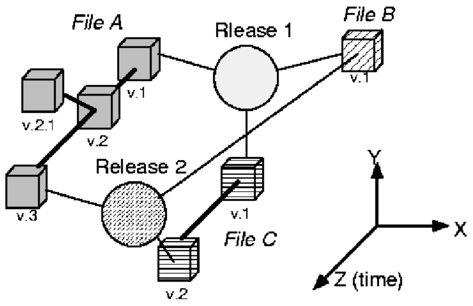
Archive
Work File: RemoteCO.java
Archive: RemoteCO.java,rcs
Description: Remote check out read only test

Revision
Name: 2.2 Author: jjh
State: Released Set by: jjh
Aliases: Release 1.0
Reserved: 15 Mar 1999 04:13:57
Saved: 15 Mar 1999 04:20:17
Description: Final version for Release 1.0

Display of Revision Graph

© Tichy 1999 2

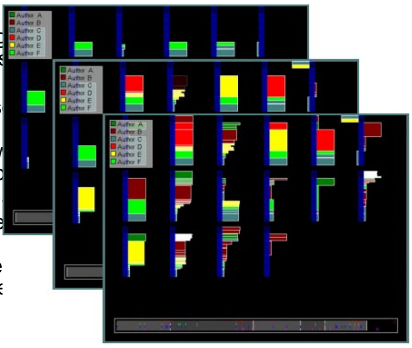
VRCS



H.Koike, H-C Chu, "VRCS: Integrating Version Control and Module Management using Interactive Three-Dimensional Graphics". *Proceedings of 1997, IEEE Symposium on Visual Languages*, Capri, Italy, 1997.

Revision Towers

- Tool for visualizing
of a RCS or CVS s
repository.
- Each block repres
files (header and
implementation), v
and coloured acco
various properties
- The process is the
making it possible
various files, or ve
became part of the

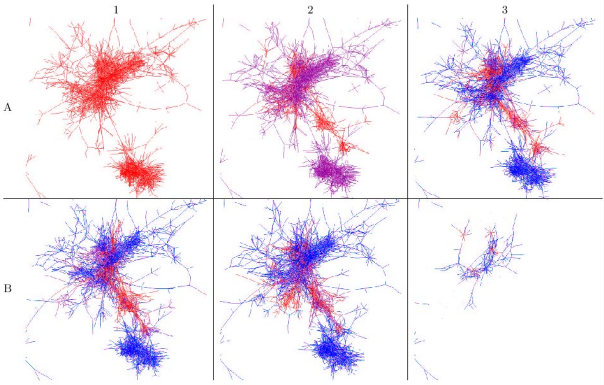


Revision Towers, Christopher M. B. Taylor and Malcolm Munro, Proceedings of the Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2002) Paris, June 26-27, 2002

GEVOL

- Uses Force-Directed Layout to draw graphs of Java programs
  - Call Graph
  - Control-Flow Graph
  - Inheritance Graph
- Color encodes age
  - colored in color of user who did the change
    ⟨🟥, 🟥, 🟪, 🟪, 🟦⟩
    ⟨🟨, 🟨, ⬜, 🟦, 🟦⟩
  - Aging => progression from user's color to blue
- Animation shows subsequent graphs
  - one graph per day
  - uses linear interpolation for smooth transitions.

GEVOL: Call Graph



Summary

- Different techniques for visualizing the change of the structure, metrics and source code of a system over time.
- There is much more information in software archives than current tools exploit, because they leave the exploration and analysis to the user.