Software Visualization

CS 4460

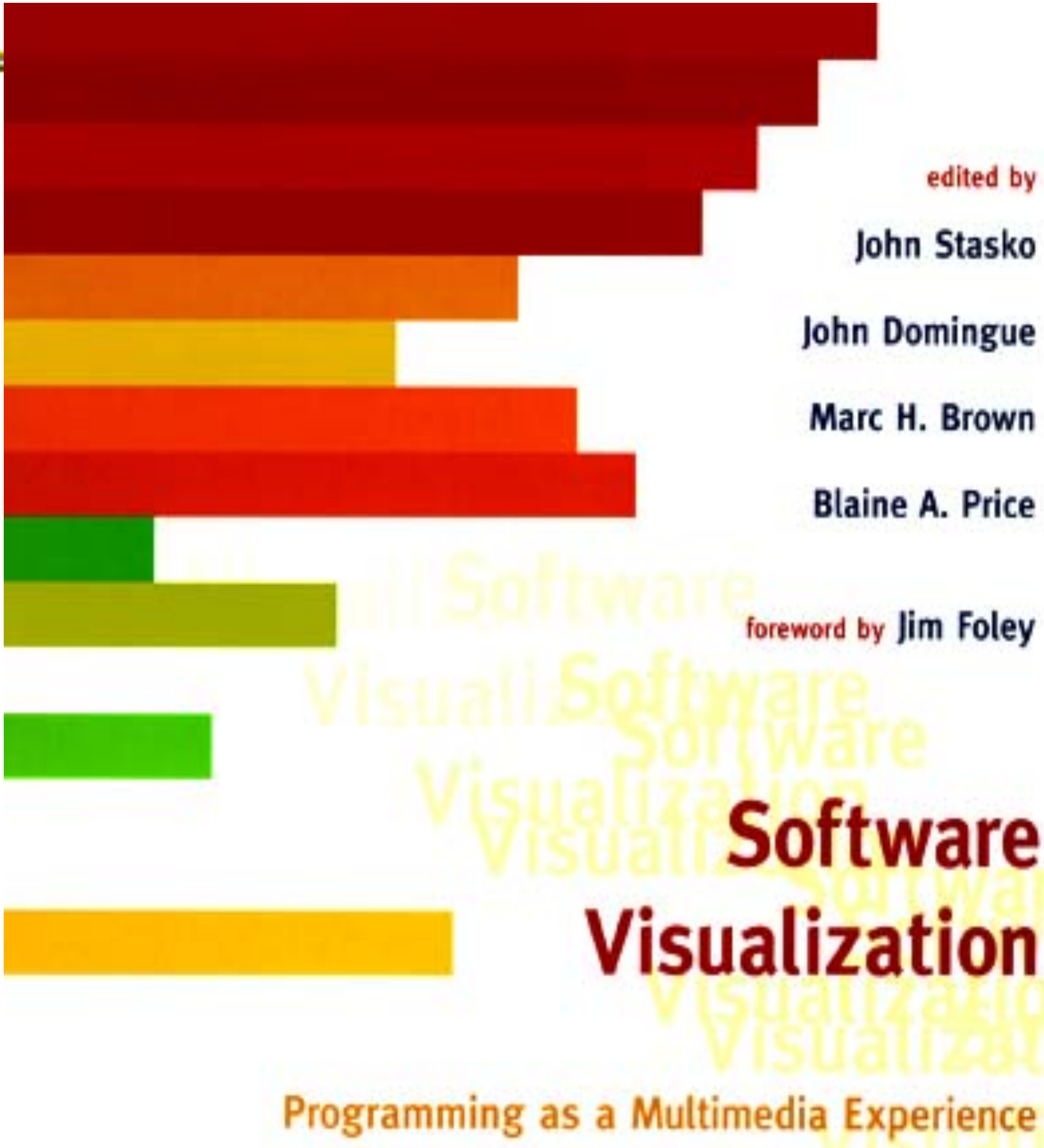
Last Revision: November 2016

Software Visualization

Definition

"The use of the crafts of typography, graphic design, animation, and cinematography with modern human-computer interaction and computer graphics technology to facilitate both the human understanding and effective use of computer software."

Price, Baecker and Small, '98



Subdomains

- Two main subareas of software visualization
  - Program visualization
    - Use of visualization to help programmers, coders, developers
    - Software engineering focus
  - Algorithm visualization
    - Use of visualization to help teach algorithms and data structures
    - Pedagogy focus

Caveat

- This is a HUGH area
- Presentation goal: provide flavor of kinds of techniques and systems that have been created
  - Lots of screen shots
  - Some videos

Program Visualization

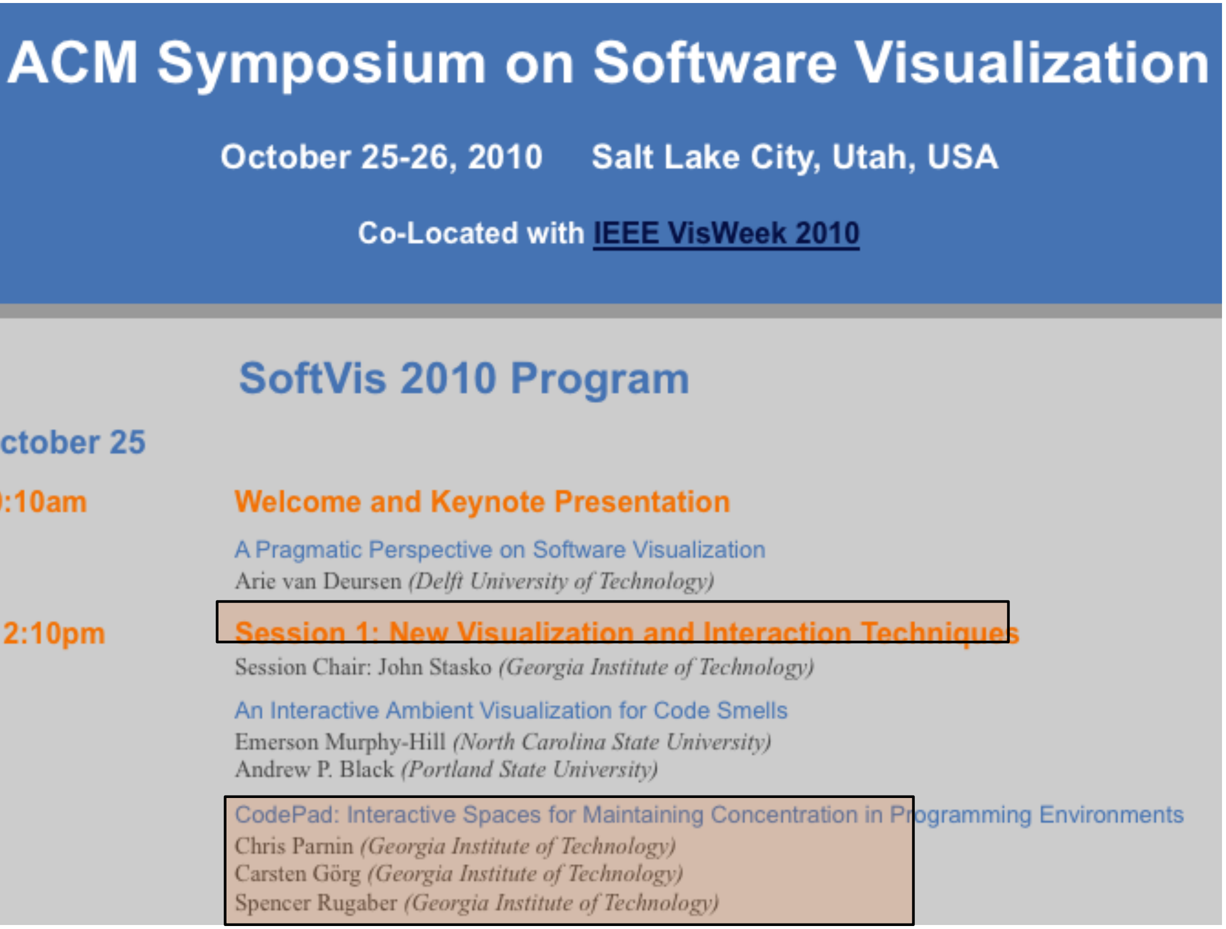
- Can be as simple as enhanced views of program source
- Can be as complex as views of the execution of a highly parallel program, its data structures, run-time heap, etc.

PV is a big research Area

VISSOFT 2014

2nd IEEE Working Conference on Software Visualization

Sep 29 - 30, 2014, Victoria, CA



ACM Symposium on Software Visualization

October 25-26, 2010 Salt Lake City, Utah, USA

Co-Located with IEEE VisWeek 2010

SoftVis 2010 Program

ctober 25

:10am **Welcome and Keynote Presentation**

A Pragmatic Perspective on Software Visualization
Arie van Deursen *(Delft University of Technology)*

2:10pm **Session 1: New Visualization and Interaction Techniques**

Session Chair: John Stasko *(Georgia Institute of Technology)*

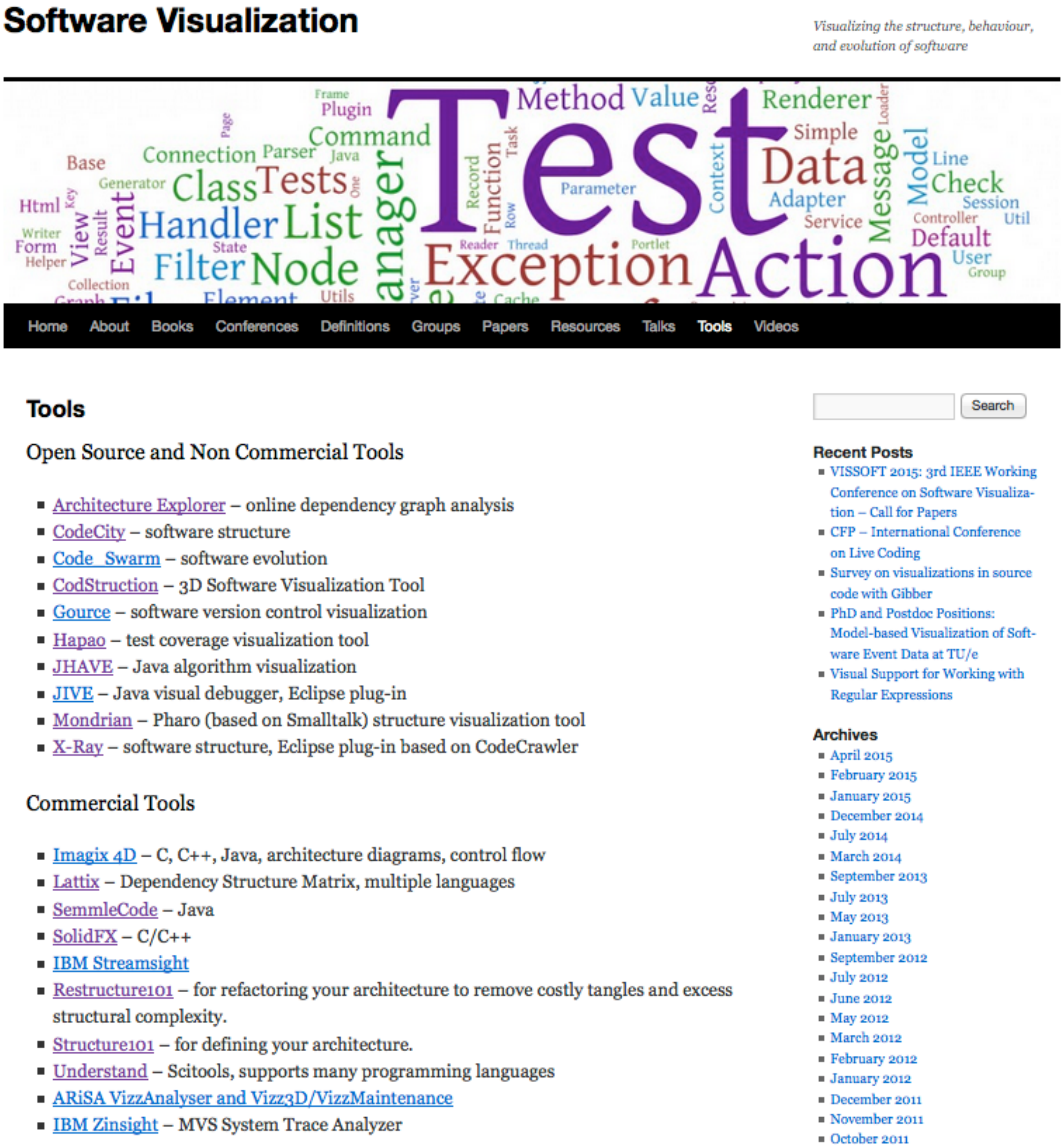
An Interactive Ambient Visualization for Code Smells
Emerson Murphy-Hill *(North Carolina State University)*
Andrew P. Black *(Portland State University)*

CodePad: Interactive Spaces for Maintaining Concentration in Programming Environments
Chris Parnin *(Georgia Institute of Technology)*
Carsten Görg *(Georgia Institute of Technology)*
Spencer Rugaber *(Georgia Institute of Technology)*

PV is a Big Product Category

- Do Google searches on
  - Program visualization
  - Code visualization
  - Software visualization
- Lots of companies/products
- List of some tools at http://softvis.wordpress.com/tools/
- https://scitools.com/feature-category/graphing/

https://softvis.wordpress.com/tools/



Software Visualization

Visualizing the structure, behaviour, and evolution of software

Home About Books Conferences Definitions Groups Papers Resources Talks Tools Videos

Tools

Open Source and Non Commercial Tools

- Architecture Explorer – online dependency graph analysis
- CodeCity – software structure
- Code_Swarm – software evolution
- CodStruction – 3D Software Visualization Tool
- Gource – software version control visualization
- Hapao – test coverage visualization tool
- JHAVE – Java algorithm visualization
- JIVE – Java visual debugger, Eclipse plug-in
- Mondrian – Pharo (based on Smalltalk) structure visualization tool
- X-Ray – software structure, Eclipse plug-in based on CodeCrawler

Commercial Tools

- Imagix 4D – C, C++, Java, architecture diagrams, control flow
- Lattix – Dependency Structure Matrix, multiple languages
- SemmleCode – Java
- SolidFX – C/C++
- IBM Streamsight
- Restructure101 – for refactoring your architecture to remove costly tangles and excess structural complexity.
- Structure101 – for defining your architecture.
- Understand – Scitools, supports many programming languages
- ARiSA VizzAnalyser and Vizz3D/VizzMaintenance
- IBM Zinsight – MVS System Trace Analyzer

Search

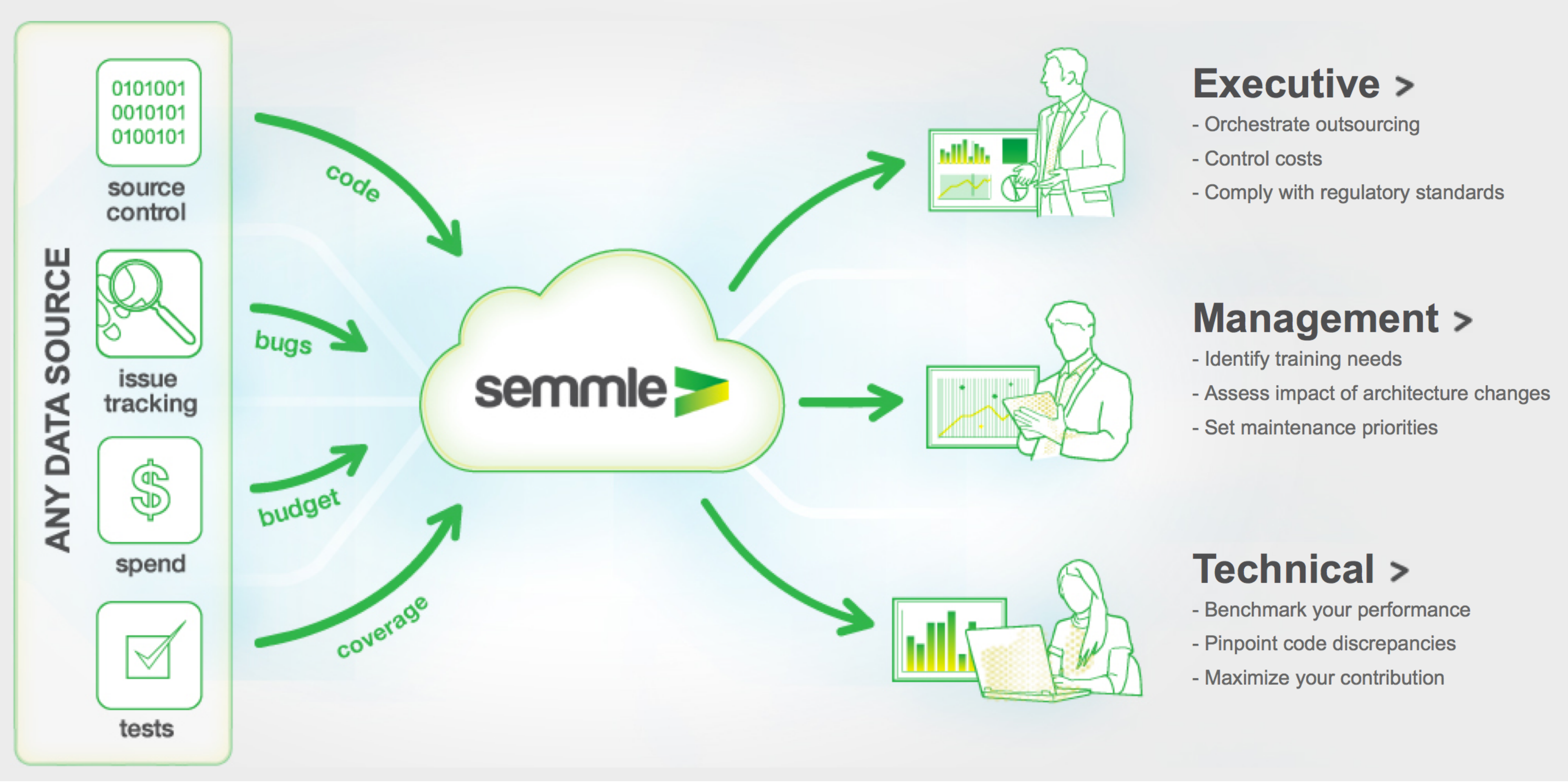
Recent Posts

- VISSOFT 2015: 3rd IEEE Working Conference on Software Visualization – Call for Papers
- CFP – International Conference on Live Coding
- Survey on visualizations in source code with Gibber
- PhD and Postdoc Positions: Model-based Visualization of Software Event Data at TU/e
- Visual Support for Working with Regular Expressions

Archives

- April 2015
- February 2015
- January 2015
- December 2014
- July 2014
- March 2014
- September 2013
- July 2013
- May 2013
- January 2013
- September 2012
- July 2012
- June 2012
- May 2012
- March 2012
- February 2012
- January 2012
- December 2011
- November 2011
- October 2011

http://semmle.com/solutions/



Executive >

- Orchestrate outsourcing
- Control costs
- Comply with regulatory standards

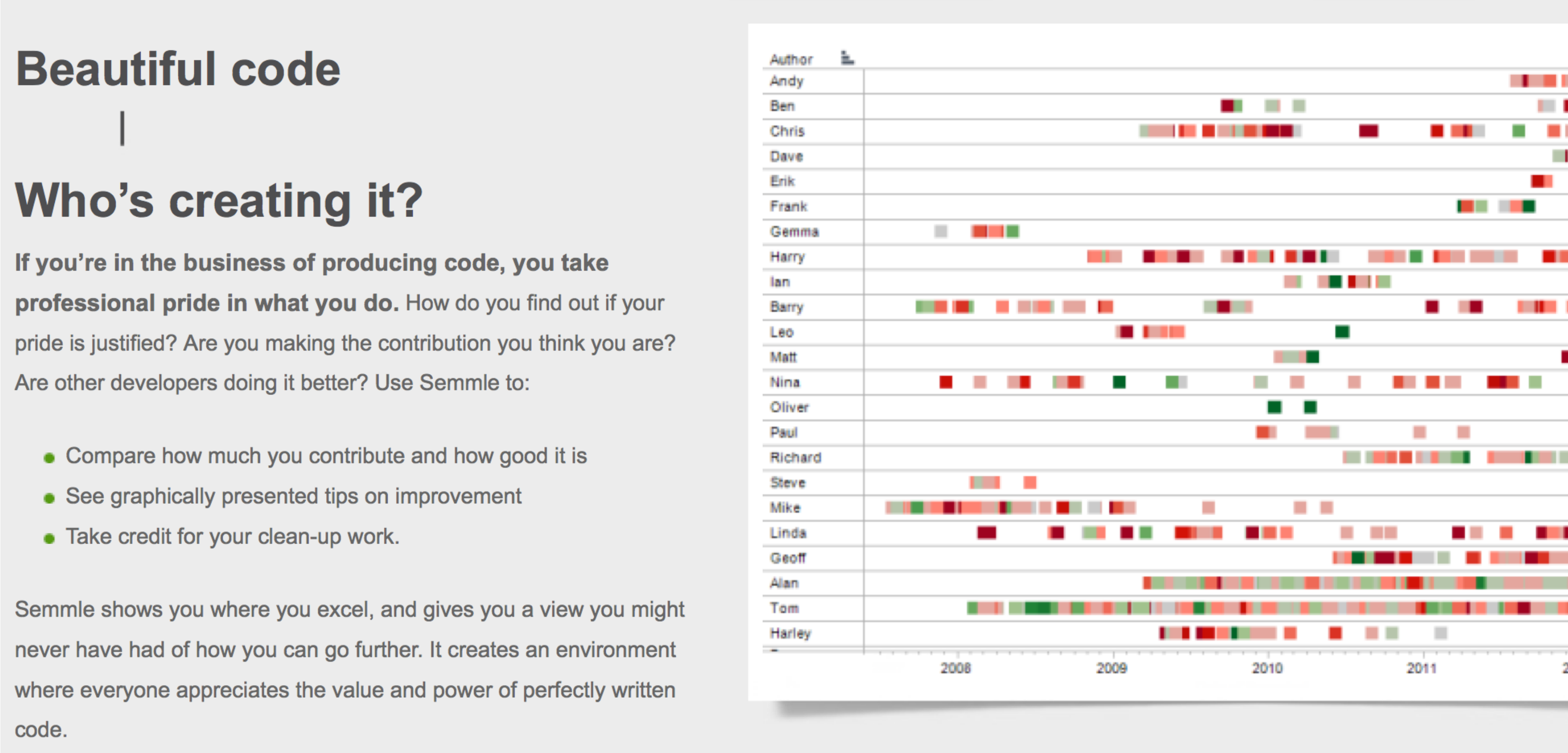
Management >

- Identify training needs
- Assess impact of architecture changes
- Set maintenance priorities

Technical >

- Benchmark your performance
- Pinpoint code discrepancies
- Maximize your contribution

http://semmle.com/solutions/technical/#tech-keypoint1



Beautiful code

Who’s creating it?

If you’re in the business of producing code, you take professional pride in what you do. How do you find out if your pride is justified? Are you making the contribution you think you are? Are other developers doing it better? Use Semmle to:

- Compare how much you contribute and how good it is
- See graphically presented tips on improvement
- Take credit for your clean-up work.

Semmle shows you where you excel, and gives you a view you might never have had of how you can go further. It creates an environment where everyone appreciates the value and power of perfectly written code.

What Might We Visualize?

1. Structure of code base
   - Static – not execution time

2. Dynamic (run-time) behavior of code base

3. Software development team & code base dynamics
   - Code repository structure/evolution
   - Program team communications patterns

Talk to your neighbor ☺

Static Structure of Code Base

- Call graphs
- Object class hierarchy
- Scope of variables
- Code module sizes
- ??
- ??

Execution Data

Summaries, such as

- Total running time
- Number of times a method was called
- Amount of time CPU was idle
- Bytes read/written
- Memory high-water mark
- Etc etc etc

Details, such as

- Memory allocations
- System calls
- Cache misses
- Page faults
- Pipeline flushes
- Process scheduling
- Completion of disk reads or writes
- Message receipt
- Application phases
- Etc etc etc

Test Results

- Hundreds, maybe thousands of tests
- For each test:
  - Purpose
  - Result (pass or fail)

    Could be per-configuration or per-version
  - Relevant parts of the code

Really Detailed Execution Data

- Logging virtual machines can capture *everything*
  - Enough data to replay program execution and recreate the entire machine state at any point in time
  - Allows “time-traveling”
  - For long running systems, data could span months
- Uses:
  - Debugging
  - Understanding attacks

Team and Code Base Evolution/Dynamics

- Who wrote code
- Who modified code
- History of code modifications (ala Wikipedia)
- Programmer productivity
- Programmer bug frequency
- Programmer interactions
- ??
- ??

VISSOFT 2014 Program

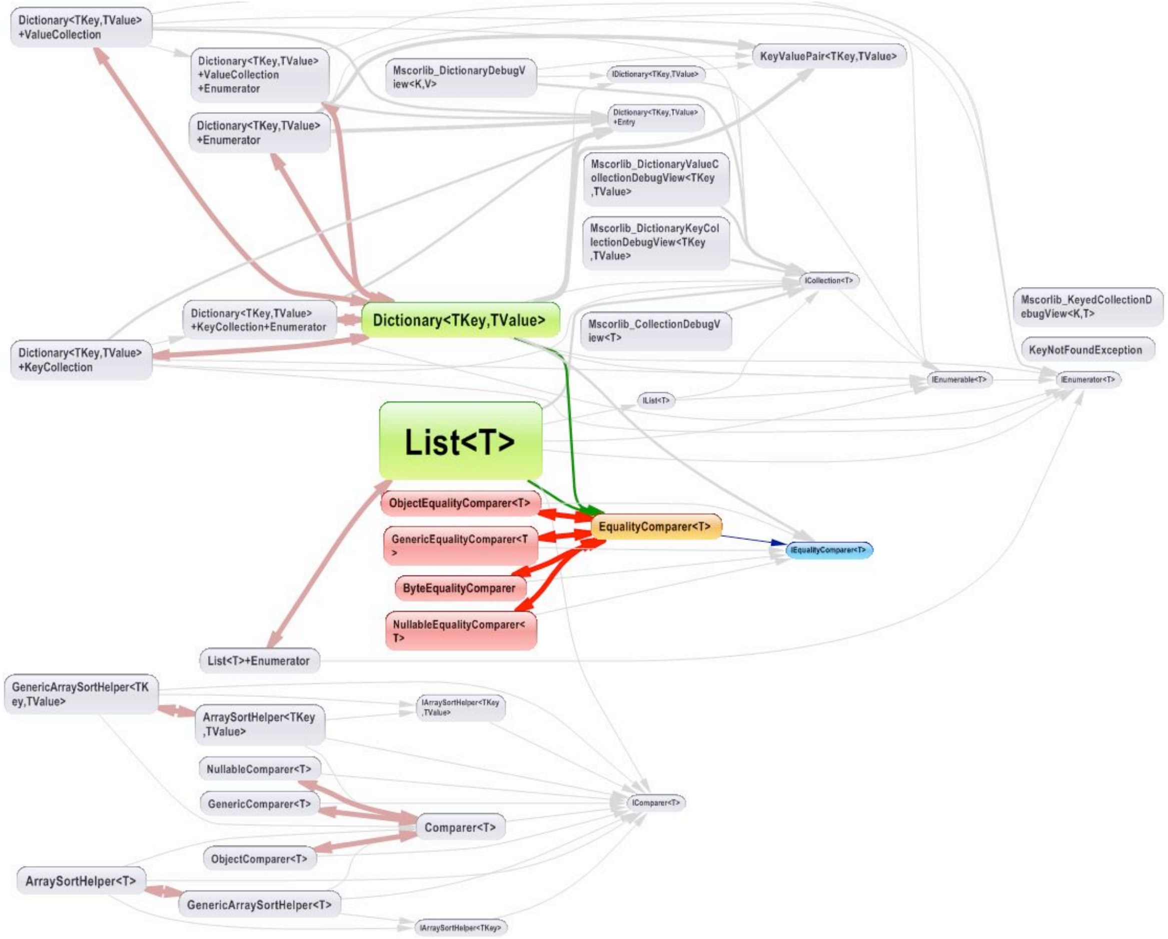
- Combining Tiled and Textual Views of Code
- Integrating Anomaly Diagnosis Techniques into Spreadsheet Environments
- Action-Based Visualization
- Slicing-Based Techniques for Visualizing Large Metamodels
- Search Space Pruning Constraints Visualization
- Live coding the SynthKit: Little Bits as an Embodied Programming Language
- A Domain-Specific Language for Visualizing Software Dependencies as a Graph
- Feature Relations Graphs: A Visualisation Paradigm for Feature Constraints in Software Product Lines
- Validation of Software Visualization Tools: A Systematic Mapping Study
- Using a Task-Oriented Framework to Characterize Visualization Approaches
- Mr. Clean: A Tool for Tracking and Comparing the Lineage of Scientific Visualization Code
- Visual Clone Analysis with SolidSDD
- Polyptychon: A Hierarchically-Constrained Classified Dependencies Visualization
- How Developers Visualize Compiler Messages: A Foundational Approach to Notification Construction
- Lightweight Structured Visualization of Assembler Control Flow Based on Regular Expressions
- Templated Visualization of Object State with Vebugger
- The Challenge of Helping the Programmer during Debugging
- ChronoTwigger: A Visual Analytics Tool for Understanding Source and Test Co-evolution
- Visualizing the Evolution of Systems and Their Library Dependencies
- AniMatrix: A Matrix-Based Visualization of Software Evolution
- Visualizing Developer Interactions
- Information Visualization for Agile Software Development
- FAVe: Visualizing User Feedback for Software Evolution

SoftVis 2010 Program

- An Interactive Ambient Visualization for Code Smells
- CodePad: Interactive Spaces for Maintaining Concentration in Programming Environments
- User Evaluation of Polymetric Views Using a Large Visualization Wall
- Software Evolution Storylines
  AllocRay: Memory Allocation Visualization for Unmanaged Languages
- Heapviz: Interactive Heap Visualization for Program Understanding and Debugging
- A Map of the Heap: Revealing Design Abstractions in Runtime Structures
- Trevis: A Context Tree Visualization & Analysis Framework and Its Use for Classifying Performance Failure Reports
- Exploring the Inventor's Paradox: Applying Jigsaw to Software Visualization
- Dependence Cluster Visualization
- Towards Anomaly Comprehension: Using Structural Compression to Navigate Profiling Call-Trees
- Embedding Spatial Software Visualization in the IDE: An Exploratory Study
- 3D Kiviat Diagrams for the Interactive Analysis of Software Metric Trends
- Graph Works - Pilot Graph Theory Visualization Tool
- Visualizing Software Entities Using a Matrix Layout
- ImpactViz: Visualizing Class Dependencies and the Impact of Changes in Software Revisions
- VIPERS: Visual Prototyping Environment for Real-Time Imaging Systems
- Towards Automated Analysis and Visualization of Distributed Software Systems
- TIE: An Interactive Visualization of Thread Interleavings
- GEM: Graphical Explorer of MPI Programs
- Fault Forest Visualization
- Visualizing Windows System Traces
- Understanding Complex Multithreaded Software Systems by Using Trace Visualization
- Zinsight: A Visual and Analytic Environment for Exploring Large Event Traces
- Jype - A Program Visualization and Programming Exercise Tool for Python Off-Screen Visualization Techniques for Class Diagrams
- An Automatic Layout Algorithm for BPEL Processes
- Visual Comparison of Software Architectures
- Representing Development History in Software Cities
- Frank xDIVA: Automatic Animation Between Debugging Break Points
- Understanding Relaxed Memory Consistency Through Interactive Visualization
  the Execution of Object Orientated Concurrent Programs

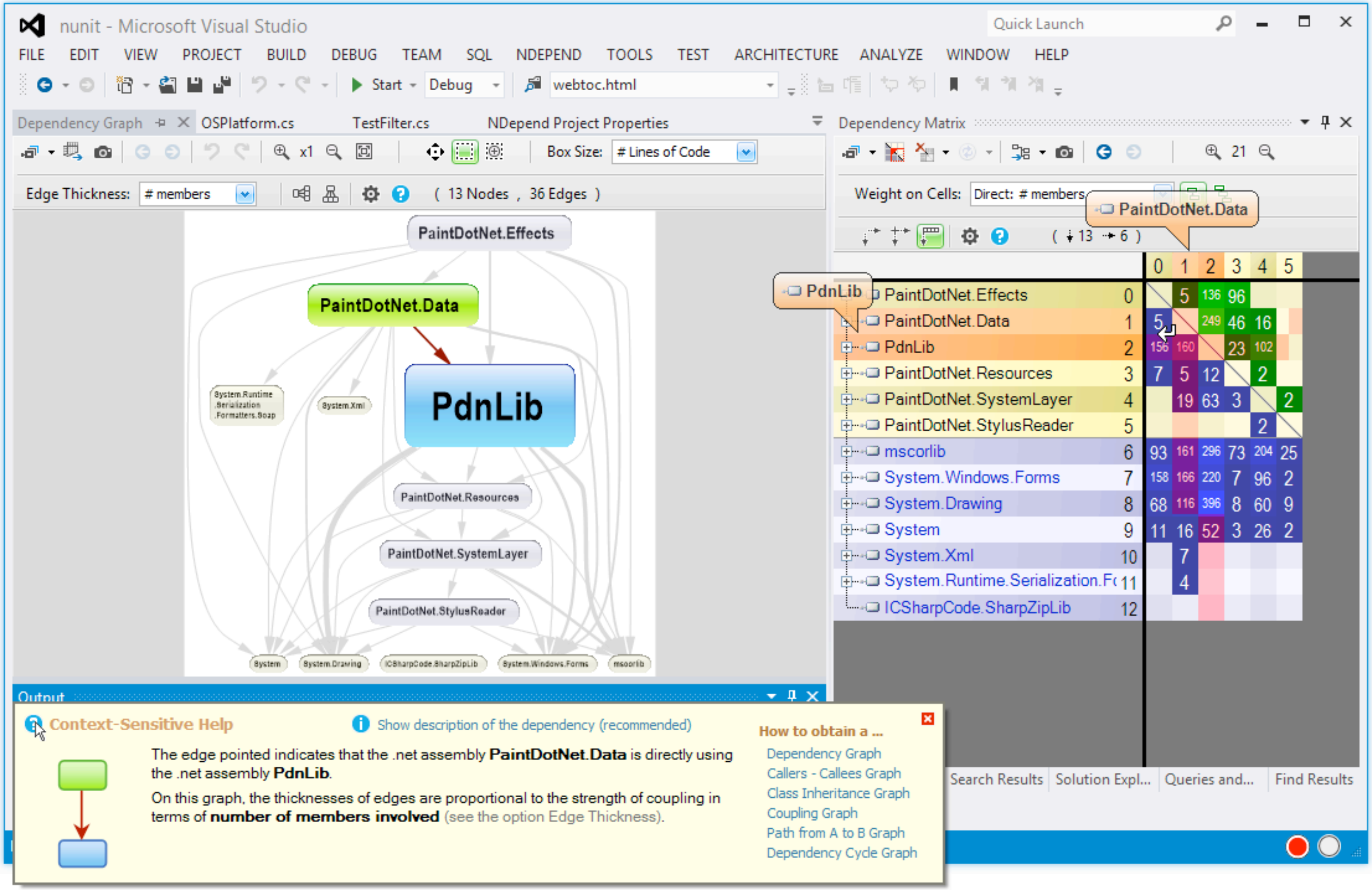
Commercial System Screen Shots

Dependency Graph



Ndepend commercial product;
http://www.ndepend.com/SampleReports/OnDb4o/NDependReport.html#/?screen=Main

Graph and Matrix Rep’n



Box Size: # Lines of Code

Edge Thickness: # members

(13 Nodes , 36 Edges)

Weight on Cells: Direct: # members

(↓13 →6)

		0	1	2	3	4	5
PaintDotNet.Effects	0		5	136	96		
PaintDotNet.Data	1	5		249	46	16	
PdnLib	2	156	160		23	102	
PaintDotNet.Resources	3	7	5	12		2	
PaintDotNet.SystemLayer	4		19	63	3		2
PaintDotNet.StylusReader	5					2	
mscorlib	6	93	161	296	73	204	25
System.Windows.Forms	7	158	166	220	7	96	2
System.Drawing	8	68	116	396	8	60	9
System	9	11	16	52	3	26	2
System.Xml	10		7				
System.Runtime.Serialization.F	11		4				
ICSharpCode.SharpZipLib	12						

PaintDotNet.Data

PdnLib

Context-Sensitive Help

Show description of the dependency (recommended)

The edge pointed indicates that the .net assembly **PaintDotNet.Data** is directly using the .net assembly **PdnLib**.

On this graph, the thicknesses of edges are proportional to the strength of coupling in terms of **number of members involved** (see the option Edge Thickness).

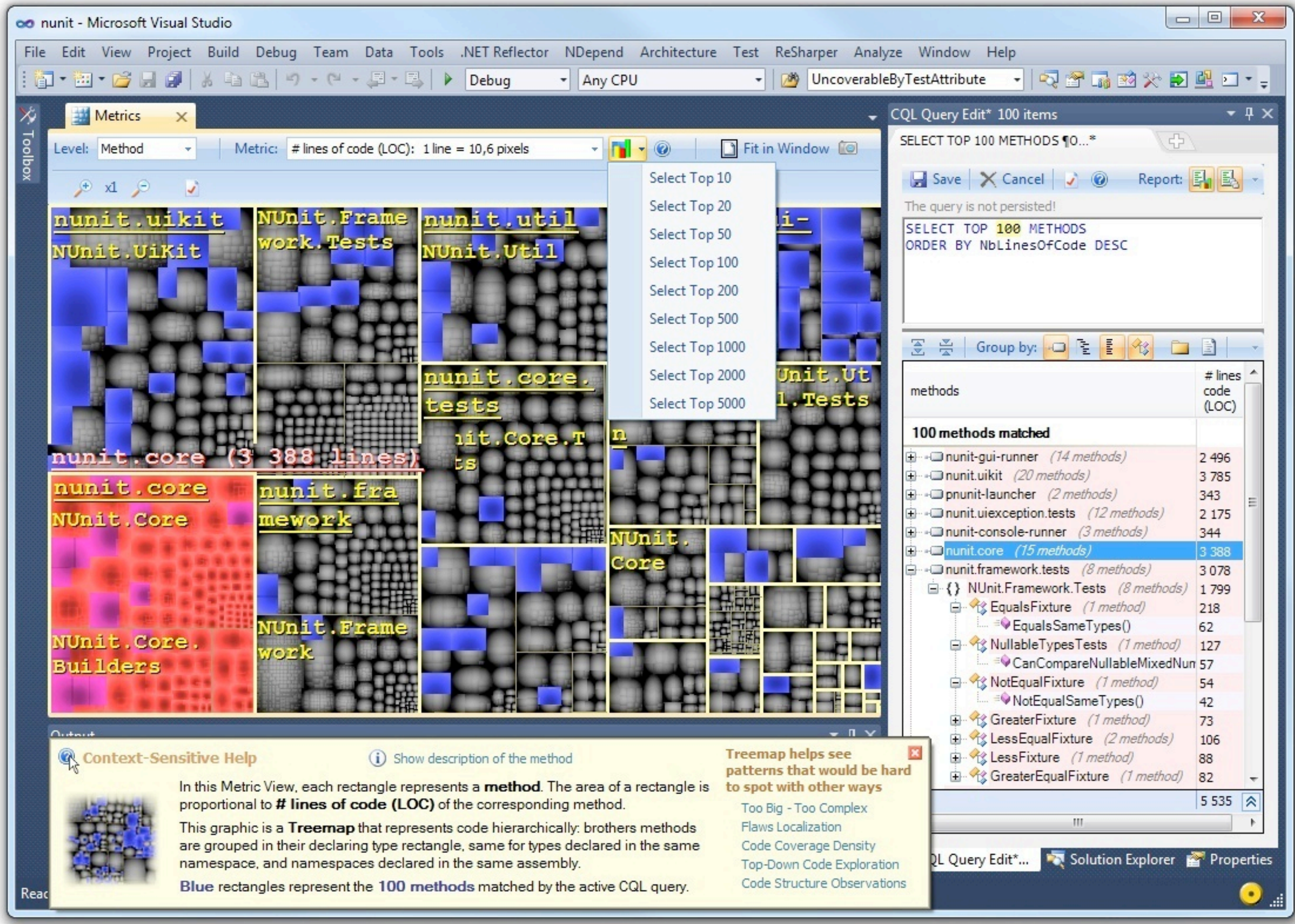
How to obtain a ...

- Dependency Graph
- Callers - Callees Graph
- Class Inheritance Graph
- Coupling Graph
- Path from A to B Graph
- Dependency Cycle Graph

Ndepend Video [http://www.ndepend.com/features/ - overview](http://www.ndepend.com/features/)

Code Hierarchy Treemap; Lines of Code

NDepend



The query is not persisted!

```
SELECT TOP 100 METHODS
ORDER BY NbLinesOfCode DESC
```

methods	# lines code (LOC)
100 methods matched	
nunit-gui-runner (14 methods)	2 496
nunit.uikit (20 methods)	3 785
pnunit-launcher (2 methods)	343
nunit.uiexception.tests (12 methods)	2 175
nunit-console-runner (3 methods)	344
nunit.core (15 methods)	3 388
nunit.framework.tests (8 methods)	3 078
{} NUnit.Framework.Tests (8 methods)	1 799
EqualsFixture (1 method)	218
EqualsSameTypes()	62
NullableTypesTests (1 method)	127
CanCompareNullableMixedNum	57
NotEqualFixture (1 method)	54
NotEqualSameTypes()	42
GreaterFixture (1 method)	73
LessEqualFixture (2 methods)	106
LessFixture (1 method)	88
GreaterEqualFixture (1 method)	82
	5 535

Context-Sensitive Help

Show description of the method

In this Metric View, each rectangle represents a **method**. The area of a rectangle is proportional to **# lines of code (LOC)** of the corresponding method.

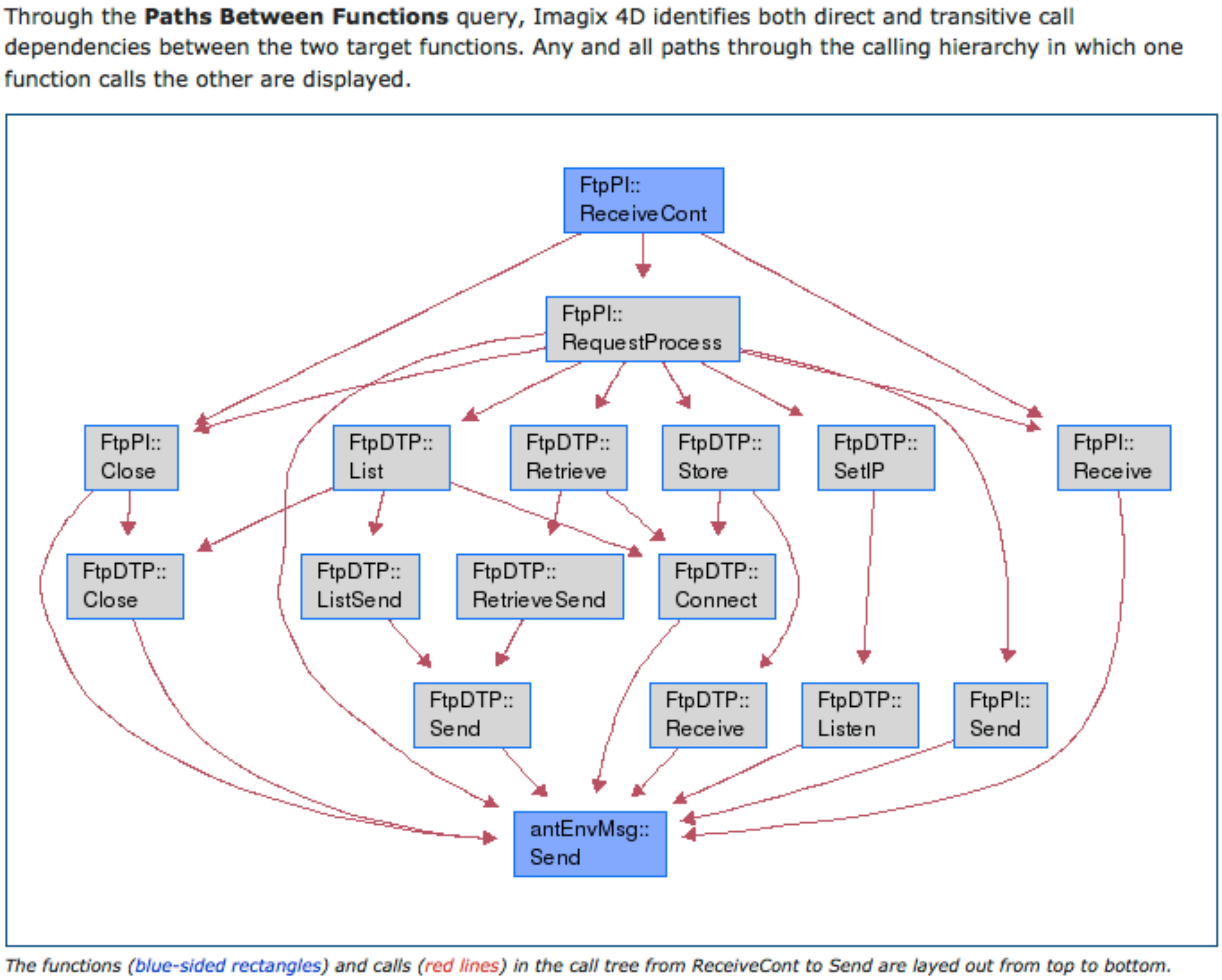
This graphic is a **Treemap** that represents code hierarchically: brothers methods are grouped in their declaring type rectangle, same for types declared in the same namespace, and namespaces declared in the same assembly.

Blue rectangles represent the **100 methods** matched by the active CQL query.

Treemap helps see patterns that would be hard to spot with other ways

Too Big - Too Complex
Flaws Localization
Code Coverage Density
Top-Down Code Exploration
Code Structure Observations

Imagix - http://www.imagix.com/



Through the **Paths Between Functions** query, Imagix 4D identifies both direct and transitive call dependencies between the two target functions. Any and all paths through the calling hierarchy in which one function calls the other are displayed.

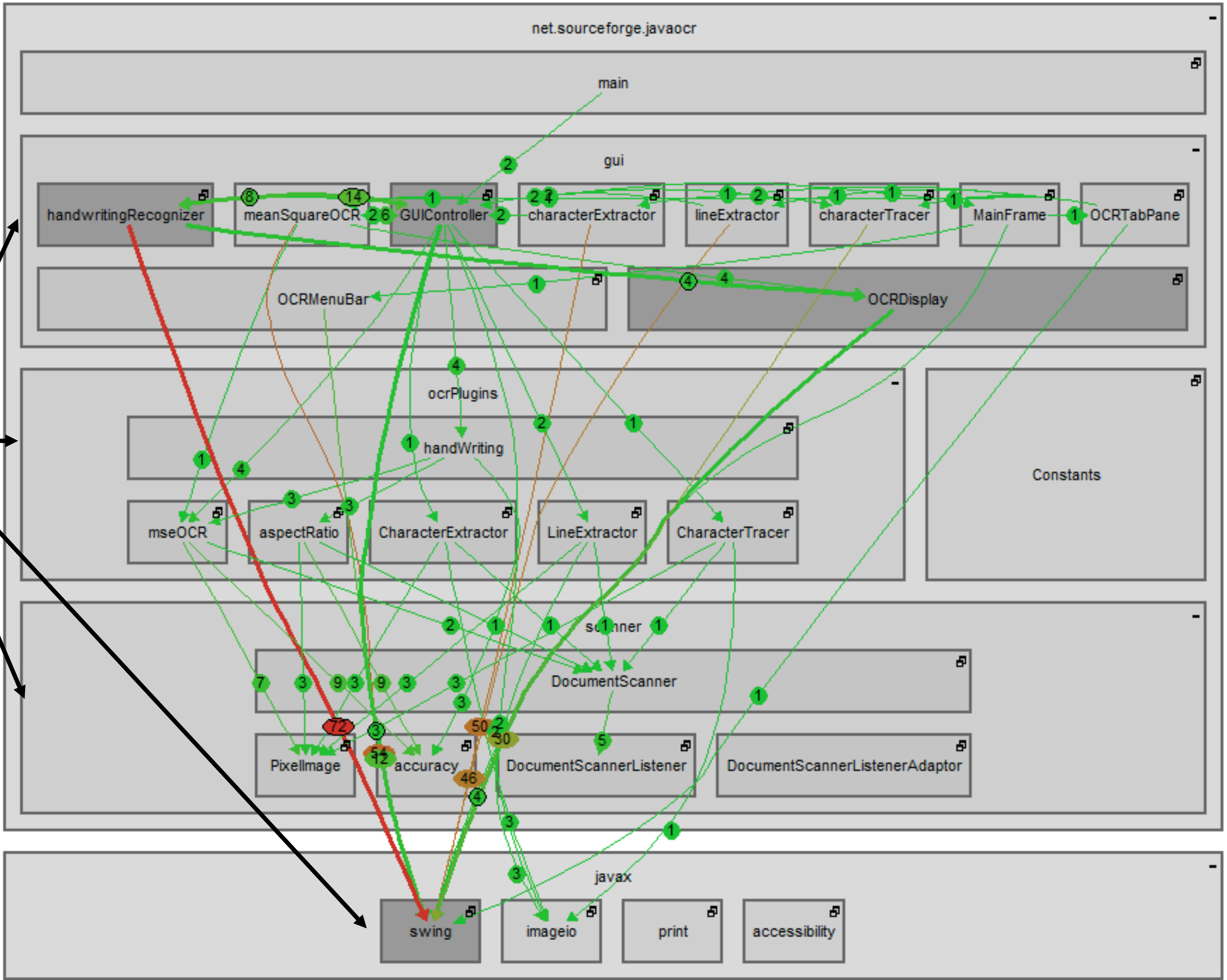
The functions (blue-sided rectangles) and calls (red lines) in the call tree from ReceiveCont to Send are layed out from top to bottom.

Imagix

Relationship strength among subsystems – shown by colors and numbers

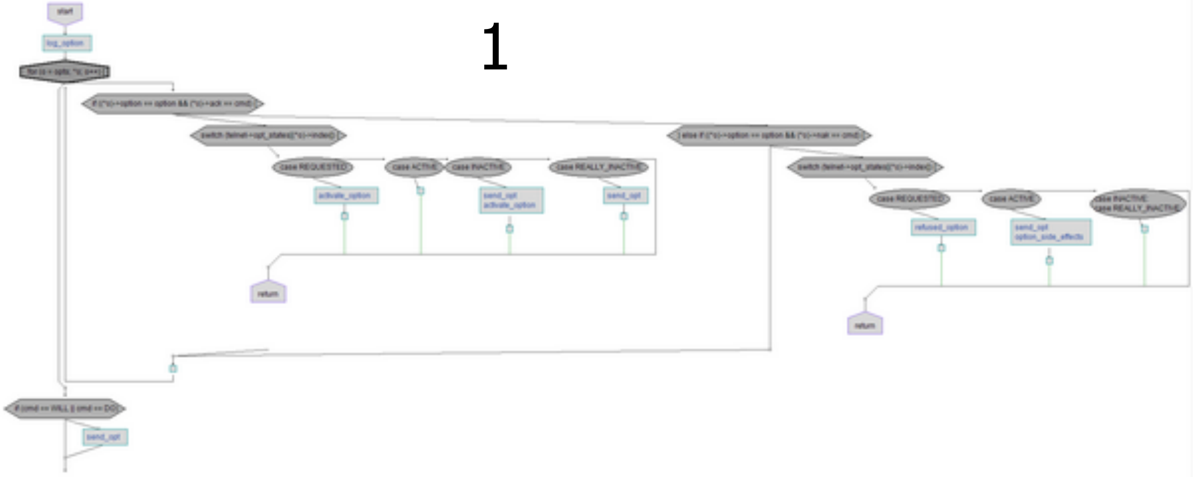
Semantic substrates?

Subsystems



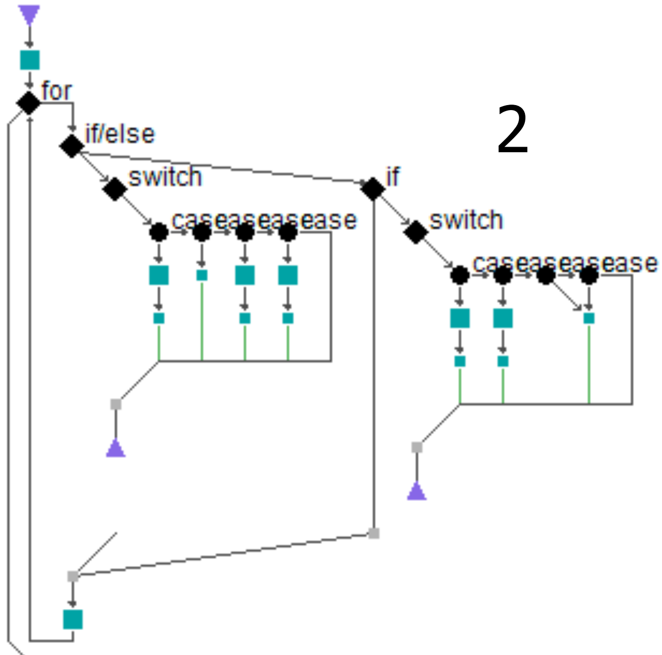
Imagix – Three Levels of Code Views

1

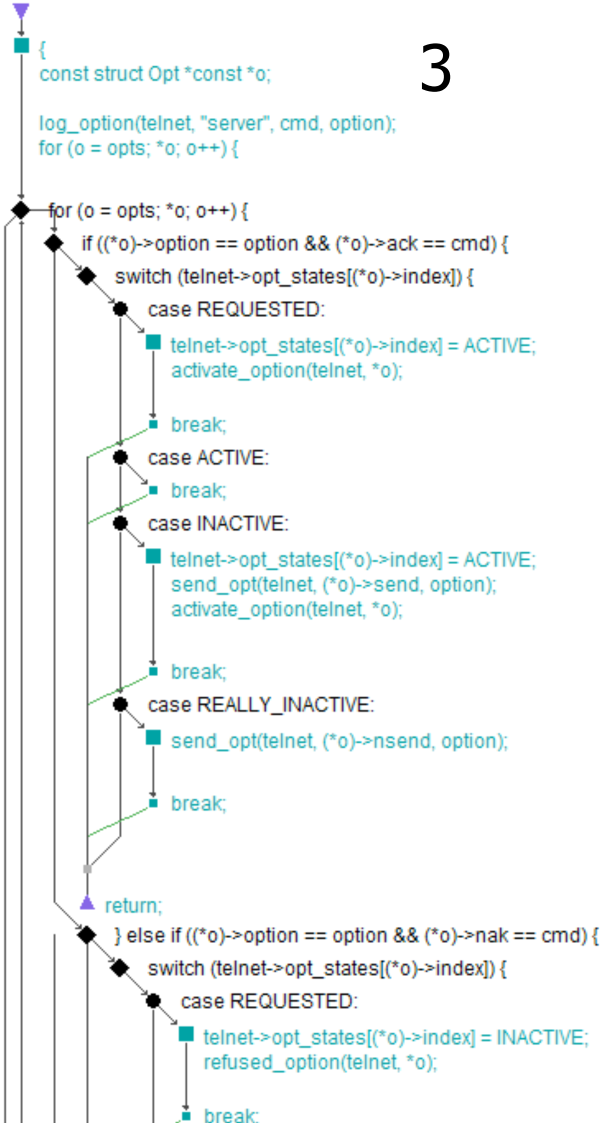


start
log_option
for (o = opts; *o; o++) {
if ((*o)->option == option && (*o)->ack == cmd) {
switch (telnet->opt_states[(*o)->index]) {
case REQUESTED
case ACTIVE
case INACTIVE
case REALLY_INACTIVE
activate_option
send_opt
activate_option
send_opt
return
} else if ((*o)->option == option && (*o)->nak == cmd) {
switch (telnet->opt_states[(*o)->index]) {
case REQUESTED
case ACTIVE
case INACTIVE
case REALLY_INACTIVE
refused_option
send_opt
option_side_effects
return
if (cmd == WILL || cmd == DO)
send_opt

2



3



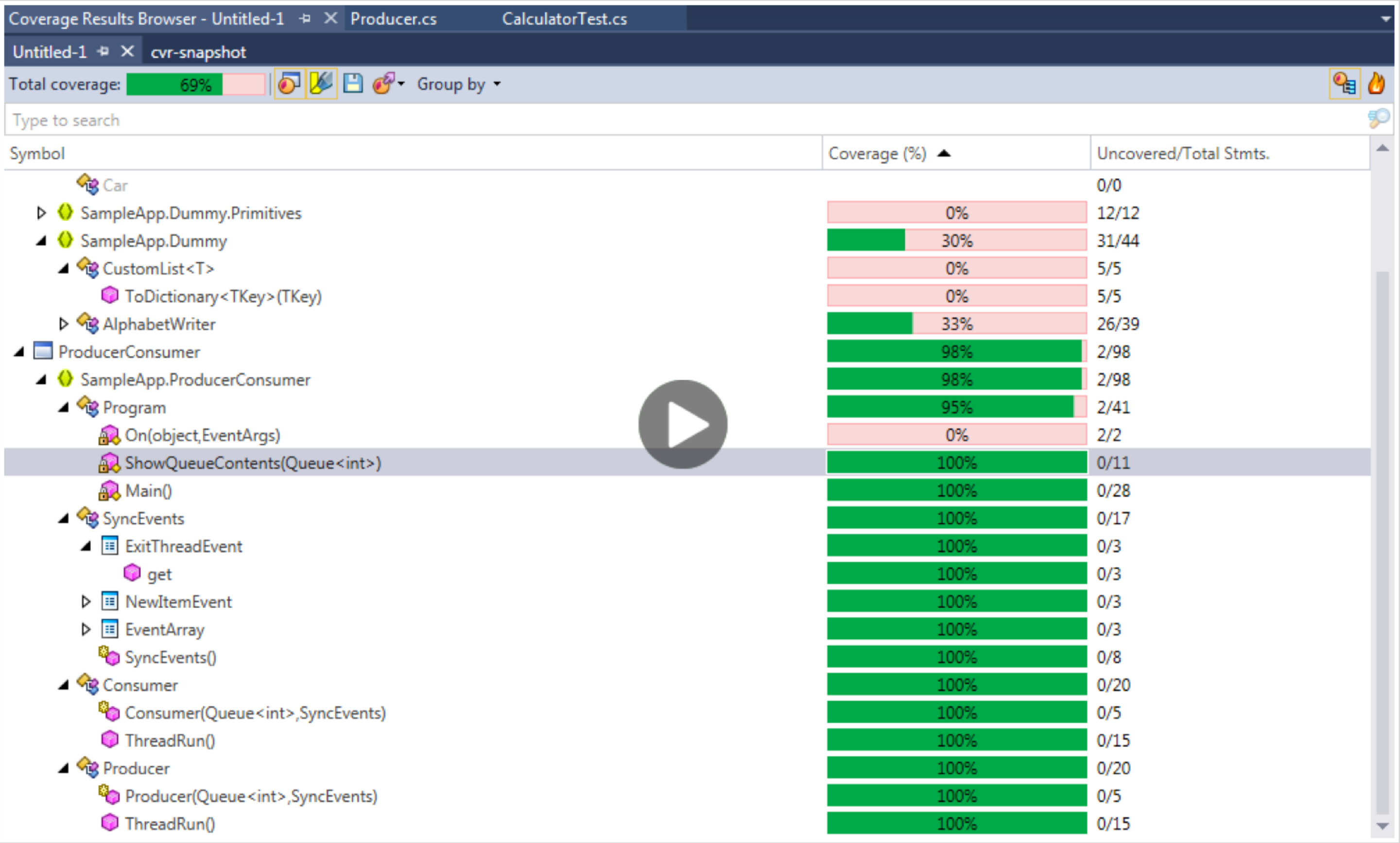
```
{
const struct Opt *const *o;

log_option(telnet, "server", cmd, option);
for (o = opts; *o; o++) {

for (o = opts; *o; o++) {
  if ((*o)->option == option && (*o)->ack == cmd) {
    switch (telnet->opt_states[(*o)->index]) {
      case REQUESTED:
        telnet->opt_states[(*o)->index] = ACTIVE;
        activate_option(telnet, *o);

        break;
      case ACTIVE:
        break;
      case INACTIVE:
        telnet->opt_states[(*o)->index] = ACTIVE;
        send_opt(telnet, (*o)->send, option);
        activate_option(telnet, *o);

        break;
      case REALLY_INACTIVE:
        send_opt(telnet, (*o)->nsend, option);

        break;

    return;
  } else if ((*o)->option == option && (*o)->nak == cmd) {
    switch (telnet->opt_states[(*o)->index]) {
      case REQUESTED:
        telnet->opt_states[(*o)->index] = INACTIVE;
        refused_option(telnet, *o);

        break;
```

Test Coverage

dotCover

Analyse .NET code coverage

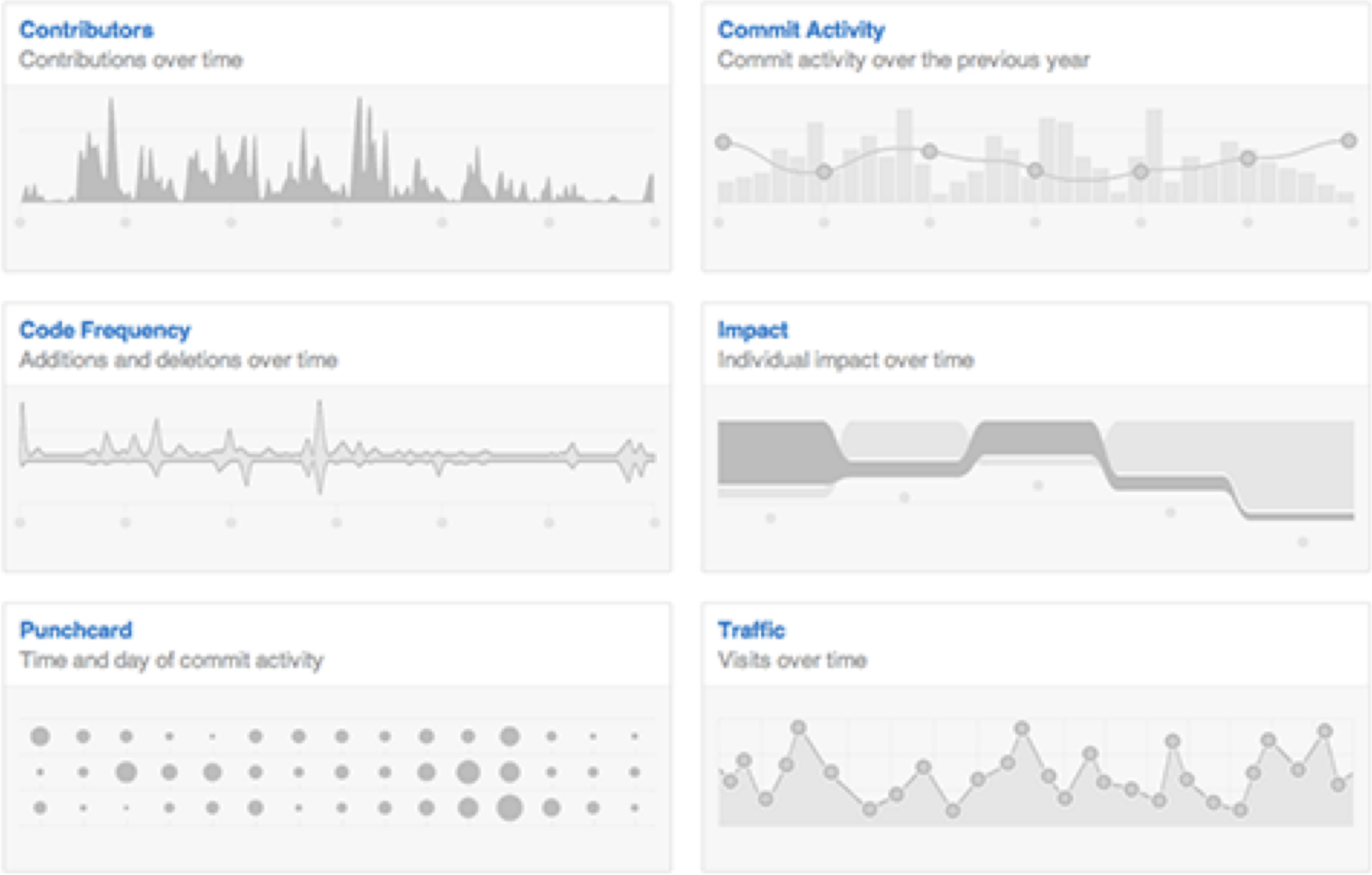
Make sure you know to what extent your code is covered with unit tests. dotCover calculates and reports statement-level code coverage in applications targeting .NET Framework 2.0 to 4.5 or Silverlight 4 or 5.



Total coverage: 69%

Symbol	Coverage (%)	Uncovered/Total Stmts.
Car		0/0
SampleApp.Dummy.Primitives	0%	12/12
SampleApp.Dummy	30%	31/44
CustomList<T>	0%	5/5
ToDictionary<TKey>(TKey)	0%	5/5
AlphabetWriter	33%	26/39
ProducerConsumer	98%	2/98
SampleApp.ProducerConsumer	98%	2/98
Program	95%	2/41
On(object,EventArgs)	0%	2/2
ShowQueueContents(Queue<int>)	100%	0/11
Main()	100%	0/28
SyncEvents	100%	0/17
ExitThreadEvent	100%	0/3
get	100%	0/3
NewItemEvent	100%	0/3
EventArray	100%	0/3
SyncEvents()	100%	0/8
Consumer	100%	0/20
Consumer(Queue<int>,SyncEvents)	100%	0/5
ThreadRun()	100%	0/15
Producer	100%	0/20
Producer(Queue<int>,SyncEvents)	100%	0/5
ThreadRun()	100%	0/15

GitHub Does Software Vis ☺



Contributors
Contributions over time

Commit Activity
Commit activity over the previous year

Code Frequency
Additions and deletions over time

Impact
Individual impact over time

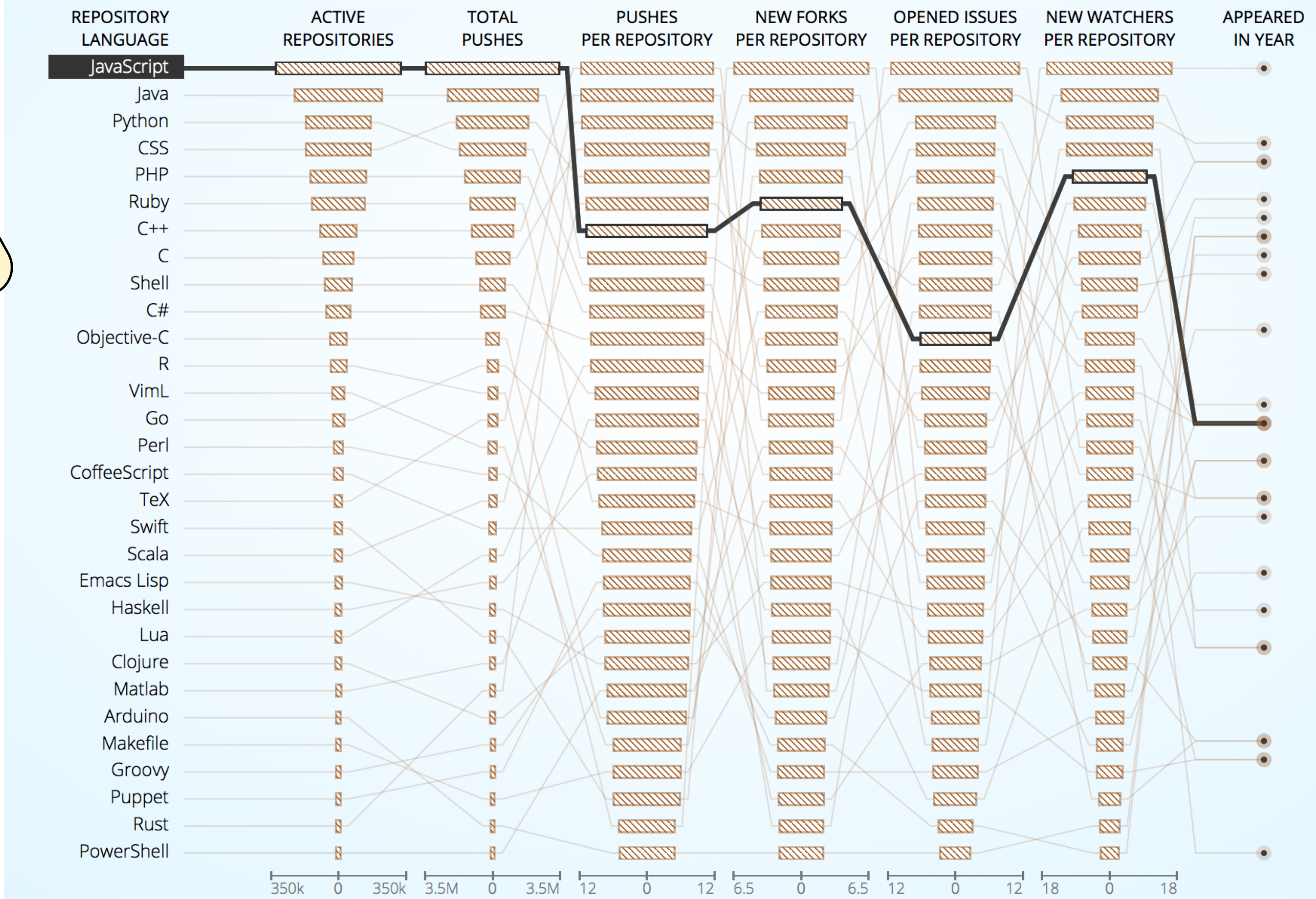
Punchcard
Time and day of commit activity

Traffic
Visits over time

https://github.com/blog/1093-introducing-the-new-github-graphs/

Languages in GitHub

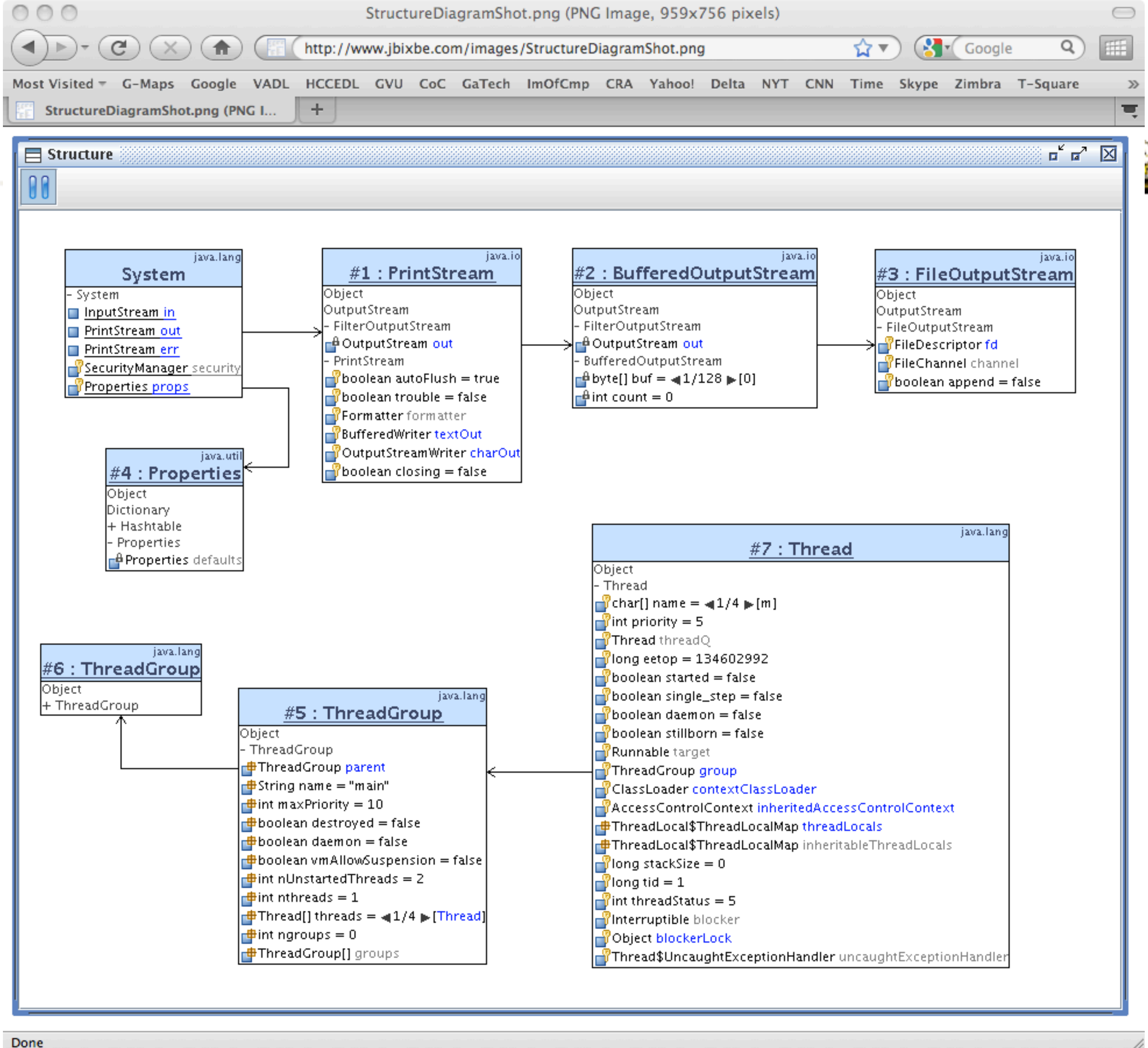
Parallel Coordinates ☺



http://githut.info/

Open Source System Screen Shots

- From jBixbe
  - http://www.jbixbe.com/index.html



StructureDiagramShot.png (PNG Image, 959x756 pixels)
http://www.jbixbe.com/images/StructureDiagramShot.png
Google
Most Visited
G-Maps
Google
VADL
HCCEDL
GVU
CoC
GaTech
ImOfCmp
CRA
Yahoo!
Delta
NYT
CNN
Time
Skype
Zimbra
T-Square
StructureDiagramShot.png (PNG I...
Structure
Done

java.lang
System
- System
InputStream in
PrintStream out
PrintStream err
SecurityManager security
Properties props

java.io
#1 : PrintStream
Object
OutputStream
- FilterOutputStream
OutputStream out
- PrintStream
boolean autoFlush = true
boolean trouble = false
Formatter formatter
BufferedWriter textOut
OutputStreamWriter charOut
boolean closing = false

java.io
#2 : BufferedOutputStream
Object
OutputStream
- FilterOutputStream
OutputStream out
- BufferedOutputStream
byte[] buf = ◀1/128 ▶[0]
int count = 0

java.io
#3 : FileOutputStream
Object
OutputStream
- FileOutputStream
FileDescriptor fd
FileChannel channel
boolean append = false

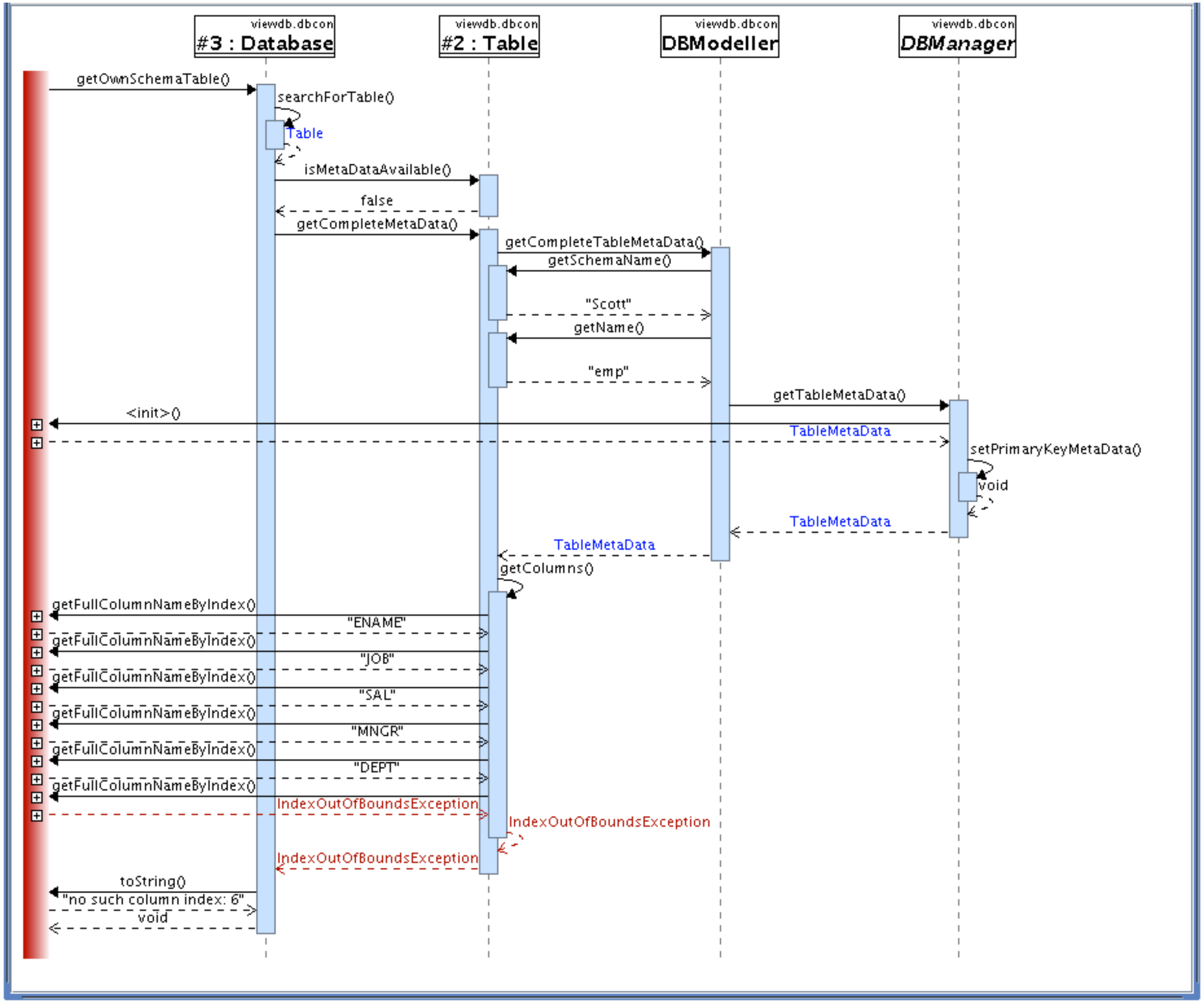
java.util
#4 : Properties
Object
Dictionary
+ Hashtable
- Properties
Properties defaults

java.lang
#7 : Thread
Object
- Thread
char[] name = ◀1/4 ▶[m]
int priority = 5
Thread threadQ
long eetop = 134602992
boolean started = false
boolean single_step = false
boolean daemon = false
boolean stillborn = false
Runnable target
ThreadGroup group
ClassLoader contextClassLoader
AccessControlContext inheritedAccessControlContext
ThreadLocal$ThreadLocalMap threadLocals
ThreadLocal$ThreadLocalMap inheritableThreadLocals
long stackSize = 0
long tid = 1
int threadStatus = 5
Interruptible blocker
Object blockerLock
Thread$UncaughtExceptionHandler uncaughtExceptionHandler

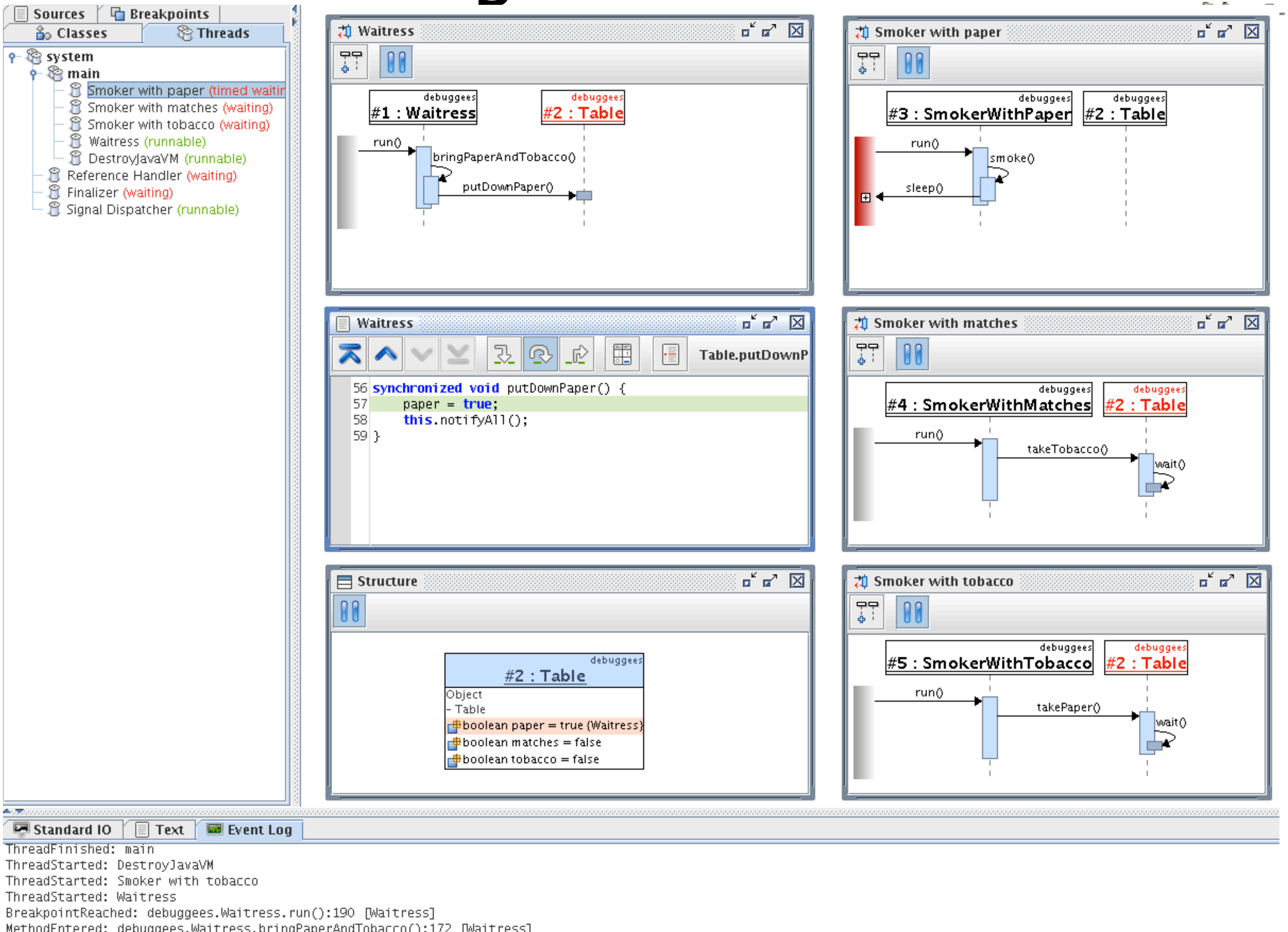
java.lang
#6 : ThreadGroup
Object
+ ThreadGroup

java.lang
#5 : ThreadGroup
Object
- ThreadGroup
ThreadGroup parent
String name = "main"
int maxPriority = 10
boolean destroyed = false
boolean daemon = false
boolean vmAllowSuspension = false
int nUnstartedThreads = 2
int nthreads = 1
Thread[] threads = ◀1/4 ▶[Thread]
int ngroups = 0
ThreadGroup[] groups

Message Exchanges



Multi-threading



- system
  - main
    - Smoker with paper (timed waitir
    - Smoker with matches (waiting)
    - Smoker with tobacco (waiting)
    - Waitress (runnable)
    - DestroyJavaVM (runnable)
  - Reference Handler (waiting)
  - Finalizer (waiting)
  - Signal Dispatcher (runnable)

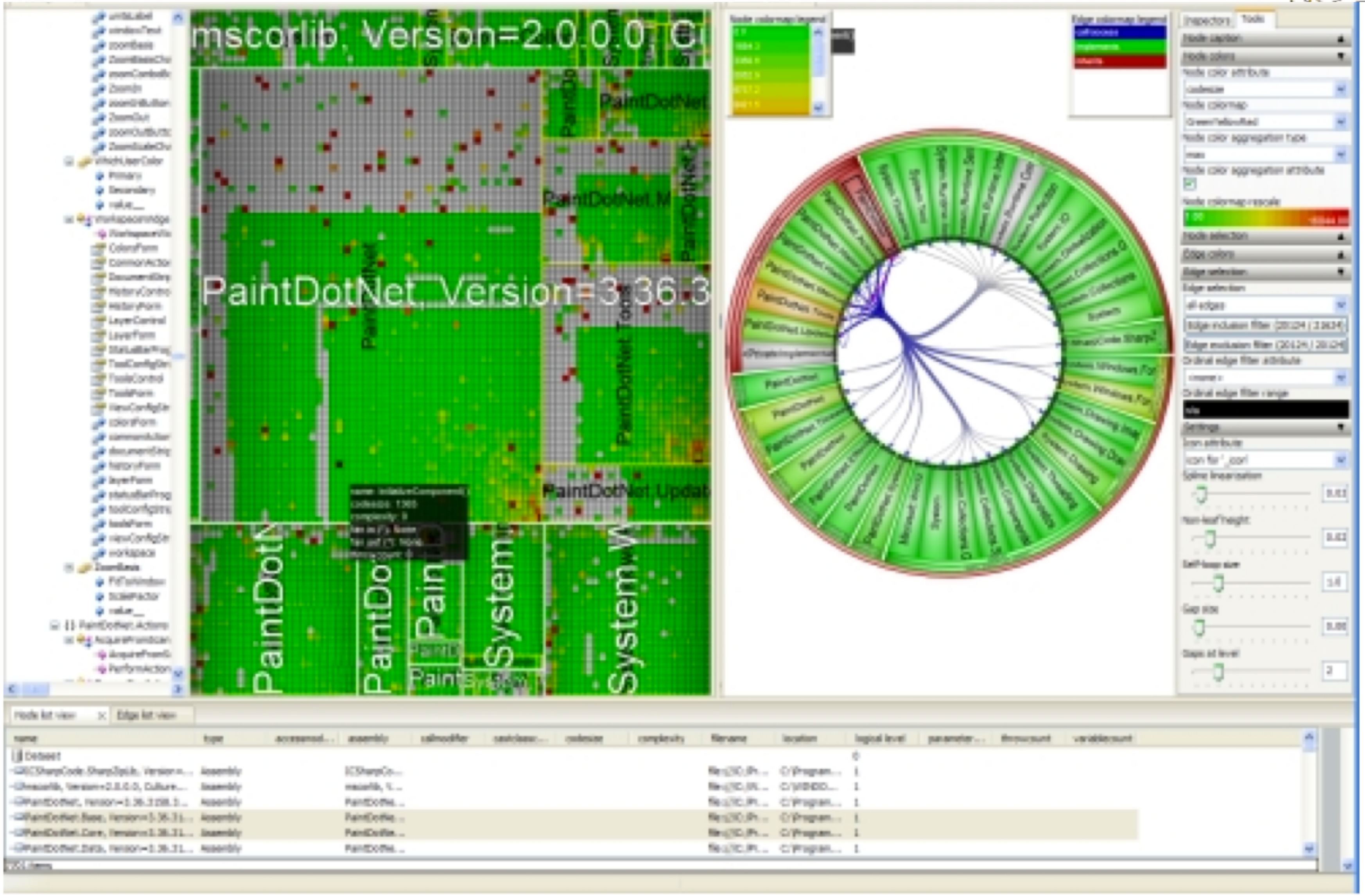
```
synchronized void putDownPaper() {
    paper = true;
    this.notifyAll();
}
```

#2 : Table

Object
- Table
- boolean paper = true {Waitress}
- boolean matches = false
- boolean tobacco = false

```
ThreadFinished: main
ThreadStarted: DestroyJavaVM
ThreadStarted: Smoker with tobacco
ThreadStarted: Waitress
BreakpointReached: debuggees.Waitress.run():190 [Waitress]
MethodEntered: debuggees.Waitress.bringPaperAndTobacco():172 [Waitress]
```

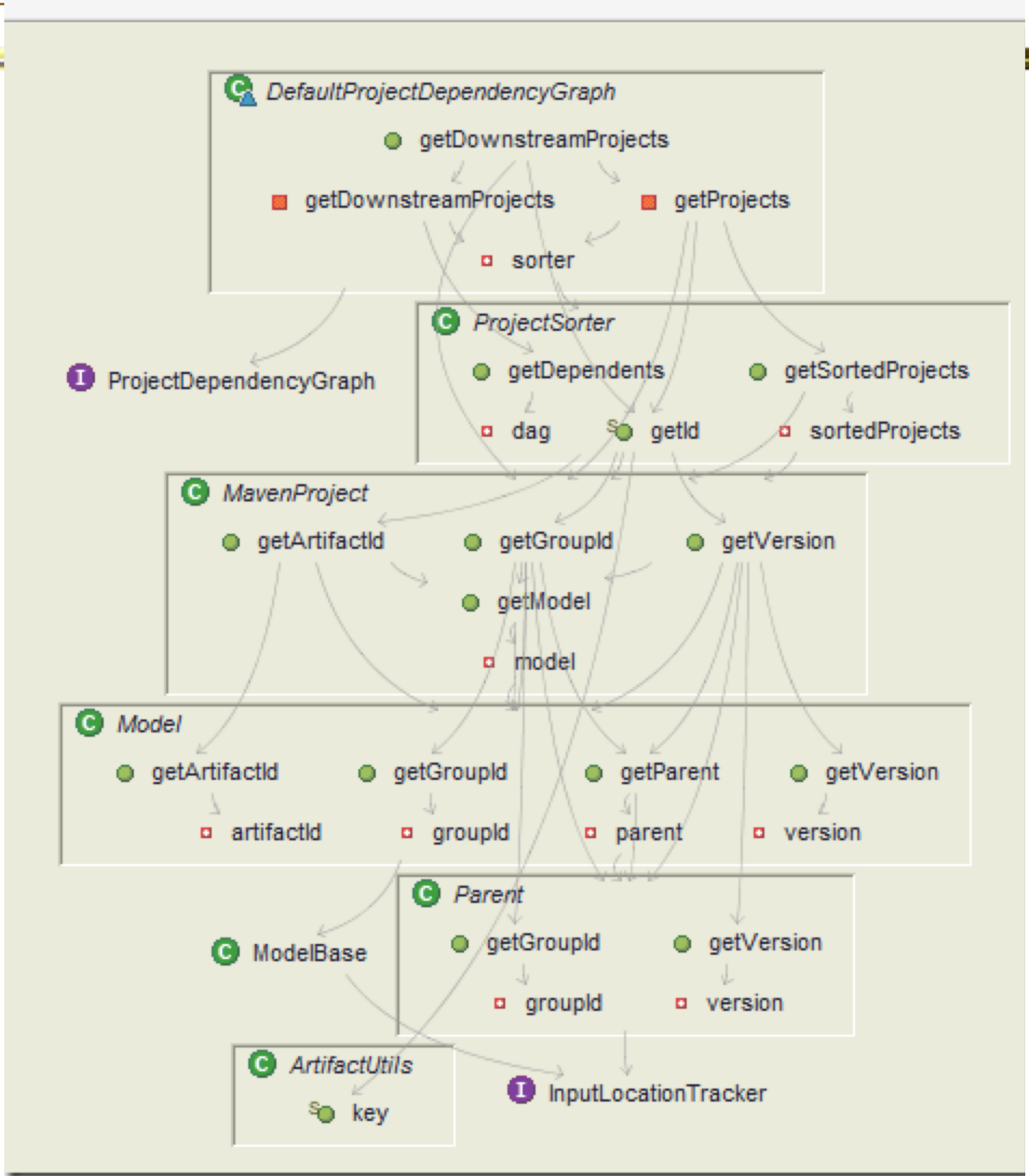
SolidSX https://www.youtube.com/watch?v=Rd-0CgqifS4#t=50



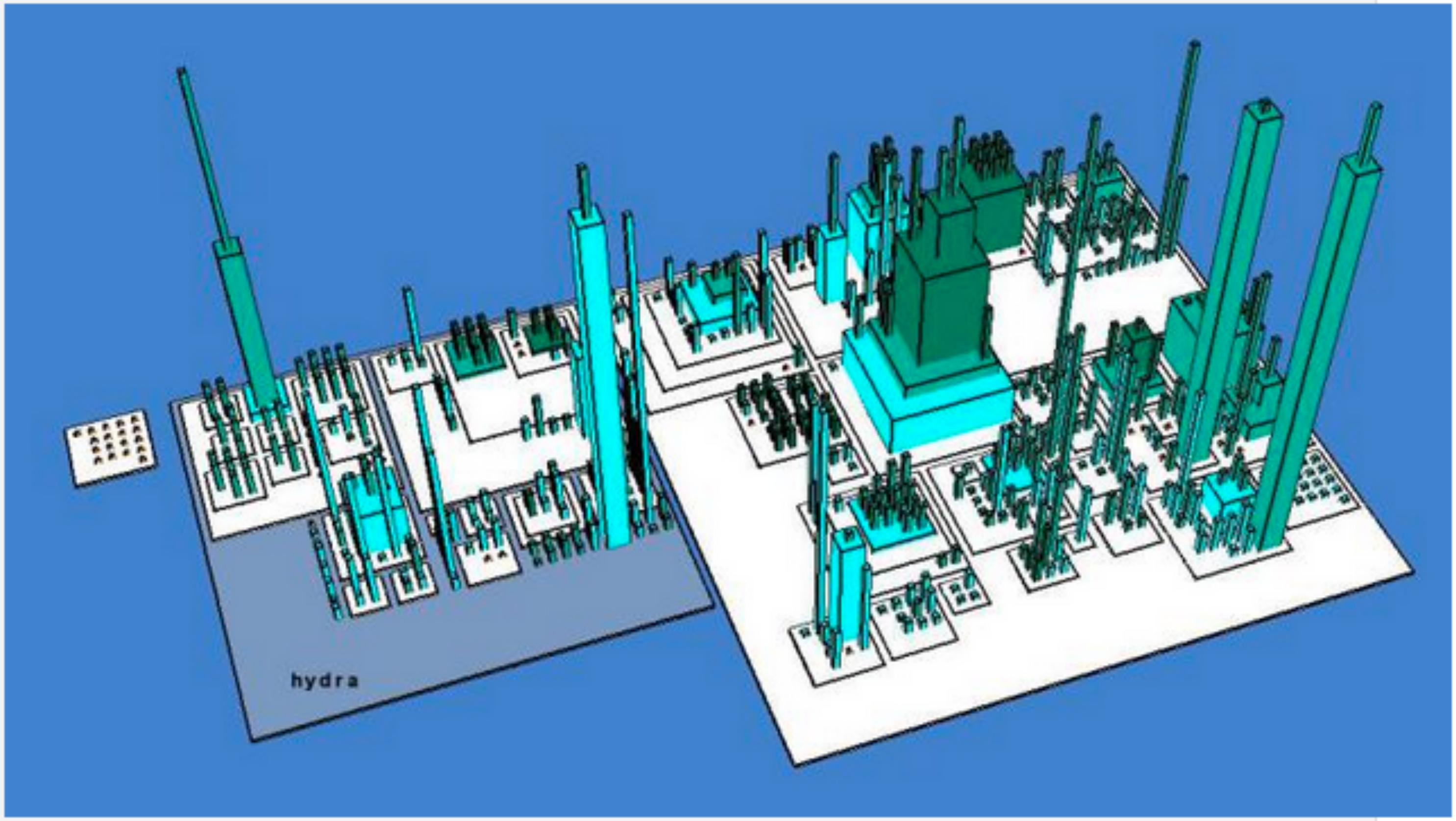
SolidSX allows you to open multiple views on the same data. This image shows the Radial, Treemap and List views on the PaintDotNet dataset produced by the SolidSX .NET importer.

Structure 101

http://structure101.com/resources/#resources=0



Codstruction – Eclipse Plug-In

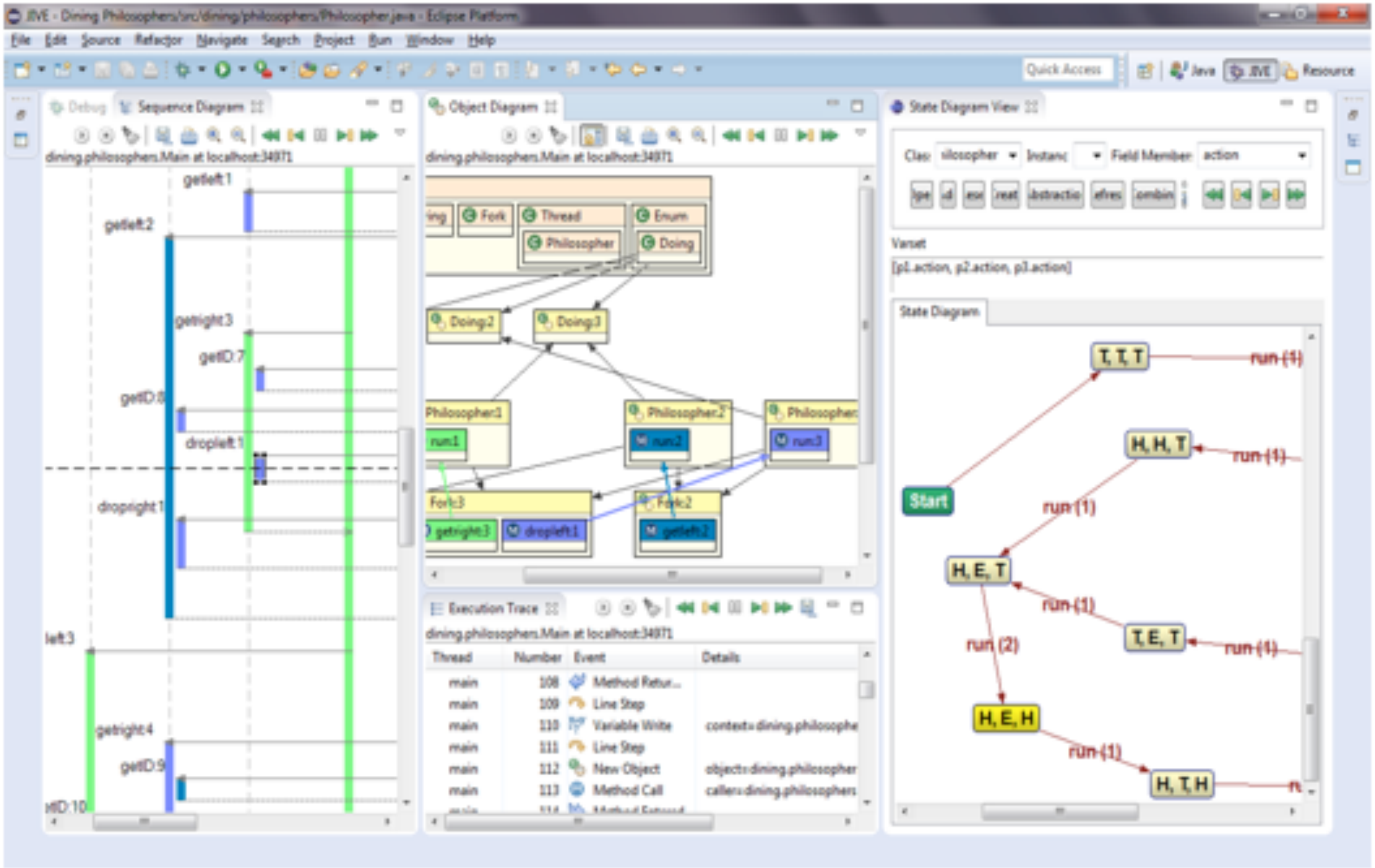


My current prototype inspired by CodeCity. This is a screen-shot of visualization on the same project the one visualized by X-ray above. Each colored box represent a class and class hierarchy is represented by box stacking. Height of box is proportional to number of methods of the class.

https://codstruction.wordpress.com/

Jive – Another Eclipse Plug-in

https://www.youtube.com/watch?v=ifQFeWT0RZ0&feature=player_embedded



Thread	Number	Event	Details
main	108	Method Retur...	
main	109	Line Step	
main	110	Variable Write	context=dining.philosophe
main	111	Line Step	
main	112	New Object	object=dining.philosopher
main	113	Method Call	caller=dining.philosophers

http://www.cse.buffalo.edu/jive/

Stargate: An Author-Centric Approach to Software Project Visualization

Ogawas and Ma, Stargate: An Author-Centric Approach to Software Project Visualization Proceedings of IEEE PacificVis 2008 March, 2008

Stargate – Start with Modified Sunburst

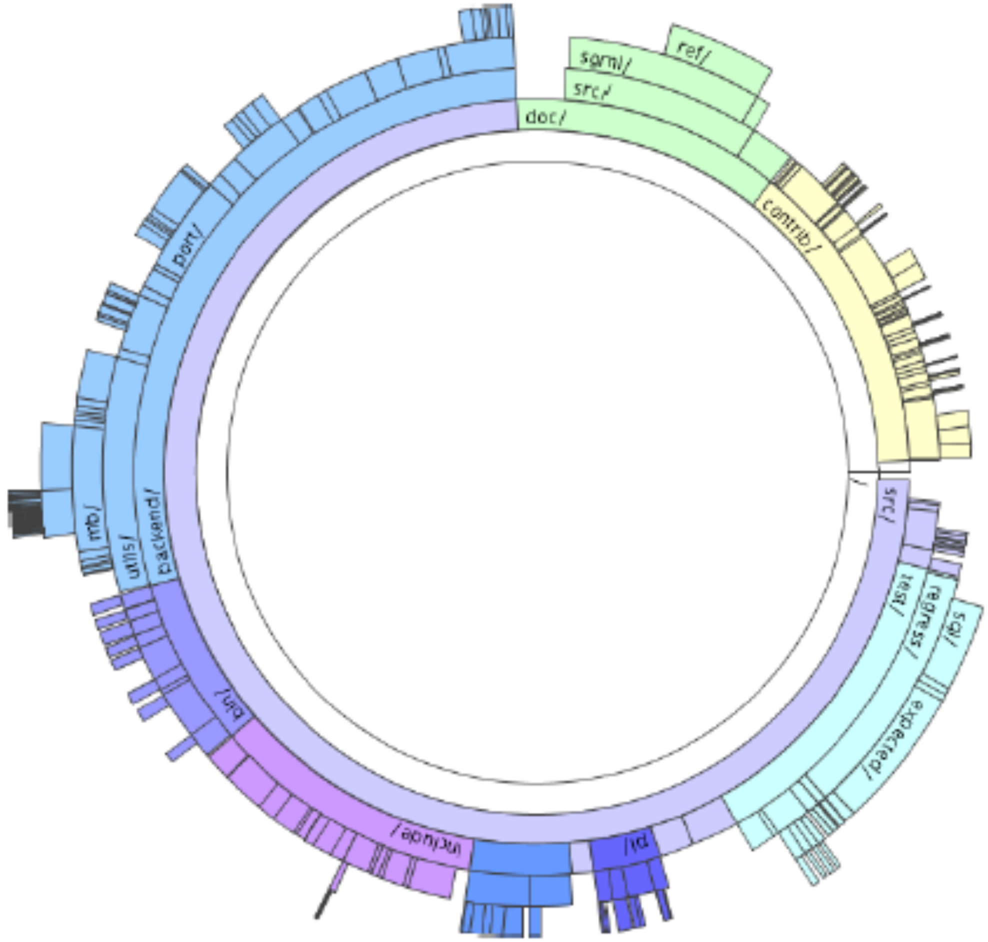
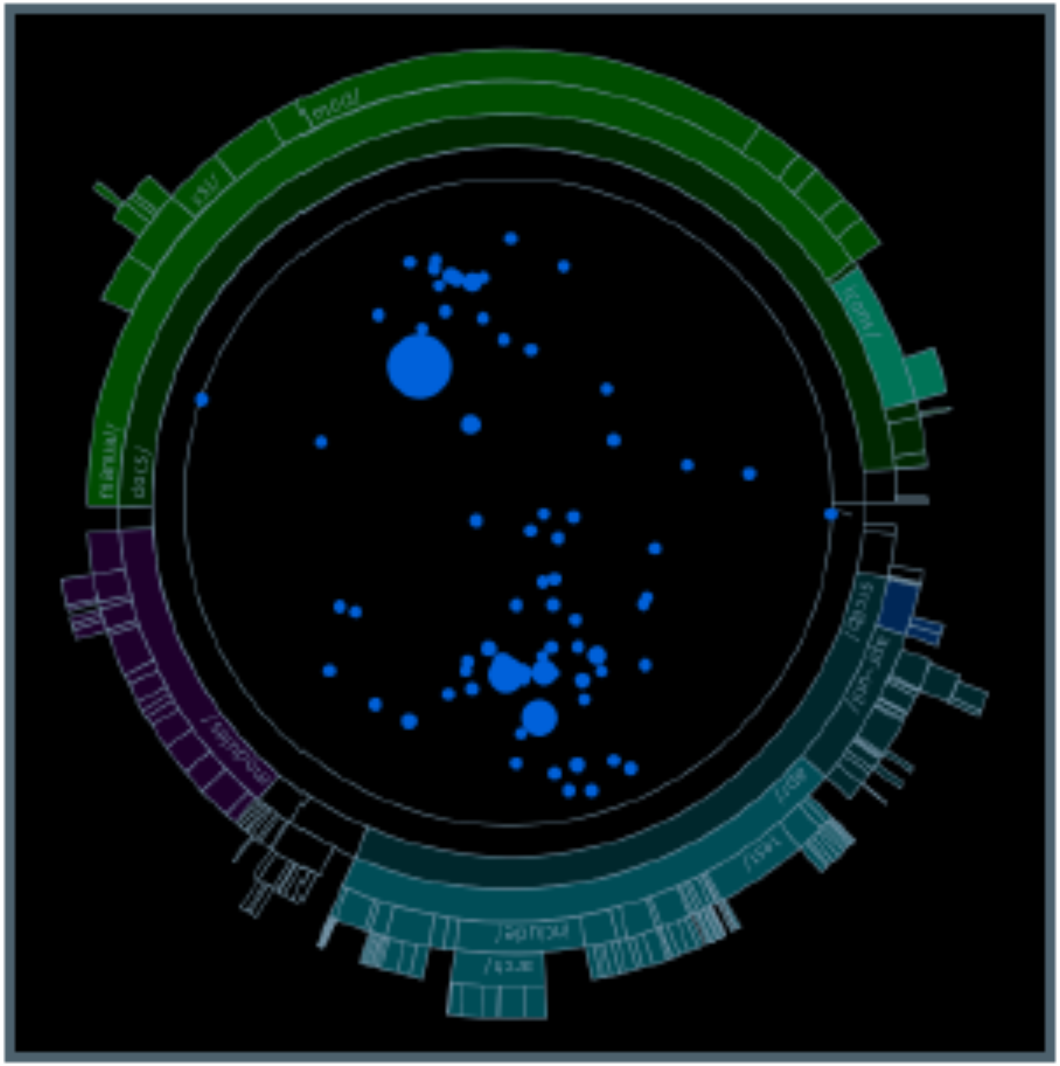


Figure 2: The Gate component. This is the directory hierarchy of the PostgreSQL version control repository. The documents directory is colored green (at the top) and the source code directory is colored in various shades of blue.

Stargate – Add Developers

- Dots are developers
  - Placed close to files they worked on
  - Size => amount of activity
- File colors => File type



Stargate – Add Communications

- Email communications among developers

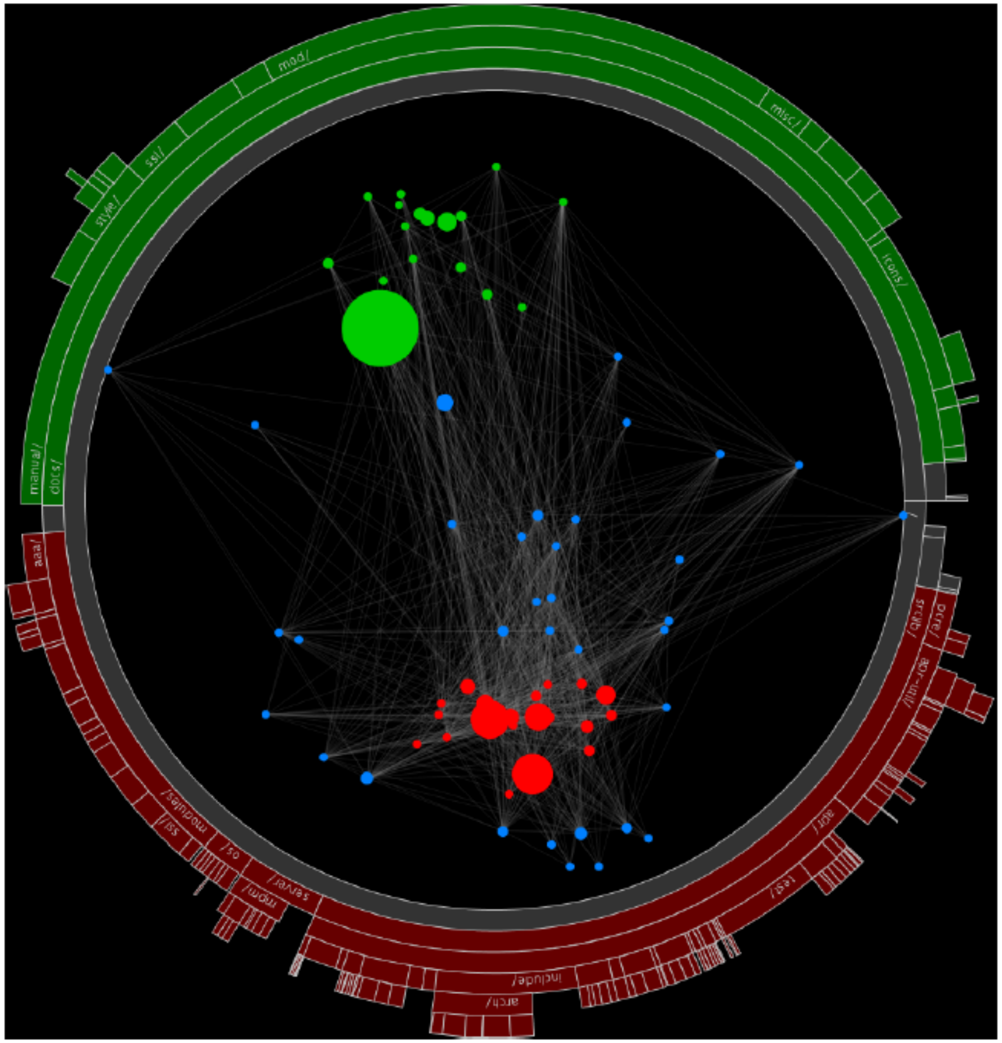
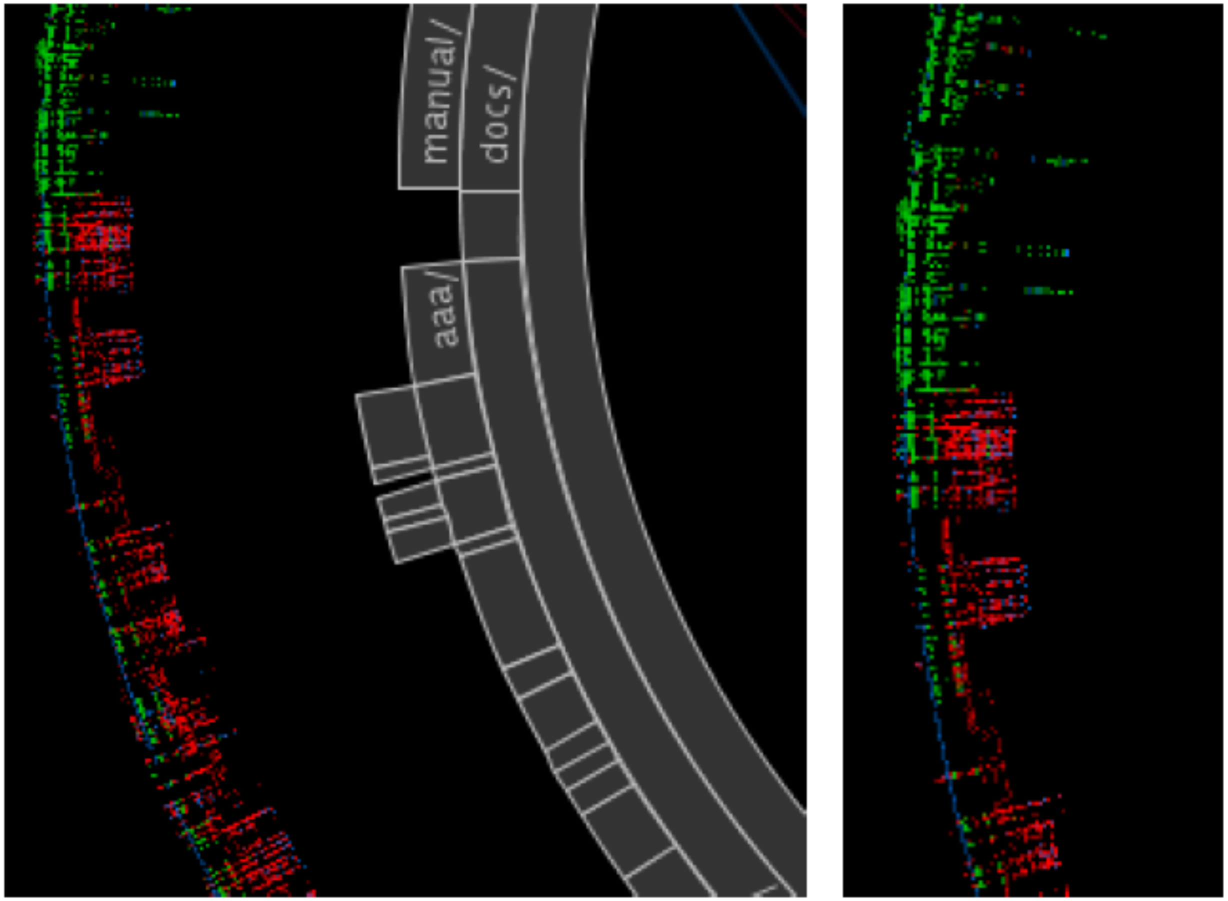


Figure 10: Overview of the Apache project at the latest timestep. The documentation directory and documenters have been colored green. The source code directory and core developers have been colored red.

Stargate – Add File History



(a) Stardust (b) Detail

Figure 6: The stardust. Each line represents a file. Each colored dot on the line represents a change to the file. Dot color corresponds to the developer that made that change. Time flows radially outward.

Stargate Video

- http://vidi.cs.ucdavis.edu/research/videos/stargate
- (Show from SV folder)

SV Takeaways

- Multiple dimensions to SV
  - Static code/program relationships
  - Dynamic code/program relationships
  - Testing/coverge
  - Project evolution
- Many methods we have studied are applicable, such as ways to depict
  - Time-varying behaviors
  - Object relationships

Back to GitHub – Design Exercise

- Use GitHub Viz tools on your repository
  - 4460 project or other
  - What are they lacking?
- Sketch ways to extend

  OR
- Sketch new Viz tools for GitHub

The End