Ontwerp van SoftwareSystemen

3 Metrics and Software Visualization

Roel Wuyts

OSS 2012-2013

Courtesy of Prof. Dr. Michele Lanza
http://www.inf.unisi.ch/faculty/lanza/

[A cool and excellent teacher and person]

Software Design & Evolution

Michele Lanza

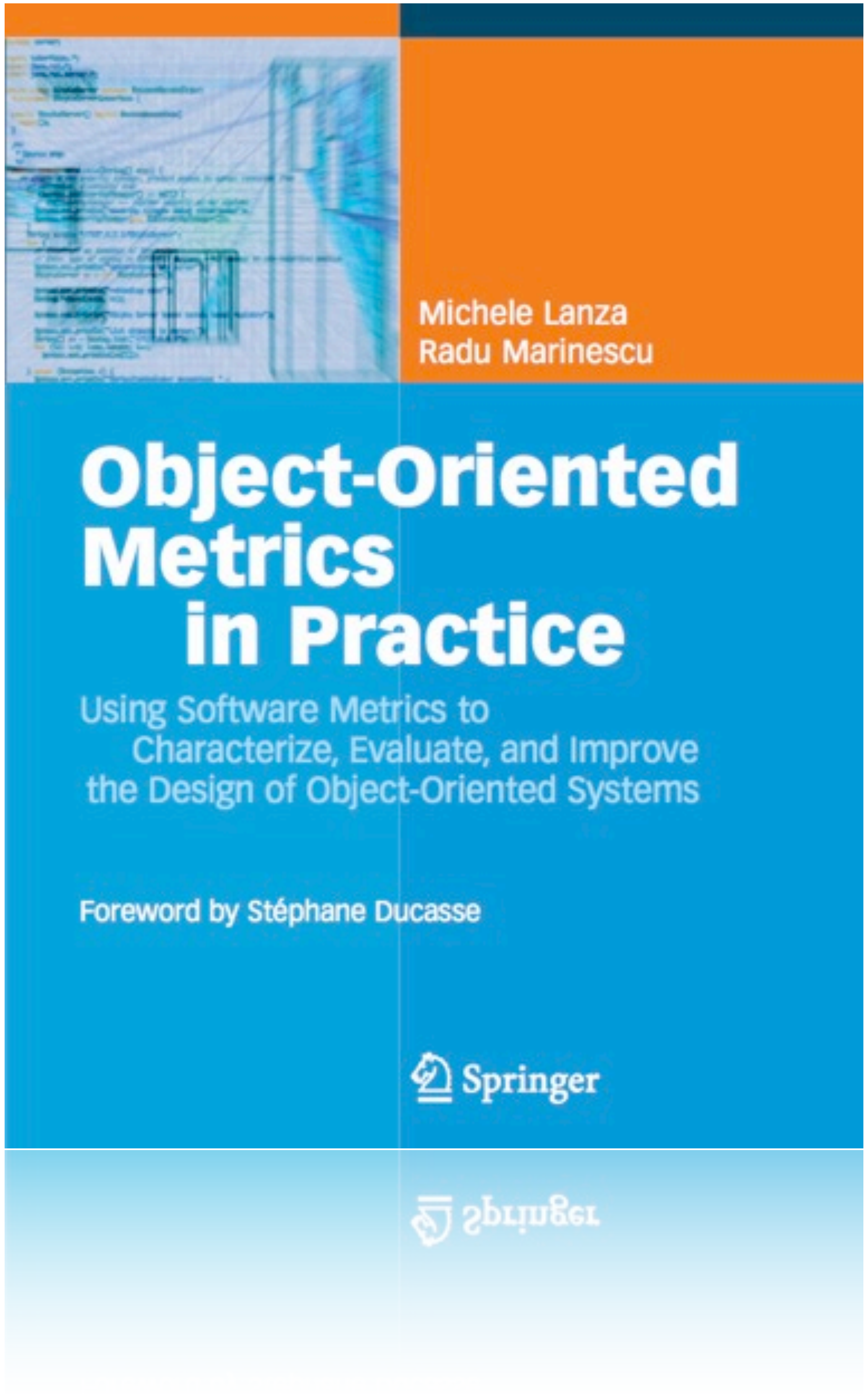
Lecture 04

Metrics & Problem Detection

Reference

M. Lanza, R. Marinescu
“Object-Oriented Metrics in Practice”

Springer, 2006
ISBN 3-540-24429-8



Michele Lanza
Radu Marinescu

Object-Oriented Metrics in Practice

Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object-Oriented Systems

Foreword by Stéphane Ducasse

Springer

You cannot control what you cannot measure

Tom de Marco

Metrics are functions that assign **numbers** to *products*, *processes* and *resources*

Software metrics are **measure**ments which relate to software **systems**, **processes** or related **documents**

Metrics compress system properties and traits into numbers

Let's see some examples..

Examples of size metrics

Chidamber & Kemerer, 1994
Lorenz & Kidd, 1994

Examples of size metrics

- NOM - Number of Methods

Chidamber & Kemerer, 1994
Lorenz & Kidd, 1994

Examples of size metrics

- NOM - Number of Methods
- NOA - Number of Attributes

Chidamber & Kemerer, 1994
Lorenz & Kidd, 1994

Examples of size metrics

- NOM - Number of Methods
- NOA - Number of Attributes
- LOC - Number of Lines of Code

Chidamber & Kemerer, 1994
Lorenz & Kidd, 1994

Examples of size metrics

- NOM - Number of Methods
- NOA - Number of Attributes
- LOC - Number of Lines of Code
- NOS - Number of Statements

Chidamber & Kemerer, 1994
Lorenz & Kidd, 1994

Examples of size metrics

- NOM - Number of Methods
- NOA - Number of Attributes
- LOC - Number of Lines of Code
- NOS - Number of Statements
- NOC - Number of Children

Chidamber & Kemerer, 1994
Lorenz & Kidd, 1994

Cyclomatic Complexity (CYCLO)

McCabe, 1976

Cyclomatic Complexity (CYCLO)

- The McCabe cyclomatic complexity (CYCLO) counts the number of independent paths through the code of a function

McCabe, 1976

Cyclomatic Complexity (CYCLO)

- The McCabe cyclomatic complexity (CYCLO) counts the number of independent paths through the code of a function
  - Good: it reveals the minimum number of tests to write

McCabe, 1976

Cyclomatic Complexity (CYCLO)

- The McCabe cyclomatic complexity (CYCLO) counts the number of independent paths through the code of a function
    - Good: it reveals the minimum number of tests to write
    - Bad: its interpretation does not directly lead to improvement actions

McCabe, 1976

Weighted Method Count (WMC)

Chidamber & Kemerer, 1994

Weighted Method Count (WMC)

- WMC sums up the complexity of a class' methods (measured by the metric of your choice, usually CYCLO)

Chidamber & Kemerer, 1994

Weighted Method Count (WMC)

- WMC sums up the complexity of a class' methods (measured by the metric of your choice, usually CYCLO)
  - Good: It is configurable, thus adaptable to our precise needs

Chidamber & Kemerer, 1994

Weighted Method Count (WMC)

- WMC sums up the complexity of a class′ methods (measured by the metric of your choice, usually CYCLO)
  - Good: It is configurable, thus adaptable to our precise needs
  - Bad: Its interpretation does not directly lead to improvement actions

Chidamber & Kemerer, 1994

Coupling Between Objects (CBO)

Chidamber & Kemerer, 1994

Coupling Between Objects (CBO)

- CBO shows the number of classes from which methods or attributes are used.

Chidamber & Kemerer, 1994

Coupling Between Objects (CBO)

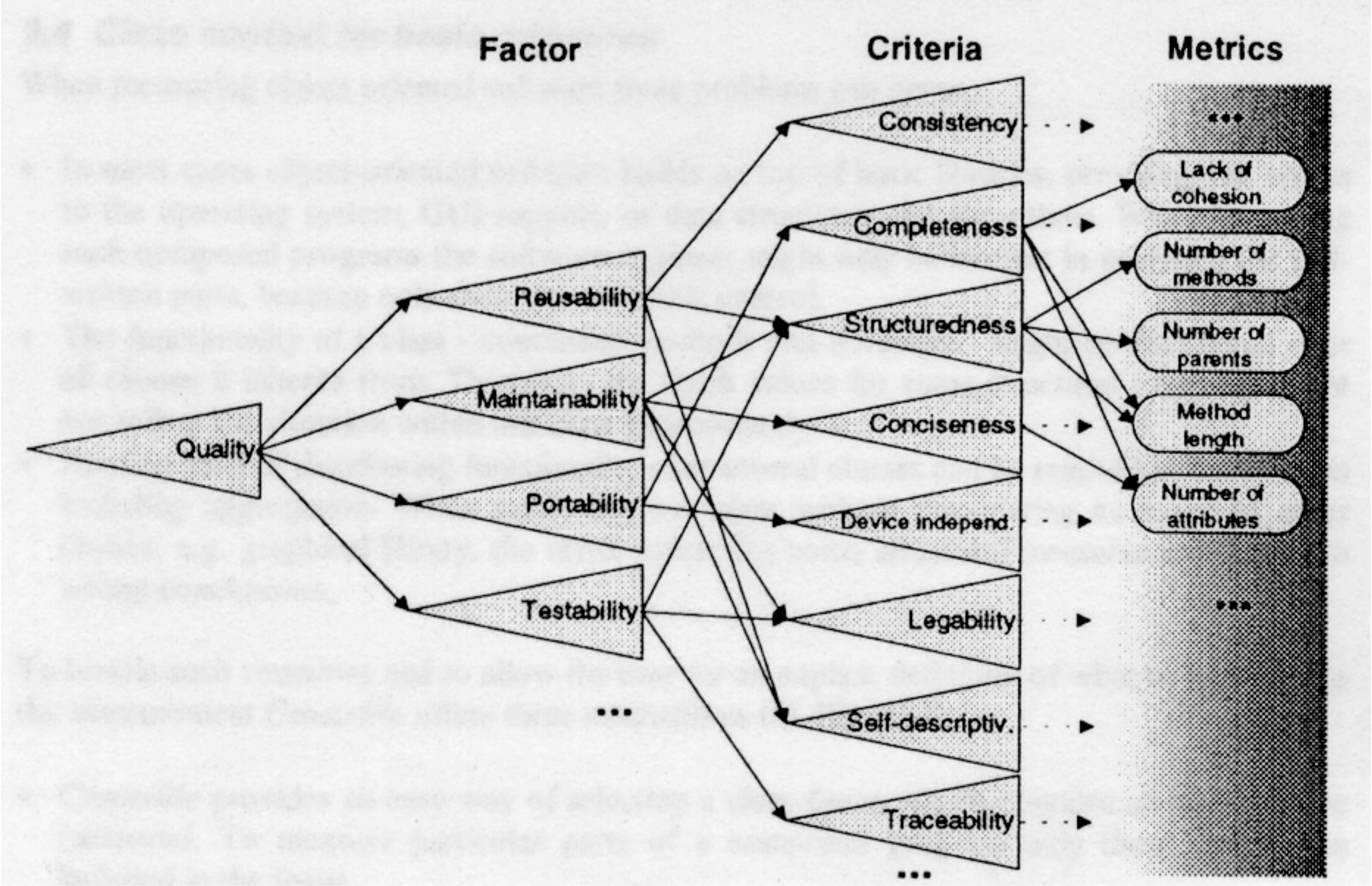
- CBO shows the number of classes from which methods or attributes are used.
  - Good: CBO takes into account real dependencies, not just declared ones

Chidamber & Kemerer, 1994

Coupling Between Objects (CBO)

- CBO shows the number of classes from which methods or attributes are used.
  - Good: CBO takes into account real dependencies, not just declared ones
  - Bad: No differentiation of types and/or intensity of coupling

Chidamber & Kemerer, 1994

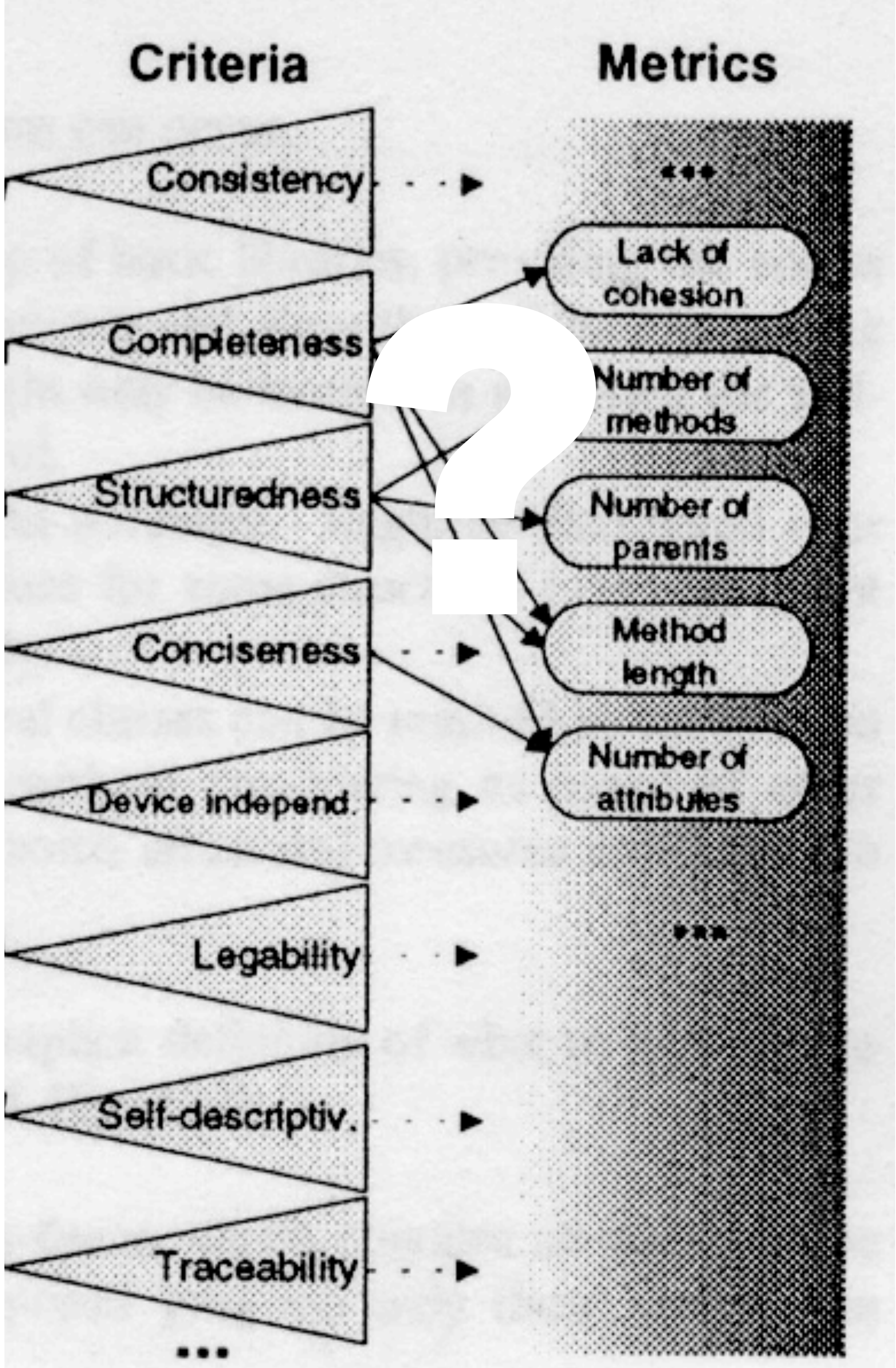


McCall, 1977
Boehm, 1978

Metrics help to assess and improve quality!

Do they?

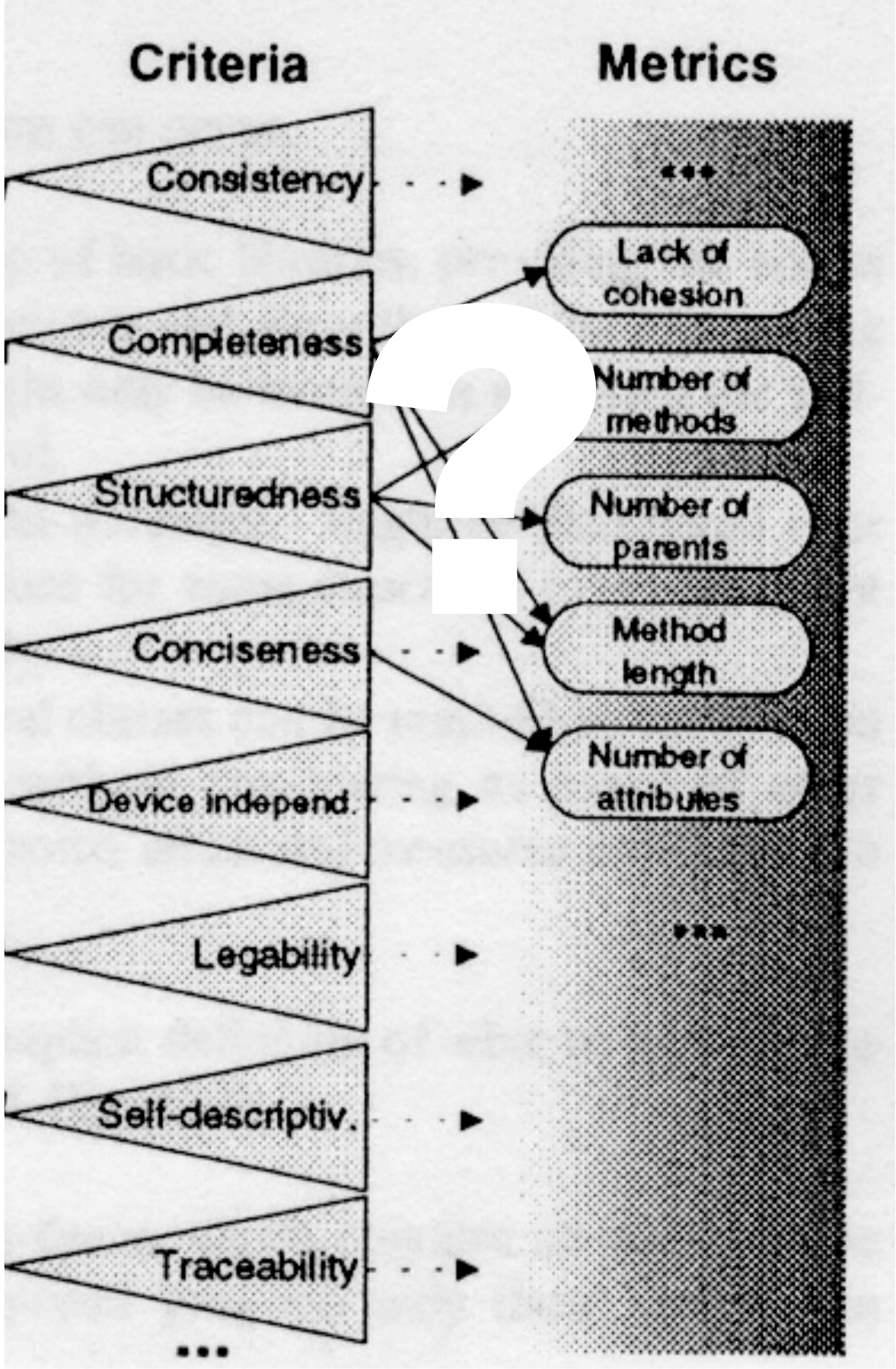
Problems..



McCall, 1977
Boehm, 1978

Problems..

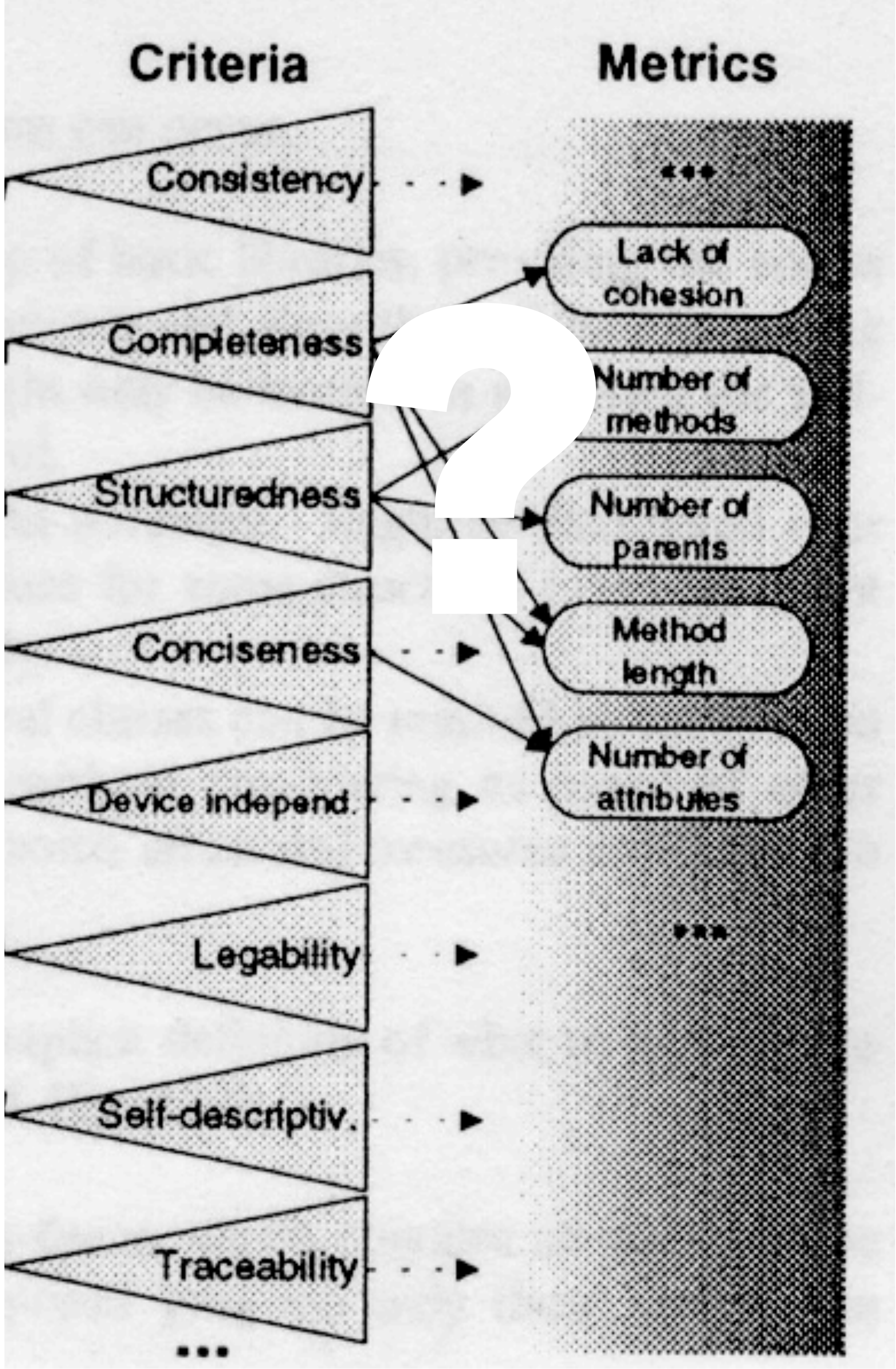
- Metrics granularity



McCall, 1977
Boehm, 1978

Problems..

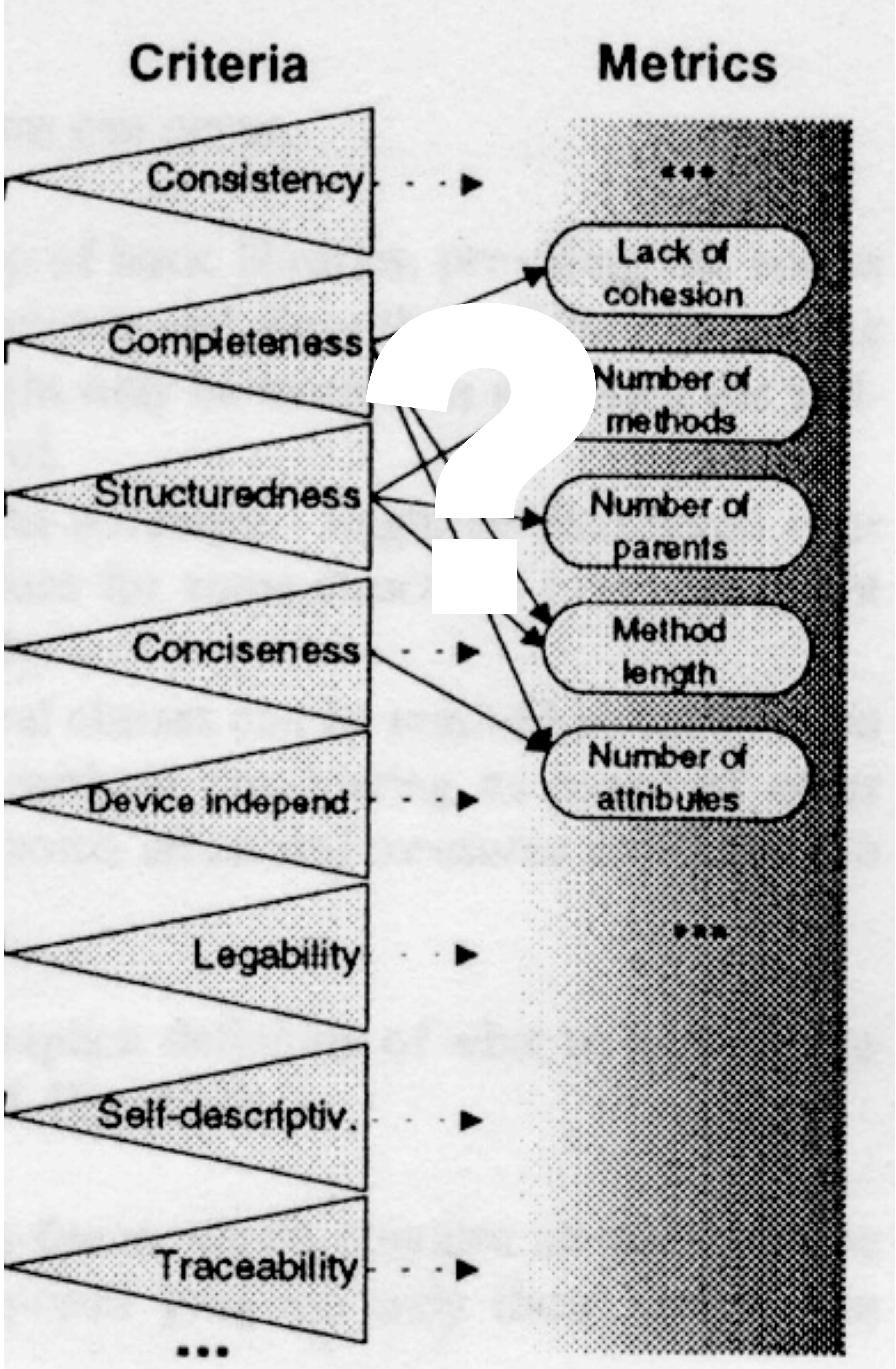
- Metrics granularity
  - metrics capture symptoms,not causes of problems



McCall, 1977
Boehm, 1978

Problems..

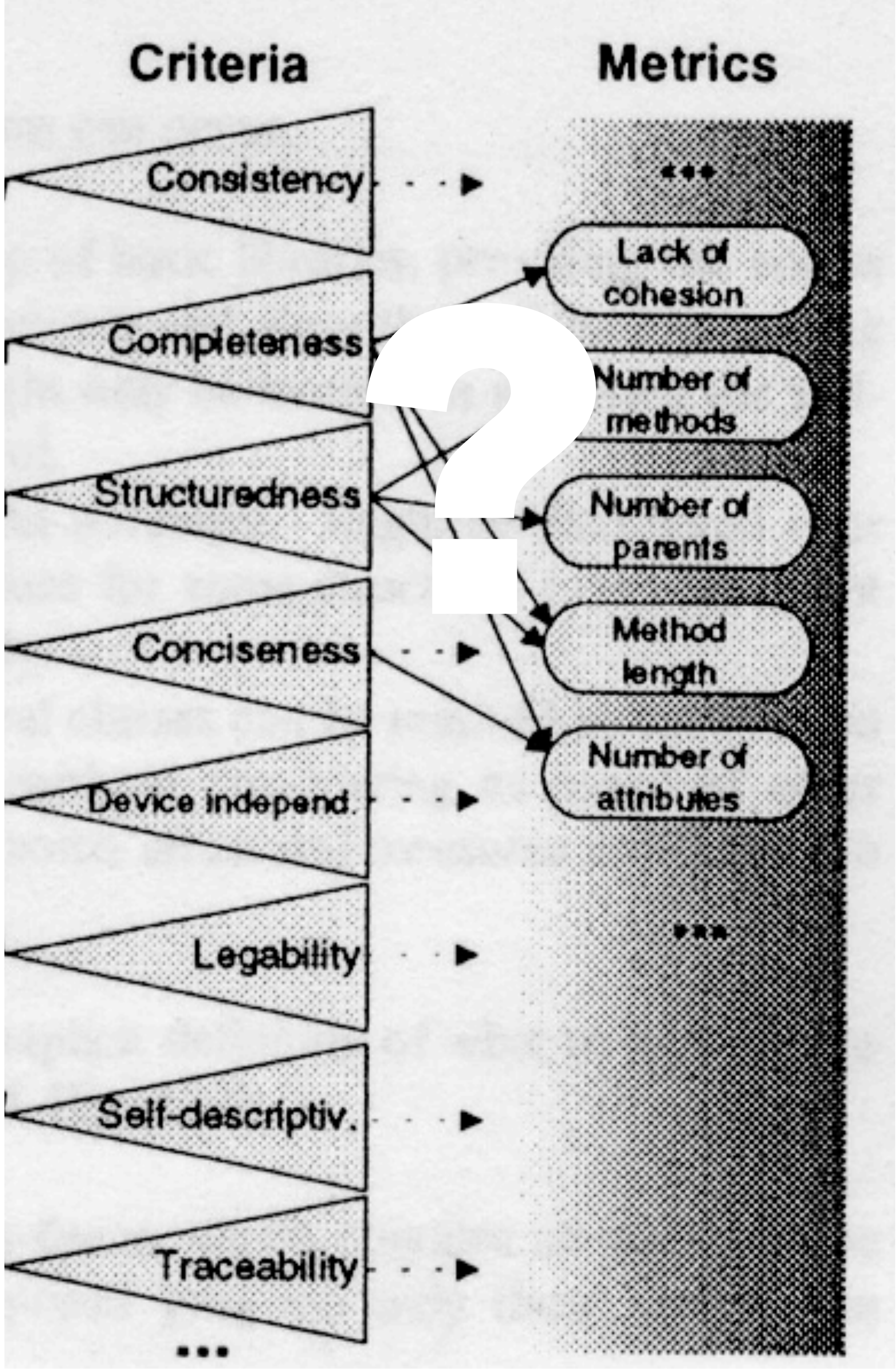
- Metrics granularity
  - metrics capture symptoms,not causes of problems
  - in isolation, metrics do not lead to improvement actions



McCall, 1977
Boehm, 1978

Problems..

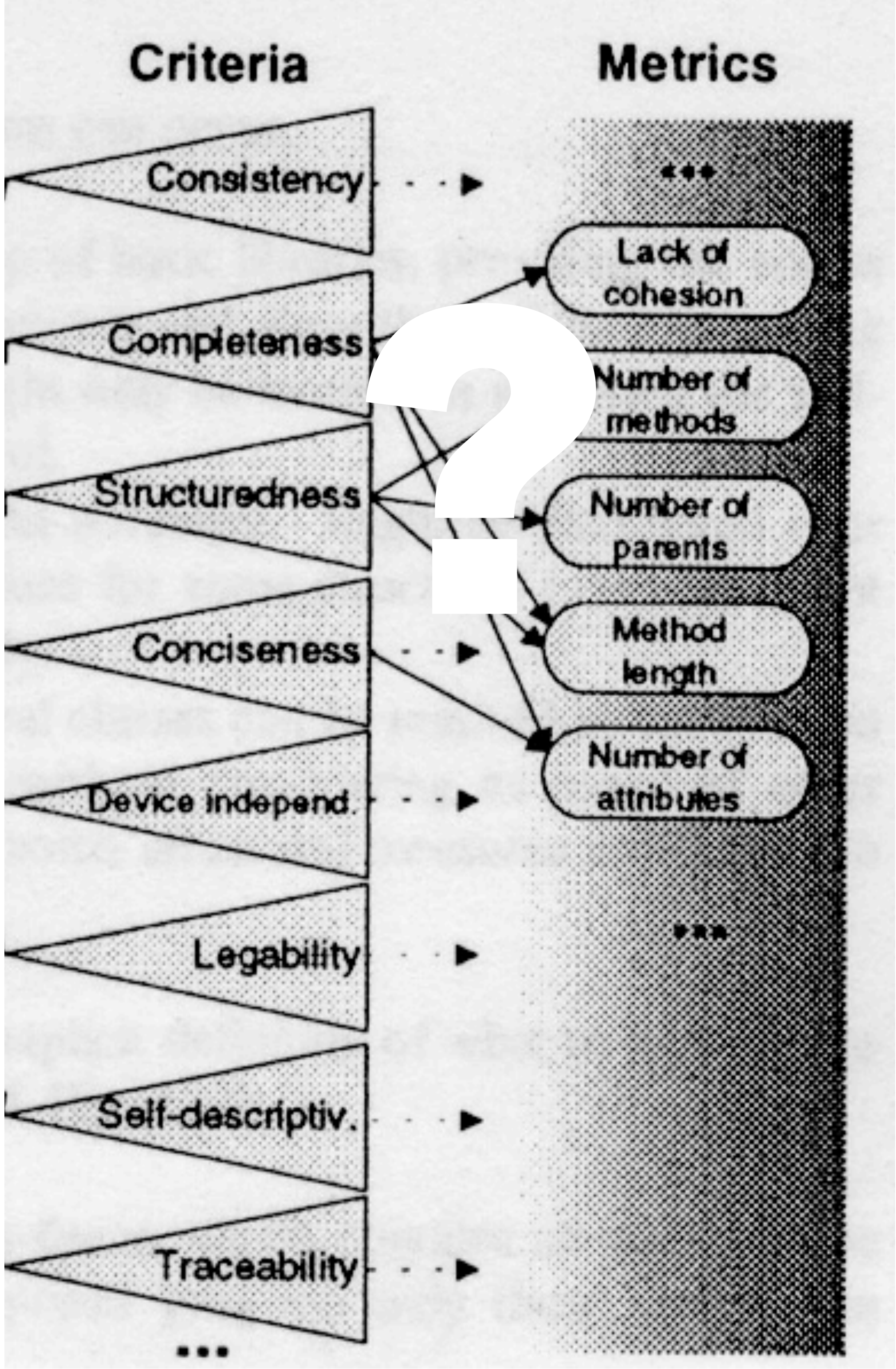
- Metrics granularity
  - metrics capture symptoms,not causes of problems
  - in isolation, metrics do not lead to improvement actions
- Implicit Mapping



McCall, 1977
Boehm, 1978

Problems..

- Metrics granularity
  - metrics capture symptoms,not causes of problems
  - in isolation, metrics do not lead to improvement actions
- Implicit Mapping
  - we do not reason in terms of metrics, but in terms of design (principles)



McCall, 1977
Boehm, 1978

2 big obstacles in using metrics:

Thresholds make metrics hard to interpret

Granularity makes metrics hard to use in isolation

How do I get an initial understanding of a system?

Metric	Value
LOC	35175
NOM	3618
NOC	384
CYCLO	5579
NOP	19
CALLS	15128
FANOUT	8590
AHH	0.12
ANDC	0.31

Metric	Value
LOC	35175
NOM	3618
NOC	384
CYCLO	5579
NOP	19
CALLS	15128
FANOUT	8590
AHH	0.12
ANDC	0.31

Metric	Value
LOC	35175
NOM	3618
NOC	384
CYCLO	5579
NOP	[illegible]
CALLS	[illegible]
FAN	8590
A	0.12
ANDC	0.31

And now what?

We need means to compare

coupling?

We need means to compare

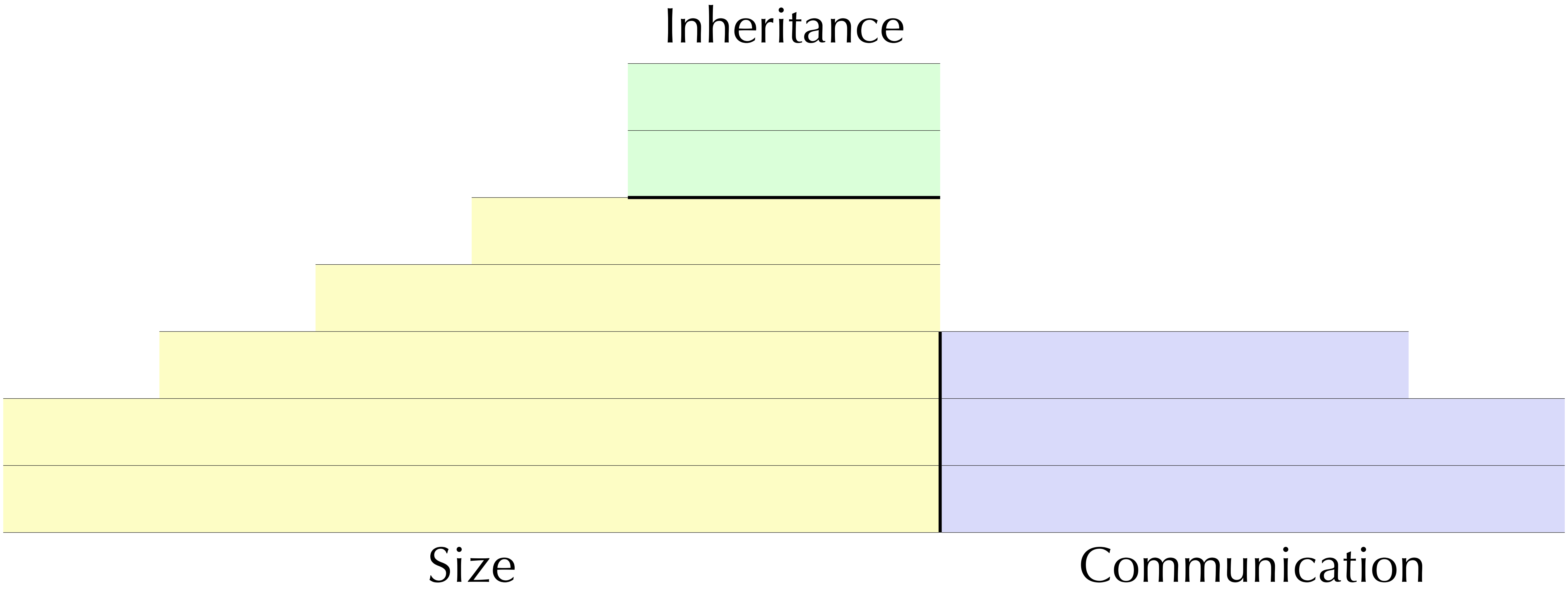
coupling?

We need means to compare

hierarchies?

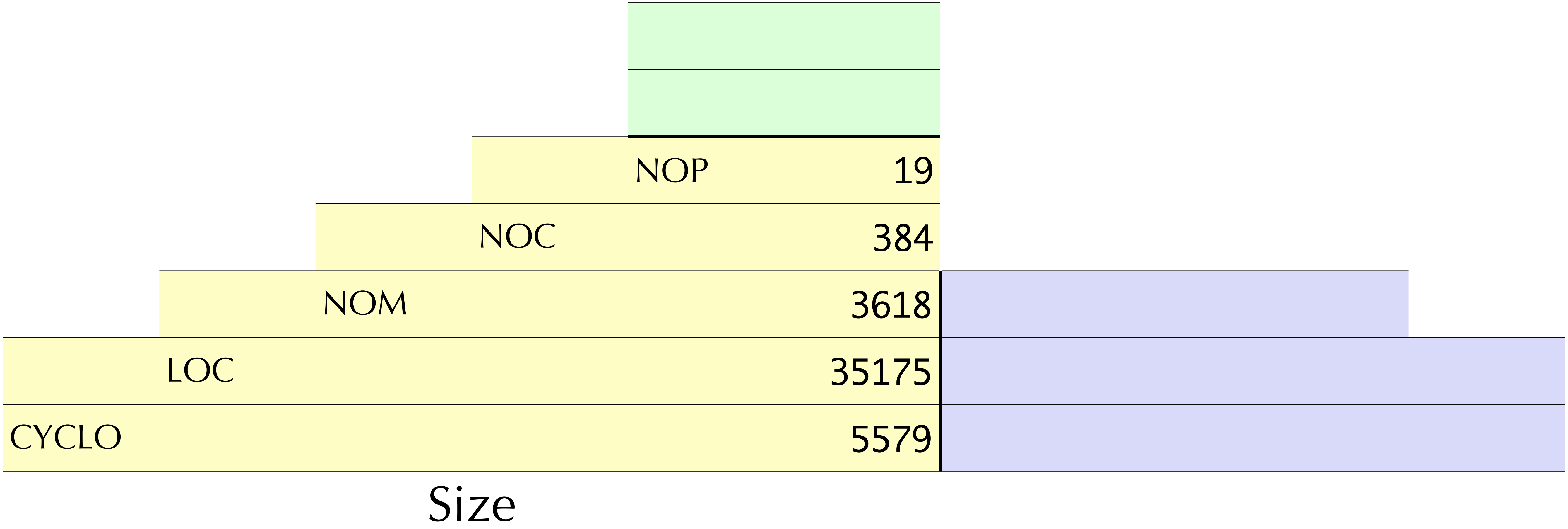
Characterizing Systems with Metrics

The Overview Pyramid provides a metrics overview



Lanza & Marinescu, 2006

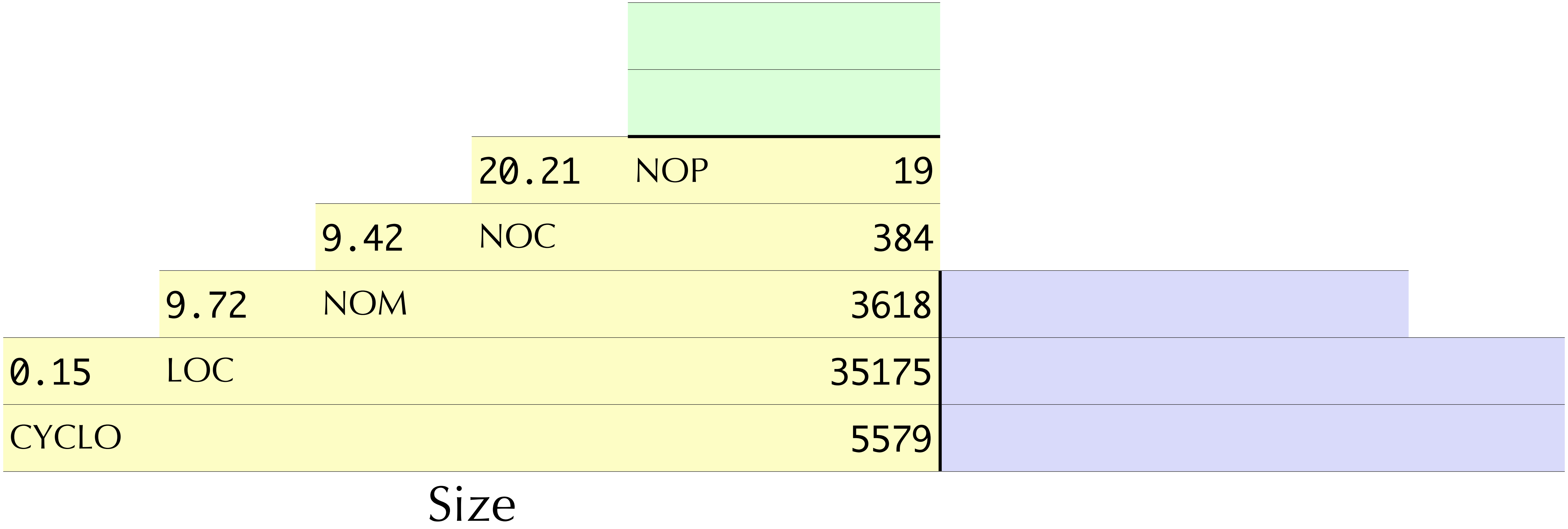
The Overview Pyramid provides a metrics overview



NOP	19
NOC	384
NOM	3618
LOC	35175
CYCLO	5579

Size

The Overview Pyramid provides a metrics overview



20.21 NOP 19

9.42 NOC 384

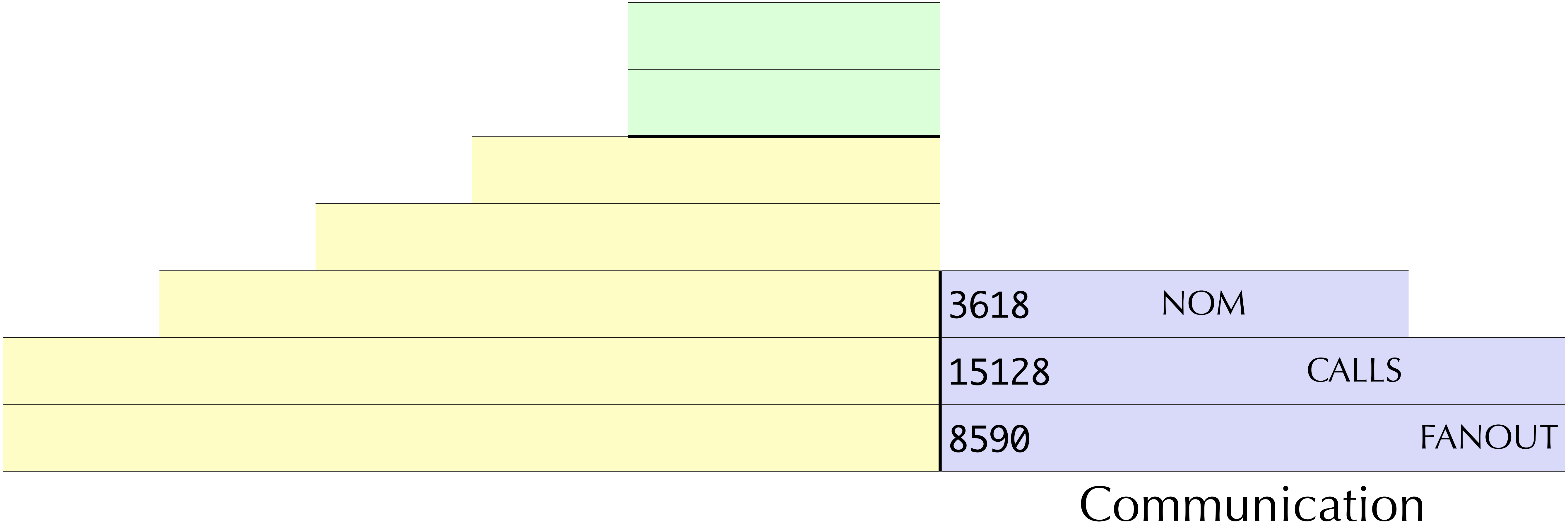
9.72 NOM 3618

0.15 LOC 35175

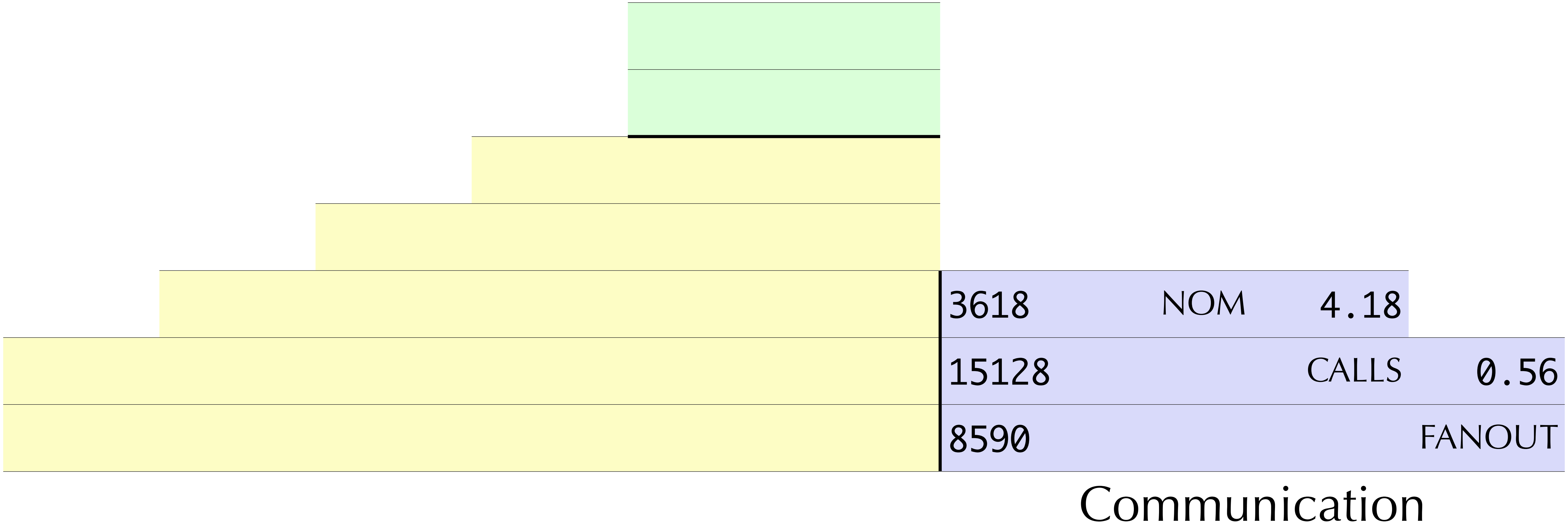
CYCLO 5579

Size

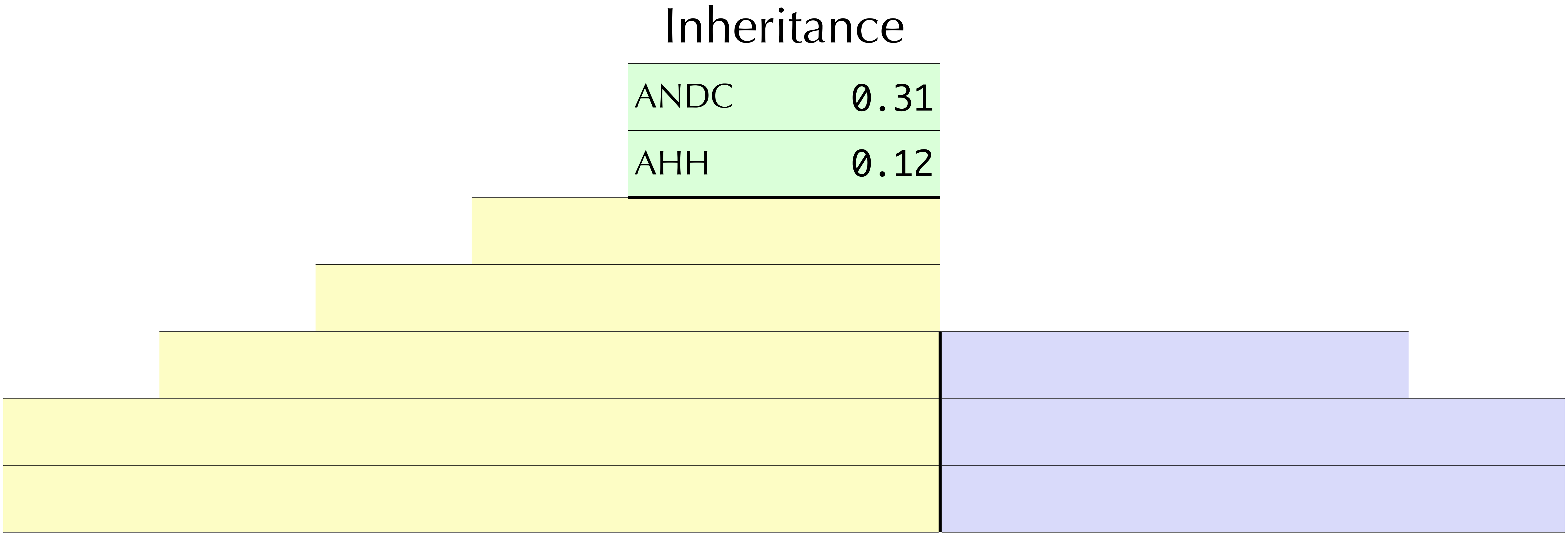
The Overview Pyramid provides a metrics overview



The Overview Pyramid provides a metrics overview



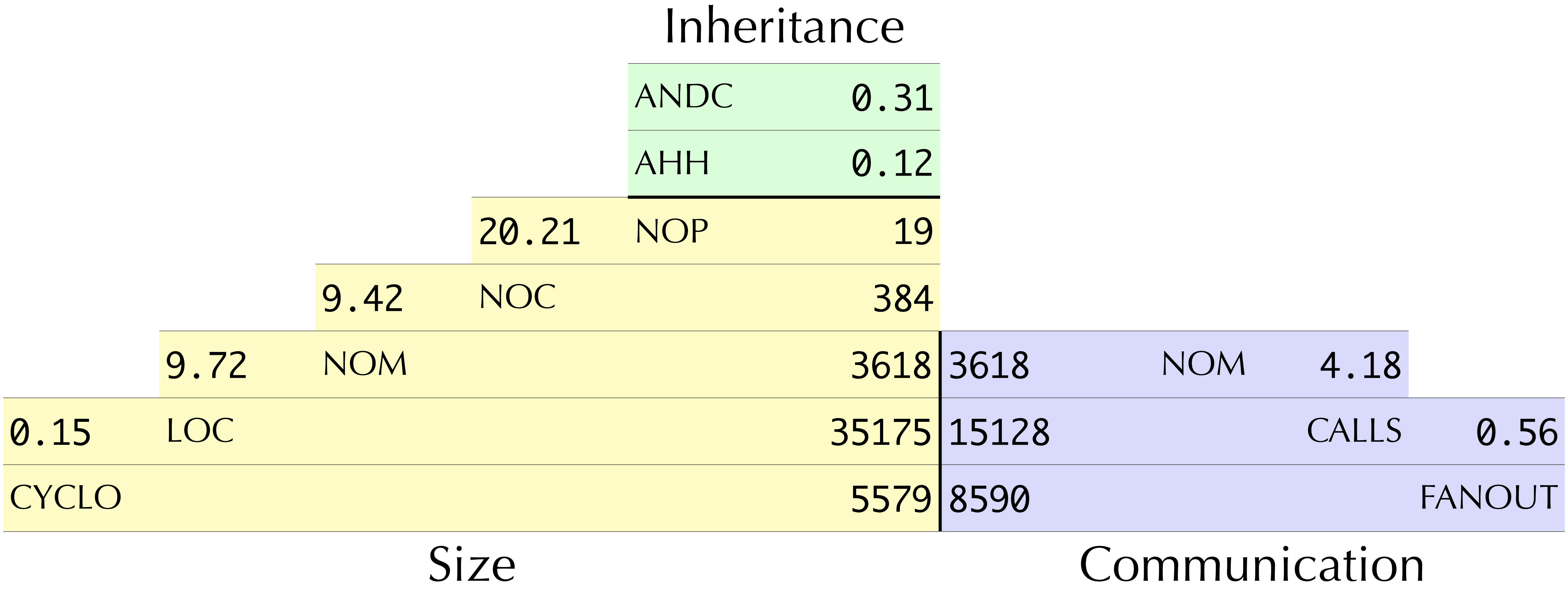
The Overview Pyramid provides a metrics overview



Inheritance

ANDC	0.31
AHH	0.12

The Overview Pyramid provides a metrics overview



Inheritance

ANDC 0.31

AHH 0.12

20.21 NOP 19

9.42 NOC 384

9.72 NOM 3618 | 3618 NOM 4.18

0.15 LOC 35175 | 15128 CALLS 0.56

CYCLO 5579 | 8590 FANOUT

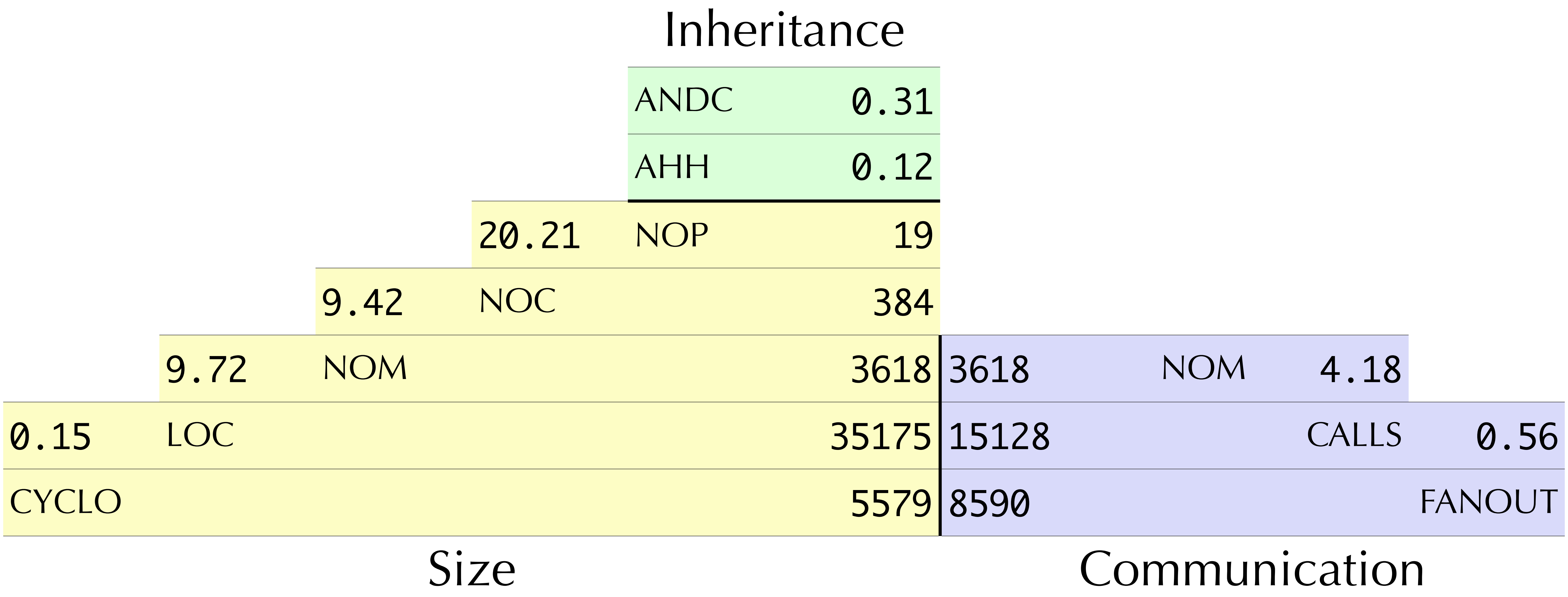
Size

Communication

Obtaining Thresholds

	Java			C++		
	LOW	AVG	HIGH	LOW	AVG	HIGH
CYCLO/LOC	0.16	0.20	0.24	0.20	0.25	0.30
LOC/NOM	7	10	13	5	10	16
NOM/NOC	4	7	10	4	9	15
...						

The Overview Pyramid provides a metrics overview



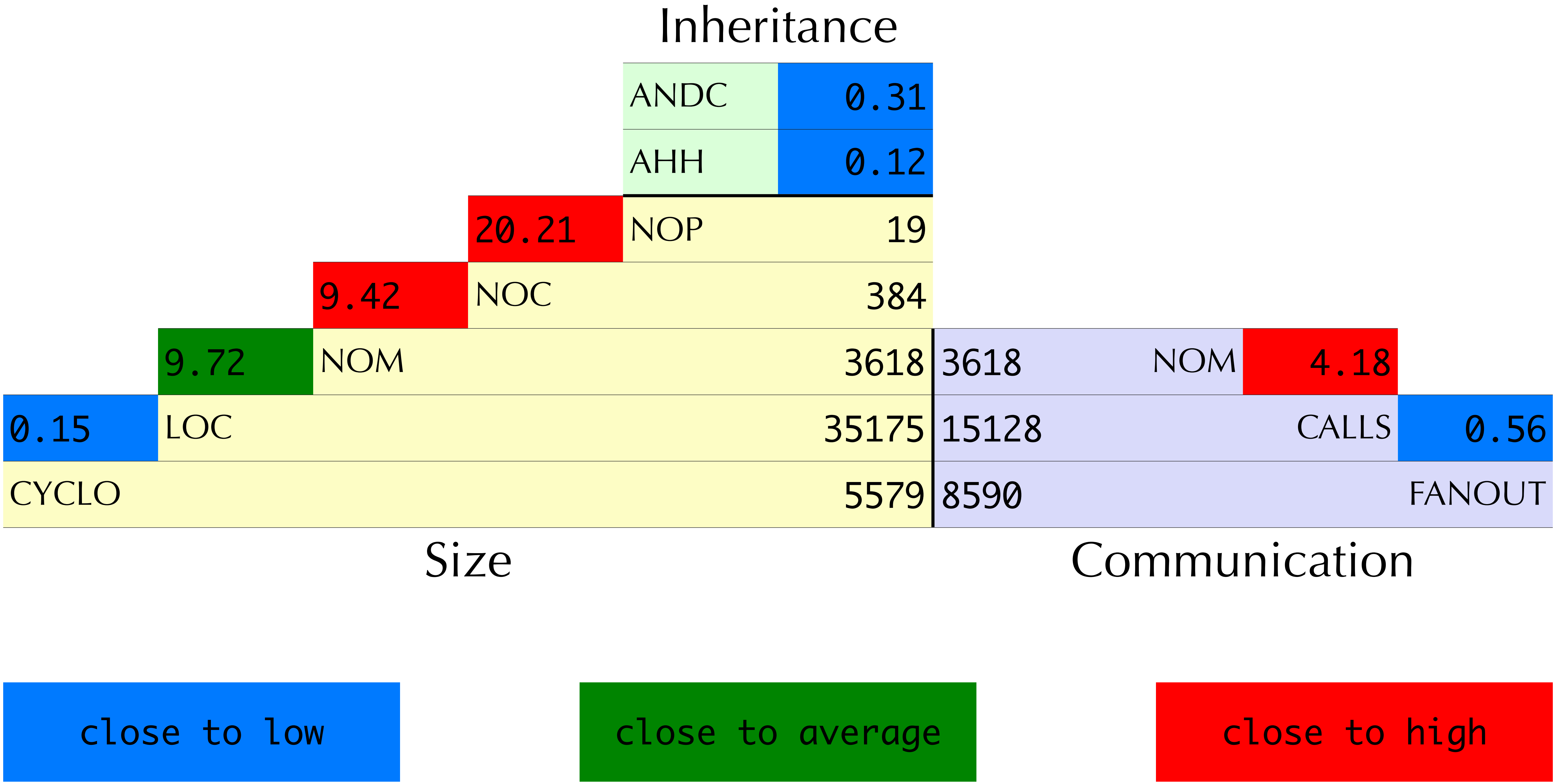
Inheritance

				ANDC	0.31			
				AHH	0.12			
			20.21	NOP	19			
		9.42	NOC		384			
	9.72	NOM			3618	3618	NOM	4.18
0.15	LOC				35175	15128	CALLS	0.56
CYCLO					5579	8590		FANOUT

Size

Communication

The Overview Pyramid provides a metrics overview



Inheritance

			ANDC	🟦 0.31			
			AHH	🟦 0.12			
		🟥 20.21	NOP	19			
	🟥 9.42	NOC		384			
🟩 9.72	NOM			3618	3618	NOM	🟥 4.18
🟦 0.15	LOC			35175	15128	CALLS	🟦 0.56
CYCLO				5579	8590	FANOUT	

Size

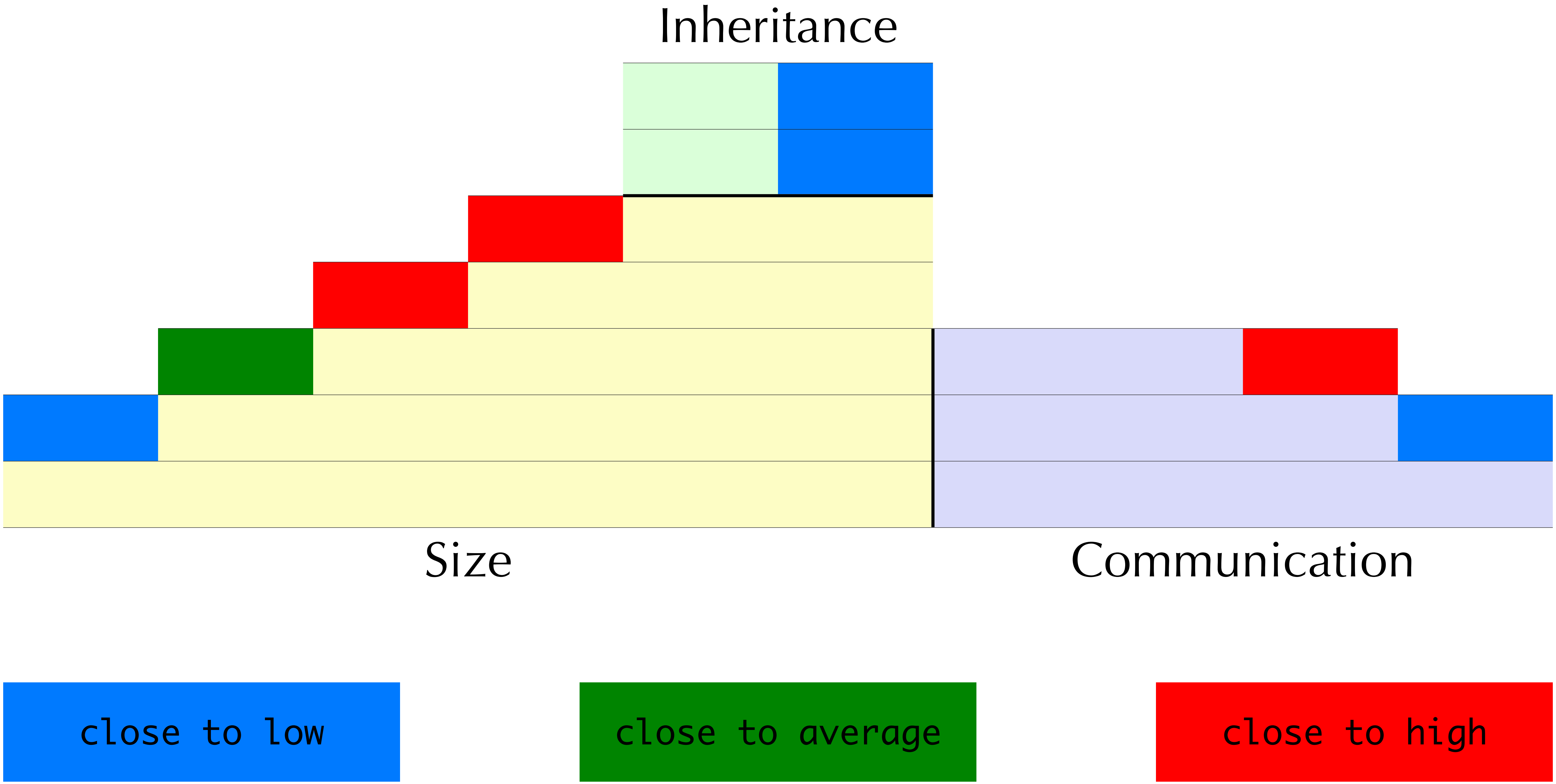
Communication

🟦 close to low

🟩 close to average

🟥 close to high

The Overview Pyramid provides a metrics overview



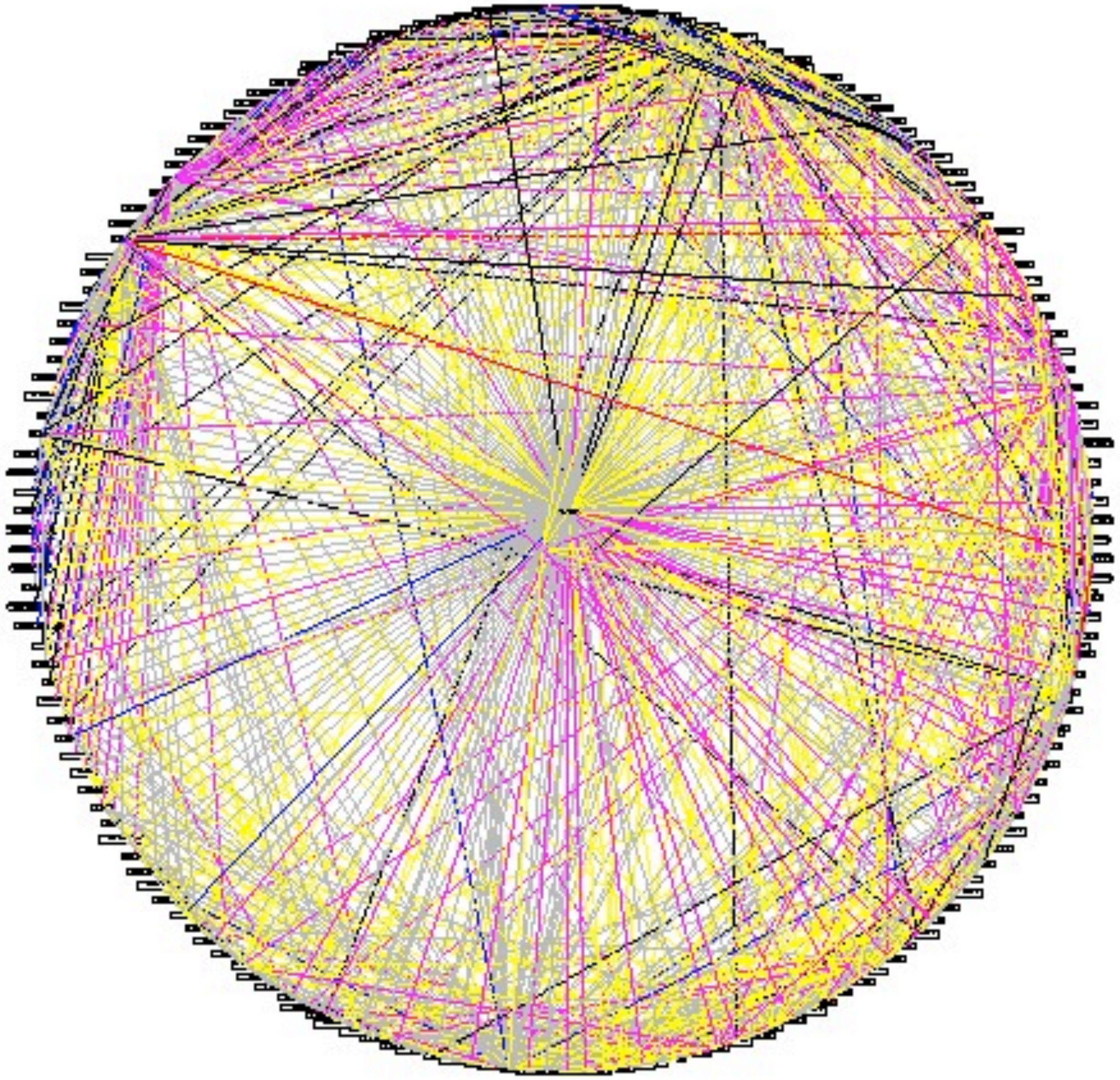
How do I improve my code?

- Quality is more than zero bugs
- Quality is about design principles, design heuristics, and best practices
- Breaking them leads to
  - Code deterioration
  - Design problems ~ Maintenance problems

Imagine...

You change a small design fragment...

...and one third of all classes require changes!



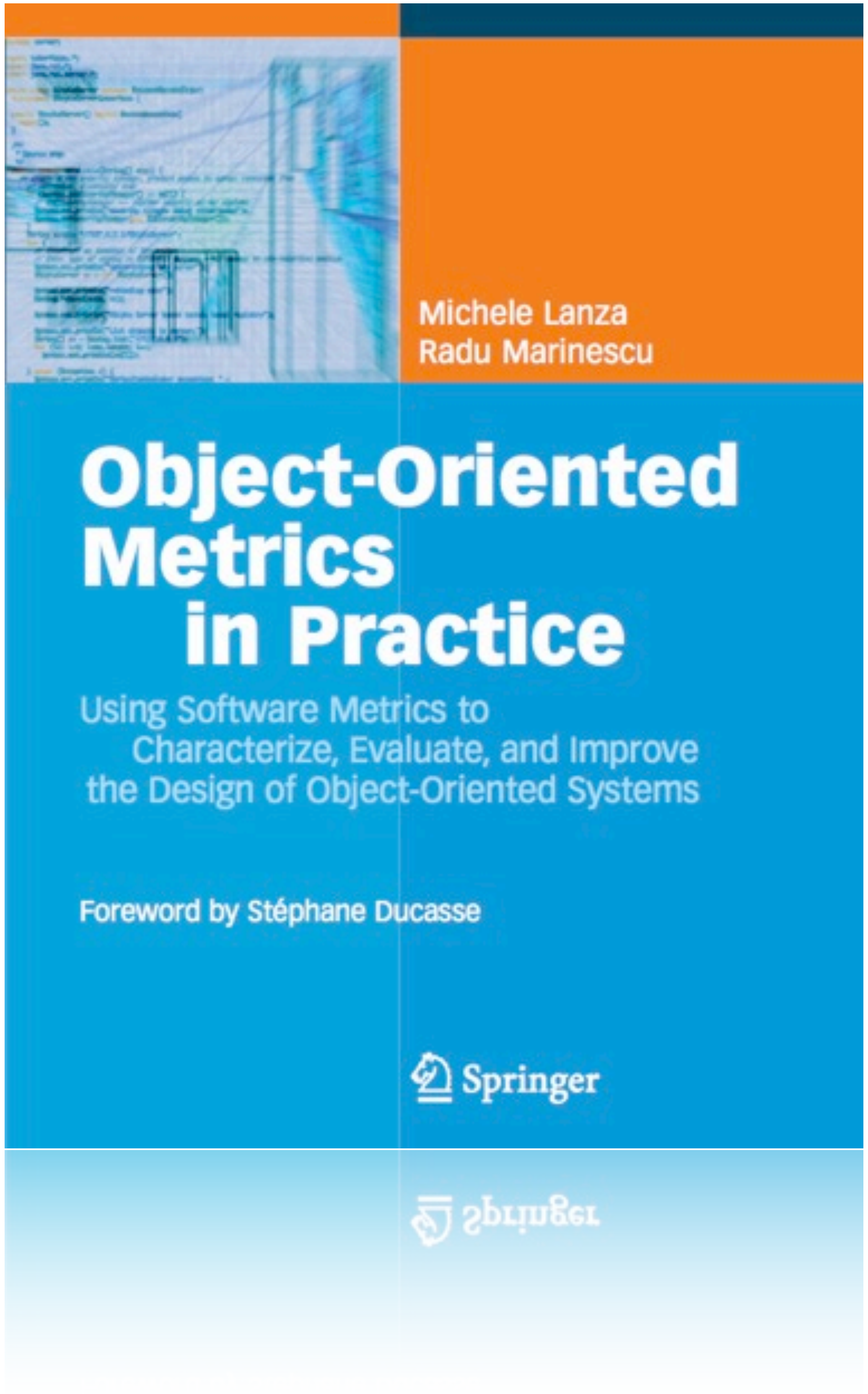
Design Problems

- Expensive
- Frequent
- Unavoidable
- How can we detect and eliminate them?

Reference

M. Lanza, R. Marinescu
"Object-Oriented Metrics in Practice"

Springer, 2006
ISBN 3-540-24429-8



Michele Lanza
Radu Marinescu

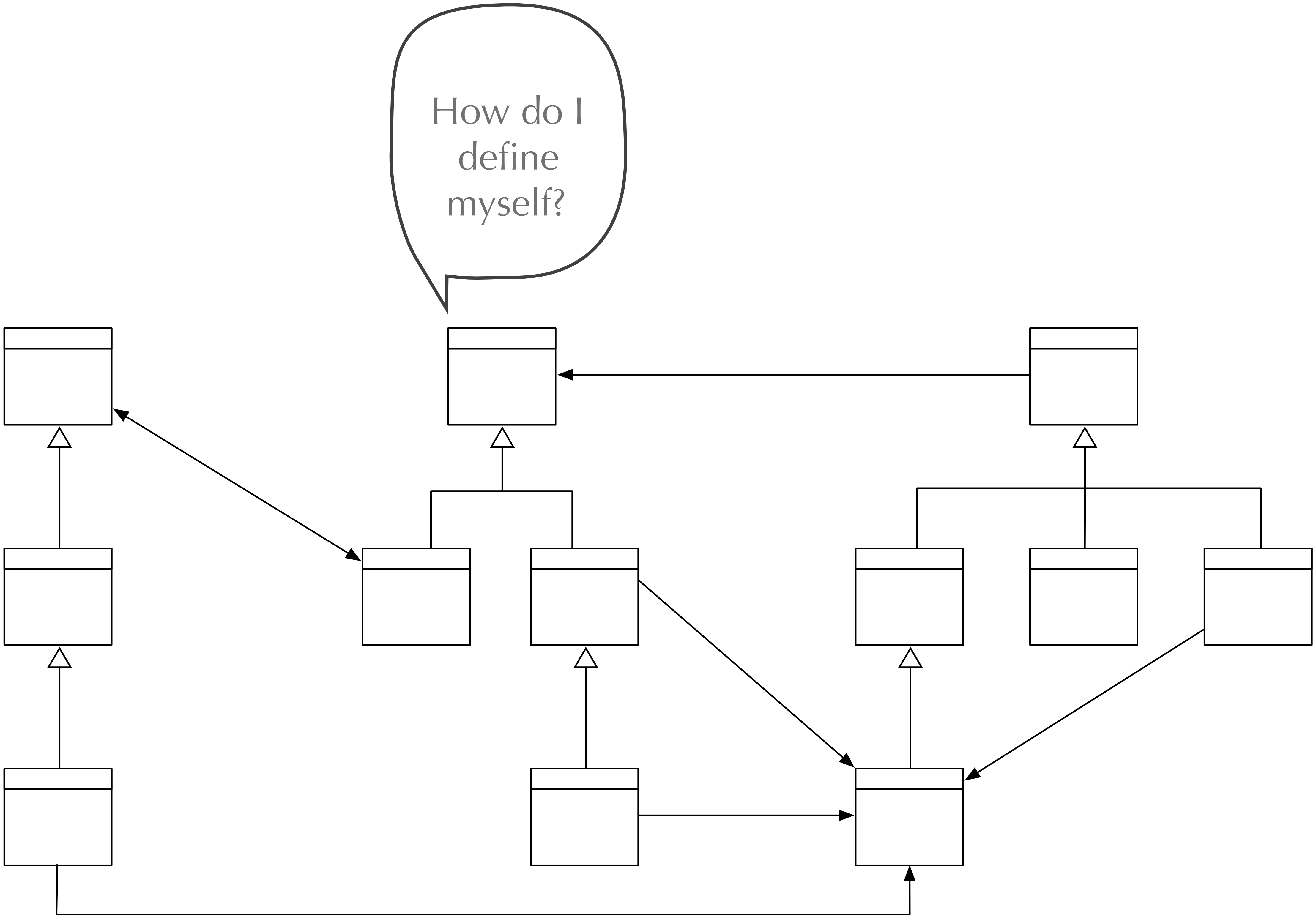
Object-Oriented Metrics in Practice

Using Software Metrics to Characterize, Evaluate, and Improve the Design of Object-Oriented Systems

Foreword by Stéphane Ducasse

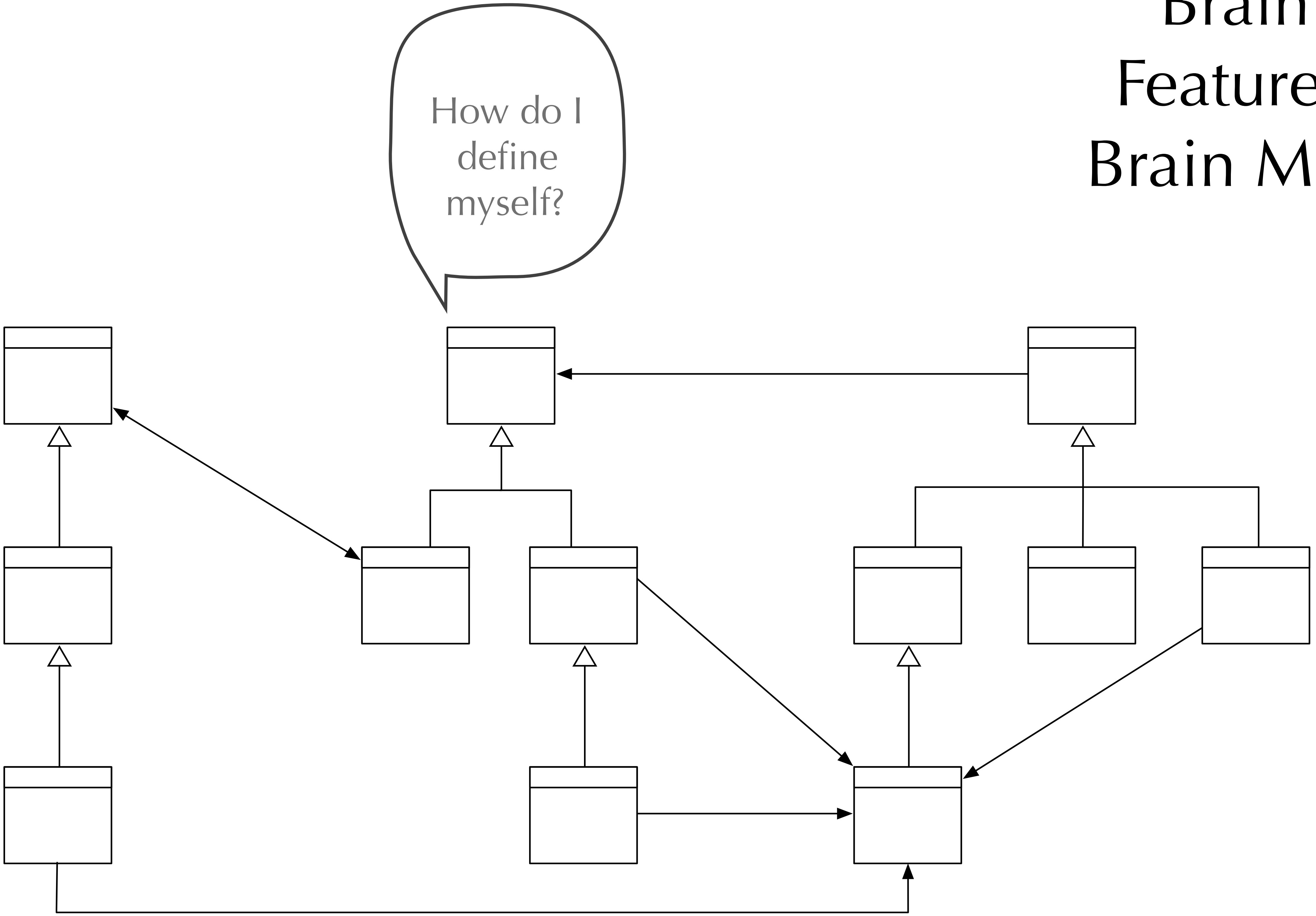
Springer

Identity Disharmony

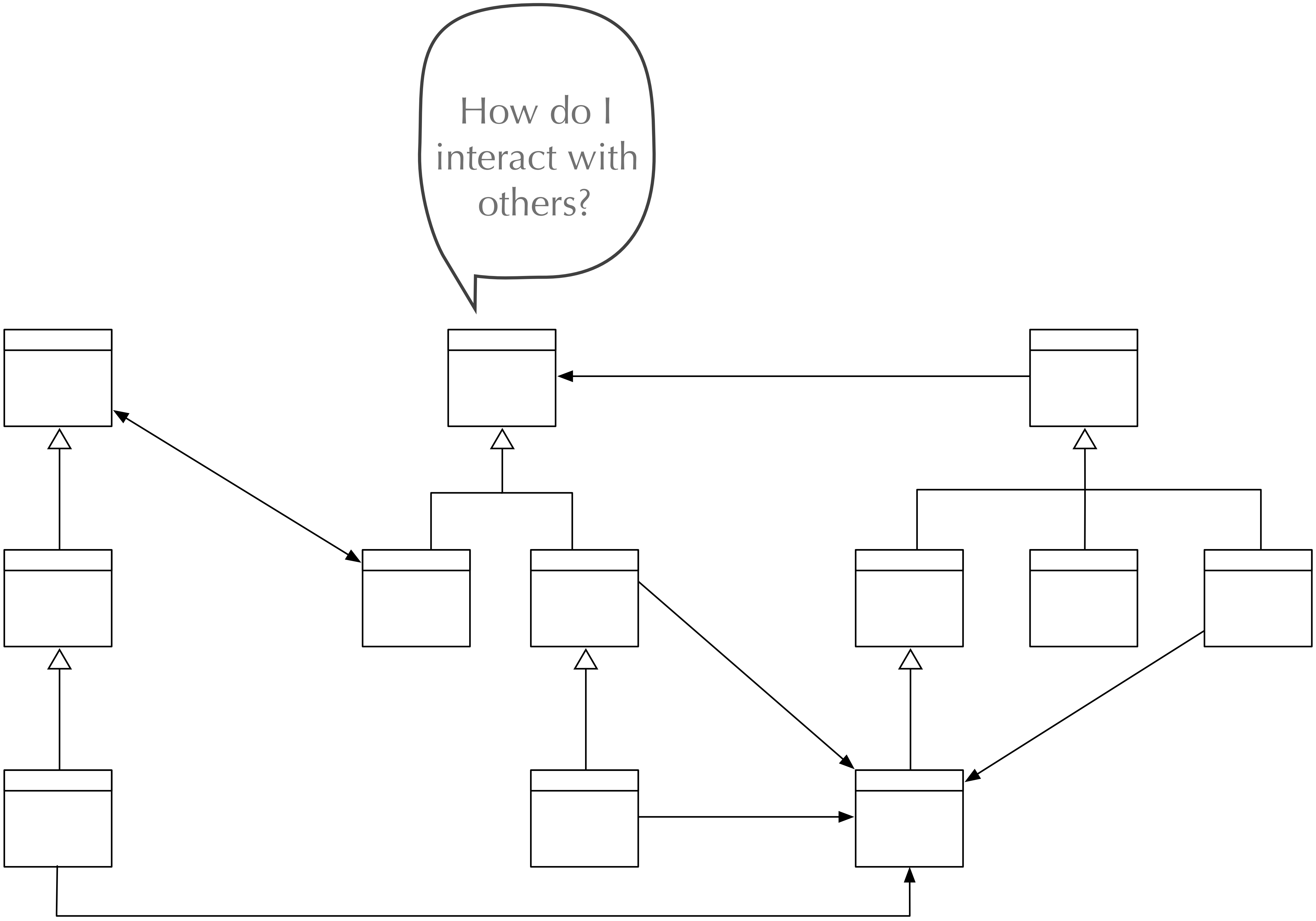


Identity Disharmony

God Class
Data Class
Brain Class
Feature Envy
Brain Method

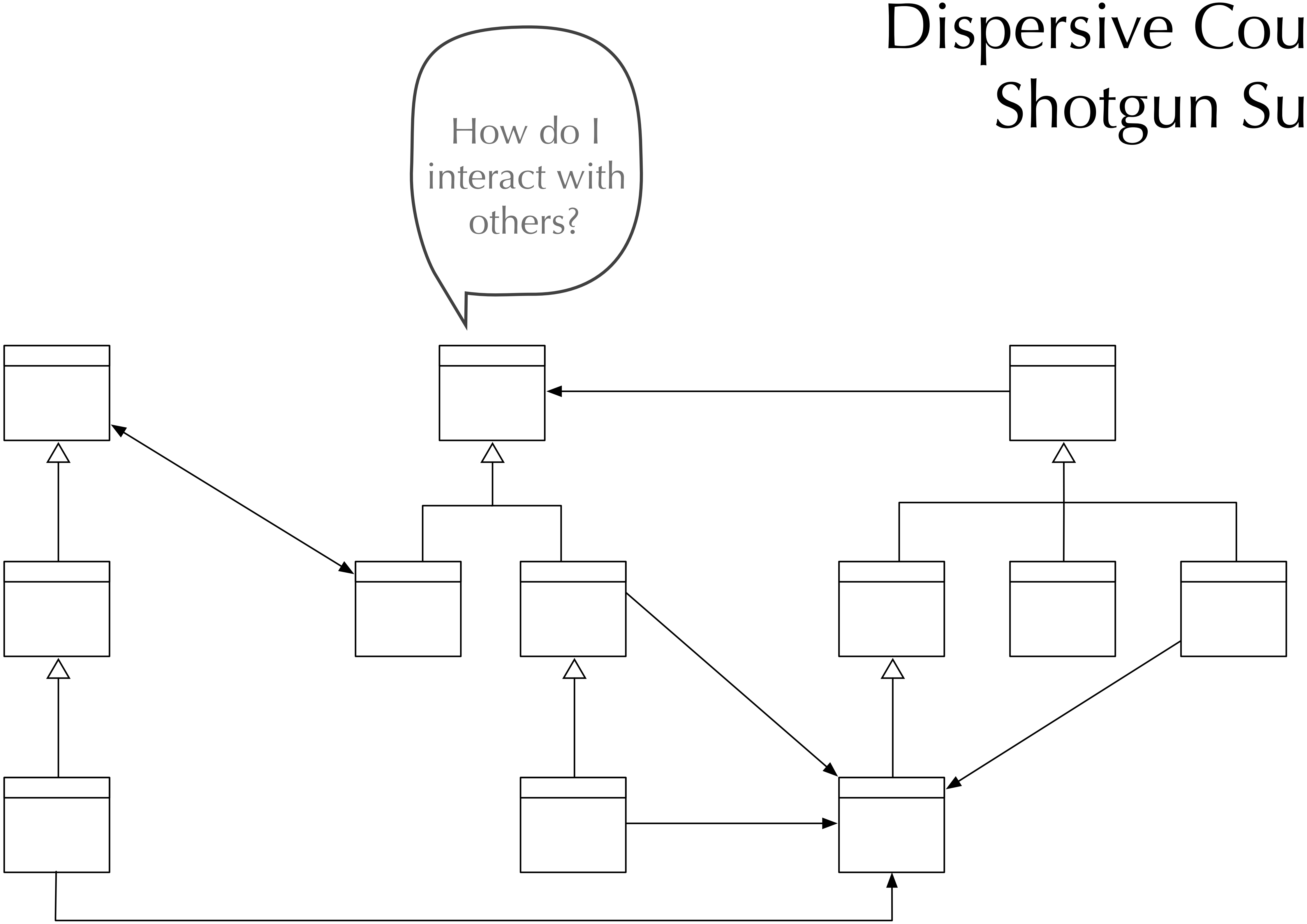


Collaboration Disharmony

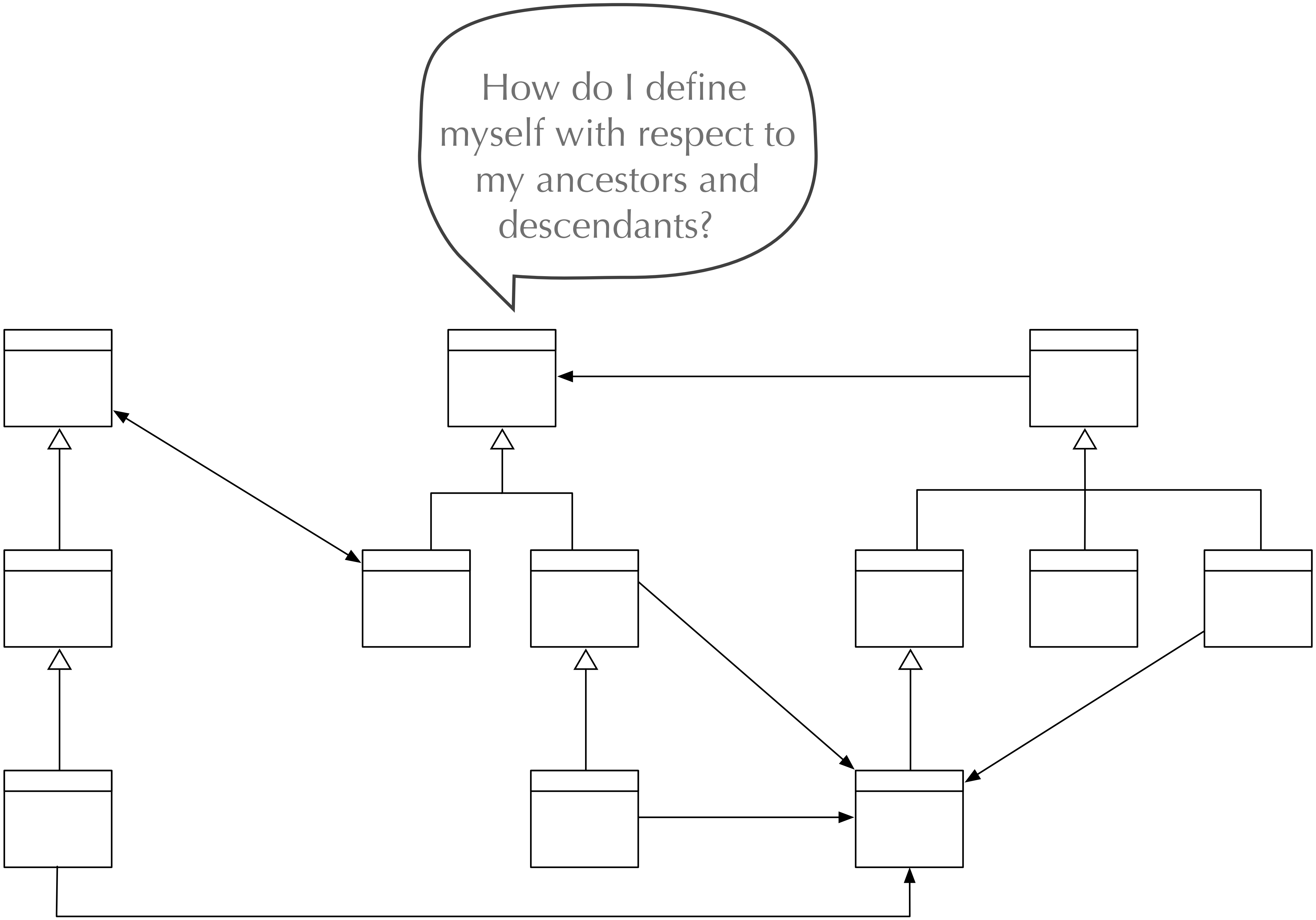


Collaboration Disharmony

Intensive Coupling
Dispersive Coupling
Shotgun Surgery



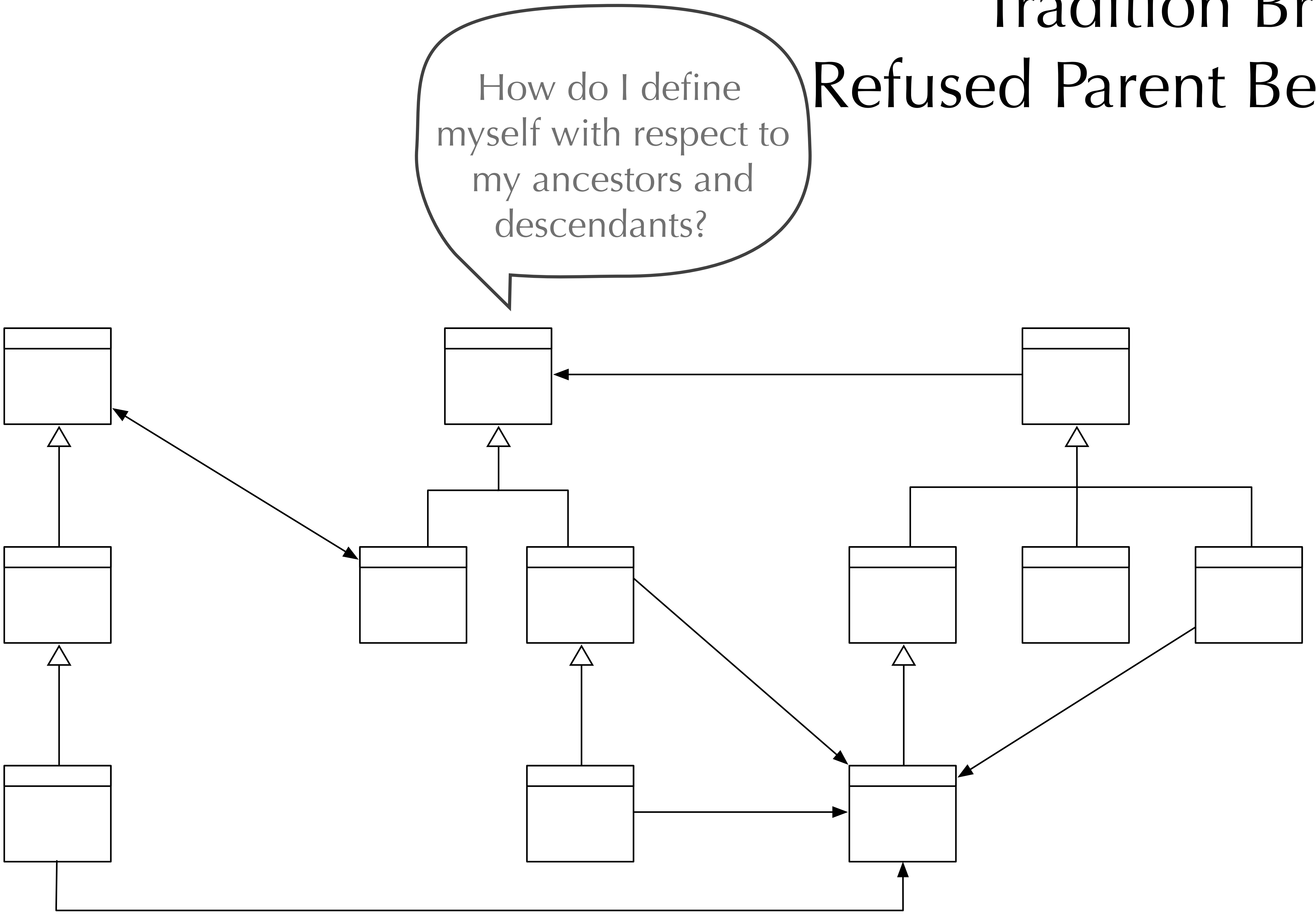
Classification Disharmony



How do I define myself with respect to my ancestors and descendants?

Classification Disharmony

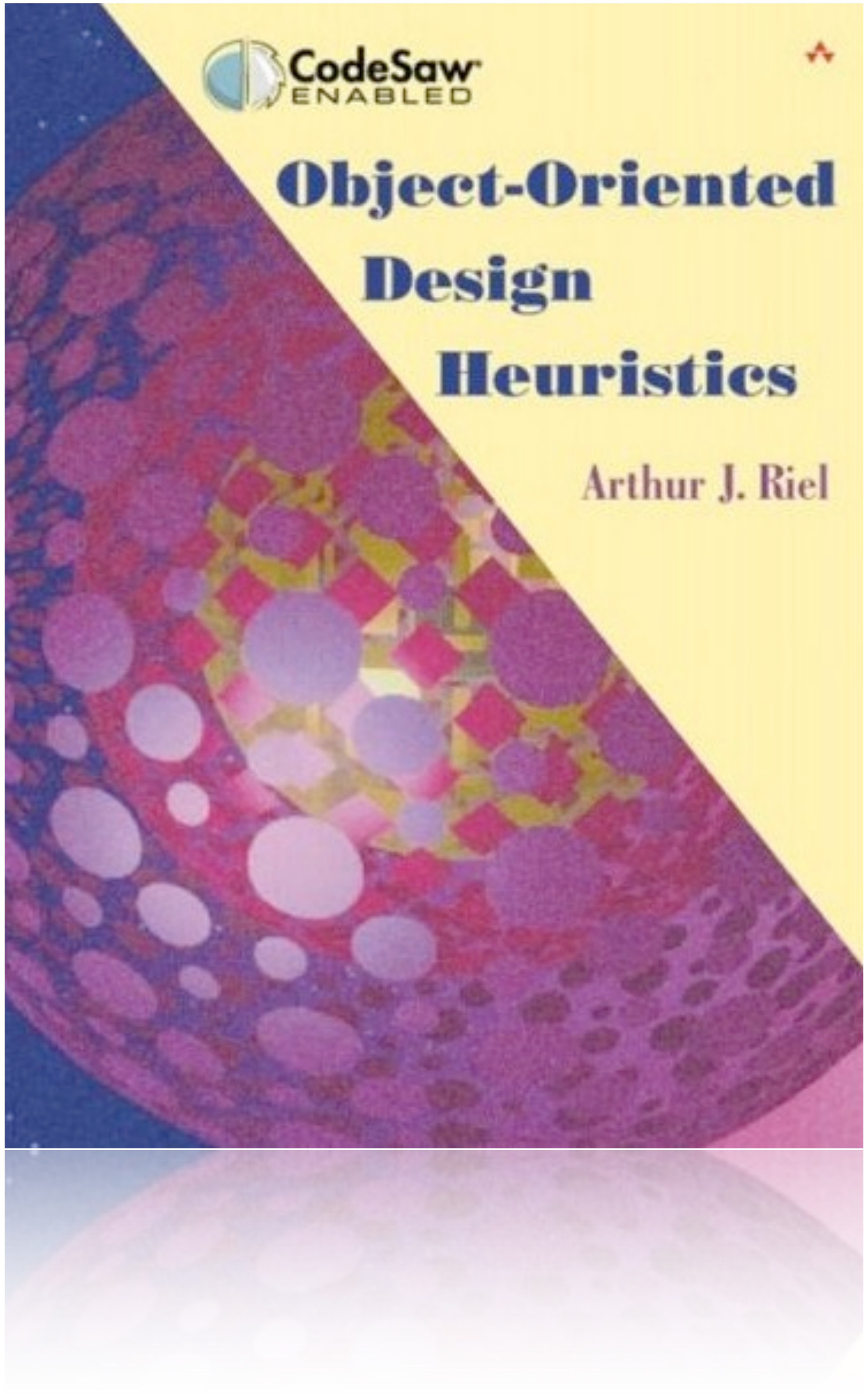
Futile Hierarchy
Tradition Breaker
Refused Parent Bequest


How do I define myself with respect to my ancestors and descendants?

God Class

"In a good object-oriented design the intelligence of a system is uniformly distributed among the top-level classes."

Arthur Riel, 1996



God Classes

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes
- God Classes tend

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes
- God Classes tend
  - to centralize the intelligence of the system

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes
- God Classes tend
  - to centralize the intelligence of the system
  - to do everything and

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes
- God Classes tend
  - to centralize the intelligence of the system
  - to do everything and
  - to use data from small data-classes

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes
- God Classes tend
  - to centralize the intelligence of the system
  - to do everything and
  - to use data from small data-classes
- God Classes

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes
- God Classes tend
  - to centralize the intelligence of the system
  - to do everything and
  - to use data from small data-classes
- God Classes
  - centralize the intelligence of the system

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes
- God Classes tend
  - to centralize the intelligence of the system
  - to do everything and
  - to use data from small data-classes
- God Classes
  - centralize the intelligence of the system
  - do everything

God Classes

- God Classes tend to centralize the intelligence of the system, to do everything and to use data from small data-classes
- God Classes tend
  - to centralize the intelligence of the system
  - to do everything and
  - to use data from small data-classes
- God Classes
  - centralize the intelligence of the system
  - do everything
  - use data from small data-classes

God Classes

God Classes

- God Classes
  - centralize the intelligence of the system
  - do everything
  - use data from small data-classes

God Classes

- God Classes
  - centralize the intelligence of the system
  - do everything
  - use data from small data-classes
- God Classes
  - are complex: high WMC
  - are not cohesive: low TCC
  - access external data: ATFD

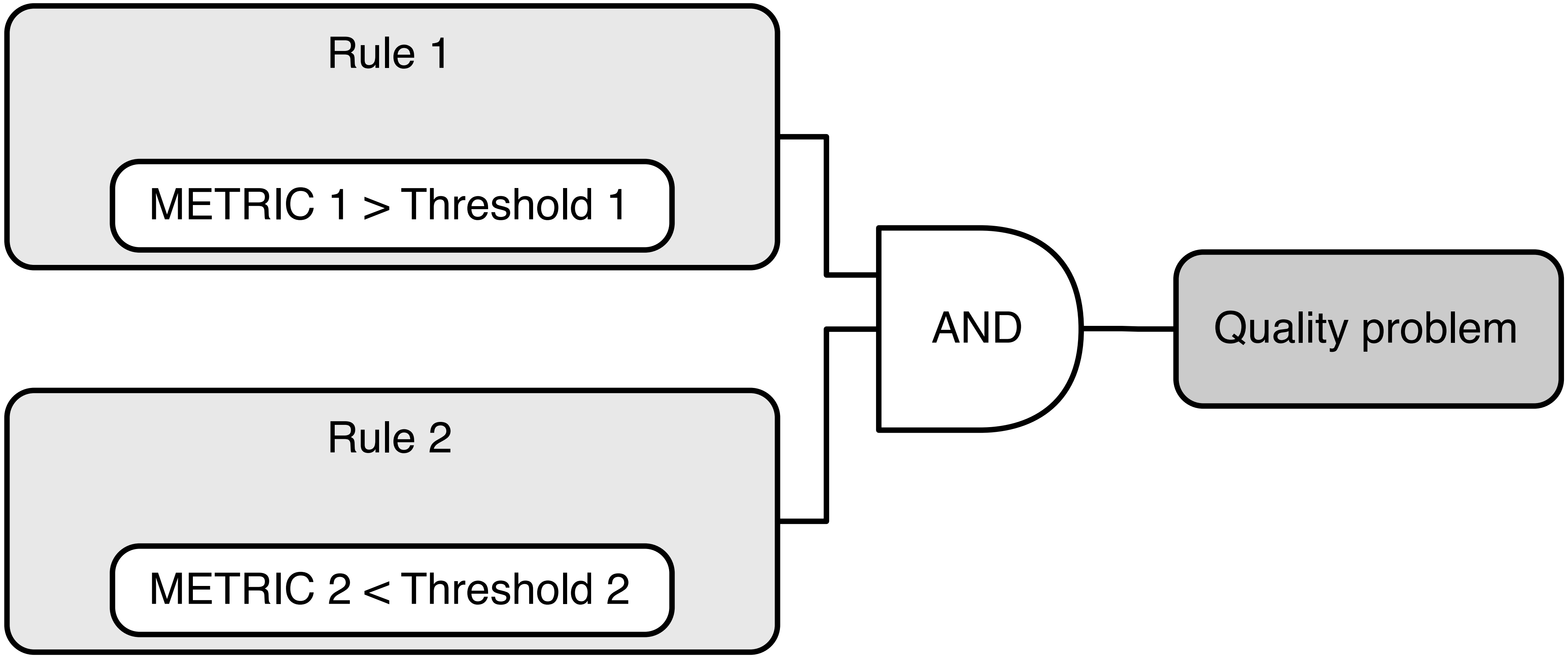
God Classes

- God Classes
  - centralize the intelligence of the system
  - do everything
  - use data from small data-classes
- God Classes
  - are complex: high WMC
  - are not cohesive: low TCC
  - access external data: ATFD

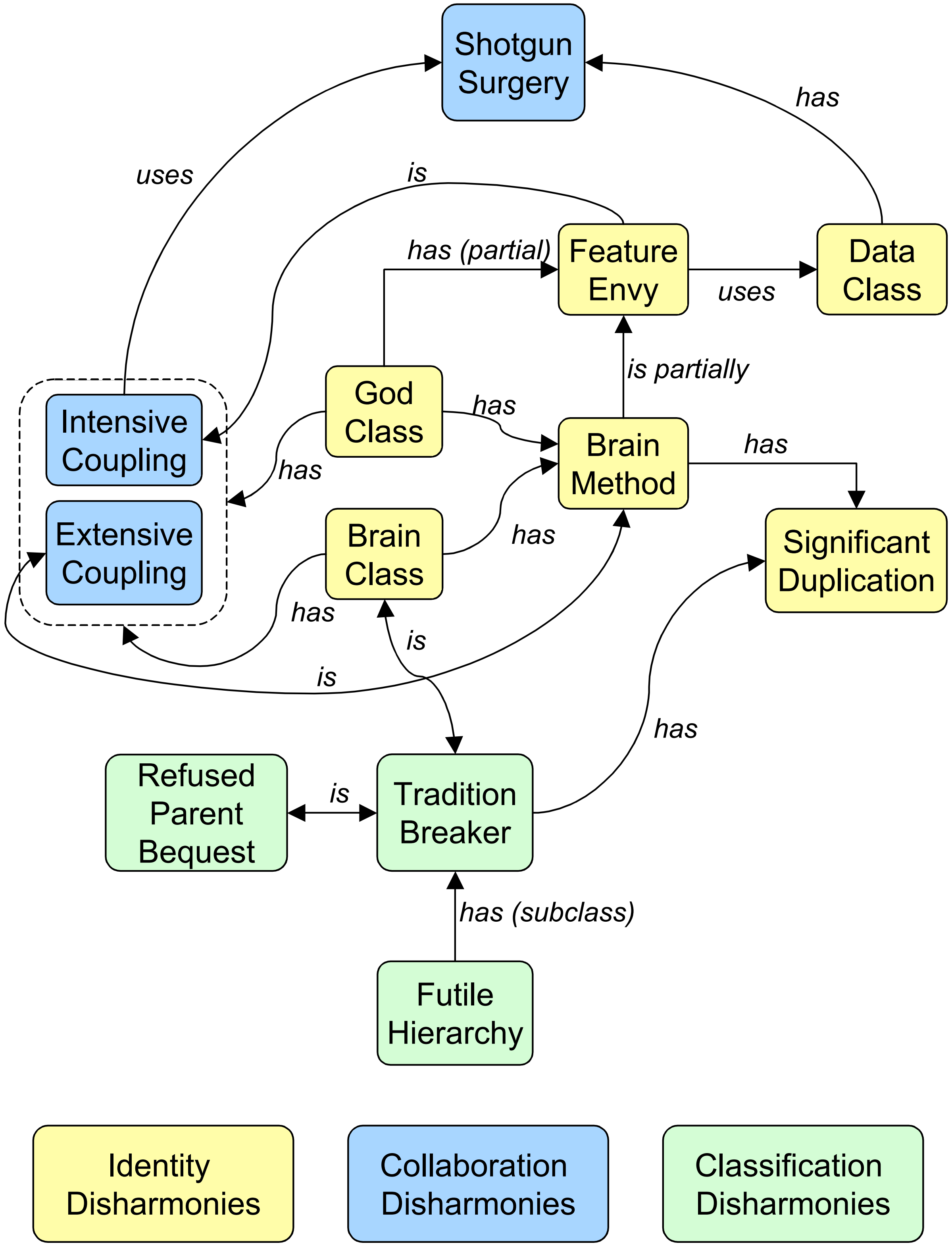
Compose metrics into queries
using logical operators

Detection Strategies

- Detection strategies are metric-based queries to detect design flaws



Design Flaws do not come alone



Shotgun Surgery

uses

is

has

has (partial)

Feature Envy

uses

Data Class

is partially

Intensive Coupling

God Class

has

has

Brain Method

has

Extensive Coupling

Brain Class

has

Significant Duplication

has

is

is

has

Refused Parent Bequest

is

Tradition Breaker

has (subclass)

Futile Hierarchy

Identity Disharmonies

Collaboration Disharmonies

Classification Disharmonies

Characteristics of a God Class

Characteristics of a God Class

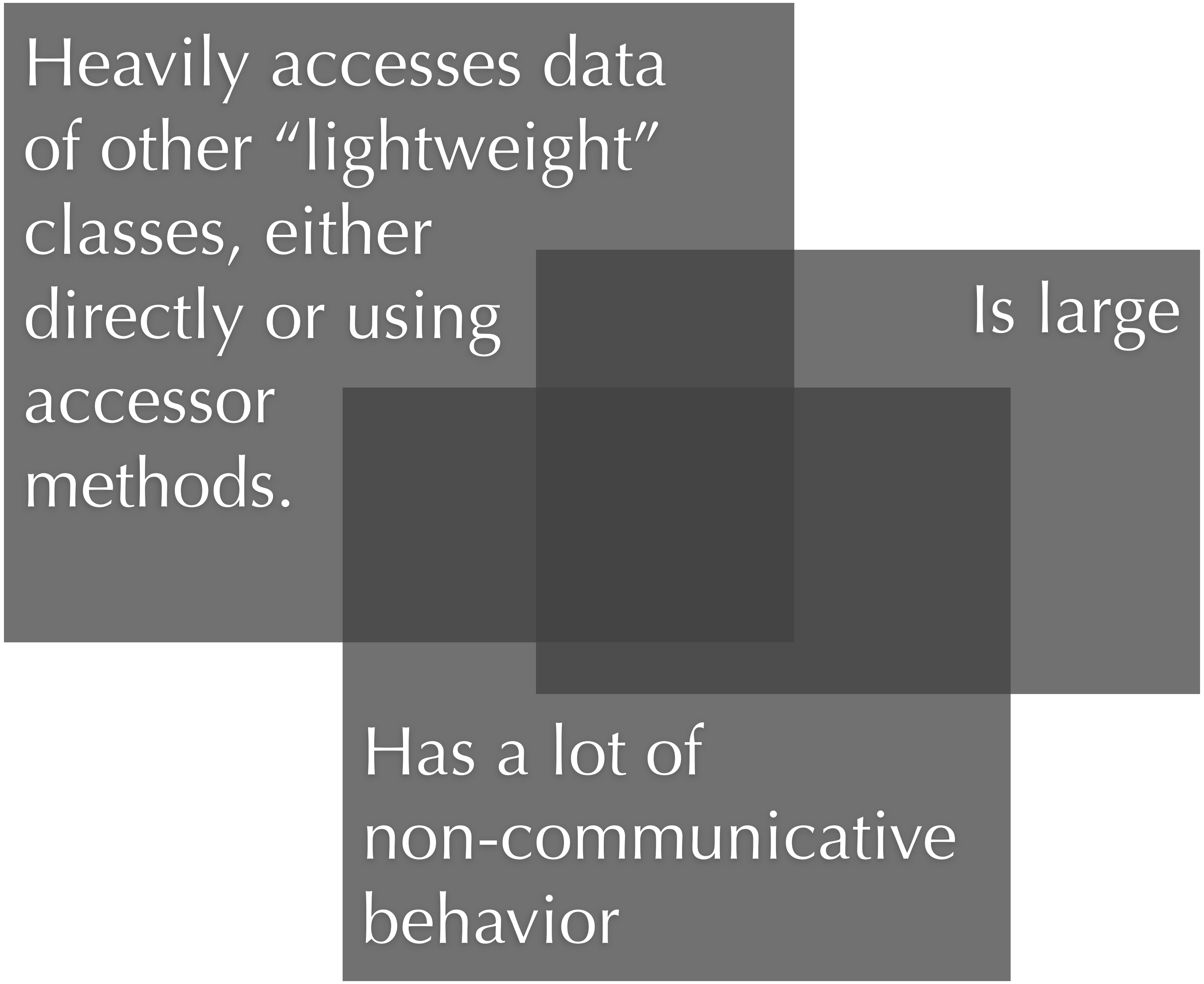
Heavily accesses data of other “lightweight” classes, either directly or using accessor methods.

Characteristics of a God Class

Heavily accesses data of other “lightweight” classes, either directly or using accessor methods.

Is large

Characteristics of a God Class

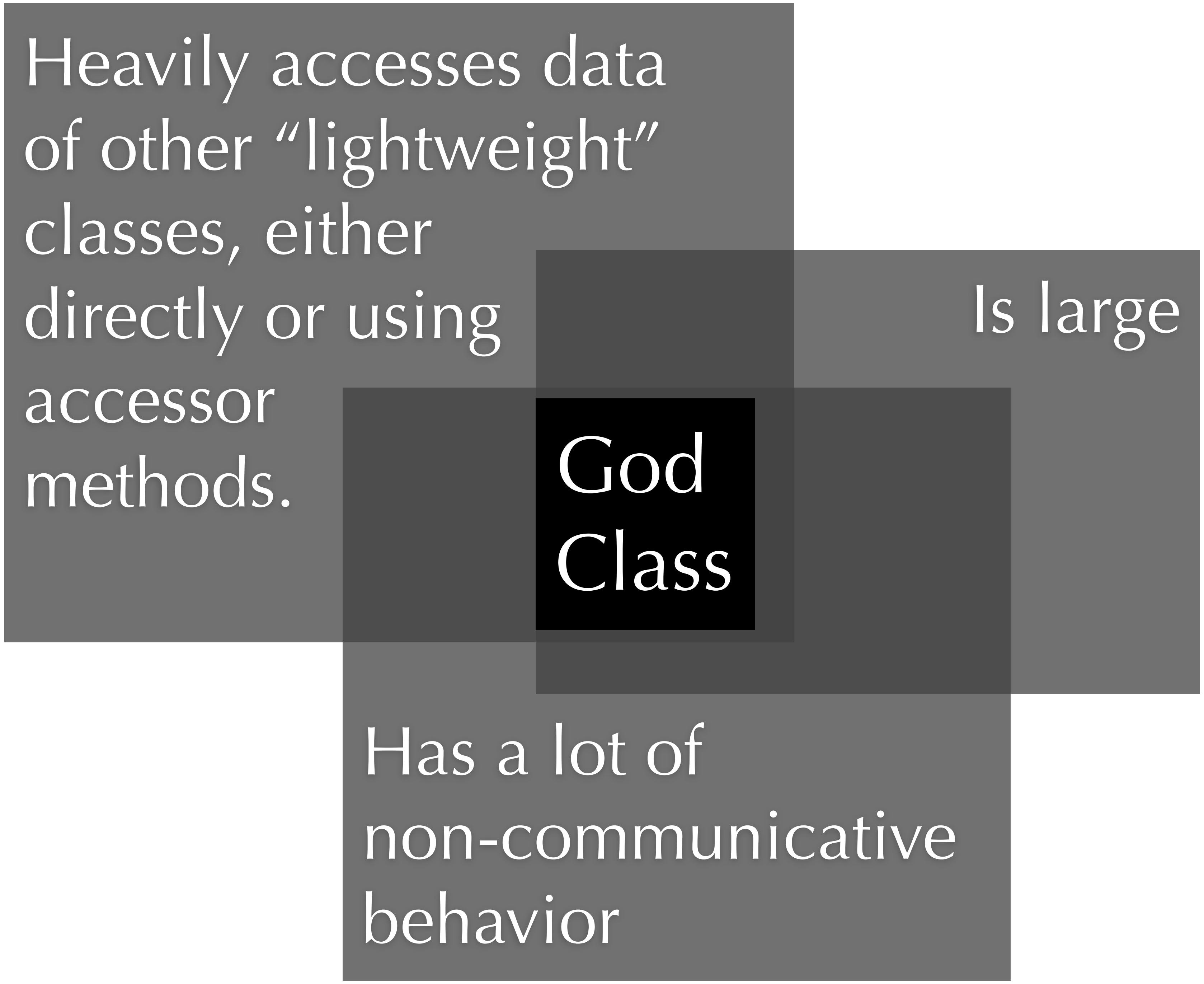


Heavily accesses data of other “lightweight” classes, either directly or using accessor methods.

Is large

Has a lot of non-communicative behavior

Characteristics of a God Class



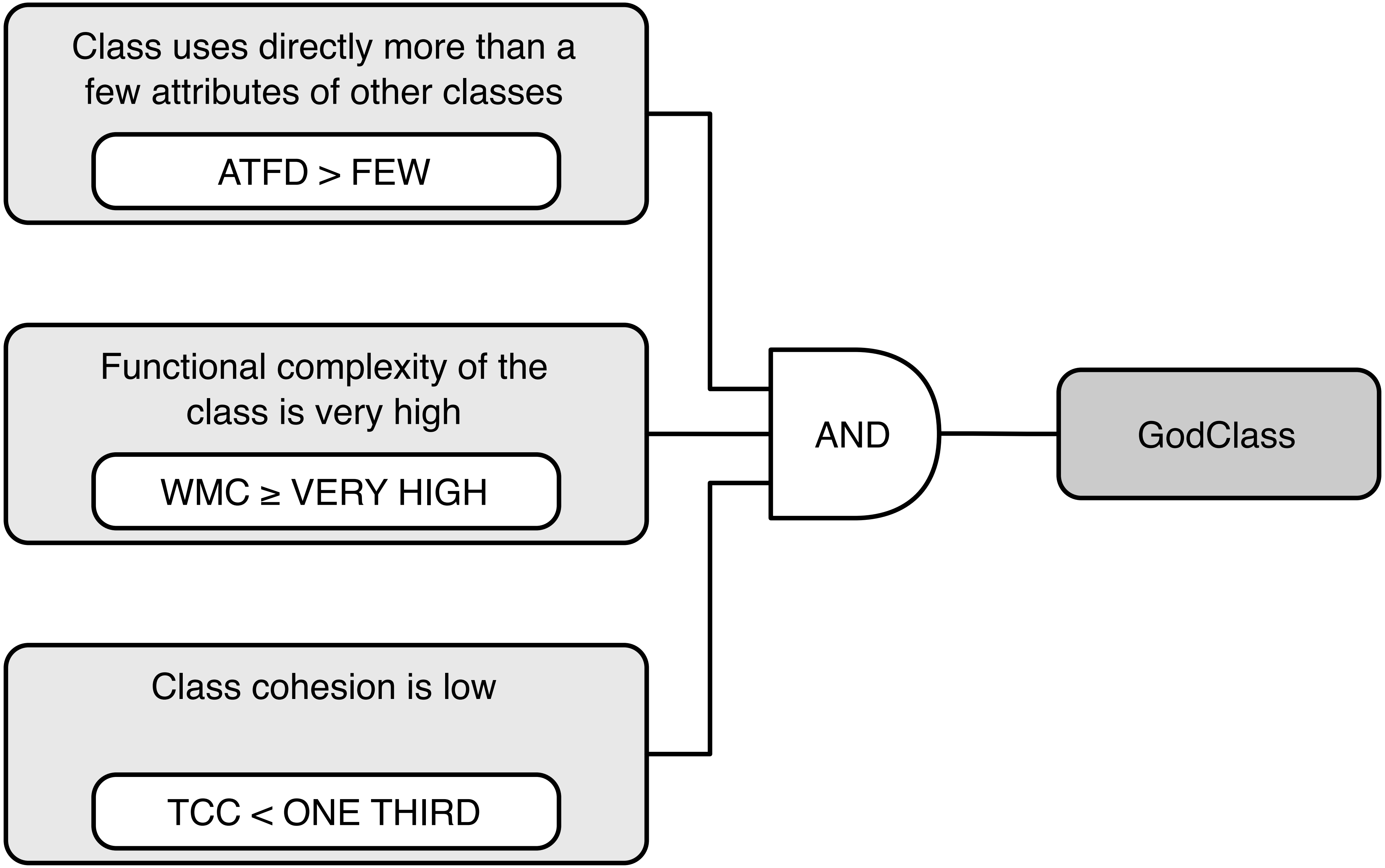
Heavily accesses data of other “lightweight” classes, either directly or using accessor methods.

Is large

Has a lot of non-communicative behavior

God Class

God Class Detection Strategy



Class uses directly more than a few attributes of other classes

ATFD > FEW

Functional complexity of the class is very high

WMC ≥ VERY HIGH

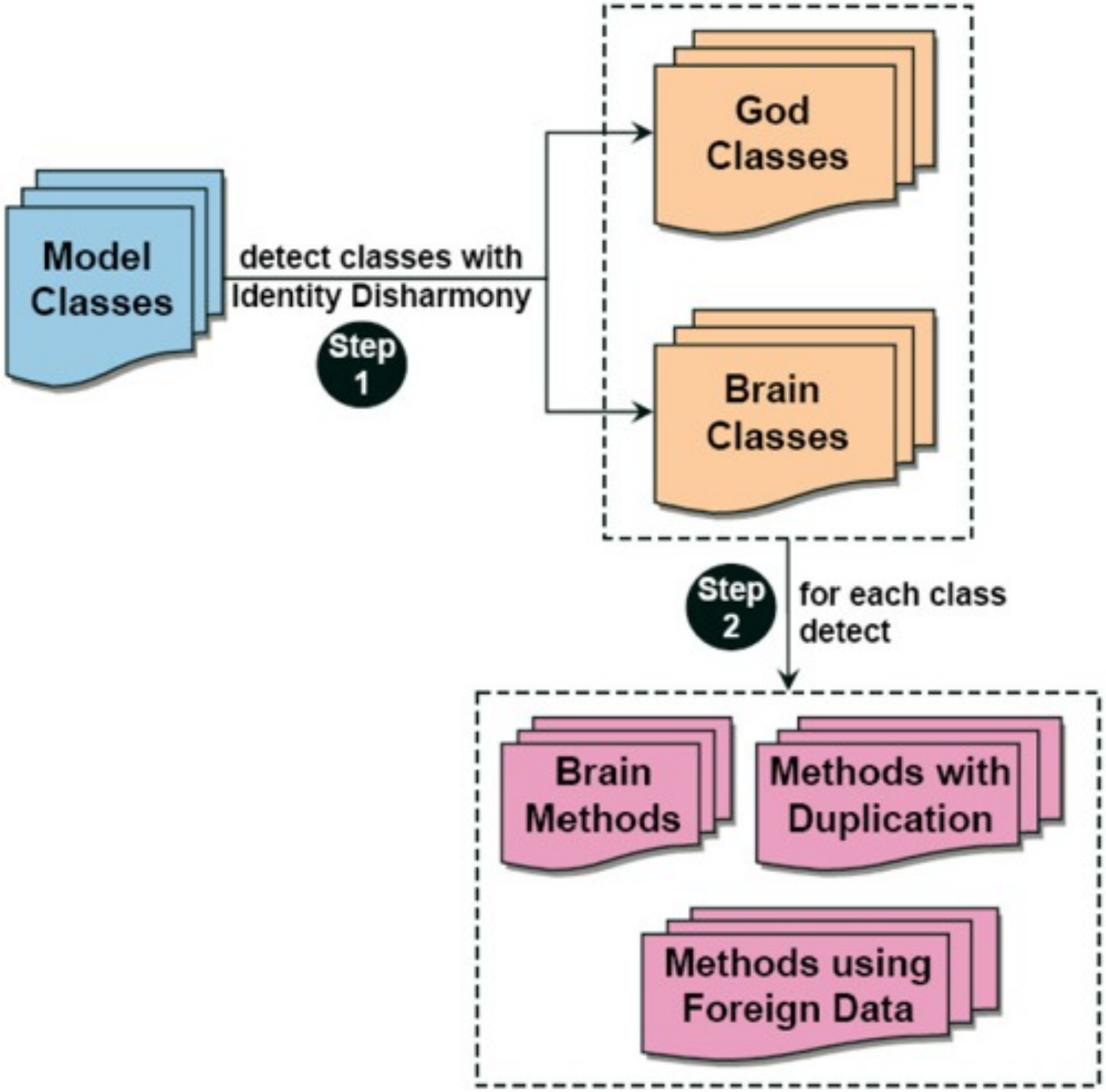
Class cohesion is low

TCC < ONE THIRD

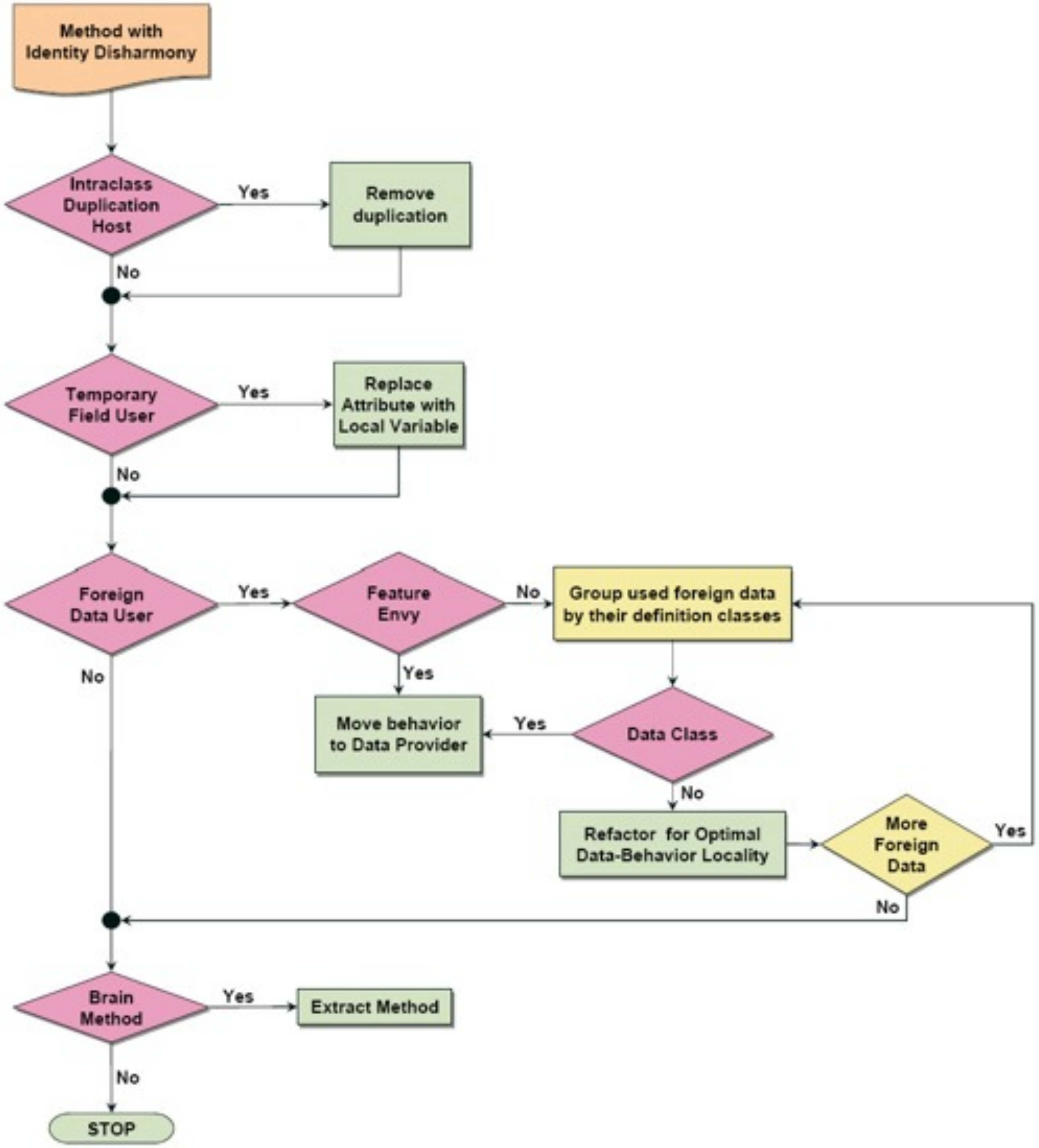
And Now?

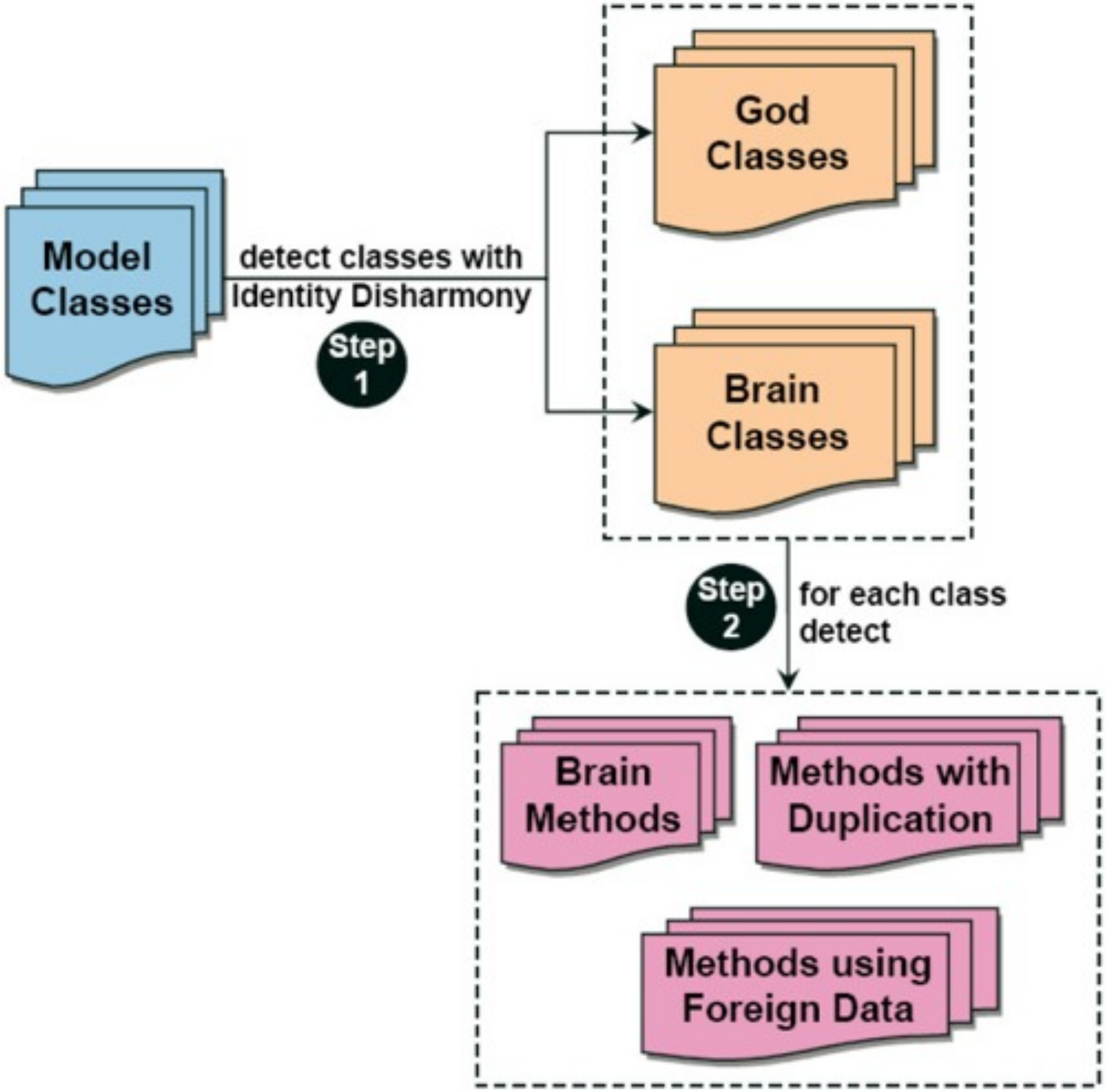
Follow A Clear and Repeatable Process

Follow A Clear and Repeatable Process

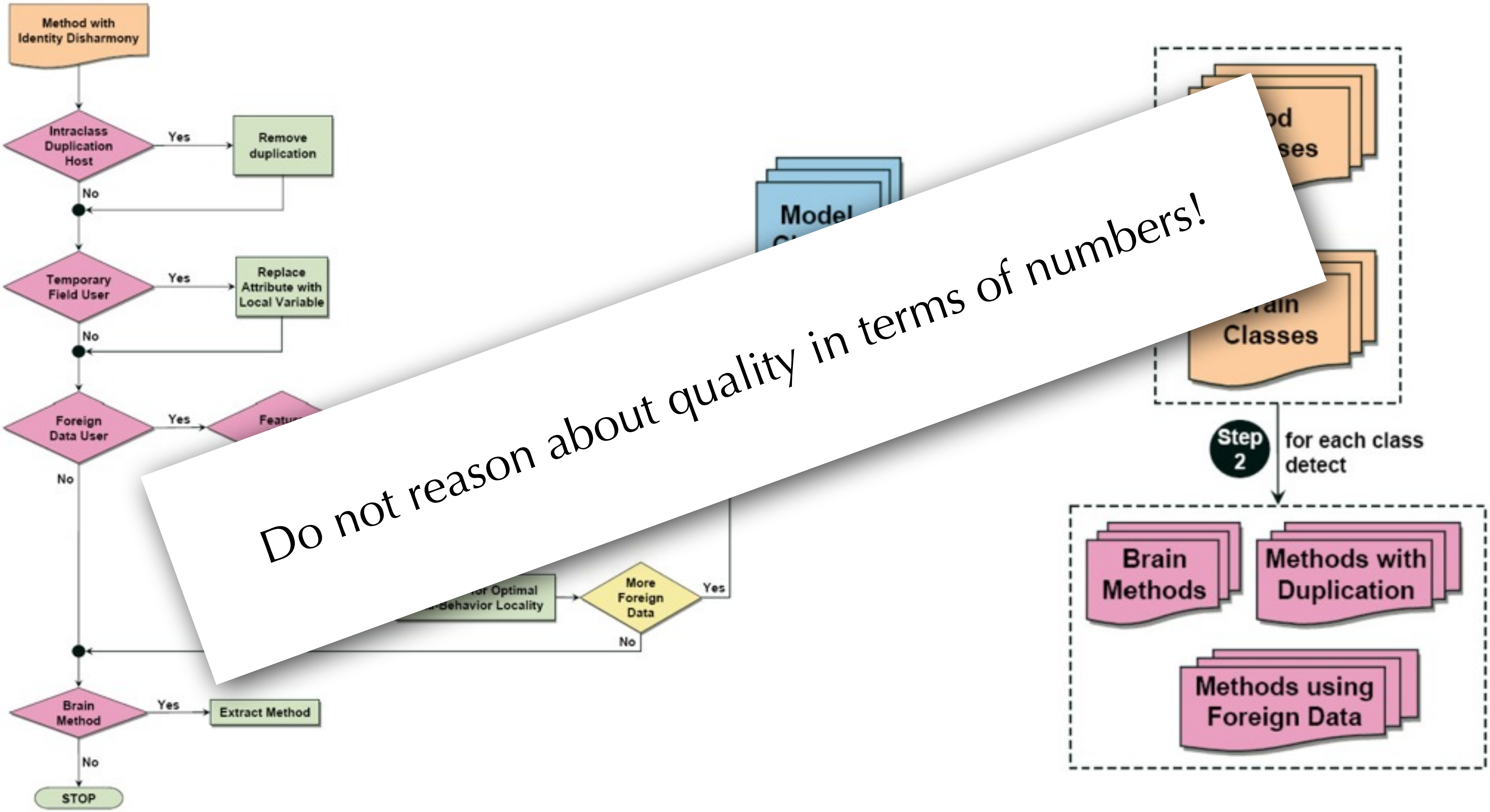


Follow A Clear and Repeatable Process


Method with Identity Disharmony
Intraclass Duplication Host
Yes
Remove duplication
No
Temporary Field User
Yes
Replace Attribute with Local Variable
No
Foreign Data User
Yes
Feature Envy
No
Group used foreign data by their definition classes
Yes
Move behavior to Data Provider
Yes
Data Class
No
Refactor for Optimal Data-Behavior Locality
More Foreign Data
Yes
No
No
Brain Method
Yes
Extract Method
No
STOP



Follow A Clear and Repeatable Process



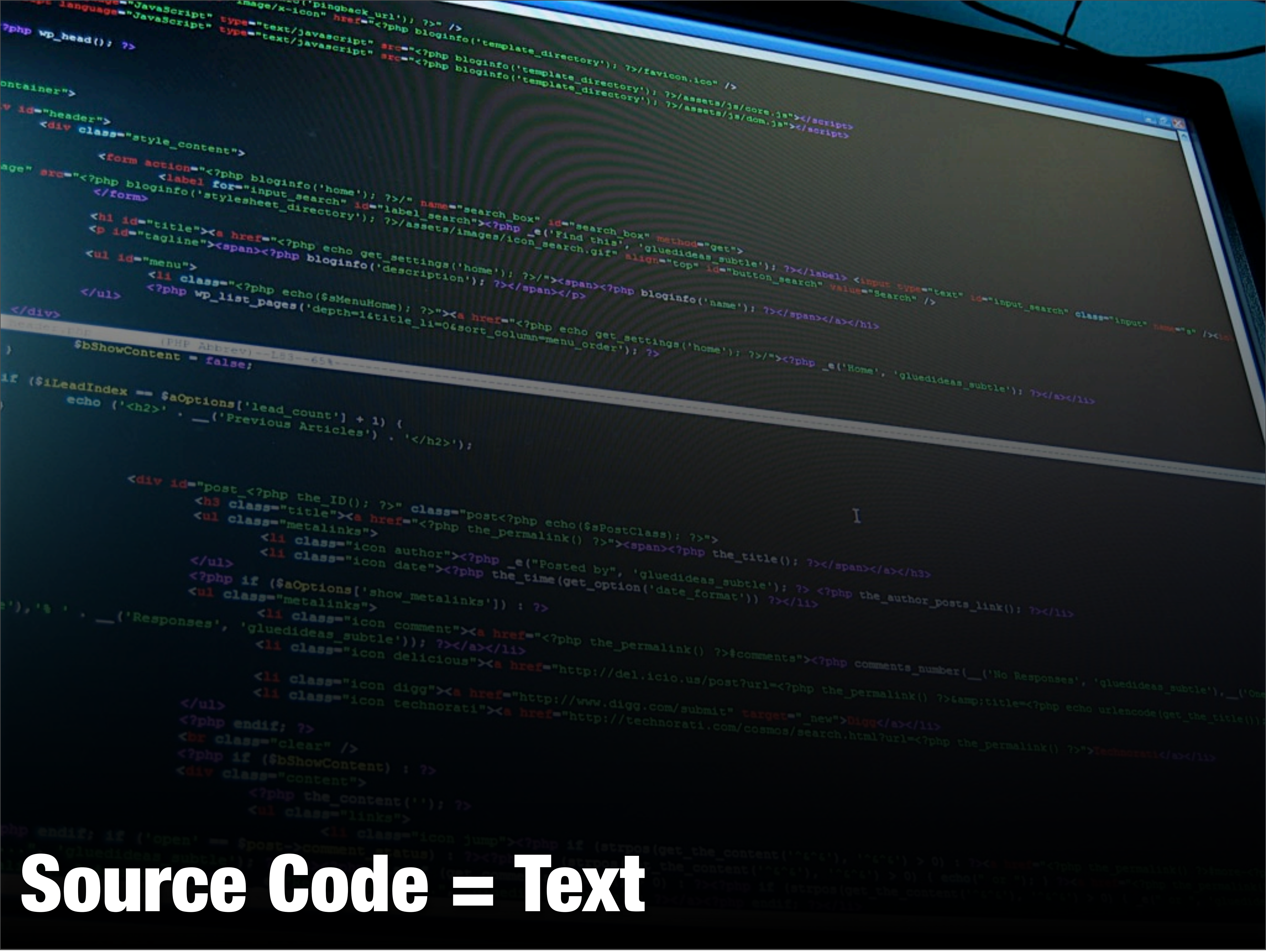
Do not reason about quality in terms of numbers!

Metrics are only half the truth

Can we understand the beauty of a painting by measuring its frame and counting its colors?

Lecture 05

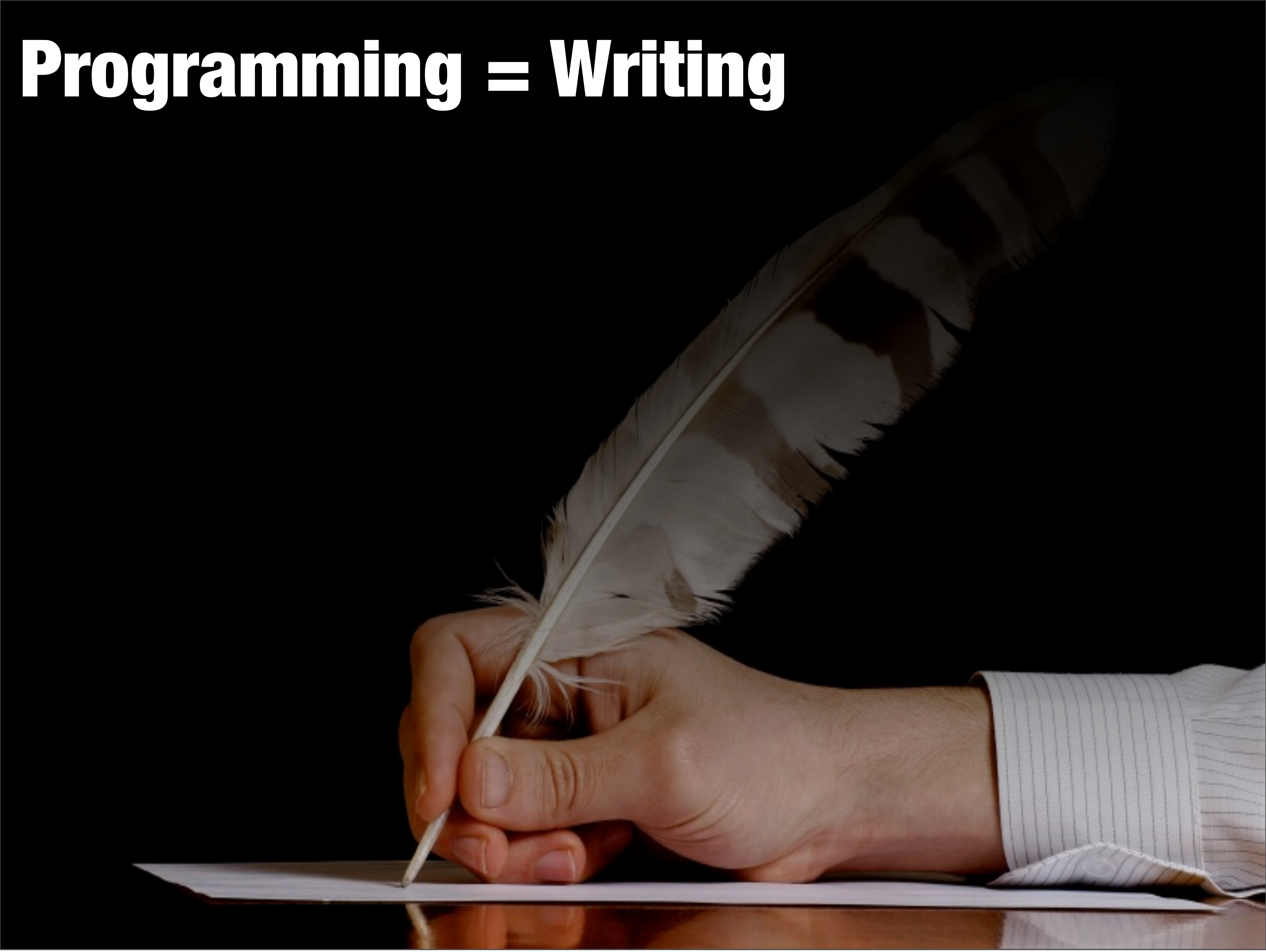
Software Visualization



```
                              ('pingback_url'); ?>" />
                   image/x-icon" href="<?php bloginfo('template_directory'); ?>/favicon.ico" />
ge="JavaScript" type="text/javascript" src="<?php bloginfo('template_directory'); ?>/assets/js/core.js"></script>
pt language="JavaScript" type="text/javascript" src="<?php bloginfo('template_directory'); ?>/assets/js/dom.js"></script>
?php wp_head(); ?>

ontainer">
v id="header">
    <div class="style_content">

        <form action="<?php bloginfo('home'); ?>/" name="search_box" id="search_box" method="get">
            <label for="input_search" id="label_search"><?php _e('Find this', 'gluedideas_subtle'); ?></label> <input type="text" id="input_search" class="input" name="s" /><i
age" src="<?php bloginfo('stylesheet_directory'); ?>/assets/images/icon_search.gif" align="top" id="button_search" value="Search" />
        </form>

        <h1 id="title"><a href="<?php echo get_settings('home'); ?>/"><span><?php bloginfo('name'); ?></span></a></h1>
        <p id="tagline"><span><?php bloginfo('description'); ?></span></p>

        <ul id="menu">
            <li class="<?php echo($sMenuHome); ?>"><a href="<?php echo get_settings('home'); ?>/"><?php _e('Home', 'gluedideas_subtle'); ?></a></li>
            <?php wp_list_pages('depth=1&title_li=0&sort_column=menu_order'); ?>
        </ul>
  </div>
header.php          (PHP Abbrev)--L83--65%------------------------------------
}
      $bShowContent = false;

if ($iLeadIndex == $aOptions['lead_count'] + 1) {
      echo ('<h2>' . __('Previous Articles') . '</h2>');
}


            <div id="post_<?php the_ID(); ?>" class="post<?php echo($sPostClass); ?>">
                  <h3 class="title"><a href="<?php the_permalink() ?>"><span><?php the_title(); ?></span></a></h3>
                  <ul class="metalinks">
                        <li class="icon author"><?php _e("Posted by", 'gluedideas_subtle'); ?> <?php the_author_posts_link(); ?></li>
                        <li class="icon date"><?php the_time(get_option('date_format')) ?></li>
                  </ul>
                  <?php if ($aOptions['show_metalinks']) : ?>
                  <ul class="metalinks">
                        <li class="icon comment"><a href="<?php the_permalink() ?>#comments"><?php comments_number(__('No Responses', 'gluedideas_subtle'),__('One
e'),'% ' . __('Responses', 'gluedideas_subtle')); ?></a></li>
                        <li class="icon delicious"><a href="http://del.icio.us/post?url=<?php the_permalink() ?>&amp;title=<?php echo urlencode(get_the_title());

                        <li class="icon digg"><a href="http://www.digg.com/submit" target="_new">Digg</a></li>
                        <li class="icon technorati"><a href="http://technorati.com/cosmos/search.html?url=<?php the_permalink() ?>">Technorati</a></li>
                  </ul>
                  <?php endif; ?>
                  <br class="clear" />
                  <?php if ($bShowContent) : ?>
                  <div class="content">
                        <?php the_content(''); ?>
                        <ul class="links">
                              <li class="icon jump"><?php if (strpos(get_the_
php endif; if ('open' == $post->comment_status) : ?><?php
```

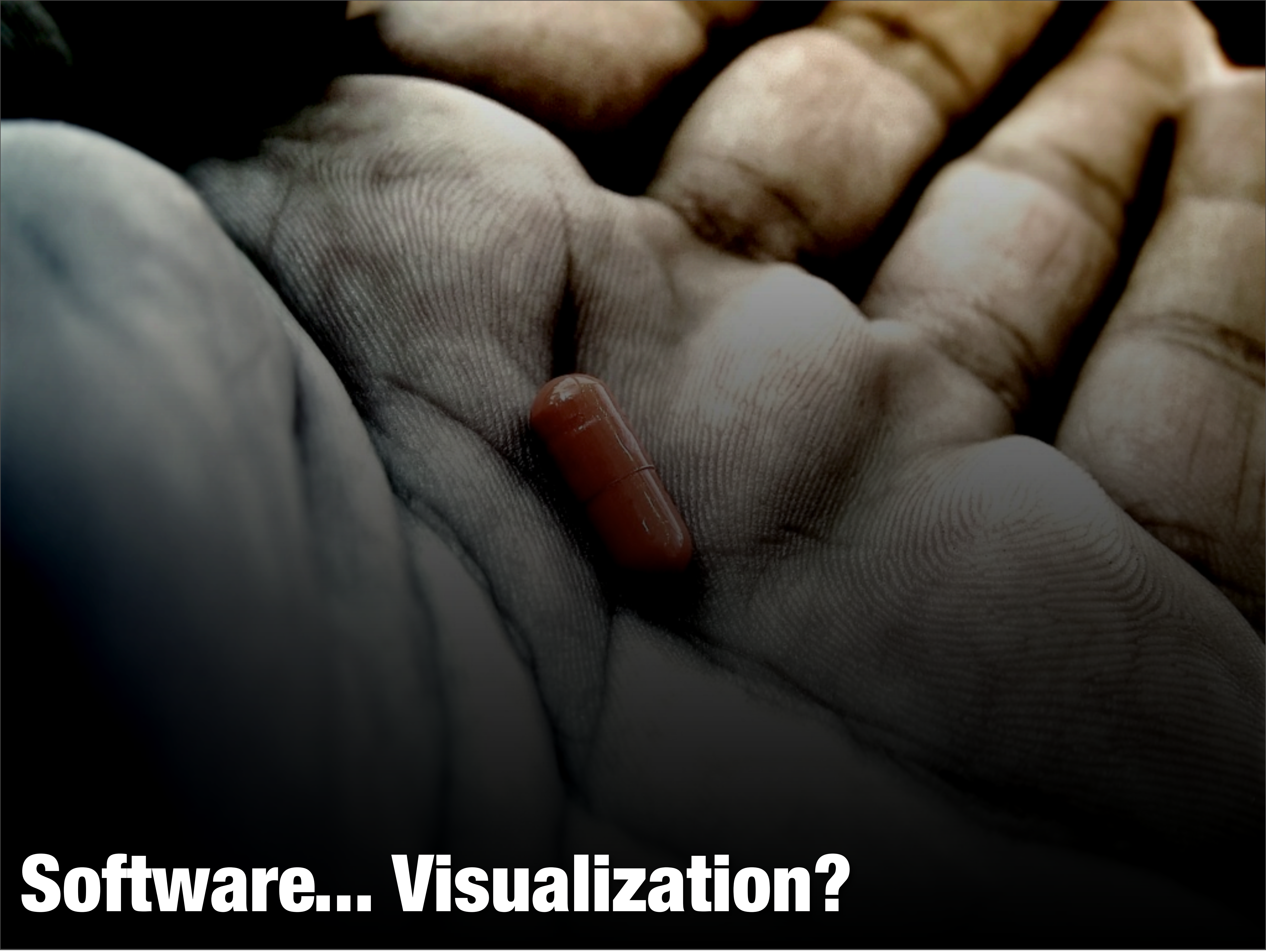
Source Code = Text

Programming = Writing

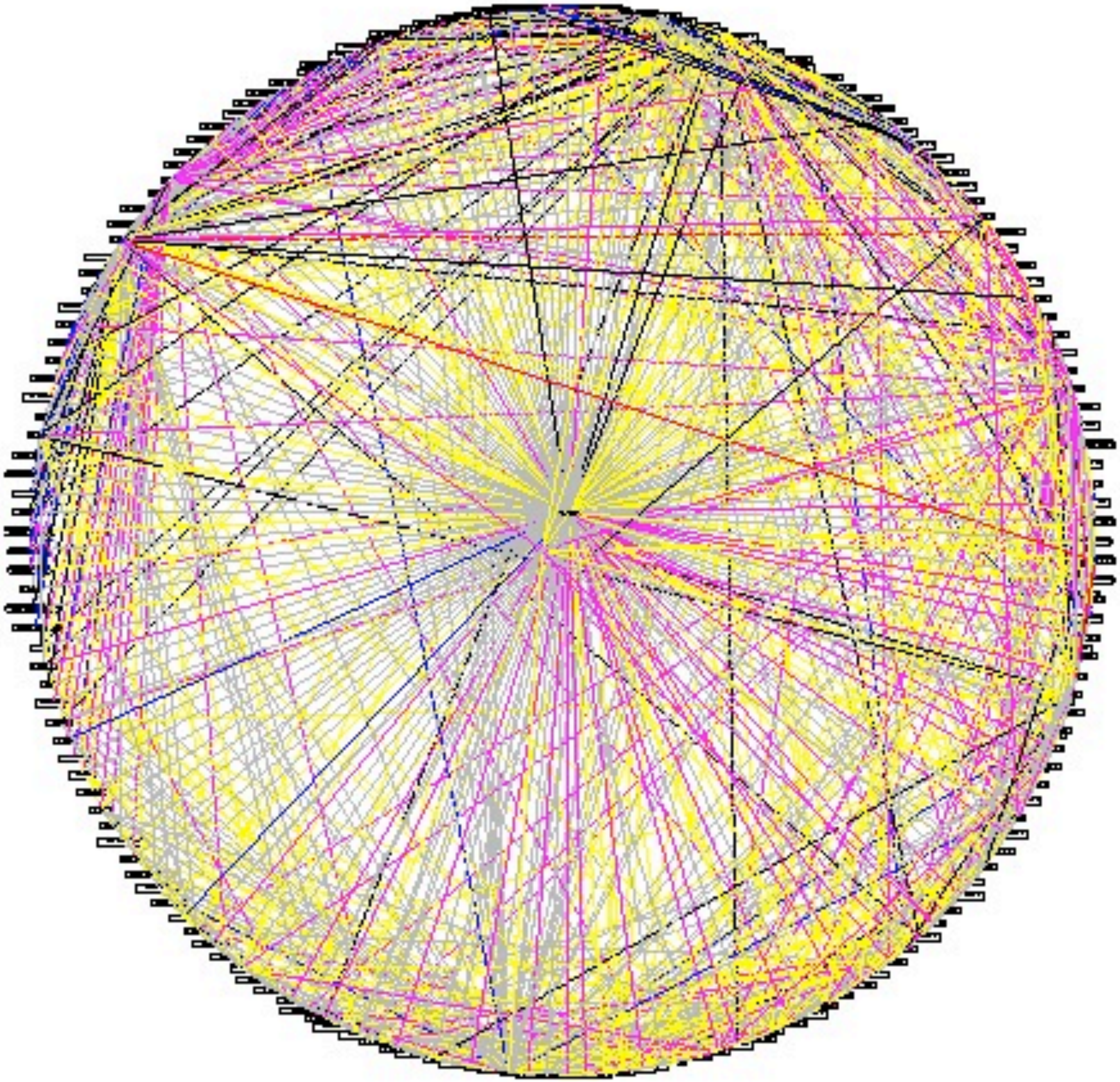


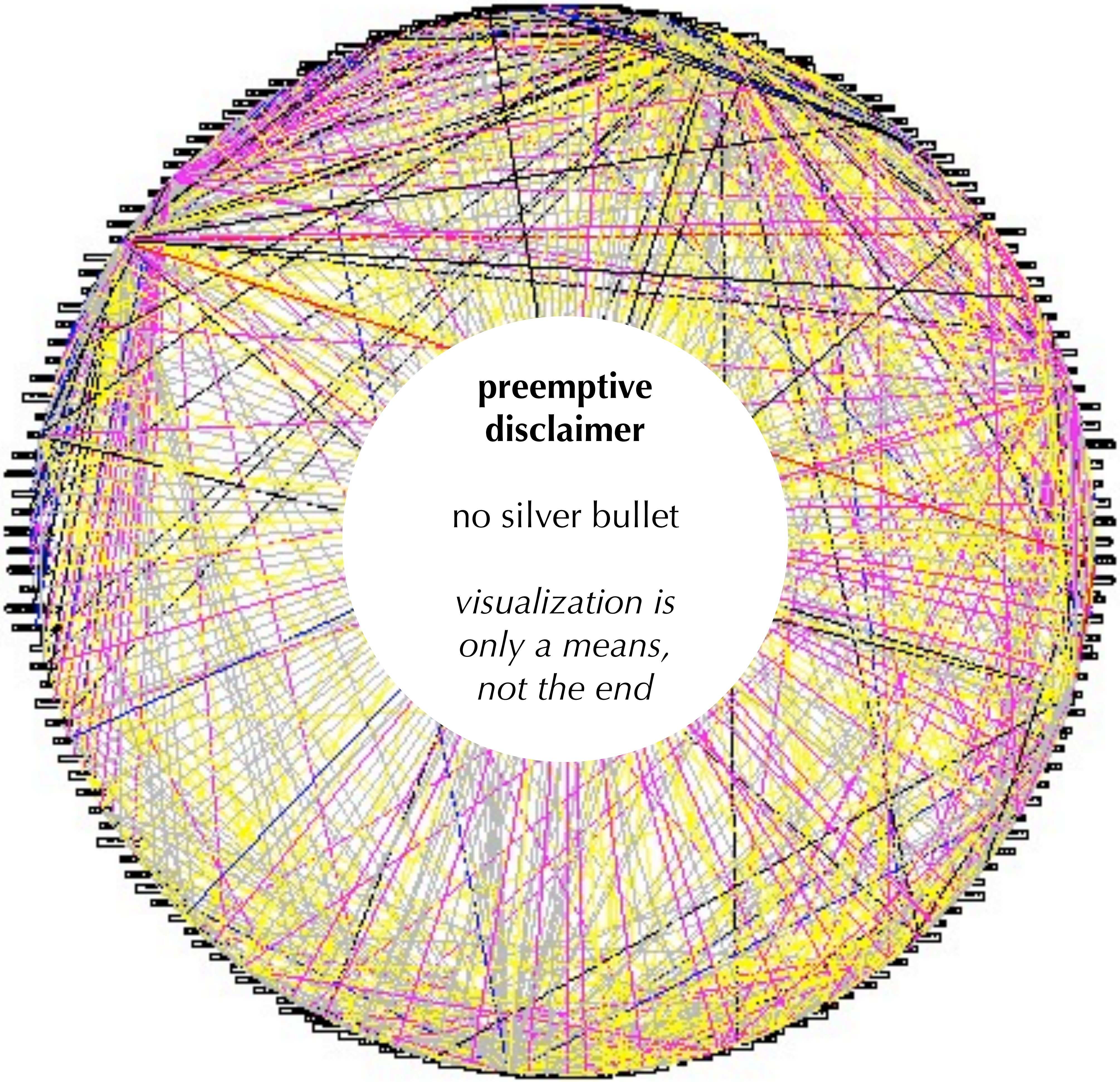
```c
/***************************************************************************/
/*                               micro-Max,                                */
/* A chess program smaller than 2KB (of non-blank source), by H.G. Muller  */
/***************************************************************************/
/* version 3.2 (2000 characters) features:                                 */
/* - recursive negamax search                                              */
/* - quiescence search with recaptures                                     */
/* - recapture extensions                                                  */
/* - (internal) iterative deepening                                        */
/* - best-move-first 'sorting'                                             */
/* - a hash table storing score and best move                              */
/* - full FIDE rules (expt minor ptomotion) and move-legality checking     */

#define F(I,S,N) for(I=S;I<N;I++)
#define W(A) while(A)
#define K(A,B) *(int*)(T+A+(B&8)+S*(B&7))
#define J(A) K(y+A,b[y])-K(x+A,u)-K(H+A,t)

#define U 16777224
struct _ {int K,V;char X,Y,D;} A[U];            /* hash table, 16M+8 entries*/

int V=112,M=136,S=128,I=8e4,C=799,Q,N,i;        /* V=0x70=rank mask, M=0x88 */

char O,K,L,
w[]={0,1,1,3,-1,3,5,9},                         /* relative piece values    */
o[]={-16,-15,-17,0,1,16,0,1,16,15,17,0,14,18,31,33,0, /* step-vector lists */
     7,-1,11,6,8,3,6,                           /* 1st dir. in o[] per piece*/
     6,3,5,7,4,5,3,6},                          /* initial piece setup      */
b[129],                                         /* board: half of 16x8+dummy*/
T[1035],                                        /* hash translation table   */

n[]=".?+nkbrq?*?NKBRQ";                         /* piece symbols on printout*/

D(k,q,l,e,J,Z,E,z,n)    /* recursive minimax search, k=moving side, n=depth*/
int k,q,l,e,J,Z,E,z,n;  /* (q,l)=window, e=current eval. score, E=e.p. sqr.*/
{                       /* e=score, z=prev.dest; J,Z=hashkeys; return score*/
 int j,r,m,v,d,h,i=9,F,G;
 char t,p,u,x,y,X,Y,H,B;
 struct _*a=A;
                                                /* lookup pos. in hash table*/
 j=(k*E^J)&U-9;                                 /* try 8 consec. locations  */
 W((h=A[++j].K)&&h-Z&&--i);                     /* first empty or match     */
 a+=i?j:0;                                      /* dummy A[0] if miss & full*/
 if(a->K)                                       /* hit: pos. is in hash tab */
 {d=a->D;v=a->V;X=a->X;                         /* examine stored data      */
  if(d>=n)                                      /* if depth sufficient:     */
  {if(v>=l|X&S&&v<=q|X&8)return v;              /* use if window compatible */
   d=n-1;                                       /* or use as iter. start    */
  }X&=~M;Y=a->Y;                                /*      with best-move hint */
  Y=d?Y:0;                                      /* don't try best at d=0    */
 }else d=X=Y=0;                                 /* start iter., no best yet */
 N++;                                           /* node count (for timing)  */
 W(d++<n|z==8&N<1e7&d<98)                       /* iterative deepening loop */
 {x=B=X;                                        /* start scan at prev. best */
  Y|=8&Y>>4;                                    /* request try noncastl. 1st*/
  m=d>1?-I:e;                                   /* unconsidered:static eval */
  do{u=b[x];                                    /* scan board looking for   */
   if(u&k)                                      /*  own piece (inefficient!)*/
   {r=p=u&7;                                    /* p = piece type (set r>0) */
    j=o[p+16];                                  /* first step vector f.piece*/
    W(r=p>2&r<0?-r:-o[++j])                     /* loop over directions o[] */
    {A:                                         /* resume normal after best */
     y=x;F=G=S;                                 /* (x,y)=move, (F,G)=castl.R*/
     do{H=y+=r;                                 /* y traverses ray          */
      if(Y&8)H=y=Y&~M;                          /* sneak in prev. best move */
      if(y&M)break;                             /* board edge hit           */
      if(p<3&y==E)H=y^16;                       /* shift capt.sqr. H if e.p.*/
      t=b[H];if(t&k|p<3&!(r&7)!=!t)break;       /* capt. own, bad pawn mode */
      i=99*w[t&7];                              /* value of capt. piece t   */
```

```c
      if(i<0||E-S&&b[E]&&y-E<2&E-y<2)m=I;       /* K capt. or bad castling  */
      if(m>=l)goto C;                           /* abort on fail high       */





      if(h=d-(y!=z))                            /* remaining depth(-recapt.)*/
      {v=p<6?b[x+8]-b[y+8]:0;                   /* center positional pts.   */
       b[G]=b[H]=b[x]=0;b[y]=u&31;              /* do move, strip virgin-bit*/
       if(!(G&M)){b[F]=k+6;v+=30;}              /* castling: put R & score  */
       if(p<3)                                  /* pawns:                   */
       {v-=9*(((x-2)&M||b[x-2]!=u)+             /* structure, undefended    */
              ((x+2)&M||b[x+2]!=u)-1);          /*        squares plus bias */
        if(y+r+1&S){b[y]|=7;i+=C;}              /* promote p to Q, add score*/
       }
       v=-D(24-k,-l-(l>e),m>q?-m:-q,-e-v-i,     /* recursive eval. of reply */
            J+J(0),Z+J(8)+G-S,F,y,h);           /* J,Z: hash keys           */
       v-=v>e;                                  /* delayed-gain penalty     */
       if(z==9)                                 /* called as move-legality  */
       {if(v!=-I&x==K&y==L)                     /*   checker: if move found */
        {Q=-e-i;O=F;return l;}                  /*   & not in check, signal */
        v=m;                                    /* (prevent fail-lows on    */
       }                                        /*   K-capt. replies)       */
       b[G]=k+38;b[F]=b[y]=0;b[x]=u;b[H]=t;     /* undo move,G can be dummy */
       if(Y&8){m=v;Y&=~8;goto A;}               /* best=1st done,redo normal*/
       if(v>m){m=v;X=x;Y=y|S&G;}                /* update max, mark with S  */
      }                                         /*          if non castling */
      t+=p<5;                                   /* fake capt. for nonsliding*/
      if(p<3&6*k+(y&V)==S                       /* pawn on 3rd/6th, or      */
          ||(u&~24)==36&j==7&&                  /* virgin K moving sideways,*/
          G&M&&b[G=(x|7)-(r>>1&7)]&32           /* 1st, virgin R in corner G*/
          &&!(b[G^1]|b[G^2])                    /* 2 empty sqrs. next to R  */
      ){F=y;t--;}                               /* unfake capt., enable e.p.*/
     }W(!t);                                    /* if not capt. continue ray*/
  }}}W((x=x+9&~M)-B);                           /* next sqr. of board, wrap */
C:if(m>I/4|m<-I/4)d=99;                         /* mate is indep. of depth  */
  m=m+I?m:-D(24-k,-I,I,0,J,Z,S,S,1)/2;          /* best loses K: (stale)mate*/
  if(!a->K|(a->X&M)!=M|a->D<=d)                 /* if new/better type/depth:*/
  {a->K=Z;a->V=m;a->D=d;A->K=0;                 /* store in hash,dummy stays*/
   a->X=X|8*(m>q)|S*(m<l);a->Y=Y;               /* empty, type (limit/exact)*/
  }                                             /*    encoded in X S,8 bits */
/*if(z==8)printf("%2d ply, %9d searched, %6d by (%2x,%2x)
\n",d-1,N,m,X,Y&0x77);*/
 }
 if(z&8){K=X;L=Y&~M;}
 return m;
}

main()
{
 int j,k=8,*p,c[9];

 F(i,0,8)
 {b[i]=(b[i+V]=o[i+24]+40)+8;b[i+16]=18;b[i+96]=9;    /* initial board setup*/
  F(j,0,8)b[16*j+i+8]=(i-4)*(i-4)+(j-3.5)*(j-3.5);    /* center-pts table   */
 }                                                    /*(in unused half b[])*/
 F(i,M,1035)T[i]=random()>>9;

 W(1)                                                 /* play loop          */
 {F(i,0,121)printf(" %c",i&8&&(i+=7)?10:n[b[i]&15]); /* print board        */
  p=c;W((*p++=getchar())>10);                         /* read input line    */
  N=0;
  if(*c-10){K=c[0]-16*c[1]+C;L=c[2]-16*c[3]+C;}else  /* parse entered move */
   D(k,-I,I,Q,1,1,O,8,0);                             /* or think up one    */
  F(i,0,U)A[i].K=0;                                   /* clear hash table   */
  if(D(k,-I,I,Q,1,1,O,9,2)==I)k^=24;                  /* check legality & do*/
 }
}
```



Software... Visualization?

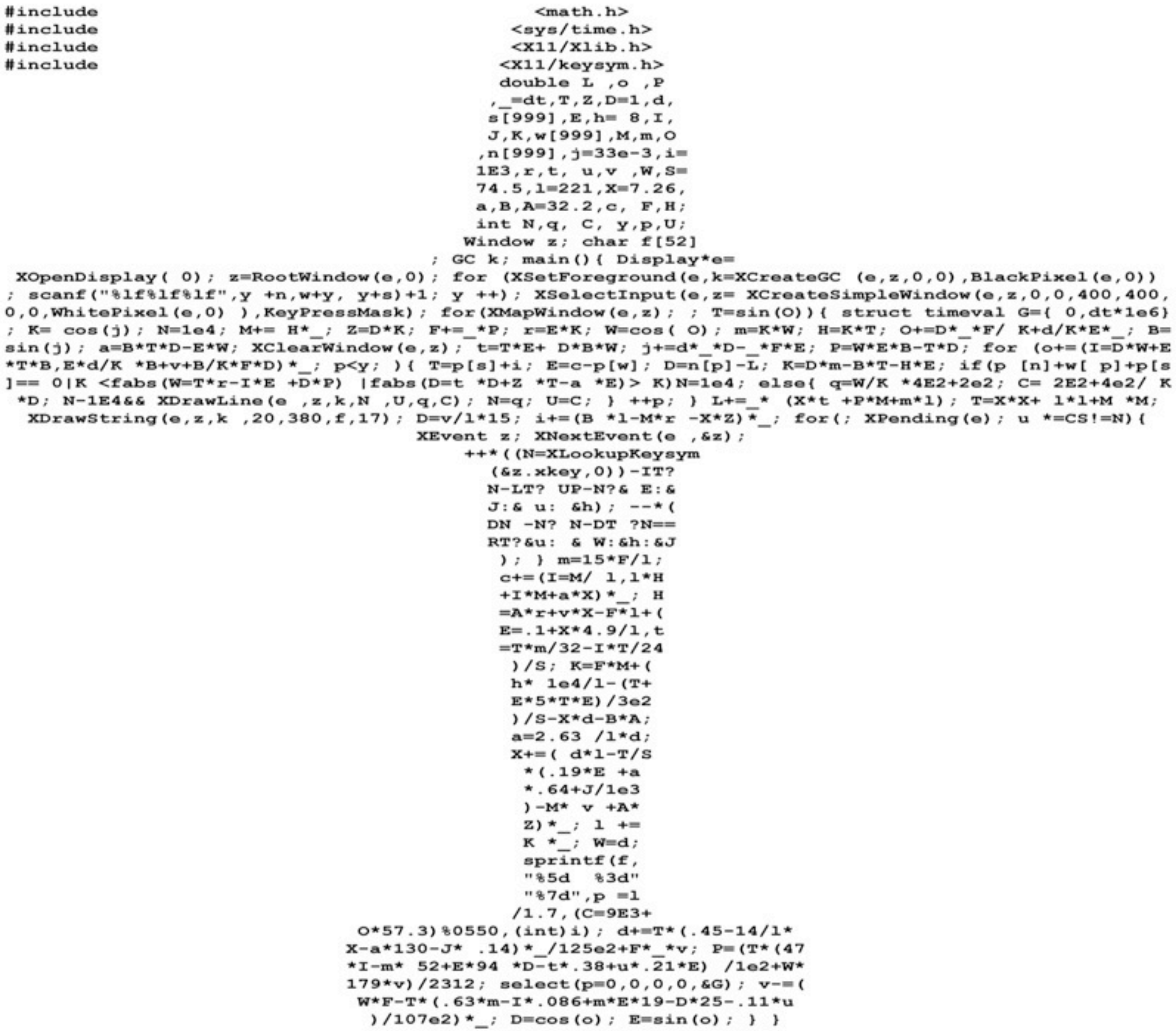




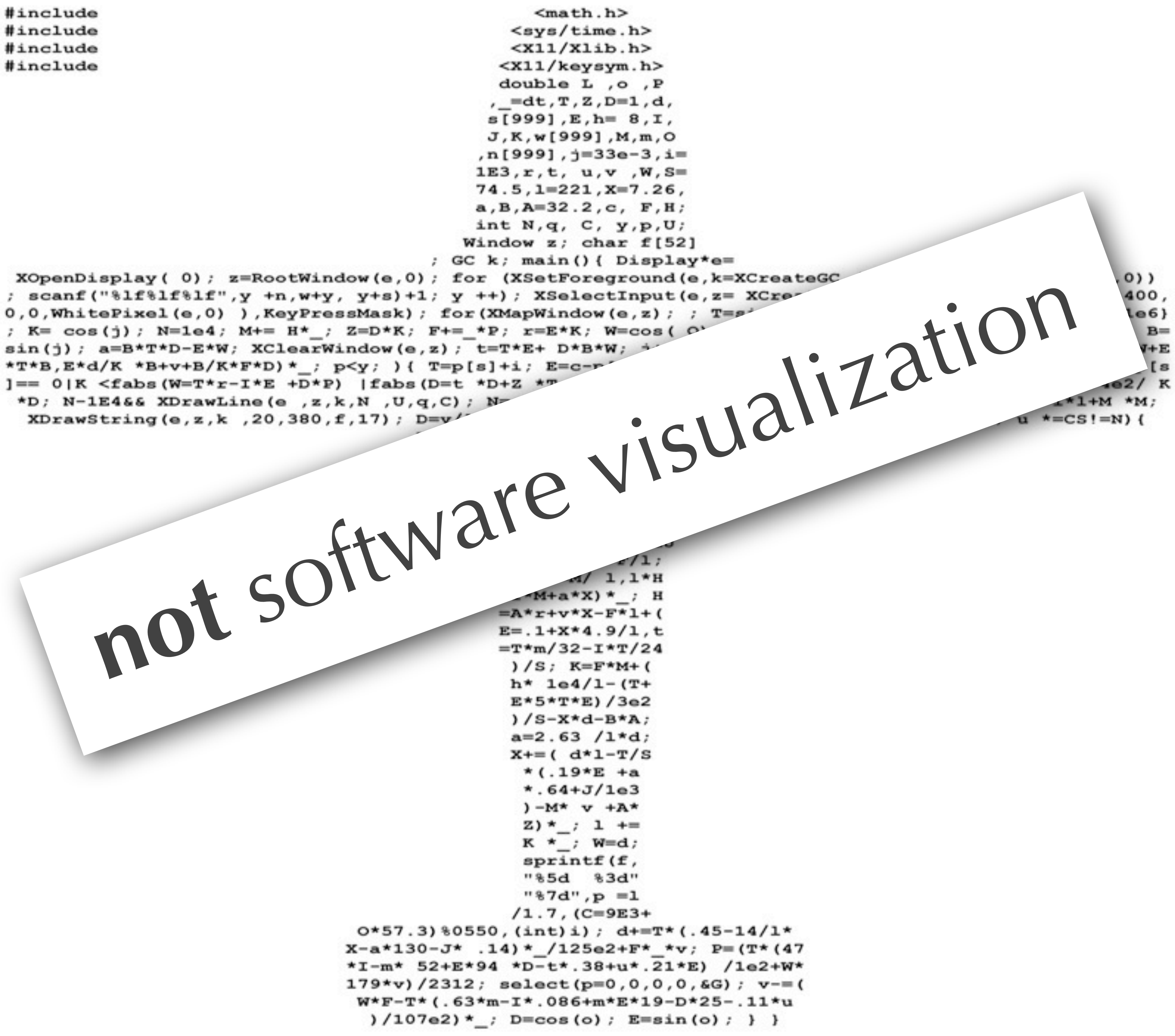
preemptive disclaimer

no silver bullet

visualization is only a means, not the end



```
#include                                     <math.h>
#include                                   <sys/time.h>
#include                                   <X11/Xlib.h>
#include                                  <X11/keysym.h>
                                          double L ,o ,P
                                         ,_=dt,T,Z,D=1,d,
                                         s[999],E,h= 8,I,
                                         J,K,w[999],M,m,O
                                        ,n[999],j=33e-3,i=
                                        1E3,r,t, u,v ,W,S=
                                        74.5,l=221,X=7.26,
                                        a,B,A=32.2,c, F,H;
                                        int N,q, C, y,p,U;
                                       Window z; char f[52]
                                    ; GC k; main(){ Display*e=
 XOpenDisplay( 0); z=RootWindow(e,0); for (XSetForeground(e,k=XCreateGC (e,z,0,0),BlackPixel(e,0))
; scanf("%lf%lf%lf",y +n,w+y, y+s)+1; y ++); XSelectInput(e,z= XCreateSimpleWindow(e,z,0,0,400,400,
0,0,WhitePixel(e,0) ),KeyPressMask); for(XMapWindow(e,z); ; T=sin(O)){ struct timeval G={ 0,dt*1e6}
; K= cos(j); N=1e4; M+= H*_; Z=D*K; F+=_*P; r=E*K; W=cos( O); m=K*W; H=K*T; O+=D*_*F/ K+d/K*E*_; B=
sin(j); a=B*T*D-E*W; XClearWindow(e,z); t=T*E+ D*B*W; j+=d*_*D-_*F*E; P=W*E*B-T*D; for (o+=(I=D*W+E
*T*B,E*d/K *B+v+B/K*F*D)*_; p<y; ){ T=p[s]+i; E=c-p[w]; D=n[p]-L; K=D*m-B*T-H*E; if(p [n]+w[ p]+p[s
]== 0|K <fabs(W=T*r-I*E +D*P) |fabs(D=t *D+Z *T-a *E)> K)N=1e4; else{ q=W/K *4E2+2e2; C= 2E2+4e2/ K
 *D; N-1E4&& XDrawLine(e ,z,k,N ,U,q,C); N=q; U=C; } ++p; } L+=_* (X*t +P*M+m*l); T=X*X+ l*l+M *M;
  XDrawString(e,z,k ,20,380,f,17); D=v/l*15; i+=(B *l-M*r -X*Z)*_; for(; XPending(e); u *=CS!=N){
                                   XEvent z; XNextEvent(e ,&z);
                                       ++*((N=XLookupKeysym
                                         (&z.xkey,0))-IT?
                                         N-LT? UP-N?& E:&
                                         J:& u: &h); --*(
                                         DN -N? N-DT ?N==
                                         RT?&u: & W:&h:&J
                                          ); } m=15*F/l;
                                          c+=(I=M/ l,l*H
                                          +I*M+a*X)*_; H
                                          =A*r+v*X-F*l+(
                                          E=.1+X*4.9/l,t
                                          =T*m/32-I*T/24
                                           )/S; K=F*M+(
                                           h* 1e4/l-(T+
                                           E*5*T*E)/3e2
                                           )/S-X*d-B*A;
                                           a=2.63 /l*d;
                                           X+=( d*l-T/S
                                            *(.19*E +a
                                            *.64+J/1e3
                                            )-M* v +A*
                                            Z)*_; l +=
                                            K *_; W=d;
                                            sprintf(f,
                                            "%5d  %3d"
                                            "%7d",p =l
                                           /1.7,(C=9E3+
                              O*57.3)%0550,(int)i); d+=T*(.45-14/l*
                             X-a*130-J* .14)*_/125e2+F*_*v; P=(T*(47
                             *I-m* 52+E*94 *D-t*.38+u*.21*E) /1e2+W*
                             179*v)/2312; select(p=0,0,0,0,&G); v-=(
                              W*F-T*(.63*m-I*.086+m*E*19-D*25-.11*u
                               )/107e2)*_; D=cos(o); E=sin(o); } }
```

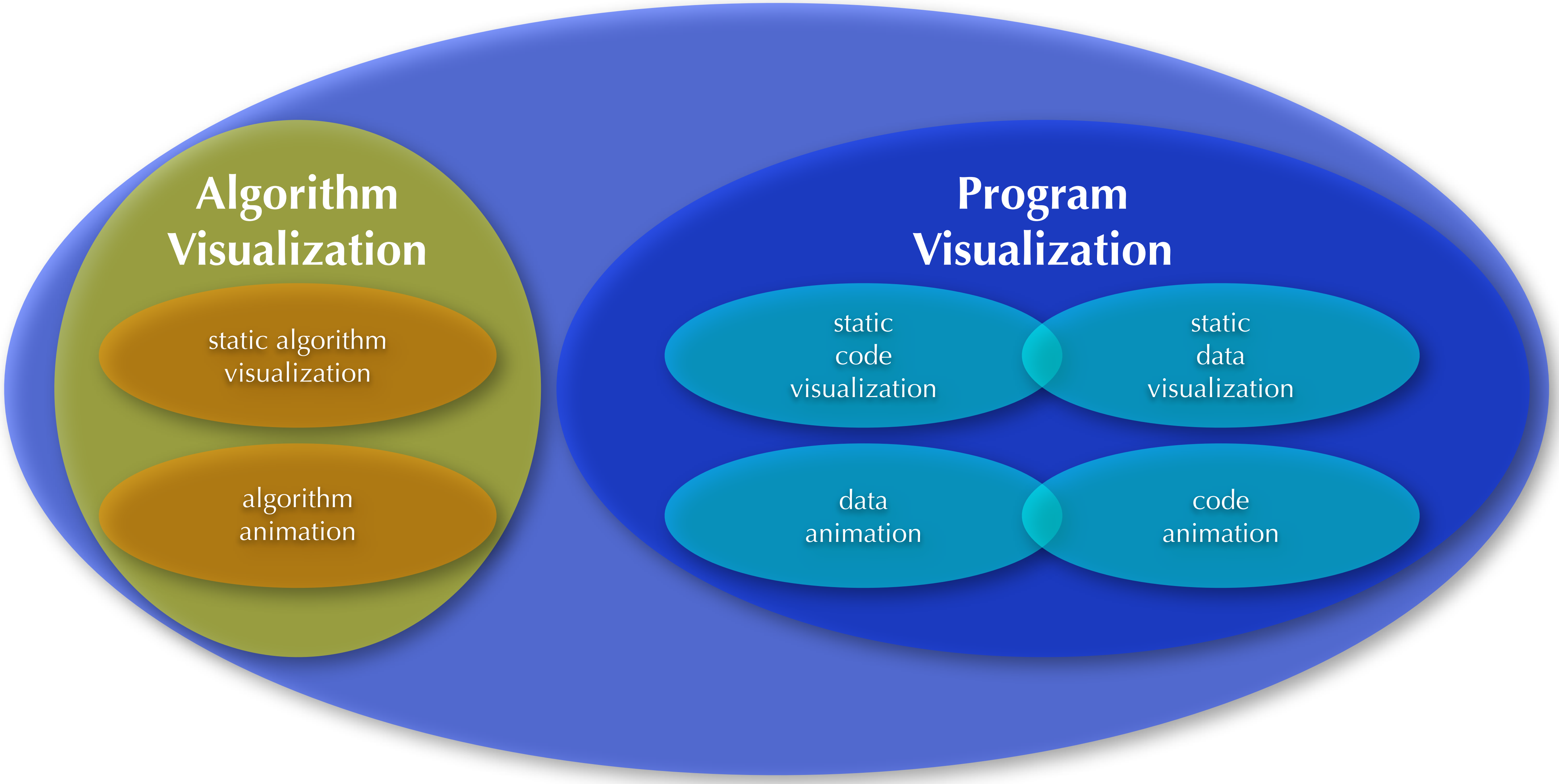


```
#include                                     <math.h>
#include                                   <sys/time.h>
#include                                   <X11/Xlib.h>
#include                                  <X11/keysym.h>
                                          double L ,o ,P
                                         ,_=dt,T,Z,D=1,d,
                                         s[999],E,h= 8,I,
                                         J,K,w[999],M,m,O
                                        ,n[999],j=33e-3,i=
                                        1E3,r,t, u,v ,W,S=
                                        74.5,l=221,X=7.26,
                                        a,B,A=32.2,c, F,H;
                                        int N,q, C, y,p,U;
                                       Window z; char f[52]
                                    ; GC k; main(){ Display*e=
 XOpenDisplay( 0); z=RootWindow(e,0); for (XSetForeground(e,k=XCreateGC                    ,0))
; scanf("%lf%lf%lf",y +n,w+y, y+s)+1; y ++); XSelectInput(e,z= XCre                       400,
0,0,WhitePixel(e,0) ),KeyPressMask); for(XMapWindow(e,z); ; T=s                           le6)
; K= cos(j); N=1e4; M+= H*_; Z=D*K; F+=_*P; r=E*K; W=cos( O                                B=
sin(j); a=B*T*D-E*W; XClearWindow(e,z); t=T*E+ D*B*W; i                                  +E
*T*B,E*d/K *B+v+B/K*F*D)*_; p<y; ){ T=p[s]+i; E=c-n                                      [s
]== 0|K <fabs(W=T*r-I*E +D*P) |fabs(D=t *D+Z *T                                        e2/ K
 *D; N-1E4&& XDrawLine(e ,z,k,N ,U,q,C); N=                                        *l+M *M;
  XDrawString(e,z,k ,20,380,f,17); D=v/                                        u *=CS!=N){

                                            F/l;
                                         M/ l,l*H
                                         M+a*X)*_; H
                                        =A*r+v*X-F*l+(
                                        E=.1+X*4.9/l,t
                                        =T*m/32-I*T/24
                                         )/S; K=F*M+(
                                         h* 1e4/l-(T+
                                         E*5*T*E)/3e2
                                         )/S-X*d-B*A;
                                         a=2.63 /l*d;
                                         X+=( d*l-T/S
                                          *(.19*E +a
                                          *.64+J/1e3
                                          )-M* v +A*
                                          Z)*_; l +=
                                          K *_; W=d;
                                          sprintf(f,
                                          "%5d  %3d"
                                          "%7d",p =l
                                         /1.7,(C=9E3+
                               O*57.3)%0550,(int)i); d+=T*(.45-14/l*
                              X-a*130-J* .14)*_/125e2+F*_*v; P=(T*(47
                              *I-m* 52+E*94 *D-t*.38+u*.21*E) /1e2+W*
                              179*v)/2312; select(p=0,0,0,0,&G); v-=(
                               W*F-T*(.63*m-I*.086+m*E*19-D*25-.11*u
                                )/107e2)*_; D=cos(o); E=sin(o); } }
```

not software visualization

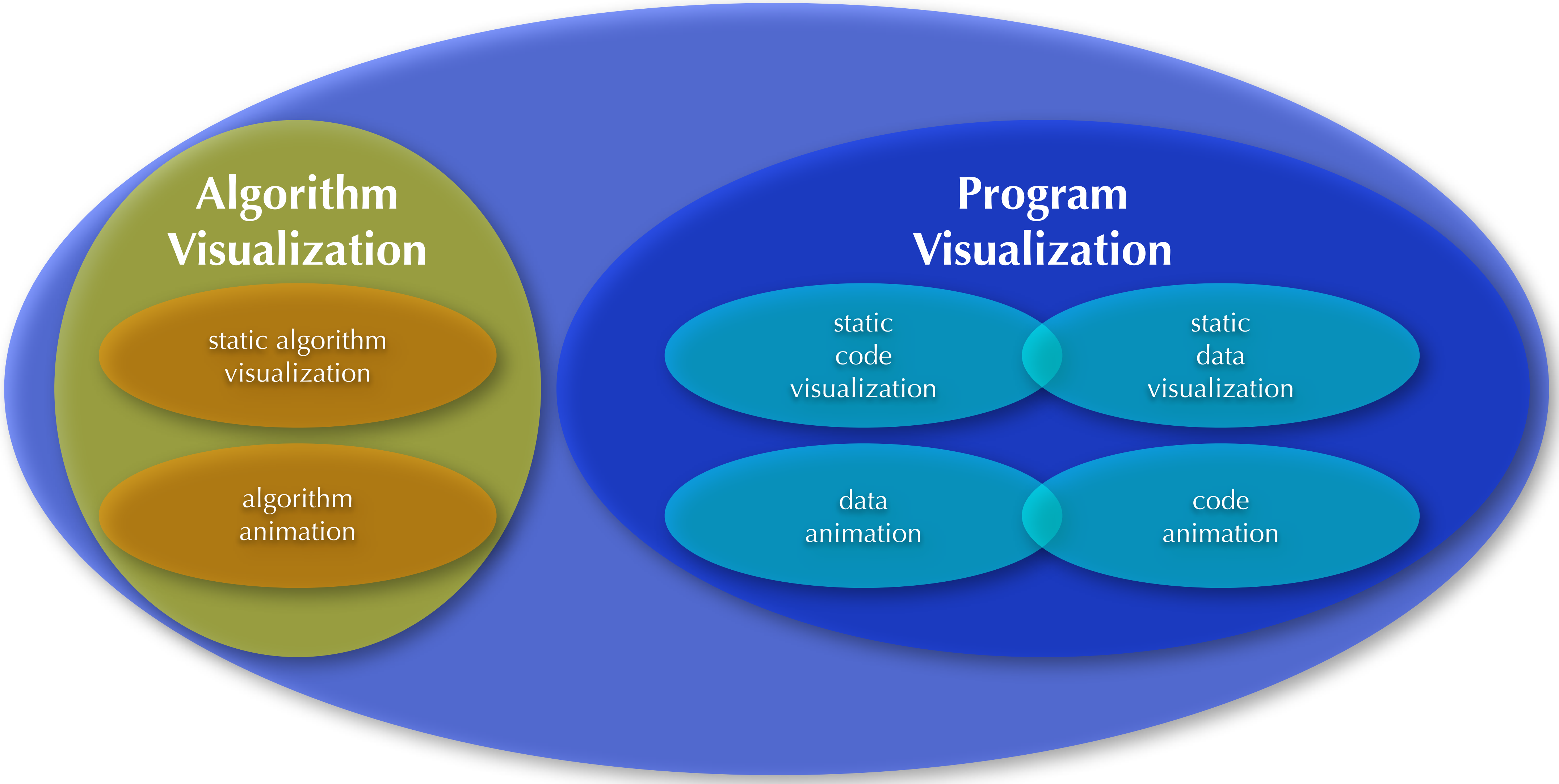
Software Visualization

- Program Visualization: “The visualization of the actual program code or data structures in static or dynamic form”
- Algorithm Visualization: “The visualization of the higher-level abstractions which describe software”



Software Visualization

- Program Visualization: "The visualization of the actual program code or data structures in static or dynamic form"
- Algorithm Visualization: "The visualization of the higher-level abstractions which describe software"



Software Visualization in Context

Software Visualization in Context

- There are many good-looking visualizations, but...

Software Visualization in Context

- There are many good-looking visualizations, but...
- When it comes to maintenance & evolution, there are several issues:

Software Visualization in Context

- There are many good-looking visualizations, but...
- When it comes to maintenance & evolution, there are several issues:
  - Scalability

Software Visualization in Context

- There are many good-looking visualizations, but...
- When it comes to maintenance & evolution, there are several issues:
  - Scalability
  - Information Retrieval

Software Visualization in Context

- There are many good-looking visualizations, but...
- When it comes to maintenance & evolution, there are several issues:
    - Scalability
    - Information Retrieval
    - What to visualize

Software Visualization in Context

- There are many good-looking visualizations, but...
- When it comes to maintenance & evolution, there are several issues:
  - Scalability
  - Information Retrieval
  - What to visualize
  - How to visualize

Software Visualization in Context

- There are many good-looking visualizations, but...
- When it comes to maintenance & evolution, there are several issues:
  - Scalability
  - Information Retrieval
  - What to visualize
  - How to visualize
  - Limited time

Software Visualization in Context

- There are many good-looking visualizations, but...
- When it comes to maintenance & evolution, there are several issues:
  - Scalability
  - Information Retrieval
  - What to visualize
  - How to visualize
  - Limited time
  - Limited resources

Program Visualization

Program Visualization

- “The visualization of the actual program code or data structures in either static or dynamic form”

Program Visualization

- “The visualization of the actual program code or data structures in either static or dynamic form”
- Overall goal: generate views of a system to understand it

Program Visualization

- "The visualization of the actual program code or data structures in either static or dynamic form"
- Overall goal: generate views of a system to understand it
- Surprisingly complex problem domain/research area

Program Visualization

- “The visualization of the actual program code or data structures in either static or dynamic form”
- Overall goal: generate views of a system to understand it
- Surprisingly complex problem domain/research area
  - Visual Aspects: Efficient use of space, overplotting problems, layout issues, HCI issues, GUI issues, lack of conventions (colors, shapes, etc.)

Program Visualization

- “The visualization of the actual program code or data structures in either static or dynamic form”
- Overall goal: generate views of a system to understand it
- Surprisingly complex problem domain/research area
  - Visual Aspects: Efficient use of space, overplotting problems, layout issues, HCI issues, GUI issues, lack of conventions (colors, shapes, etc.)
  - Software Aspects

Program Visualization

- “The visualization of the actual program code or data structures in either static or dynamic form”
- Overall goal: generate views of a system to understand it
- Surprisingly complex problem domain/research area
    - Visual Aspects: Efficient use of space, overplotting problems, layout issues, HCI issues, GUI issues, lack of conventions (colors, shapes, etc.)
    - Software Aspects
        - Granularity (complete systems, subsystems, modules, classes, etc.)

Program Visualization

- “The visualization of the actual program code or data structures in either static or dynamic form”
- Overall goal: generate views of a system to understand it
- Surprisingly complex problem domain/research area
  - Visual Aspects: Efficient use of space, overplotting problems, layout issues, HCI issues, GUI issues, lack of conventions (colors, shapes, etc.)
  - Software Aspects
    - Granularity (complete systems, subsystems, modules, classes, etc.)
    - When to apply (first contact, known/unknown parts, forward engineering?)

Program Visualization

- “The visualization of the actual program code or data structures in either static or dynamic form”
- Overall goal: generate views of a system to understand it
- Surprisingly complex problem domain/research area
    - Visual Aspects: Efficient use of space, overplotting problems, layout issues, HCI issues, GUI issues, lack of conventions (colors, shapes, etc.)
    - Software Aspects
        - Granularity (complete systems, subsystems, modules, classes, etc.)
        - When to apply (first contact, known/unknown parts, forward engineering?)
        - Methodology

Static Code Visualization

Static Code Visualization

- The visualization of information that can be extracted from a system at “compile-time”

Static Code Visualization

- The visualization of information that can be extracted from a system at "compile-time"
- Directly influenced by programming languages and their paradigms

Static Code Visualization

- The visualization of information that can be extracted from a system at “compile-time”
- Directly influenced by programming languages and their paradigms
    - Object-Oriented: classes, methods, attributes, inheritance, ...

Static Code Visualization

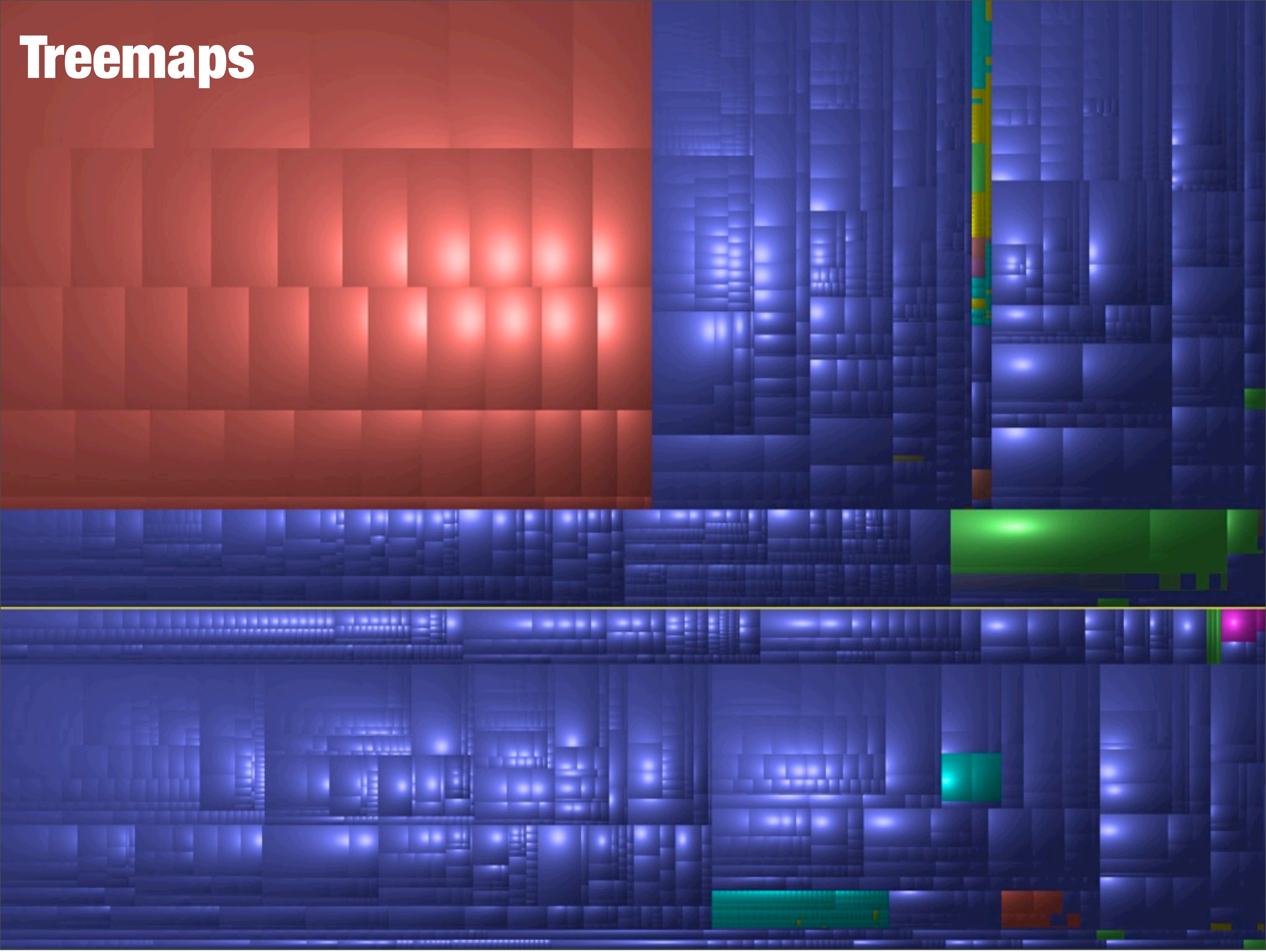
- The visualization of information that can be extracted from a system at “compile-time”
- Directly influenced by programming languages and their paradigms
    - Object-Oriented: classes, methods, attributes, inheritance, ...
    - Procedural: procedures, invocations, imports, ...

Static Code Visualization

- The visualization of information that can be extracted from a system at “compile-time”
- Directly influenced by programming languages and their paradigms
  - Object-Oriented: classes, methods, attributes, inheritance, ...
  - Procedural: procedures, invocations, imports, ...
  - Functional: functions, function calls, ...

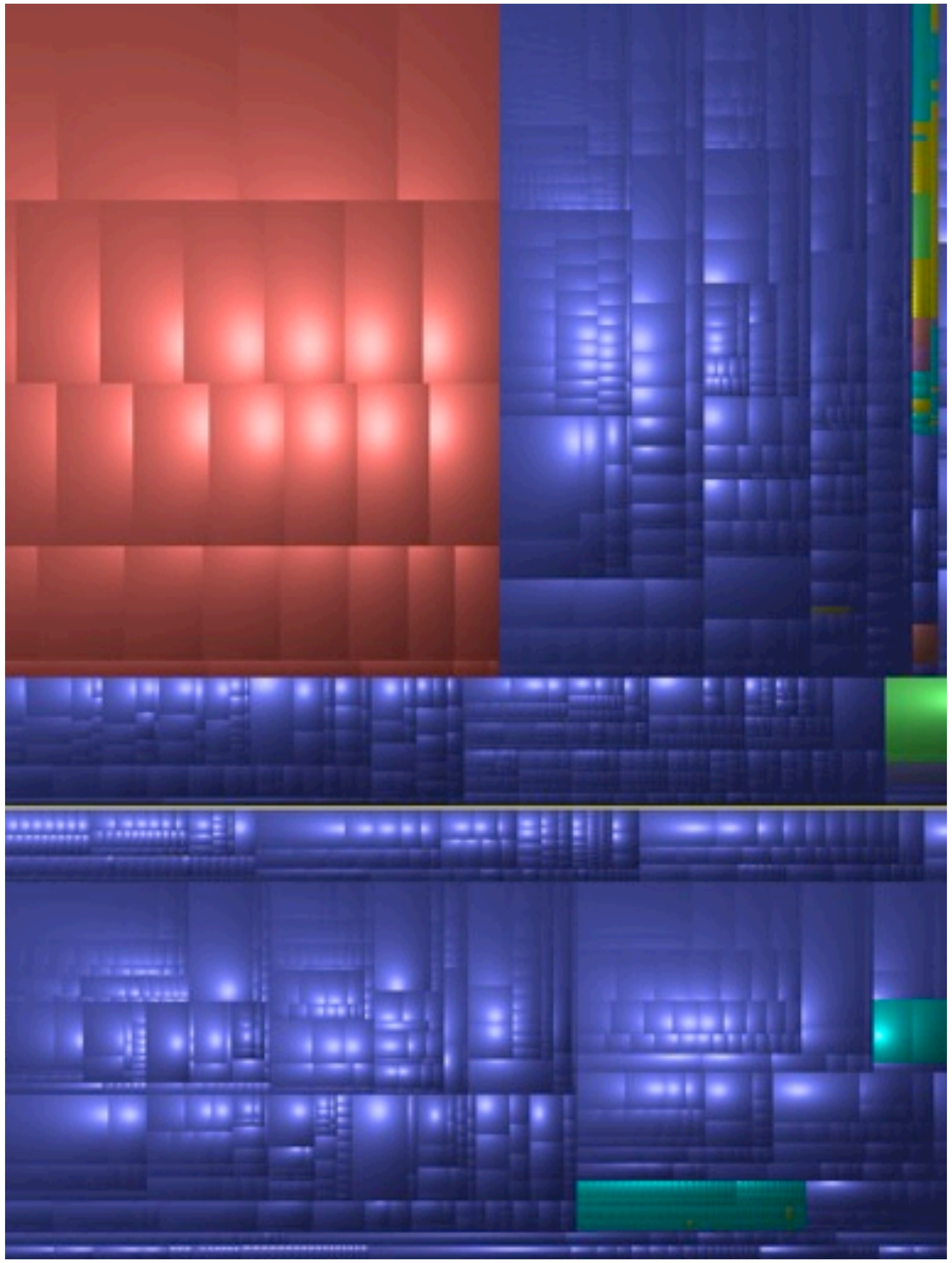
Examples

Treemaps

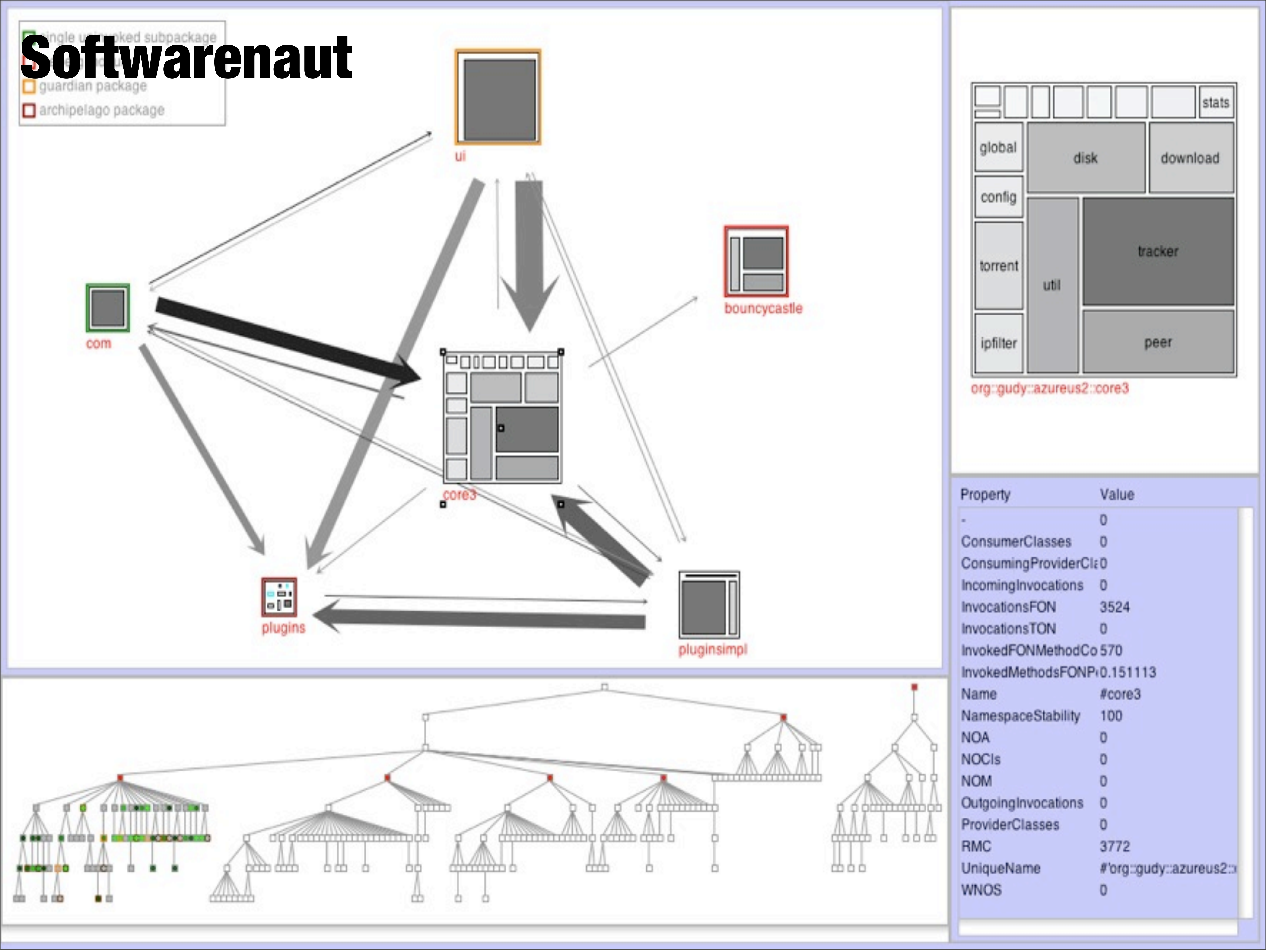


Treemaps

- Pros
  - 100% screen usage
  - Scalability
- Cons
  - Interpretation
  - Information overload
- Reflections
  - Excellent for hierarchical data



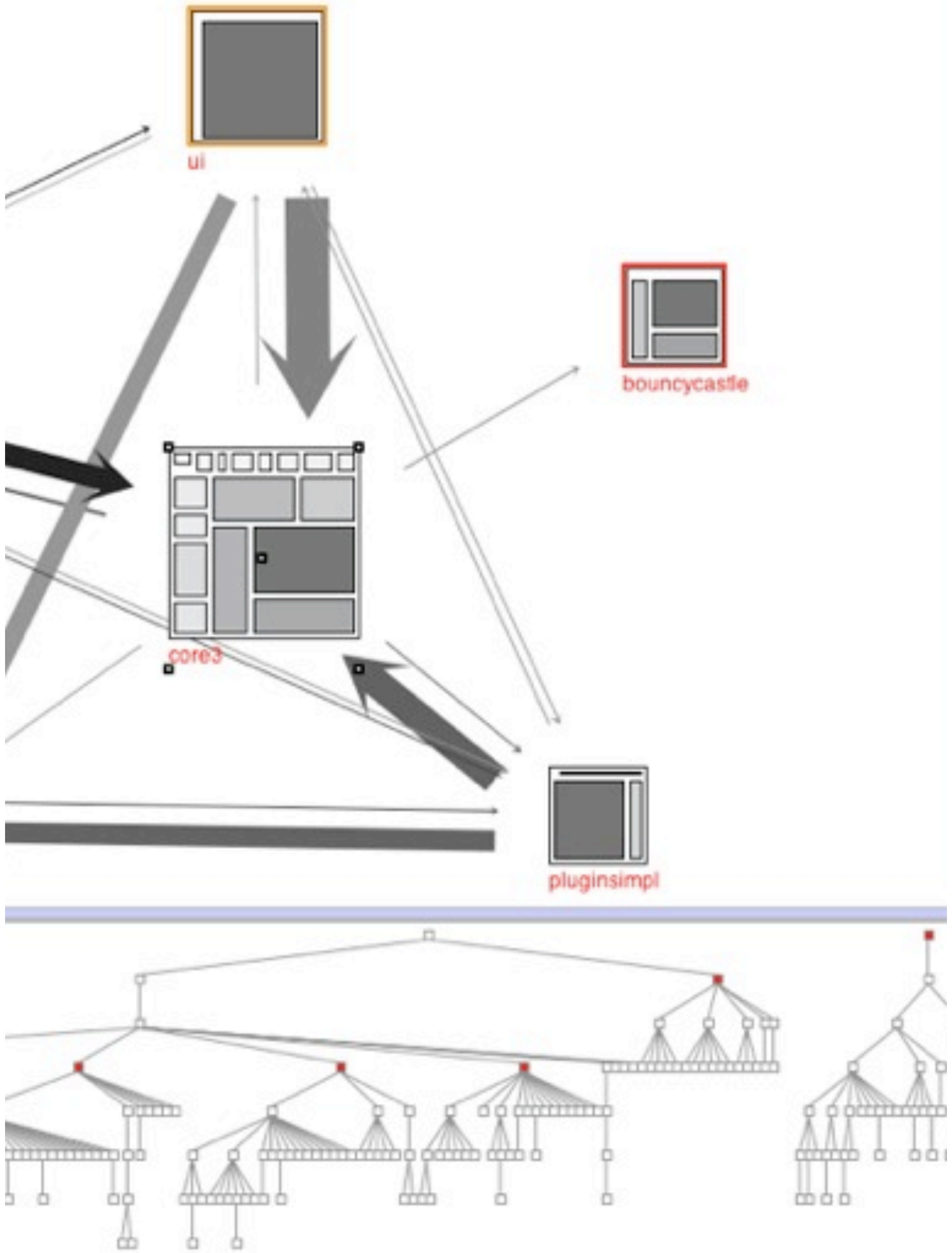
Softwarenaut



Property	Value
-	0
ConsumerClasses	0
ConsumingProviderCl	0
IncomingInvocations	0
InvocationsFON	3524
InvocationsTON	0
InvokedFONMethodCo	570
InvokedMethodsFONP	0.151113
Name	#core3
NamespaceStability	100
NOA	0
NOCls	0
NOM	0
OutgoingInvocations	0
ProviderClasses	0
RMC	3772
UniqueName	#'org::gudy::azureus2::
WNOS	0

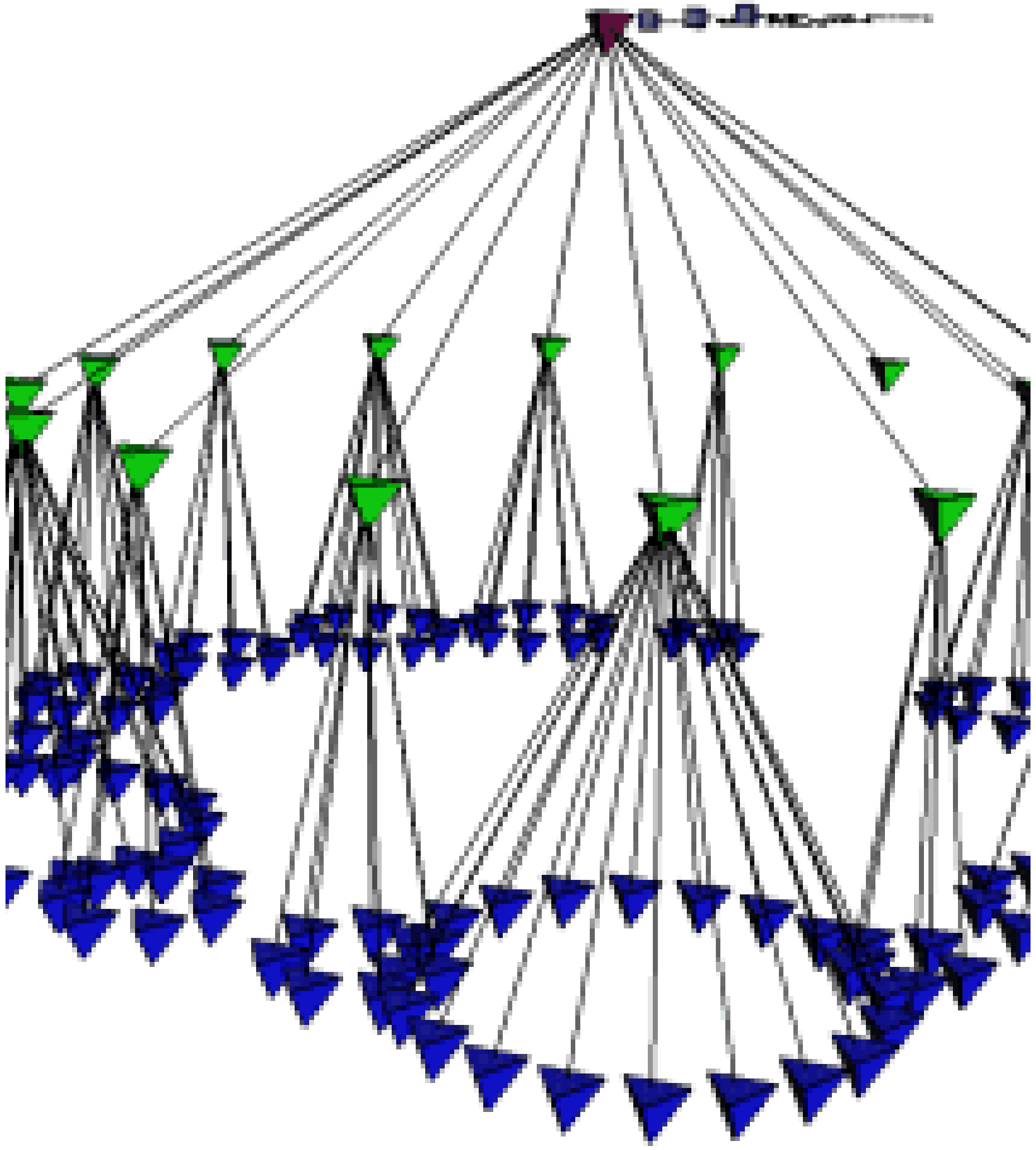
Softwarenaut

- Pros
  - Intuitive, metrics-based, interactive visualization
- Cons
  - Distance to source code
- Reflections
  - The best vertical software exploration tool ever



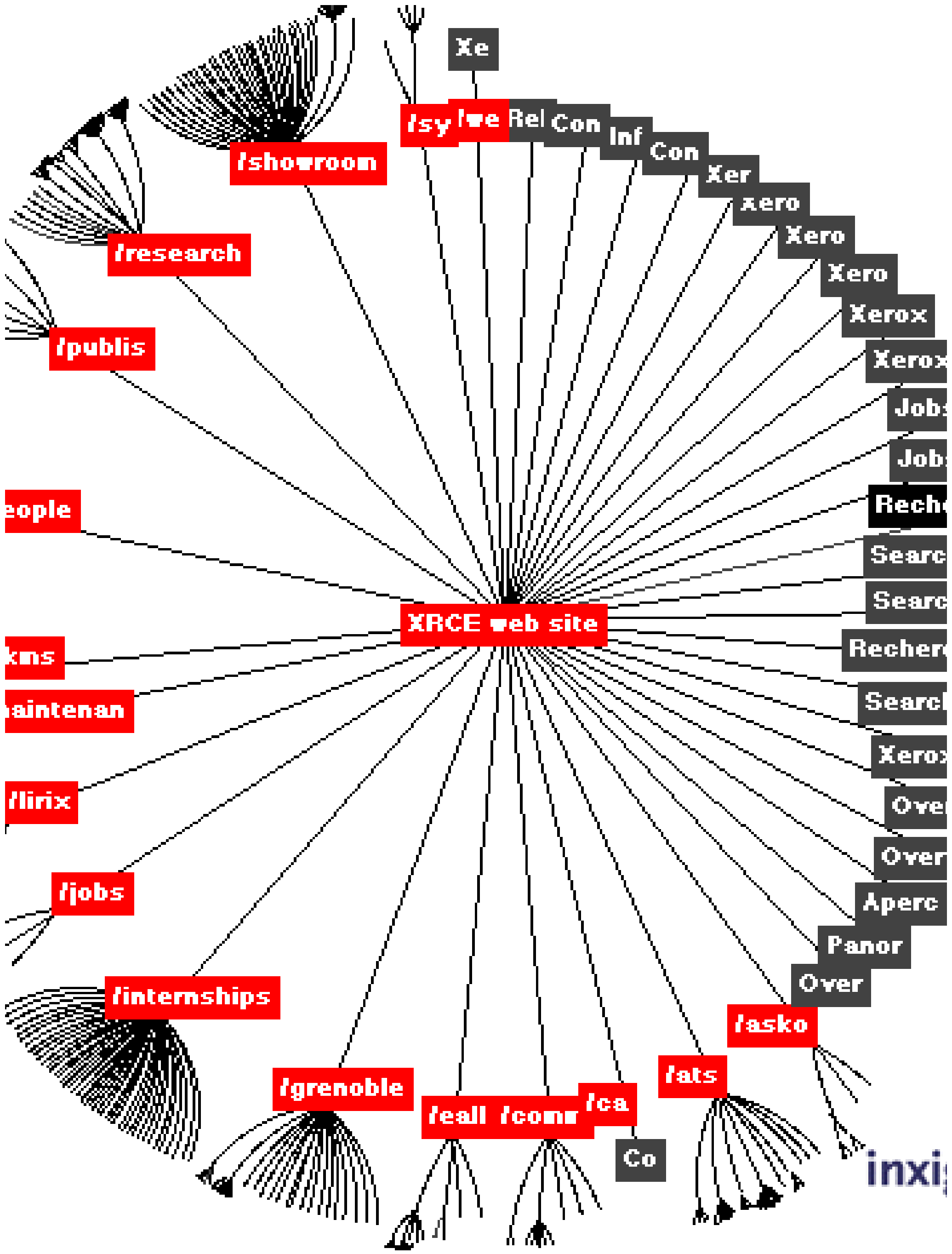
Euclidean Cones

- Pros
  - More information than 2D
- Cons
  - Lack of depth
  - Navigation



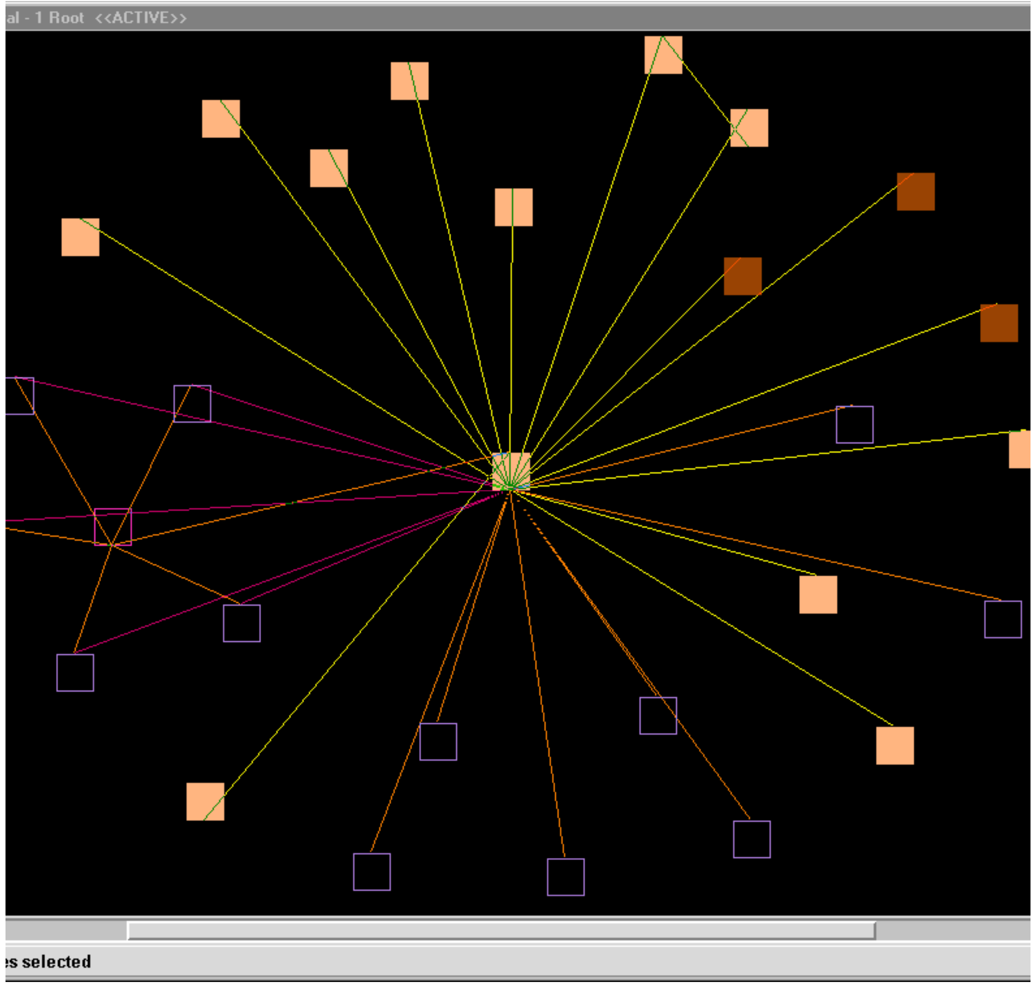
Hyperbolic Trees

- Pros
  - Good focus
  - Dynamic
- Cons
  - Copyrighted!

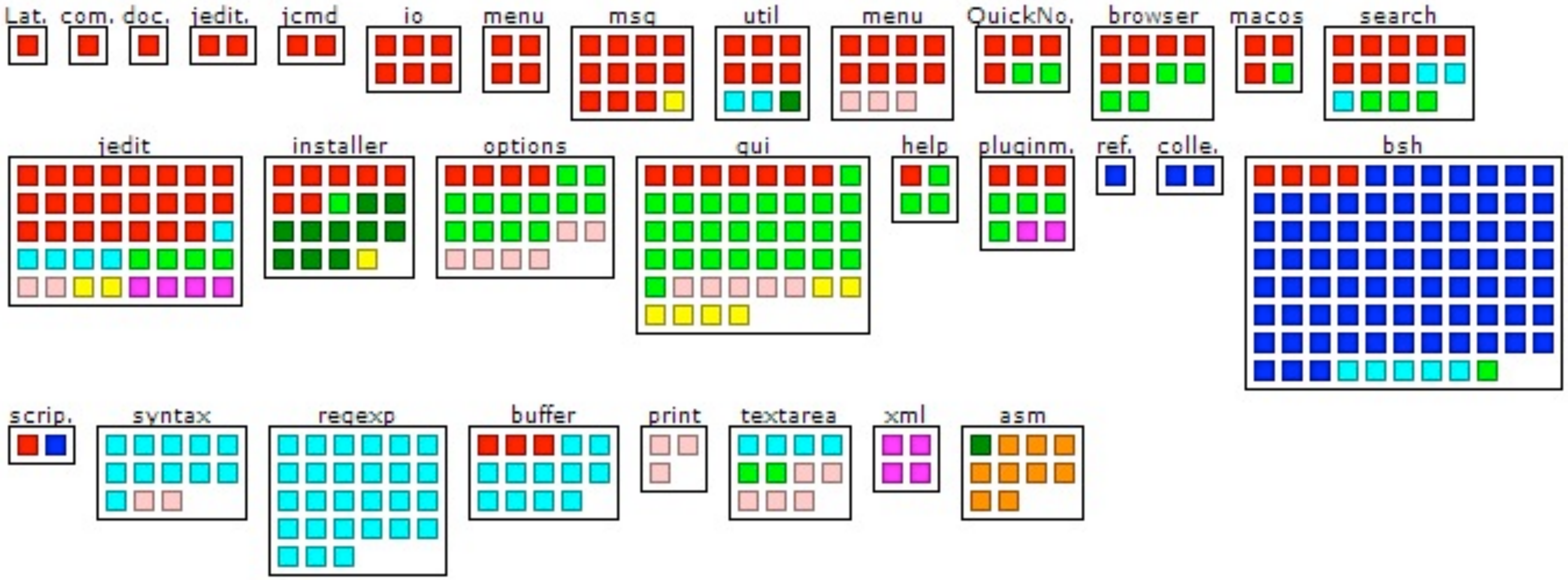


Rigi

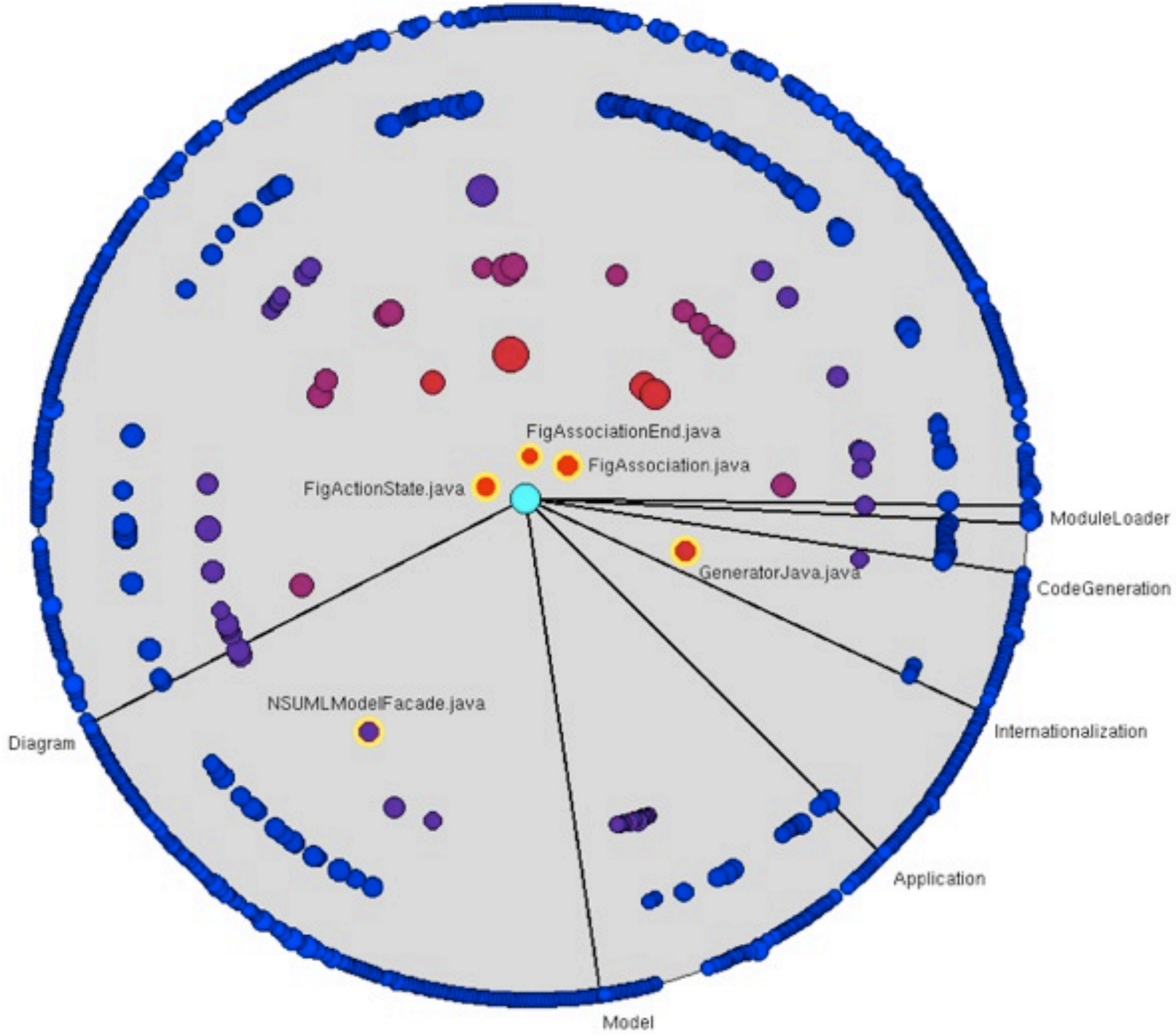
- The grandfather of software visualization tools
- Pros
  - Scalability
  - Domain-independent
- Cons
  - Lack of code semantics



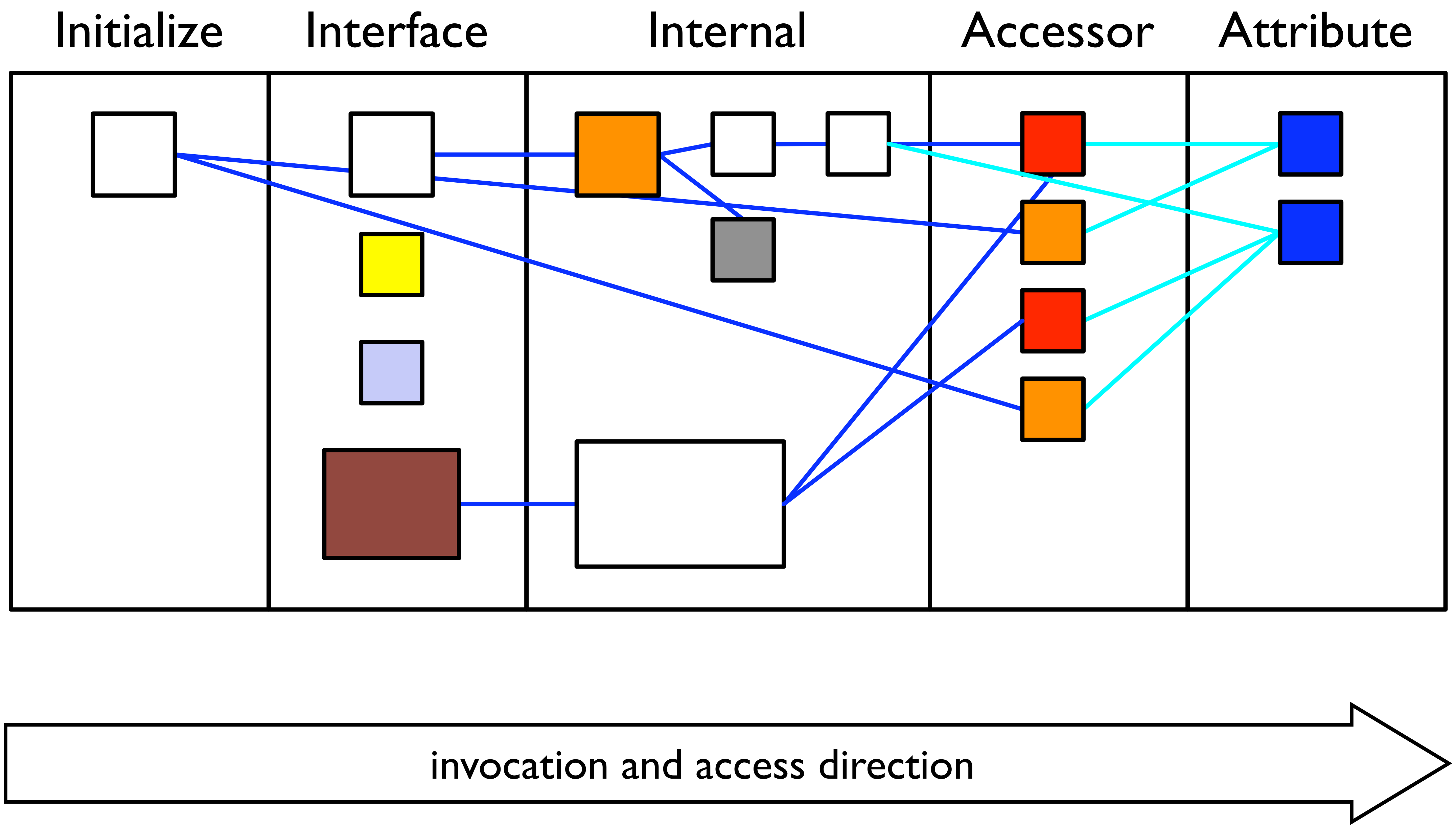
Distribution Maps



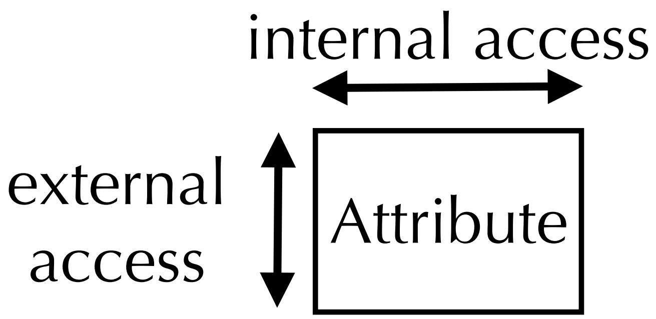
The Evolution Radar

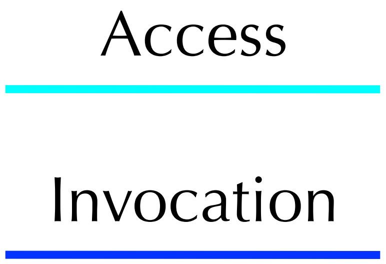


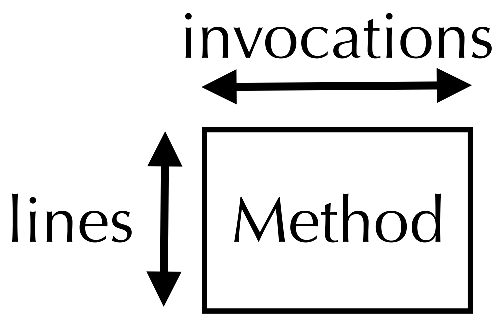
Increasing Information Granularity: The Class Blueprint

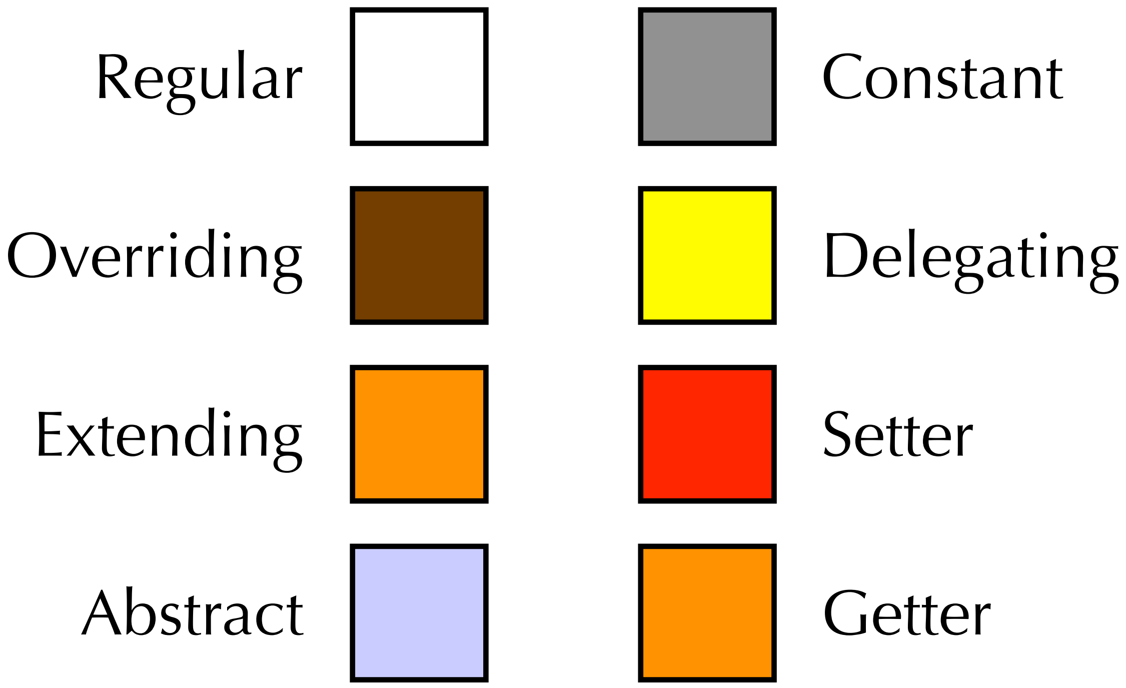


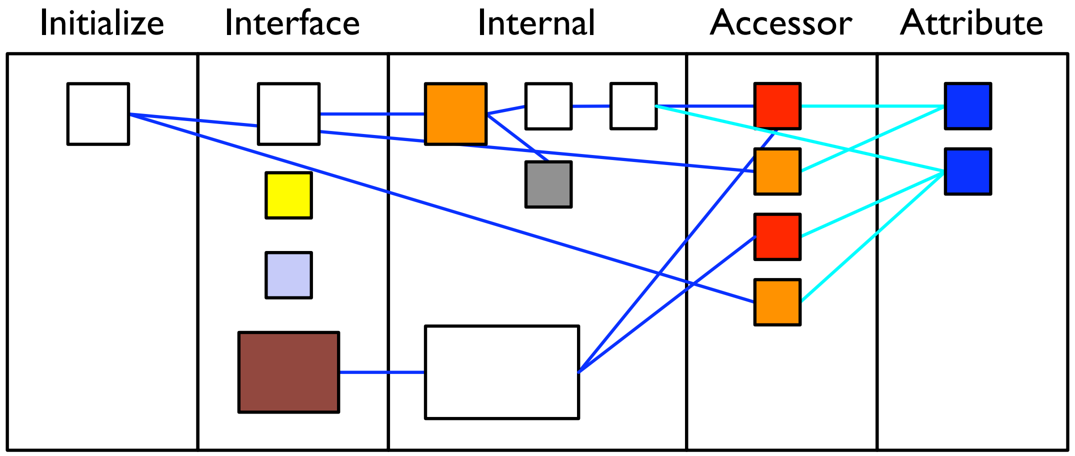
Detailing Class Blueprints



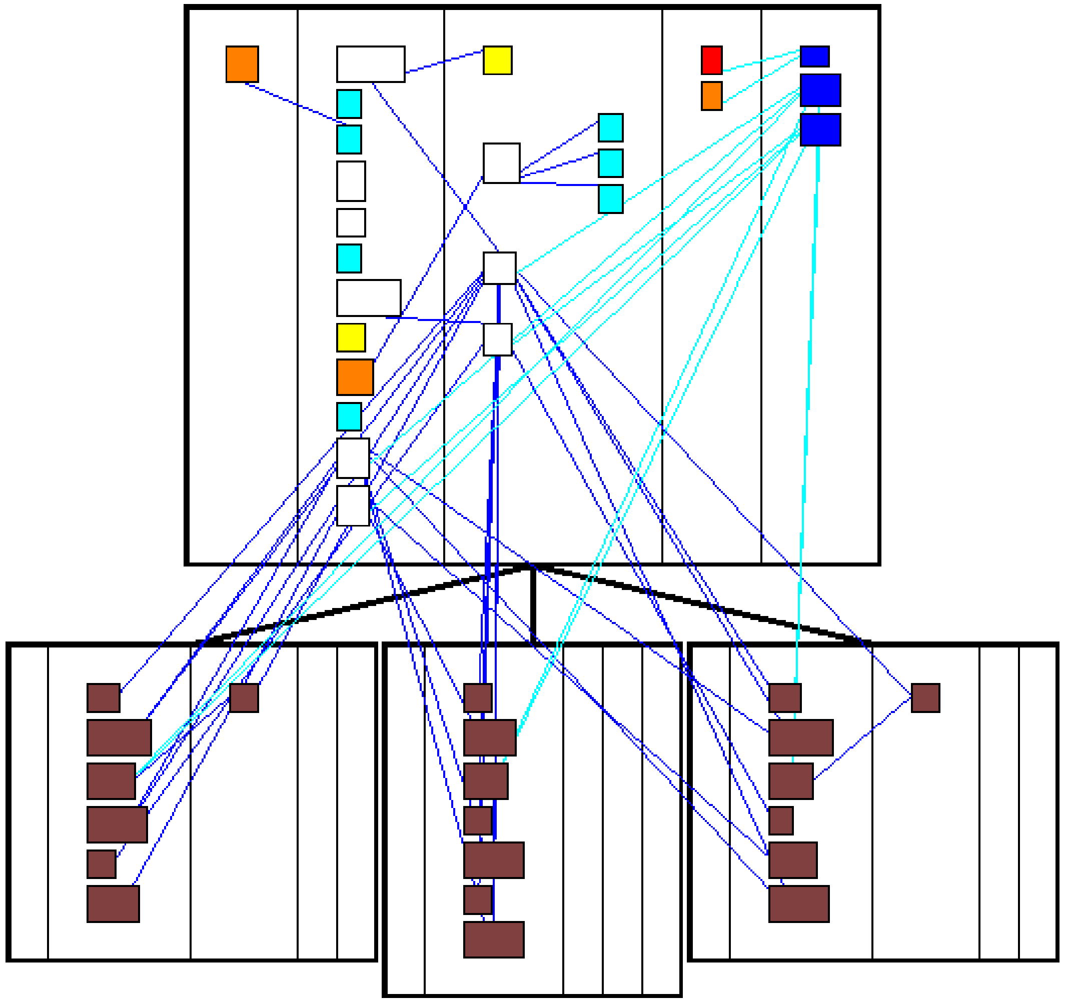


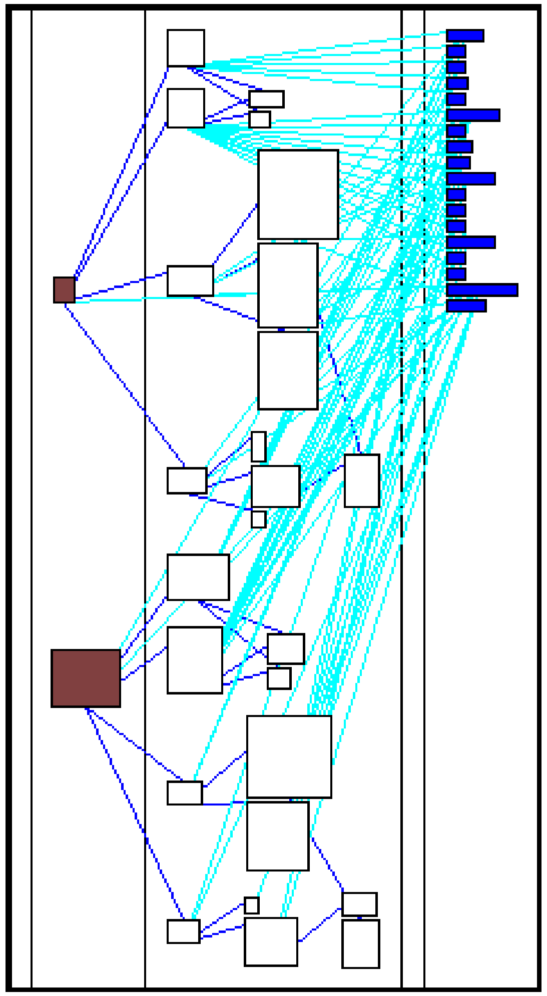




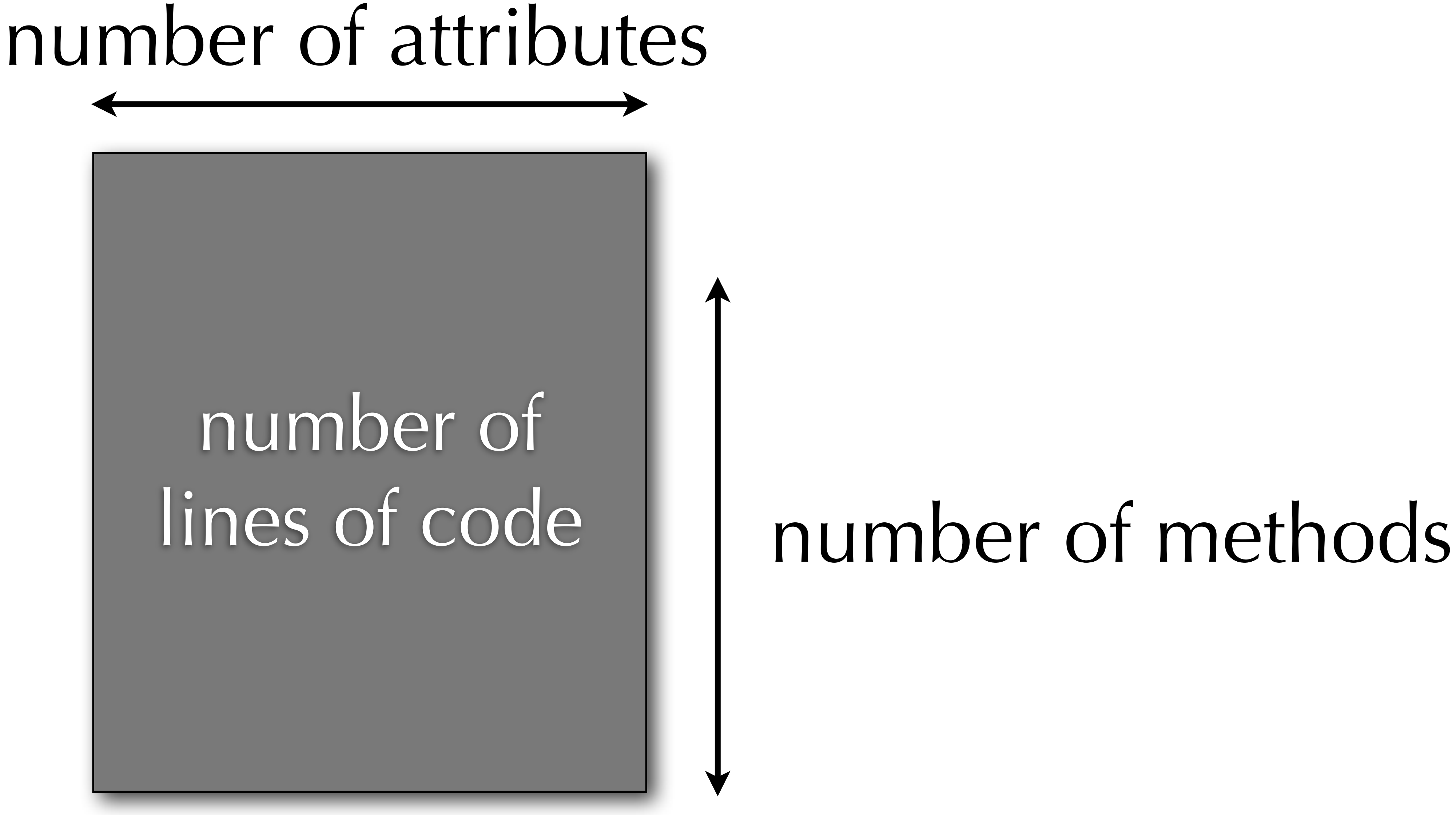


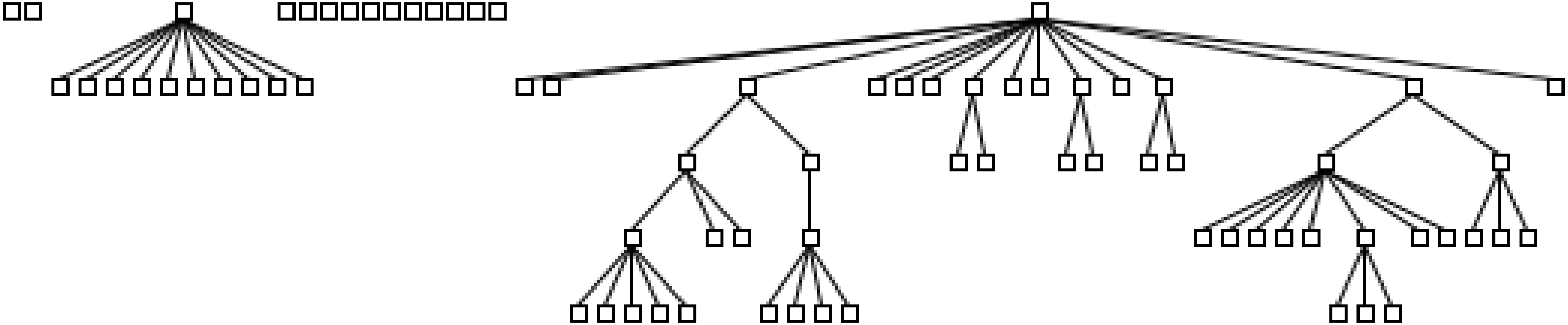
A Pattern Language based on Class Blueprints



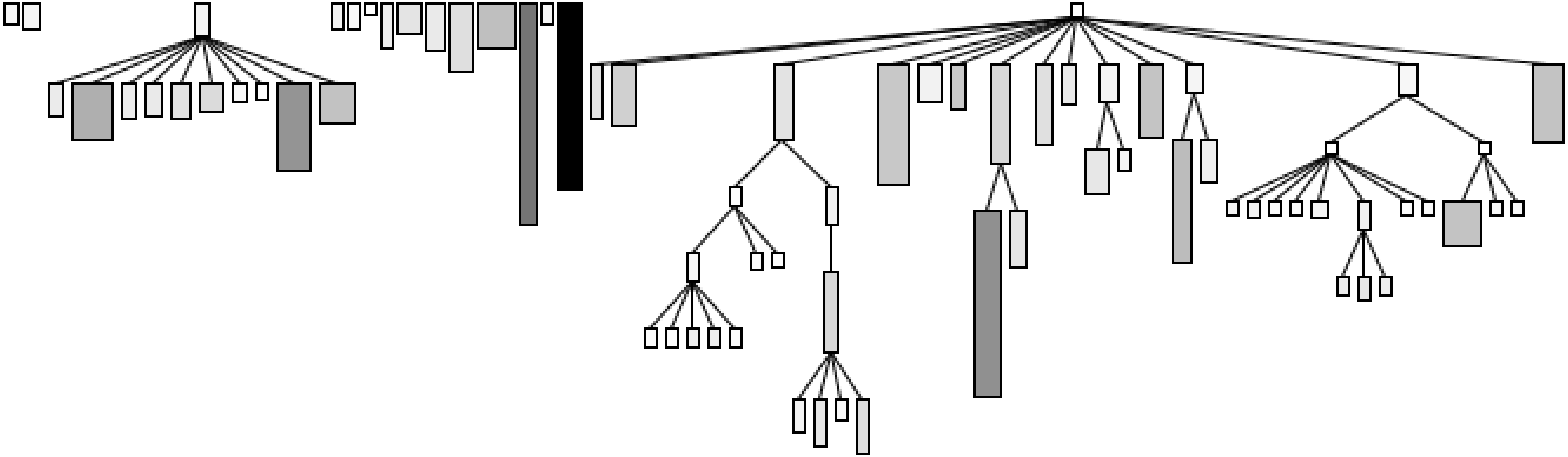


The Polymetric View Principle

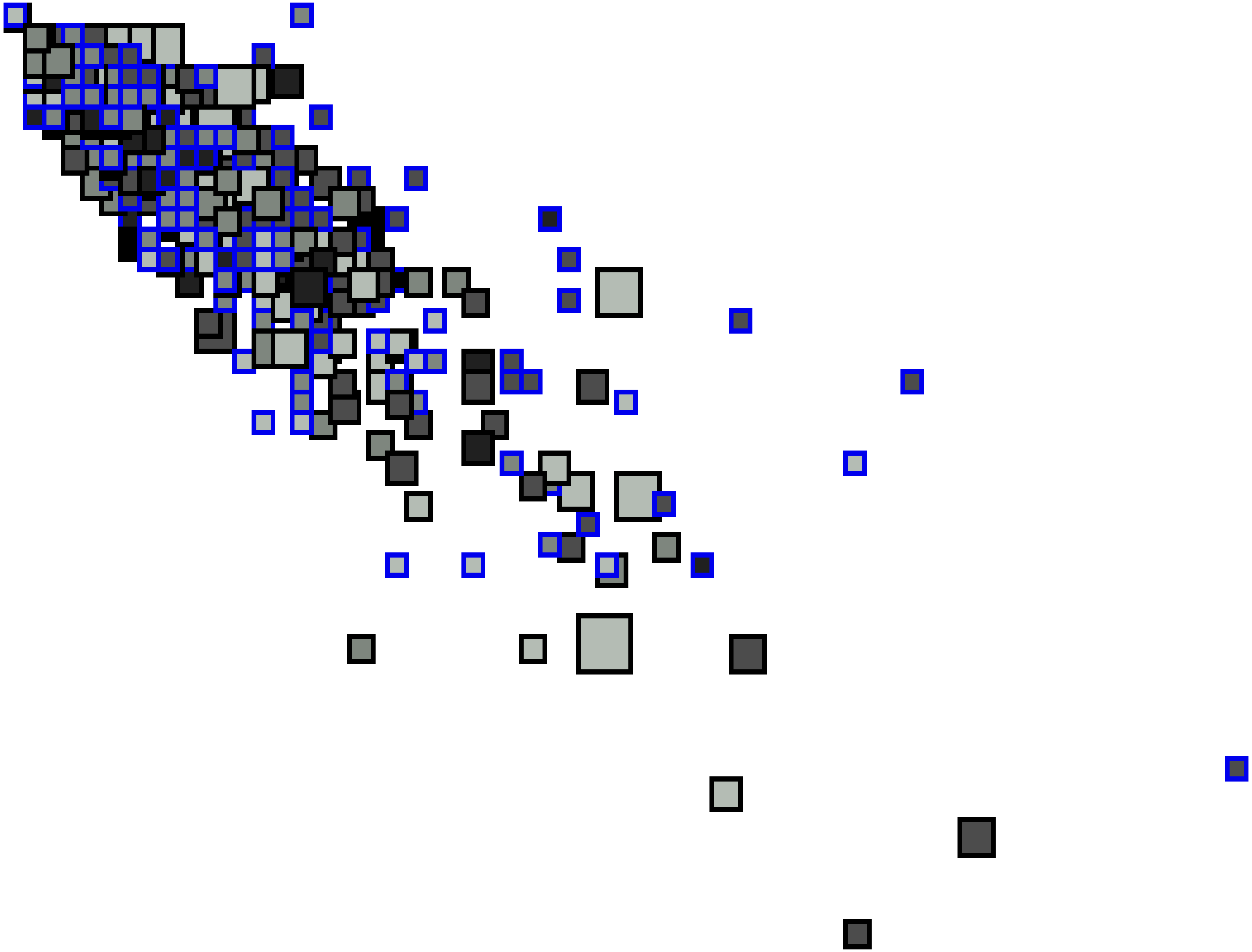


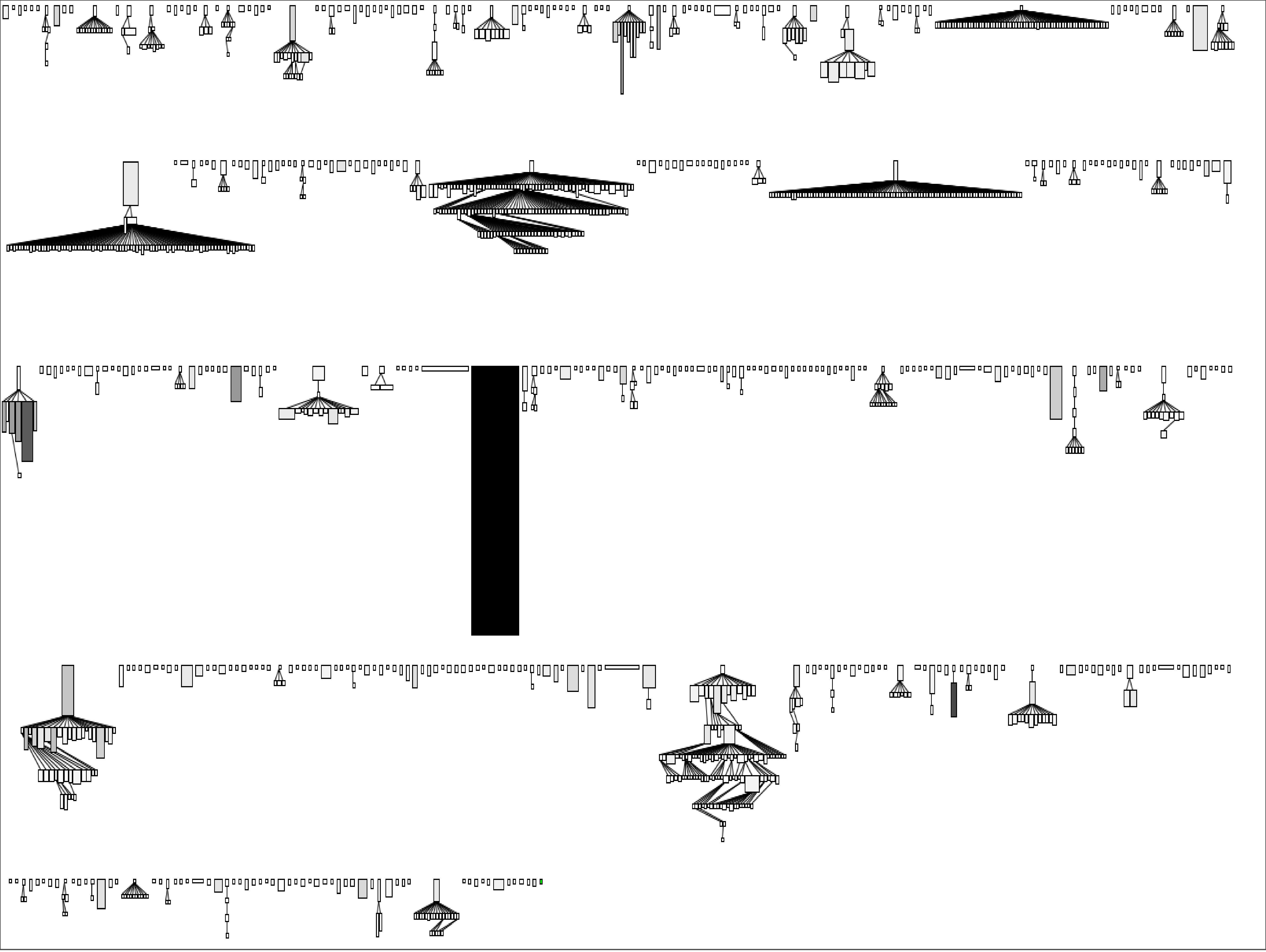


System Complexity View

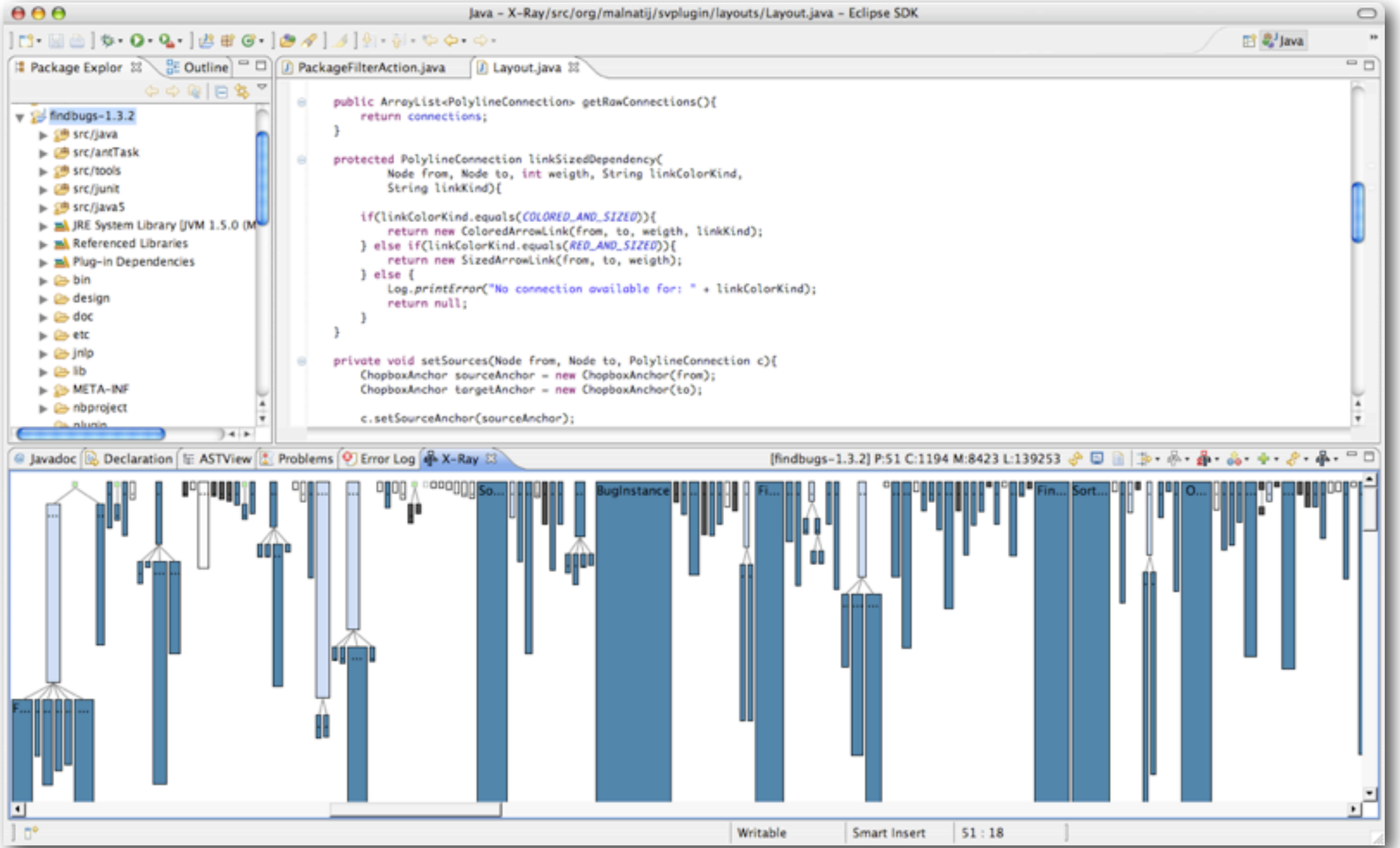


a simple and powerful concept





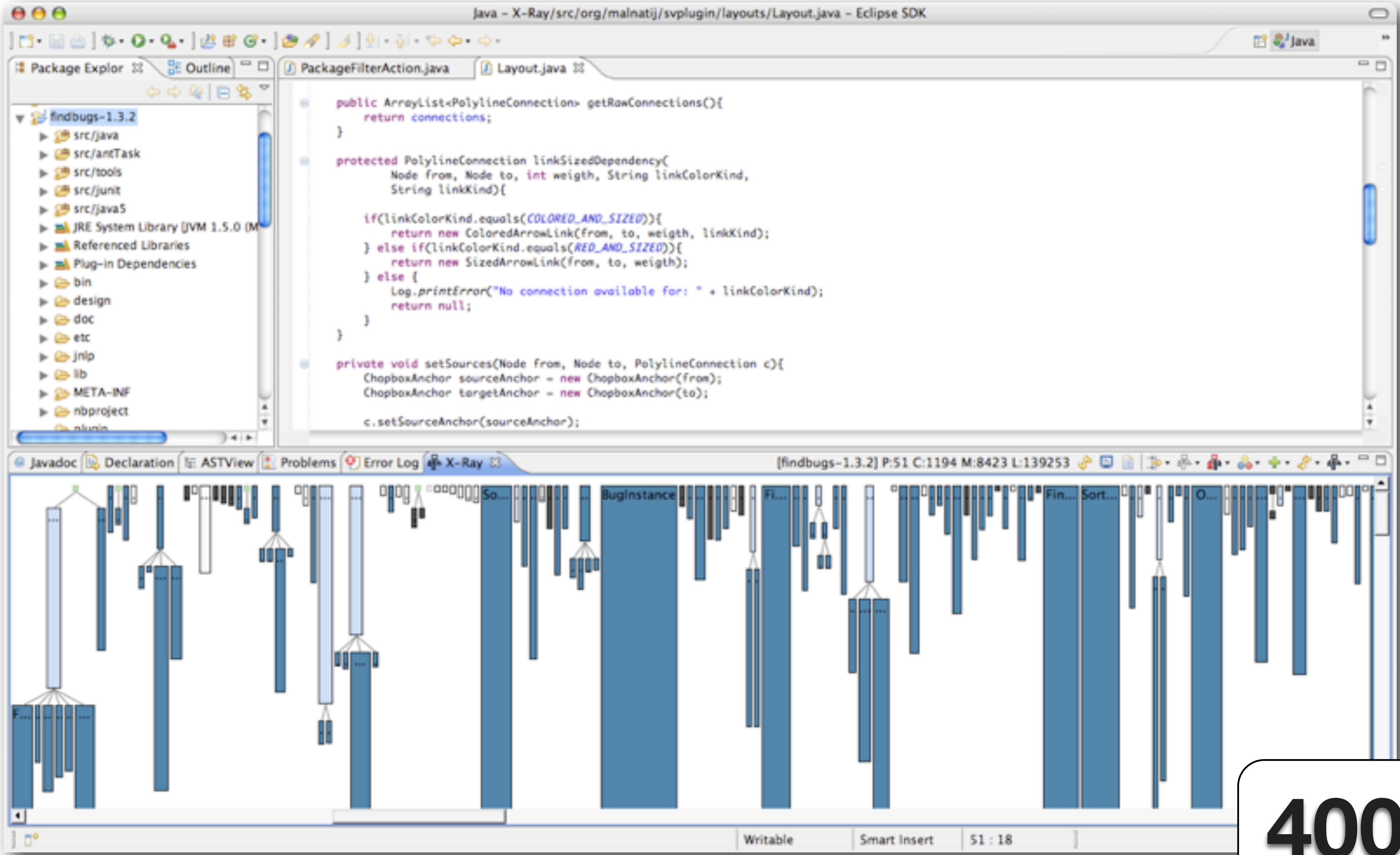
http://xray.inf.usi.ch/xray.php



```
public ArrayList<PolylineConnection> getRawConnections(){
    return connections;
}

protected PolylineConnection linkSizedDependency(
        Node from, Node to, int weigth, String linkColorKind,
        String linkKind){

    if(linkColorKind.equals(COLORED_AND_SIZED)){
        return new ColoredArrowLink(from, to, weigth, linkKind);
    } else if(linkColorKind.equals(RED_AND_SIZED)){
        return new SizedArrowLink(from, to, weigth);
    } else {
        Log.printError("No connection available for: " + linkColorKind);
        return null;
    }
}

private void setSources(Node from, Node to, PolylineConnection c){
    ChopboxAnchor sourceAnchor = new ChopboxAnchor(from);
    ChopboxAnchor targetAnchor = new ChopboxAnchor(to);

    c.setSourceAnchor(sourceAnchor);
```

Released:
Nov 2007

http://xray.inf.usi.ch/xray.php



```
public ArrayList<PolylineConnection> getRawConnections(){
    return connections;
}

protected PolylineConnection linkSizedDependency(
        Node from, Node to, int weigth, String linkColorKind,
        String linkKind){

    if(linkColorKind.equals(COLORED_AND_SIZED)){
        return new ColoredArrowLink(from, to, weigth, linkKind);
    } else if(linkColorKind.equals(RED_AND_SIZED)){
        return new SizedArrowLink(from, to, weigth);
    } else {
        Log.printError("No connection available for: " + linkColorKind);
        return null;
    }
}

private void setSources(Node from, Node to, PolylineConnection c){
    ChopboxAnchor sourceAnchor = new ChopboxAnchor(from);
    ChopboxAnchor targetAnchor = new ChopboxAnchor(to);

    c.setSourceAnchor(sourceAnchor);
```

Released:
Nov 2007

4000 +
downloads
free

Reflections on Static Visualization

Reflections on Static Visualization

- Pros

Reflections on Static Visualization

- Pros
  - Intuitive

Reflections on Static Visualization

- Pros
  - Intuitive
  - Aesthetically pleasing

Reflections on Static Visualization

- Pros
  - Intuitive
  - Aesthetically pleasing
- Cons

Reflections on Static Visualization

- Pros
  - Intuitive
  - Aesthetically pleasing
- Cons
  - Several approaches are orthogonal to each other

Reflections on Static Visualization

- Pros
  - Intuitive
  - Aesthetically pleasing
- Cons
  - Several approaches are orthogonal to each other
  - No conventions

Reflections on Static Visualization

- Pros
    - Intuitive
    - Aesthetically pleasing
- Cons
    - Several approaches are orthogonal to each other
    - No conventions
    - Too easy to produce meaningless results

Reflections on Static Visualization

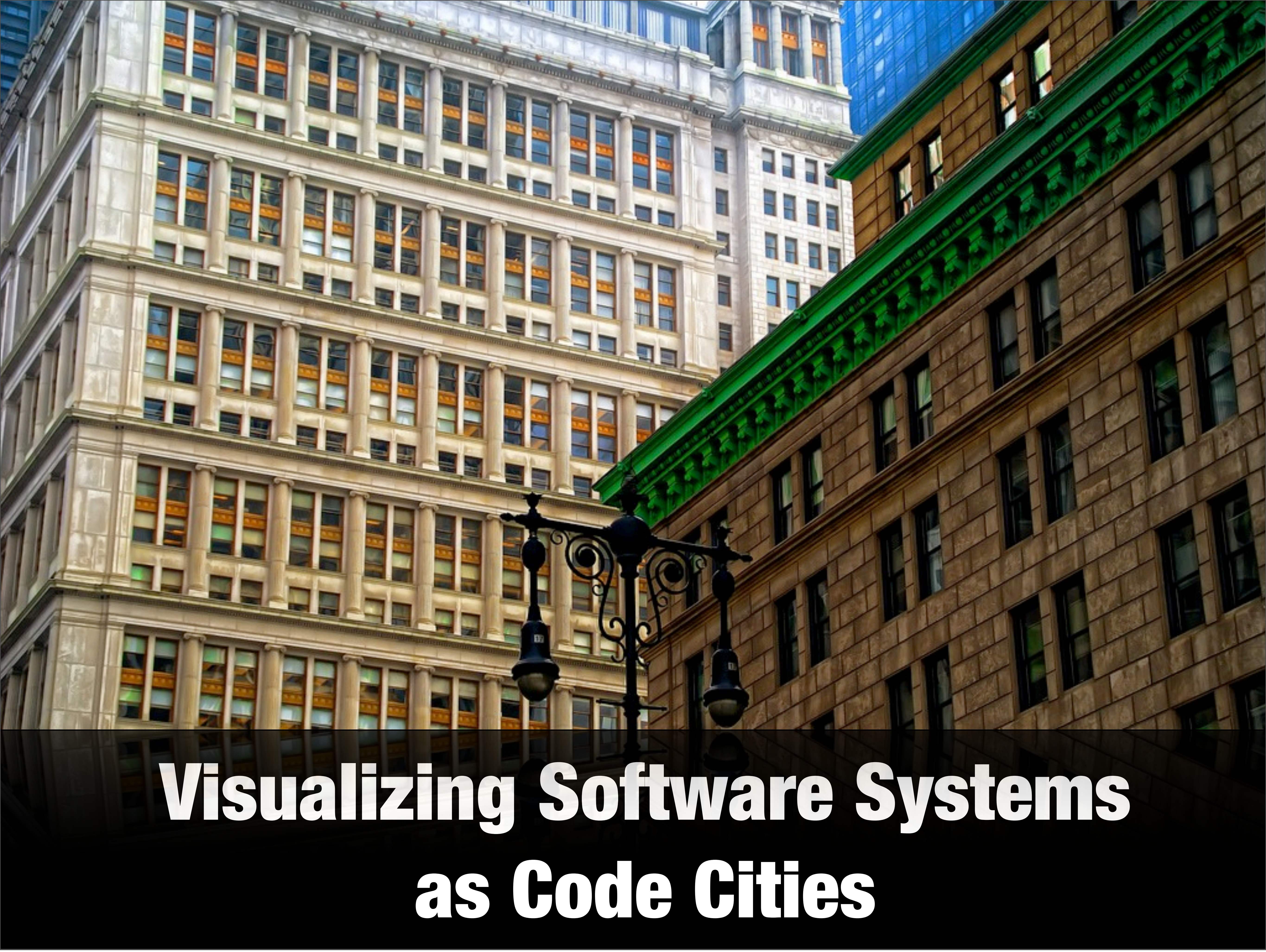
- Pros
  - Intuitive
  - Aesthetically pleasing
- Cons
  - Several approaches are orthogonal to each other
  - No conventions
  - Too easy to produce meaningless results
  - Scaling up is possible at the expense of semantics

Reflections on Static Visualization

- Pros
  - Intuitive
  - Aesthetically pleasing
- Cons
  - Several approaches are orthogonal to each other
  - No conventions
  - Too easy to produce meaningless results
  - Scaling up is possible at the expense of semantics
- Orthogonally

Reflections on Static Visualization

- Pros
  - Intuitive
  - Aesthetically pleasing
- Cons
  - Several approaches are orthogonal to each other
  - No conventions
  - Too easy to produce meaningless results
  - Scaling up is possible at the expense of semantics
- Orthogonally
  - Without programming knowledge it's only colored boxes and arrows..



Visualizing Software Systems as Code Cities

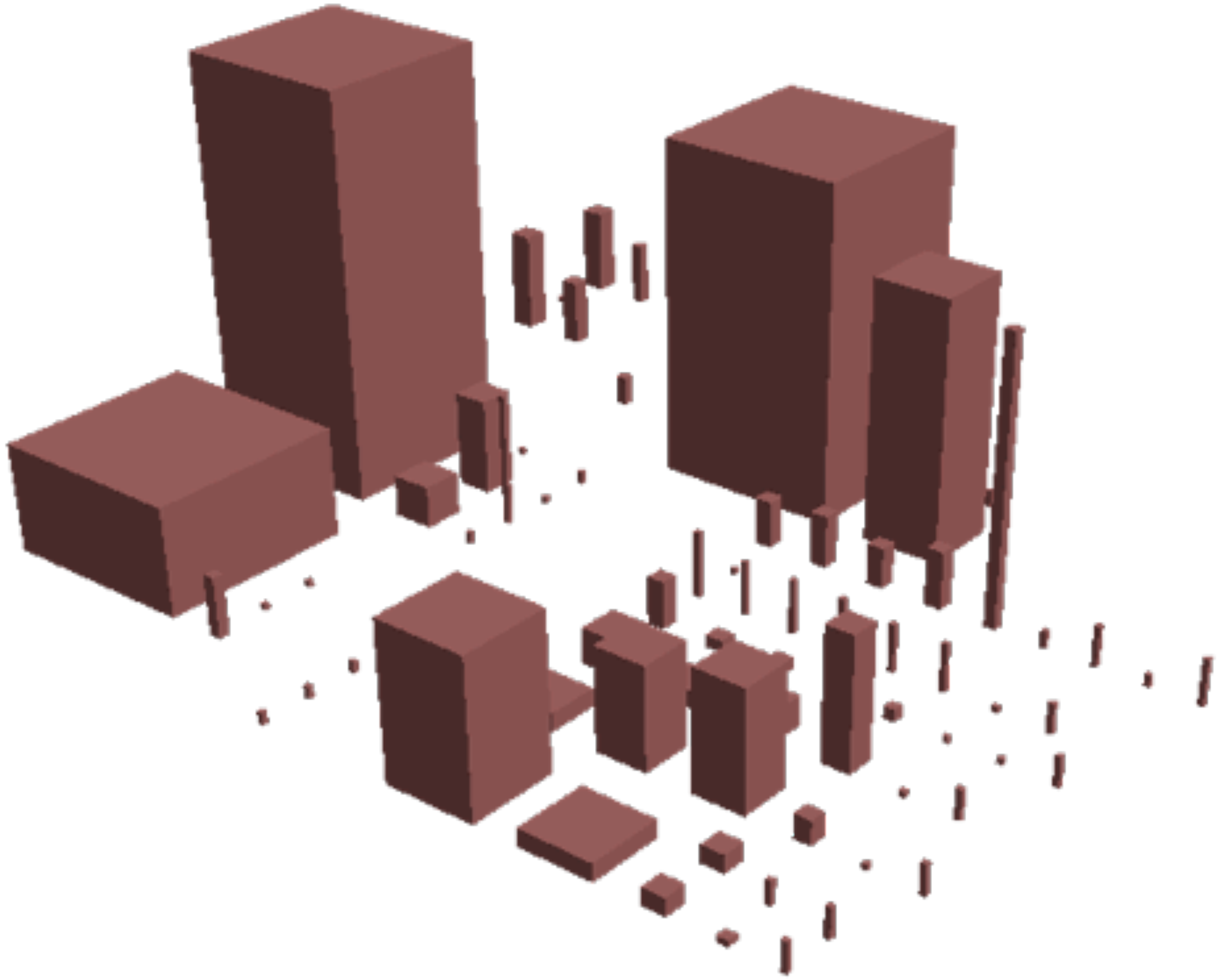
The City Metaphor

The City Metaphor

domain mapping	

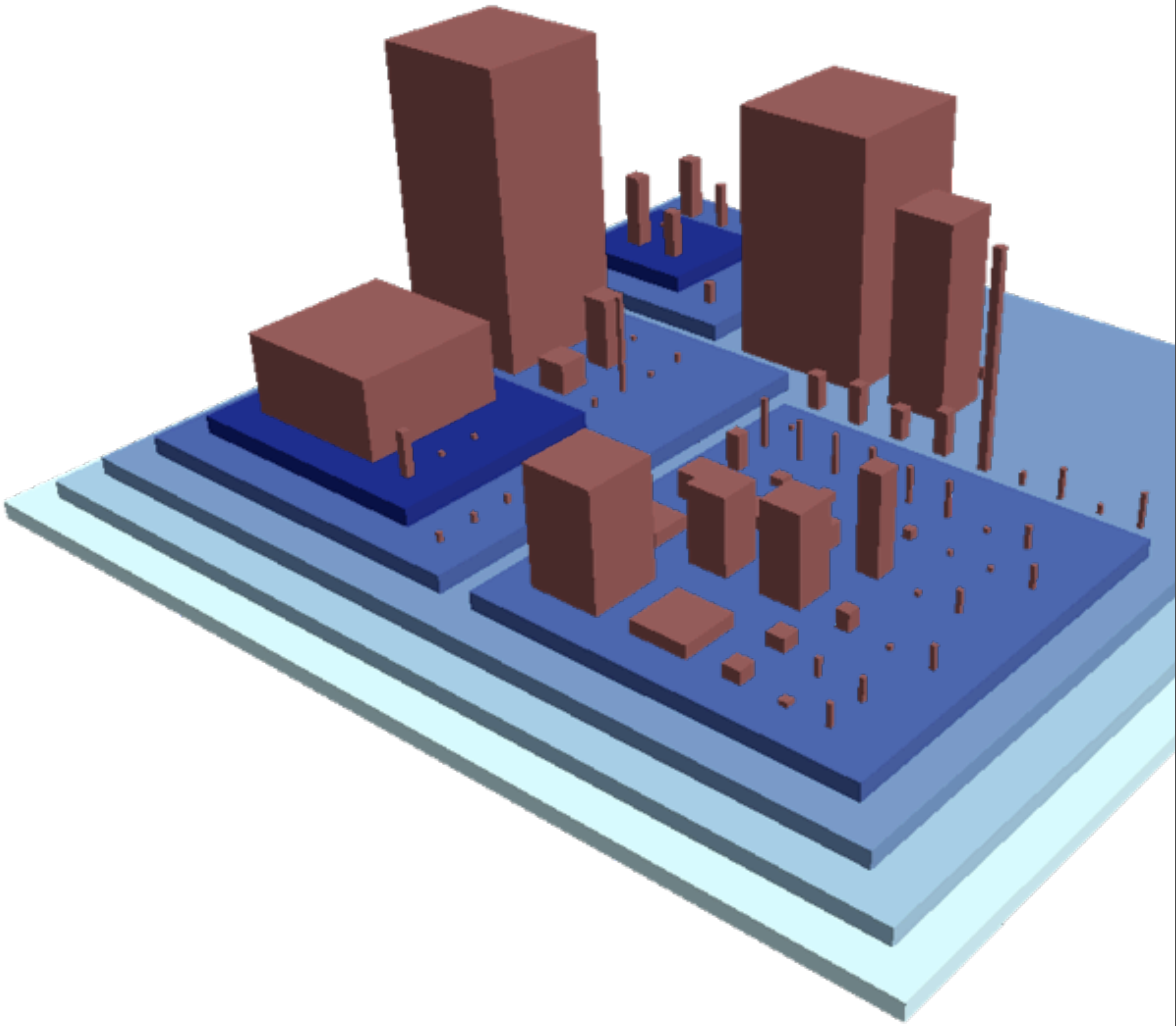
The City Metaphor

domain mapping	
classes	buildings



The City Metaphor

domain mapping	
classes	buildings
packages	districts

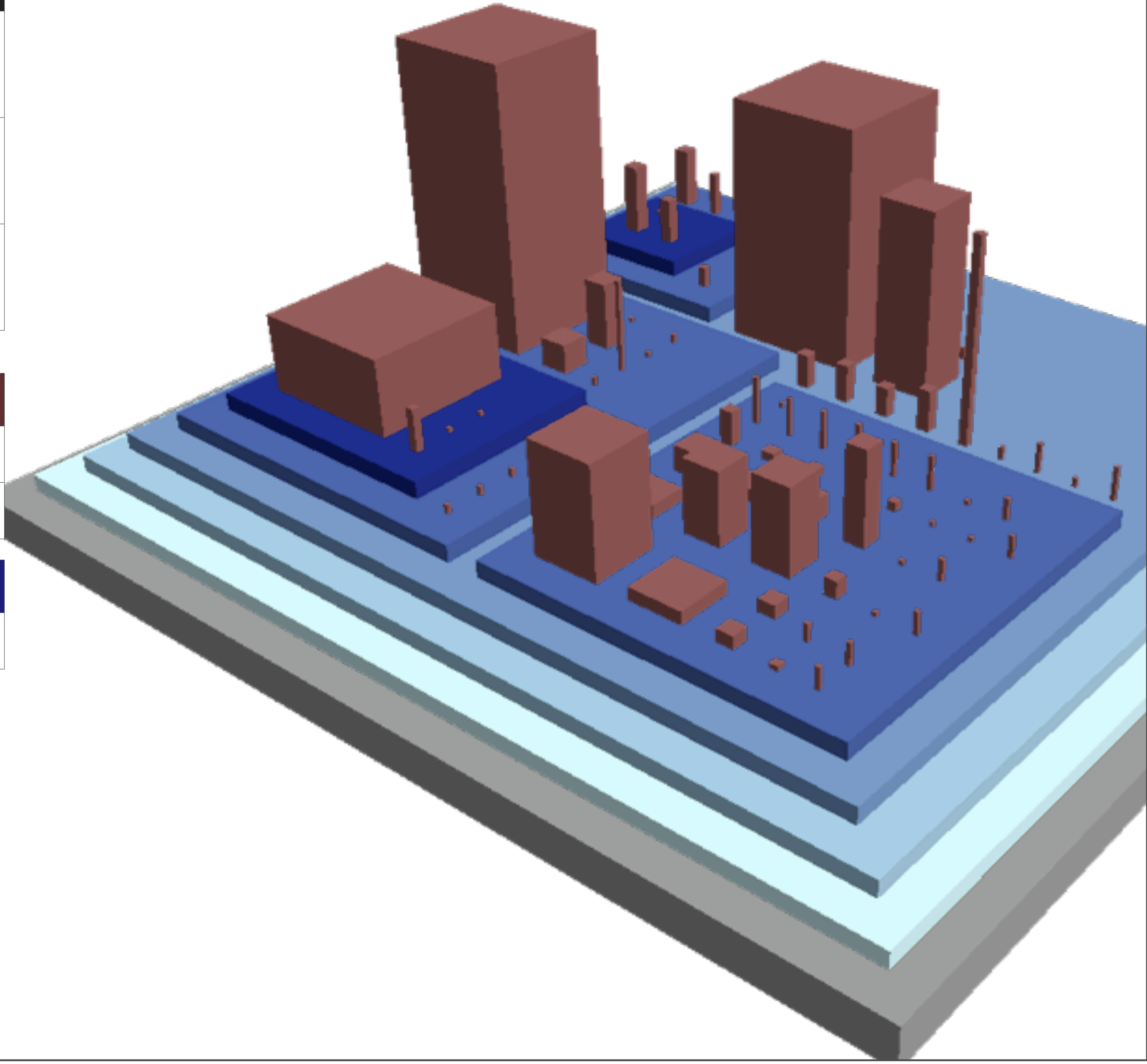


The City Metaphor

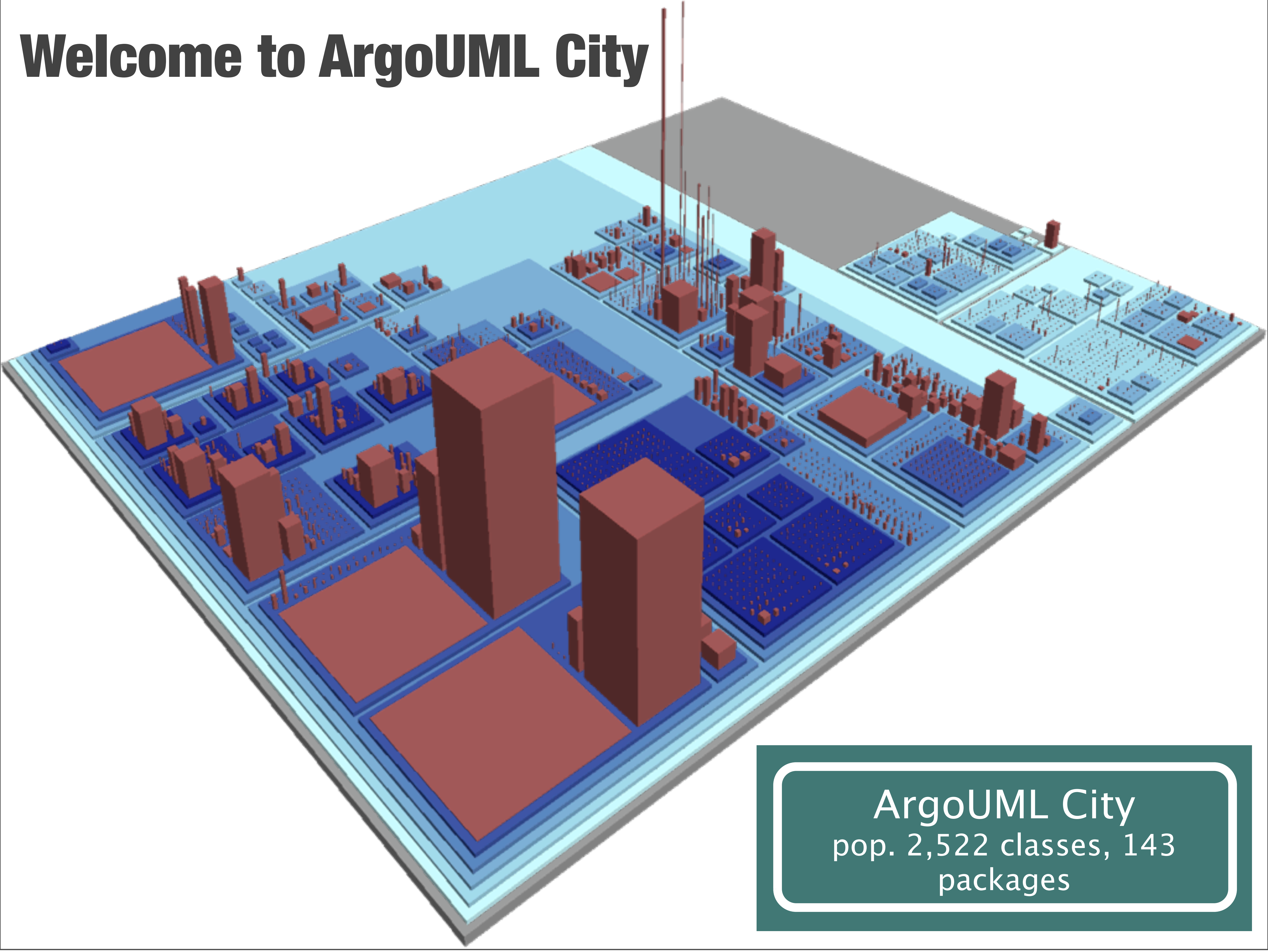
domain mapping	
classes	buildings
packages	districts
system	city

class metric	building property
number of methods (NOM)	height
number of attributes (NOA)	width, length

package metric	district property
nesting level	color



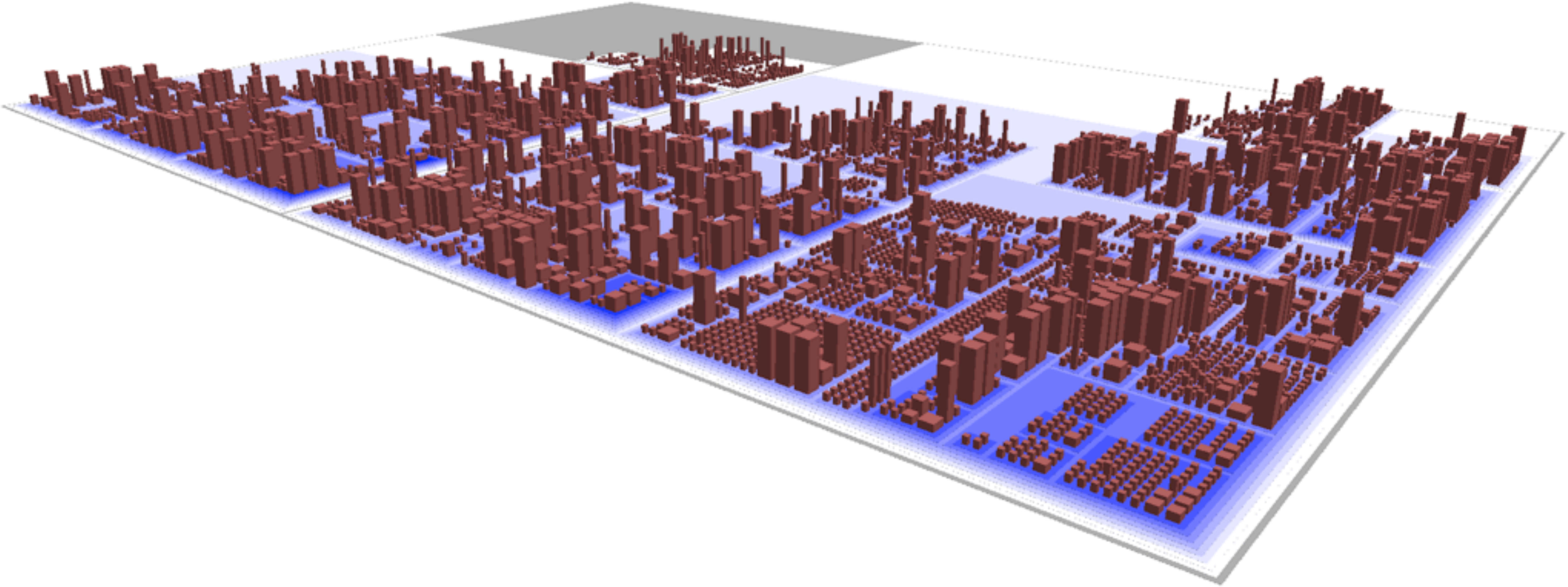
Welcome to ArgoUML City



ArgoUML City

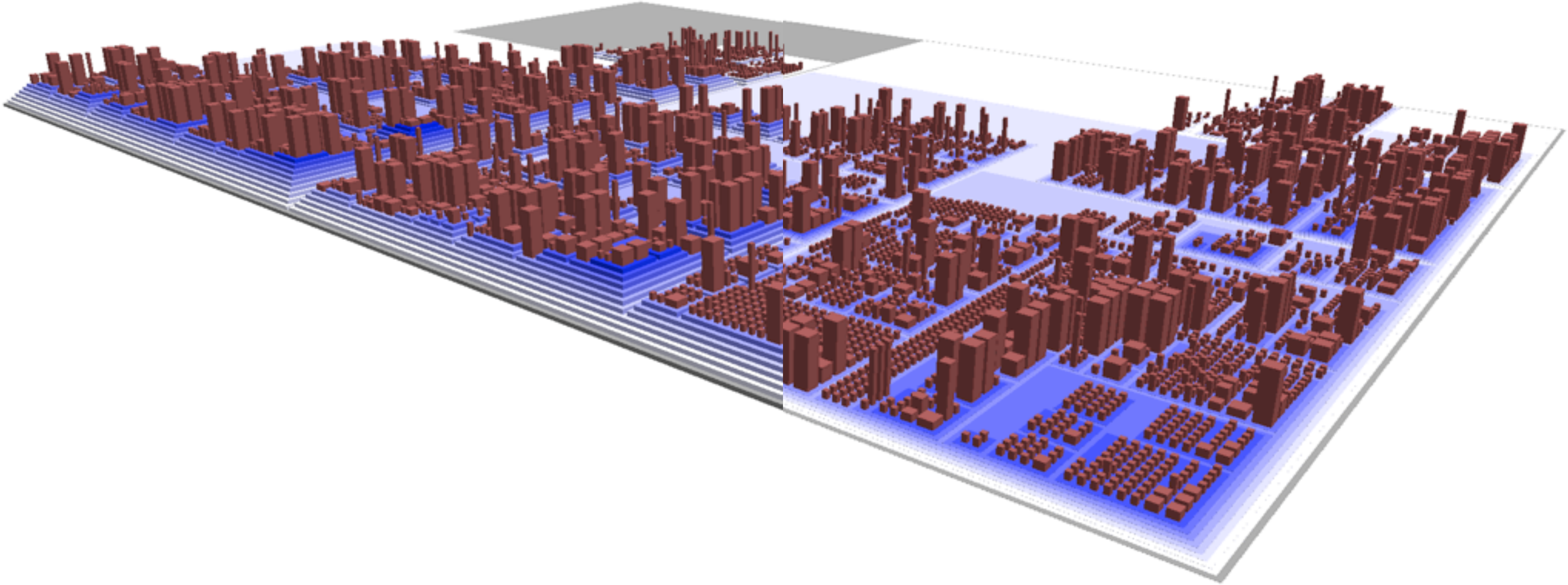
pop. 2,522 classes, 143 packages

Software Topology



Azureus City
pop. 4’500+ classes

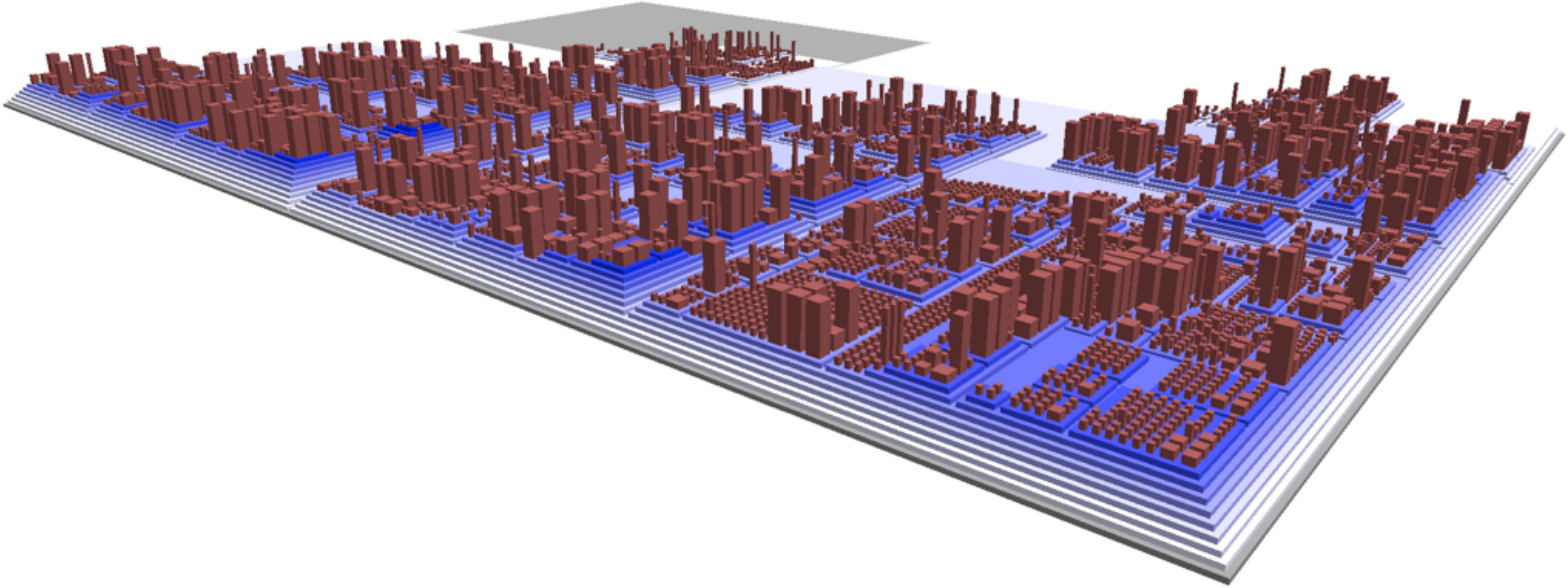
Software Topology



Azureus City

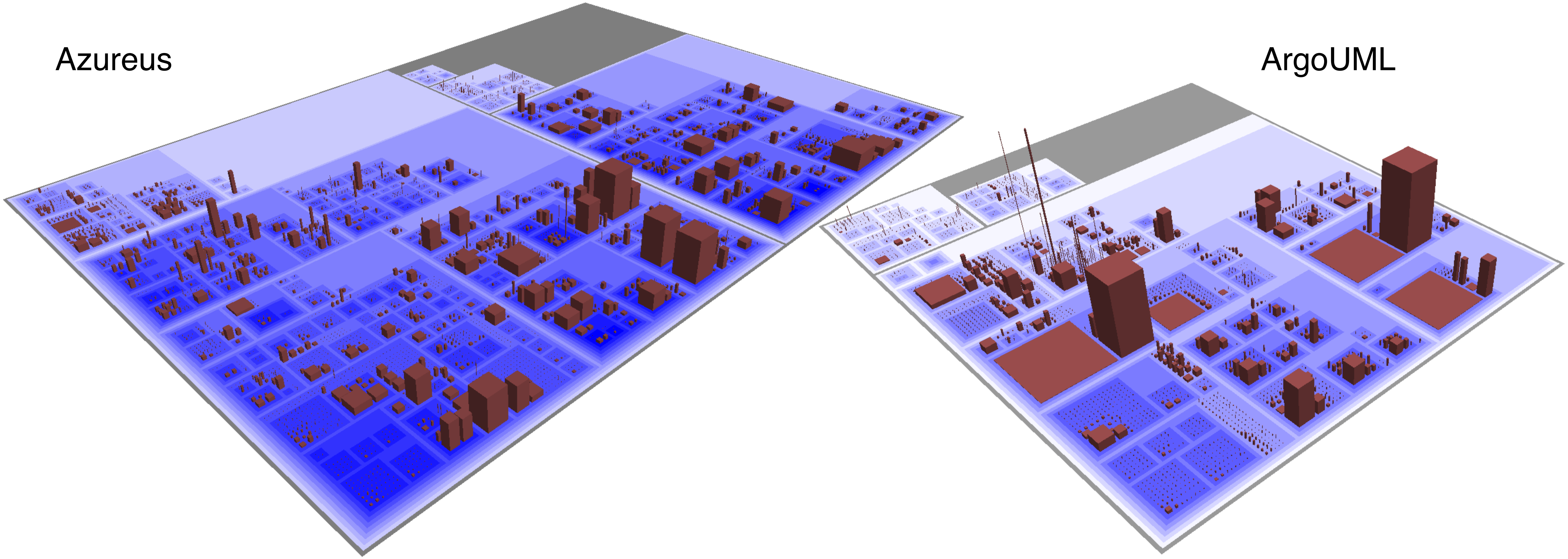
pop. 4’500+ classes

Software Topology

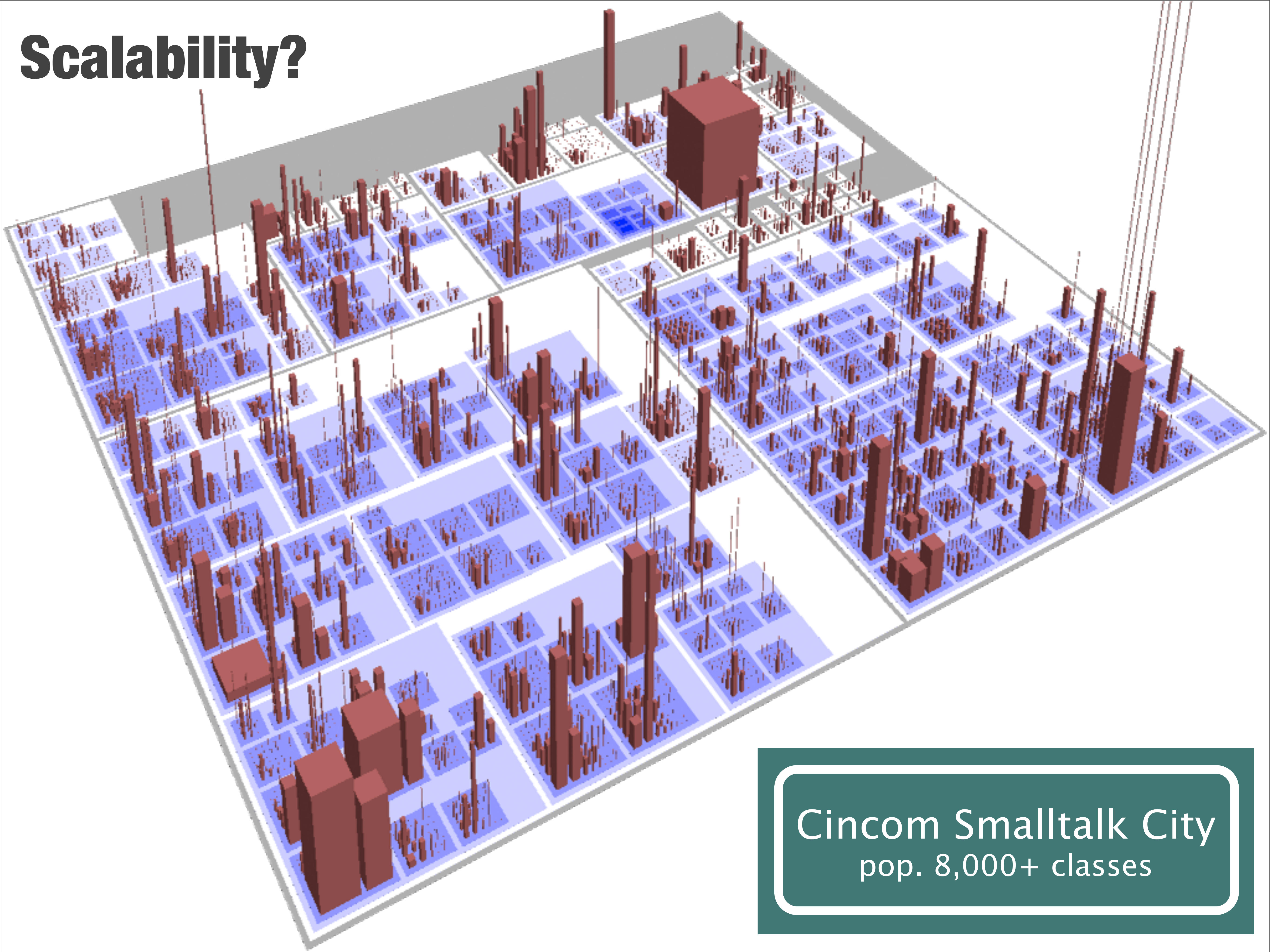


Azureus City
pop. 4’500+ classes

Crossing System Boundaries



Scalability?



Cincom Smalltalk City
pop. 8,000+ classes

Mapping Metrics

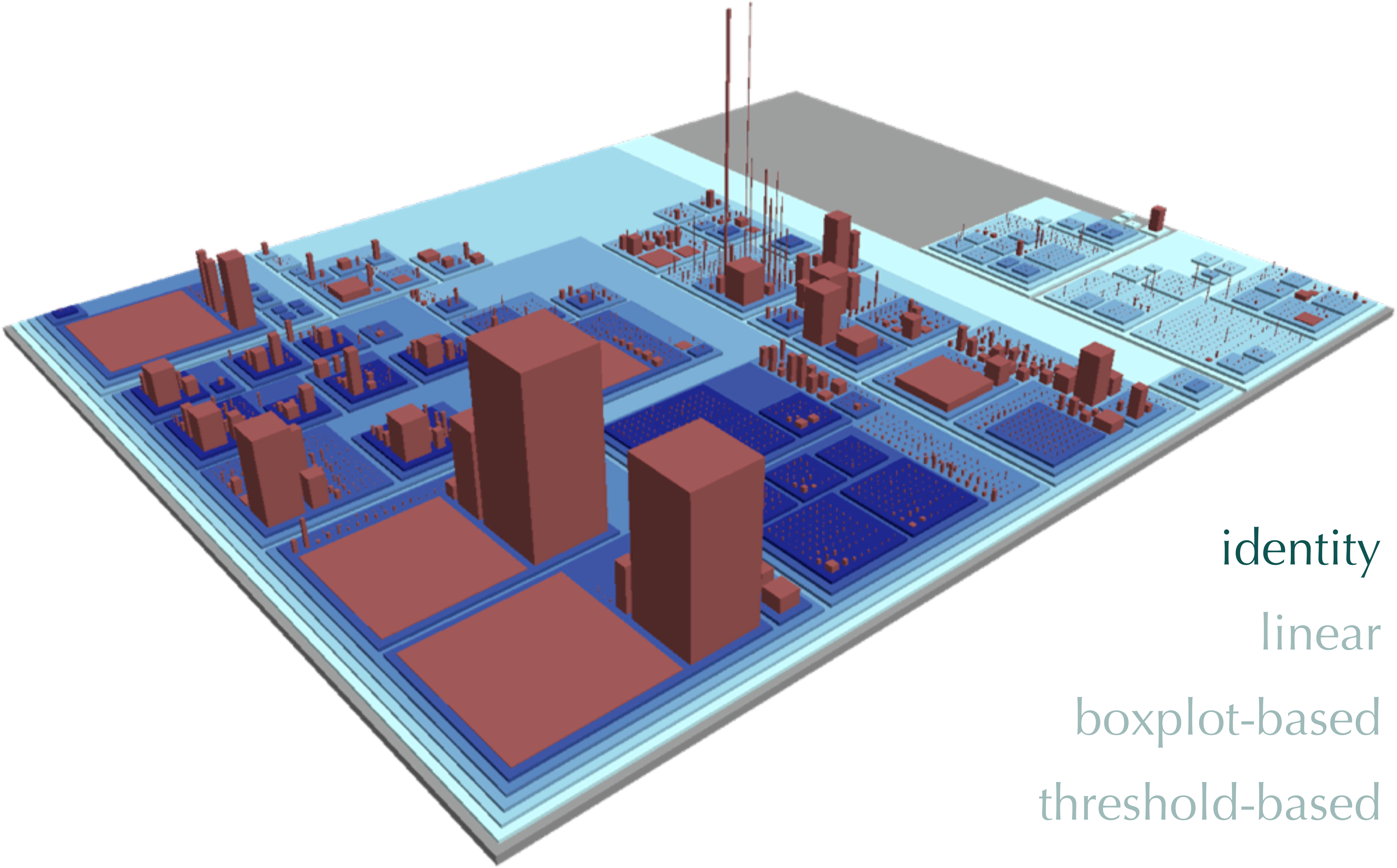
identity

linear

boxplot-based

threshold-based

Mapping Metrics



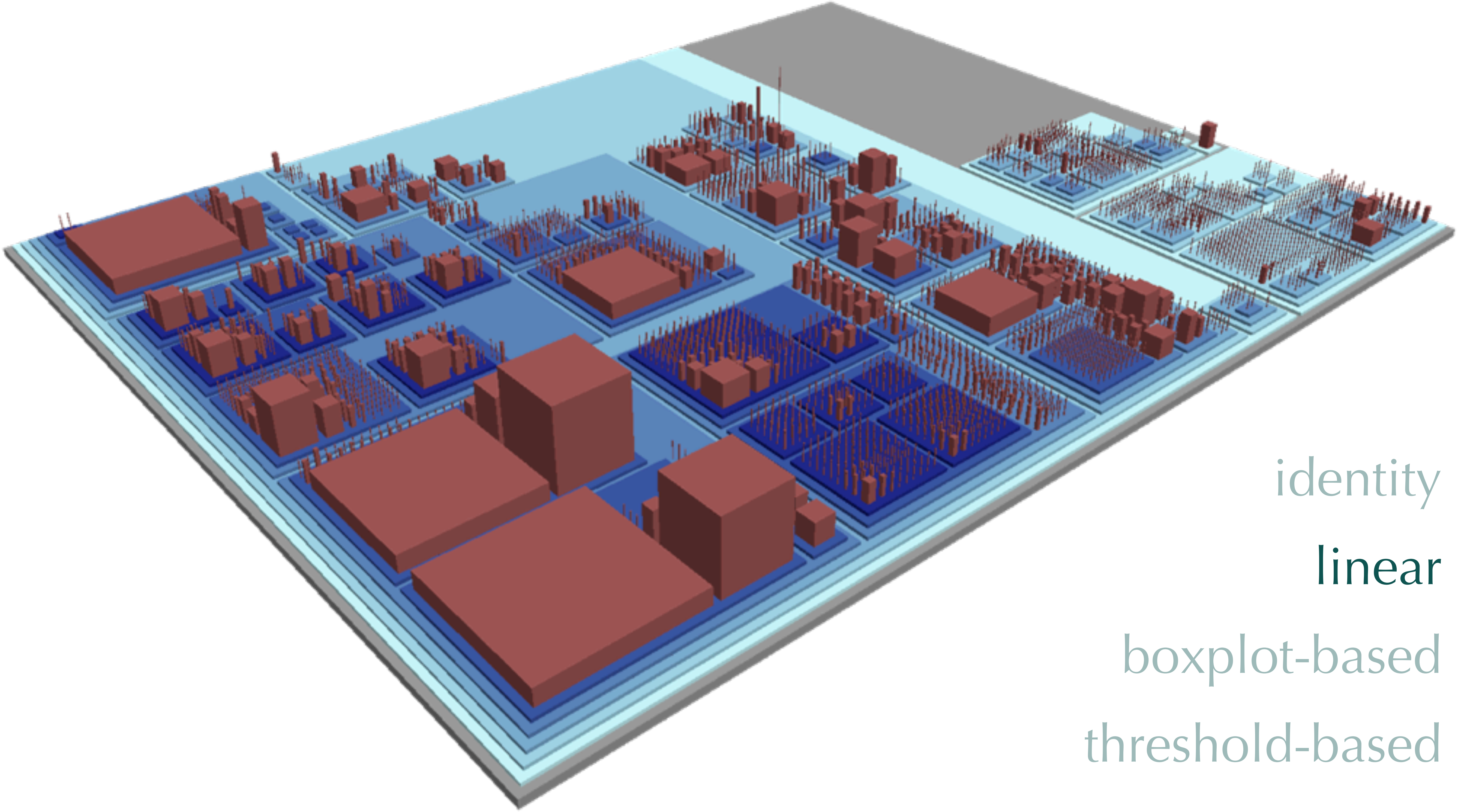
identity

linear

boxplot-based

threshold-based

Mapping Metrics



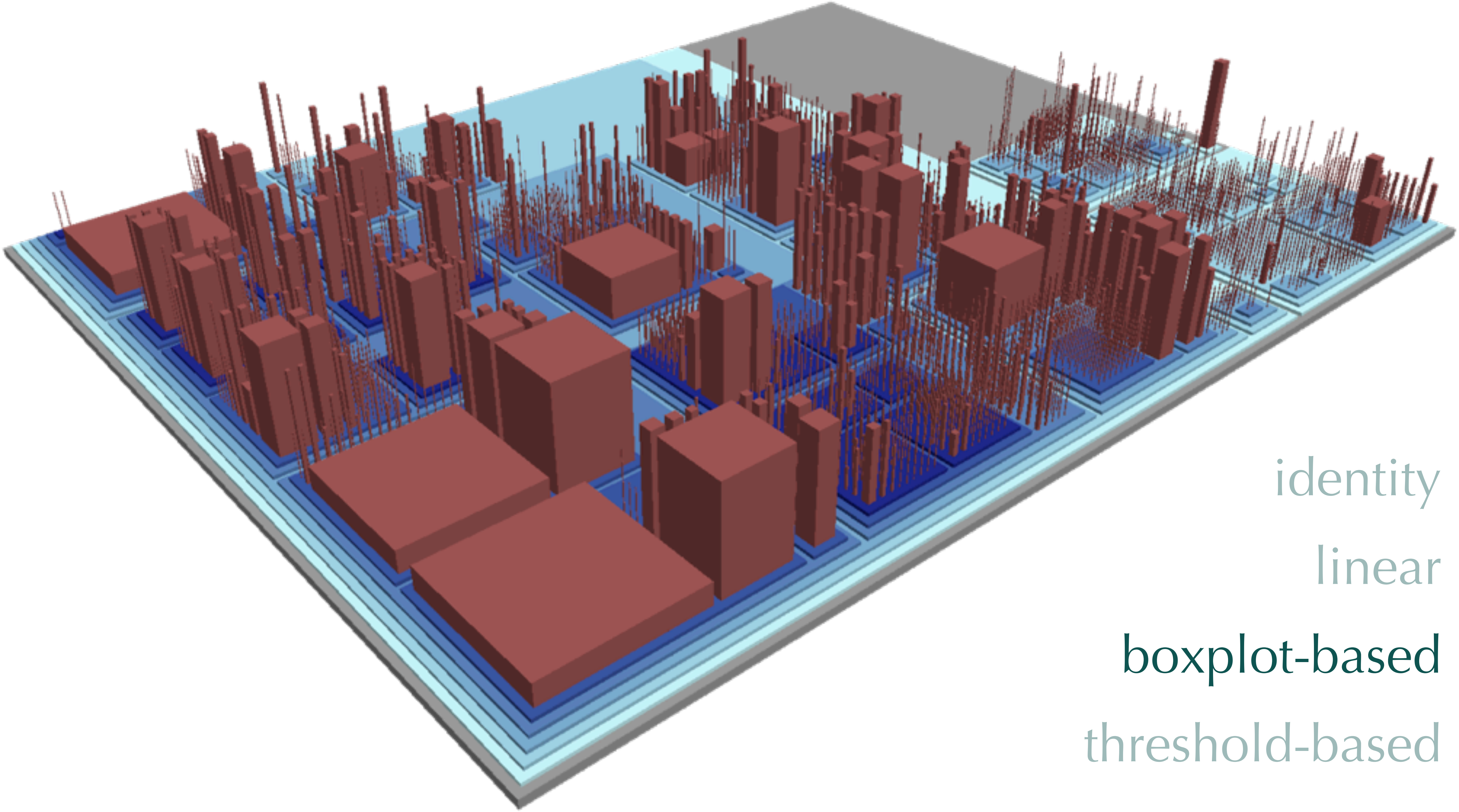
identity

linear

boxplot-based

threshold-based

Mapping Metrics



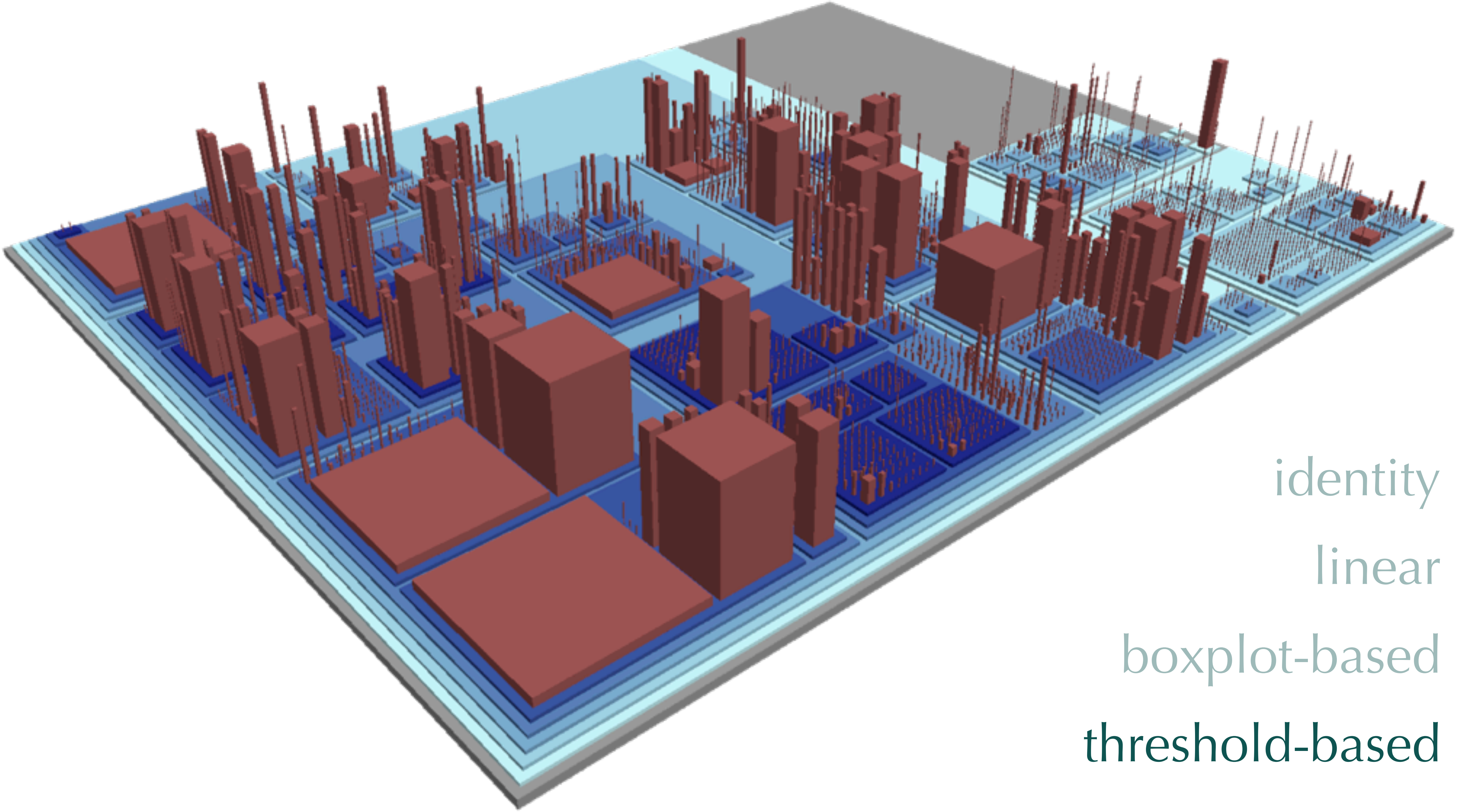
identity

linear

boxplot-based

threshold-based

Mapping Metrics



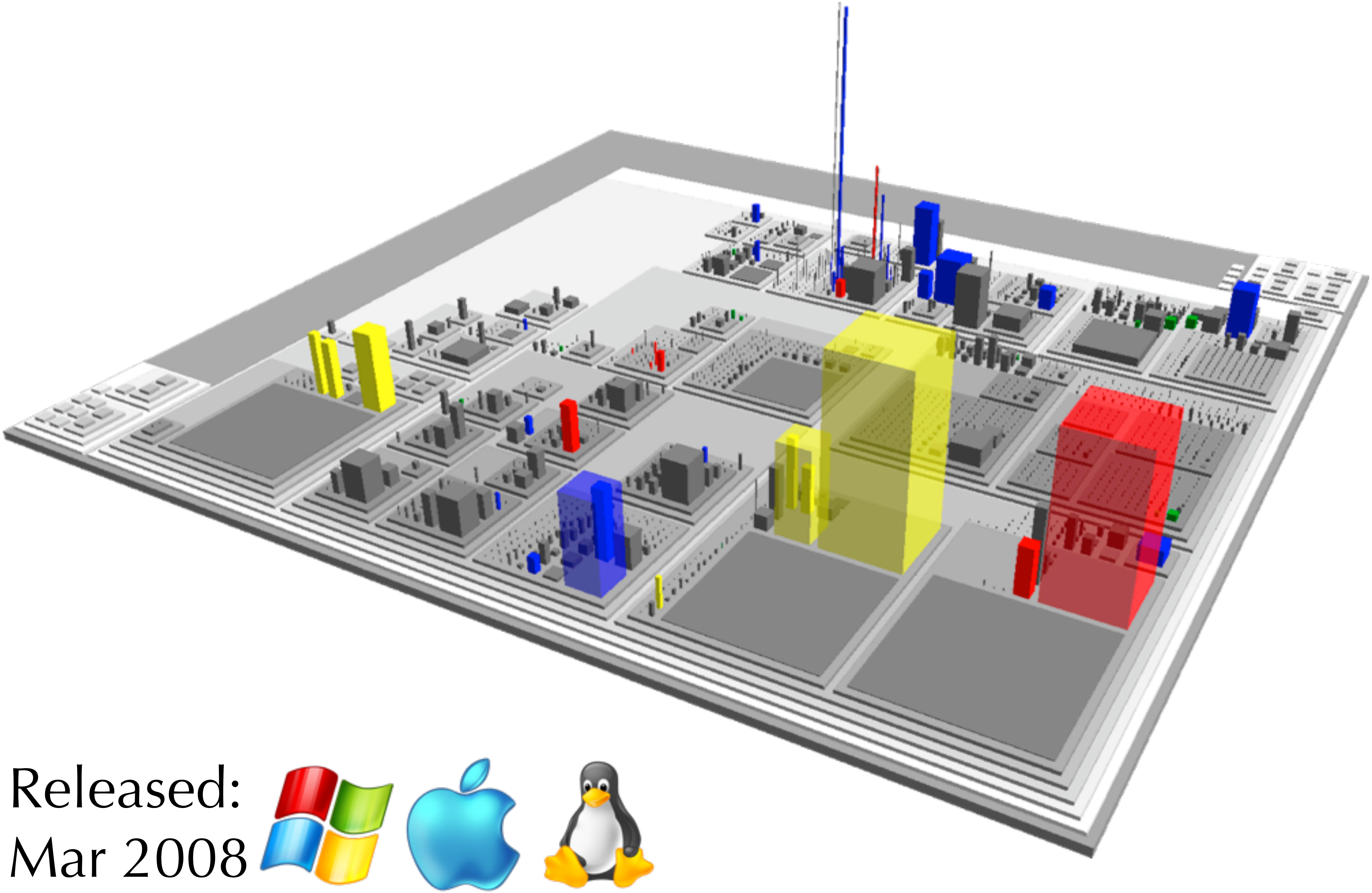
identity

linear

boxplot-based

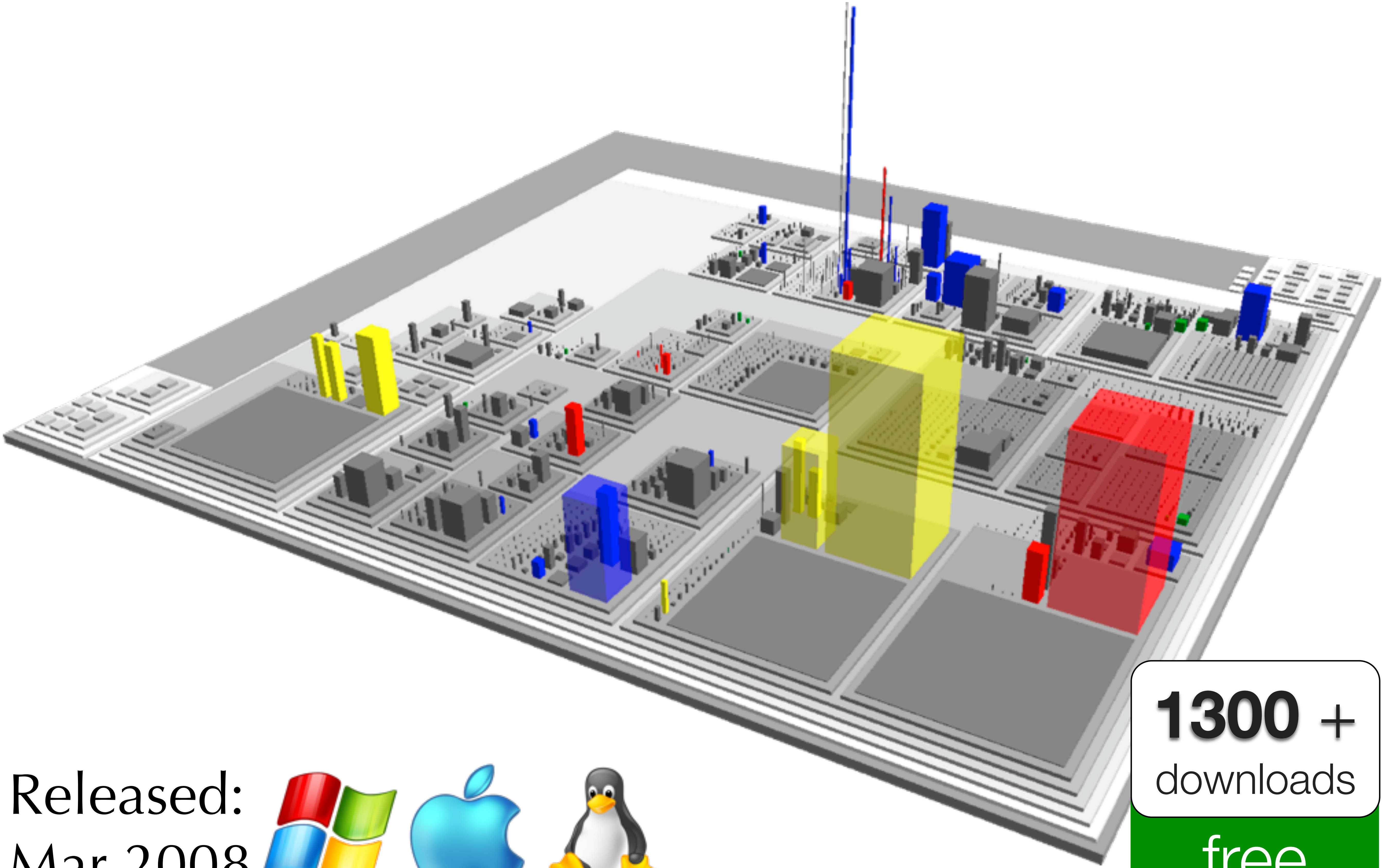
threshold-based

http://www.inf.unisi.ch/phd/wettel/codecity.html



Released:
Mar 2008

http://www.inf.unisi.ch/phd/wettel/codecity.html



Released:
Mar 2008

1300 +
downloads

free