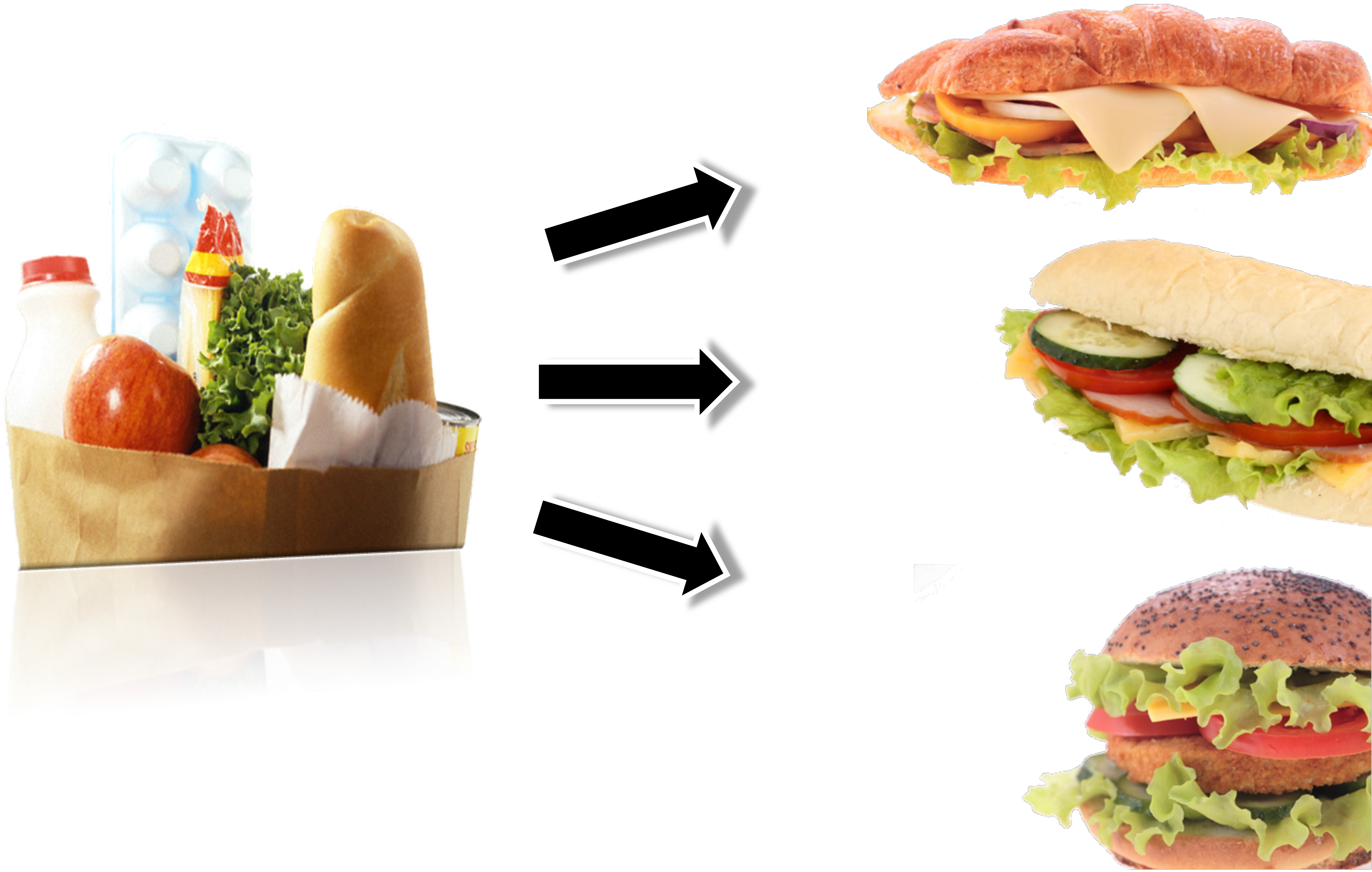
Granularity and Code Cloning in Software Product Lines

Sandro Schulze

Co-work with:
Sven Apel
Christian Kästner
Jörg Liebig
Janet Feigenspan

UNIVERSITÄT PASSAU
Philipps Universität Marburg
OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG INF



Who are your parents?

Where is the code, belonging to a feature?

Who are the guys you hang out with?

How are features related to each other?

What does you education cost?

How expensive is it to change features?

Conditional Compilation

```
static int _rep_queue_filedone(...)
        DB_ENV *dbenv;
        REP *rep;
        __rep_fileinfo_args *rfp; {
#ifndef HAVE_QUEUE
        COMPQUIET(rep, NULL);
        COMPQUIET(rfp, NULL);
        return (__db_no_queue_am(dbenv));
#else
        db_pgno_t first, last;
        u_int32_t flags;
        int empty, ret, t_ret;
#ifdef DIAGNOSTIC
        DB_MSGBUF mb;
#endif
                                ...
```

Excerpt from Oracle's Berkeley DB

Objections / Criticism

Designed in the 70th and hardly evolved since

“#ifdef considered harmful”

“#ifdef hell”

“maintenance becomes a ‘hit or miss’ process”

“is difficult to determine if the code being viewed is actually compiled into the system”

“incomprehensible source texts”

“programming errors are easy to make and difficult to detect”

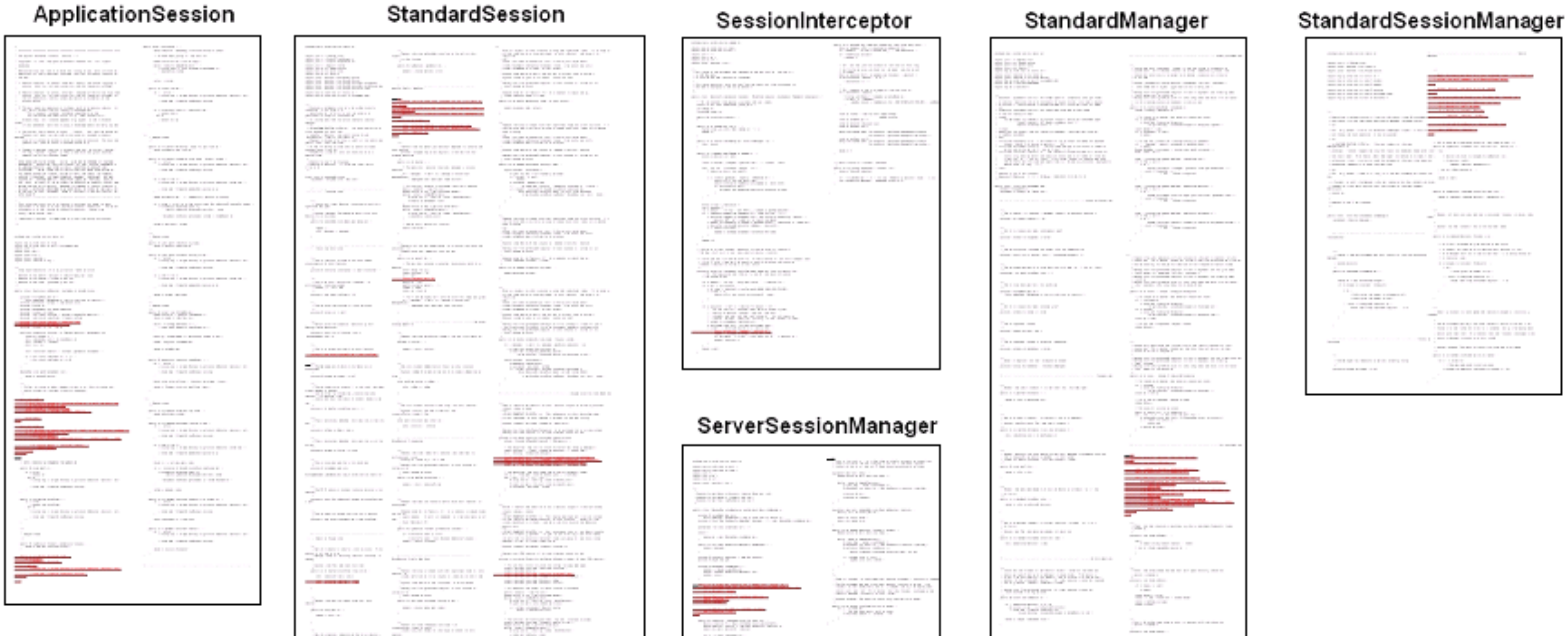
“CPP makes maintenance difficult”

“source code rapidly becomes a maze”

“preprocessor diagnostics are poor”

A CLOSER LOOK AT PREPROCESSORS

Separation of Concerns



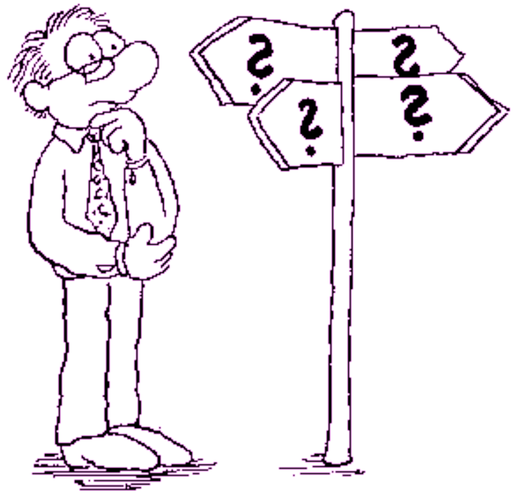
2000 features

10000

features

```
find <ordner> -type f \( -name "*.h" -o -name "*.c" \) -exec
egrep '^\s*#if' '{}' \; | sed -e 's/#ifndef//' | sed -e 's/
#ifdef//' | sed -e 's/#if//' | sort | uniq | wc -l
```

```
int put_eol(fd)
    FILE *fd;
{
  if (
#ifdef USE_CRNL
    (
#ifdef MKSESSION_NL
      !mksession_nl &&
#endif
      (putc('\r', fc) < 0)) ||
#endif
      (putc('\n', fd) < 0))
    return FAIL;
  return OK;
}
```



How is the preprocessor used ? (Liebig et al., ICSE '10)

23% of the code is variable

Variable code mostly heterogeneous (89%)

At which granularity ? (Liebig et al., AOSD '11)

Variability mostly at coarse granularity (84%)

What about the remaining 16% ?

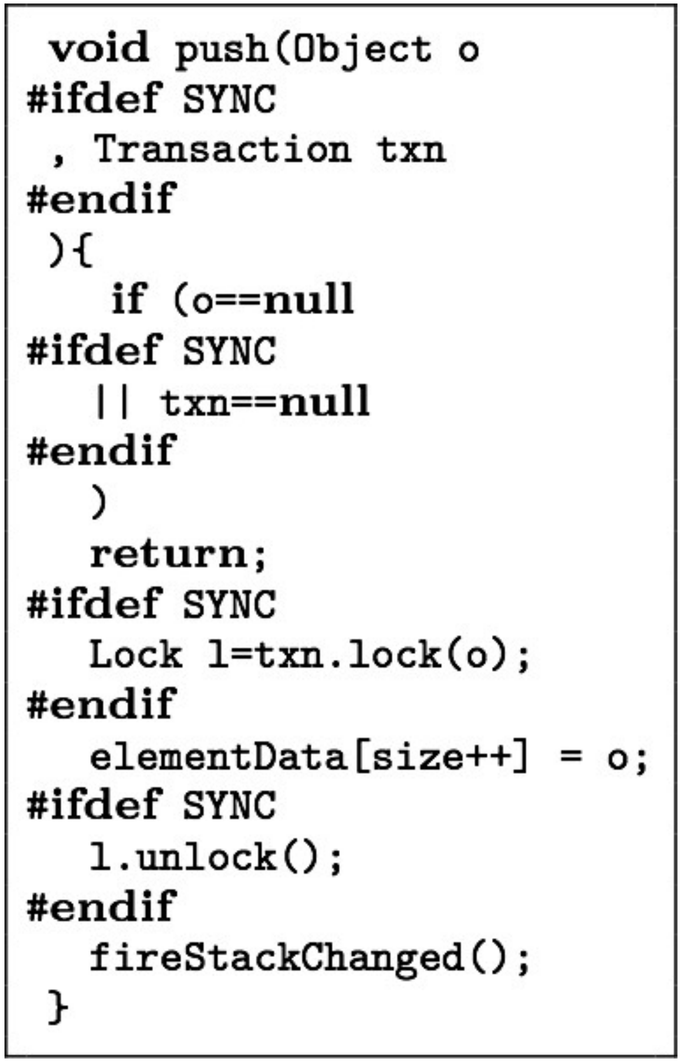
```
typedef struct {
  typebuf_T save_typebuf;
  int typebuf_valid;
  struct buffheader save_stuffbuff;
#if USE_INPUT_BUF
  char_u *save_inputbuf;
#endif
} tasave_T;
```

```
  for ( ; mp != NULL;
#ifdef FEAT_LOCALMAP
    mp->m_next == NULL ?
    (mp = mp2, mp2 =
          NULL) :
#endif
    (mp = mp->m_next)) {
```

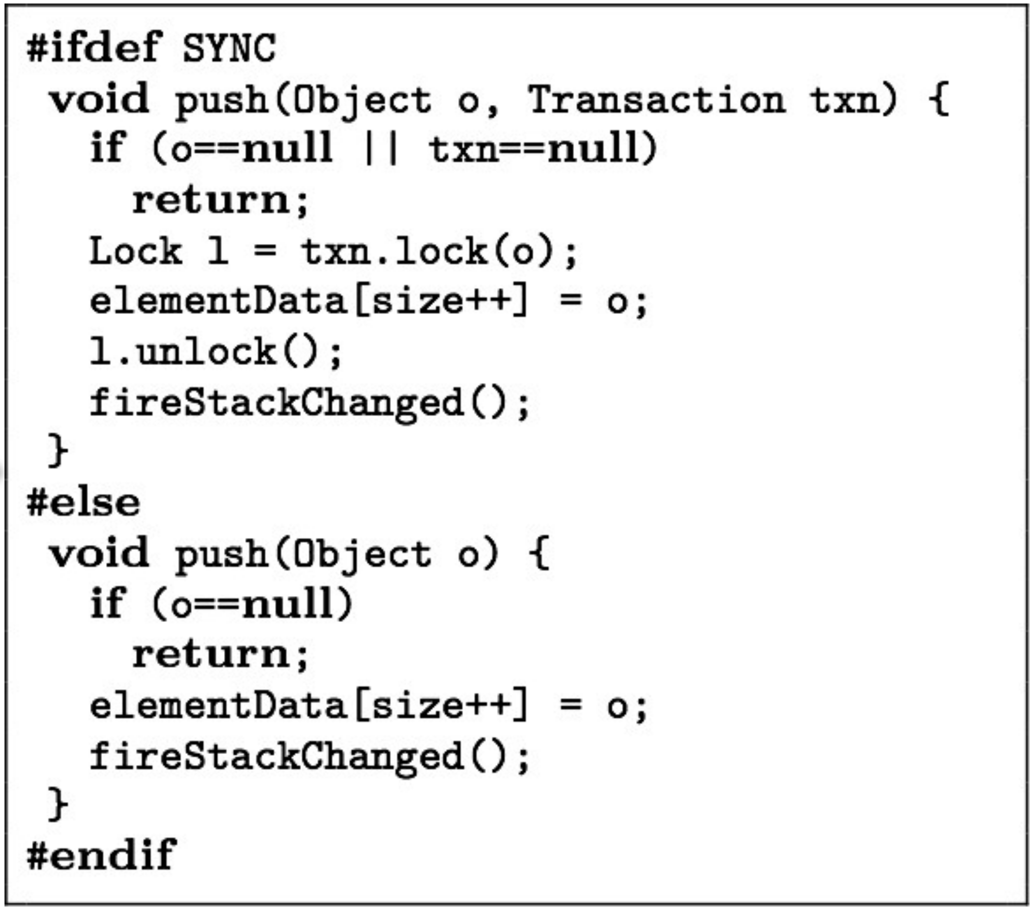
```
 void tcl_end() {
#ifdef DYNAMIC_TCL
  if (hTclLib) {
    FreeLibrary(hTclLib);
    hTclLib = NULL;
  }
#endif
}
```

```
#ifdef FEAT_CLIENTSERVER
  case SPEC_CLIENT:
    sprintf((char *)strbuf,
    PRINTF_HEX_LONG_U,
    (long_u)clientWindow);
  result = strbuf;
  break;
#endif
```

Transformation (Refactoring)



```
 void push(Object o
#ifdef SYNC
 , Transaction txn
#endif
 ){
    if (o==null
#ifdef SYNC
   || txn==null
#endif
   )
   return;
#ifdef SYNC
   Lock l=txn.lock(o);
#endif
   elementData[size++] = o;
#ifdef SYNC
   l.unlock();
#endif
   fireStackChanged();
 }
```

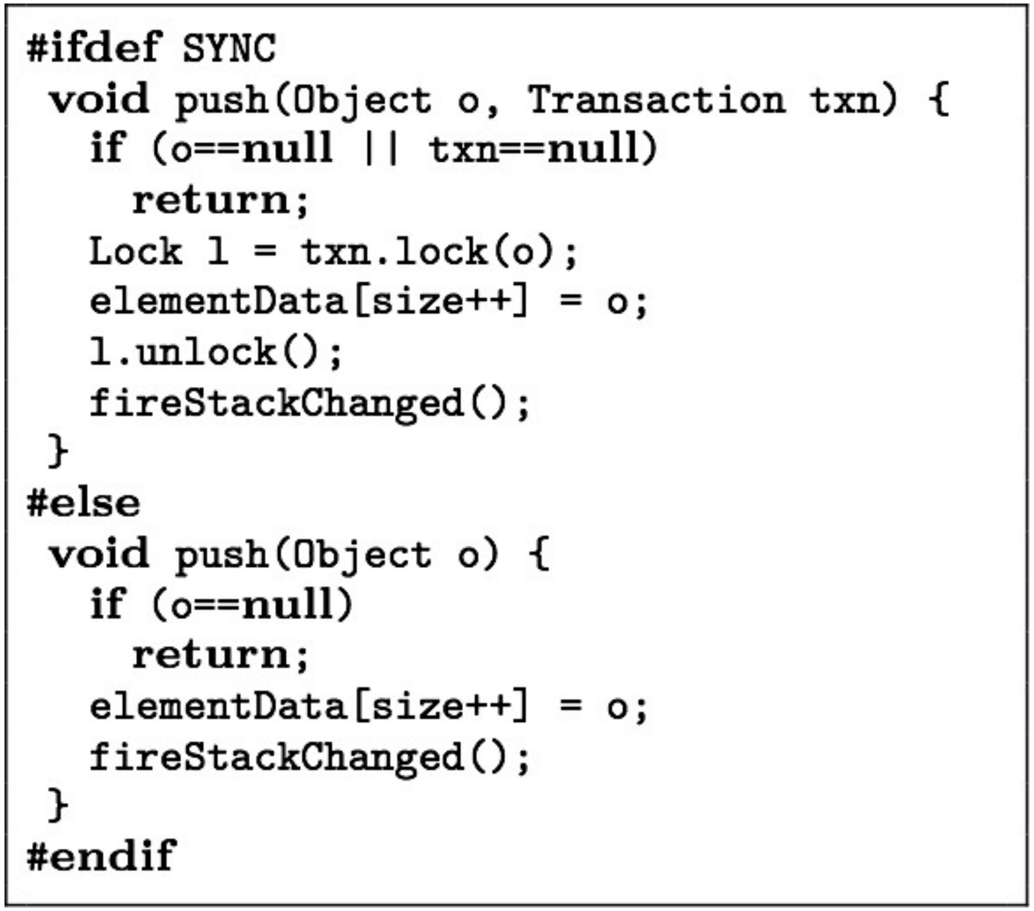


```
#ifdef SYNC
 void push(Object o, Transaction txn) {
   if (o==null || txn==null)
     return;
   Lock l = txn.lock(o);
   elementData[size++] = o;
   l.unlock();
   fireStackChanged();
 }
#else
 void push(Object o) {
   if (o==null)
     return;
   elementData[size++] = o;
   fireStackChanged();
 }
#endif
```

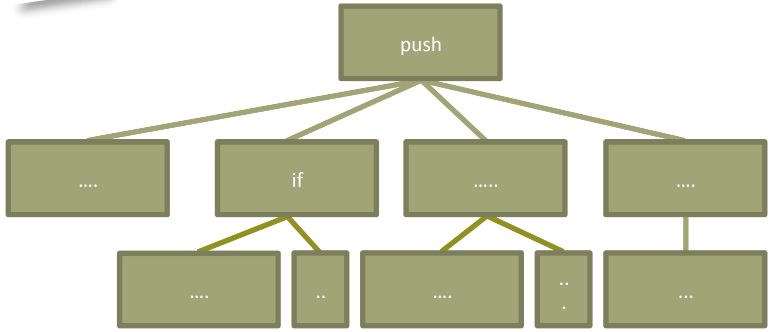
Coarse-grained annotations prone to clones (Schulze et al., SCAM '11)

Good or Bad?

What about Program Comprehension?



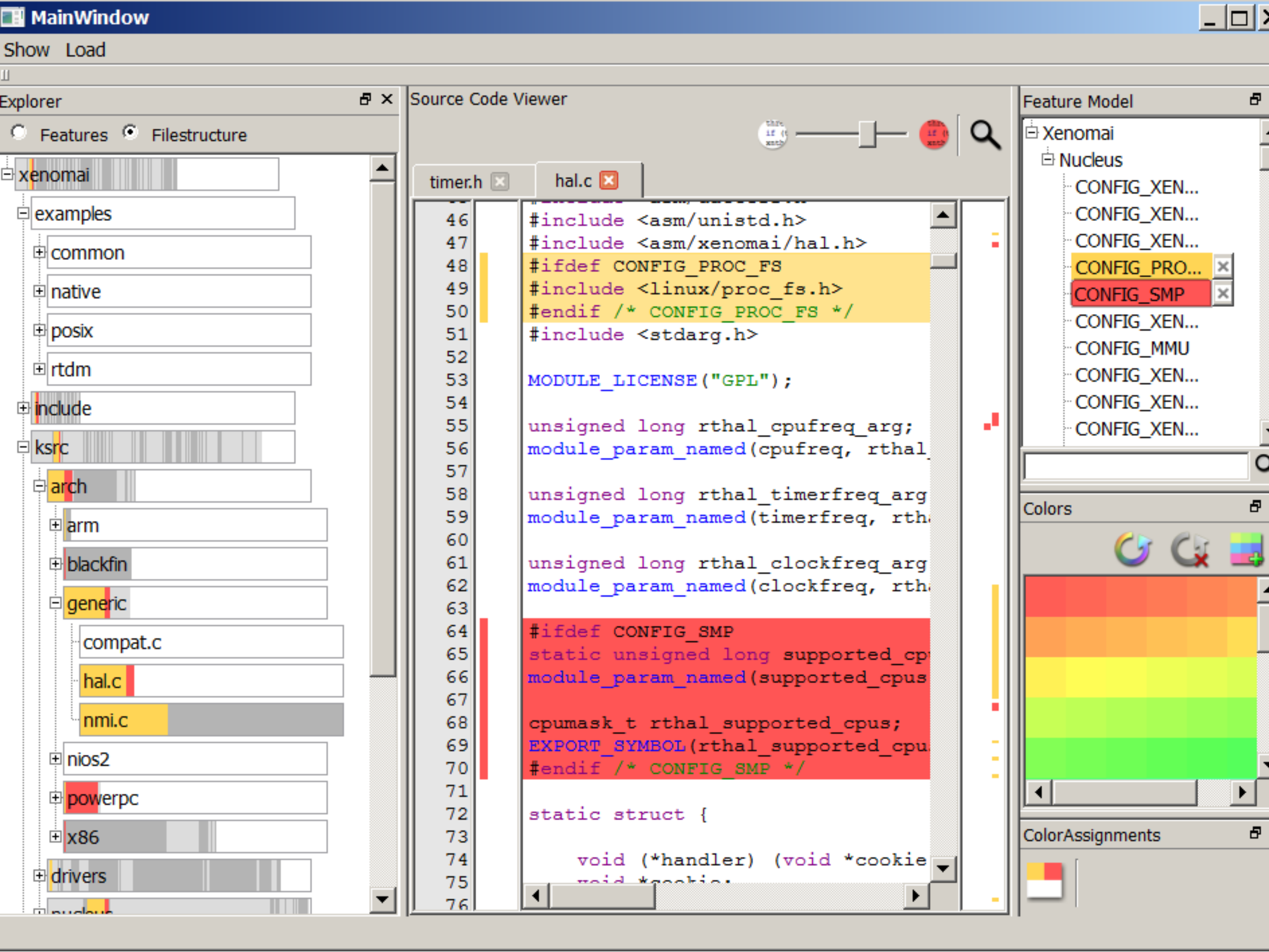
```
#ifdef SYNC
 void push(Object o, Transaction txn) {
   if (o==null || txn==null)
     return;
   Lock l = txn.lock(o);
   elementData[size++] = o;
   l.unlock();
   fireStackChanged();
 }
#else
 void push(Object o) {
   if (o==null)
     return;
   elementData[size++] = o;
   fireStackChanged();
 }
#endif
```



Experiment on Program Comprehension of preprocessor annotations

Coarse-grained vs. fine-grained

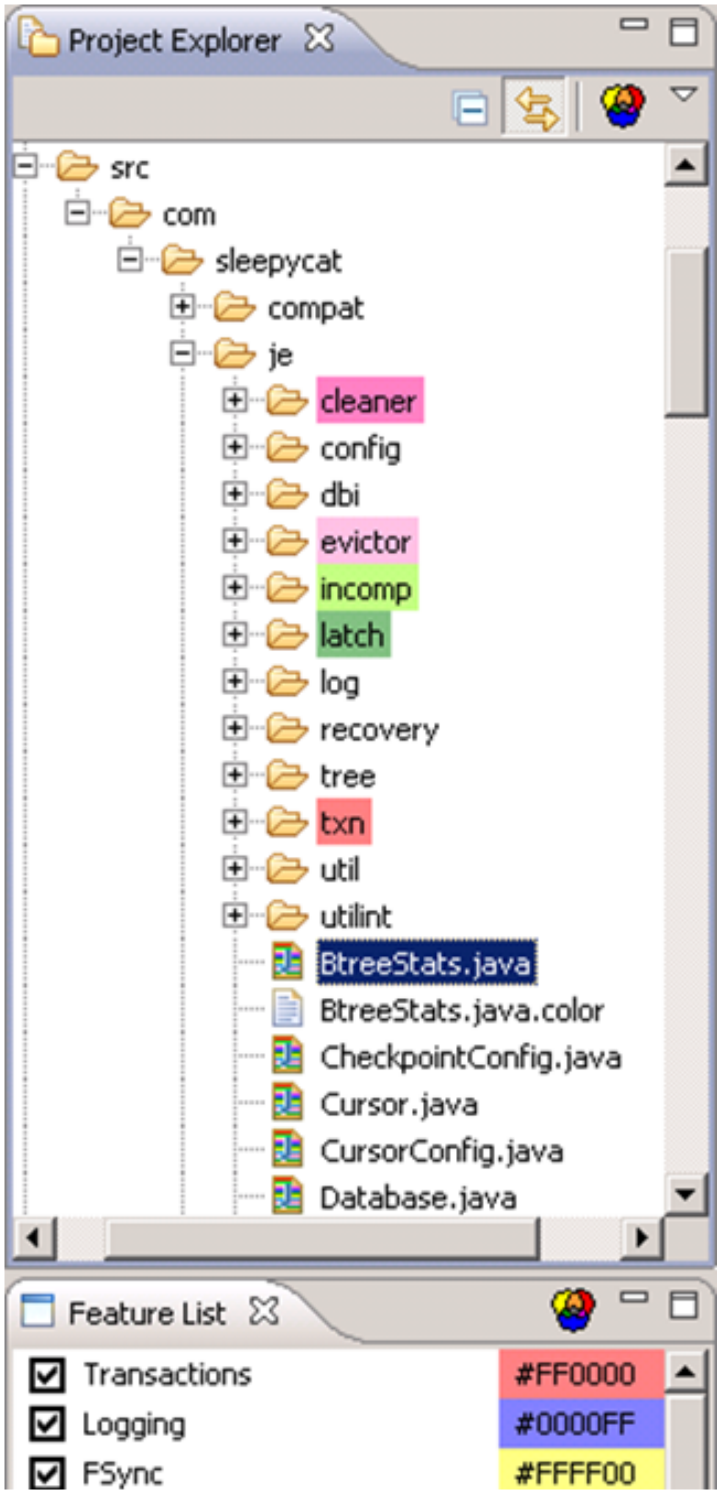
e.g., Remove all code that belongs to Feature IP_V6 !



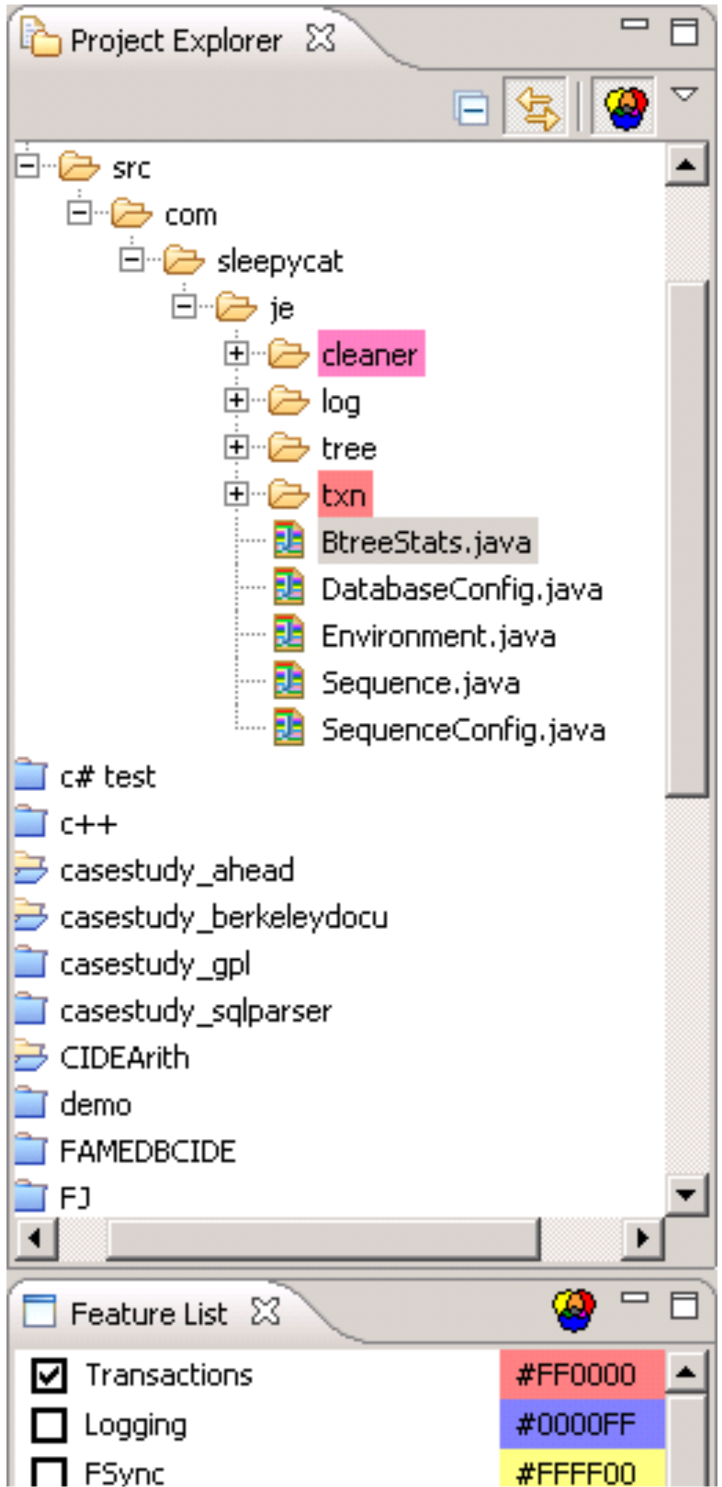
```
#include <asm/unistd.h>
#include <asm/xenomai/hal.h>
#ifdef CONFIG_PROC_FS
#include <linux/proc_fs.h>
#endif /* CONFIG_PROC_FS */
#include <stdarg.h>

MODULE_LICENSE("GPL");

unsigned long rthal_cpufreq_arg;
module_param_named(cpufreq, rthal_

unsigned long rthal_timerfreq_arg
module_param_named(timerfreq, rth

unsigned long rthal_clockfreq_arg
module_param_named(clockfreq, rth

#ifdef CONFIG_SMP
static unsigned long supported_cp
module_param_named(supported_cpus

cpumask_t rthal_supported_cpus;
EXPORT_SYMBOL(rthal_supported_cpu
#endif /* CONFIG_SMP */

static struct {

    void (*handler) (void *cookie
```

[Kästner et al., ICSE '08]

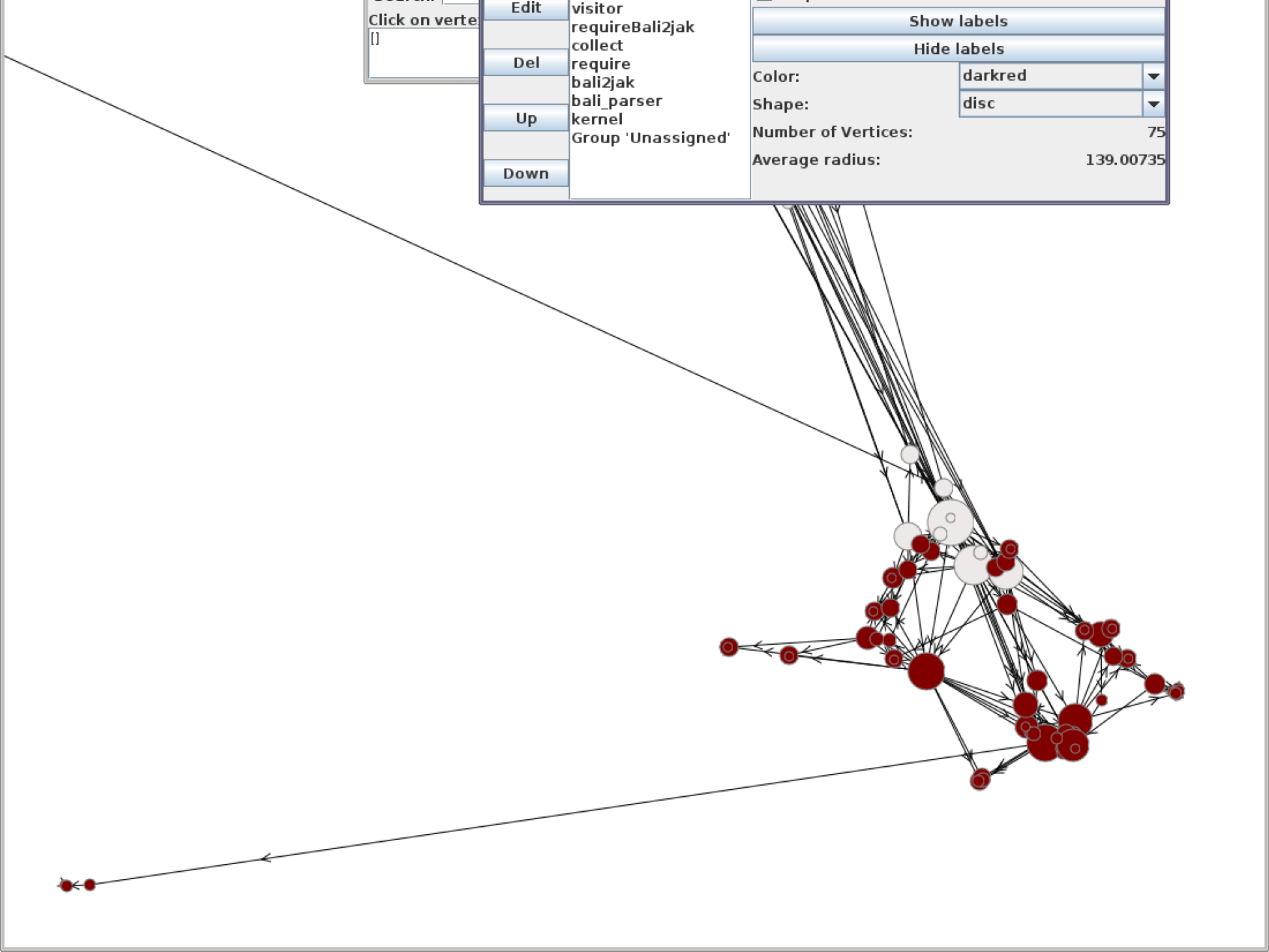
Views



✓	Transactions	#FF0000
✓	Logging	#0000FF
✓	FSync	#FFFF00



✓	Transactions	#FF0000
☐	Logging	#0000FF
☐	FSync	#FFFF00


Click on verte

[]

Edit

Del

Up

Down

visitor
requireBali2jak
collect
require
bali2jak
bali_parser
kernel
Group 'Unassigned'

Show labels

Hide labels

Color: darkred

Shape: disc

Number of Vertices: 75

Average radius: 139.00735

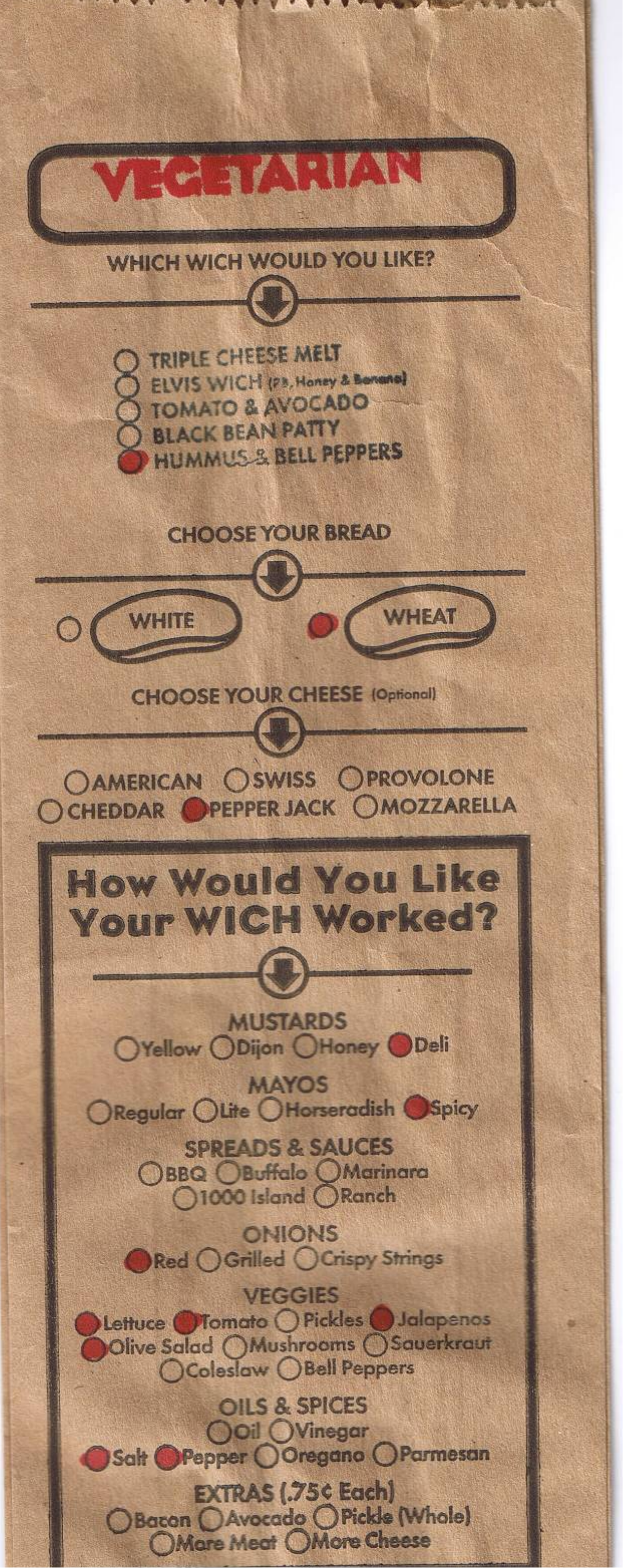
- ✓ improved Readability
- ✓ reduced programmer effort
- ✓ compositionality

- ✗ lower expressiveness
- ✗ bloated code → clones

- ✓ high expresiveness
- ✓ language-independent
- ✓ easy to use

- ✗ tangled/scattered code
- ✗ obfuscation → hard to understand/modify

Feature-Oriented Product Lines



VEGETARIAN

WHICH WICH WOULD YOU LIKE?

- TRIPLE CHEESE MELT
- ELVIS WICH (PB, Honey & Banana)
- TOMATO & AVOCADO
- BLACK BEAN PATTY
- [✓ HUMMUS & BELL PEPPERS]

CHOOSE YOUR BREAD

WHITE [✓ WHEAT]

CHOOSE YOUR CHEESE (Optional)

AMERICAN SWISS PROVOLONE
CHEDDAR [✓ PEPPER JACK] MOZZARELLA

How Would You Like Your WICH Worked?

MUSTARDS
Yellow Dijon Honey [✓ Deli]

MAYOS
Regular Lite Horseradish [✓ Spicy]

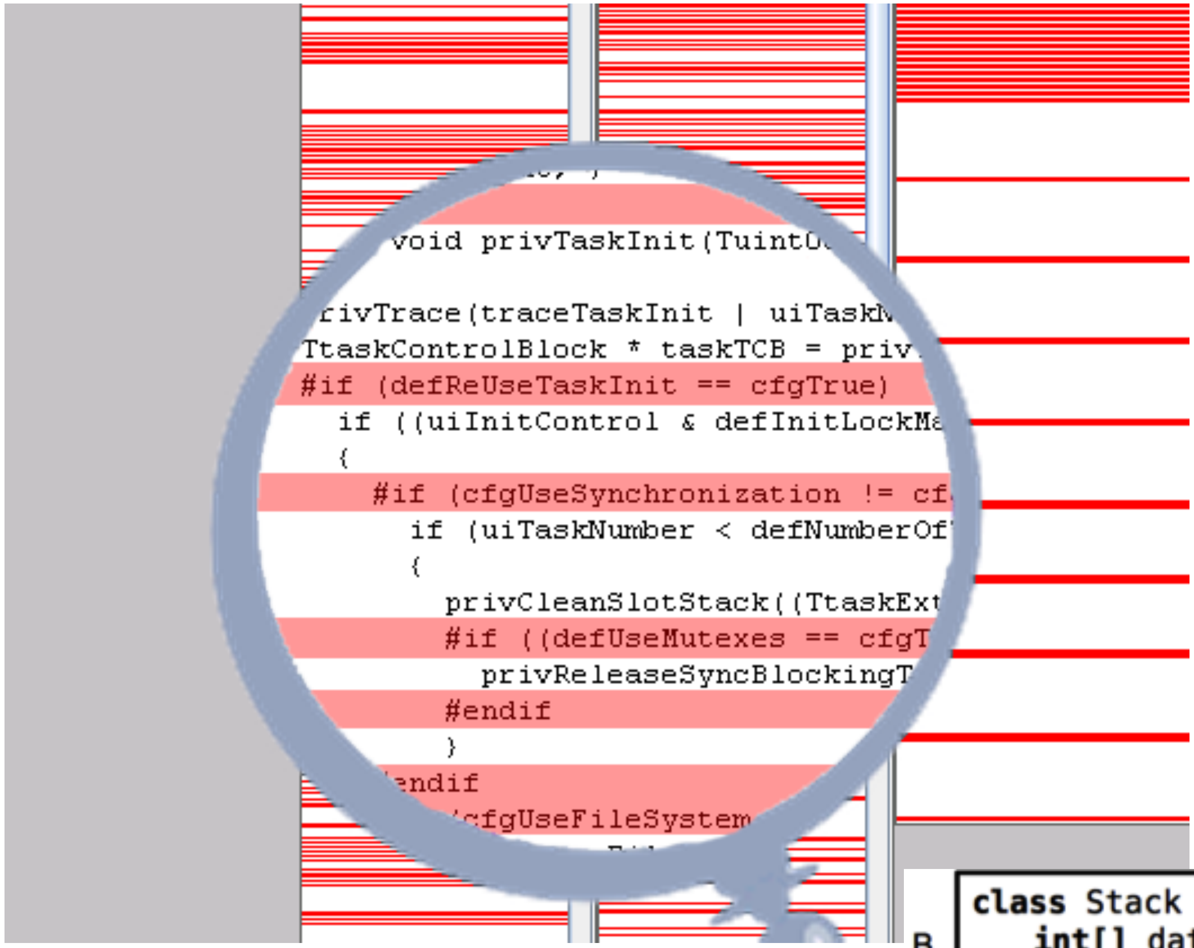
SPREADS & SAUCES
BBQ Buffalo Marinara
1000 Island Ranch

ONIONS
[✓ Red] Grilled Crispy Strings

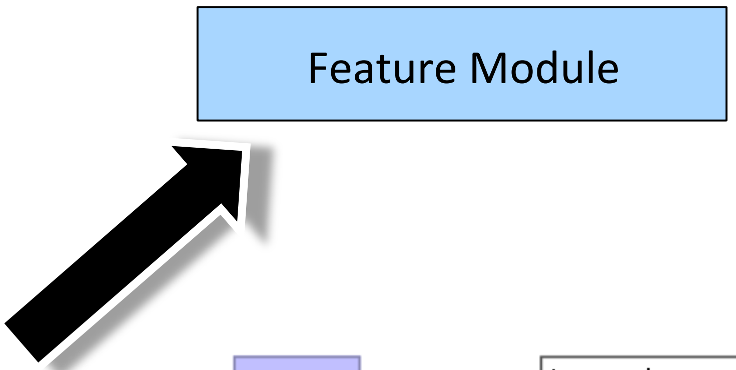
VEGGIES
[✓ Lettuce] [✓ Tomato] Pickles [✓ Jalapenos]
[✓ Olive Salad] Mushrooms Sauerkraut
Coleslaw Bell Peppers

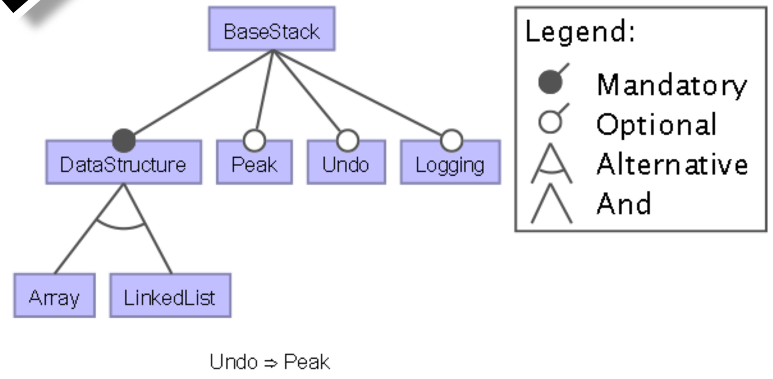
OILS & SPICES
Oil Vinegar
[✓ Salt] [✓ Pepper] Oregano Parmesan

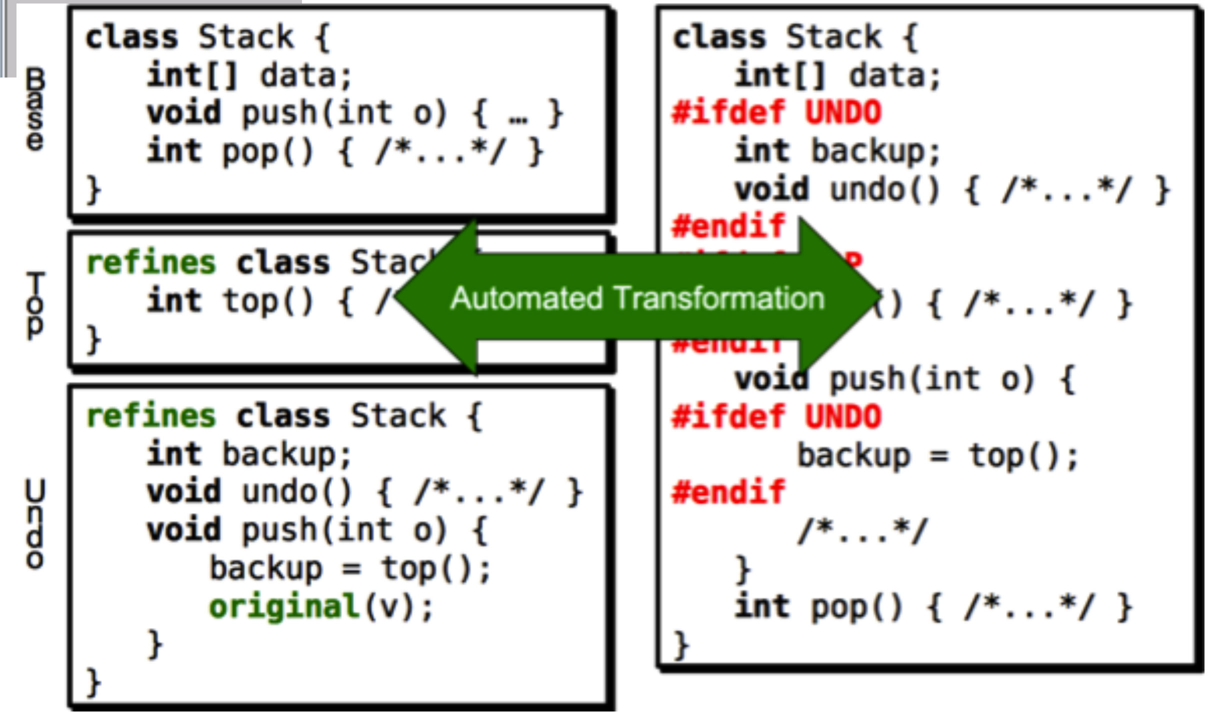
EXTRAS (.75¢ Each)
Bacon Avocado Pickle (Whole)
More Meat More Cheese



```
void privTaskInit(TuintO
riv Trace(traceTaskInit | uiTaskN
TtaskControlBlock * taskTCB = priv
#if (defReUseTaskInit == cfgTrue)
  if ((uiInitControl & defInitLockMa
  {
    #if (cfgUseSynchronization != cf
      if (uiTaskNumber < defNumberOf
      {
        privCleanSlotStack((TtaskExt
        #if ((defUseMutexes == cfgT
          privReleaseSyncBlockingT
        #endif
        }
  endif
    cfgUseFileSystem
```







```
class Stack {
    int[] data;
    void push(int o) { … }
    int pop() { /*...*/ }
}
```

```
refines class Stac
    int top() { /
}
```

```
refines class Stack {
    int backup;
    void undo() { /*...*/ }
    void push(int o) {
        backup = top();
        original(v);
    }
}
```

```
class Stack {
    int[] data;
#ifdef UNDO
    int backup;
    void undo() { /*...*/ }
#endif
#
                ) { /*...*/ }
#
    void push(int o) {
#ifdef UNDO
        backup = top();
#endif
        /*...*/
    }
    int pop() { /*...*/ }
}
```

✓ ✗

Where is the code, belonging to a feature ?

Hard to determine with preprocessors; modularity helps

How are features related to each other ?

Alternative representations required, e.g., visualization

How expensive is it to change features ?

Depends on granularity and separation of concerns

What's next...

...from a Provenance Perspective

Identifying featues in legacy appications → variability mining

Reverse Engineering Software Product Lines

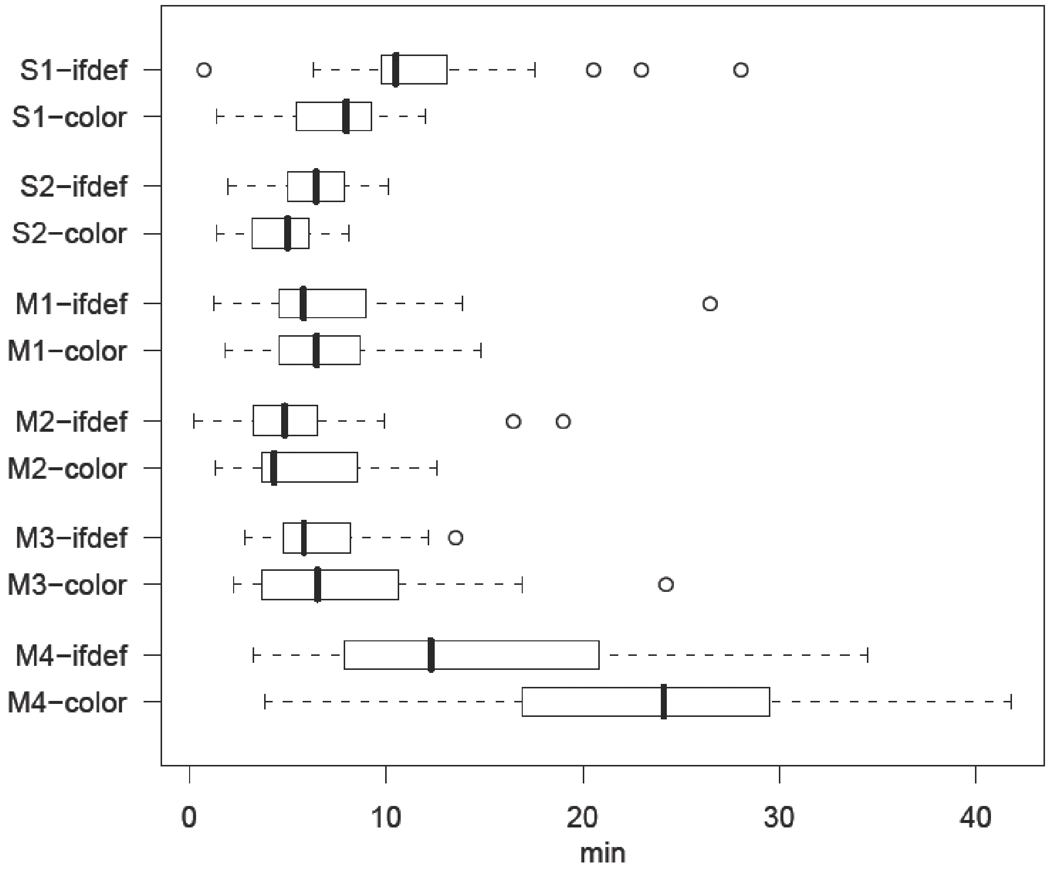
→ commonalities and variabilities

Evolution of Software Product Lines

Questions

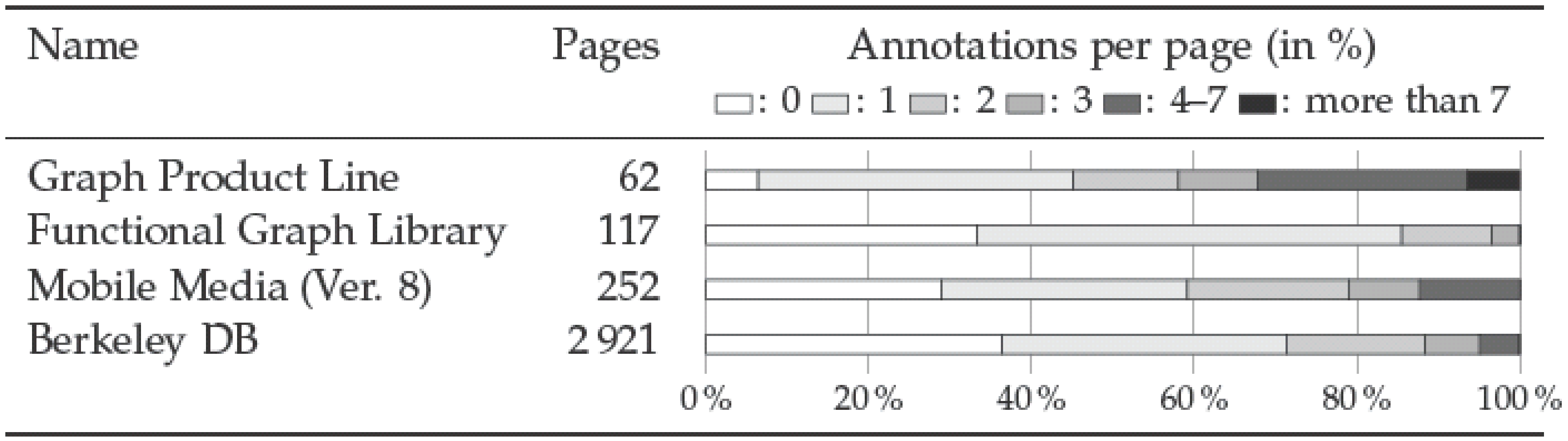
Program Comprehension: An Experiment

- #ifdef vs. colors
- 43 subjects in 2 groups
- S1-2: search tasks faster with colors (43% & 23%)
- M1-3: maintenance tasks same perform.
- M4: maintenance task with red back-ground color -37%
- No influence on correctness
- Subjects prefer colors



Scaling Visual Representations

- Focus on few features at a time
- Repeating colors / manual assignment sufficient
- Analysis of 4 Java ME and 40 C programs:
  - 96 % pages of source code with ≤ 3 colors
  - 99 % pages of source code with ≤ 7 colors



Name	Pages	Annotations per page (in %)
Graph Product Line	62	
Functional Graph Library	117	
Mobile Media (Ver. 8)	252	
Berkeley DB	2 921	

Error-Prone

```
static int _rep_queue_filedone(...)
        DB_ENV *dbenv;
        REP *rep;
        __rep_fileinfo_args *rfp; {
#ifndef HAVE_QUEUE
        COMPQUIET(rep, NULL);
        COMPQUIET(rfp, NULL);
        return (__db_no_queue_am(dbenv));
#else
        db_pgno_t first, last;
        u_int32_t flags
        int empty, ret, t
#ifdef DIAGNOSTIC
        DB_MSGBUF
#endif
        // over 100 line
    }
#endif
```

```
#ifdef TABLES
class Table {
  void insert(Object data, Txn txn) {
      storage.set(data, txn.getLock());
  }
}
#endif
class Storage {
#ifdef WRITE
   boolean set(…) { ... }
#endif
}
```