Software Product Lines

15-214
Charlie Garrod, Christian Kästner
Homework 0: A Friendship Graph

The goal of this homework is to get familiar with the infrastructure used in this course and to attempt some first steps in Java programming. To simplify the programming task, we have selected a simple task from a familiar domain.

**Task:** We want to represent the friendship graph. We want to calculate the shortest connection between two friends. Test your implementation with the following graph:

- Philip
- Tom
- Marv
- Jessie
- Paul
- Emily

We expect a client implementation roughly:

```
FriendGraph g = new FriendGraph();
Person anne = new Person("Anne");
```

---

Homework #1: Graph Algorithms at Facecook

Due Thursday, January 31st at 11:59 p.m.

After graduating from CMU you have been hired by Facecook, an up-and-coming social network company, to develop analytical algorithms for their website. The Facecook design committee has developed a `Graph` interface that will be used to store social network information for Facecook users. You have decided to experiment with two different `Graph` implementations to find which representation is best for different purposes. Be careful! To ensure that your implementation works with existing Facecook code, you must not alter the `Graph` interface in any way!

The goals for this assignment are:

- Understand and apply the concepts of polymorphism and encapsulation
- Learn to use Javadoc
- Better understand graphs and their multiple representations
- Familiarize yourself with Java and Eclipse
public class Backend {

    private Map<String, List<JETIListener>> eventListeners;
    private Map<JID, JETIListener> presenceListeners;
    private Map<JID, JETIListener> messageListeners;
    private int identifier;
    private Connect connect;
    private Roster roster;
    private Server server;
    private Jeti main;
    private JFrame mainFrame;
    private Container container;

    private Start start;
    private IQTimerQueue iqTimerQueue;
    private Handlers handlers;

    //private Capabilities capabilities;

    public Backend(Start start) {
        eventListeners = start.eventListeners;
        iqTimerQueue = start.iqTimerQueue;
    }
}
Reuse and Variation
Configuration in Software

• Systems cover all possible functionality
  – e.g., Windows, Open Office, Oracle, SAP myERP, Photoshop

• Specialized software and software for embedded systems increasingly important
  – Mobile devices, sensor networks, automotive systems, consumer electronics, smart cards, ubiquitous computing
  – 98% of all CPUs in embedded devices [2000]
  – Resource constraints, heterogeneous hardware
Database Systems

- Increasing data volumes
- Resource constraints in embedded environments
Printer Firmware
Linux Kernel
Linux kernel

- ~6 000 000 Lines of C code
- Highly configurable
  - > 10,000 configuration options! (x86, 64bit, …)
- Most source code is “optional”
15

Linux Kernel v2.6.33.3 Configuration

Processor type and features

Arrow keys navigate the menu. <Enter> selects submenus --->. Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc>Esc to exit, <?> for Help, </> for Search. Legend: [*] built-in [ ] excluded <M> module < > module capable

[ ] Tickless System (Dynamic Ticks)
[ ] High Resolution Timer Support
[ ] Symmetric multi-processing support
[ ] Support for extended (non-PC) x86 platforms
[ ] Single-depth WCHAN output
[ ] Paravirtualized guest support --->
[ ] Memtest
  Processor family (Generic-x86-64) --->
  Preemption Model (No Forced Preemption (Server)) --->
[ ] Reroute for broken boot IRqs (NEW)
[ ] Machine Check / overheating reporting
[ ] Dell laptop support
[ ] /dev/cpu/microcode - microcode support
[ ] /dev/cpu/*/msr - Model-specific register support
[ ] /dev/cpu/*/cpuid - CPU information support
  Memory model (Sparse Memory) --->
[*] Sparse Memory virtual memmap (NEW)
[ ] Allow for memory hot-add (NEW)
[ ] Enable KSM for page merging
  (4096) Low address space to protect from user allocation
[ ] Check for low memory corruption
[ ] Reserve low 64K of RAM on AMI/Phoenix BIOSen
-[*- MTRR (Memory Type Range Register) support
  MTRR cleanup support
[ ] Enable seccomp to safely compute untrusted bytecode
[ ] Enable -fstack-protector buffer overflow detection (EXPERIMENTAL)
  Timer frequency (250 HZ) --->
[ ] kexec system call

v(+)

<Select> < Exit > < Help >
2013 XTS STANDARD FWD  $44,995†

Standard Equipment

SELECT DRIVE TYPE
- Front Wheel Drive  MSRP Starting at $44,995**
- All Wheel Drive    MSRP Starting at $51,835**

SELECT TRIM
- Standard  MSRP Starting at $44,995**
- Luxury     MSRP Starting at $49,610**
- Premium    MSRP Starting at $54,505**
- Platinum   MSRP Starting at $59,080**

EXTERIOR:  Front  Back  Side  INTERIOR:  Driver  Side

Some selected options and accessories may not be shown on photos.
VEGETARIAN

WHICH WICH WOULD YOU LIKE?

- TRIPLE CHEESE MELT
- ELVIS WICH (w. 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
Software Product Lines in Industry

Boeing
Bosch Group
Cummins, Inc.
Ericsson
General Dynamics
General Motors
Hewlett Packard
Lockheed Martin
Lucent
NASA
Nokia
Philips
Siemens
...

Software Product Lines in Industry
CHALLENGES
a unique configuration for every person on this planet
320 features

more configurations than estimated atoms in the universe

optional, independent
199 Configuration parameters for the optimizer (1600 total)
Correctness?
Understanding?
Software Product Lines

“A software product line is a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.”

SEI
The Promise

Costs

Development without reuse

Development with reuse

# Products
CONQUERING VARIABILITY
Domain Analysis
(feature modeling)

Domain Implementation
(models, source code)

Requirements Analysis
(feature selection)

Application Derivation
(generator, testing)

Domain Engineering

Application Engineering

Customer needs

Domain knowledge

Product
keyless_entry → power_locks

MST  => Undirected ∧ Weighted
Cycle  => Directed
Linux Kernel v2.6.33.3 Configuration

- TestA
  - General setup
  - RCU Subsystem
  - Control Group support
  - Configure standard kernel features (for small systems)
  - Kernel Performance Events And Counters
  - GCOV-based kernel profiling
- Enable loadable module support
- Enable the block layer (NEW)
- IO Schedulers
- Processor type and features
  - Paravirtualized guest support
  - Power management and ACPI options
    - ACPI (Advanced Configuration and Power Interface) Support
    - SFI (Simple Firmware Interface) Support
  - Memory power savings
  - Bus options (PCI etc.)
    - PC Card (PCMCIA/CardBus) support
    - Support for PCI Hotplug
    - Executable file formats / Emulations
  - Networking support
    - Networking options
      - Network packet filtering framework (Netfilter)
      - Core Netfilter Configuration
      - IPv6: Netfilter Configuration
      - The DCCP Protocol (EXPERIMENTAL)
      - The SCTP Protocol (EXPERIMENTAL)
      - The TIPC Protocol (EXPERIMENTAL)
      - Distributed Switch Architecture support
      - QoS and/or fair queuing
      - Network testing
      - Amateur Radio support
      - CAN bus subsystem support

Option

- CPU Frequency scaling
  - CPU Frequency governor
    - powersave
    - userspace
      - userspace governor for userspace frequency scaling
      - 'ondemand' cpufreq policy governor
      - 'conservative' cpufreq governor
      - CPUFreq processor drivers
      - ACPI Processor P-States driver
        - AMD Opteron/Athlon64 PowerNow!
      - Intel Enhanced SpeedStep (deprecated)
      - Intel Perfium 4 clock modulation
        - shared options

'powersave' governor (CPU_FREQ_GOVERNORS)

CONFIG_CPU_FREQ_GOVERNORS:

This cpufreq governor sets the frequency statically to the lowest available CPU frequency.

To compile this driver as a module, choose M here: the module will be called cpufreq_powersave.

If in doubt, say Y.

---

SPL

OS

- Win
  - DirectX
  - Transactions
- Unix

API

- Get
- Put
- Delete

DirectX because set by user
Win because (DirectX) implies (Win)
not Unix because ((OS is (Unix or Win)) and stdmostne( Unix, Win))
Domain Engineering

Domain Analysis
(feature modeling)

Domain Implementation
(models, source code)

Requirements Analysis
(feature selection)

Application Derivation
(generator, testing)

Customer needs

Domain knowledge

Product
IMPLEMENTATION
Domain Analysis
(feature modeling)

Domain Implementation
(models, source code)

Requirements Analysis
(feature selection)

Application Derivation
(generator, testing)

Domain knowledge
Customer needs
Product
Runtime Parameters

Parameters, variables, constants
Parameter


[drive:][path][filename]
  Specifies drive, directory, and/or files to list.

/A
  Displays files with specified attributes.
  attributes
  D Directories
  H Hidden files
  S System files
  L Reparse Points

/B
  Uses bare format (no heading information or summary).

/C
  Display the thousand separator in file sizes. This is the default. Use /-C to disable display of separator.

/D
  Same as wide but files are list sorted by column.

/L
  Uses lowercase.

/N
  New long list format where filenames are on the far right.

/O
  List by files in sorted order.

/sortorder
  N By name (alphabetic)
  E By extension (alphabetic)
  G Group directories first

/P
  Pauses after each screenful of information.
Parameter –i in grep

```c
int match_icase;

int main (int argc, char **argv)
{
    [...]
    while ((opt = get_nondigit_option (argc, argv, &default_c,
    switch (opt)
    {
    [...]
    case 'i':
    match_icase = 1;
    break;
    }
}

static const char *
print_line_middle (const char *beg, const char *lim,
    const char *line_color, const char *match_color)
{
    [...]
    if (match_icase)
    {
        ibeg = buf = (char *) xmalloc(i);
        while (--i >= 0)
            buf[i] = tolower(beg[i]);
    }
```
Global configuration options

class Config {
    public static boolean isLogging = false;
    public static boolean isWindows = false;
    public static boolean isLinux = true;
}

class Main {
    public void foo() {
        if (isLogging)
            log("running foo()");
        if (isWindows)
            callWindowsMethod();
        else if (isLinux)
            callLinuxMethod();
        else
            throw RuntimeException();
    }
}
Propagating Parameters

```java
public Sequence openSequence(Transaction txn, DatabaseEntry key, SequenceConfig config) throws DatabaseException {
    checkEnv();
    DatabaseUtil.checkForNullDbt(key, "key", true);
    checkRequiredDbState(OPEN, "Can't call Database.openSequence:");
    checkWritable("openSequence");
    trace(Level.FINEST, "Database.openSequence", txn, key, null, null);

    return new Sequence(this, txn, key, config);
}

/**
 * Javadoc for this public method is generated via
 * the doc templates in the doc_src directory.
 */
public void removeSequence(Transaction txn, DatabaseEntry key) throws DatabaseException {
    delete(txn, key);
}
```
Selecting configurations

- Command line parameters
- Config file
- User dialog
- Source code
class Graph {
    Vector nv = new Vector(); Vector ev = new Vector();
    Edge add(Node n, Node m) {
        Edge e = new Edge(n, m);
        nv.add(n); nv.add(m); ev.add(e);
        if (Conf.WEIGHTED) e.weight = new Weight();
        return e;
    }
    Edge add(Node n, Node m, Weight w)
        if (!Conf.WEIGHTED) throw RuntimeException();
        Edge e = new Edge(n, m);
        nv.add(n); nv.add(m); ev.add(e);
        e.weight = w; return e;
}

void print() {
    for (int i = 0; i < ev.size(); i++) {
        ((Edge)ev.get(i)).print();
    }
}
}

class Edge {
    Node a, b;
    Color color = new Color();
    Weight weight;
    Edge(Node _a, Node _b) { a = _a; b = _b; }
    void print() {
        if (Conf.COLORED) Color.setDisplayColor(color);
        a.print(); b.print();
        if (!Conf.WEIGHTED) weight.print();
    }
}

class Node {
    int id = 0;
    Color color = new Color();
    void print() {
        if (Conf.COLORED) Color.setDisplayColor(color);
        System.out.print(id);
    }
}

class Color {
    static void setDisplayColor(Color c) { ... }
}

class Weight {
    void print() { ... }
}
199 Configuration parameters for the optimizer
(1600 total)
Limitations

- Variable code scattered in entire program
  - Hard to test in isolation
- Global variables or long parameter lists

- All code always included
  - Binary size, memory consumption
  - Performance
  - Unused functionality as attack vector
- Changes at runtime or load-time?
Limitations

• Variable code scattered in entire program
  – Hard to test in isolation
• Global variables or long parameter lists

• All code always included
  – Binary size, memory consumption
  – Performance
  – Unused functionality as attack vector
• Changes at runtime or load-time?
Preprocessors

conditional compilation

#ifdef
A Performance Analysis

What’s the performance problem?

```java
public foo() {
    ...
    if (logger.inDebug()) {
        logger.debug("We have " + conn + " connections.");
    }
}
```

Seems minor...
but if this performance gain on 1000 servers means we need 1 less machine, we could be saving a lot of money.
#include "db_int.h"

static int __rep_queue_filedone(dbenv, rep, rfp)
{
    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;
#endif
#ifndef DIAGNOSTIC
    DB_MSGBUF mb;
#else
    // over 100 lines of additional code
#endif

}
Preprocessor in Java?

- No native preprocessor
- Some compilers support conditional compilation at statement level

```java
class Example {
    public static final boolean DEBUG = false;

    void main() {
        System.out.println("immer");
        if (DEBUG) {
            System.out.println("debug info");
            printDetails();
        }
    }

    void printDetails() {
        ...
    }
}
```

External tools
Munge

- Simple external preprocessor for Java
- Originally developed for Swing 1.2

```java
public class Example {
    public static void main() {
        System.out.println("immer");
        /*if[DEBUG]*/
        System.out.println("debug info");
        /*end[DEBUG]*/
    }
}
```

```
java Munge -DDEBUG -DFEATURE2 file1.java file2.java
```
class Graph {
    Vector nv = new Vector(); Vector ev = new Vector();
    Edge add(Node n, Node m) {
        Edge e = new Edge(n, m);
        nv.add(n); nv.add(m); ev.add(e);
        /*if[WEIGHT]*/
        e.weight = new Weight();
        /*end[WEIGHT]*/
        return e;
    }
    /*if[WEIGHT]*/
    Edge add(Node n, Node m, Weight w) {
        Edge e = new Edge(n, m);
        nv.add(n); nv.add(m); ev.add(e);
        e.weight = w; return e;
    }
    /*end[WEIGHT]*/
    void print() {
        for(int i = 0; i < ev.size(); i++) {
            ((Edge)ev.get(i)).print();
        }
    }
}
/*if[COLOR]*/
class Color {
    static void setDisplayColor(Color c) {
        ... 
    }
}
/*end[COLOR]*/
/*if[WEIGHT]*/
class Weight { void print() { ... } }
/*end[WEIGHT]*/
/*if[COLOR]*/
class Node {
    int id = 0;
    /*if[COLOR]*/
    Color color = new Color();
    /*end[COLOR]*/
    void print() {
        Color.setDisplayColor(color);
        a.print(); b.print();
        /*if[WEIGHT]*/
        weight.print();
        /*end[WEIGHT]*/
    }
}
/*end[COLOR]*/
2000 Features

100000 Features
Discussion

• Compile-time configuration
• Can remove arbitrary code before compilation
• Simple programming model

• Bad reputation
  – Scattered code
  – Error prone
  – Hard to understand
  – Invites neglecting design
  – Hinder tool support
class Stack {
    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();
    }
}
Femto OS
int n = NUM2INT(num);
#ifndef FEAT_WINDOWS
    w = curwin;
#else
    for (w = firstwin; w != NULL;
        w = w->w_next, --n)
#endif
    if (n == 0)
        return window_new(w);
int n = NUM2INT(num);
#ifdef FEAT_WINDOWS
    w = curwin;
#else
    for (w = firstwin; w != NULL; w = w->w_next, --n)
#endif
    if (n == 0)
        return window_new(w);

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;
}
Error Prone

```c
static int _rep_queue_filedone(...) {
    DB_ENV *dbenv;
    REP *rep;
    __rep_fileinfo_args *rfp; {
#endif
    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;
#endif
    //over 100 lines of additional code
    
#ifdef TABLES
    class Table {
        void insert(Object data, Txn txn) {
            storage.set(data, txn.getLock());
        }
    }
#endif
#endif
#endif

#ifdef WRITE
    boolean set(...) { ... }
#endif
```
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
    //over 100 lines of additional code
    }
#endif
#endif

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

A matter of scale

Example: Session expiration in the Apache Tomcat Server
Design Pattern and Frameworks

best practice
separation
Strategy Pattern
Template Method Pattern

AbstractClass
{abstract}

templateMethod()

opA()
opB()

ConcreteClass

ConcreteClass

opA()
opB()

{method}

{...
opA();
...
opB();}
Inheritance for Variability

modular, but inflexible
Inflexible Extension Mechanism

cf. White-Box Framework
Work arounds?

• Combining inheritance hierarchies
  – Combinatorical explosion
  – Massive code replication

• Multiple inheritance
  – Diamond problem
Decorator Pattern

- `IStack` Interface:
  - `push()`, `pop()`, `size()`

- `Stack` Class:
  - `values`
  - `push()`, `pop()`, `size()`

- `StackDecorator` Class:
  - `delegate`
  - `push()`, `pop()`, `size()`

- `LockedStack` Class:
  - `lock()`, `unlock()`

- `UndoStack` Class:
  - `log`
  - `push()`, `pop()`, `undo()`

- `SecureStack` Class:
  - `keyphrase`
  - `push()`, `pop()`, `encrypt()`, `decrypt()`
Frameworks

• Set of abstract and concrete classes
• Reusable solutions of a set of problems
• Planned hot spots, extended by plug-ins

• Extension mechanisms often based on design patterns
  – Strategy
  – Template method
  – Observer
Discussion

• Separated, modular extensions
  – Compile and test plug-ins separately
• Compile-time/load-time configuration
• Well understood in practice
• Hard to design, hard to evolve
• Small runtime overhead
• Suited for coarse-grained extensions
ANALYSIS
Example: Interrupt Problem

1. int foo() {
2.   unsigned long flags;
3.   int rv;
4.   save_flags(flags);
5.   cli();
6.   rv = dont_interrupt();
7.   if (rv > 0) {
8.     do_stuff();
9.     restore_flags();
10.   } else {
11.     handle_error_case();
12.   }
13.   return rv;
14. }

Abstraction (before statement)

2-4: enabled
5: enabled
6: disabled
7: disabled
8: disabled
9: disabled
11: disabled
13: unknown

Error: did not reenable interrupts on some path
320 features

more configurations than estimated

atoms in the universe
From HP to Linux

2000 Features
Inhouse configuration
100 Printers (Product Map)
30 New Printers per Year

∀ p ∈ PL . . .

10000 Features
End user configures
$2^{10000}$ Configurations
Dead-Code Detection

```c
#ifdef A
#ifndef A
#endif
#elif defined(X)
#else
#endif
true
A
A ^ not A
A
not A ^ X
not A ^ not X
```

Analysis:
SAT(pc(block i))

Dead code
Variability Models

keyless_entry → power_locks

DirectX because set by user
Win because (DirectX) implies (Win)
not Unix because ((OS if (Unix or Win)) and atmostone(Unix,Win))
```c
#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, rthal_
unsigned long rthal_clockfreq_arg
module_param_named(clockfreq, rthal_

#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
```
static void
prline (char const *beg, char const *lim, int sep)
{
  int matching;
  const char *line_color;
  const char *match_color;

  if (!only_matching)
    print_line_head(beg, lim, sep);

  matching = (sep == SEP_CHAR_SELECTED) ^ !out_invert;

  if (color_option)
  {
    line_color = (sep == SEP_CHAR_SELECTED)
      ^ (out_invert && (color_option < 0))
      ? selected_line_color : context_line_color;
    match_color = (sep == SEP_CHAR_SELECTED)
      ? selected_match_color : context_match_color;
  }
  else
    line_color = match_color = NULL; /* Shouldn't be used. */

  if ( (only_matching && matching)
    || (color_option && (*line_color || *match_color))
  )
  {
    /* We already know that non-matching lines have no match (to colorize). */
    if (matching && (only_matching || *match_color))
      beg = print_line_middle(beg, lim, line_color, match_color);

    /* FIXME: this test may be removable. */
    if (!only_matching && *line_color)
      beg = print_line_tail(beg, lim, line_color);
  }

  if (!only_matching && lim > beg)
    fwrite (beg, 1, lim - beg, stdout);
Pairwise Testing

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>on</td>
<td>on</td>
<td>on</td>
<td>on</td>
</tr>
<tr>
<td>on</td>
<td>on</td>
<td>on</td>
<td>off</td>
</tr>
<tr>
<td>on</td>
<td>off</td>
<td>on</td>
<td>on</td>
</tr>
<tr>
<td>on</td>
<td>off</td>
<td>off</td>
<td>off</td>
</tr>
<tr>
<td>off</td>
<td>on</td>
<td>on</td>
<td>on</td>
</tr>
<tr>
<td>off</td>
<td>on</td>
<td>off</td>
<td>off</td>
</tr>
<tr>
<td>off</td>
<td>off</td>
<td>on</td>
<td>on</td>
</tr>
<tr>
<td>off</td>
<td>off</td>
<td>off</td>
<td>off</td>
</tr>
</tbody>
</table>

Cohen et al. *Interaction testing of highly-configurable systems in the presence of constraints*. ISSTA 2007
Pairwise Testing

<table>
<thead>
<tr>
<th></th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>on</td>
<td>on</td>
<td>on</td>
<td>on</td>
</tr>
<tr>
<td>on</td>
<td>on</td>
<td>on</td>
<td>off</td>
</tr>
<tr>
<td>on</td>
<td>off</td>
<td>on</td>
<td>on</td>
</tr>
<tr>
<td>on</td>
<td>off</td>
<td>off</td>
<td>off</td>
</tr>
<tr>
<td>off</td>
<td>on</td>
<td>on</td>
<td>on</td>
</tr>
<tr>
<td>off</td>
<td>on</td>
<td>off</td>
<td>off</td>
</tr>
<tr>
<td>off</td>
<td>off</td>
<td>off</td>
<td>on</td>
</tr>
<tr>
<td>off</td>
<td>off</td>
<td>off</td>
<td>off</td>
</tr>
</tbody>
</table>

Cohen et al. *Interaction testing of highly-configurable systems in the presence of constraints.* ISSTA 2007
Closed-World Analysis

∀ p ∈ PL . p is well-typed

Specified Variability

(WORLD v BYE) ^ ¬ (WORLD ^ BYE)

Implementation Variability

Check Consistency

```c
#include <stdio.h>

#ifndef WORLD
    char * msg = "Hello_World\n";
#else
    char * msg = "Bye_bye!\n";
#endif

main() {
    printf(msg);
}
```
Other Topics

• Product planning, product-line scoping
• Team organization
• Development and design process (domain engineering)
• More implementation techniques (components, version control systems, feature/aspect-oriented development)
• Product-line analysis
BUY OUR PRODUCT
(we’re on a billboard)
Further Reading


15-313: Foundations of Software Engineering

• Advanced Topics
  – From Design Pattern to Architectures
  – Dynamic and Static Analysis
  – Inspection, Advanced Testing
  – Git, Continuous Integration

• Human Aspects in Software Engineering
  – Teams
  – Process
  – Requirements
  – Economics, Risks
  – Empirical Results

• Concurrency and Security as Themes
Summer Project: Analyzing Github

- Setup Analysis Infrastructure
- Metrics for Active Developers
- Speculative Unit Test Execution

- Unix, Databases +
Summary

• Software product lines: Systematic reuse by planning variations
• From runtime parameters and conditional compilation to design patterns and frameworks
• Analysis for product lines