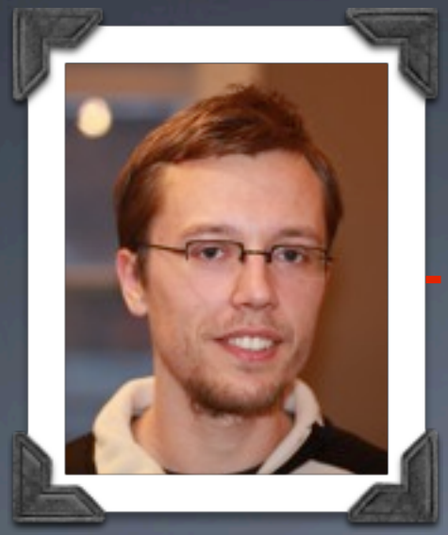
On Software **Release** Engineering

Bram Adams

ÉCOLE POLYTECHNIQUE MONTRÉAL

MCIS

Latest version: http://mcis.polymtl.ca/~bram/publications/2012/icse_tb.pdf



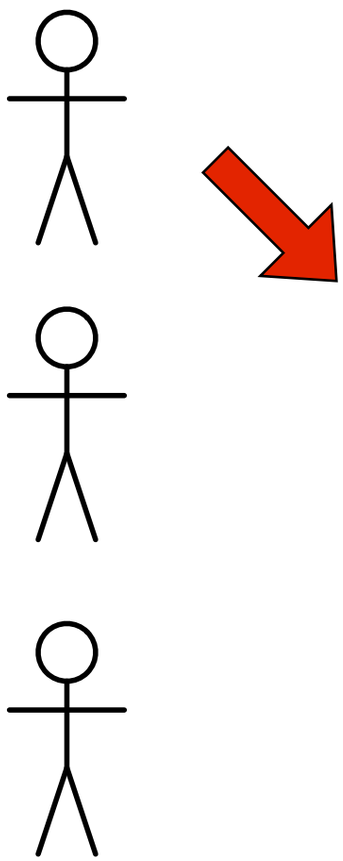
M
C·I·S

(Lab on Maintenance, Construction and Intelligence of Software)

ÉCOLE POLYTECHNIQUE MONTRÉAL


Express yourself in the world's largest 3D Chat and Dress-Up community!
Member Login
On average we deploy new code **fifty** times a day.
in 66 different countries!
Sign in with:
Choose Your FREE Avatar
Over 2 Million people *like* IMVU on Facebook!
FREE

Continuous Delivery



15k tests

test

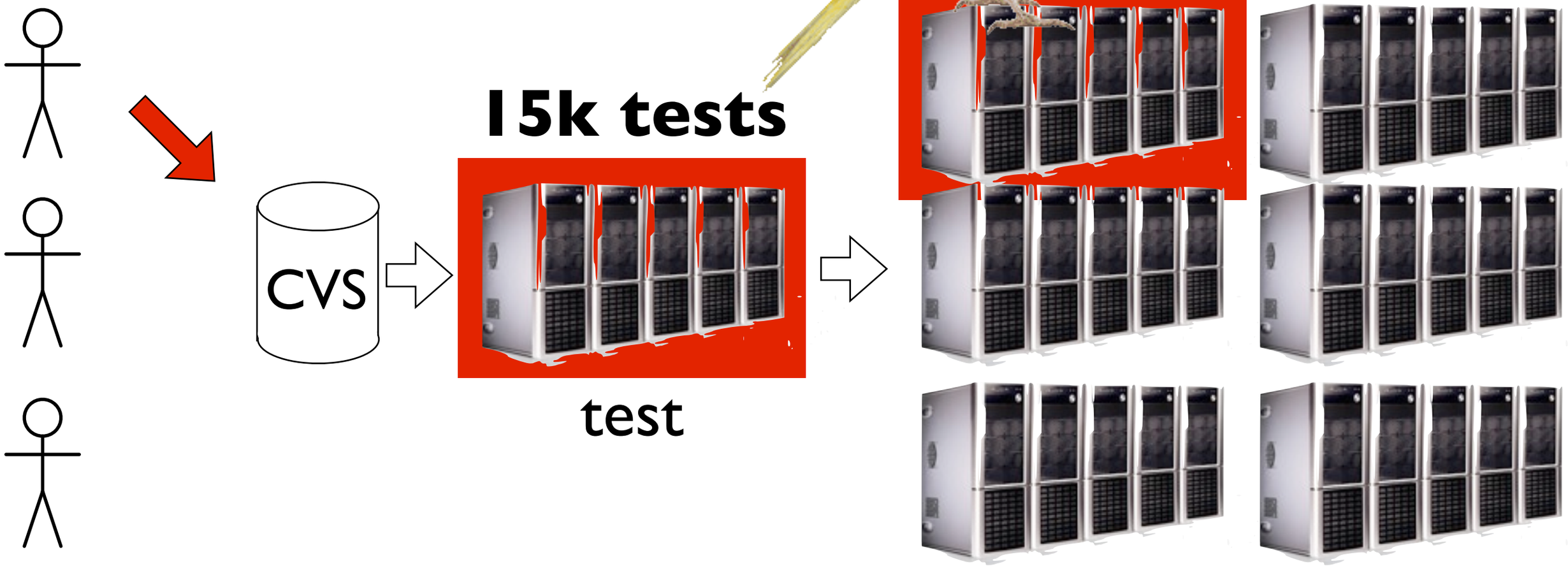
staging/production

continuous integration

9 min.

http://goo.gl/qPT6

Continuous Delivery

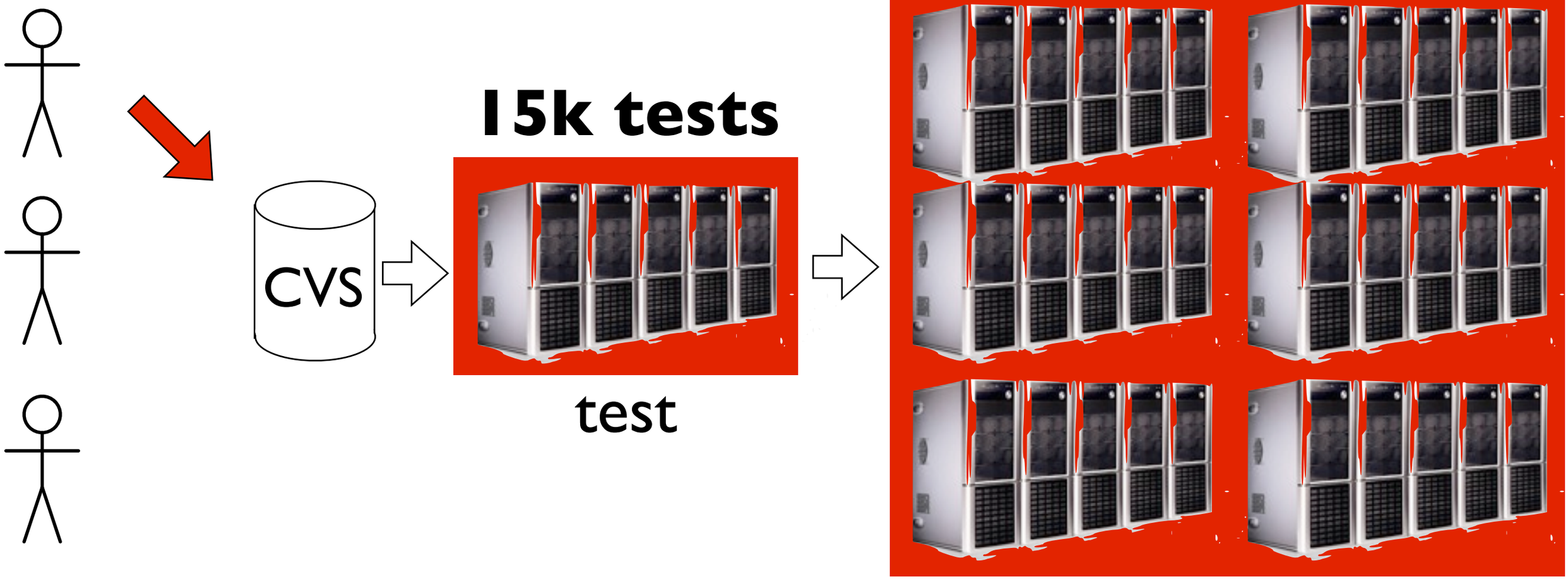


continuous integration

9 min.

http://goo.gl/qPT6

Continuous Delivery



continuous integration

9 min.

6 min.

http://goo.gl/qPT6

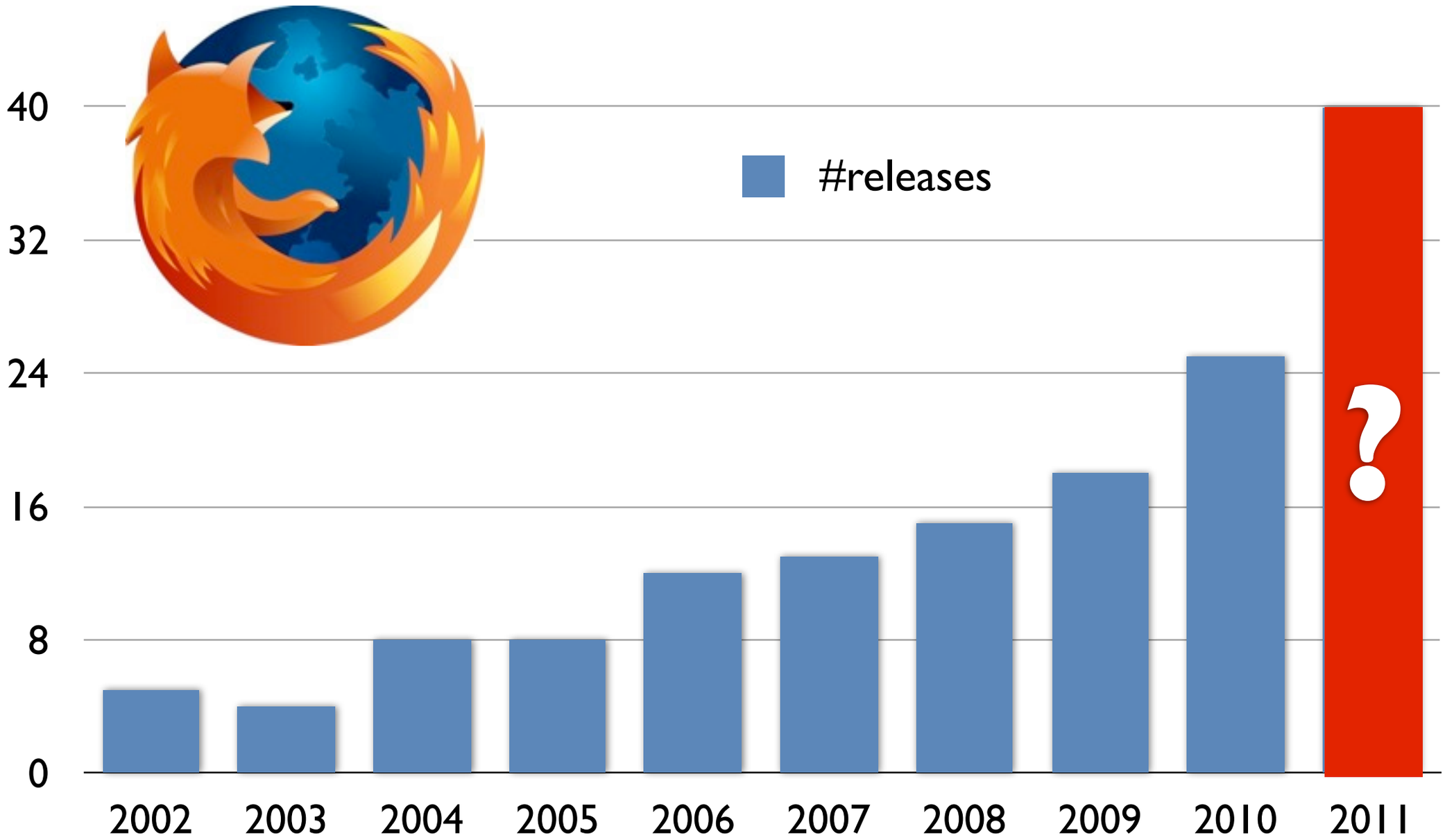
Work fast and
don’t be afraid to break
things.

Mark Zuckerberg
CEO & Founder, Facebook

http://goo.gl/UICW


Build a
little and then test it.
Build some more and test
some more.
James Whittaker

Even Desktop Apps Release More Frequently



http://en.wikipedia.org/wiki/History_of_Firefox

Firefox' New Release Schedule (1)

http://hacks.mozilla.org/wp-content/uploads/2012/05/firefox-releases.jpg

Firefox' New Release Schedule (2)

http://hacks.mozilla.org/wp-content/uploads/2012/05/firefox-releases.jpg

Release Pipeline

http://hacks.mozilla.org/wp-content/uploads/2012/05/rapid-release.jpg

Firefox Release Howto (1)

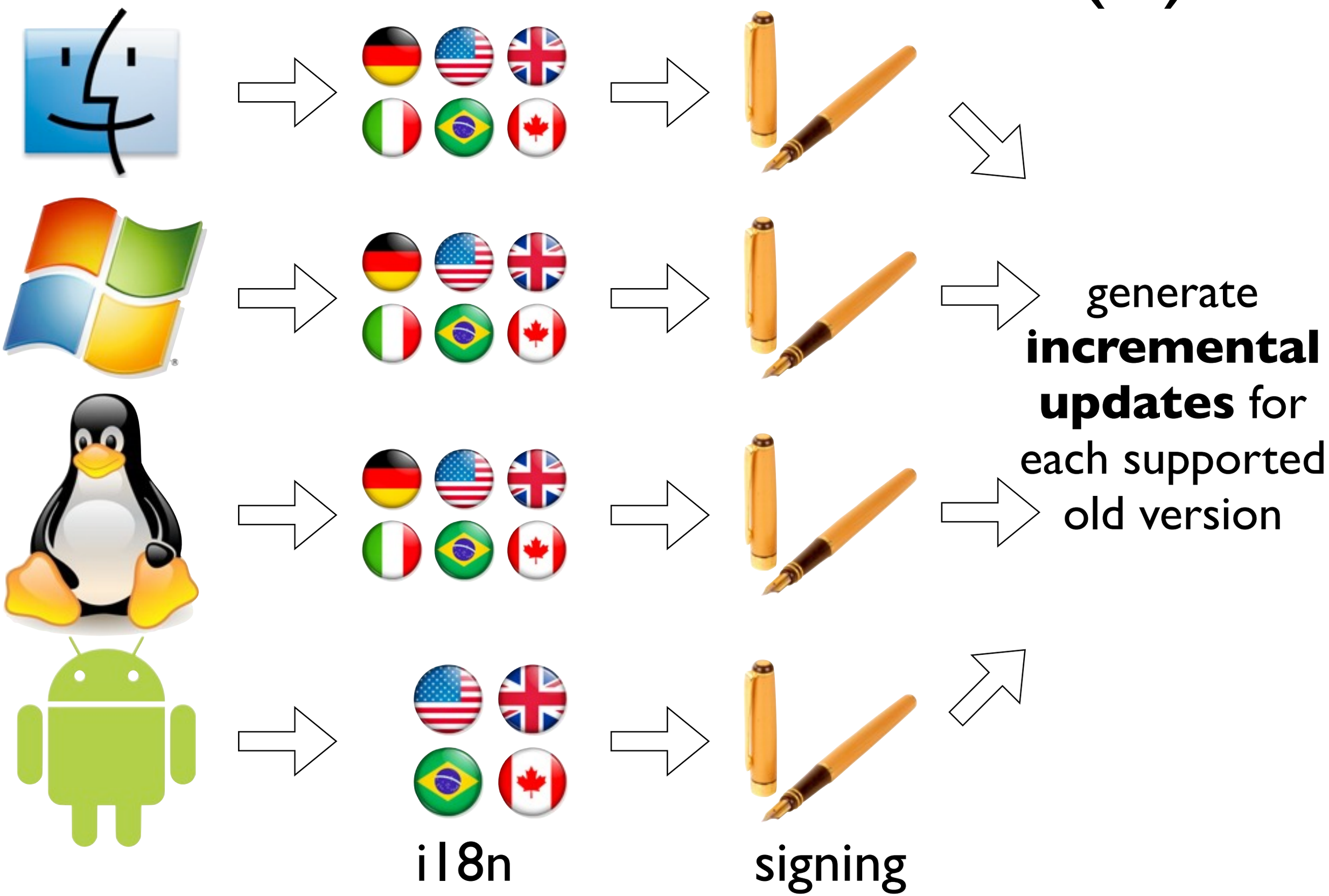
85+ repositories

configuration sanity check

tag everything

build

Firefox Release Howto (2)



Google™ ... yet Software Systems keep on Growing!

>5k developers

<150k tests/ commit

>50k builds/day

build cache with >50TB memory

>2k projects

>50M tests/day

20 code changes/min.

compilation in the cloud

http://tims-ideas.blogspot.com/2011/05/live-blog-from-icse-view-of-icses.html


http://behrns.files.wordpress.com/2008/03/ikea-car.jpg

Release Engineering

in-house/3rd party development

integrate

test

build

reduce cycle time!

deployment

Out of Scope (but Highly Related)

release management

variability management

release planning

app stores

...

software process

impact on project management

packaging technologies

software product lines

http://behrns.files.wordpress.com/2008/03/ikea-car.jpg

Release Engineering

in-house/3rd party development

integrate

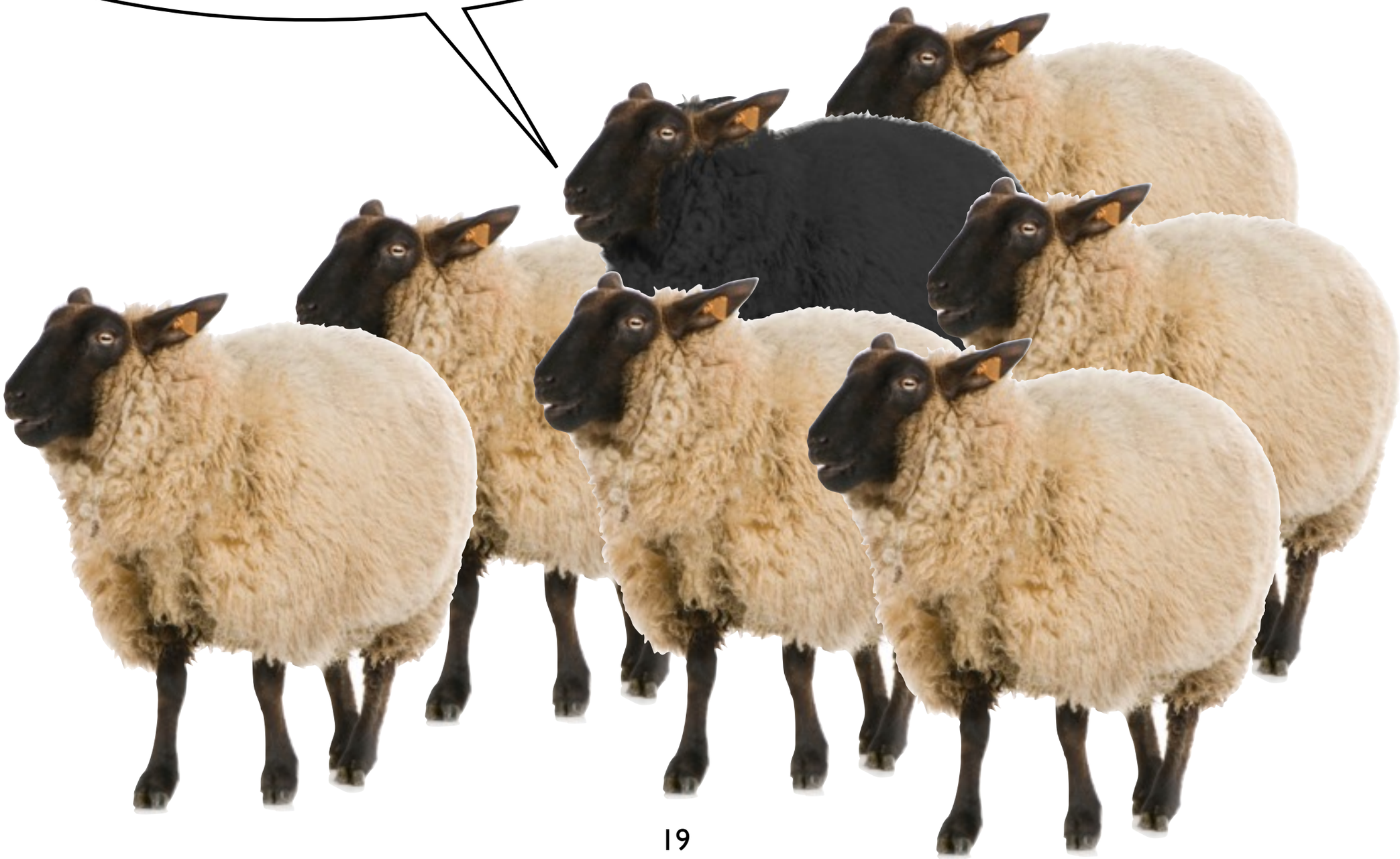
test

build

reduce cycle time!

deployment

The Build Process



The Build Process

http://behrns.files.wordpress.com/2008/03/ikea-car.jpg

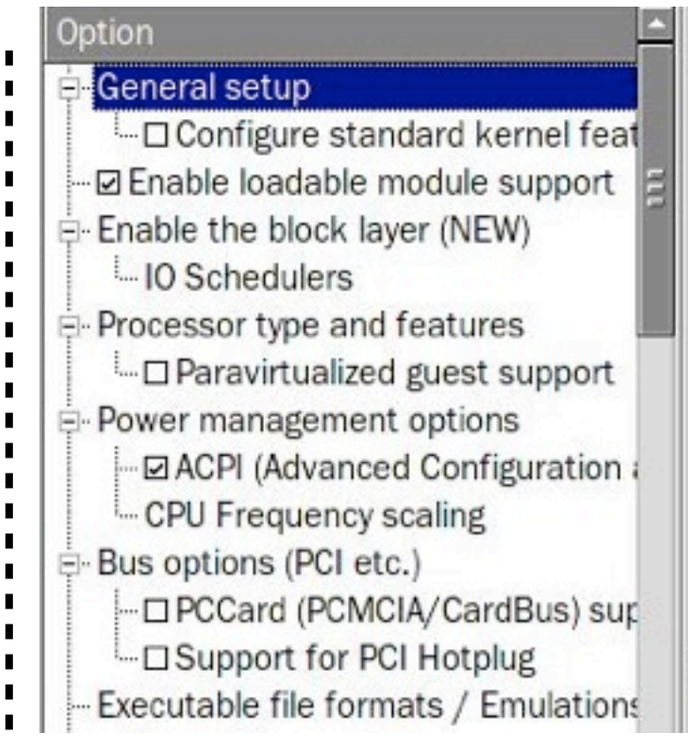
?

Step 1 - Configuration

Autotools

CMake
Cross-platform Make

Features



Tools

GCC

python

Step 2 - Construction

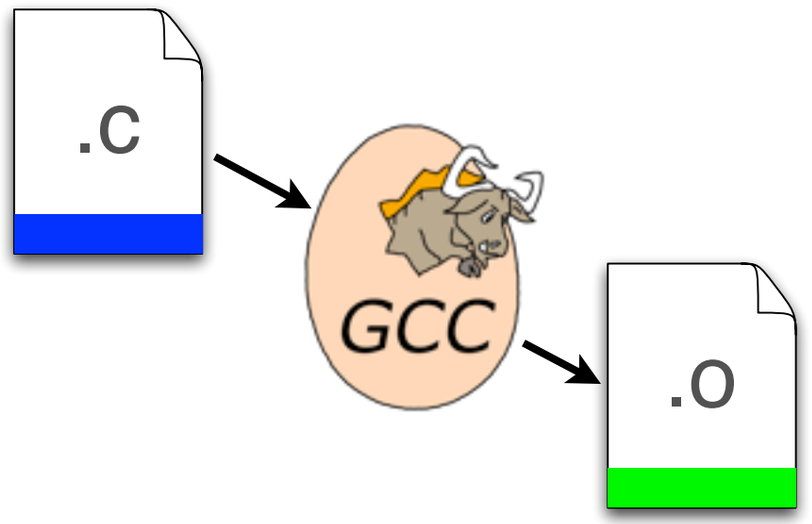
Make

maven

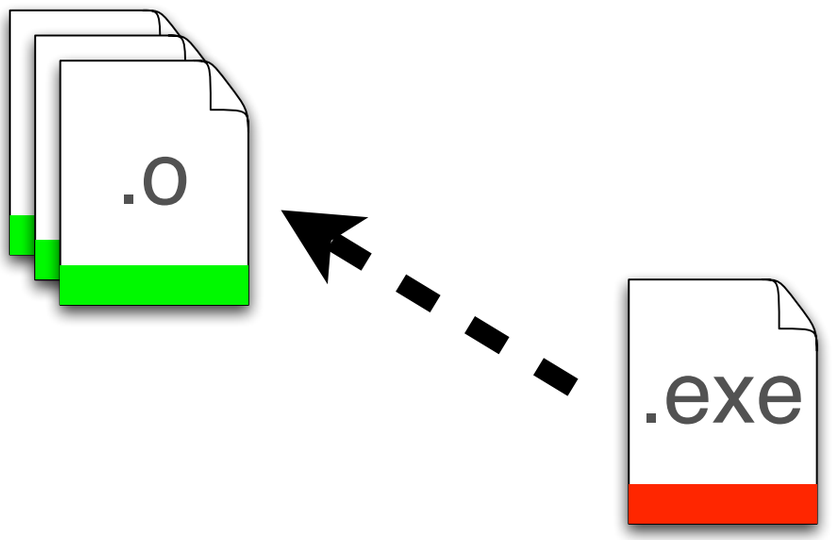
<APACHE ANT>

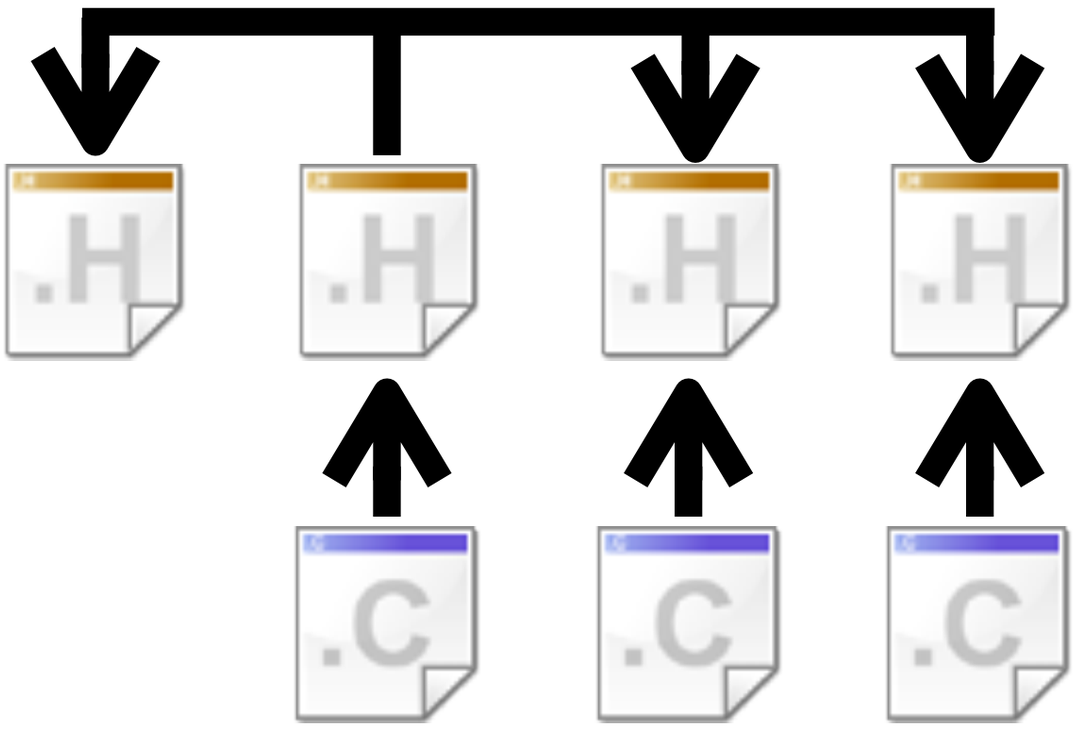
Prescriptions

=



Dependencies





Source Code

Build System

```
# KERNELRELEASE can change from a few different places, meaning version.h
# needs to be updated, so this check is forced on all builds

uts_len := 64
define filechk_utsrelease.h
        if [ `echo -n "$(KERNELRELEASE)" | wc -c ` -gt $(uts_len) ]; then \
          echo '"$(KERNELRELEASE)" exceeds $(uts_len) characters' >&2;    \
          exit 1;                                                         \
        fi;                                                               \
        (echo \#define UTS_RELEASE \"$(KERNELRELEASE)\";)
endef

define filechk_version.h
        (echo \#define LINUX_VERSION_CODE $(shell                        \
        expr $(VERSION) \* 65536 + $(PATCHLEVEL) \* 256 + $(SUBLEVEL));   \
        echo '#define KERNEL_VERSION(a,b,c) (((a) << 16) + ((b) << 8) + (c))';)
endef

include/linux/version.h: $(srctree)/Makefile FORCE
        $(call filechk,version.h)

include/generated/utsrelease.h: include/config/kernel.release FORCE
        $(call filechk,utsrelease.h)

PHONY += headerdep
headerdep:
        $(Q)find include/ -name '*.h' | xargs --max-args 1 scripts/headerdep.pl
```

@deanberris
Dean Michael Berris

I hate GNU Make especially those Makefiles that are hand-crafted and un-documented. It all feels like a bad hack.

24 Feb via web ☆ Favorite ↻ Retweet ↩ Reply

KDE 4 is leaving the aging "autotool" build chain behind. Some developers, not only in KDE, like to **nickname the autotools as "auto-hell"** because of its difficulty to comprehend architecture. [http://lwn.net/Articles/188693/]

Our record so far is a project we inherited with an Ant script weighing in at 10,000 lines of XML. Needless to say, this project required an entire team devoted to keeping the build working—a complete **waste of resources**. [Jez Humble & David Farley]

Build Systems are **Complex**

>5 MLOC & ~20 Years of History

"Design Recovery and Maintenance of Build Systems" (Adams et al.)

"The Evolution of the Linux Build System" (Adams et al.)

Guess Visualization

o
a
h
c
dylib

Quake 3

server

game logic

client UI

client renderer

quake3.exe

network

d(a)emon

>>>
>>>

Guess Visualization

Quake 3

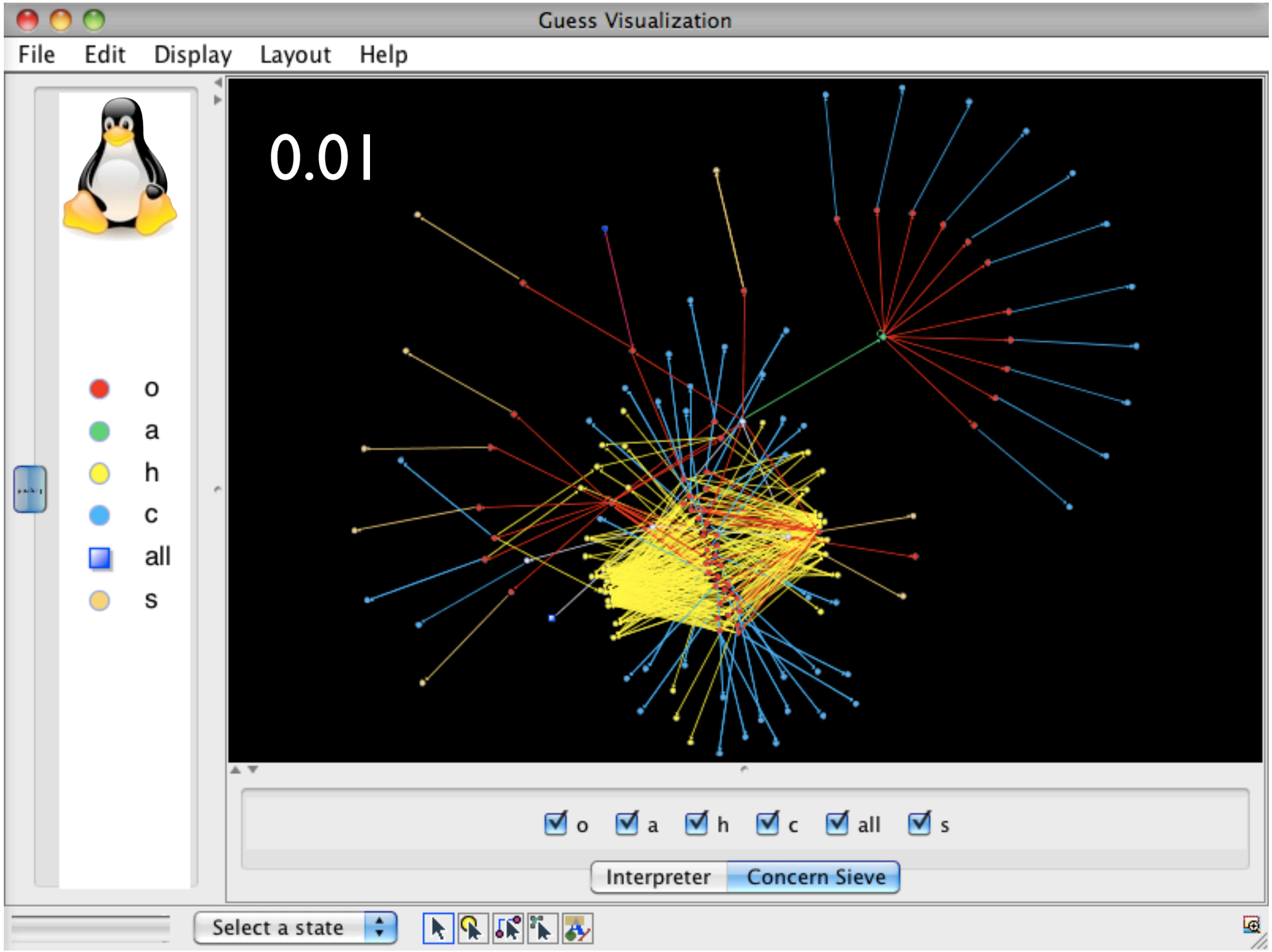
conditional compilation

original game

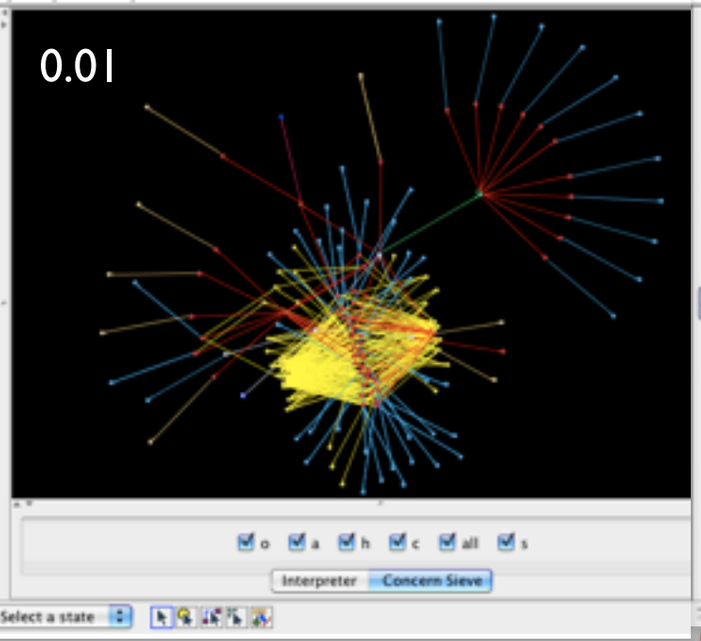
expansion pack

- o
- a
- h
- c
- dylib

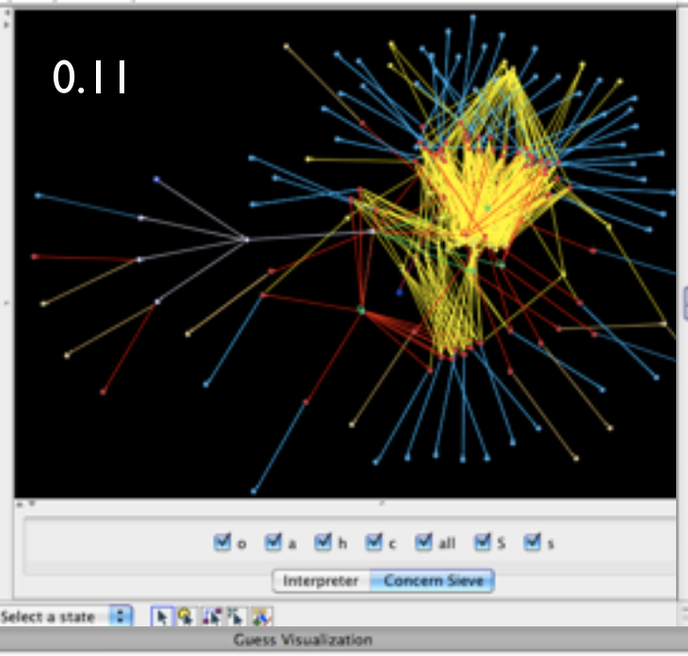
"The Evolution of the Linux Build System" (Adams et al.)



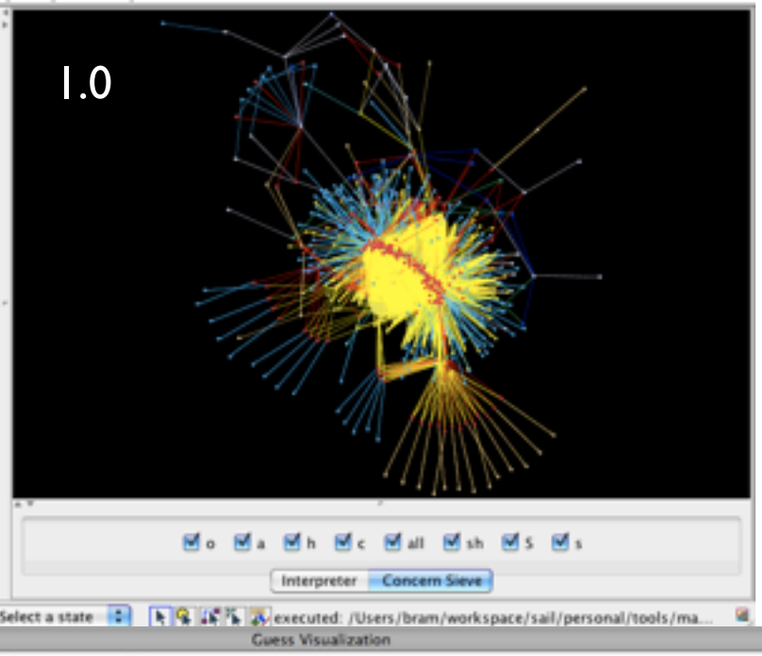
0.01



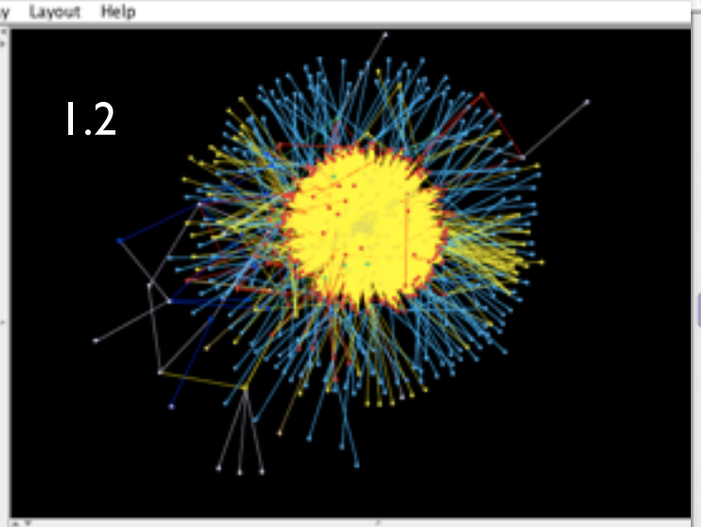
0.11



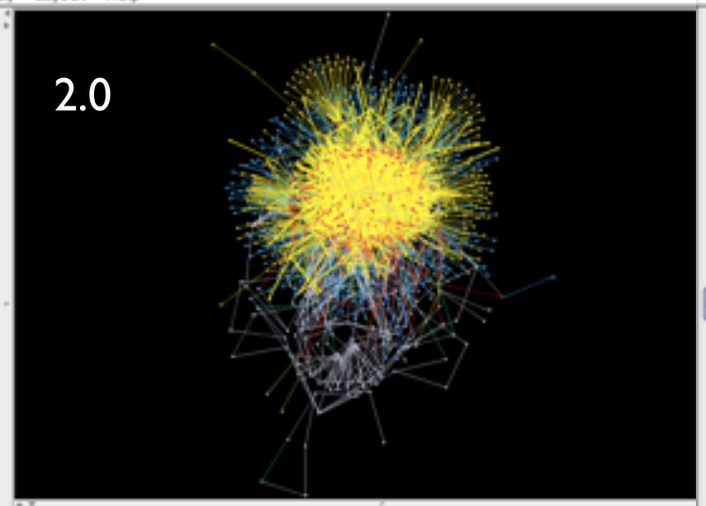
1.0



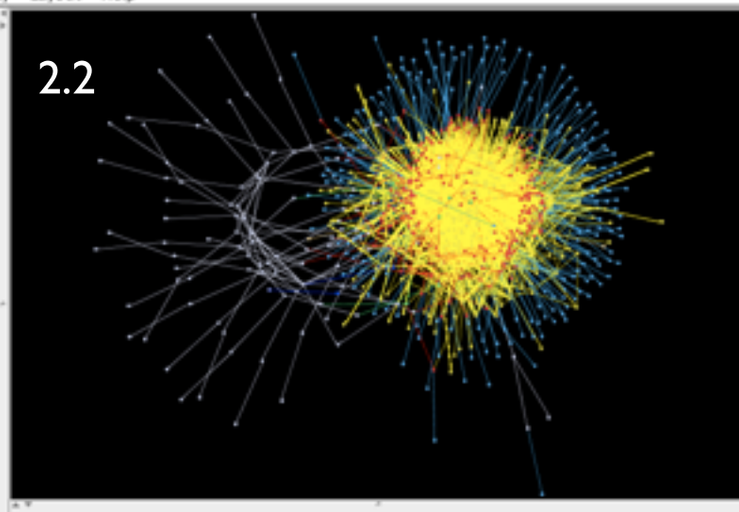
1.2



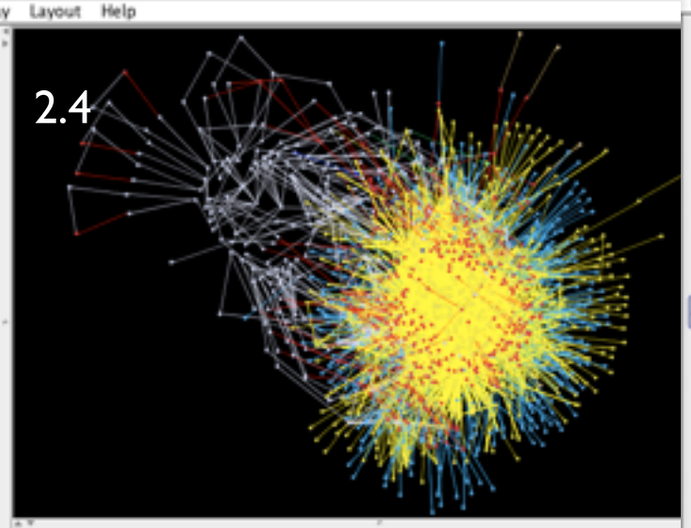
2.0



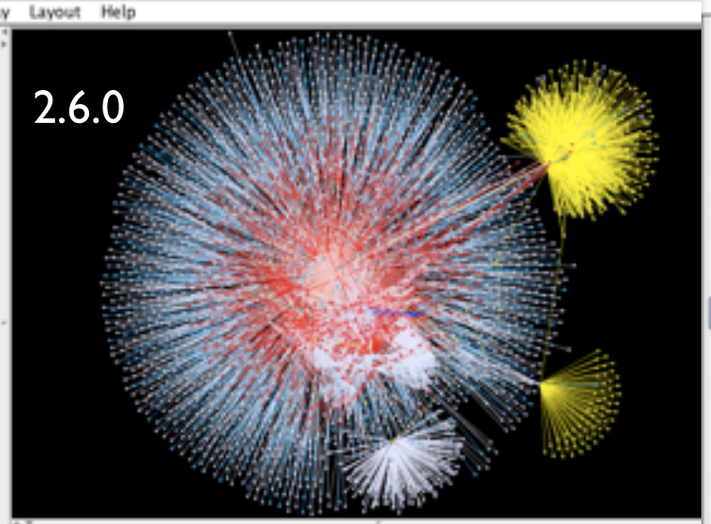
2.2



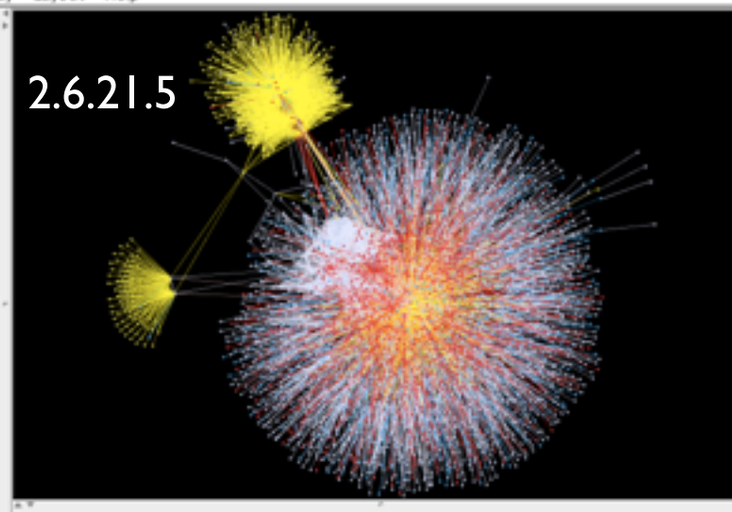
2.4

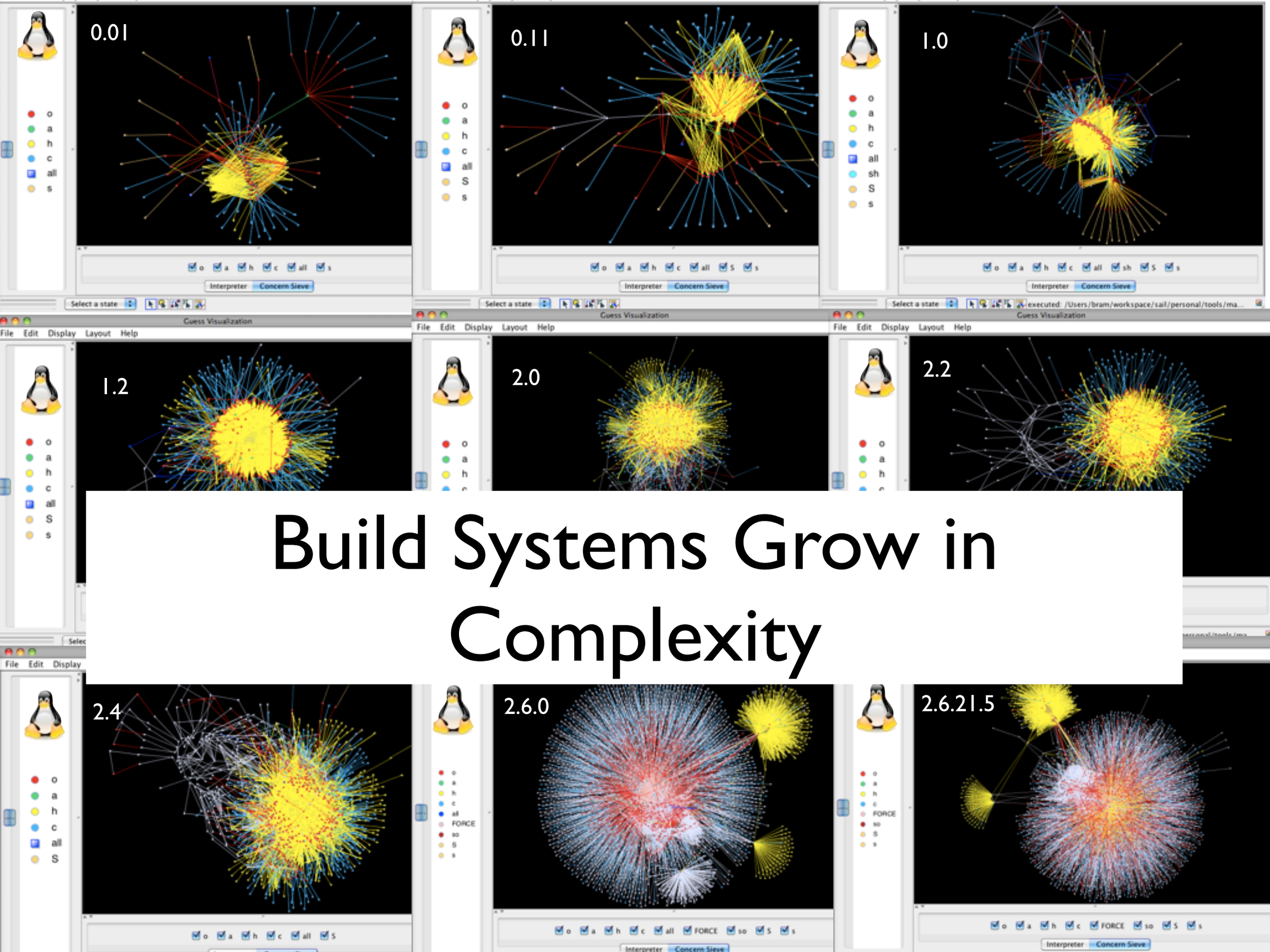


2.6.0



2.6.21.5





Build Systems Grow in Complexity

Build Systems Require **12%** of a Developer's Time (on average)

Build maintenance slows down development!

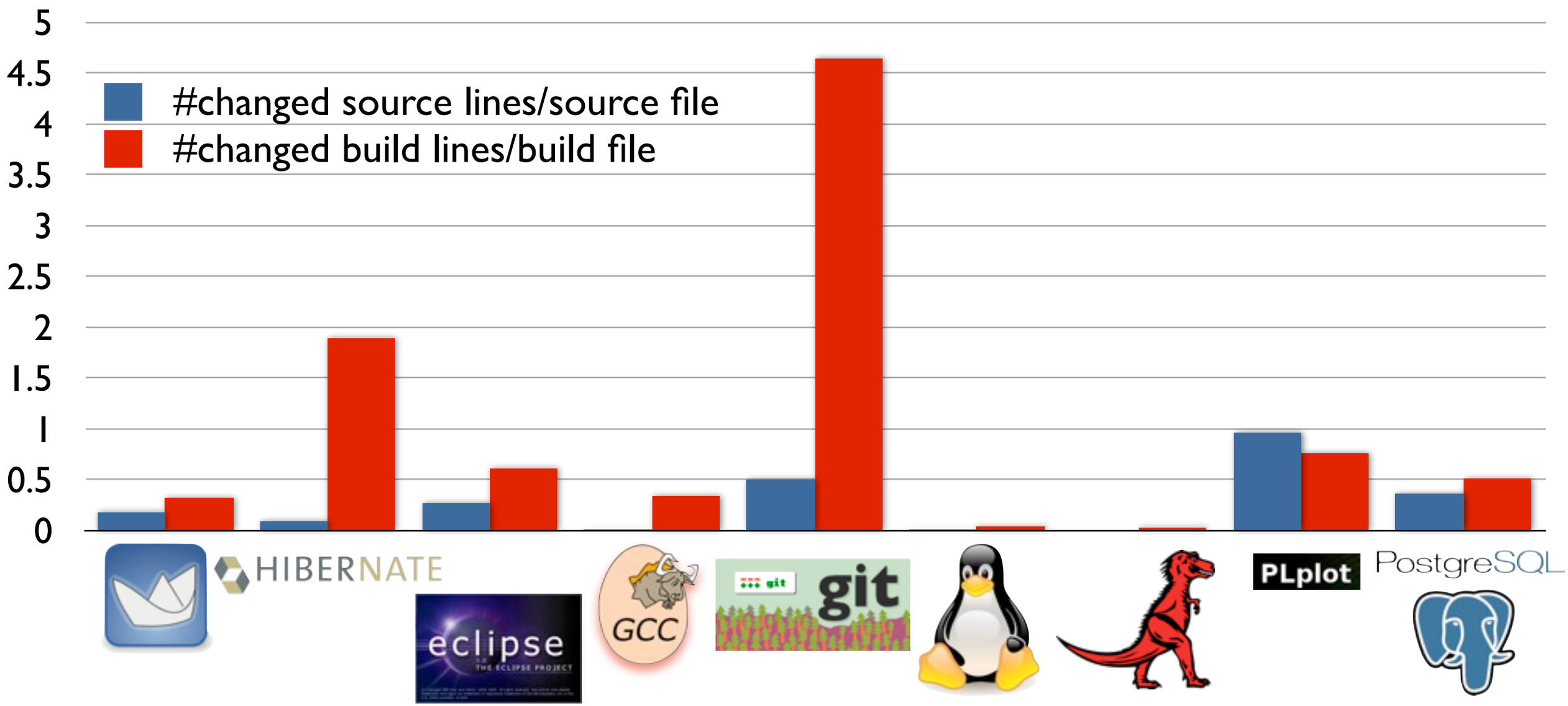
Software in the DOE: The Hidden Overhead of the "Build" (Kumfert et al.)

An Empirical Study of Build Maintenance Effort (McIntosh et al.)

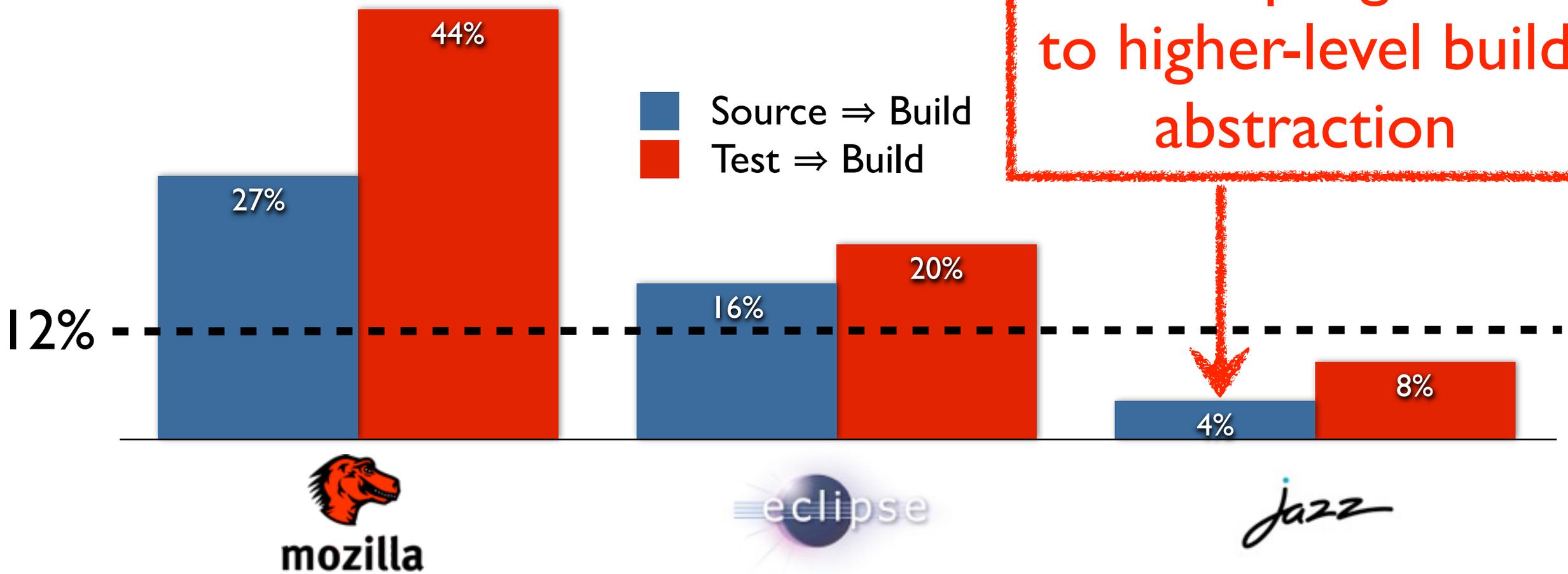
>36 MLOC & ~120 Years of History

An Empirical Study of Build Maintenance Effort (McIntosh et al.)

Build Files **Change Relatively More** than Source Code Files



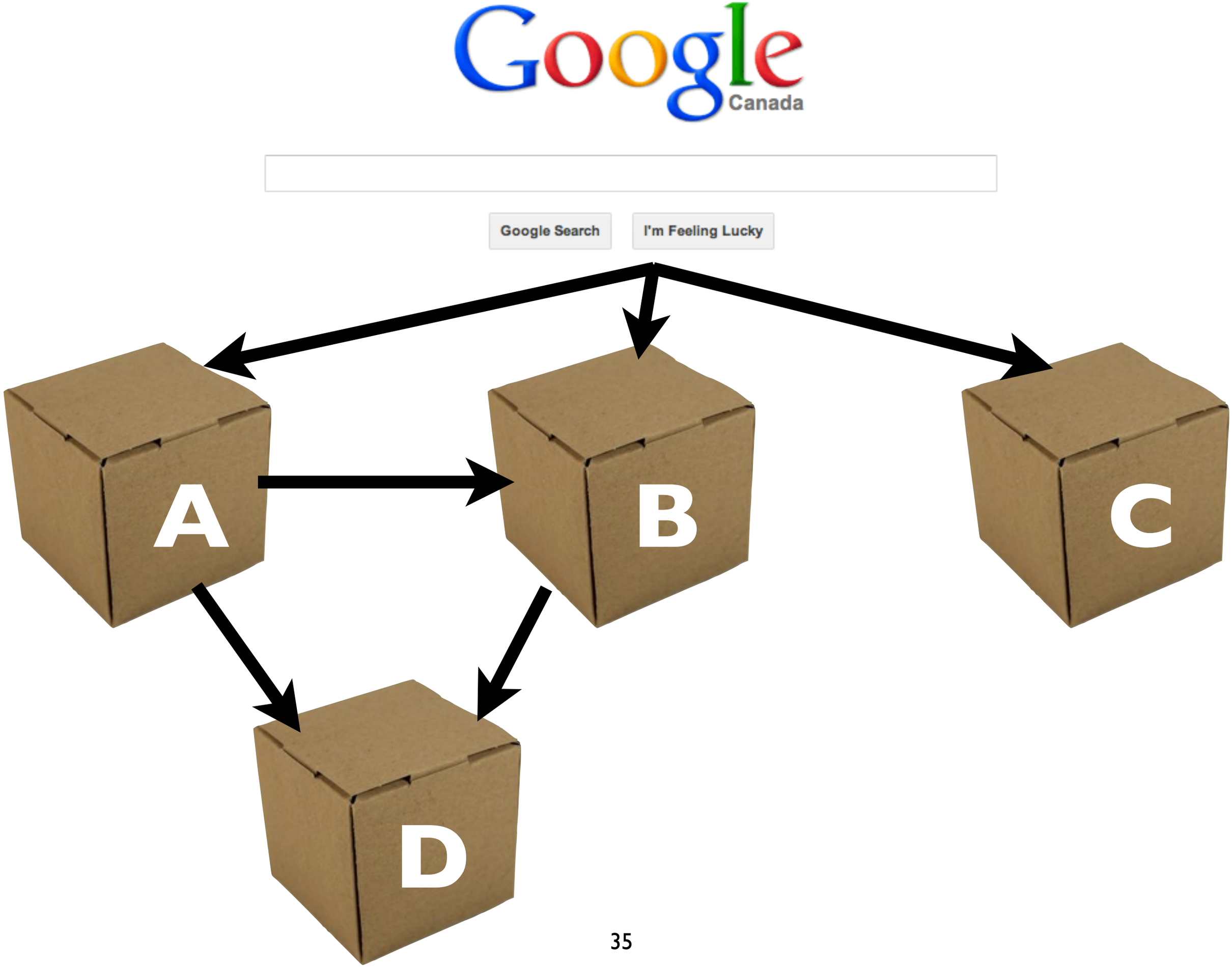
The Build System Requires Significant **Maintenance**

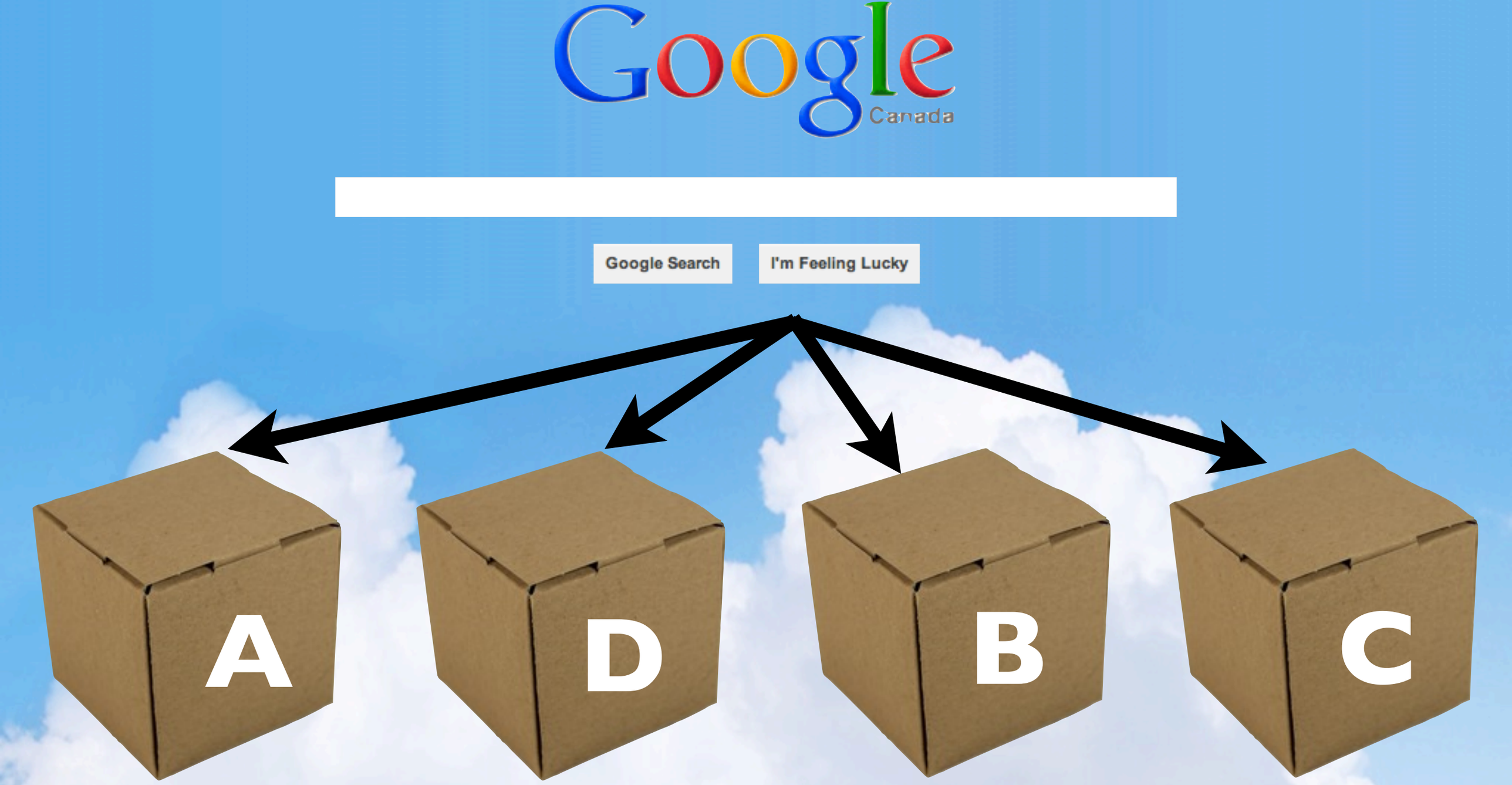


Low coupling thanks to higher-level build abstraction

Our Build Systems need HELP

Google Search I'm Feeling Lucky


Google
Canada
Google Search
I'm Feeling Lucky
A
B
C
D


Google
Canada
Google Search
I'm Feeling Lucky
A
D
B
C

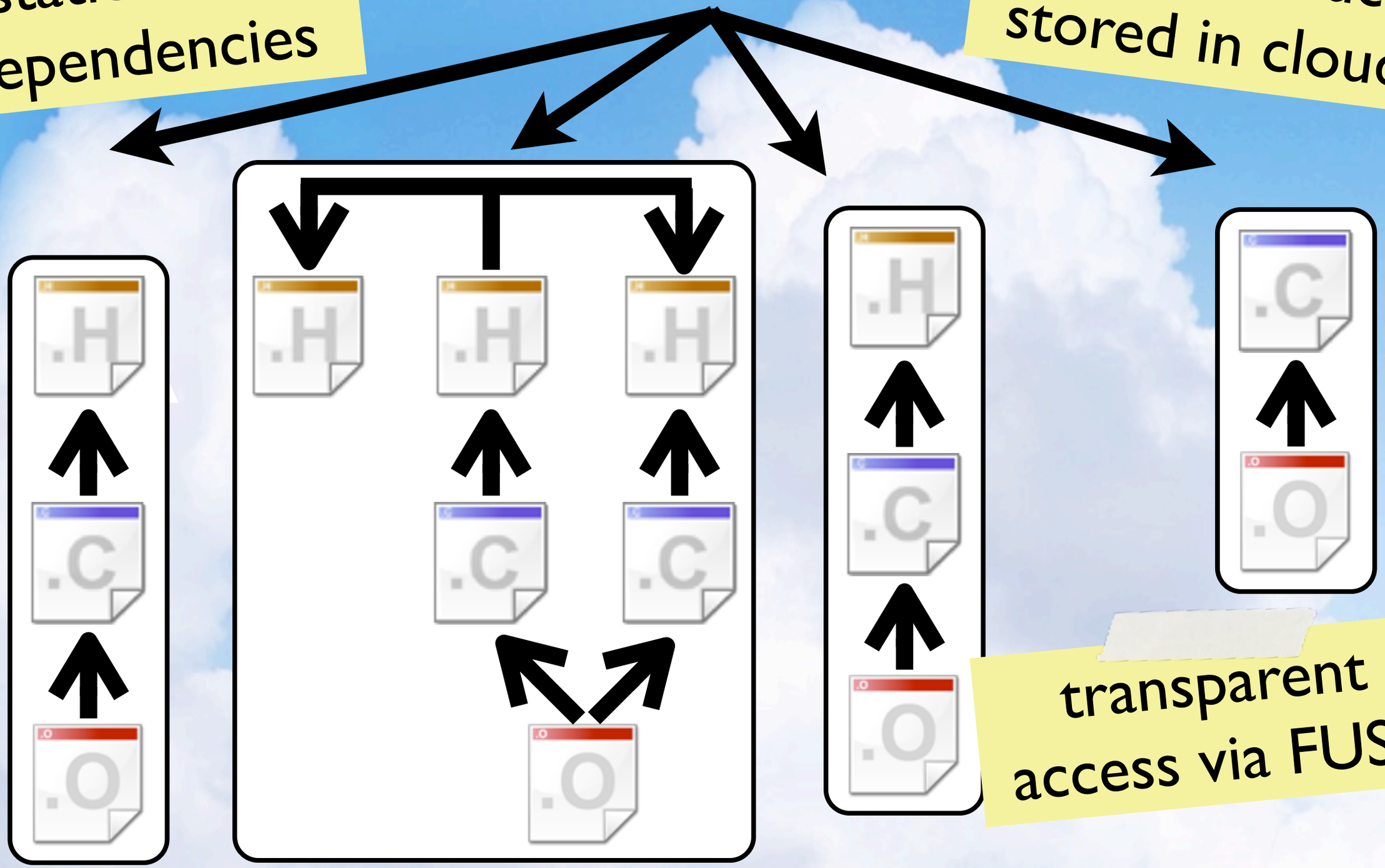
Google

Canada

Google Search | I'm Feeling Lucky

static build dependencies

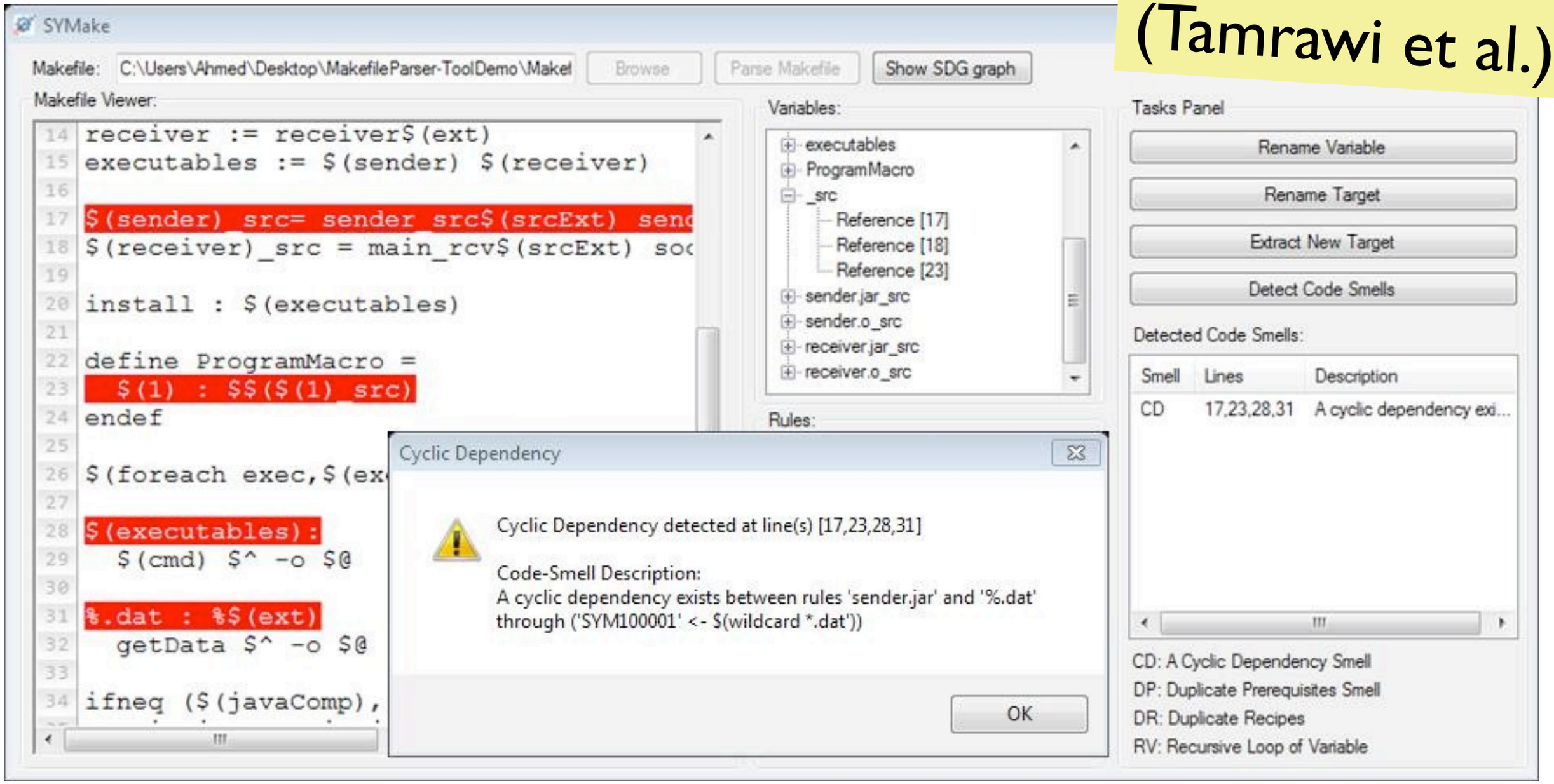
all files/products stored in cloud



transparent access via FUSE

Build System Quality

SYMake
(Tamrawi et al.)



Build System Quality

Clipper
(Morgenthaler et al.)

Entry Points | Rule Protobuf

com/google/devtools/deps/demolition/clipper/Clipper

Refactoring Candidates

Filter Target Names:

Target Name	Type	Alias	Depth	Density	Symbols	Used	Utilization	Transitive	Used	Utilization
//third_party/java/jung:jung_algorithms	java_library	yes	2	1	138	12	8.7%	1513	41	2.7%
//third_party/java/jung:jung_api	java_library	yes	2	1	38	11	28.9%	424	29	6.8%
//third_party/java/jung:jung_graph_impl	java_library	yes	2	1	31	4	12.9%	455	33	7.3%
//third_party/java/jung:jung_visualization	java_library	yes	2	1	221	94	42.5%	1734	135	7.8%
//third_party/java/jakarta_velocity:jakarta_velocity	java_library	yes	2	1	246	150	61.0%	899	205	22.8%
//java/com/google/devtools/deps/demolition/common:p...	java_library	yes	2	1	138	86	62.3%	2997	1653	55.2%
//third_party/java/asm:asm-commons	java_library	yes	3	1	24	1	4.2%	24	1	4.2%
//third_party/java/jung/v2_0_1:jung_graph_impl	java_library		3	1	31	4	12.9%	455	33	7.3%
//third_party/java/jung/v2_0_1:jung_visualization	java_library		3	1	221	94	42.5%	1734	135	7.8%
//third_party/java/jakarta_velocity/v1_5:v1_5	java_library		3	1	246	150	61.0%	899	205	22.8%

Calculate Transitive Cost

//third_party/java/jung:jung_api

Symbols | Rule Protobuf | Reachability

Tu et al., Nadi et al., Berger et al., Tartler et al., ...

http://behrns.files.wordpress.com/2008/03/ikea-car.jpg

Release Engineering

in-house/3rd party development

integrate

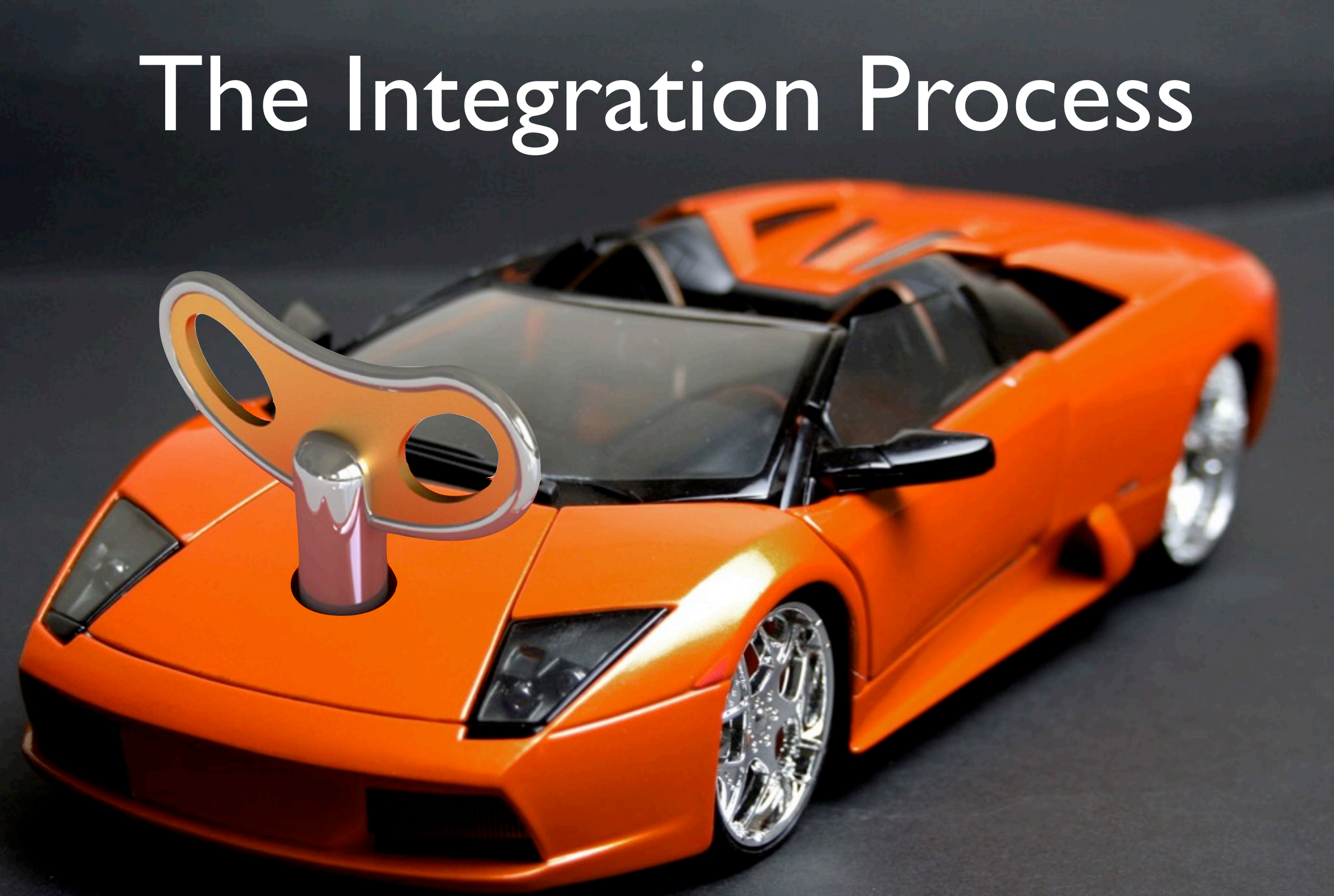
test

build

reduce cycle time!

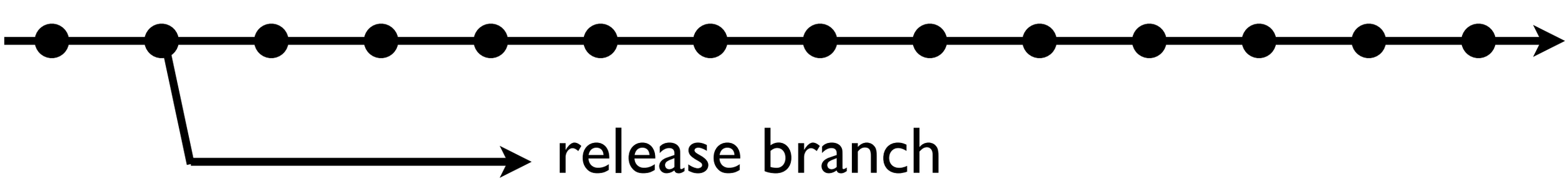
deployment

The Integration Process

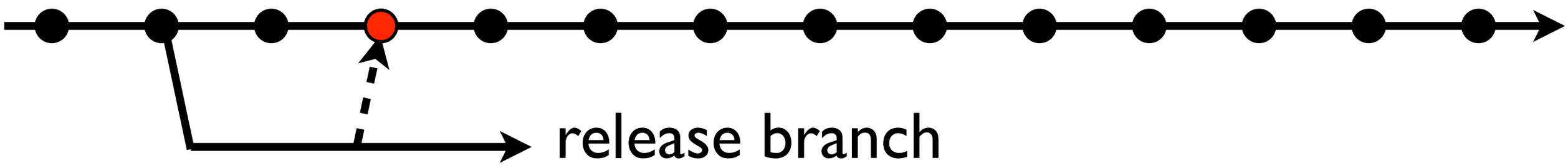


1. The Paradox of Early Integration

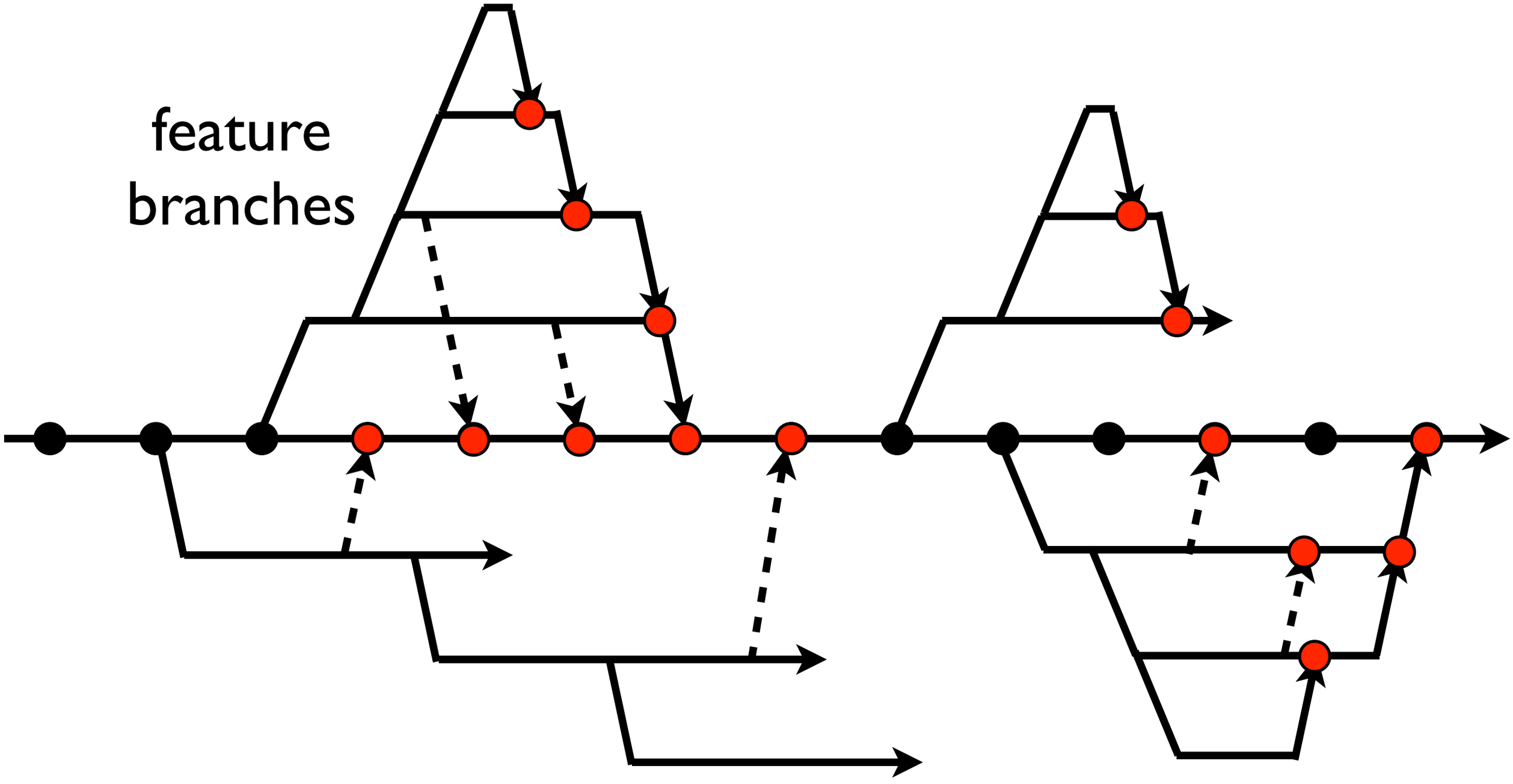
1. The Paradox of Early Integration



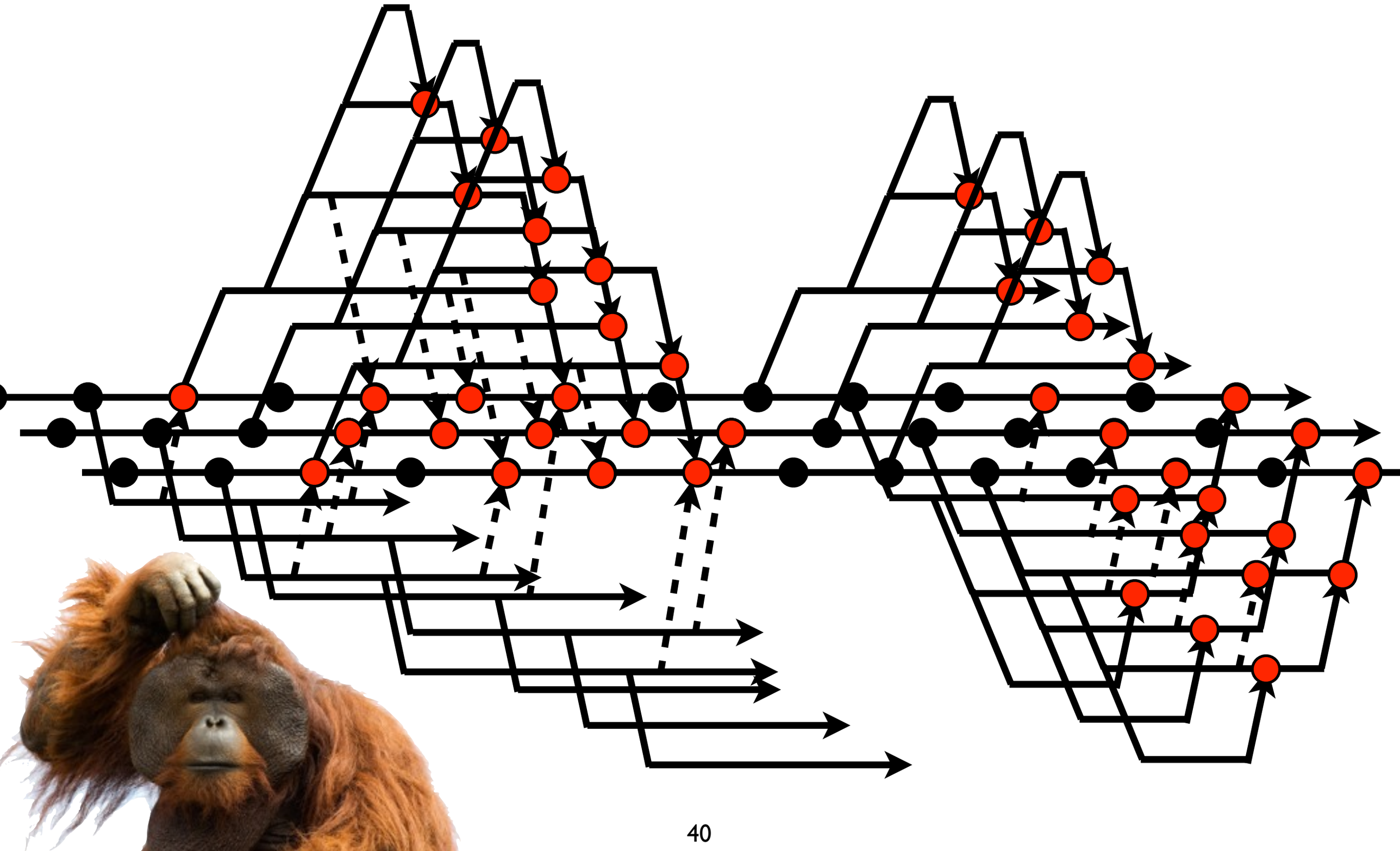
1. The Paradox of Early Integration



1. The Paradox of Early Integration

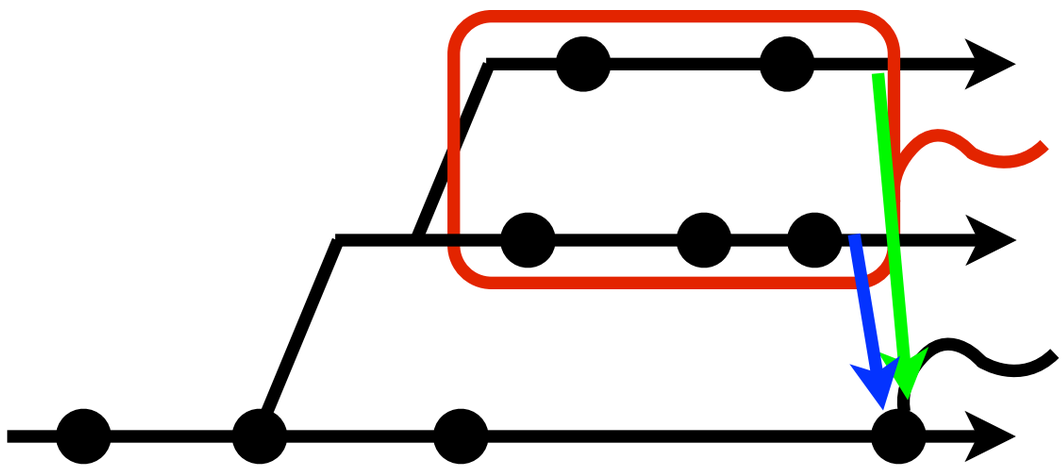


1. The Paradox of Early Integration



Cohesive and Isolated Development with Branches (Barr et al.)

Integrations Interrupt Development once in every 24.4 Commits in DVC



changed files: A, B, C, D and E

changed files: A, E and F

$$\text{distraction on merge} = \frac{|\{A, B, C, D, E\}| \cap |\{A, E, F\}|}{|\{A, E, F\}|} = 66.6\%$$

assumption: Pr[real integration conflict] = 90%

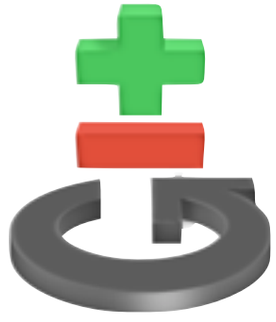
Proactive Detection of Collaboration Conflicts (Brun et al.)

... and when they Do, Integrations cause Trouble

on average **16%** of all merges had textual conflicts (manual resolution needed)

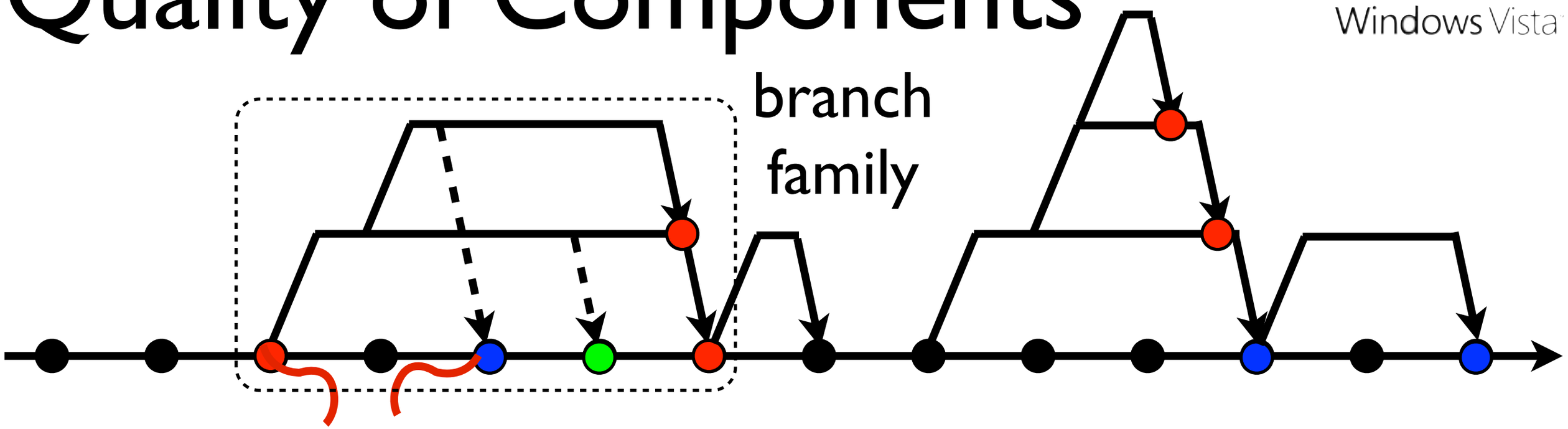
on average **5%** of all merges had build issues

on average **11.7%** of all merges had test issues



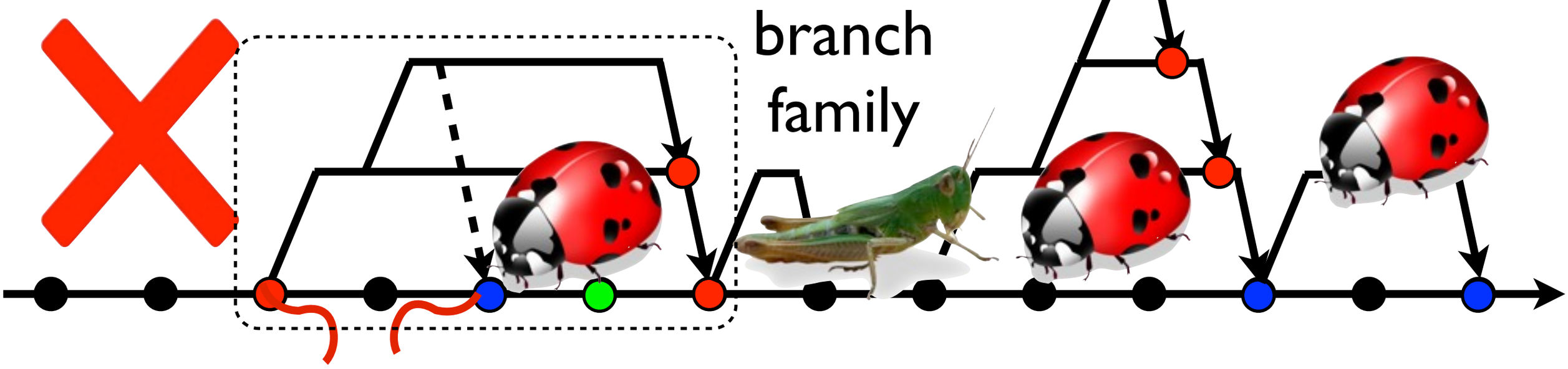
The Effect of Branching Strategies on Software Quality (Shihab et al.)

Branching Degrades Quality of Components



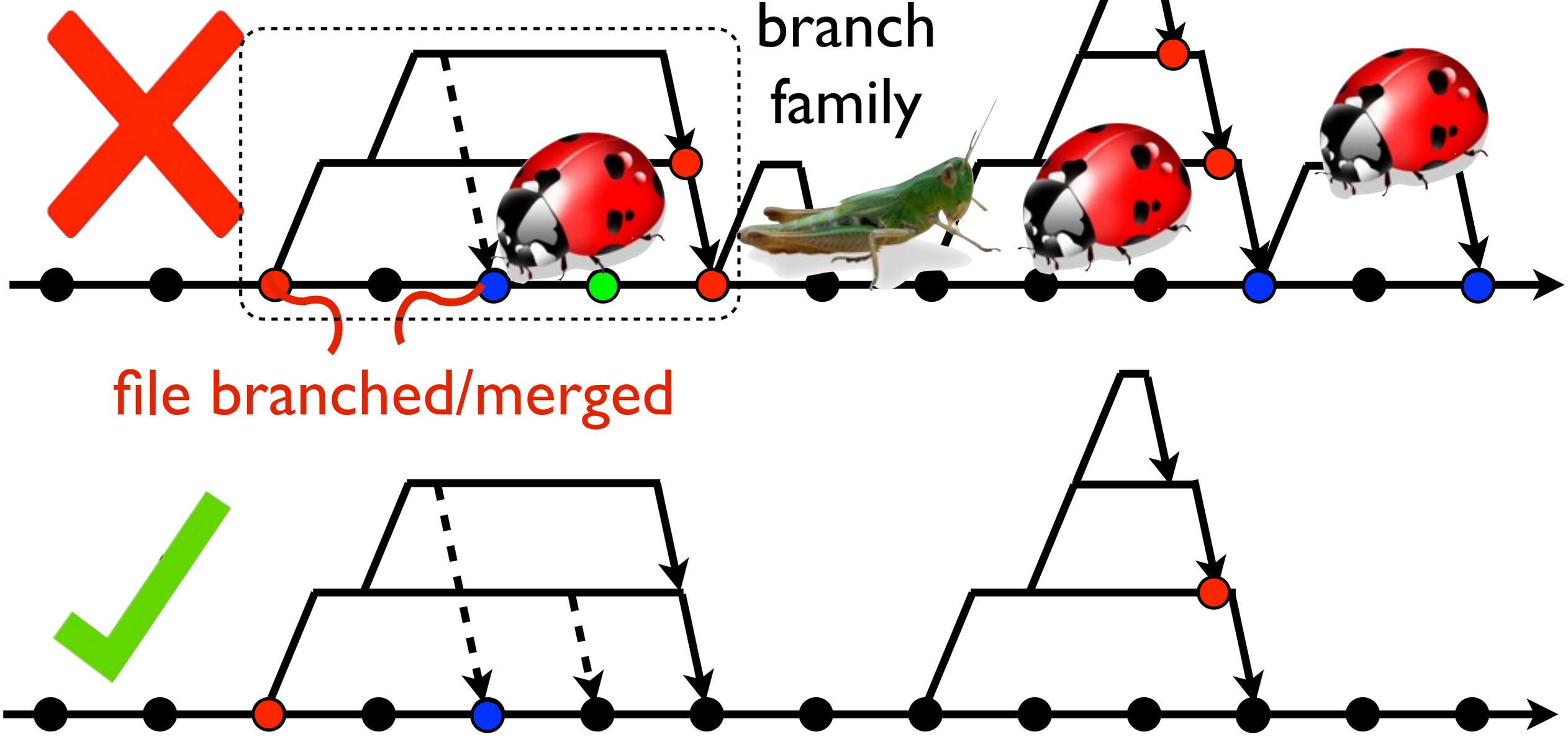
The Effect of Branching Strategies on Software Quality (Shihab et al.)

Branching Degrades Quality of Components



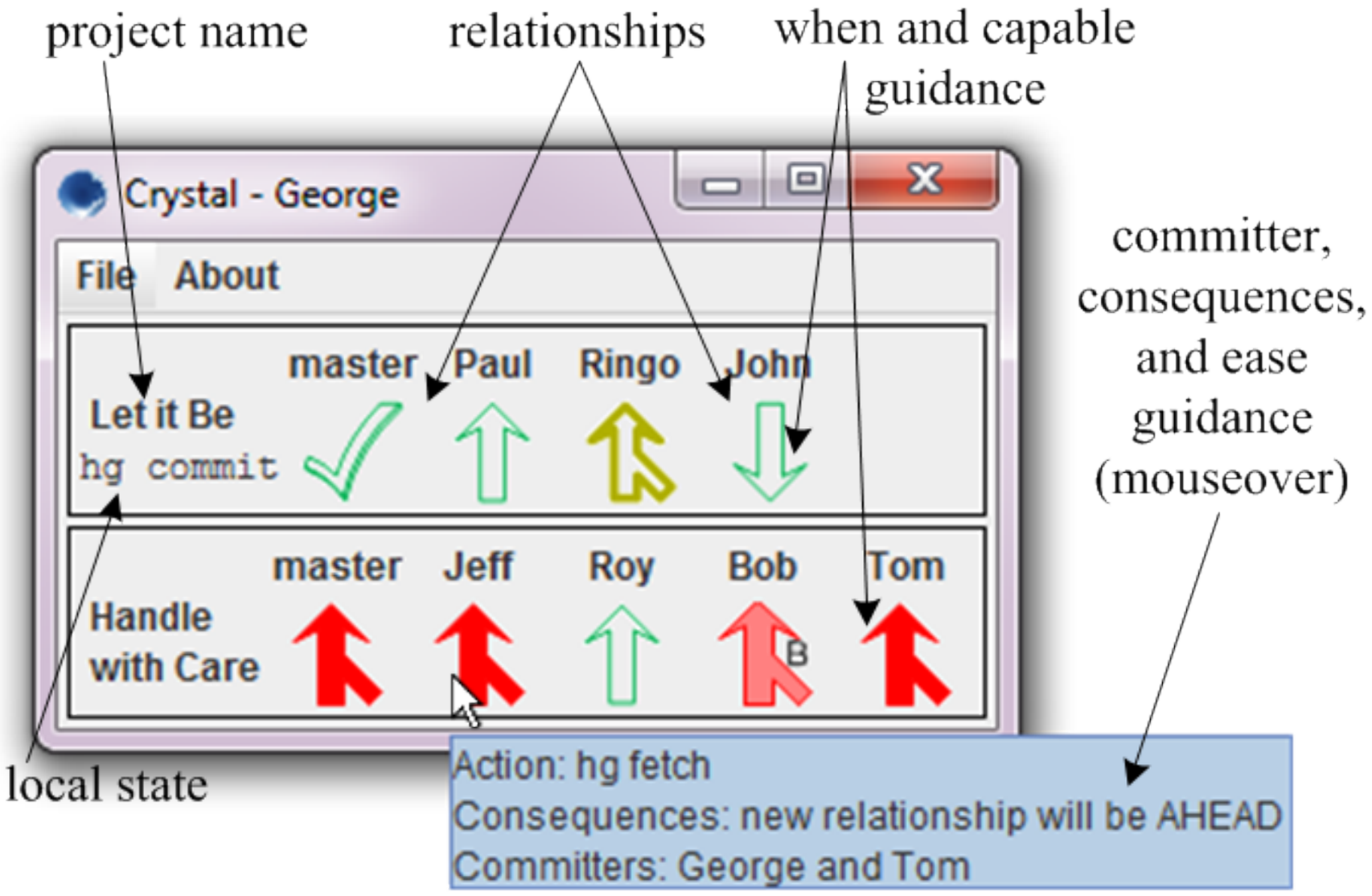
The Effect of Branching Strategies on Software Quality (Shihab et al.)

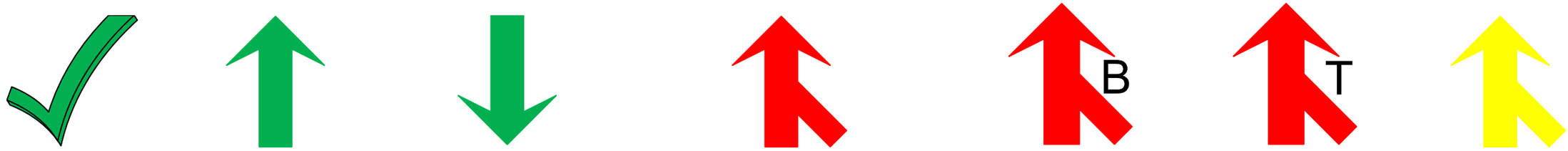
Branching Degrades Quality of Components



Proactive Detection of Collaboration Conflicts (Brun et al.)

Speculative Merge Prevention

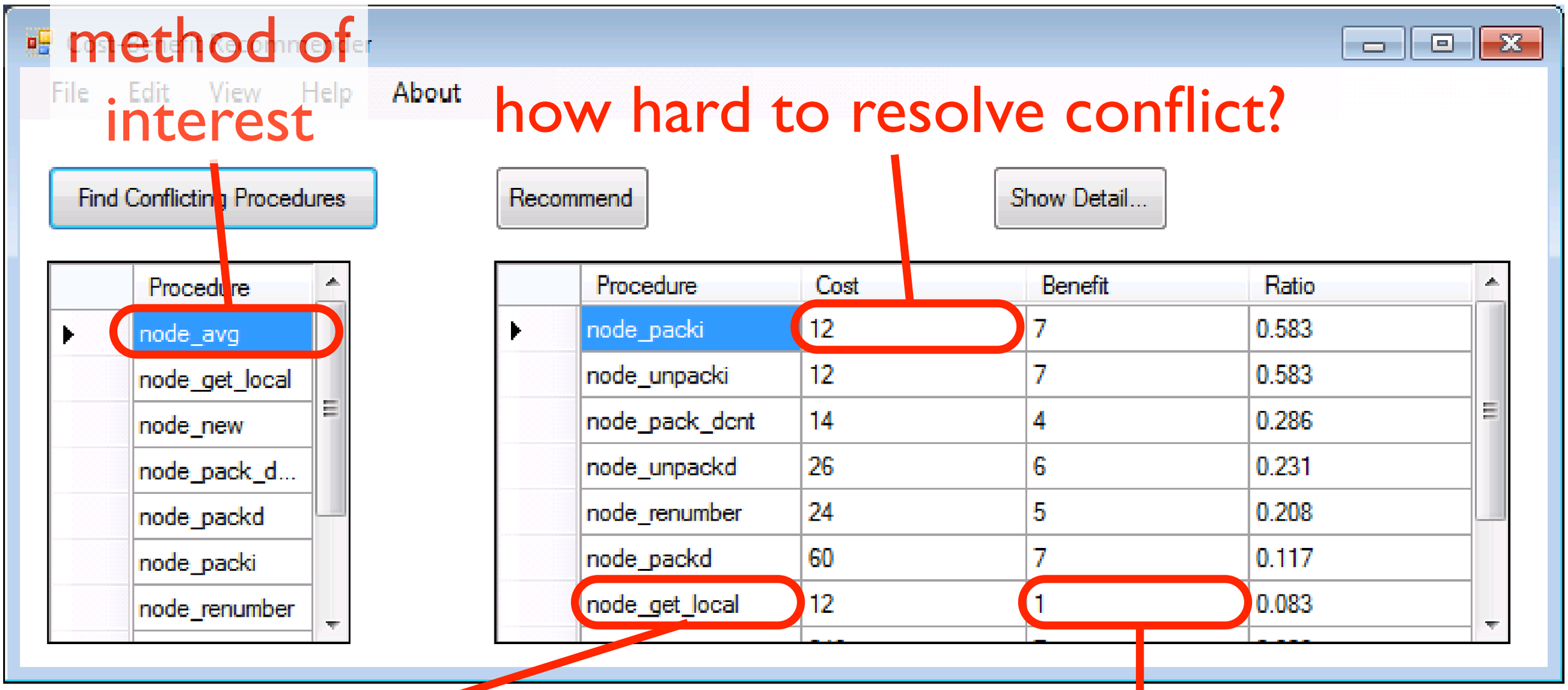




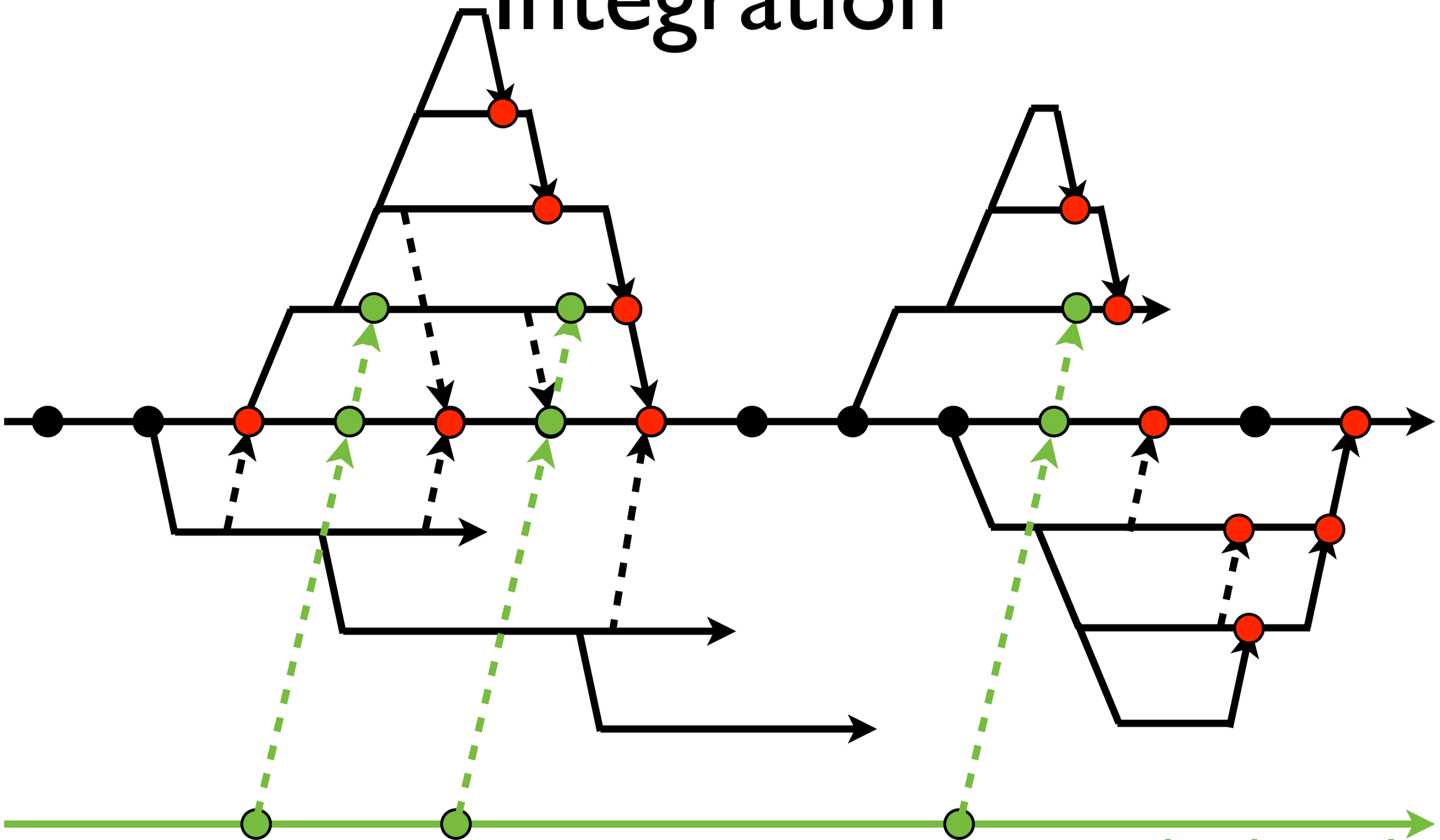
SAME AHEAD BEHIND TEXTUAL✘ BUILD✘ TEST✘ TEST✓

A Cost-Benefit Approach to Recommending Conflict Resolution for Parallel Software Development (Niu et al.)

Conflict Resolution Recommendation



2. The Nightmare of 3rd Party Integration




FreeBSD

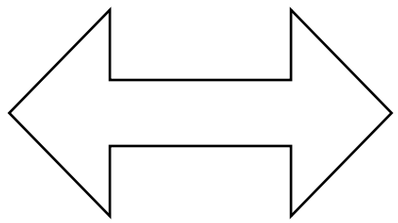

debian


ubuntu

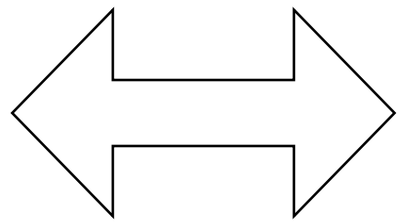
Integrators are Gatekeepers

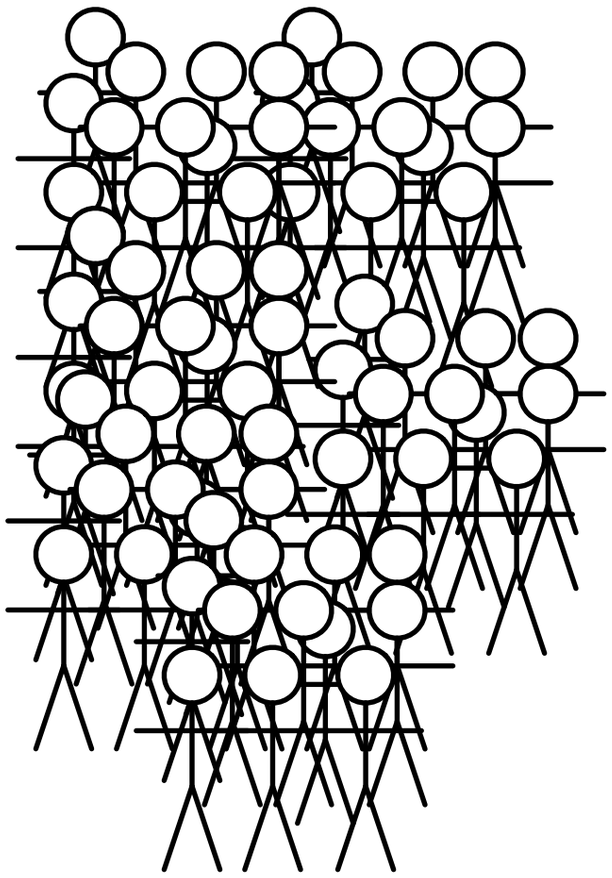
OpenOffice.org

FreeBSD



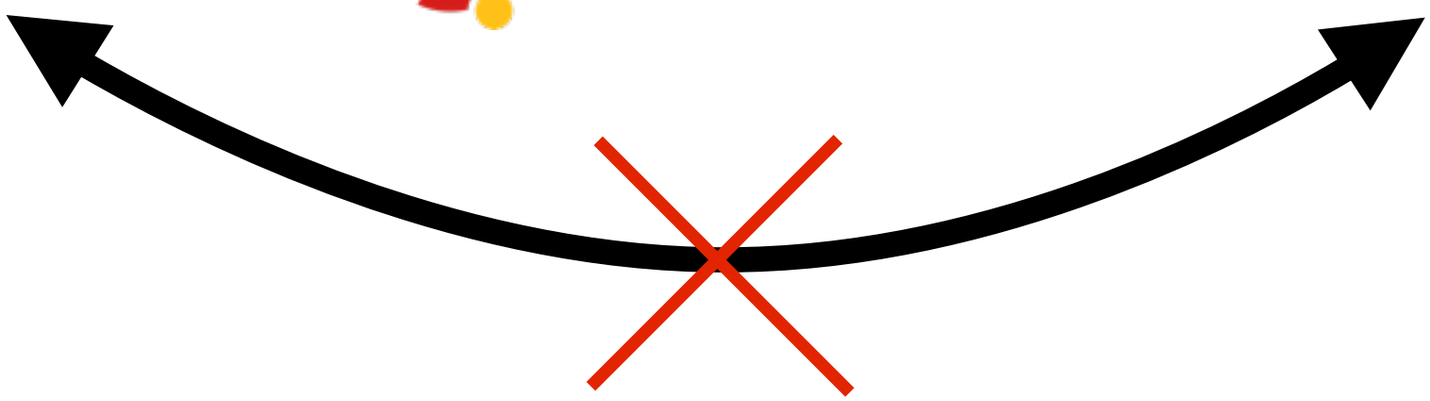
debian





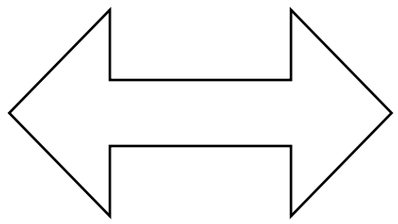
MySQL

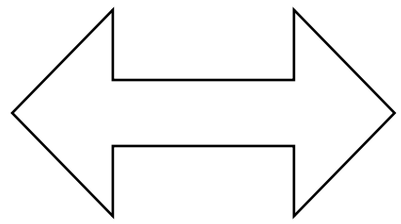
ubuntu

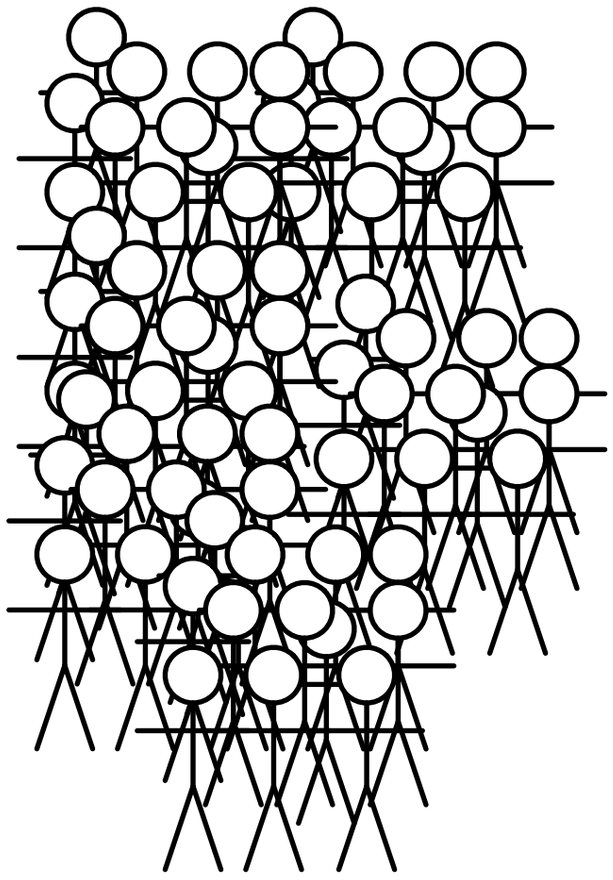


end users

Integrators are Gatekeepers

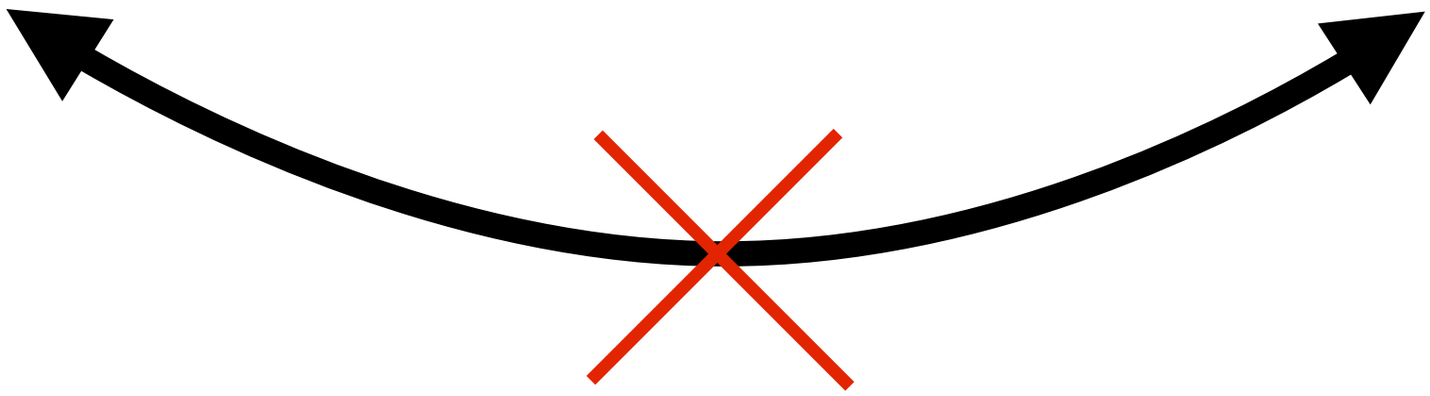






maintainers

end users

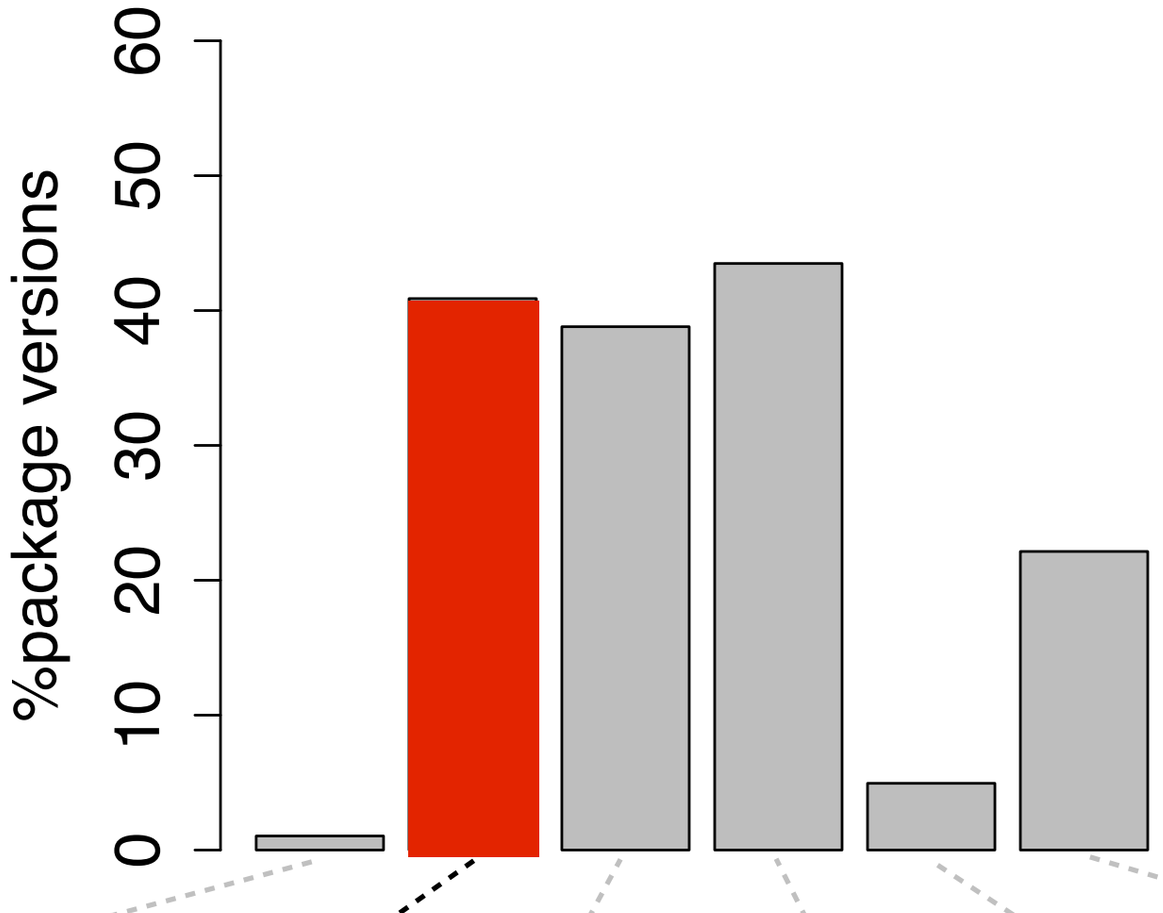


>180k Packages & ~28 Years of History

debian

FreeBSD®

ubuntu



New Package

Local Patch

Upstream Sync

Dependency Management

Build Change

Product-wide Concern

Upstream Sync

Mozilla Delivers **New Version** of Firefox – First Web Browser to Support Do Not Track on Multiple Platforms! [Mozilla Blog]

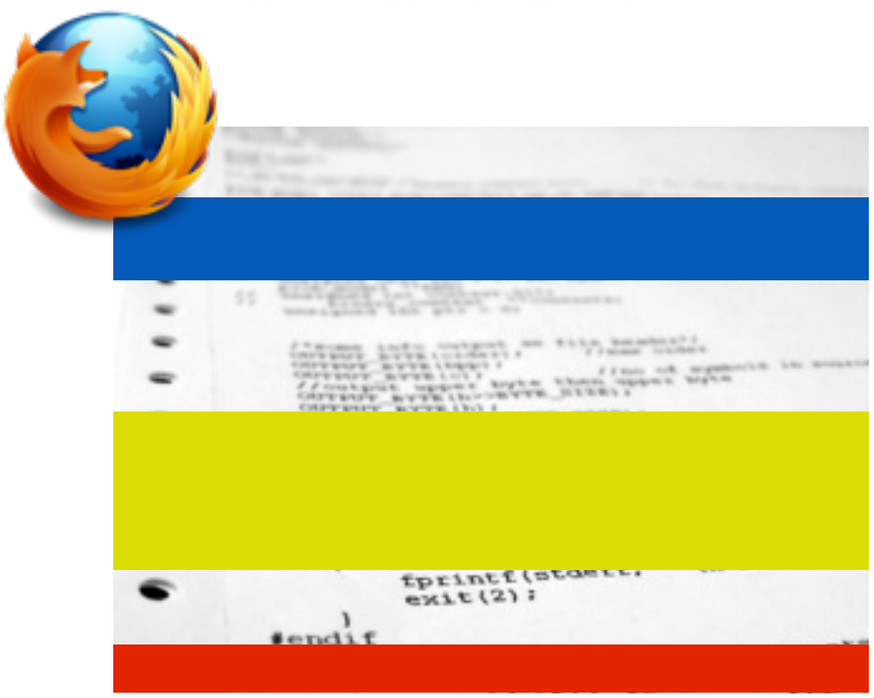
Necessary?

Buggy?

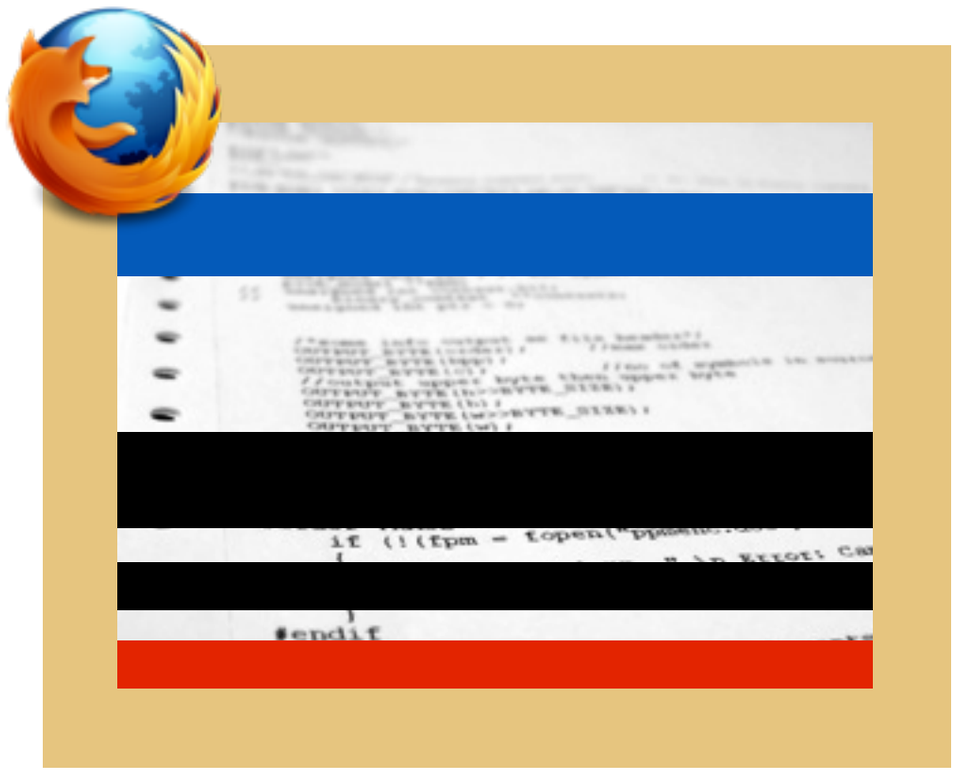
Secure?

Risk Analysis & Cherry-picking

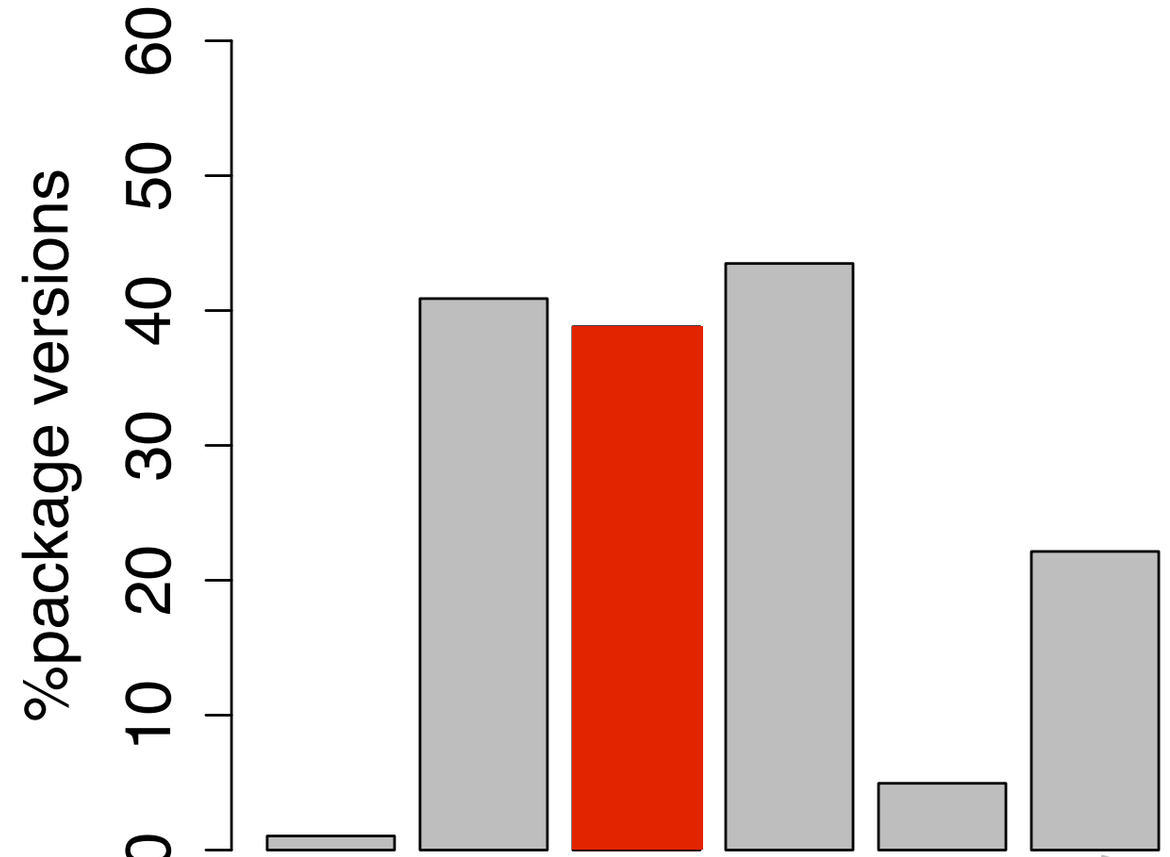
UNIX diff



maintainer



Upstream Sync



New Package

Local Patch

Upstream Sync

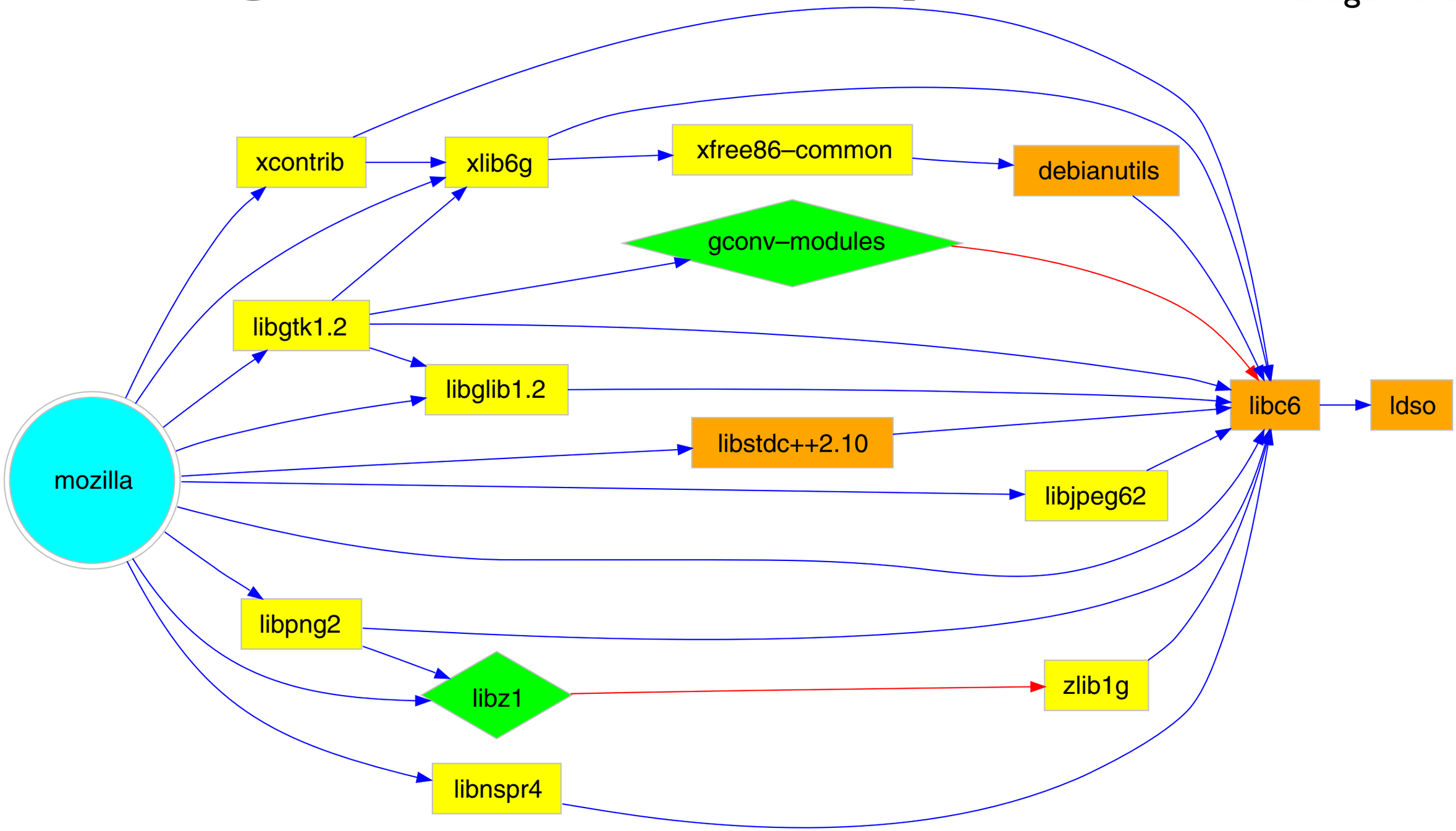
Dependency Management

Build Change

Product-wide Concern

Library Transitions Ripple through the whole System

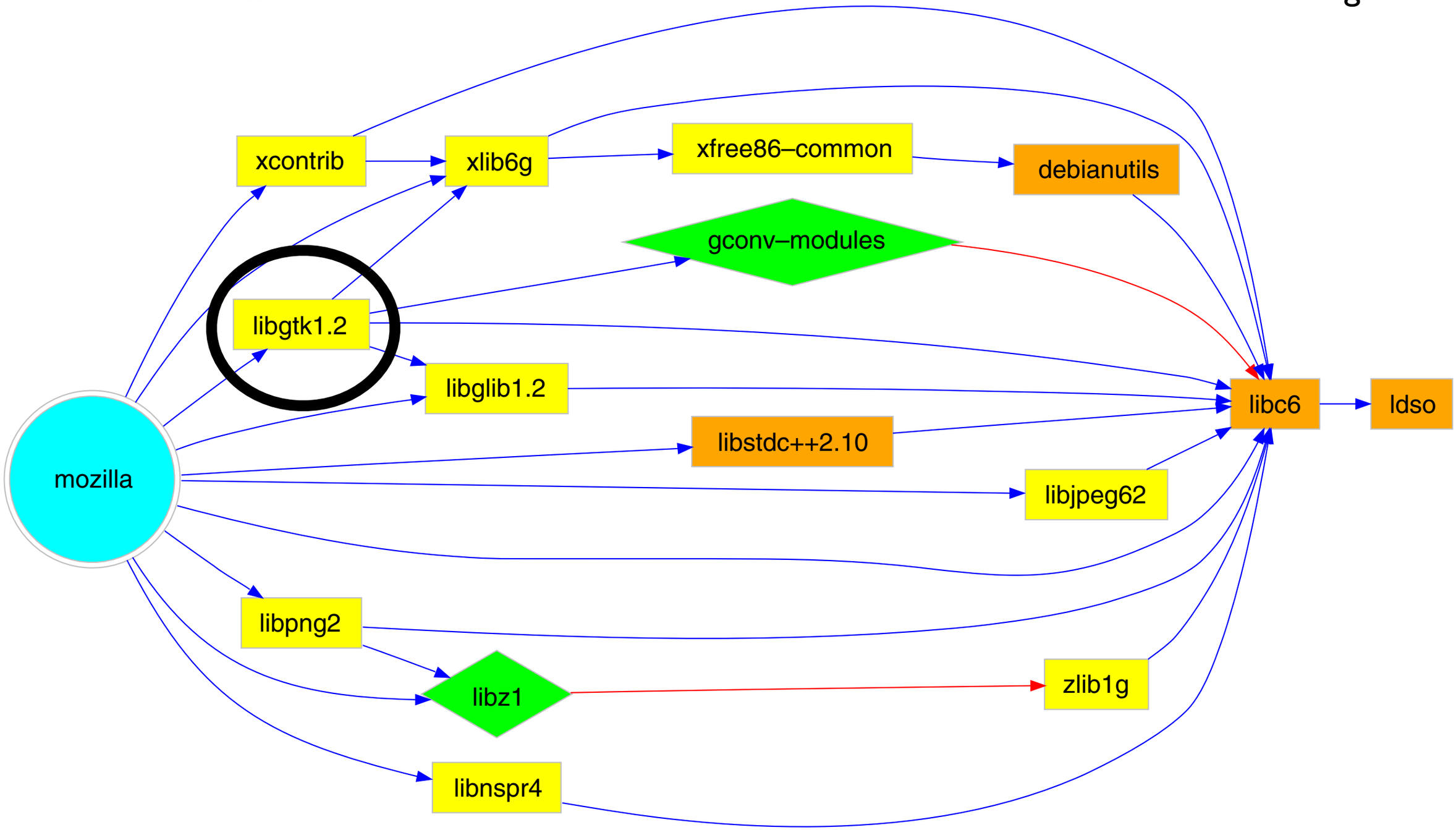
Dependency Management



Macro-level software evolution: a case study of a large software compilation (Gonzalez-Barahona et al.)

Library Transitions Ripple through the whole System

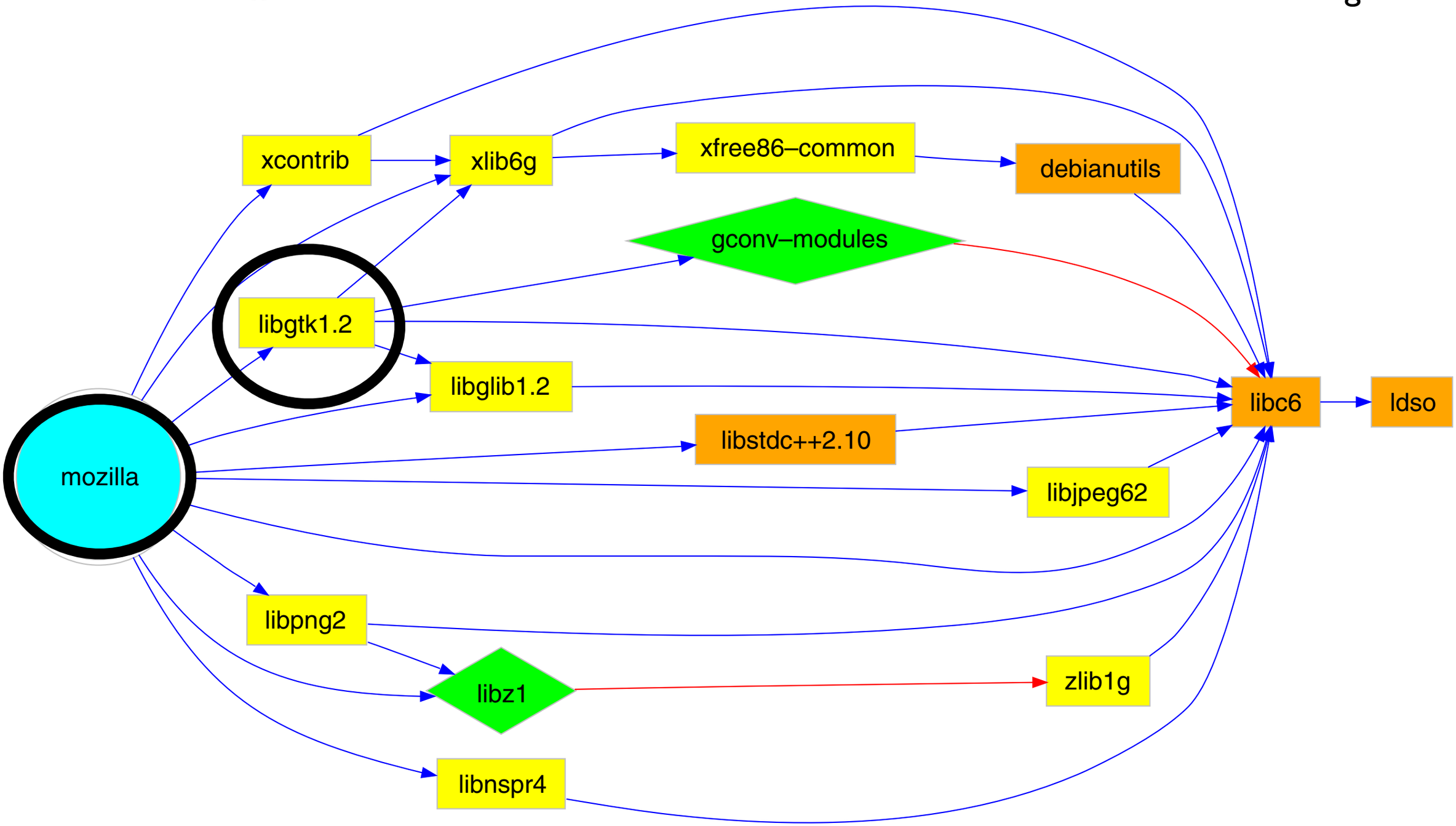
Dependency Management



Macro-level software evolution: a case study of a large software compilation (Gonzalez-Barahona et al.)

Library Transitions Ripple through the whole System

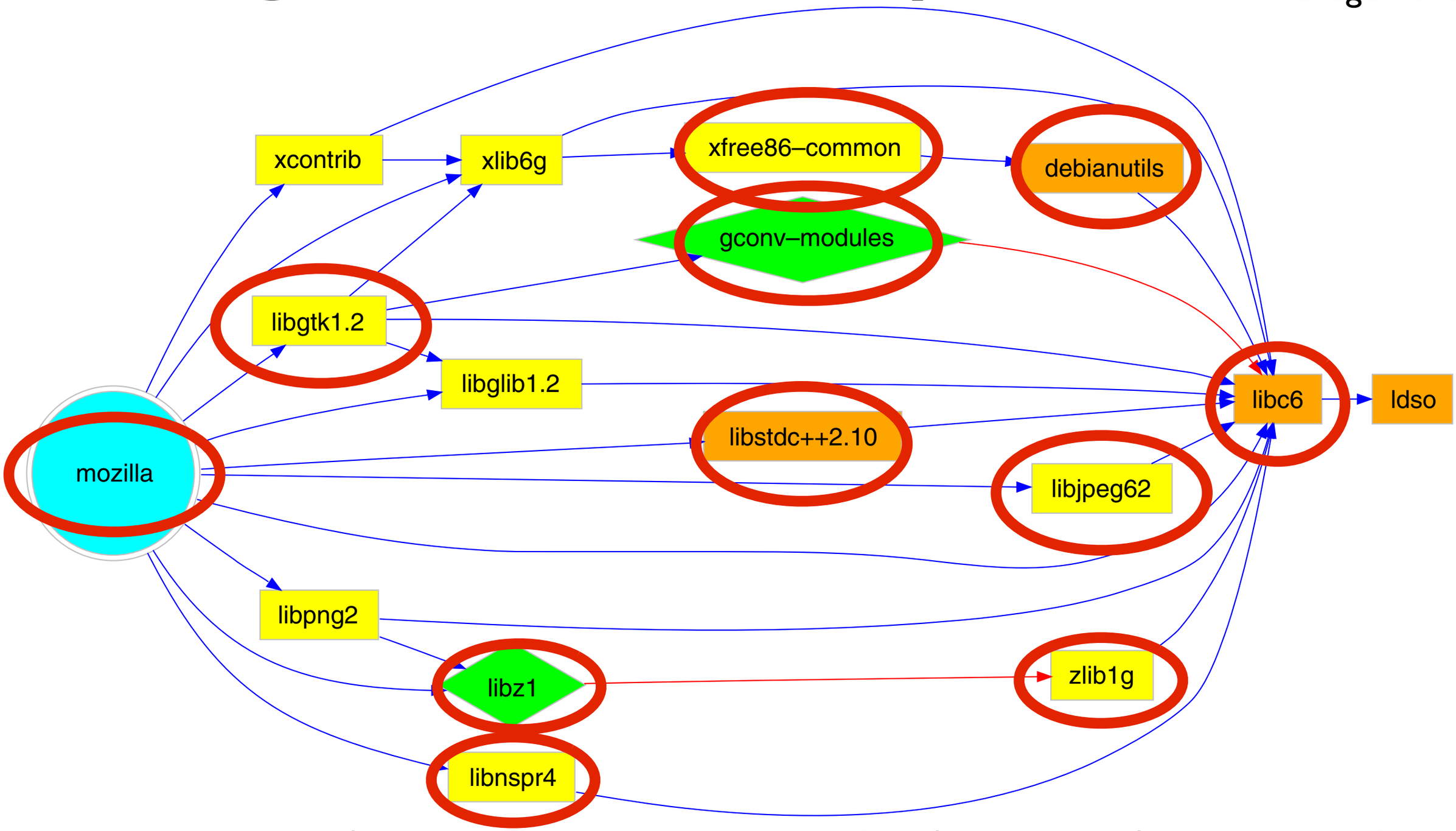
Dependency Management



Macro-level software evolution: a case study of a large software compilation (Gonzalez-Barahona et al.)

Library Transitions Ripple through the whole System

Dependency Management



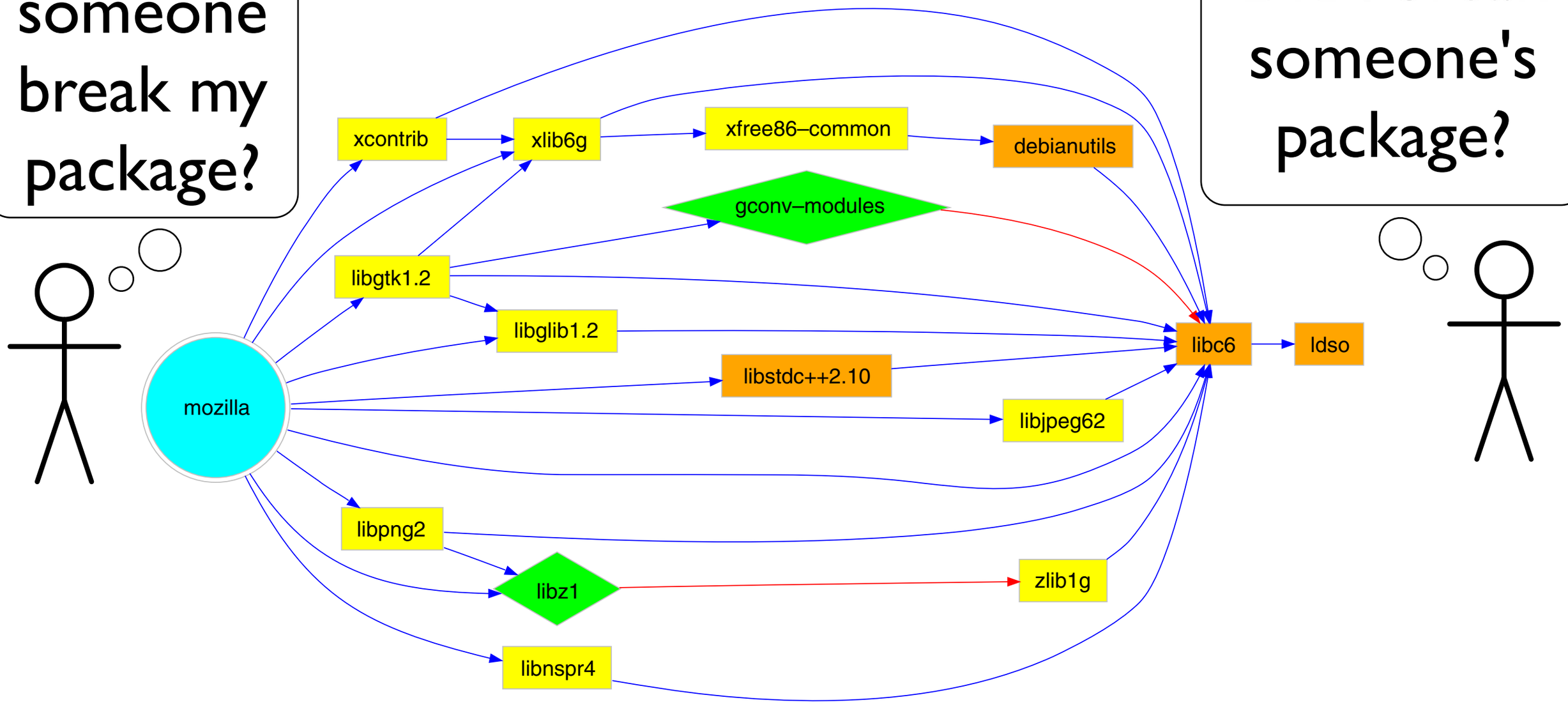
Macro-level software evolution: a case study of a large software compilation (Gonzalez-Barahona et al.)

2-way Impact Analysis

Dependency Management

Did someone break my package?

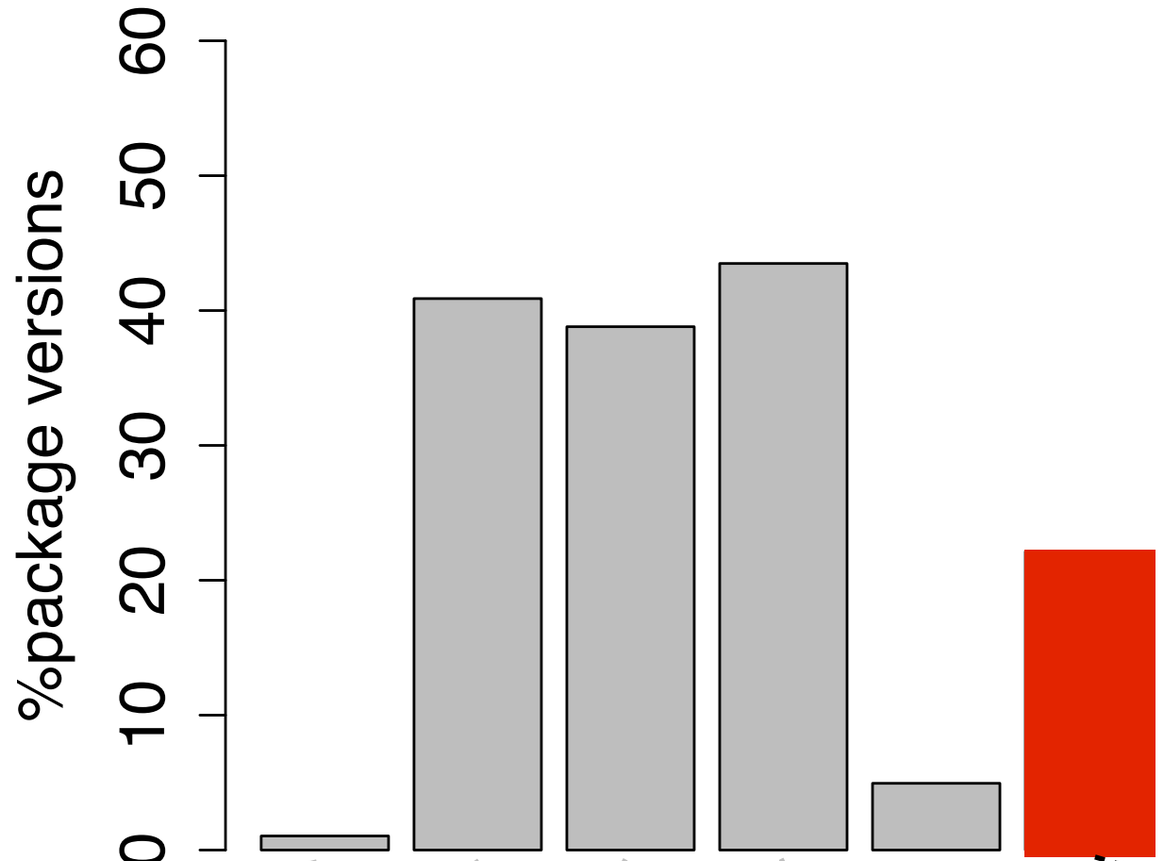
Did I break someone's package?



If an engineer comes up with a better way of doing something, he is tasked with refactoring all existing libraries and **assisting dependent projects to migrate** to the new libraries

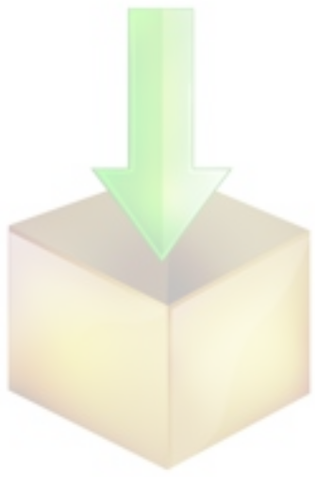
James Whittaker

debian



New Package

Local Patch

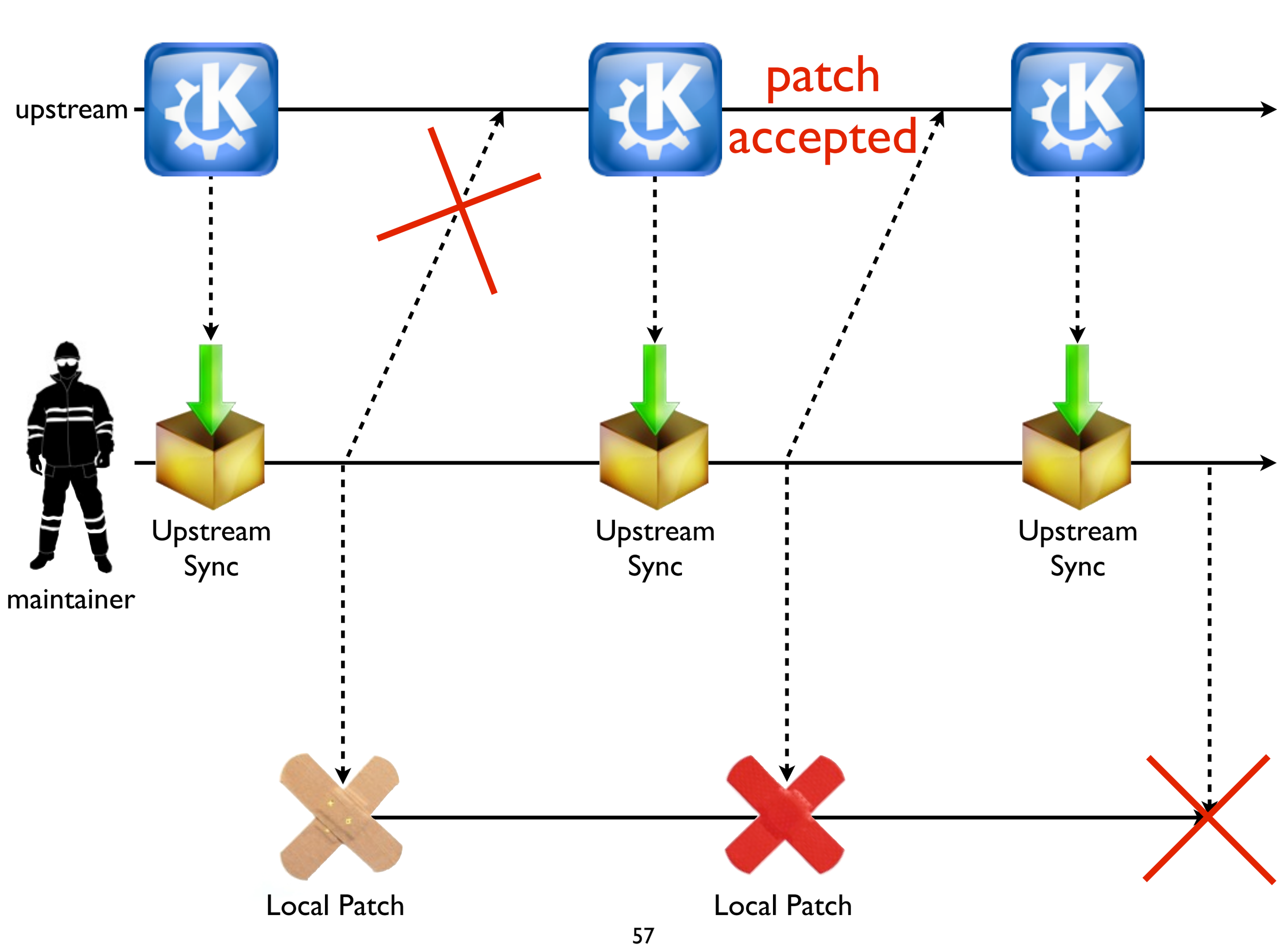


Upstream Sync

Dependency Management

Build Change

Product-wide Concern


upstream
patch
accepted
maintainer
Upstream
Sync
Upstream
Sync
Upstream
Sync
Local Patch
Local Patch

Local Patch

```
Package          : openssl
Vulnerability    : predictable random number generator
Problem type     : remote
Debian-specific: yes
CVE Id(s)        : CVE-2008-0166
Date             : 2008-05-13

Luciano Bello discovered that the random number generator in Debian's
openssl package is predictable. This is caused by an incorrect
Debian-specific change to the openssl package (CVE-2008-0166). As a
result, cryptographic key mater[illegible] guessable.

It is strongly recommended that all cryptographic key material which
has been generated by OpenSSL versions starting with 0.9.8c-1 on
Debian systems is recreated from scratch. Furthermore, all DSA keys
ever used on affected Debian systems for signing or authentication
purposes should be considered compromised; the Digital Signature
Algorithm relies on a secret random value use[illegible] signature
generation.

The first vulnerable version, 0.9.8c-1, was uploaded to the unstable
distribution on 2006-09-17, and has since propagated to the testing
and current stable (etch) distributions. The old stable distribution
(sarge) is not affected.
```

>=1.5 years 8-(

>=44 distributions ;-)

http://behrns.files.wordpress.com/2008/03/ikea-car.jpg

Release Engineering

in-house/3rd party development

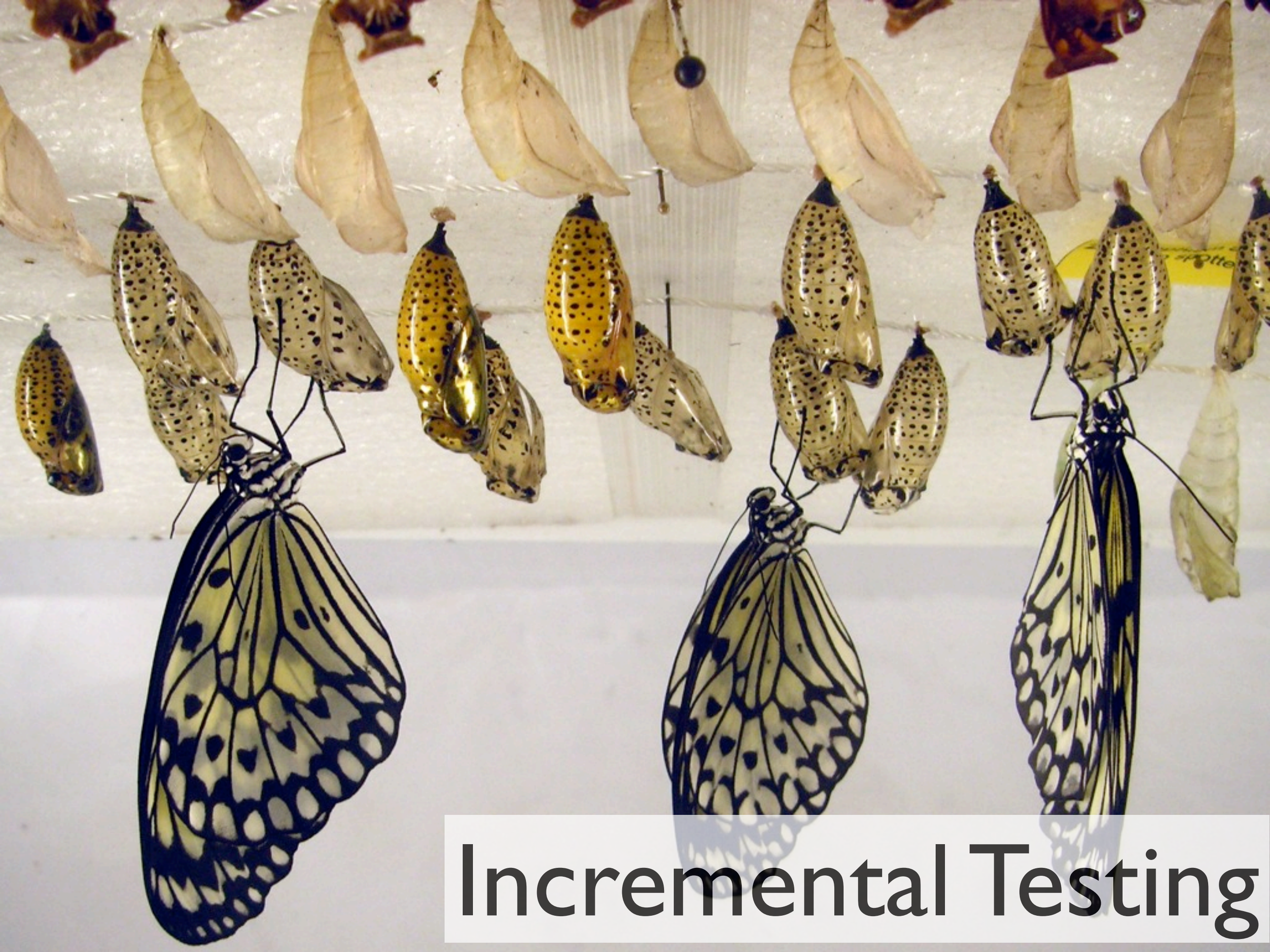
integrate

test

build

reduce cycle time!

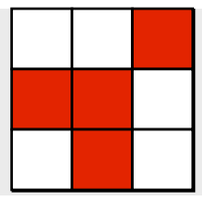
deployment



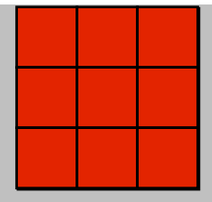
Incremental Testing

Different Test Phases

(selection of) unit tests

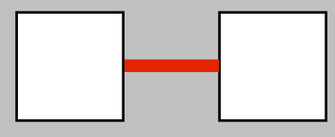


integration + build

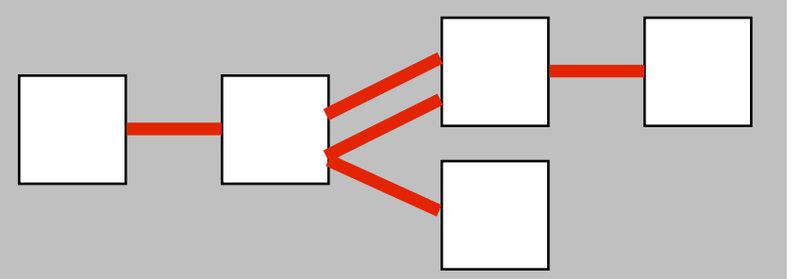


all unit tests

acceptance/integration tests

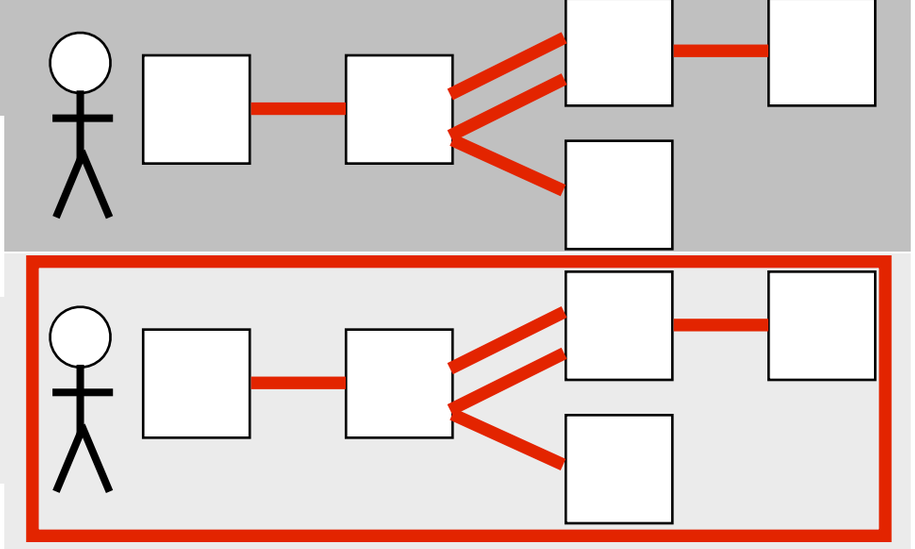


capacity tests



user acceptance tests

release

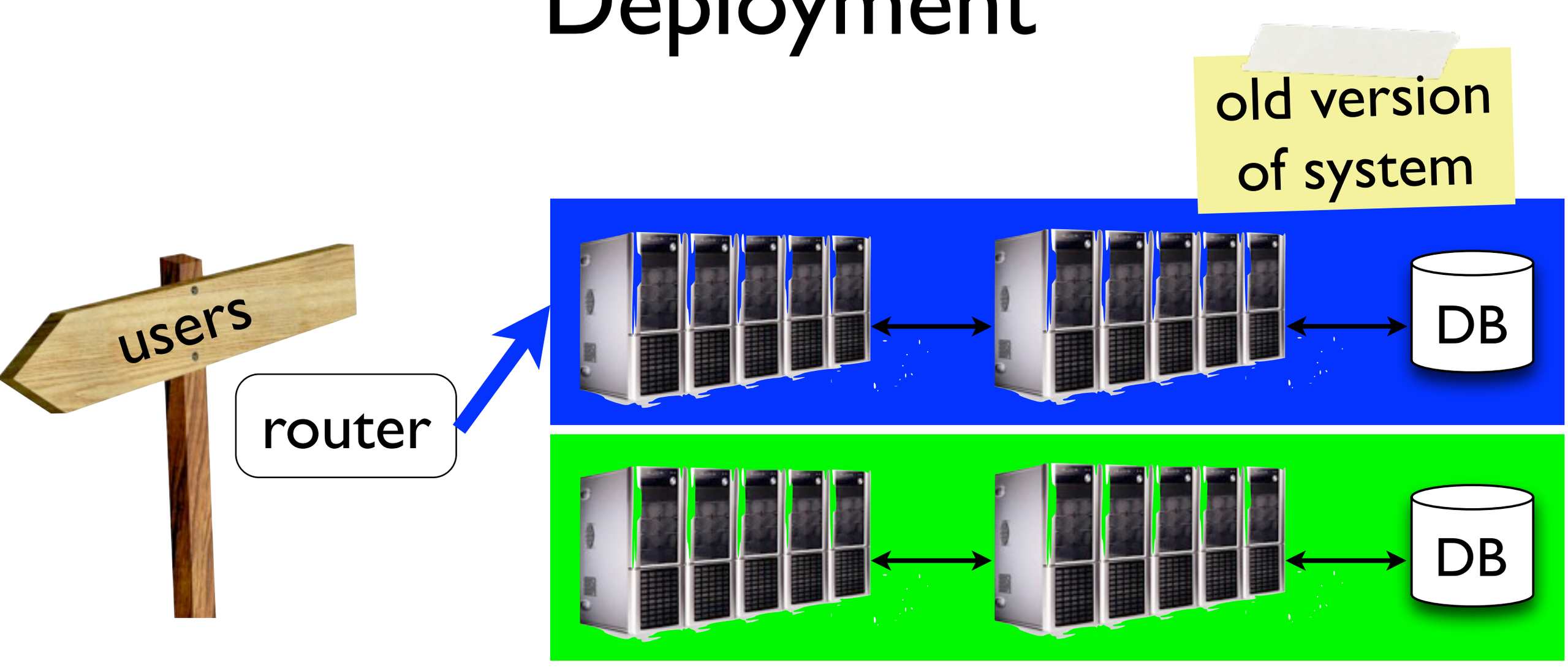


smoke tests

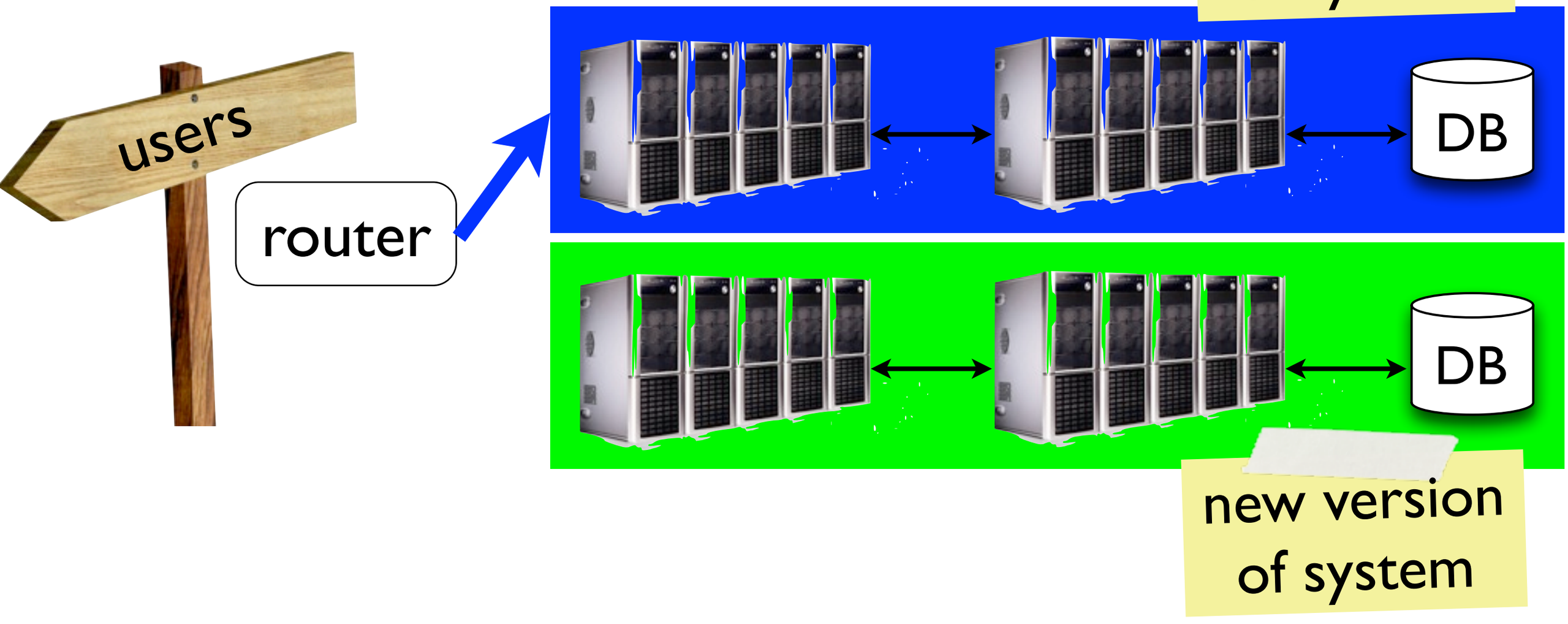
test phase

time to run (hours)

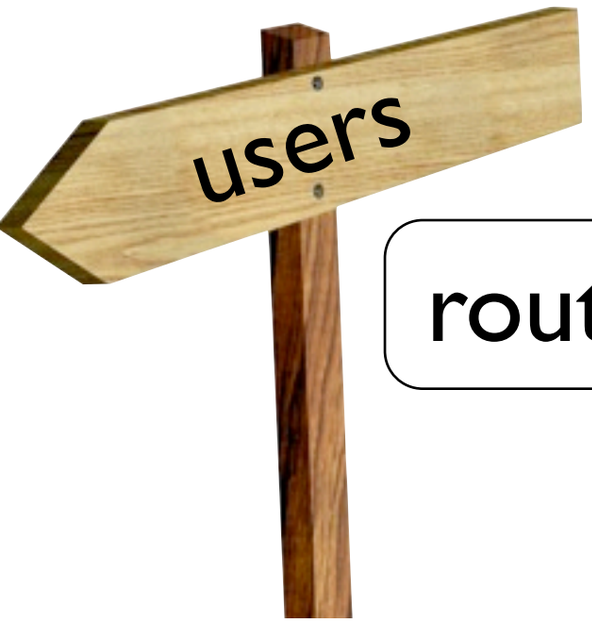
Seamless Upgrades (1): Blue/Green Deployment



Seamless Upgrades (1): Blue/Green Deployment



Seamless Upgrades (1): Blue/Green Deployment

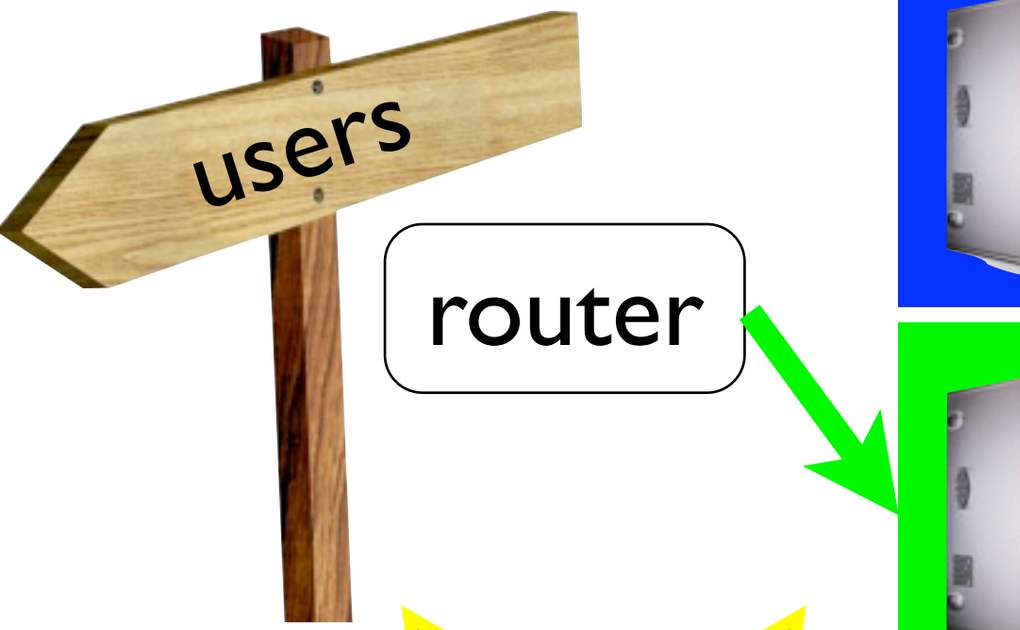


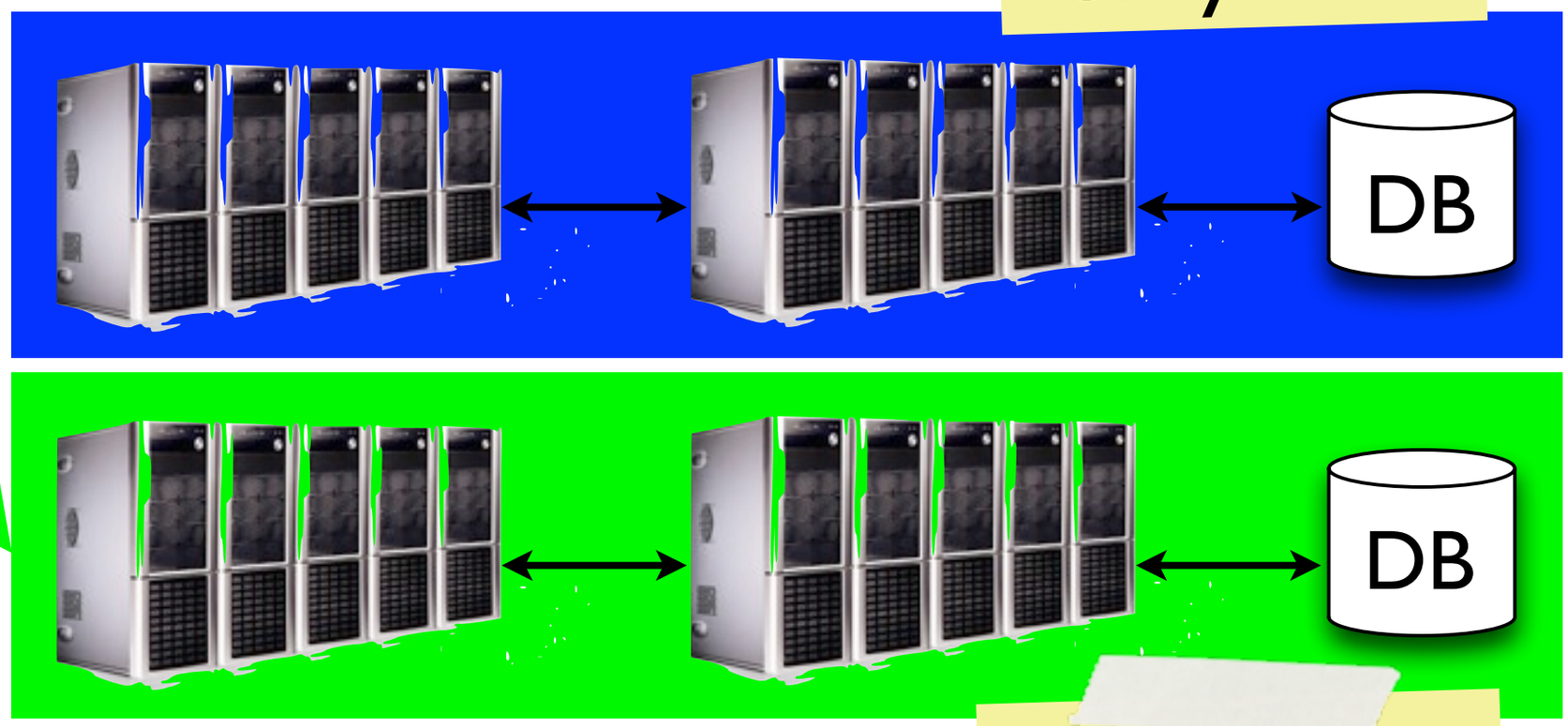
router

old version of system

new version of system

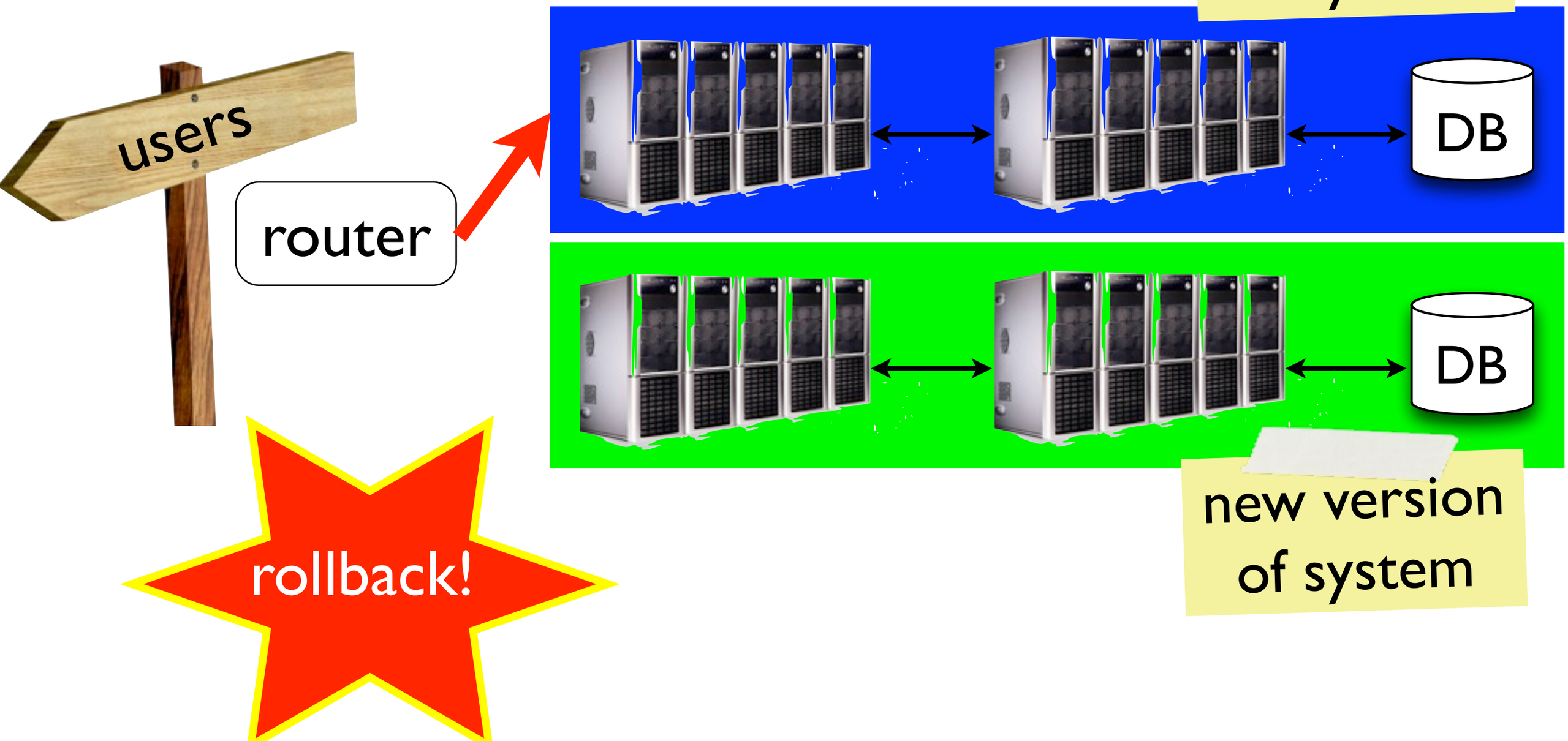
Seamless Upgrades (1): Blue/Green Deployment



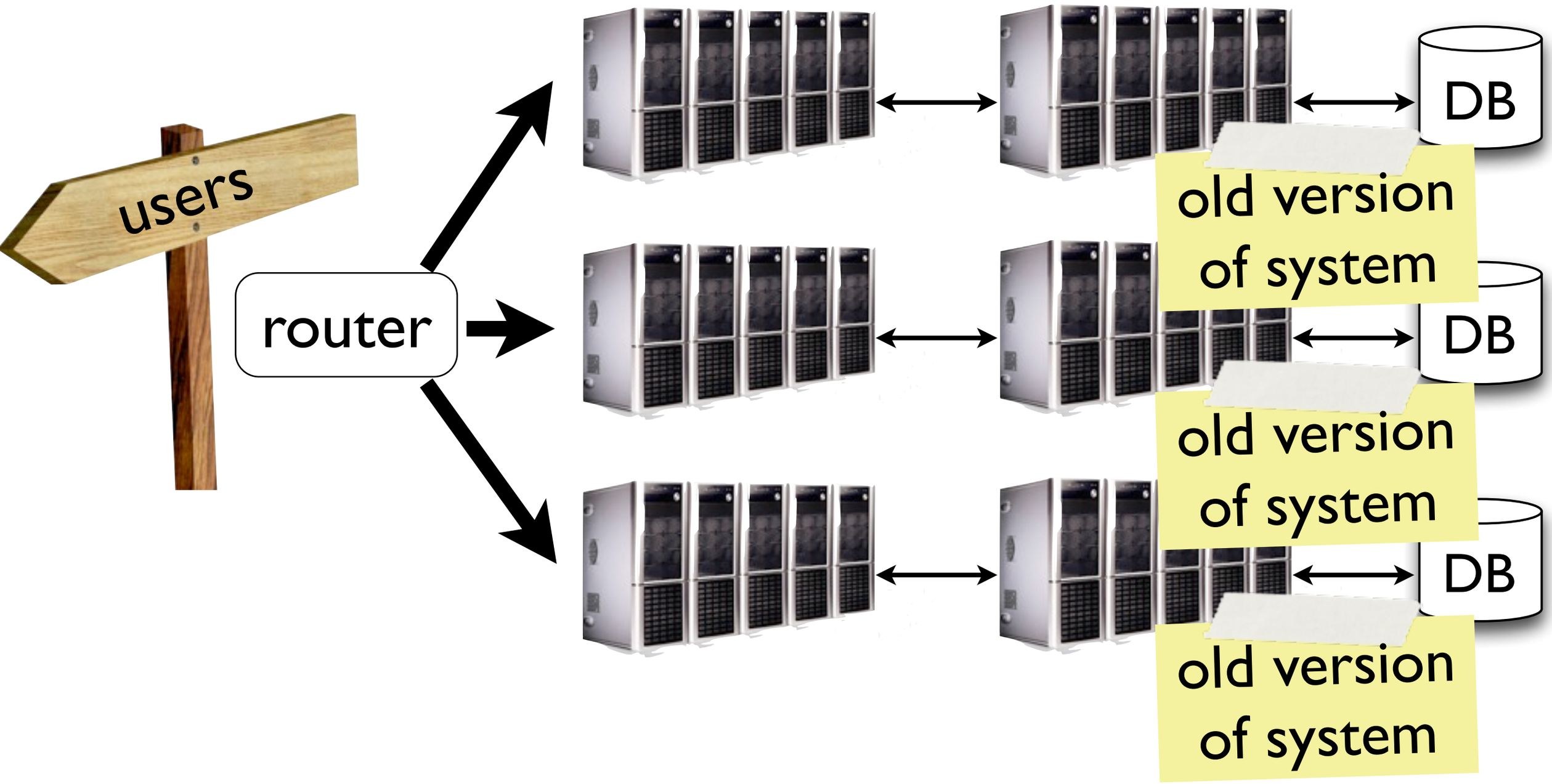


rollback!

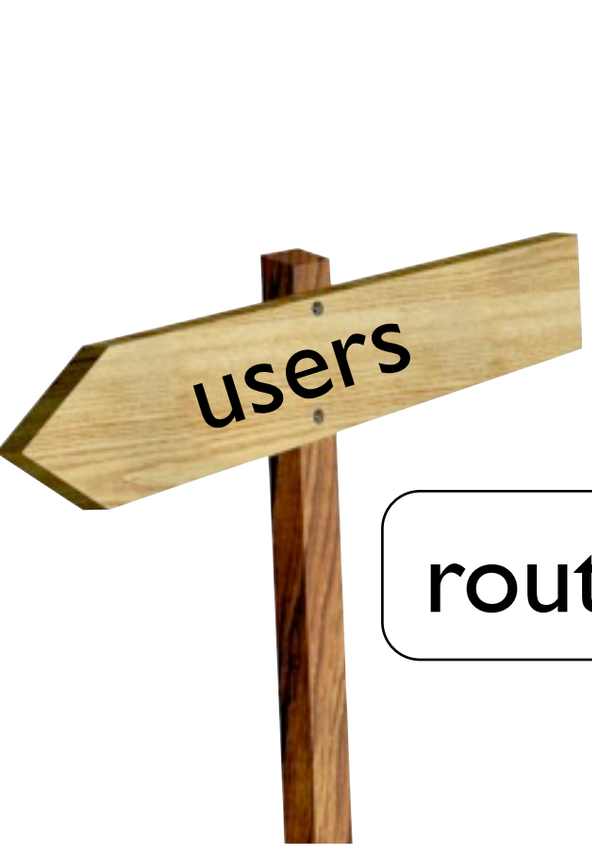
Seamless Upgrades (1): Blue/Green Deployment



Seamless Upgrades (2): Canary Deployment

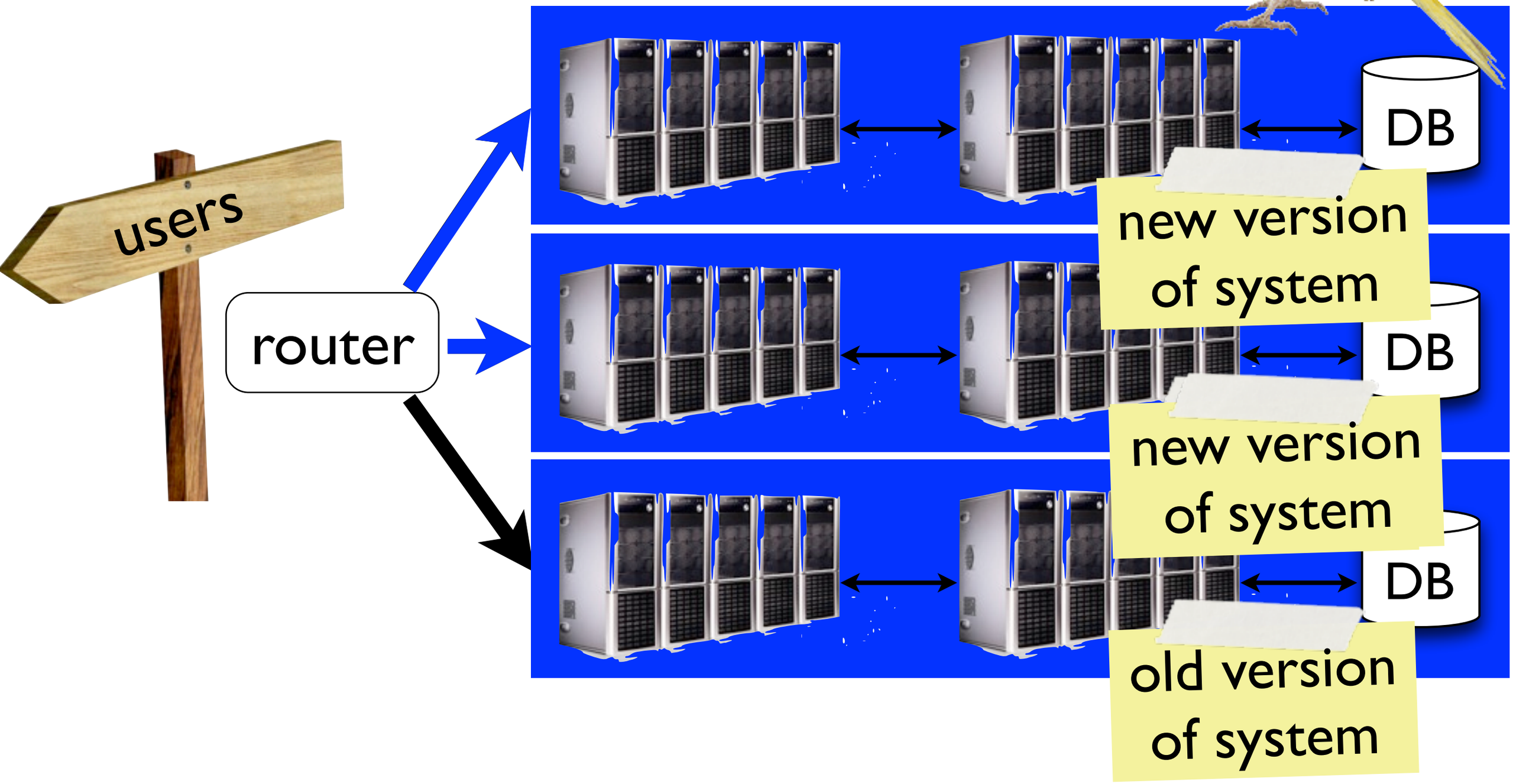


Seamless Upgrades (2): Canary Deployment

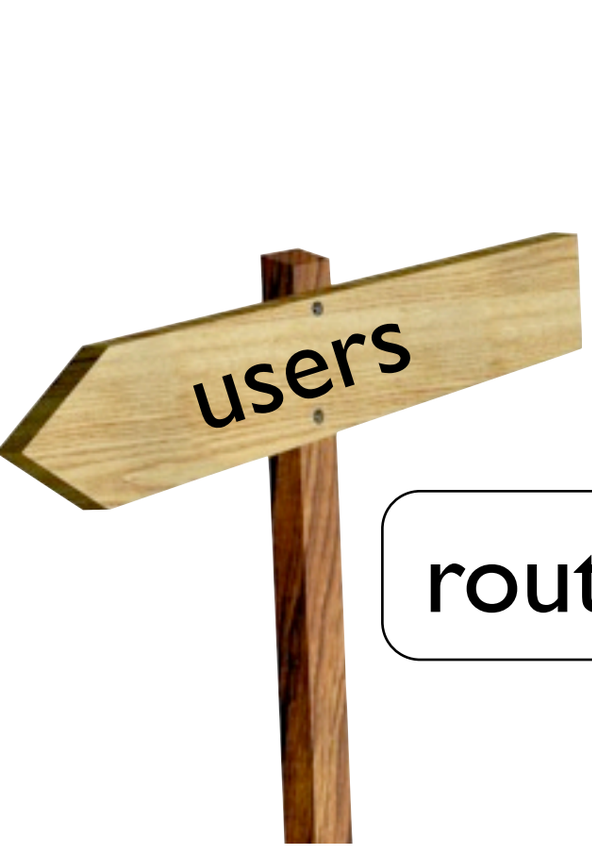


router

Seamless Upgrades (2): Canary Deployment



Seamless Upgrades (2): Canary Deployment

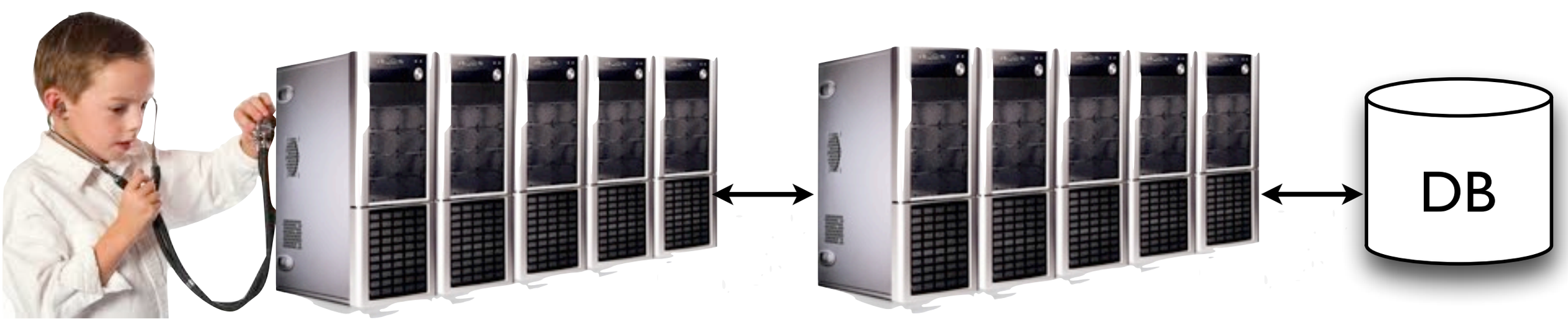


router

Testing Challenge 1: Parallelization

Testing Challenge 2: Incrementalizing Tests

Testing Challenge 3: Optimizing Test Speed



Testing Challenge 4: Data Migration

Testing Challenge 5: Process Integration

Open Challenges for Release Engineering

rapid release

?

=

Do Faster Releases Improve Software Quality? - An Empirical Case Study of Mozilla Firefox (Khomh et al.)

Firefox®

Traditional Release Cycle

Rapid Release Cycle

1.0 1.5 2.0 3.0 3.5 3.6 4.0 5.0 6.0 7.0 8.0 9.0

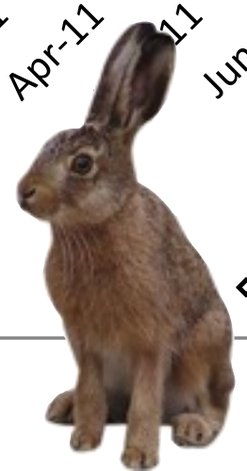
Jun-04 Dec-04 Jun-05 Dec-05 Jun-06 Dec-06 Jun-07 Dec-07 Jun-08 Dec-08 Jun-09 Dec-09 Jun-10 Dec-10 Jun-11 Dec-11

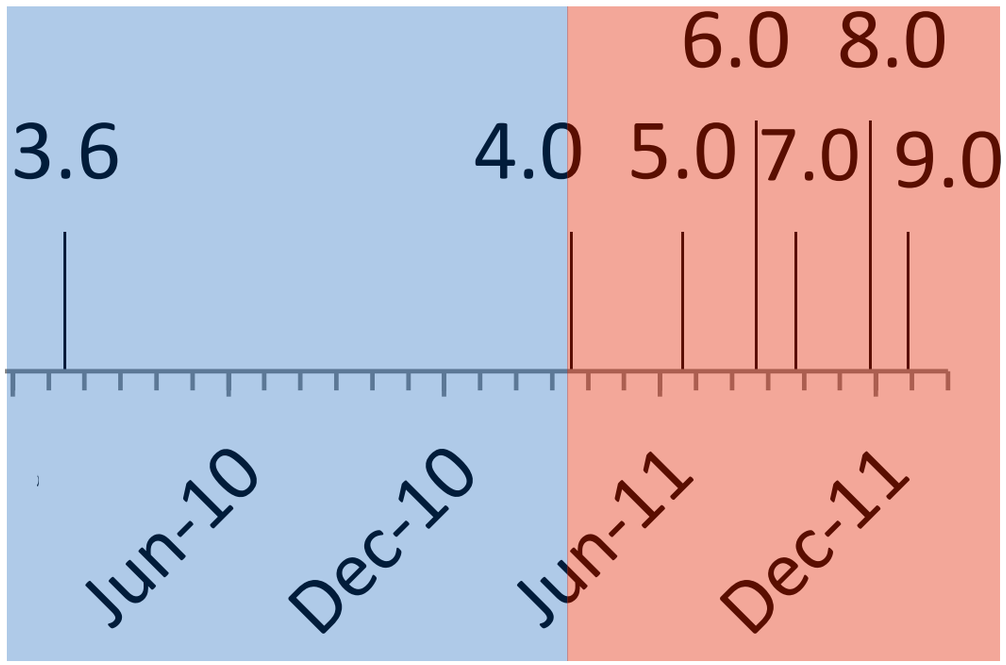
3.6 3.6.2 3.6.3 3.6.4 3.6.6 3.6.7 3.6.8 3.6.9 3.6.10 3.6.11 3.6.12 3.6.13 3.6.14 **4.0** 3.6.15 3.6.16 3.6.17 4.0.1 3.6.18 3.6.19 3.6.20 3.6.21 3.6.22 3.6.23 3.6.24

Dec-09 Jan-10 Feb-10 Mar-10 Apr-10 May-10 Jun-10 Jul-10 Aug-10 Sep-10 Oct-10 Nov-10 Dec-10 Jan-11 Feb-11 Mar-11 Apr-11 Jun-11 Jul-11 Aug-11 Sep-11 Oct-11 Nov-11 Dec-11

5.0.1 6.0.1 6.0.2 7.0.1 8.0.1

5.0 **6.0** **7.0** **8.0** **9.0**



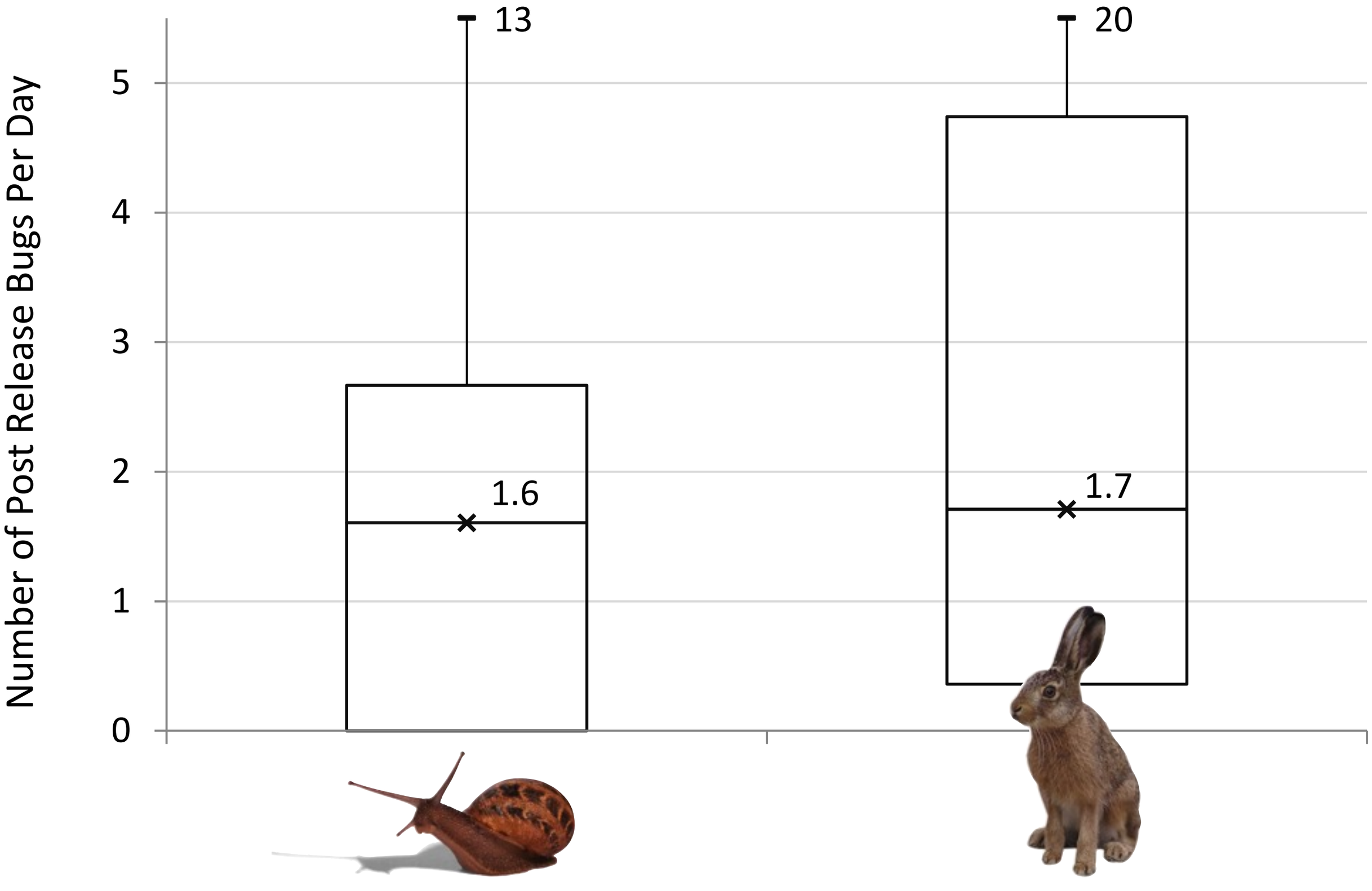


release cycle length

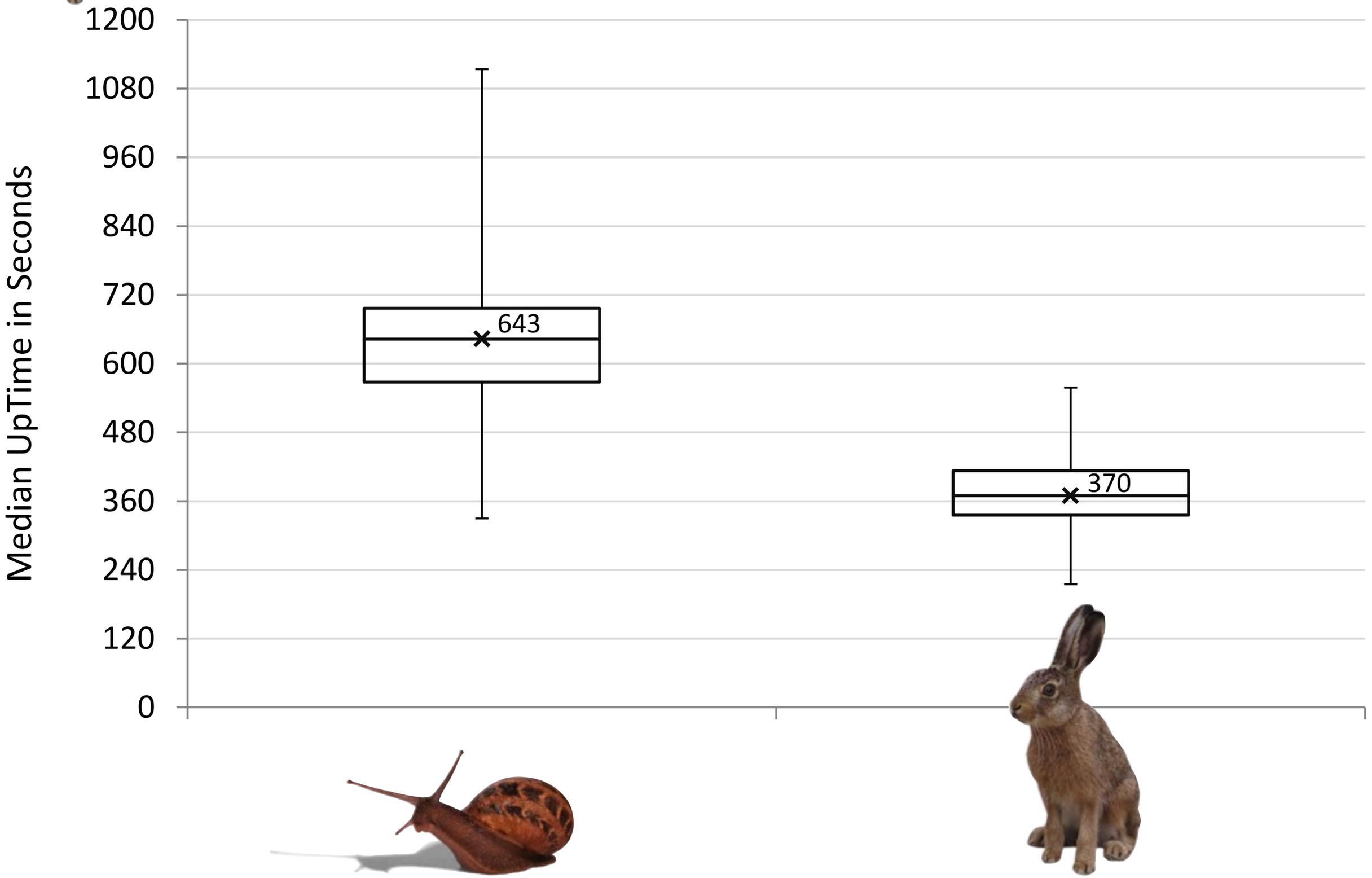
vs.

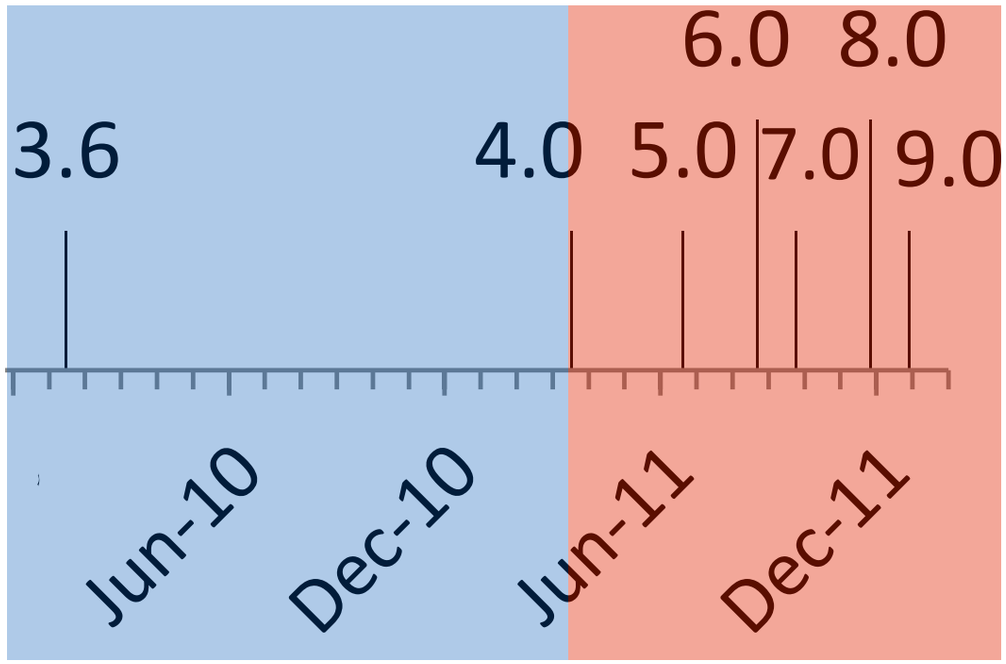
Do Faster Releases Improve Software Quality? - An Empirical Case Study of Mozilla Firefox (Khomh et al.)

Same # Post-release Bugs



Crashes Pop Up Earlier





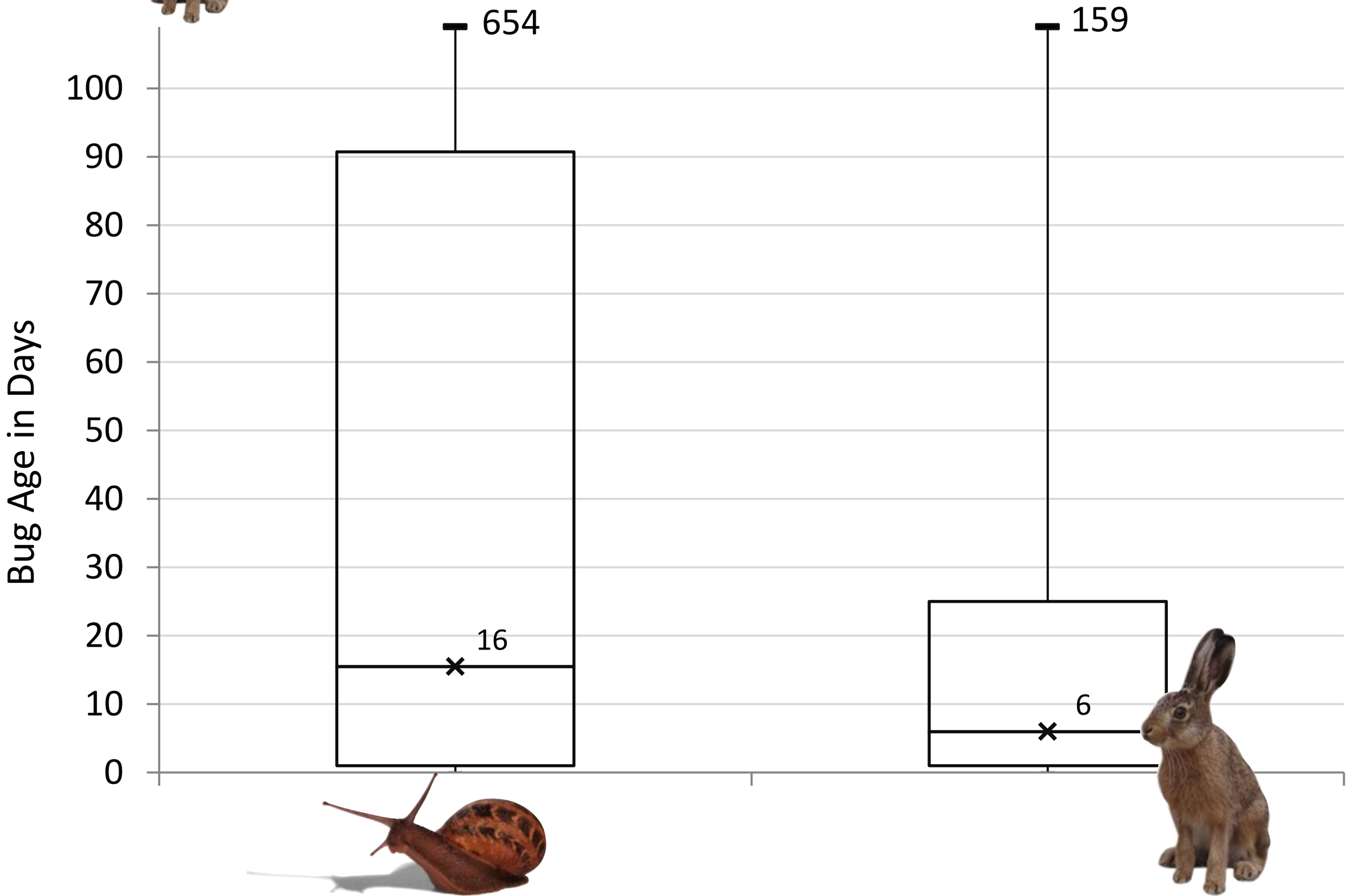
release cycle length

VS.

bug fixing

Do Faster Releases Improve Software Quality? - An Empirical Case Study of Mozilla Firefox (Khomh et al.)

Bugs are Fixed Faster



Proportionally Less  Bugs Fixed



100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%

% of Bugs Fixed

59%
37%
67%
43%




What about Other Projects?

What about Other Projects?

How to Make Software and Build Comprehension

Pragmatic?

Where are the Tools?

refactor your makefiles!

test your build!

what did the other team break now ;-)

keep poking those upstream guys until they give in :-)

Work fast and don't be afraid to break things.

We need more canaries!

Mark Zuckerb

CEO & Founder, Fa

http://goo.gl/UICW


Express yourself in the world's largest 3D Chat and Dress-Up community!
Member Login
On average we deploy new code **fifty** times a day.
in 66 different countries!
Sign in with:
Choose Your FREE Avatar
Over 2 Million people *like* IMVU on Facebook!
FREE


Express yourself in the world's largest 3D Chat and Dress-Up community!
Member Login
On average we deploy new code **fifty** times a day.
in 00 different countries!
Sign in with:
Choose Your FREE Avatar
Over 2 Million people like IMVU on Facebook!
FREE


http://behrns.files.wordpress.com/2008/03/ikea-car.jpg

Release Engineering

in-house/3rd party development

integrate

test

build

keeping cycle time down

deployment

Release Engineering

in-house/3rd party development

integrate

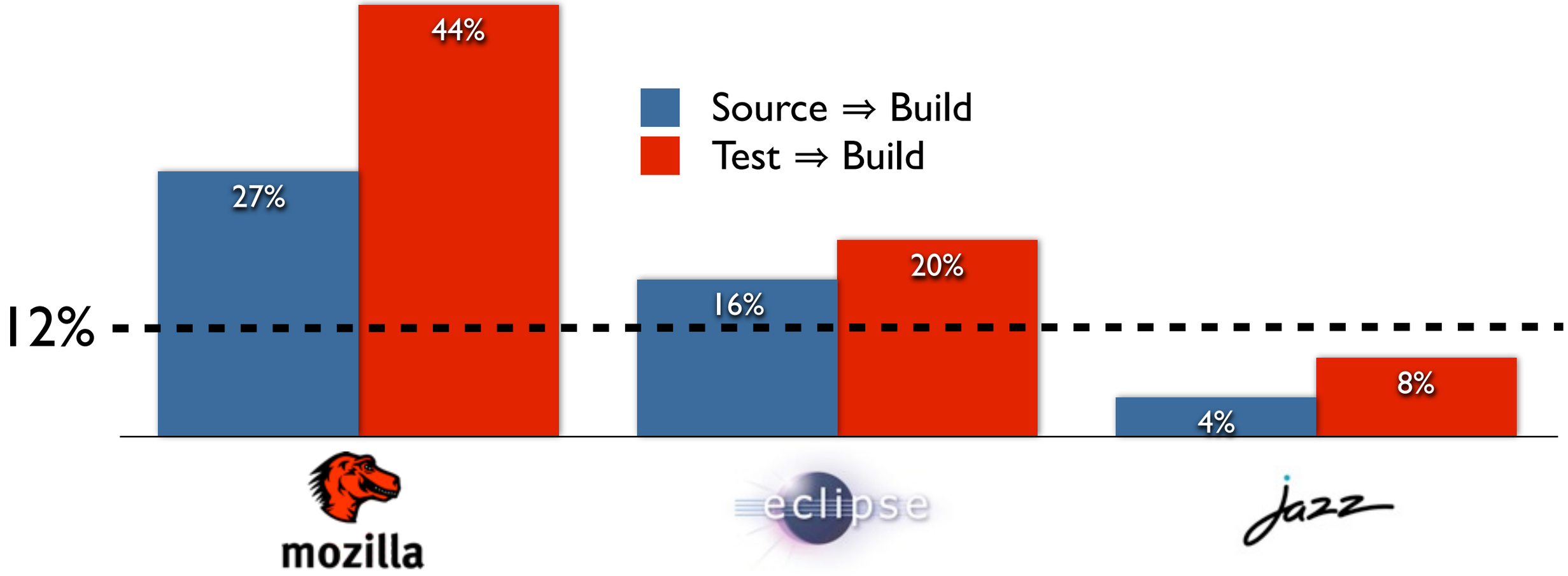
test

build

keeping cycle time down

deployment

The Build System Requires Significant **Maintenance**



Release Engineering

in-house/3rd party development

integrate

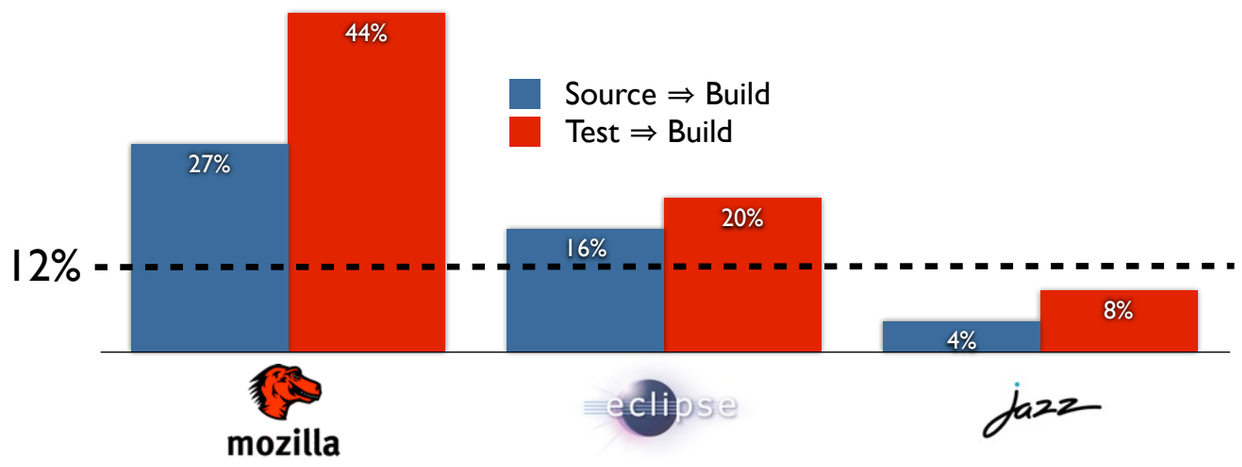
test

build

keeping cycle time down

deployment

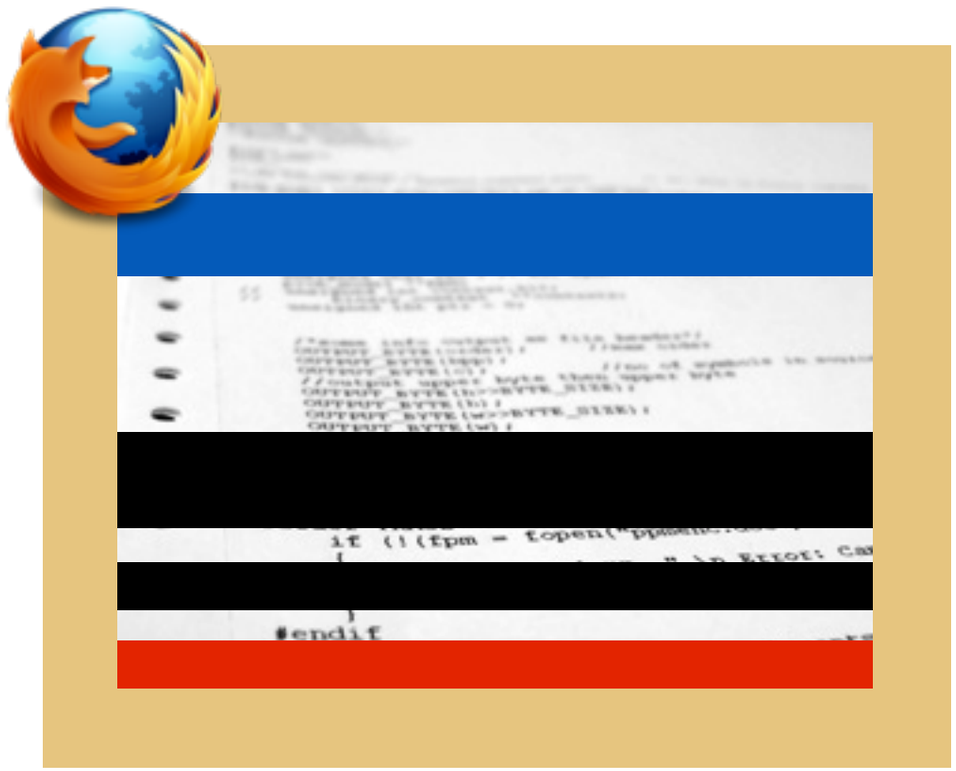
The Build System Requires Significant **Maintenance**



Risk Analysis & Cherry-picking

diff

maintainer



Upstream Sync

Release Engineering

in-house/3rd party development

integrate

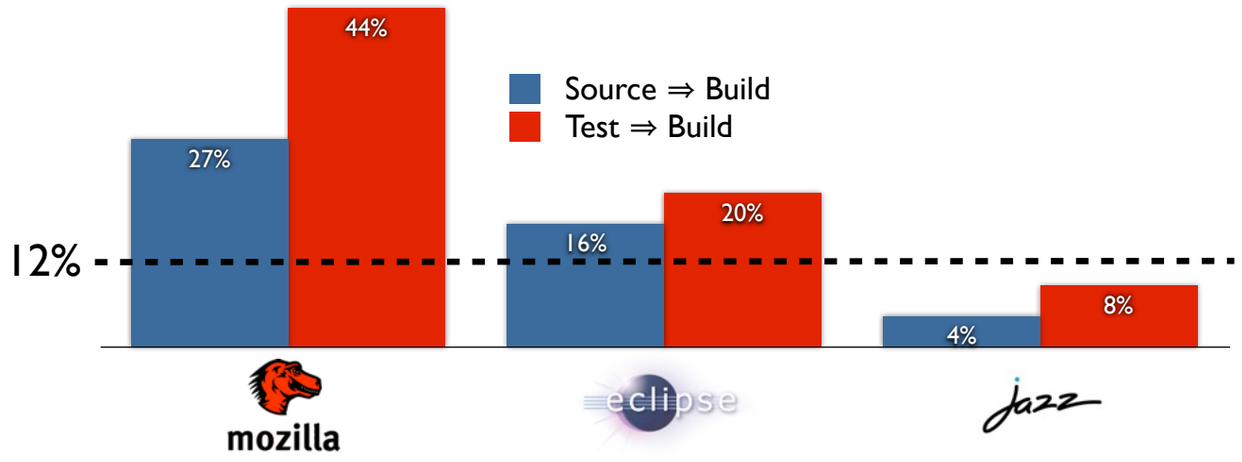
test

build

keeping cycle time down

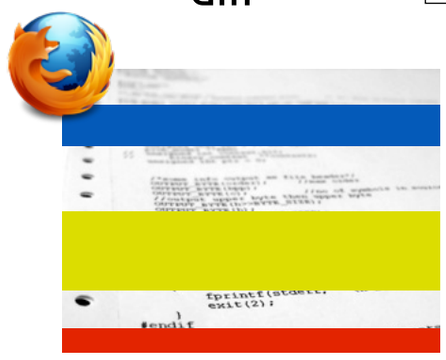
deployment

The Build System Requires Significant **Maintenance**

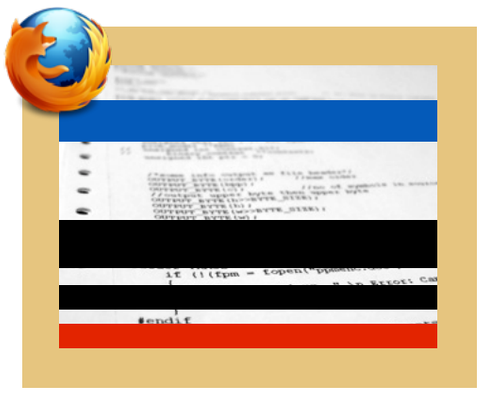


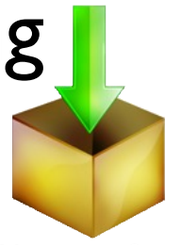
Risk Analysis & Cherry-picking

diff

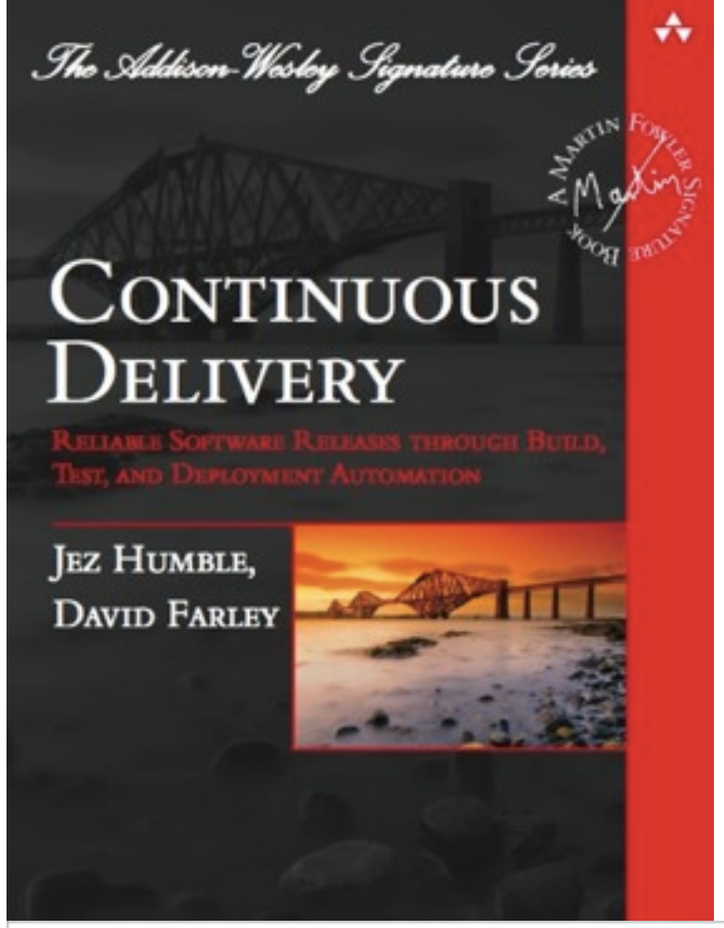


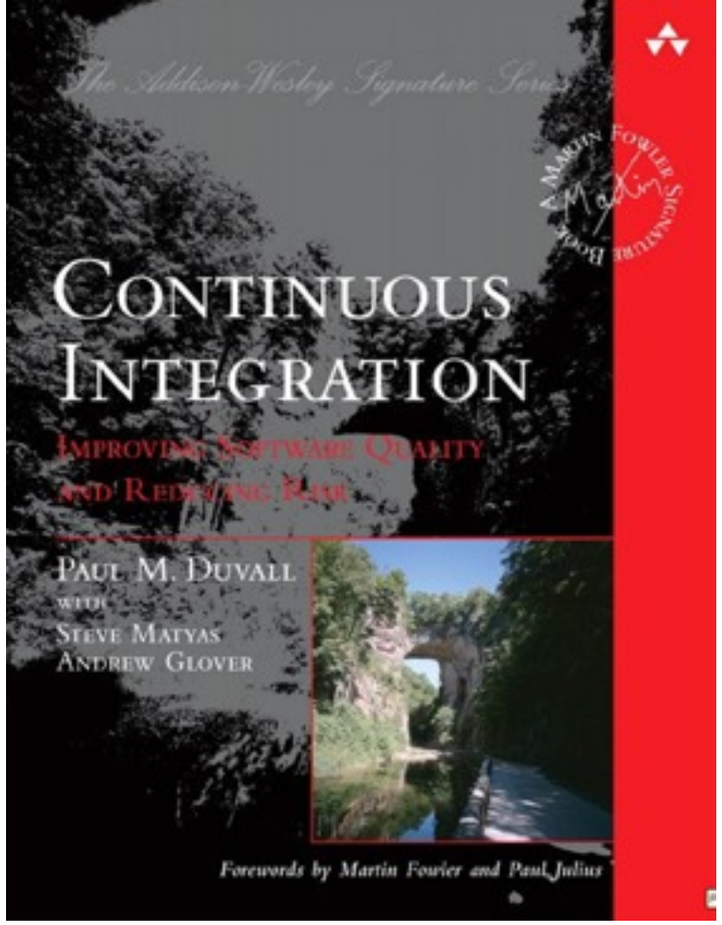
maintainer

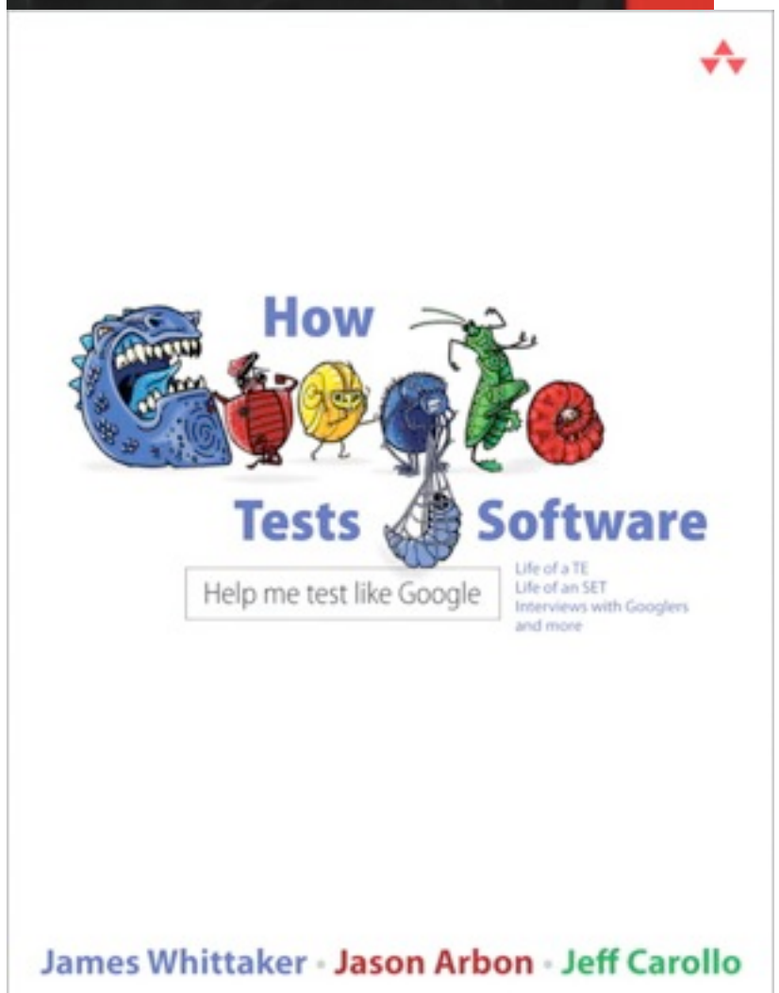


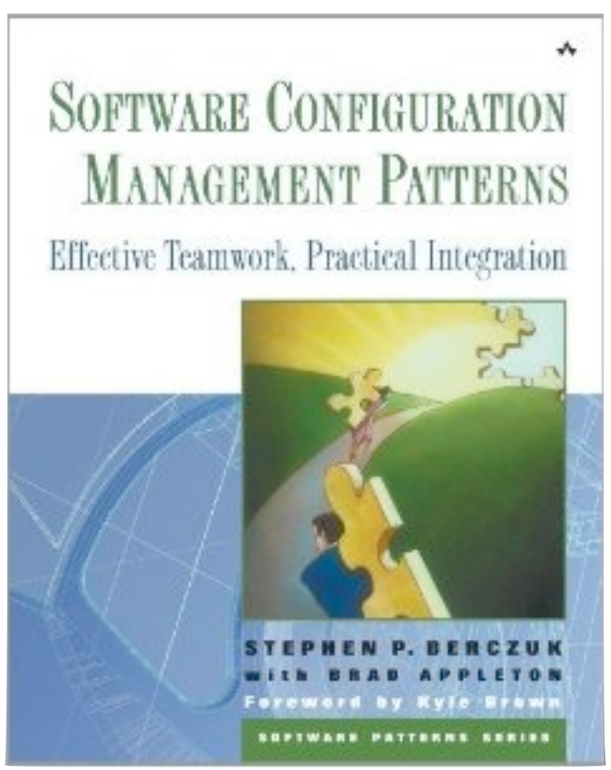


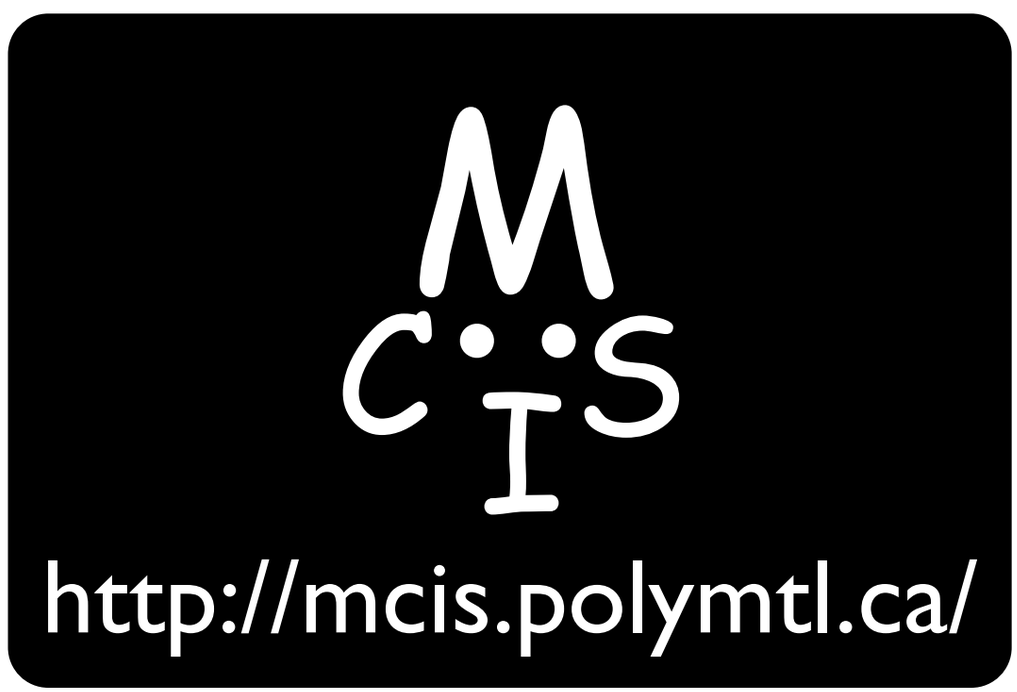
Upstream Sync











http://google-engtools.blogspot.ca/