CTSC

CENTER FOR TRUSTWORTHY
SCIENTIFIC CYBERINFRASTRUCTURE

The NSF Cybersecurity Center of Excellence

Secure Software Engineering Best Practices

Randy Heiland and Susan Sons
CACR, Indiana University
https://cacr.iu.edu/

NSF Cybersecurity Summit
Aug 16, 2016

trustedci.org

Center for Trustworthy Scientific Cyberinfrastructure

The mission of CTSC is to provide the NSF community with a coherent understanding of cybersecurity, its importance to computational science, and the resources to achieve and maintain an appropriate cybersecurity program.

CENTER FOR APPLIED CYBERSECURITY RESEARCH
INDIANA UNIVERSITY
Pervasive Technology Institute

NCSA

PITTSBURGH SUPERCOMPUTING CENTER

THE UNIVERSITY of WISCONSIN MADISON

NSF

Audience Participation

- Encourage questions, comments, and interaction during the presentation, esp.
  - personal/project-specific stories, both positive and not-so-positive
  - experience with tools

- Welcome constructive feedback

Challenges for this presentation

- Not knowing audience in advance
- Right level of detail:
  - *"developers need to be aware of secure coding techniques and tools"* (too high)

  vs.
  - *"if you're writing a web application using javascript, you need to …"* (too low)

Audience?

Not mutually exclusive, obviously.

- software developers?
- scientists?
- students?
- managers?
- system admins?
- analysts?

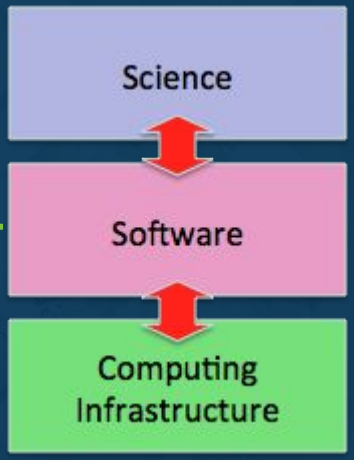
Background and Motivation

NSF "CI Framework for 21st century" (CIF21)

Software is fundamentally computer code. It can be delivered to end users in multiple formats, ranging from an archive that a user downloads and builds to an executable or a service running on a remote system to which a user connects. Especially at large scale, software is generally difficult to design, implement and then maintain, and the software needed by the science, engineering, and education communities is particularly complex. Software must be reliable, robust, and secure; able to produce trustable and reproducible scientific results; …

http://www.nsf.gov/pubs/2012/nsf12113/nsf12113.pdf

Software and the NSF



- Software (including services) essential for the bulk of science
  - About half the papers in recent issues of Science were software-intensive projects
  - Research becoming dependent upon advances in software
  - Significant software development being conducted across NSF: NEON, OOI, NEES, NCN, iPlant, etc
  - Wide range of software types: system, applications, modeling, gateways, analysis, algorithms, middleware, libraries
- Software is not a one-time effort, it must be sustained
  - Development, production, and **maintenance** are people intensive
  - Software life-times are long vs hardware
  - Software has under-appreciated value

CyVerse

http://www.slideshare.net/danielskatz/metrics-citation-for-software-and-data

CTSC

"It's clear that open and reproducible science and engineering will need an integrated approach to code and data management, as both are complex and evolving."

LeVeque, Randall J., Ian M. Mitchell, and Victoria Stodden. 2012. "Reproducible Research for Scientific Computing: Tools and Strategies for Changing the Culture." *Computing in Science and Engineering* 14 (4).

Further reading

- Howison, J., E. Deelman, M. J. McLennan, R. Ferreira da Silva, and J. D. Herbsleb. 2015. “Understanding the Scientific Software Ecosystem and Its Impact: Current and Future Measures.” *Research Evaluation*, July. doi:10.1093/reseval/rvv014

Secure Software Engineering

Secure SE vs. SE

How is Secure Software Engineering different from Software Engineering?

From CIF21, why not also have:

{Reliable, Robust, Secure, Trustable, Reproducible} SE?

Secure SE vs. SE

How is Secure Software Engineering different from Software Engineering?

From CIF21, why not also have:

{Reliable, Robust, Secure, Trustable, Reproducible} SE ?

→ SE should be comprehensive.

Introduction

Software engineering (SE) is concerned with developing and maintaining software systems that behave reliably and efficiently, are affordable to develop and maintain, and satisfy all the requirements that customers have defined for them.

http://computingcareers.acm.org/?page_id=12

Introduction

Software engineering (SE) is concerned with developing and maintaining software systems that behave reliably and efficiently, are affordable to develop and maintain, and satisfy all the requirements that customers have defined for them.

→ security

Software engineering (SE) is about

problem modeling and analysis, software design, software verification and validation, software quality, software process, software management, etc.

http://computingcareers.acm.org/?page_id=12

Motivation

Why do we care about secure software?

- prevent loss of data
- prevent premature leaks of data
- prevent downtime of resources

(CIA: Confidentiality, Integrity, Availability)

→ better science, better public trust

SE is language agnostic; but tools may not be

Language Rank	Types	Spectrum Ranking
1. C		100.0
2. Java		98.1
3. Python		98.0
4. C++		95.9
5. R		87.9
6. C#		86.7
7. PHP		82.8
8. JavaScript		82.2
9. Ruby		74.5
10. Go		71.9

http://spectrum.ieee.org/computing/software/the-2016-top-programming-languages

http://spectrum.ieee.org/ns/IEEE_TPL_2016/methods.html (12 metrics; 10 sources)
Cf. http://www.tiobe.com/tiobe-index/

Community Engineering

Community Engineering - definition

Community Engineering is the continuing process of establishing expectations and social environment that support, rather than hinder, effective, adaptive, and ethical engineering practice within a team or community.

No one develops software in a vacuum. Our tools, our expectations, our relationships with teammates and upstream developers, and the overall environments in which we operate have huge cumulative impact on our behavior and the quality of our work.

You can’t change people…
...but you can change their environment.

Community Engineering Overview

- Split agency is NEVER acceptable.
- Make the right thing as easy to do as possible.
- Communication should be cheap, frequent, and clear.
- Look at your incentives and disincentives like a gamer: if you minmax hard, what do you get?
- No environment makes every human happy. Think hard about who you want to attract and keep, and what is a good culture for those people.

Split Agency

"Split agency" is the condition where the person who controls how something is done and with what resources is not the person who is responsible for the outcome.

Split agency kills morale, creates perverse incentives, and lowers quality of work.

Example: Project lead budgets zero developer time for documentation, then holds developers responsible for lack of documentation.

Make doing the right thing easier.

In an ideal world, developers do the right thing no matter what stands in their way.

In the real world, where we have schedules and limited skill sets and interpersonal conflicts, most developers will do as good a thing as seems plausible.

This means that, with better tools and processes, you can get better work out of most people...

Make doing the right thing easier. (cont.)

Generally, look ways to automate away repetitive or time-consuming tasks, and reduce developer friction. Examples:

- Set up a pre-receive hook to reject commits that fail tests or don't meet style standards so developers fix small problems on code that's fresh in their minds, not big ones on code they have to re-analyze.
- Migrate away from outdated tools like CVS and SVN… the time invested learning and moving to git or hg is quickly paid back in saved effort/frustration.

Communication should be…

Cheap: developers should be focused on development; communication should be fast, easy, and not take much time and attention.

Frequent: a quick question or clarification early will save a lot of time over only addressing things when they've become problems.

Clear: precise, concrete communication will get you furthest with most engineers and cause the least stress

What does your project incentivize?

"minmaxing" is a gamer term that means to carefully optimize a set of variables for optimal game performance. Most good programmers are at least decent game theorists...which means that it is VERY easy to destroy a team with perverse incentives.

Example: If you measure developer performance by lines of code produced, you will get bloated code that no one wants to refactor and little to no documentation.

What does your project disincentivize?

If the boss gets angry when coders are standing around talking instead of coding, collaboration plummets.

If the continuous integration tool breaks on 30% of commits, and has to be manually cleaned up, programmers will abandon “commit early, commit often” in favor of fewer, bigger, hard-to-review commits.

Creating culture...

There are many “good” development team cultures, for varying definitions of “good”. Optimal culture for a team that quietly and competently maintains important infrastructure is likely to be different than for a team that’s trying to bring an edgy and still somewhat undefined product to market, which is in turn different from a team that only does incident response.

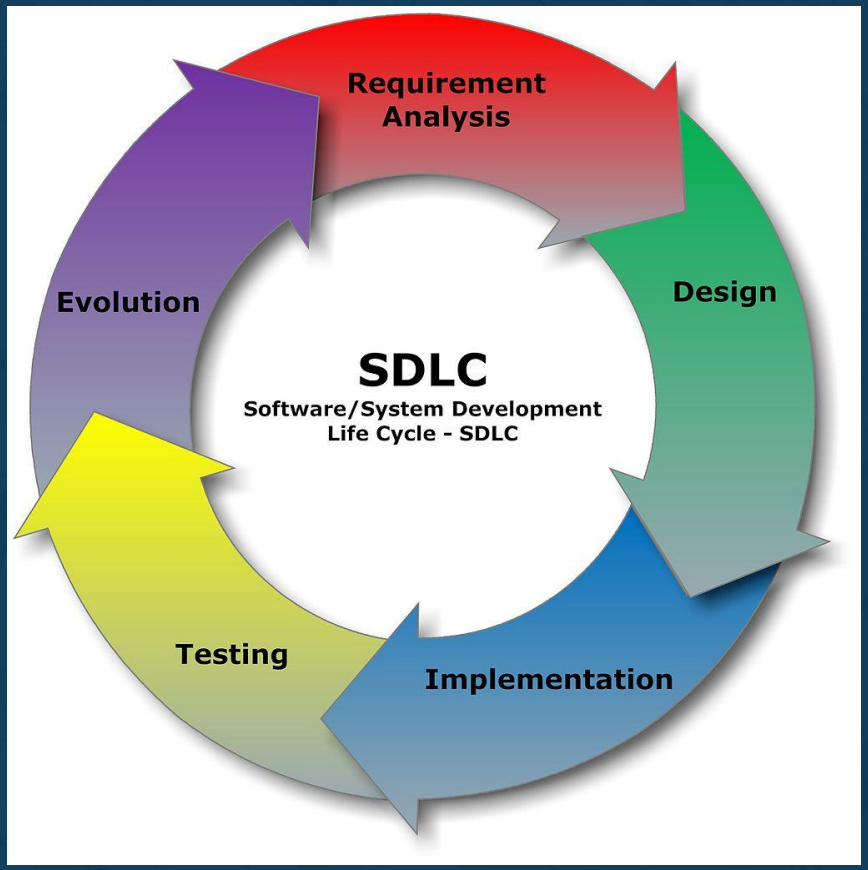
What your team does should define its culture, to attract and keep the most qualified people possible.

Secure SwEng BP: Goal

Help software developers and operators deliver and maintain secure software over its entire lifecycle.

SwEng Processes/Lifecycle

1) Requirements
2) Design
3) Implement
4) Test
5) Maintain



https://commons.wikimedia.org/wiki/File:SDLC_-_Software_Development_Life_Cycle.jpg

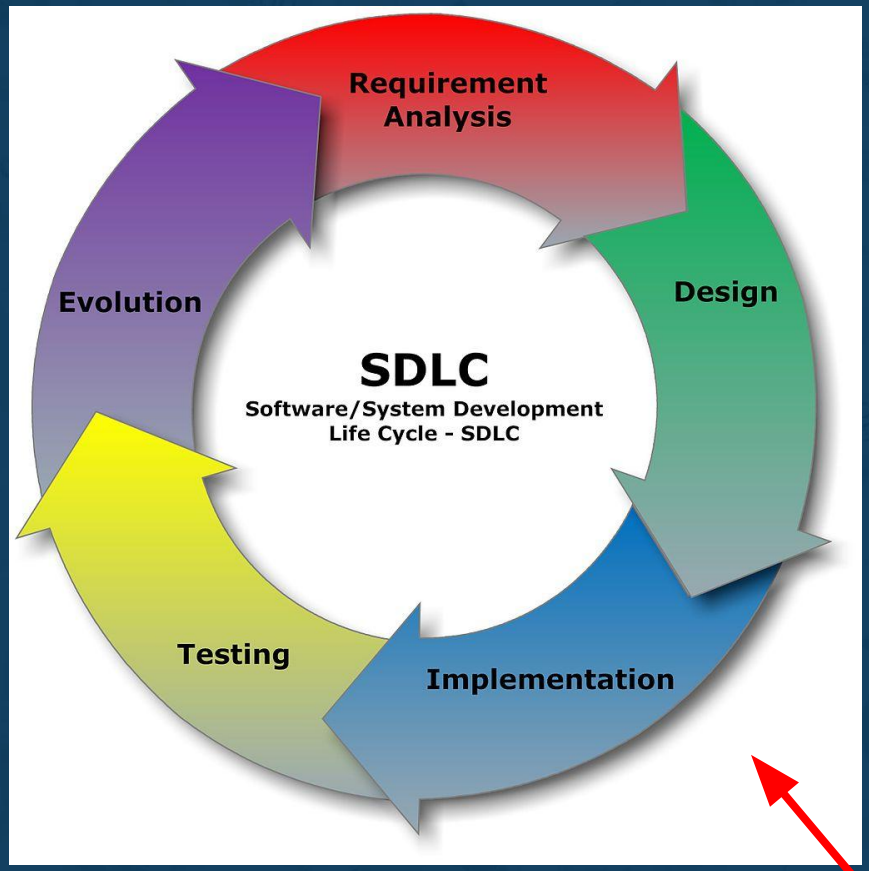
CTSC

SwEng Lifecycle + Security

code signing

vuln mgt

dynamic analysis



logging

secure coding

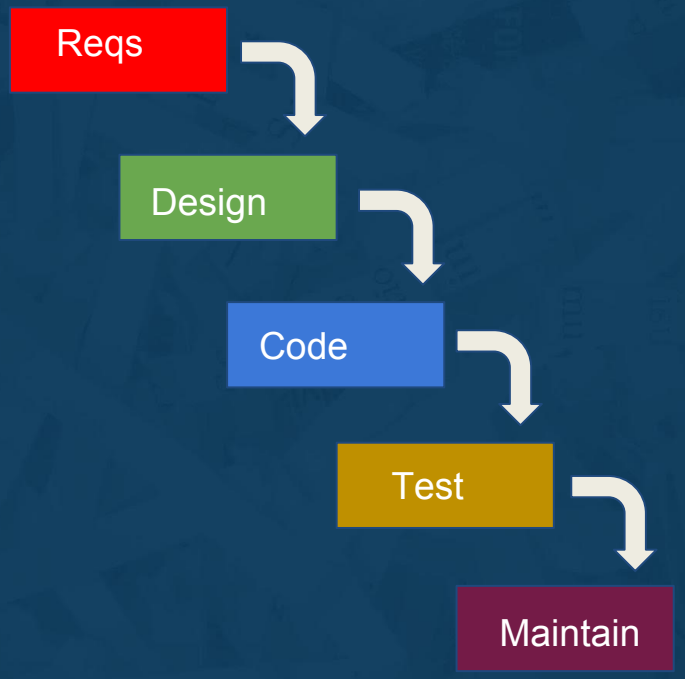
static analysis

Be security conscious during each phase.

CTSC

Evolution of SwEng Models

- Waterfall
- Incremental
- Extreme
- Spiral
- Agile
- CI/CD



CTSC

- Waterfall
- Incremental
- Extreme
- Spiral
- Agile
- CI/CD

http://agilemanifesto.org/ (2001)

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

Traditional vs. Agile

	Traditional view	Agile perspective
Design process	Deliberate and formal, linear sequence of steps, separate formulation and implementation, rule-driven	Emergent, iterative and exploratory, knowing and action inseparable, beyond formal rules
Goal	Optimization	Adaptation, flexibility, responsiveness
Problem-solving process	Selection of the best means to accomplish a given end through well-planned, formalized activities	Learning through experimentation and introspection, constantly reframing the problem and its solution
View of the environment	Stable, predictable	Turbulent, difficult to predict
Type of learning	Single-loop/adaptive	Double-loop/generative
Key characteristics	Control and direction Avoids conflict Formalizes innovation Manager is controller Design precedes implementation	Collaboration and communication; integrates different worldviews Embraces conflict and dialectics Encourages exploration and creativity; opportunistic Manager is facilitator Design and implementation are inseparable and evolve iteratively
Rationality	Technical/functional	Substantial
Theoretical and/or philosophical roots	Logical positivism, scientific method	Action learning, John Dewey's pragmatism, phenomenology

http://dx.doi.org/10.1109/MS.2009.145

Courtesy XSEDE.

XSEDE community builds an Agile student

- Collaboration
- Spiral development
- Pair-programming
- Rapid release

https://sciencenode.org/feature/xsede-community-builds-an-agile-student.php
https://www.xsede.org/

CTSC

- Waterfall
- Incremental
- Extreme
- Spiral
- Agile
- CI/CD

Continuous Integration / Continuous Delivery:

- prioritize deployable s/w (at any moment) vs. working on new features
- incremental s/w change → automated test & feedback

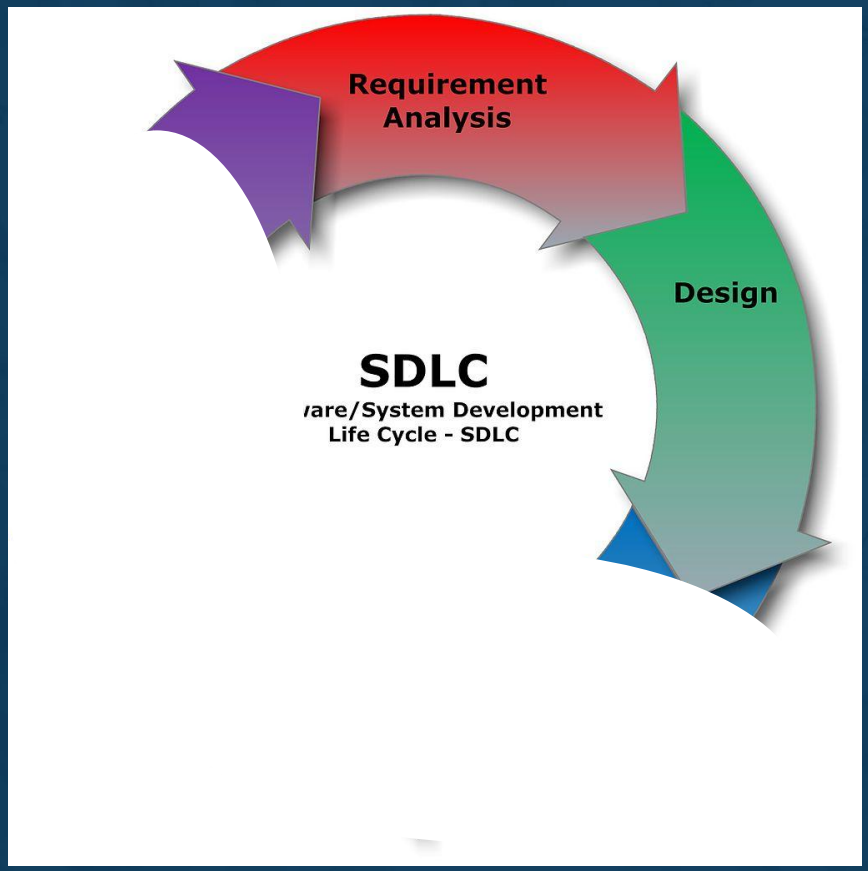
SwEng Models

Extreme ~ Incremental ~ Agile ~ CI/CD

$\longrightarrow$ DevOps

The idea of doing/automating frequent builds and tests, after incremental changes, and making it operational.

security at each phase



Saltzer & Schroeder (1975):

1. Economy of Mechanism (simple & small)
2. Separation of Privilege: (2+ pieces of info for access)
3. Least Privilege (each process has min priv)

…

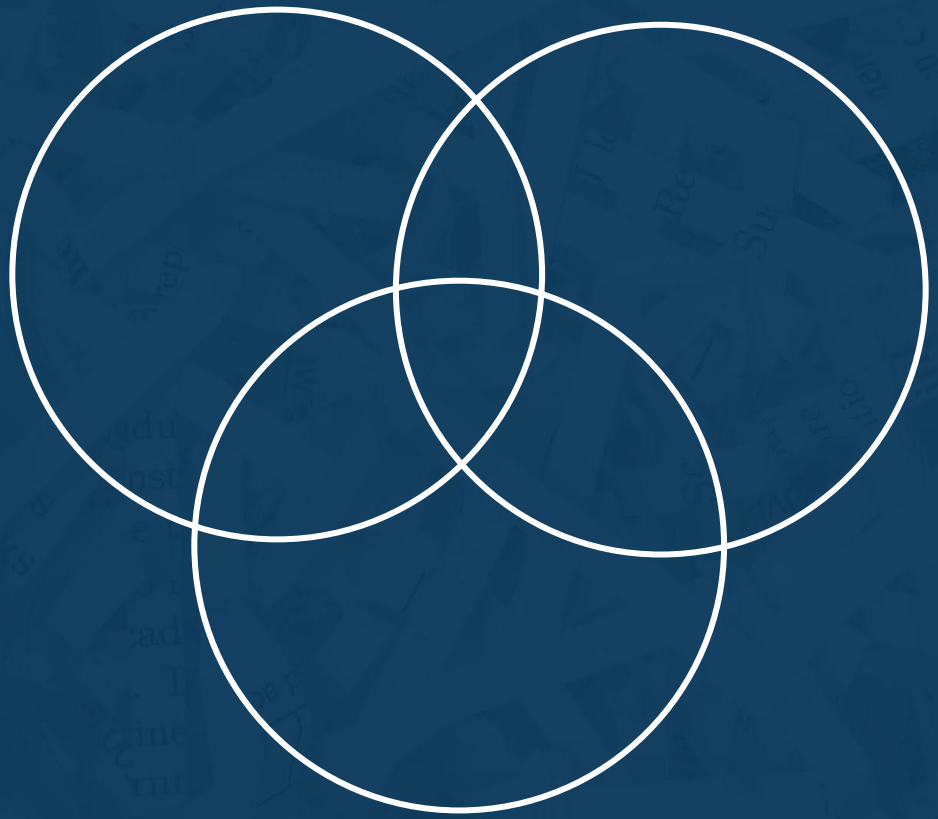
8. Psychological Acceptability (easy to use)

Further reading

- Dyba, T., and T. Dingsoyr. 2009. “What Do We Know about Agile Software Development?” *IEEE Software*. http://dx.doi.org/10.1109/MS.2009.145
- http://martinfowler.com/bliki/ContinuousDelivery.html
- https://insights.sei.cmu.edu/devops/2014/03/an-introduction-to-devops.html
- https://buildsecurityin.us-cert.gov/process-agnostic-navigational-view
- Best Practices in Scientific Computing, G.Wilson, 2014: http://swcarpentry.github.io/slideshows/best-practices
- Saltzer, J. H. & Schroeder, M. D. "The Protection of Information in Computer Systems," *Proc of the IEEE 63,* 9 (1975).

Software-related Thrusts at CTSC

Software Security



CTSC has a thrust in each of these.

Software Assurance (SwA)

#1) SwA is the level of confidence that software is free from vulnerabilities, either intentionally designed into the software or accidentally inserted at any time during its life cycle, and that the software functions in the intended manner.

https://samate.nist.gov/Main_Page.html

#2) The processes (e.g., secure coding, static analysis) that help improve this level of confidence.

→ secure coding instruction (http://trustedci.org/trainingmaterials)

Situational Awareness

Being aware of software vulnerabilities and how they might affect a user community. Offering advice on how to patch or update vulnerable software.

http://trustedci.org/situational-awareness

http://blog.trustedci.org/2016/08/situational-awareness.html

Situational Awareness: example

perfSONAR provides tools and architecture to help monitor network performance.
...released updated packages on July 7th to address two security issues:

1. An unauthenticated remote access vulnerability that could allow an attacker to view local files as the 'perfsonar' user.

2. A local privilege escalation issue.
(instructions for updating software follow)

https://list.indiana.edu/sympa/arc/ctsc-announce-inf-l/2016-07/msg00000.html

Secure SwEng BP: Approach

- Instill security awareness in software engineers - developers and testers.
- Educate them in appropriate processes, practices, and tools.

Secure SwEng BP Topics

- Repositories
- Testing
- Static Analysis
- Vulnerability Management
- Release & Delivery
- Coding/Project Tools
- Documentation

Secure SwEng BP Topics

- Repositories
- Testing
- Static Analysis
- Vulnerability Management
- Release & Delivery
- Coding/Project Tools
- Documentation

Repositories

What's in a repository? Everything (we hope).

e.g. https://github.com/TAlexPerkins/Zika_nmicrobiol_2016

Modern SCMs offer more than just the history of a code base:

- integrity checking
- automation of common tasks
- cheaper branching and merging, which encourages better development practices
- the ability to work in a decentralized manner

Source code repositories and version control

- CVCS: RCS, CVS, SVN
  - Outmoded, should be migrated away from
- DVCS: Git, Mercurial
  - More modern, but each has trade-offs.
  - If unsure, default to git.
- Each of these systems, especially git and Mercurial, can also function as a part of a bigger continuous integration system.

Repositories and Hosting Services

Regardless of the repo/hosting service you choose, be mindful of security considerations:

- physical security
- server logging
- encrypted access
- granularity of access control
- 2FA
- do not commit sensitive data to public repos
  - keep in mind that a currently-private repo may need to be shared more widely later: keep credentials separate from code, or you'll be sanitizing history.

Further reading

- https://help.github.com/articles/github-security/
- http://www.theregister.co.uk/2015/01/06/dev_blunder_shows_github_crawling_with_keyslurping_bots/

Secure SwEng BP Topics

- Repositories
- Testing
- Static Analysis
- Vulnerability Management
- Release & Delivery
- Coding/Project Tools
- Documentation

Software Testing

- why is it necessary?
- why is it difficult?
- how well does it work?
- can it be made easier?

Software Testing

- why is it necessary?
  - test for “correctness”
  - help prevent bugs/vulnerabilities
  - improve usability
- why is it difficult?
- how well does it work?
- can it be made easier?

Software Testing

- why is it necessary?
- why is it difficult?
  - time-consuming
  - combinatorial challenge
- how well does it work?
- can it be made easier?

Software Testing

- why is it necessary?
- why is it difficult?
- how well does it work?
  - as well as your tests
- can it be made easier?

Software Testing

- why is it necessary?
- why is it difficult?
- how well does it work?
- can it be made easier? yes:
  - testing frameworks
  - automated testing (e.g. via CI)

Types of Testing

- Static
  - code not executing
  - code walkthroughs
- Dynamic
  - code is executing
  - written by software dev/test engineer

- Black-box
  - don't know source code
- White-box
  - know source code

Levels of Dynamic Testing

- Unit (small)
  - test single functions
  - written by software dev
- Integration (medium)
  - test interacting functions/packages
  - written by software dev/test engineer
- Acceptance (large)
  - overall testing
  - written by test engineer

More Dynamic Testing

- Regression
  - as software is modified, make sure no new (or old) bugs have been introduced
- Combinatorial
  - all combinations of input parameters
- Fuzz
  - with random/noisy inputs
- Security
  - for Confidentiality, Integrity, Availability (CIA)

DARPA Grand Challenge 2016

Find (and fix) vulnerabilities in binary codes.
Fuzzing was a favorite technique.

https://www.cybergrandchallenge.com/
https://github.com/CyberGrandChallenge/

CTSC

Testing: think globally, act locally

Acting locally:

Use Assertions in code!

"primary purpose is to instrument code with test probes that will detect errors as close as possible to their place of occurrence."
Tony Hoare, 2002

"... the programmer should make assertions about the various states that the machine can reach."
Alan Turing, 1949

Assertions

Assertions are always expected to be True:

```
assert(condition)
```

If they are false at runtime, they will throw an error.

(They can be disabled if desired).

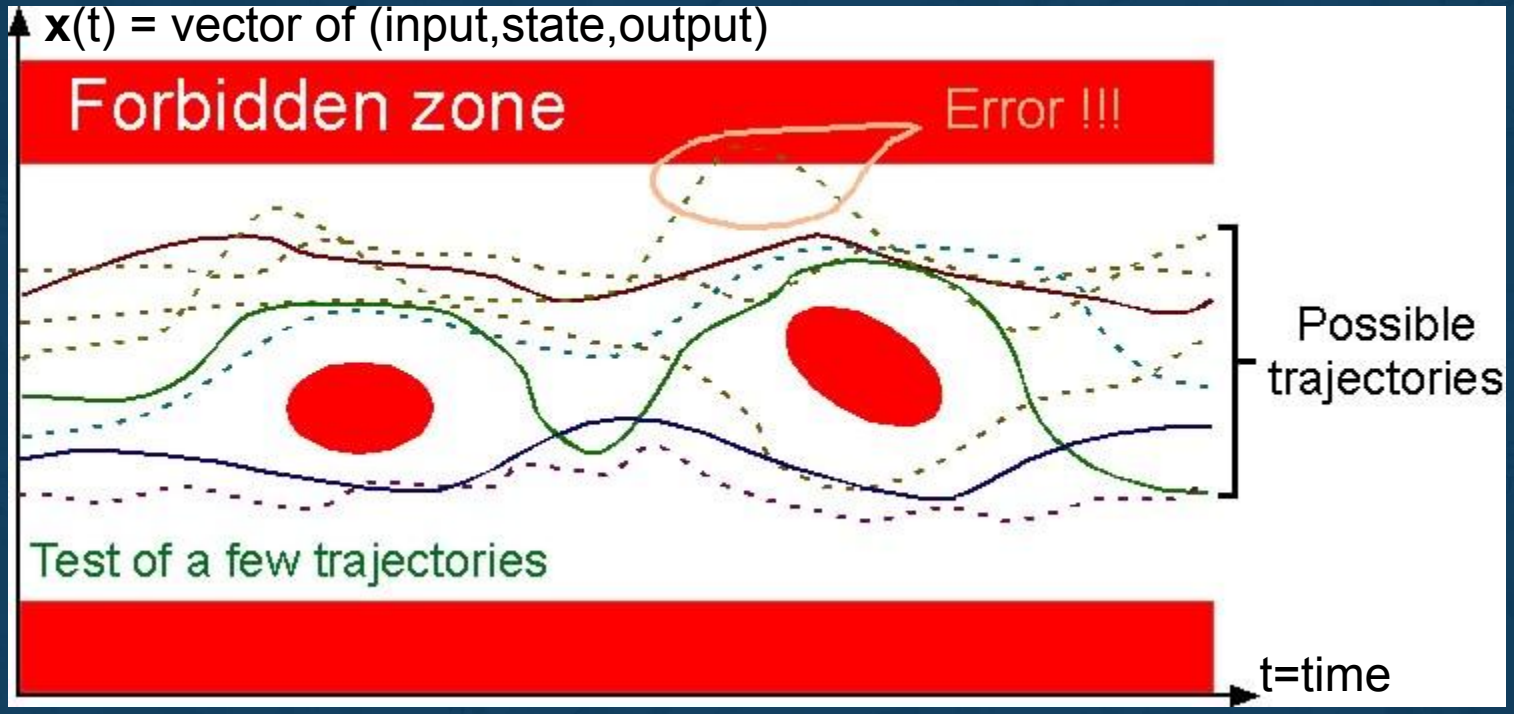
C/C++:

```
assert(ptr);
```

Java:

```
Assert.assertTrue((project1.getCreationTime() -
                   project2.getCreationTime()) > 0);
```

Testing: theory



"It is not possible to write a program able to represent and to compute all possible executions of any program in all its possible execution environments."

www.di.ens.fr/~cousot/AI/IntroAbsInt.html

CTSC

Testing Frameworks

- Primarily for Unit Testing
- xUnit: for Unit testing
  - JUnit (Java), PyUnit (Python), etc.

wikipedia: List_of_unit_testing_frameworks

- lots of languages, lots of frameworks

Automated Testing

www.owasp.org/index.php/Appendix_A:_Testing_Tools

e.g.:

- github.com/google/googletest
  - Google's (open source) C++ testing framework
- http://docs.seleniumhq.org/
  - OSS for testing web applications

Automated Testing

Some languages are better equipped for testing than others.

https://golang.org/pkg/testing/

testing Package testing provides support for automated testing of Go pkgs.

iotest implements Readers and Writers useful mainly for testing.

quick implements utility functions to help with black box testing.

This may be useful for the next generation of software projects, but may not help us today.

Thoughts?

Testing Suites

The Software Assurance Reference Dataset (SARD) provide users, researchers, and software security assurance tool developers with a set of known security flaws.

http://samate.nist.gov/index.php/SARD.html

https://samate.nist.gov/SARD/testsuite.php?login=Guest

Further reading

- Kanewala, U. and J. M. Bieman. 2014. "Testing Scientific Software: A Systematic Literature Review." *Information and Software Technology* 56 (10): 1219–32. http://dx.doi.org/10.1016/j.infsof.2014.05.006
- Introducing Combinatorial Testing in a Large Organization: Experience Report, J. Hagar. D.R. Kuhn, R.N. Kacker, T. Wissink. *IEEE Computer*, April 2015.
- http://csrc.nist.gov/groups/SNS/acts/documents/kuhn-kacker-lei-hunter09.pdf
- https://www.wired.com/2016/06/hacker-lexicon-fuzzing/
- Hoare, T. 2002. "Assertions in Modern Software Engineering Practice." In *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International*, 459–459.
- http://www.turingarchive.org/viewer/?id=462&title=01a

Secure SwEng BP Topics

- Repositories
- Testing
- Static Analysis
- Vulnerability Management
- Release & Delivery
- Coding/Project Tools
- Documentation

```
$ make hello
c++ -std=c++11    hello.cpp   -o hello
hello.cpp:15:7: warning: using the result of an assignment as a
condition without parentheses [-Wparentheses]
        if (a=b) {
...
hello.cpp:15:7: note: use '==' to turn this assignment into an
equality comparison
```

Static Analysis

Static analysis tools try to find bugs/vulnerabilities in source code. Bugs are then categorized by severity.

Q: why doesn't every software developer use static analysis tools?

Do you/your team?

Static Analysis

Static analysis tools try to find bugs/vulnerabilities in source code. Bugs are then categorized by severity.

Q: why doesn't every software developer use static analysis tools?

A (typically): hassle (time, learning curve),

false positives, doesn't catch complex vulnerabilities, ...

https://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis

Coverity Scan (free for OSS)

e.g.,

Scilab — Open source software for numerical computation

Coverity Scan: scilab

Project Name	scilab
Lines of code analyzed	685,109
On Coverity Scan since	Sep 27, 2013
Last build analyzed	about 10 hours ago
Language	C/C++
Secondary Language	Java
Repository URL	git://git.scilab.org/scilab/
Homepage URL	http://www.scilab.org/

Jul 26, 2016	685,109	0.35
Last Analyzed	Lines of Code Analyzed	Defect Density

Defect changes since previous build dated Jul 21, 2016

0	8
Newly detected	Eliminated

Defects by status for current build

3,684	243	3,242
Total defects	Outstanding	Fixed

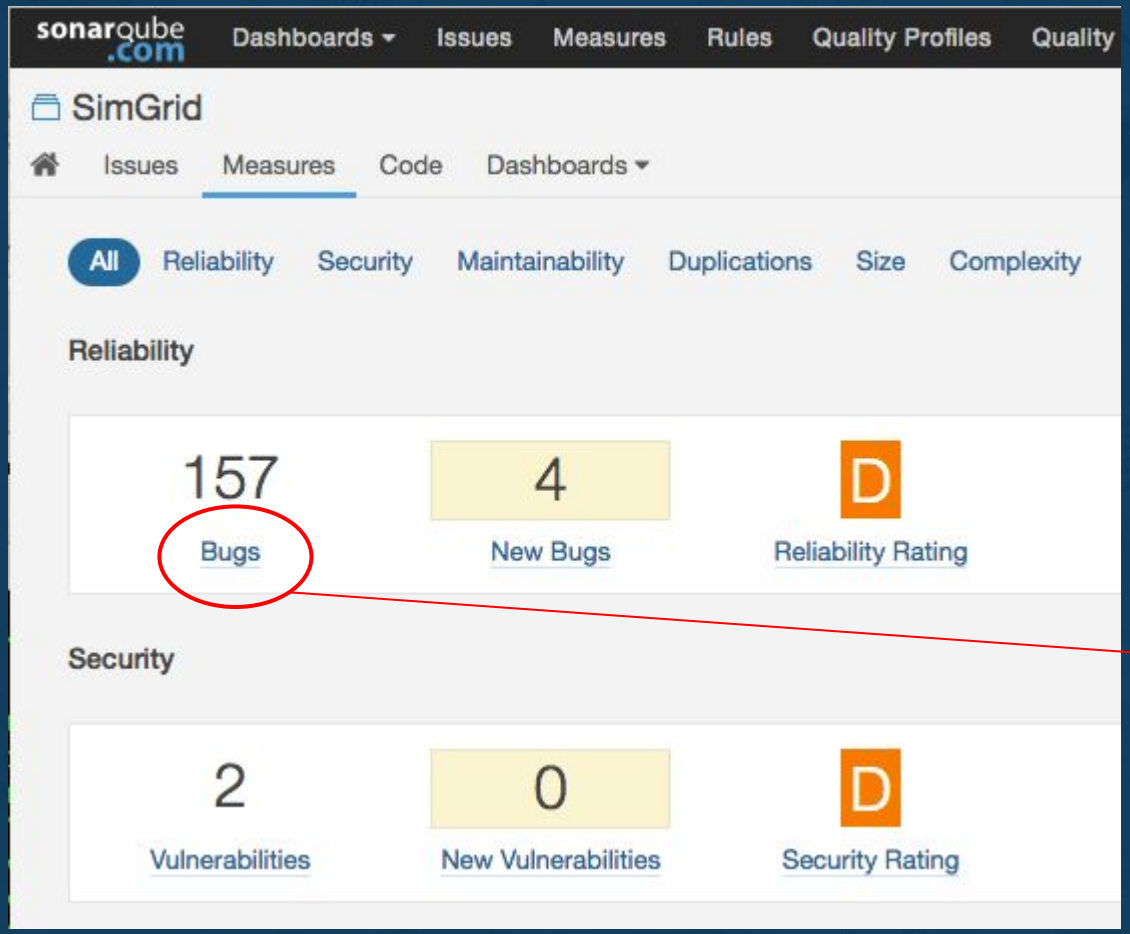
Defect density is measured by the number of defects per 1,000 lines of code

CTSC

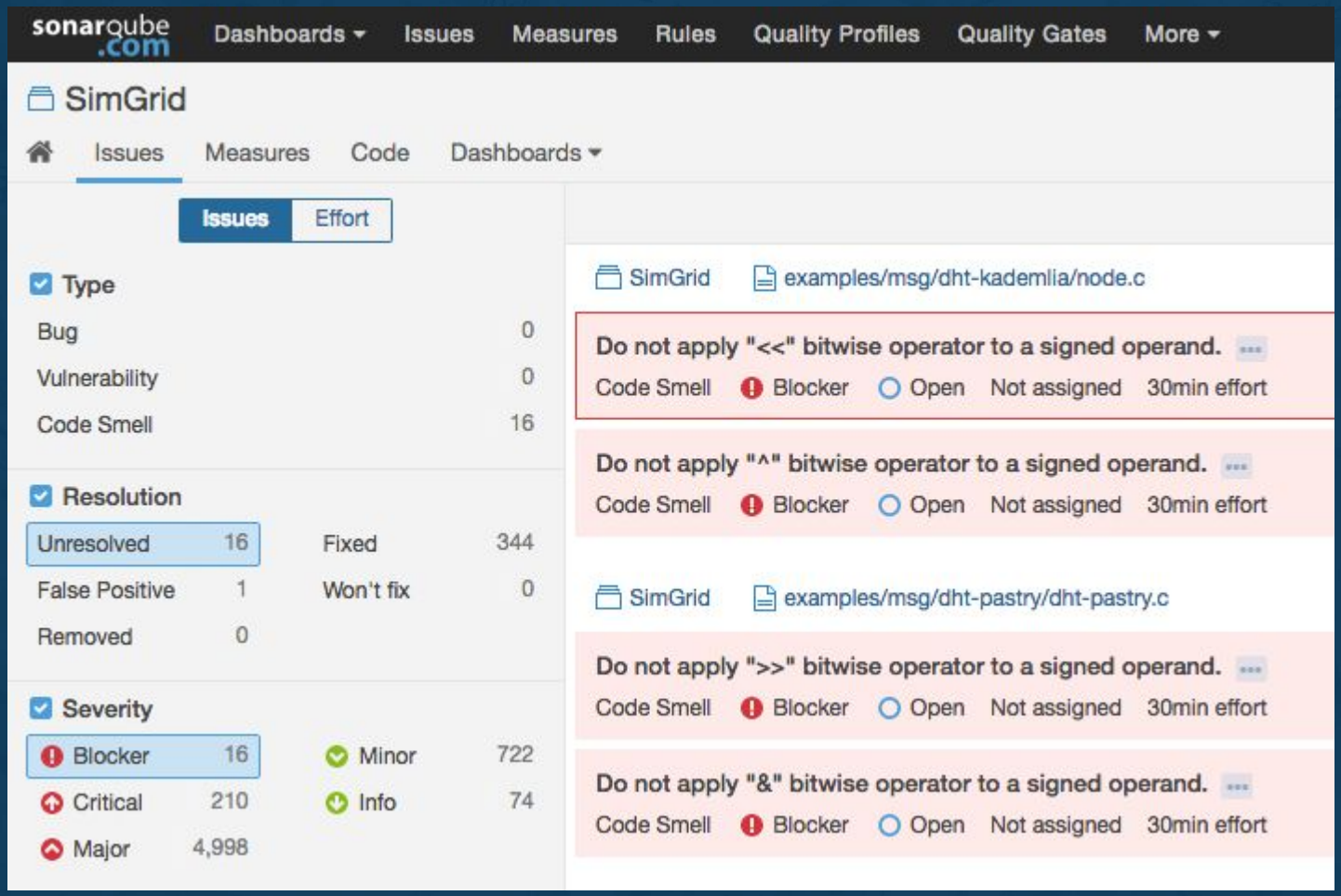
SonarQube (https://sonarqube.com/)

- OSS
- Used by many projects
- Can be integrated with Eclipse IDE

CTSC

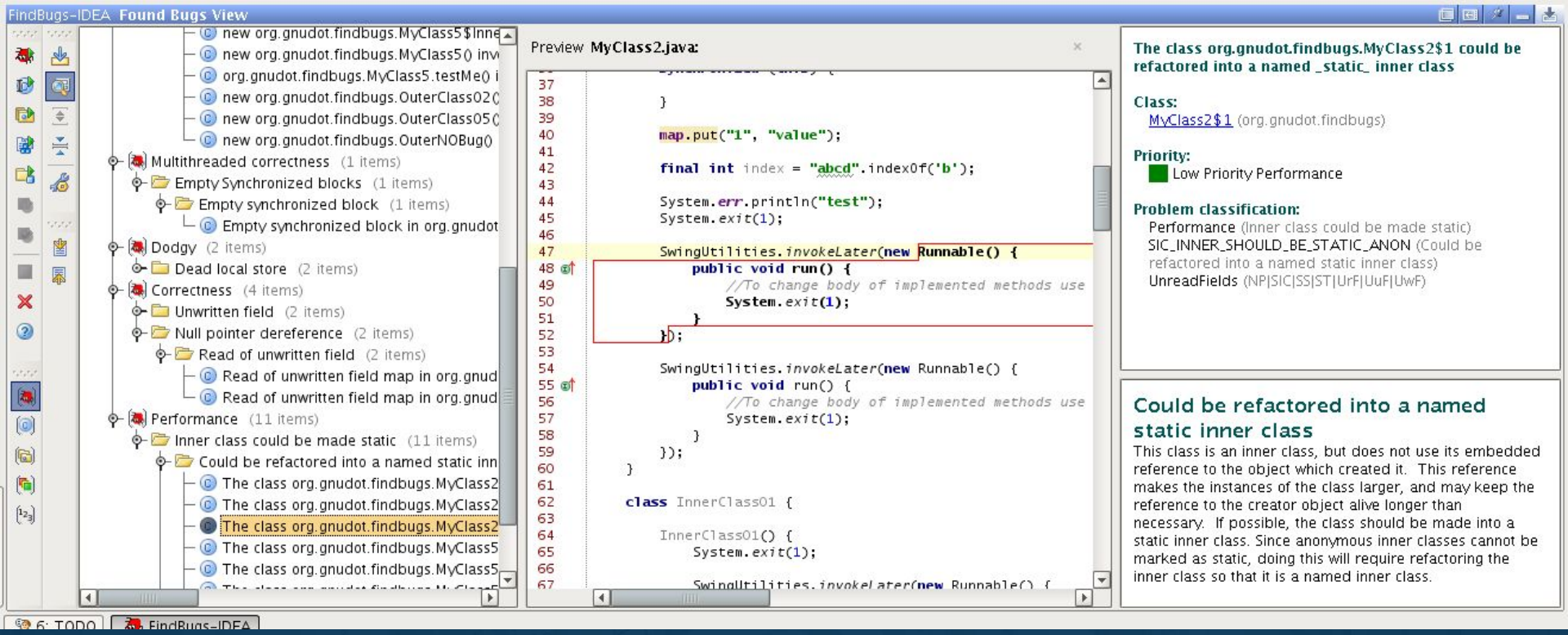

sonarqube .com
Dashboards
Issues
Measures
Rules
Quality Profiles
Quality
SimGrid
Issues
Measures
Code
Dashboards
All
Reliability
Security
Maintainability
Duplications
Size
Complexity
Reliability
157
Bugs
4
New Bugs
D
Reliability Rating
Security
2
Vulnerabilities
0
New Vulnerabilities
D
Security Rating

CTSC


sonarqube .com
Dashboards
Issues
Measures
Rules
Quality Profiles
Quality Gates
More
SimGrid
Issues
Measures
Code
Dashboards
Issues
Effort
Type
Bug 0
Vulnerability 0
Code Smell 16
Resolution
Unresolved 16
Fixed 344
False Positive 1
Won't fix 0
Removed 0
Severity
Blocker 16
Minor 722
Critical 210
Info 74
Major 4,998
SimGrid
examples/msg/dht-kademlia/node.c
Do not apply "<<" bitwise operator to a signed operand.
Code Smell Blocker Open Not assigned 30min effort
Do not apply "^" bitwise operator to a signed operand.
Code Smell Blocker Open Not assigned 30min effort
SimGrid
examples/msg/dht-pastry/dht-pastry.c
Do not apply ">>" bitwise operator to a signed operand.
Code Smell Blocker Open Not assigned 30min effort
Do not apply "&" bitwise operator to a signed operand.
Code Smell Blocker Open Not assigned 30min effort

CTSC

Static Analysis Plugins: e.g. IntelliJ IDEA + FindBugs



(We will re-visit static analysis plugins for IDEs in the Tools section)

CTSC

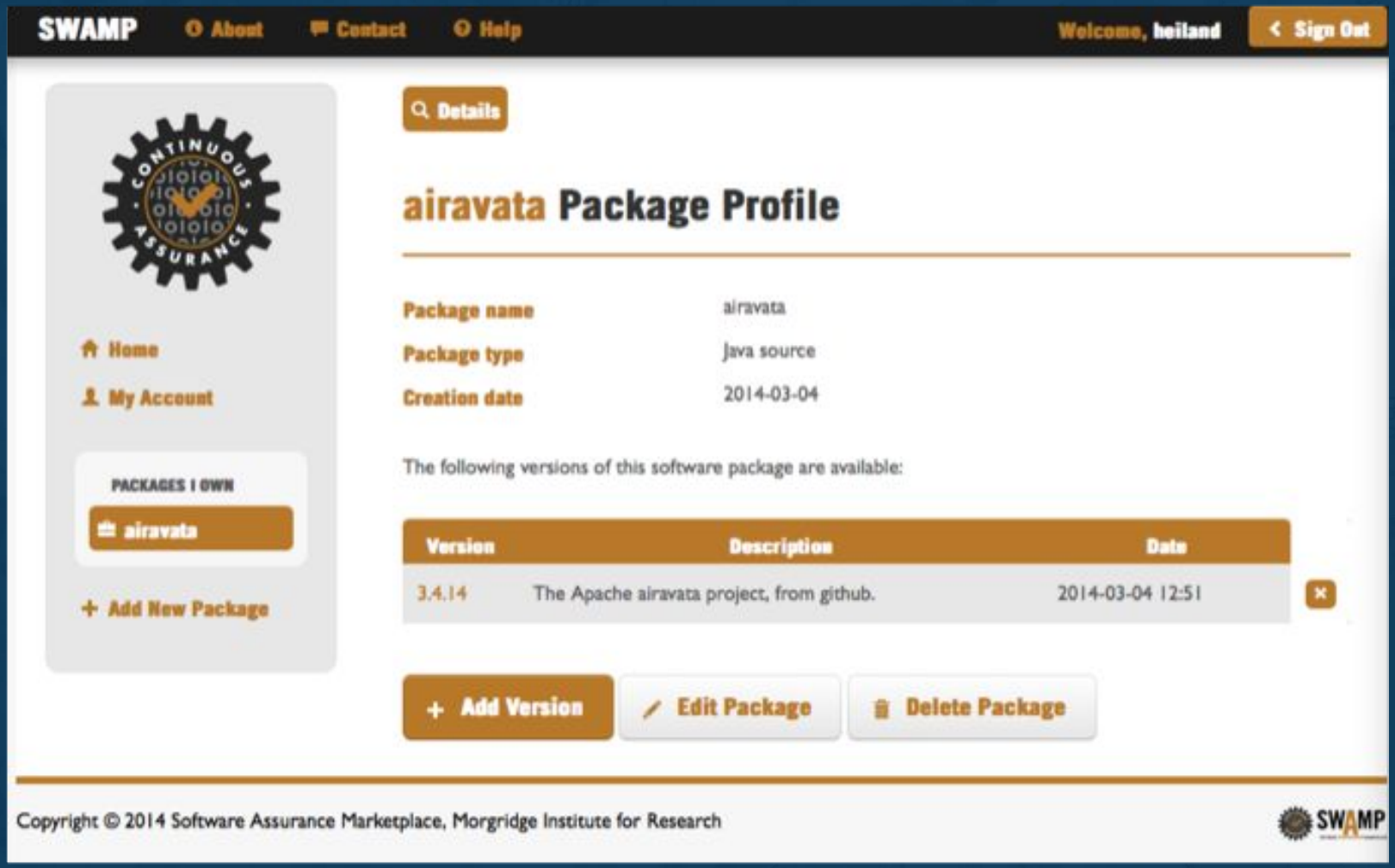
Static analysis as a service: SWAMP

SWAMP - SoftWare Assurance MarketPlace

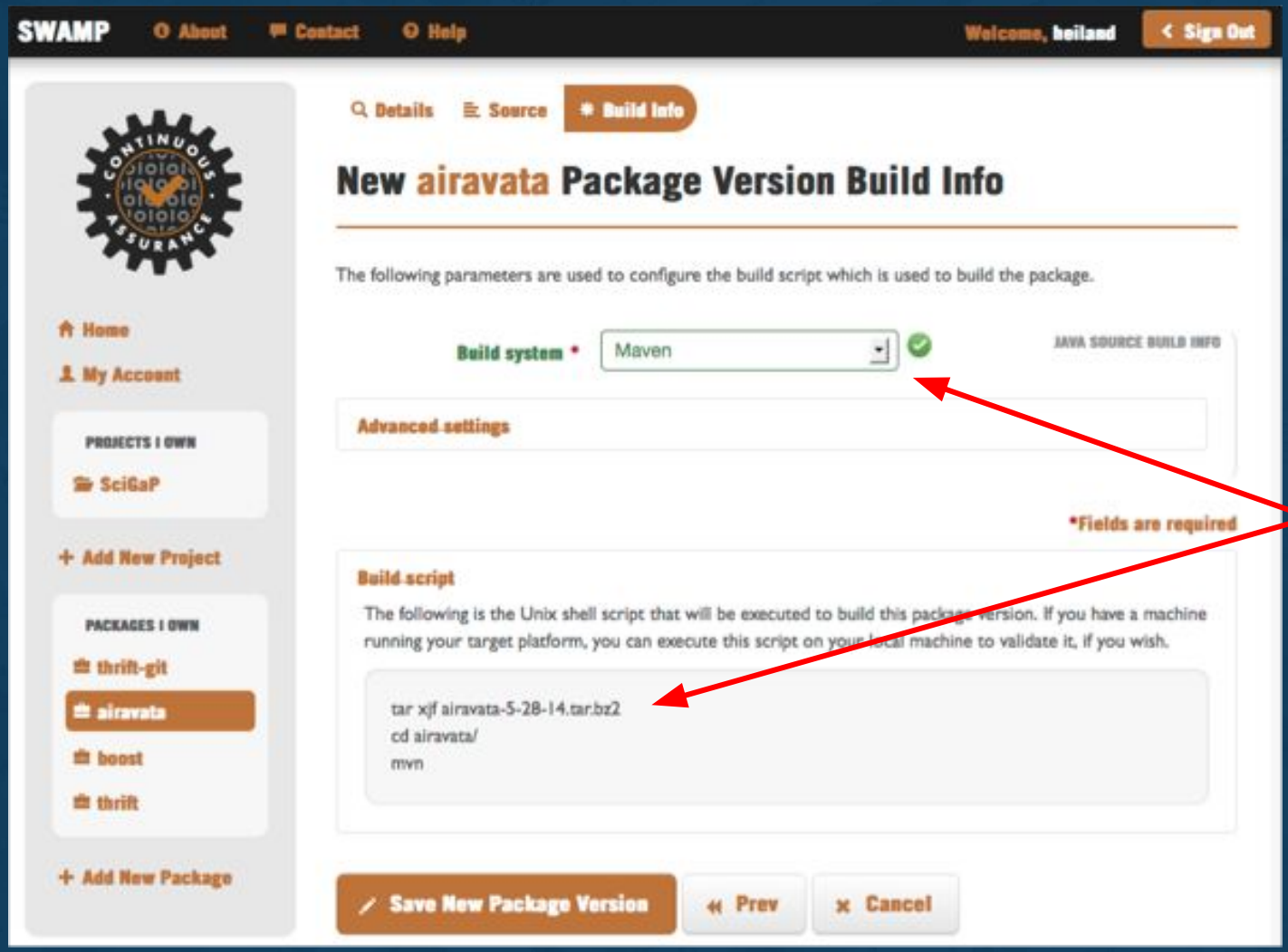
https://continuousassurance.org/

CTSC

Example: Upload, Build, Analyze



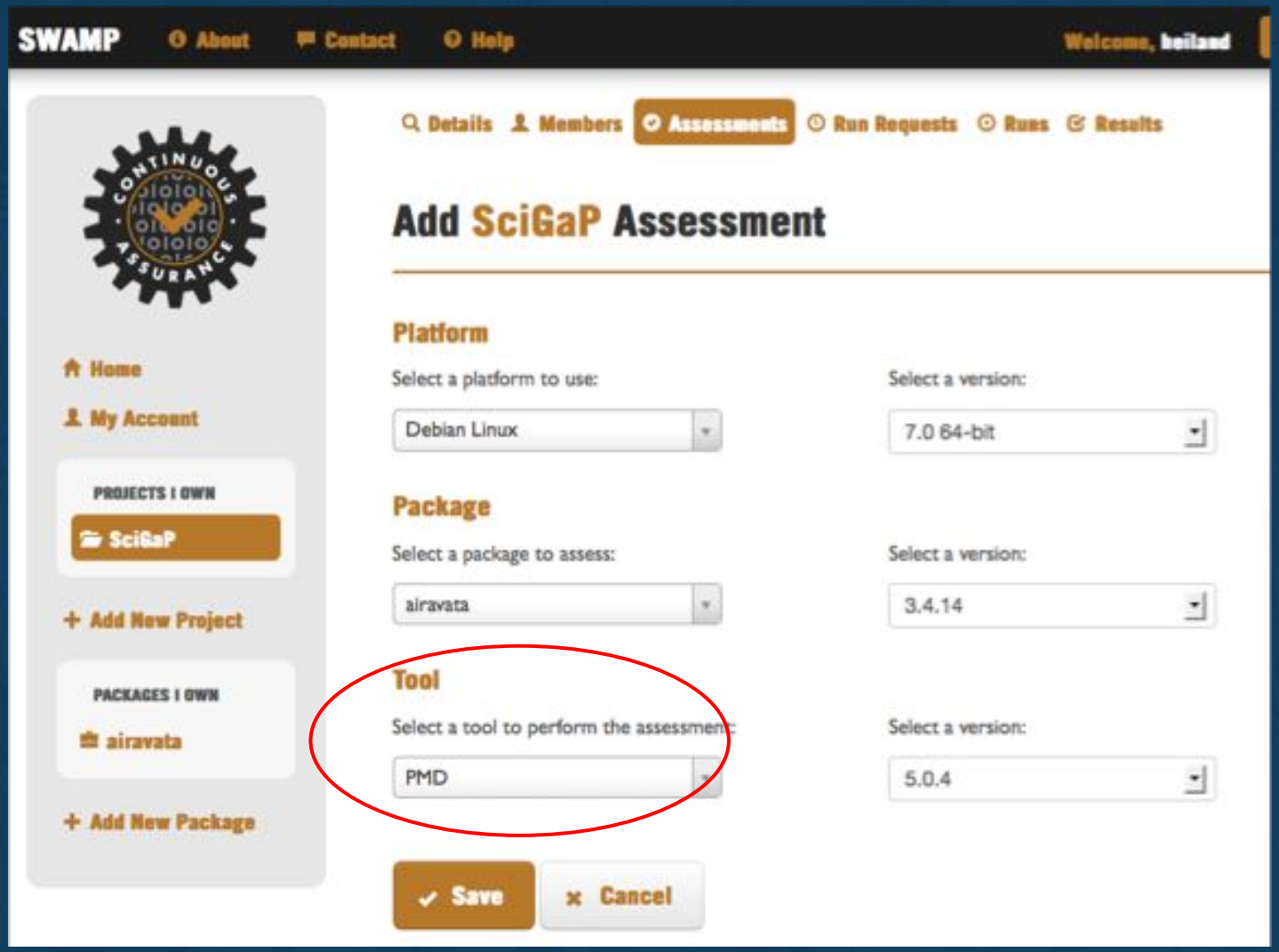
Example: Upload, Build, Analyze



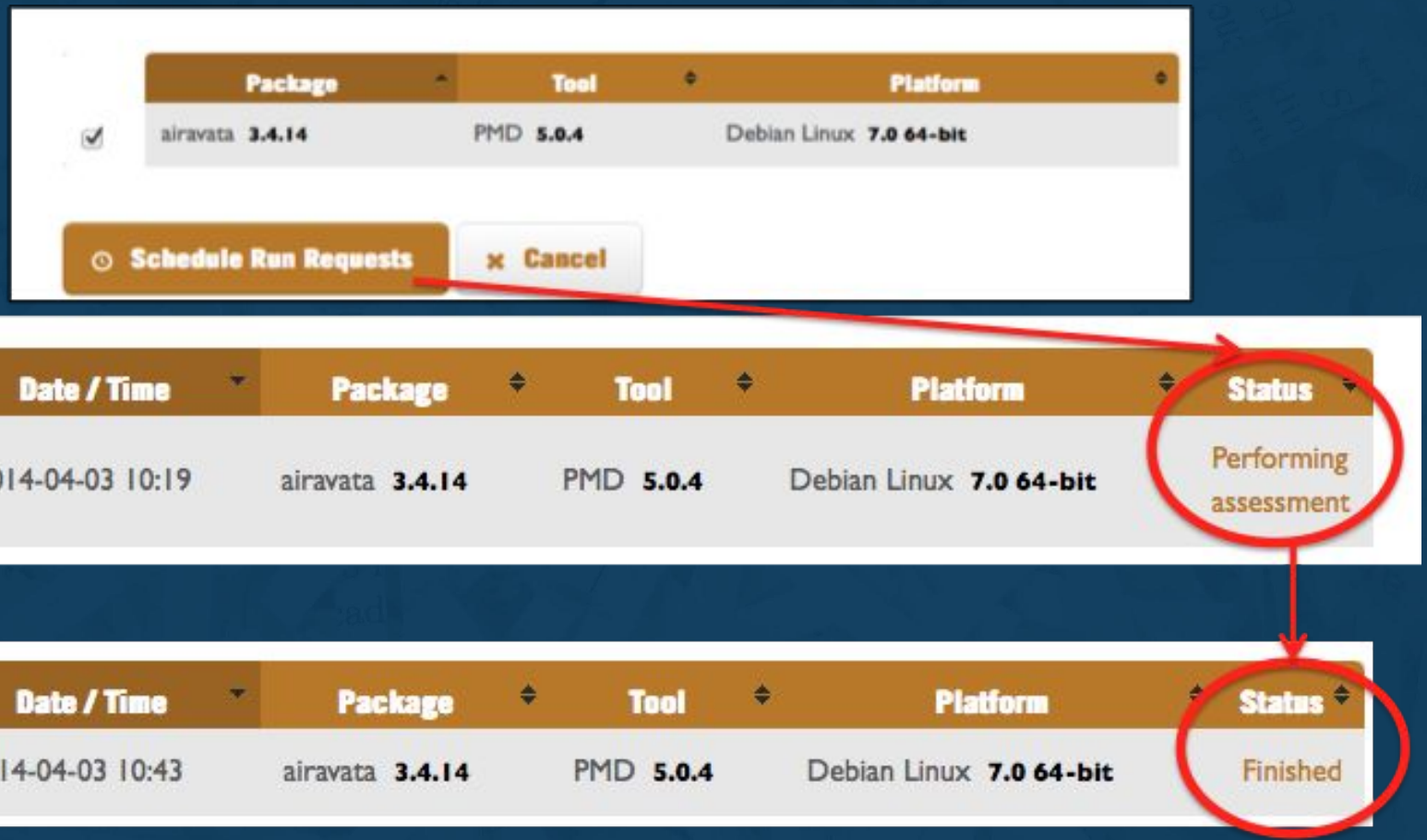
Describe build process

CTSC

Example: Upload, Build, Analyze



Example: Upload, Build, Analyze



CTSC

Examples of potential vulnerabilities

- CWE-398: Indicator of Poor Code Quality
- CWE-547: Use of Hard-coded, Security-relevant Constants
- CWE-252: Unchecked Return Value
- CWE-571: Expression is Always True
- CWE-581: Object Model Violation: Just One of Equals and Hashcode Defined
- CWE-584: Return Inside Finally Block
- CWE-563: Assignment to Variable without Use ('Unused Variable')
- CWE-478: Missing Default Case in Switch Statement
- CWE-495: Private Array-Typed Field Returned From A Public Method

cwe.mitre.org - Common Weakness Enumeration:
a dictionary of software weakness types.

For more in-depth details, see “secure coding” related slides at:

http://trustedci.org/trainingmaterials/

Further reading

- Johnson, B., Y. Song, E. Murphy-Hill, and R. Bowdidge. 2013. “Why Don’t Software Developers Use Static Analysis Tools to Find Bugs?” In *Proceedings of the 2013 International Conference on Software Engineering*, 672–81. ICSE ’13. Piscataway, NJ, USA: IEEE Press.
- Kupsch, J. A., E. Heymann, B. Miller, and V. Basupalli. 2016. “Bad and Good News about Using Software Assurance Tools.” *Software: Practice & Experience*, doi:10.1002/spe.2401 . http://onlinelibrary.wiley.com/doi/10.1002/spe.2401/full
- http://trustedci.org/trainingmaterials/

Secure SwEng BP Topics

- Repositories
- Testing
- Static Analysis
- Vulnerability Management
- Release & Delivery
- Coding/Project Tools
- Documentation

Vulnerability Management

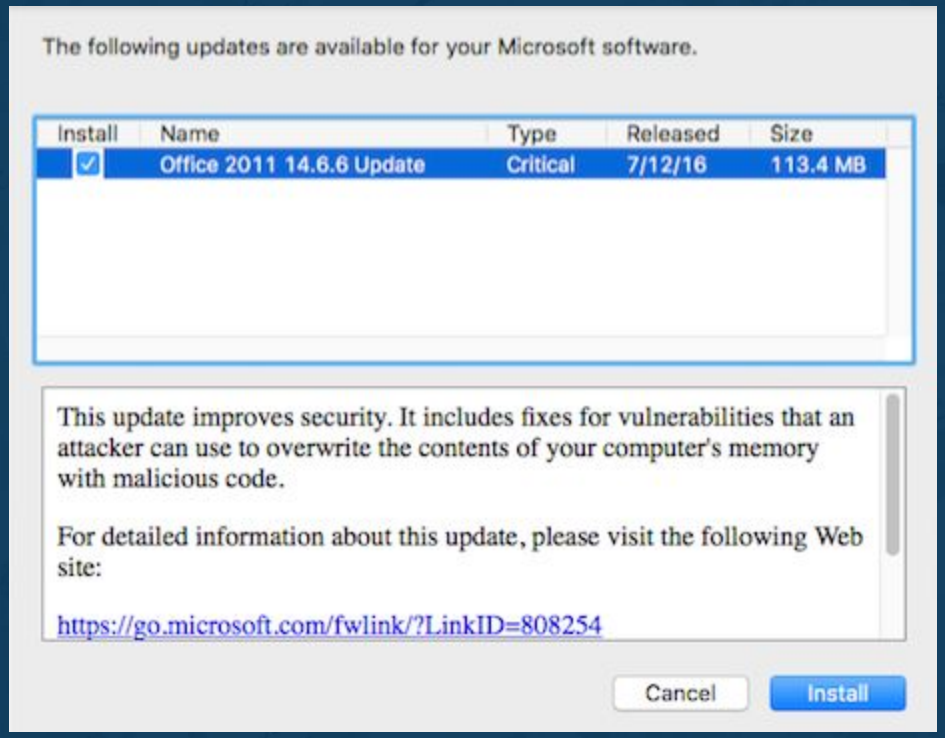
Phases of vulnerability management (after one has been found):

- Notifying* appropriate people
- Fixing/Patching
- Testing
- Communicating* fix

patch: a software update that can be applied to an existing code base in order to eliminate one or more vulnerabilities.

* responsibly, hopefully

Vulnerability Management



Wouldn't it be great if it was this simple?

CTSC

Vulnerability Management

It can be complicated:

- software dependencies
- complex configuration
- mission-critical uptime
- difficult to reach resources
- what else?

Further reading

- http://blog.trustedci.org/search/label/vulnerabilities
- https://www.sans.org/reading-room/whitepapers/application/building-application-vulnerability-management-program-35297
- http://www.pcworld.com/article/2059580/opensource-software-projects-need-to-improve-vulnerability-handling-practices-researchers-say.html
- https://www.apache.org/security/committers.html
- https://blog.jupyter.org/2016/08/03/security-fix-notebook-4-2-2/
- https://www.debian.org/doc/manuals/securing-debian-howto/ch7.en.html

Secure SwEng BP Topics

- Repositories
- Testing
- Static Analysis
- Vulnerability Management
- Release & Delivery
- Coding/Project Tools
- Documentation

Release & Delivery

How can one help ensure the authenticity and integrity of software (and data)?

- cryptographic checksums, hashes
- SHA-{1,2,3} (Secure Hash Alg) …
- digital signatures (e.g., GPG)

1) Download a file
2) Compute a hash on it
3) Compare to published hash

Simple example

```
#include <stdio.h>
int main()
{
  printf("hello, world\n");
}
```

$ md5 hello.c

MD5 (hello.c) = 86d1a675a06b1ea6e7ddc90e79153cdf

---------- edit hello.c and add another blank space before ‘world’

$ md5 hello.c

MD5 (hello.c) = 3a0e40763afa9337c5275c4e70a86943

$ shasum -a 256 hello.c

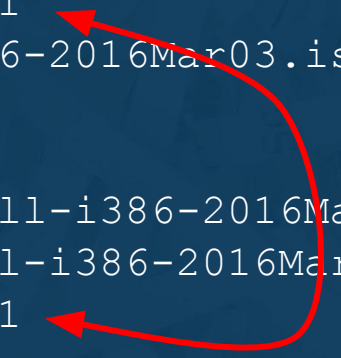
f5f3cff1beb5cfb9b9be6702c0da3964c996b78c4e1db286a96712a8bd37ef47 hello.c

Example: MD5

pS-Toolkit-3.5.1-NetInstall-i386-2016Mar03.iso	03-Mar-2016 15:15	225M
pS-Toolkit-3.5.1-NetInstall-i386-2016Mar03.iso.md5	03-Mar-2016 15:15	81

Verify validity:

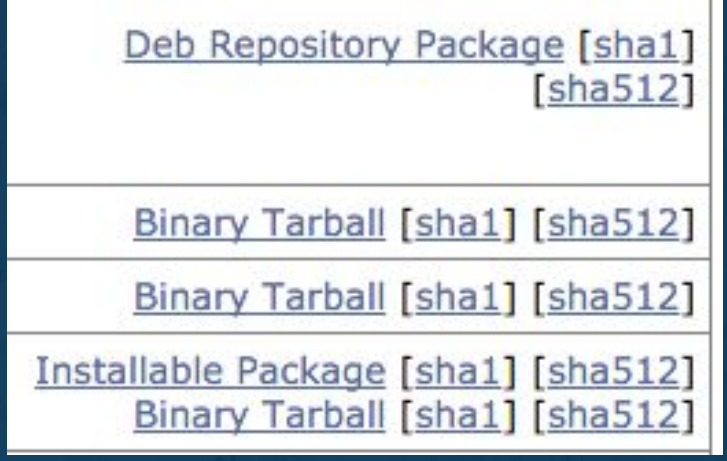
```
$ more pS-Toolkit-3.5.1-NetInstall-i386-2016Mar03.iso.md5
bfa2972732fe2a04abea1de368cdae61
pS-Toolkit-3.5.1-NetInstall-i386-2016Mar03.iso


$ md5 pS-Toolkit-3.5.1-NetInstall-i386-2016Mar03.iso
MD5 (pS-Toolkit-3.5.1-NetInstall-i386-2016Mar03.iso) =
bfa2972732fe2a04abea1de368cdae61
```



R. Rivest, The MD5 Message-Digest Algorithm , RFC Editor, 1992

You will likely encounter codes with MD5 hashes.
But, do NOT use MD5 for your own code/data.

SHA-Secure Hash Alg.



http://toolkit.globus.org/.../globus_toolkit-6.0.1453307864.pkg.sha1

39e9fb34c8dd3f9025ffbe21392e9b071ac57c36

http://toolkit.globus.org/.../globus_toolkit-6.0.1453307864.pkg.sha512

4bea23ea575cd1924b0843699c5bb74ff137410d8bb5bd1f01b3c46530981bc97d2b162
716bd0dfdb373e95e63af05d199e31dacebd1013620d0dfb11c2d2719

-------- verify after downloading:

$ shasum globus_toolkit-6.0.1453307864.pkg # defaults to SHA1

39e9fb34c8dd3f9025ffbe21392e9b071ac57c36

$ shasum -a 512 globus_toolkit-6.0.1453307864.pkg

4bea23ea575cd1924b0843699c5bb74ff137410d8bb5bd1f01b3c46530981bc97d2b162
716bd0dfdb373e95e63af05d199e31dacebd1013620d0dfb11c2d2719

CTSC

SHA-n: how long to compute?

```
$ time shasum -a 1 globus_toolkit-6.0.1453307864.pkg
39e9fb34c8dd3f9025ffbe21392e9b071ac57c36  globus_toolkit-6.0.1453307864.pkg

real  0m0.073s
user 0m0.059s
sys   0m0.010s

$ time shasum -a 512 globus_toolkit-6.0.1453307864.pkg
4bea23ea575cd1924b0843699c5bb74ff137410d8bb5bd1f01b3c46530981bc97d2b162
716bd0dfdb373e95e63af05d199e31dacebd1013620d0dfb11c2d2719
globus_toolkit-6.0.1453307864.pkg

real  0m0.100s
user 0m0.085s
sys   0m0.011s
```

~10M in size

Digital signatures: e.g. GPG

A digital signature certifies and timestamps a document. If the document is subsequently modified in any way, a verification of the signature will fail.

“signature” via private key.

GPG (Gnu Privacy Guard):

Free implementation of the OpenPGP standard (www.ietf.org/rfc/rfc4880.txt)

Further reading

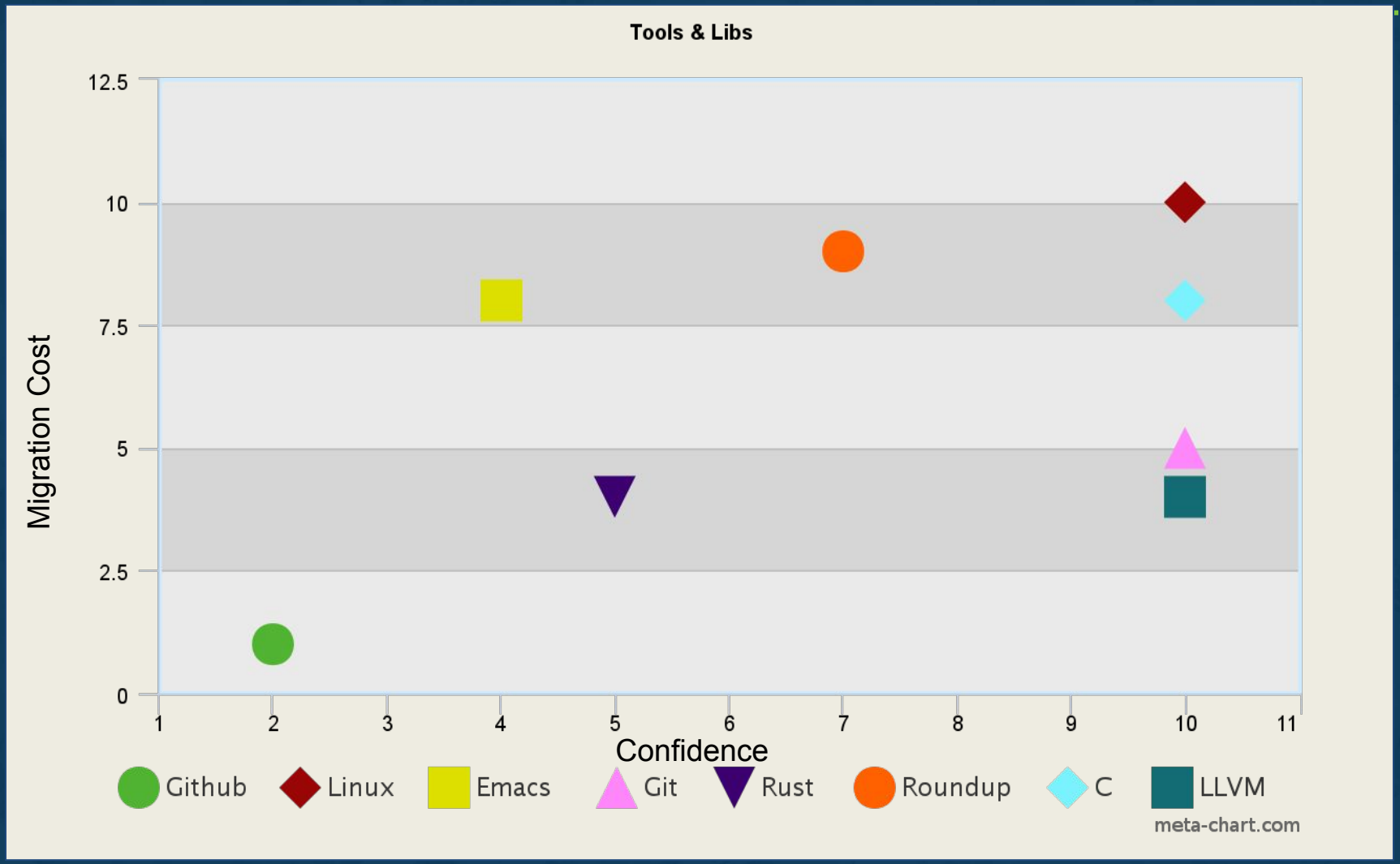
- https://www.apache.org/dev/release-signing (PGP signatures)
- https://www.gnupg.org/faq/gnupg-faq.html
- http://oss-watch.ac.uk/resources/releasemanagementbestpractice (PGP signatures, hashes)
- http://blog.sonatype.com/2010/01/how-to-generate-pgp-signatures-with-maven/ (Maven central repo requires PGP signatures)
- https://www.schneier.com/blog/archives/2012/10/keccak_is_sha-3.html
- https://git-scm.com/book/en/v2/Git-Tools-Signing-Your-Work

Secure SwEng BP Topics

- Repositories
- Testing
- Static Analysis
- Vulnerability Management
- Release & Delivery
- Coding/Project Tools
- Documentation

Choosing Tools and Libraries in an Imperfect World

Confidence vs. Migration Cost



CTSC

Qualities that inspire confidence in tools and libraries:

- Resources appropriate to the scope and complexity of the project.
- Adoption/dependance by players capable of resourcing the project if it is in trouble.
- License that facilitates forking should the project be mishandled by or lose the interest of current maintainers.
- Maturity of software development practices (behaviors we're teaching in this training).
- Quality of architecture and maintainability of code.

Qualities that reduce migration cost from tools and libraries:

- Ability to get copies of data (if you aren't already self-hosting).
- Open, standard data formats (where applicable).
- Use of standard protocols and interfaces.
- Tool/library criticality to your projects (trivial use is trivial to give up)

Coding/Project Tools

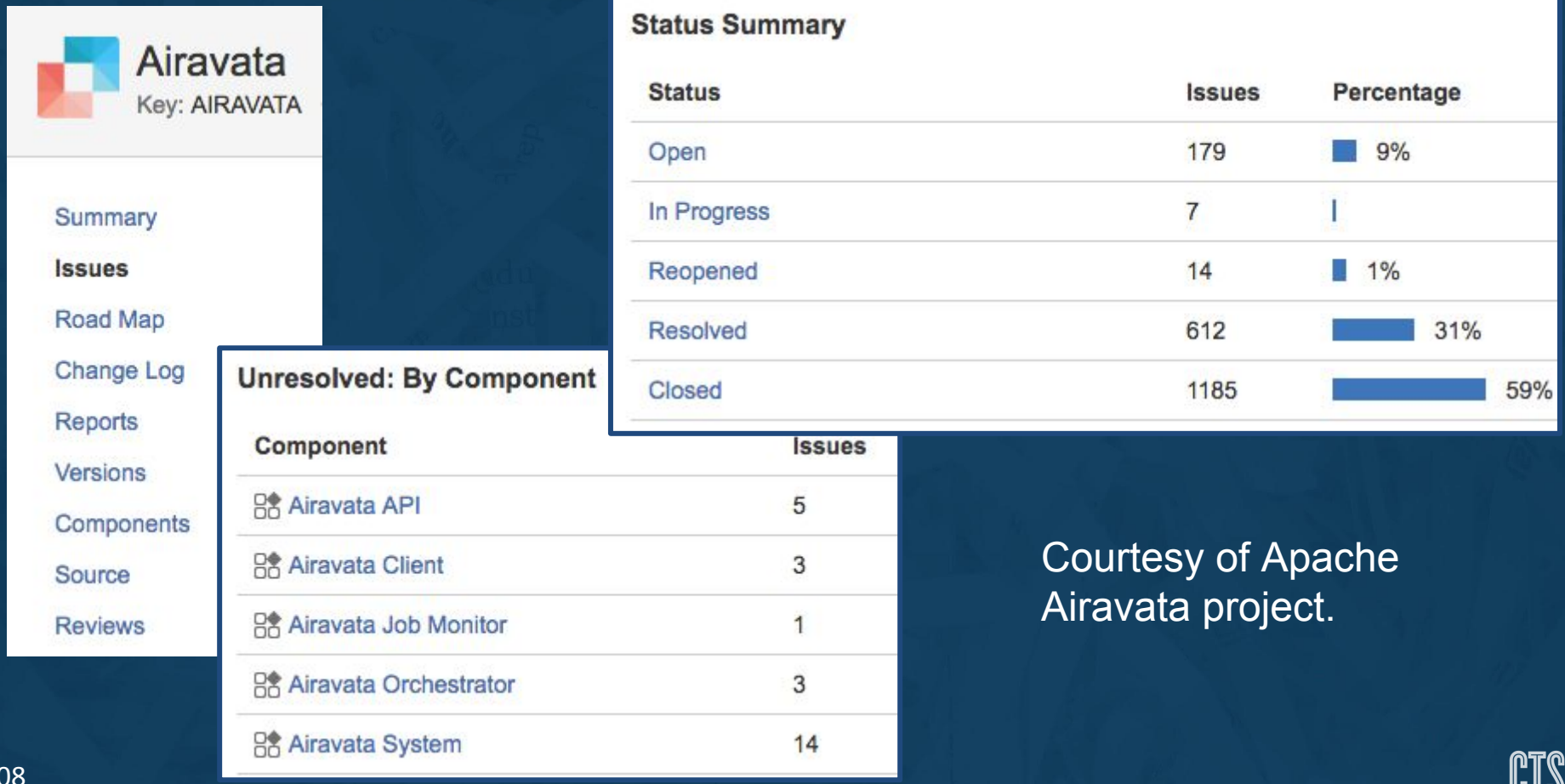
Tools for writing/modifying/maintaining/managing code and overall software project.

How do coding/project tools help improve software security?

- convenience of integrated functionality:
  - code navigation, repo access, debugging, etc.
- community reviewing
- integrated static analysis
- issue tracking: prioritize, assign responsibility

Project and Issue Tracking tools

e.g. JIRA (https://www.atlassian.com/software/jira)



Courtesy of Apache Airavata project.

CTSC

Issue Tracking & Vuln Patching

For a public project with issue tracking, there needs to be a mechanism to keep certain issues private, e.g., vulnerability patches until they are ready for release.

Experiences?

Continuous Integration

Continuous Integration (CI), which some people think is a relatively new concept, is actually not so new:

Grady Booch, *Object-Oriented Analysis and Design with Applications, 2nd Ed, 1993.*

"Individual developers can create their own stable release into which they integrate new versions of the software for which they are responsible, before releasing it to the rest of the team. In this manner, we have a platform for continuous integration of new code."

CI in the Cloud

Continuous Integration as a cloud service is newer.

Some popular CI tools include:

- Travis - travis-ci.org (limited to github)
- Bamboo - www.atlassian.com/software/bamboo
- Jenkins - jenkins.io

Meyer, M. 2014. "Continuous Integration and Its Tools." *IEEE Software* 31 (3): 14–16.

Travis CI (Linux & OSX)

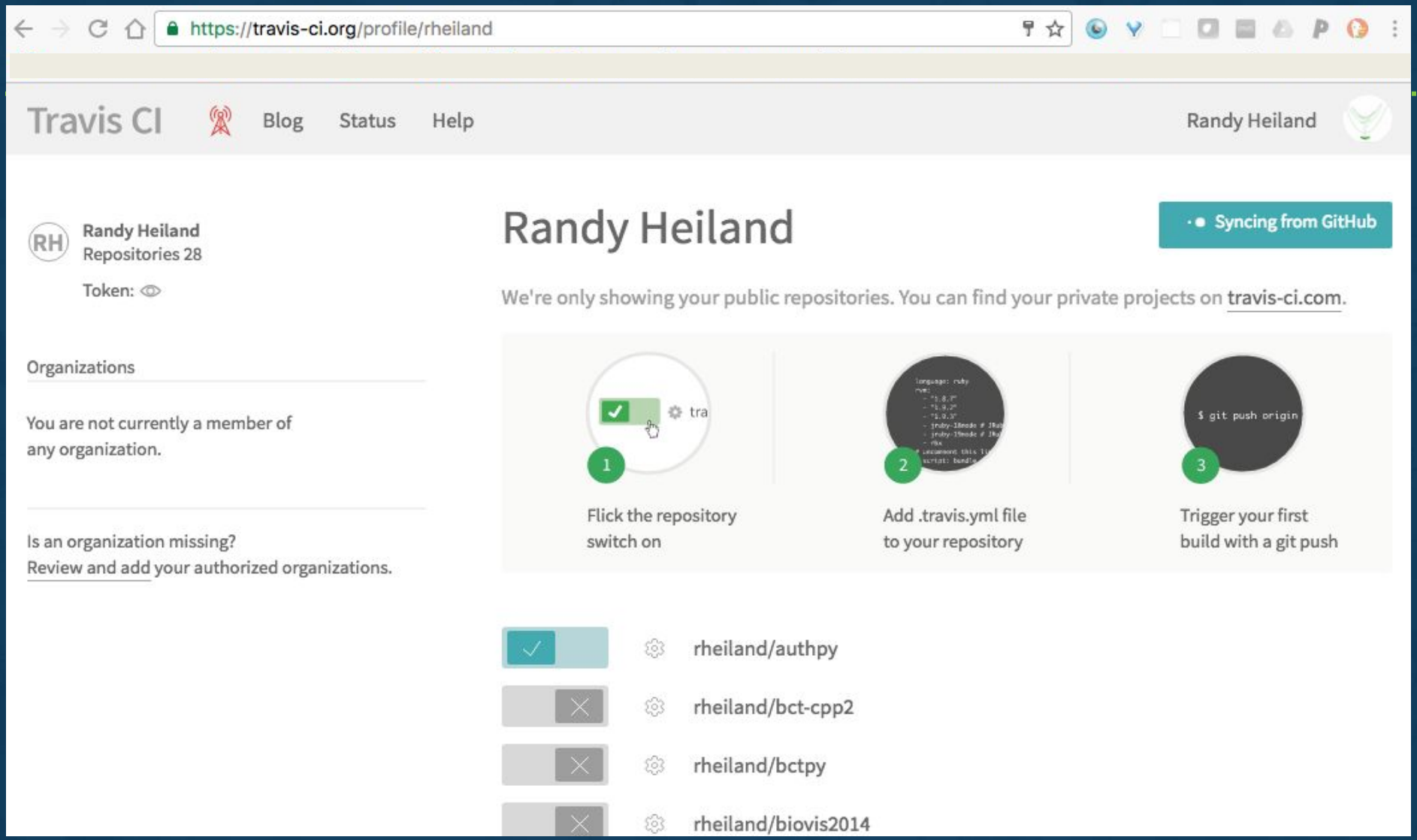
- a hosted CI service
- integrates with GitHub
- free for open source projects
- “easy to use”

Basic idea:

- Allow Travis CI to access your github repo
- Create a .yml file to describe your build

https://docs.travis-ci.com/user/customizing-the-build/

- A “push” will auto-generate a build


https://travis-ci.org/profile/rheiland
Travis CI
Blog
Status
Help
Randy Heiland
RH
Randy Heiland
Repositories 28
Token:
Organizations
You are not currently a member of any organization.
Is an organization missing?
Review and add your authorized organizations.
Randy Heiland
Syncing from GitHub
We're only showing your public repositories. You can find your private projects on travis-ci.com.
1
Flick the repository switch on
2
Add .travis.yml file to your repository
3
$ git push origin
Trigger your first build with a git push
rheiland/authpy
rheiland/bct-cpp2
rheiland/bctpy
rheiland/biovis2014

AppVeyor (Windows)

- free for open source projects
- www.appveyor.com
- software as a service

Continuous Integration: Pegasus WMS

Each commit:

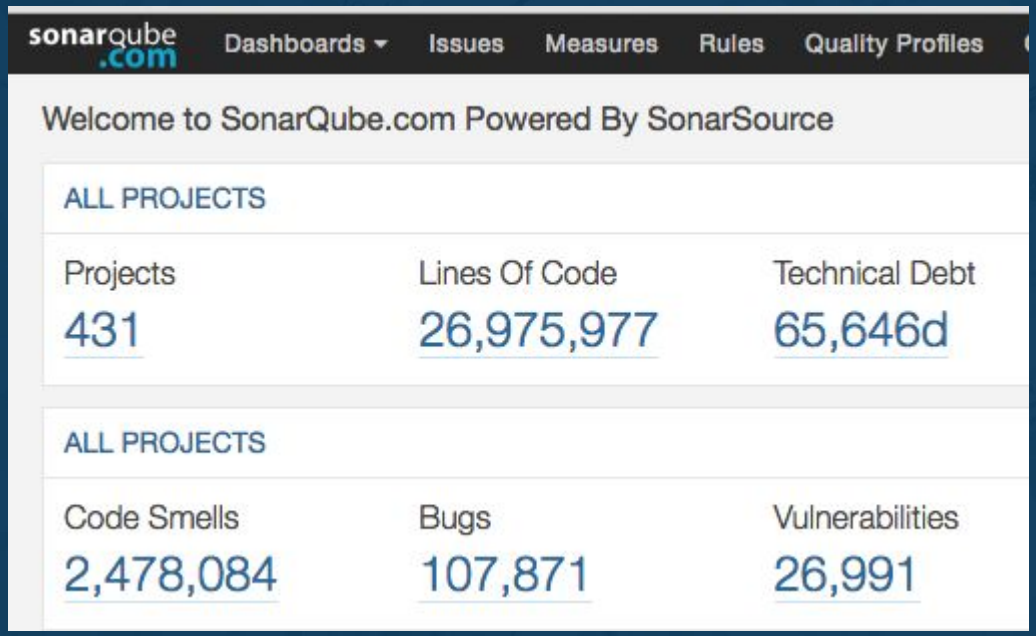
- triggers a build of the current dev branch. This results in documentation and rpm, deb and binary packages.
- triggers units tests of the various components.

Nightly: end to end workflow tests (workflows are for the last major release branch and the current dev branch).

https://github.com/pegasus-isi/pegasus

https://www.atlassian.com/software/bamboo

Continuous Integration & Quality Control



The SonarQube® platform is an open source quality management platform, dedicated to continuously analyzing and measuring the technical quality of source code, from project portfolio down to the method level, and tracking the introduction of new Bugs, Vulnerabilities, and Code Smells in the Leak Period.

CTSC

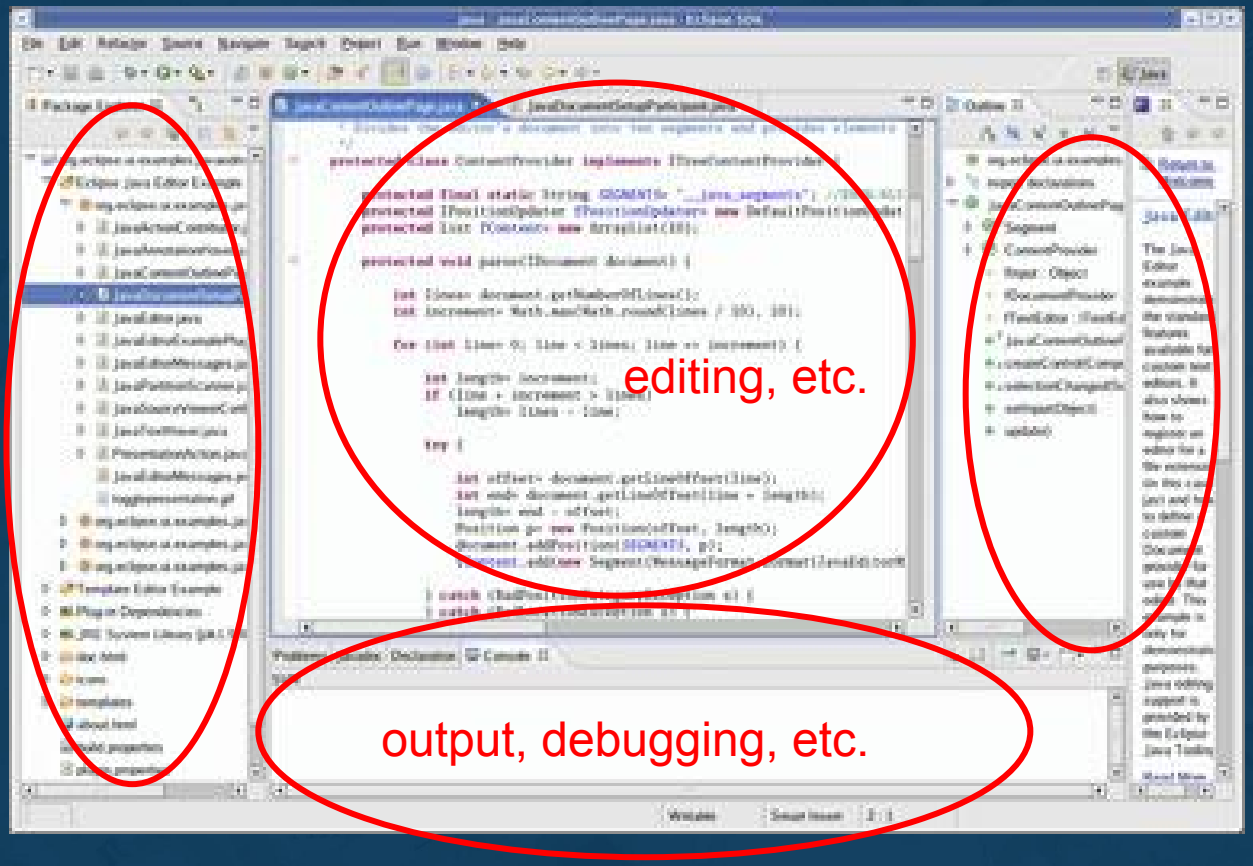
Integrated Dev Environments (IDEs)

Integrated Dev Environments (IDEs)

- Convenience of integrated functionality:
  - editing
  - debugging
  - profiling
  - testing
  - repos/version control
- Most allow 3rd party plugins:
  - static analysis
  - memory checking
  - …

IDEs are just another tool.

Generic IDE layout

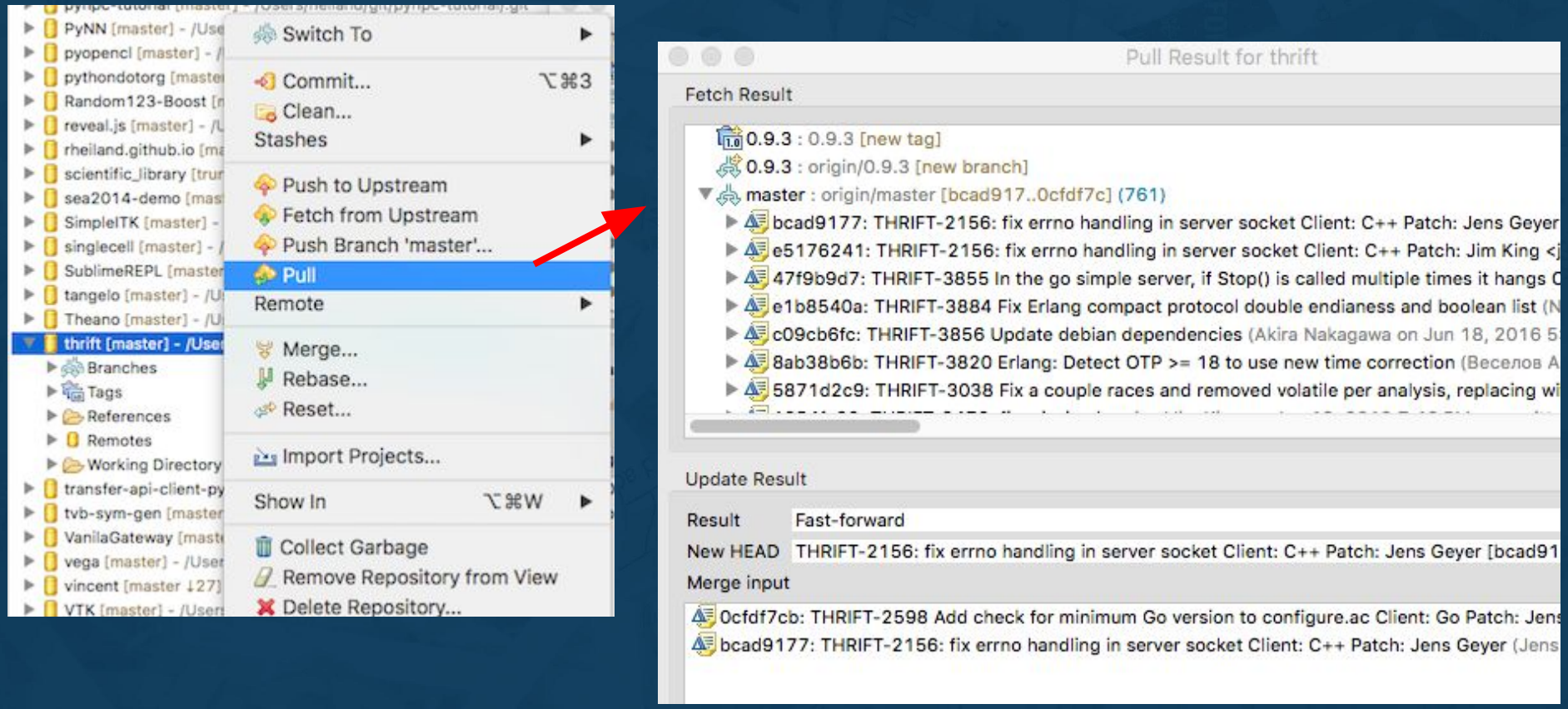


"biggest advantage of IDE is debugging is easier, and code navigation is one click" (developer w/ a CTSC engagement)

CTSC

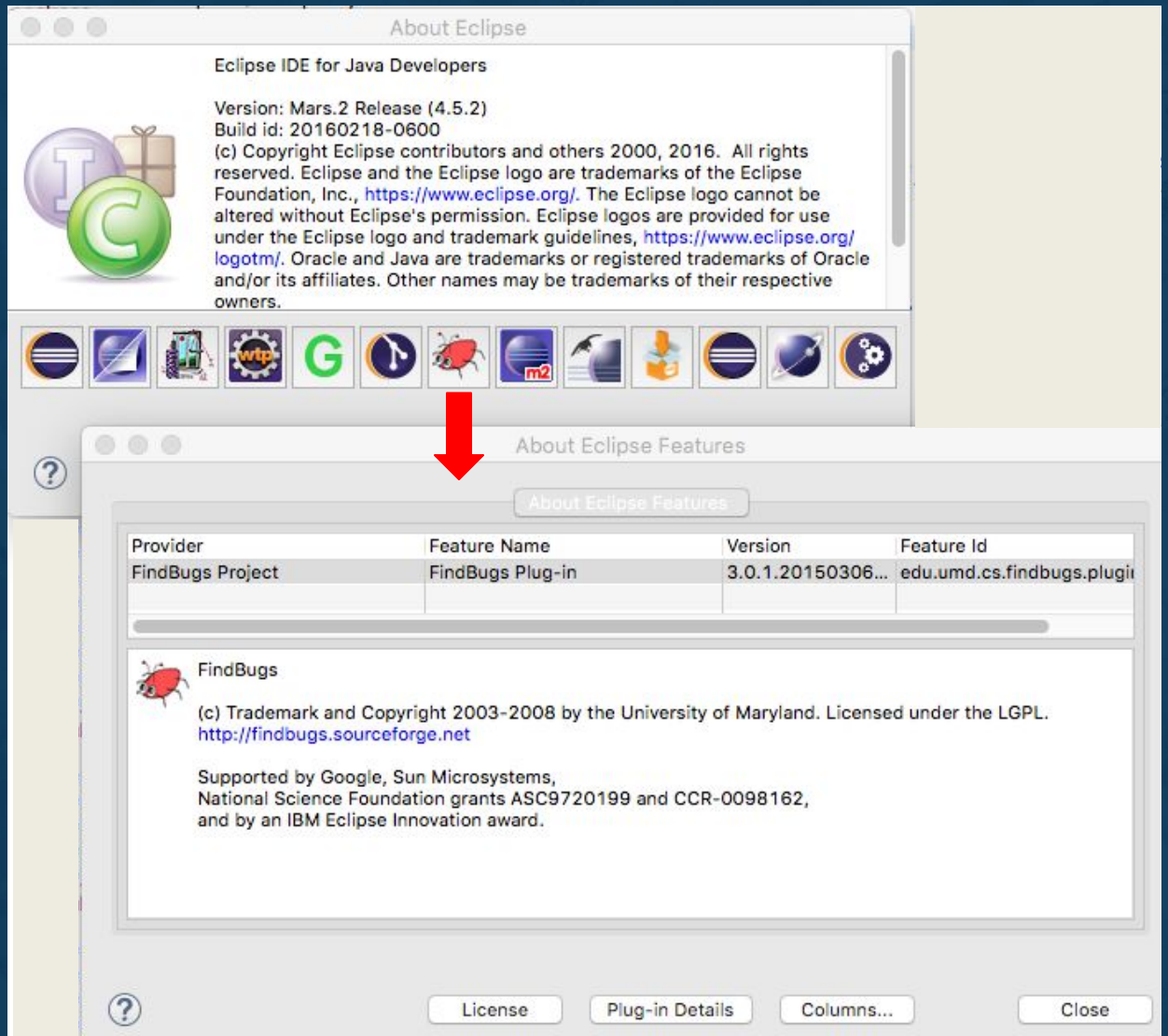
IDEs: mostly GUI-driven

E.g. git perspective



CTSC

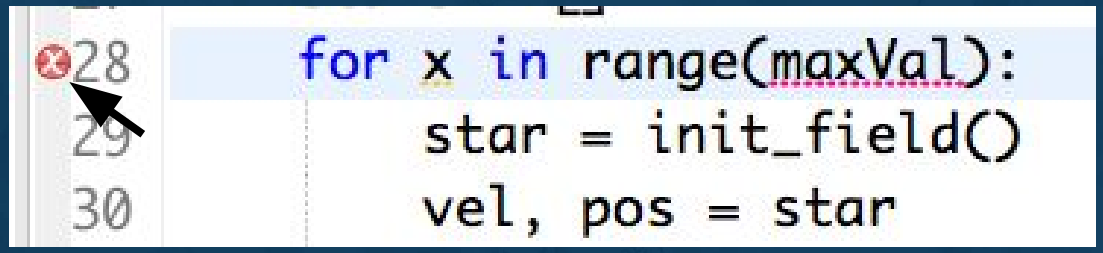
IDEs: most allow 3rd party plugins



e.g., FindBugs: static analysis for Java code

IDEs: Warning/Error highlighting

warning: reserved symbol



error: undefined variable

IDEs: both OSS and commercial

- Eclipse
- XCode
- Visual Studio
- IntelliJ
- NetBeans
- Nuclide - javascript
- ...
- https://www.jetbrains.{com,org}
- https://pypl.github.io/IDE.html

many have support for multiple languages

IDEs continue to evolve

"Jupyter web-based IDEs… teaching Data Science course with 500 students, largely freshmen, entirely through Jupyter running on Azure and campus-based servers. I find these platforms significantly lower the bar of getting students up and running without installing software on heterogeneous & under-powered machines…"

- Carl Boettiger, UC Berkeley (comment in ctsc-discuss mailing list)

Further Reading

- https://docs.travis-ci.com/
- https://pypl.github.io/IDE.html - popularity ranking of IDEs

Secure SwEng BP Topics

- Repositories
- Testing
- Static Analysis
- Vulnerability Management
- Release & Delivery
- Coding/Project Tools
- Documentation

Documentation

Document design & purpose, not mechanics.

a) Document interfaces and reasons, not implementations.
b) Refactor code in preference to explaining how it works.
c) Embed the documentation for a piece of software in that software.

Wilson, Greg, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Richard T. Guy, Steven H. D. Haddock, et al. 2014. "Best Practices for Scientific Computing." *PLoS Biology* 12 (1): e1001745. dx.doi.org/10.1371/journal.pbio.1001745

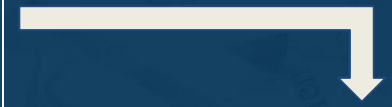
Automatic documentation

Tools exist that generate useful docs for your code if you include that documentation in your code and follow the tools' syntactic rules.

- motivation for embedding your documentation
- generates easy-to-navigate HTML/Latex/etc docs

Javadoc: Generates HTML pages of API documentation from Java source files

```
/**
 * Returns an Image object that can then be painted on the screen.
 * The url argument must specify an absolute {@link URL}. The name
 * argument is a specifier that is relative to the url argument.
 * <p>
 * This method always returns immediately, whether or not the
 * image exists. When this applet attempts to draw the image on
 * the screen, the data will be loaded. The graphics primitives
 * that draw the image will incrementally paint on the screen.
 *
 * @param  url  an absolute URL giving the base location of the image
 * @param  name the location of the image, relative to the url argument
 * @return      the image at the specified URL
 * @see         Image
 */
public Image getImage(URL url, String name) {
    try {
        return getImage(new URL(url, name));
    } catch (MalformedURLException e) {
        return null;
    }
}
```



getImage

public Image getImage(URL url,
String name)

Returns an `Image` object that can then be painted on the screen. The `url` argument must specify an absolute URL. The `name` argument is a specifier that is relative to the `url` argument.

This method always returns immediately, whether or not the image exists. When this applet attempts to draw the image on the screen, the data will be loaded. The graphics primitives that draw the image will incrementally paint on the screen.

Parameters:

`url` - an absolute URL giving the base location of the image.

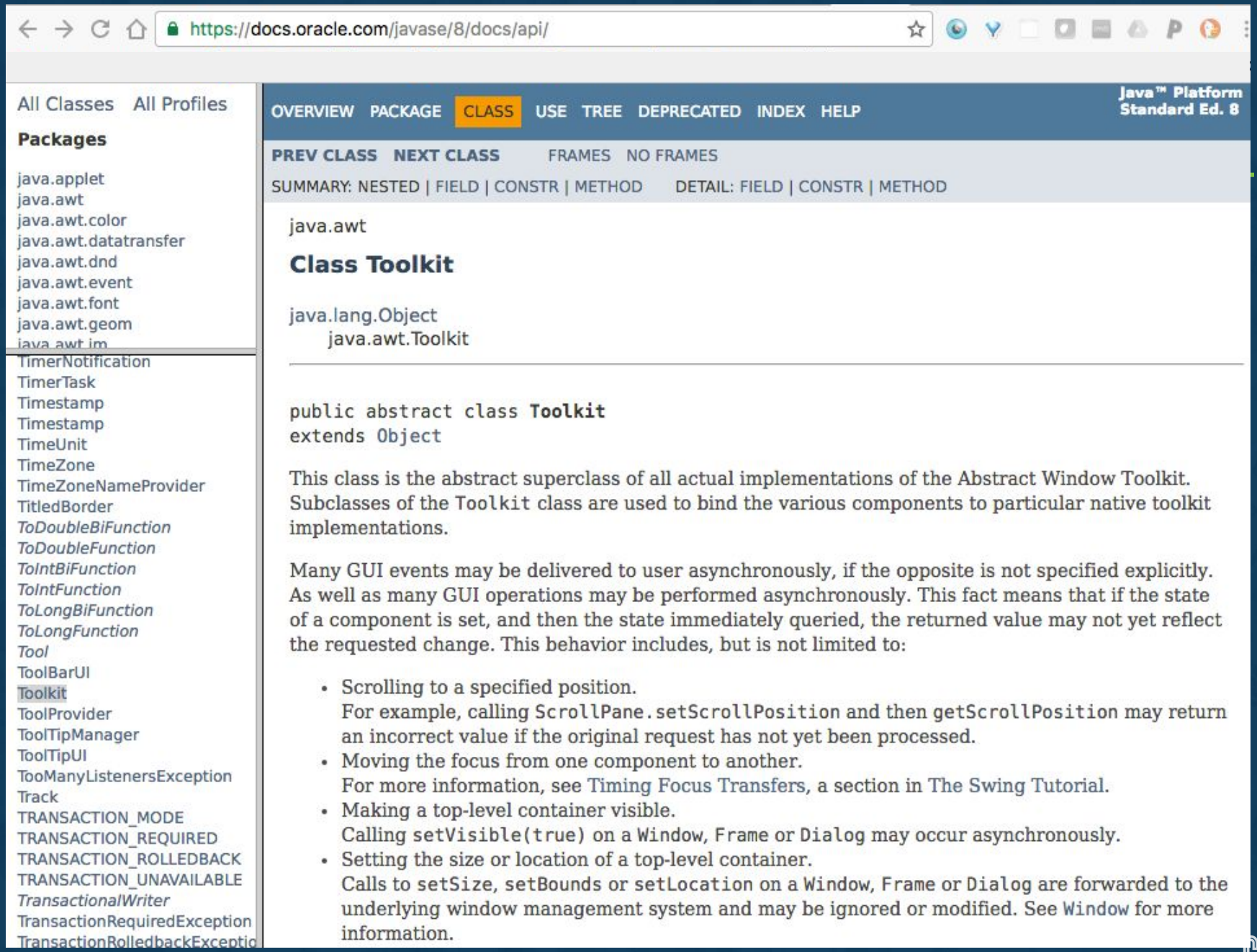
`name` - the location of the image, relative to the `url` argument.

Returns:

the image at the specified URL.

See Also:

`Image`


https://docs.oracle.com/javase/8/docs/api/
All Classes All Profiles
Packages
java.applet
java.awt
java.awt.color
java.awt.datatransfer
java.awt.dnd
java.awt.event
java.awt.font
java.awt.geom
java.awt.im
TimerNotification
TimerTask
Timestamp
Timestamp
TimeUnit
TimeZone
TimeZoneNameProvider
TitledBorder
ToDoubleBiFunction
ToDoubleFunction
ToIntBiFunction
ToIntFunction
ToLongBiFunction
ToLongFunction
Tool
ToolBarUI
Toolkit
ToolProvider
ToolTipManager
ToolTipUI
TooManyListenersException
Track
TRANSACTION_MODE
TRANSACTION_REQUIRED
TRANSACTION_ROLLEDBACK
TRANSACTION_UNAVAILABLE
TransactionalWriter
TransactionRequiredException
TransactionRolledbackExceptio
OVERVIEW PACKAGE CLASS USE TREE DEPRECATED INDEX HELP
Java™ Platform Standard Ed. 8
PREV CLASS NEXT CLASS FRAMES NO FRAMES
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD
java.awt
Class Toolkit
java.lang.Object
java.awt.Toolkit
public abstract class Toolkit
extends Object
This class is the abstract superclass of all actual implementations of the Abstract Window Toolkit. Subclasses of the Toolkit class are used to bind the various components to particular native toolkit implementations.
Many GUI events may be delivered to user asynchronously, if the opposite is not specified explicitly. As well as many GUI operations may be performed asynchronously. This fact means that if the state of a component is set, and then the state immediately queried, the returned value may not yet reflect the requested change. This behavior includes, but is not limited to:
Scrolling to a specified position.
For example, calling ScrollPane.setScrollPosition and then getScrollPosition may return an incorrect value if the original request has not yet been processed.
Moving the focus from one component to another.
For more information, see Timing Focus Transfers, a section in The Swing Tutorial.
Making a top-level container visible.
Calling setVisible(true) on a Window, Frame or Dialog may occur asynchronously.
Setting the size or location of a top-level container.
Calls to setSize, setBounds or setLocation on a Window, Frame or Dialog are forwarded to the underlying window management system and may be ignored or modified. See Window for more information.

Doxygen

"Doxygen is the de facto standard tool for generating documentation from annotated C++ sources, but it also supports other popular programming languages such as C, Objective-C, C#, PHP, Java, Python, IDL, Fortran, VHDL, Tcl, …"

C/C++: Docs annotation is inserted into headers (.h):

```
// .NAME classname - brief description
// .SECTION Description
// more detailed description
…
  // Description:
  // Assign a data object as input. Note that this method ...
  void SetInputData(int index, vtkDataObject* obj);
```

Public Types

typedef vtkAlgorithm Superclass

- Public Types inherited from vtkAlgorithm
- Public Types inherited from vtkObject

Public Member Functions

virtual int	IsA (const char *type)
vtkUndirectedGraphAlgorithm *	NewInstance () const
void	PrintSelf (ostream &os, vtkIndent indent)
virtual int	ProcessRequest (vtkInformation *, vtkInformationVect
vtkUndirectedGraph *	GetOutput ()
vtkUndirectedGraph *	GetOutput (int index)
void	SetInputData (vtkDataObject *obj)
void	SetInputData (int index, vtkDataObject *obj)

- Public Member Functions inherited from vtkAlgorithm
- Public Member Functions inherited from vtkObject
- Public Member Functions inherited from vtkObjectBase

Static Public Member Functions

static vtkUndirectedGraphAlgorithm *	New ()
static int	IsTypeOf (const char *type)
static vtkUndirectedGraphAlgorithm *	SafeDownCast (vtkObjectBase *o)

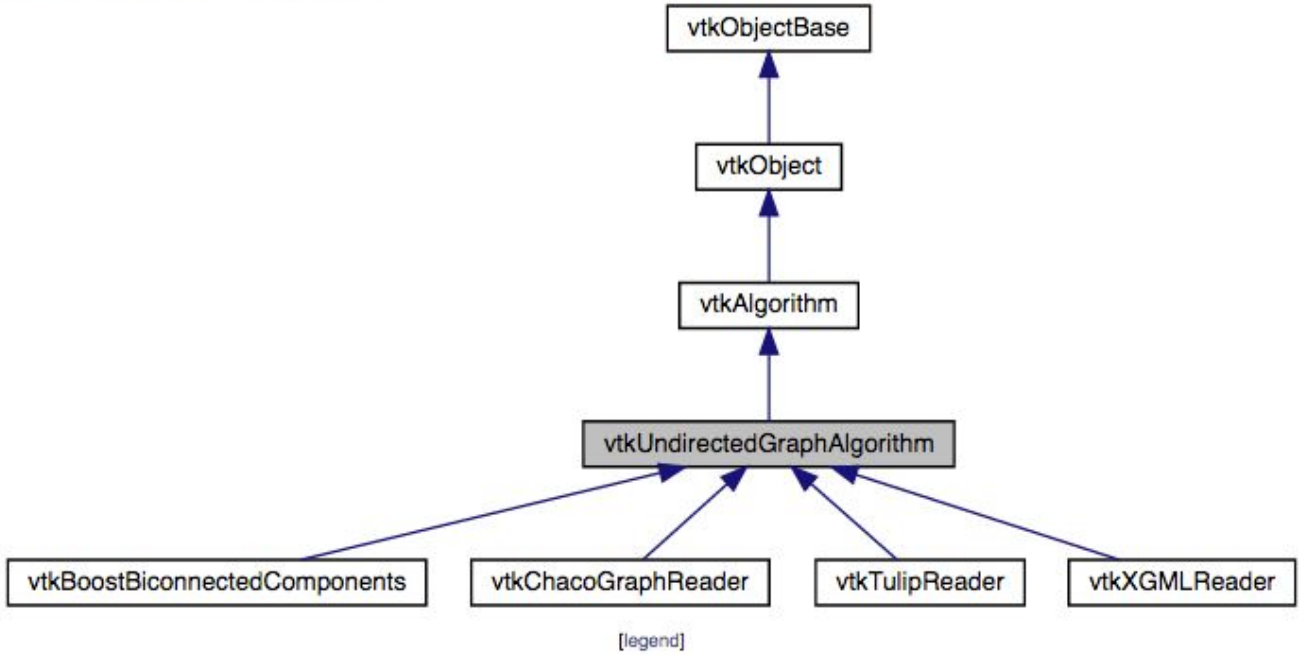
- Static Public Member Functions inherited from vtkAlgorithm

www.vtk.org/doc/release/7.0/html/classvtkUndirectedGraphAlgorithm.html

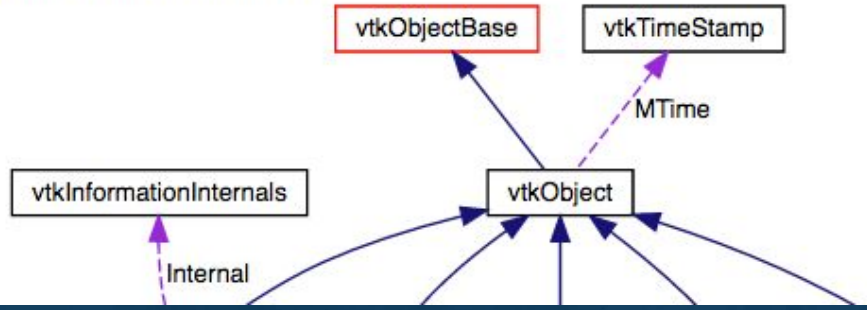
Superclass for algorithms that produce undirected graph as output. More...

`#include <vtkUndirectedGraphAlgorithm.h>`

Inheritance diagram for vtkUndirectedGraphAlgorithm:

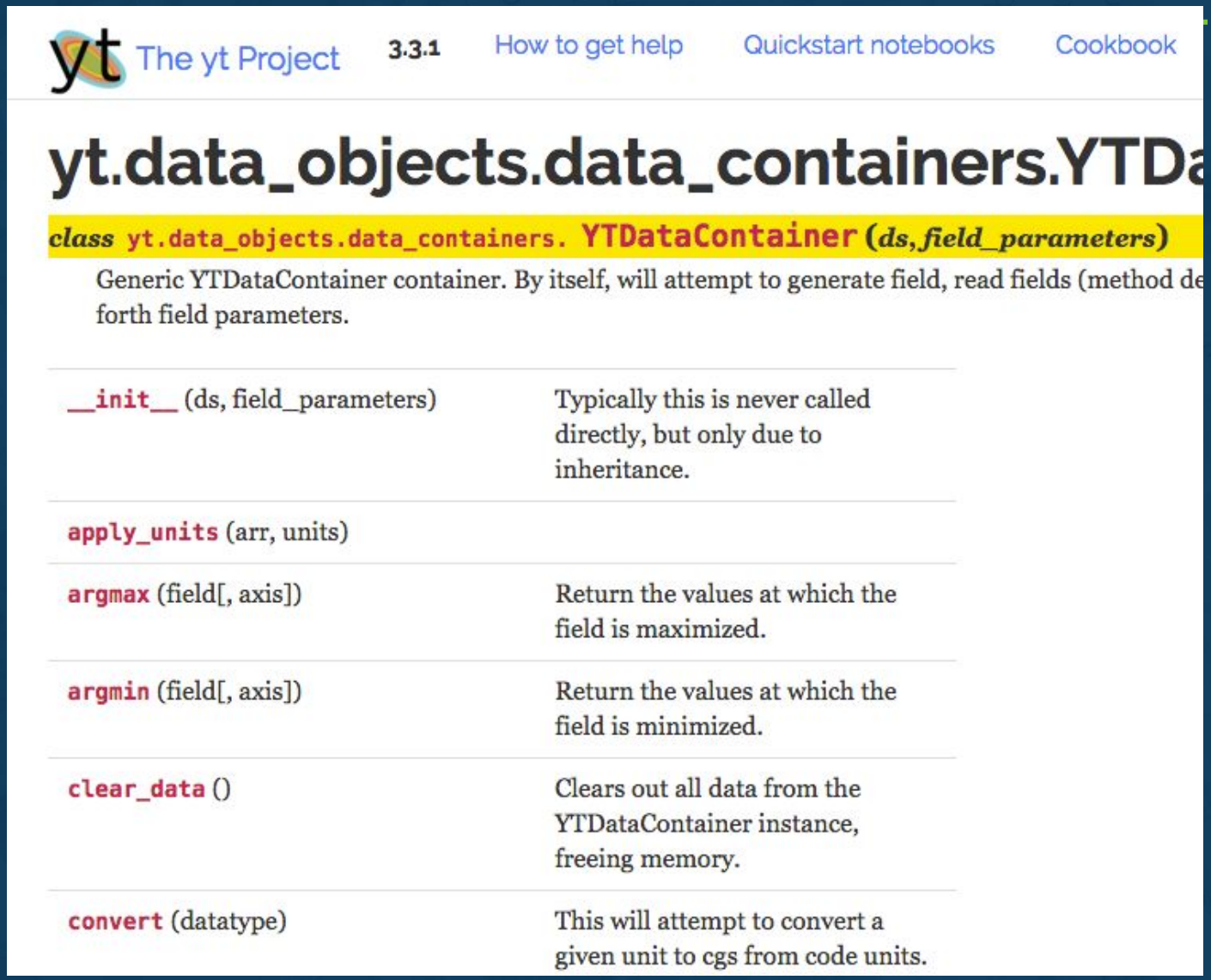


Collaboration diagram for vtkUndirectedGraphAlgorithm:



CTSC

Python → pydoc, Sphinx



yt The yt Project 3.3.1 How to get help Quickstart notebooks Cookbook

yt.data_objects.data_containers.YTDa

class yt.data_objects.data_containers. **YTDataContainer** (*ds, field_parameters*)

Generic YTDataContainer container. By itself, will attempt to generate field, read fields (method de
forth field parameters.

__init__ (ds, field_parameters)	Typically this is never called directly, but only due to inheritance.
apply_units (arr, units)	
argmax (field[, axis])	Return the values at which the field is maximized.
argmin (field[, axis])	Return the values at which the field is minimized.
clear_data ()	Clears out all data from the YTDataContainer instance, freeing memory.
convert (datatype)	This will attempt to convert a given unit to cgs from code units.

yt Project. Last updated on Jul 25, 2016. Created using Sphinx 1.3.4.

Further Reading

- https://docs.oracle.com/javase/8/docs/api/
- http://www.stack.nl/~dimitri/doxygen/index.html
- http://www.sphinx-doc.org/en/stable/index.html
- http://www.sphinx-doc.org/en/stable/ext/autodoc.html
- https://docs.python.org/3/library/pydoc.html

Best Practices: Summary

Best Practices

- Join/contribute to mailing lists related to secure software.
- Use a trustworthy software repository and hosting service.
- Use an issue tracking tool.
- Adopt a continuous integration (CI) process and tool.
- Incorporate static analysis into your CI process.
- Address at least the most severe issues from static analysis.
- Provide people-friendly documentation at multiple levels of the software lifecycle.

Best Practices

- Provide a digital signature/hash for your code.
- Validate the authenticity of code you download.
- Perform multiple levels of testing and, when possible, automate it.
- Use assertions in your code.
- Keep issue tracking, etc, private for vulnerability patches in progress.
- Put someone in charge of the vulnerability management process.
- Routinely test web apps using a trusted vulnerability scanner.

Further Reading

General:

http://csrc.nist.gov/publications/PubsTC.html

http://www.sei.cmu.edu/

https://buildsecurityin.us-cert.gov/

More specific:

http://csrc.nist.gov/publications/drafts/800-160/sp800_160_second-draft.pdf

https://samate.nist.gov/Other_Test_Collections.html

http://csrc.nist.gov/groups/SNS/acts/index.html

http://trustedci.org/trainingmaterials/

CTSC

CENTER FOR TRUSTWORTHY SCIENTIFIC CYBERINFRASTRUCTURE

The NSF Cybersecurity Center of Excellence

Thank You

Questions/Discussion

trustedci.org

@TrustedCI

We thank the National Science Foundation (grant 1547272) for supporting our work.