

Managing Software Integrity in an Agile World

Behrooz Zahiri

Director of Product Marketing

Coverity



- The Software Integrity Crisis
- What is Agile?
- How Static Analysis Increases Software Integrity
- About Coverity

The Software Integrity Crisis



Software vulnerability is everywhere!



Michael Krigsman
Get IT Project Failures via: [Mobile](#) [RSS](#) [Email Alerts](#) Blog: [Michael's Blog](#)
Pick a blog category

February 21st, 2008

System update kicks Heathrow baggage system offline

Posted by Michael Krigsman @ 7:32 pm

Categories: [Recent failures](#), [Government projects](#), [IT issues](#), [CIO issues](#)

Tags: [Business](#), [Information Technology](#), [System Update](#), [Tools & Techniques](#), [Software Upgrades](#), [Transaction](#), [Strategy](#), [Management](#), [Enterprise Software](#), [Software](#)

4 TalkBacks

Essential Topics

Green IT Center

ESG Report: Emerging Green IT and needs in IT

Watch The Webcast: Going Green

White Paper: Mastering carbon management. Balancing Trade Optimize Supply Chain Efficiency

Webcast: Turn Down The Power Turn Up The Profits

BusinessWeek

HOME INVESTING COMPANIES TECHNOLOGY INNOVATION MANAGING SMALL BIZ B-SCHOOLS ASIA

Current Issue Past Issues Cover Story Podcasts Figures of the Week Small Biz Magazine BW TV Subscribe FAQs

COVER STORY April 10, 2008, 5:00PM EST

text size:

The New E-spying Threat

A *BusinessWeek* probe of rising attacks on America's most sensitive computer networks uncovers startling security gaps



SFGate

home of the
San Francisco Chronicle
Home Delivery | Today's Paper | Ads

Find a buyer for your TV.
Sell stuff on Kaango.
FREE ads and photos reaching millions of SFGate visitors.

SEARCH SFGate Web Search by YAHOO! | Advanced Search

Home News Sports Business Entertainment Food Living Travel Columns Classifieds Jobs Homes Cars Index

Bay Area & State | Nation | World | Politics | Crime | Tech | Obituaries | Education | Environment | Science | Health | Weird

BART'S BLUNDER

35,000 evening commuters left in lurch -- system takes blame for software snafu

Simone Sebastian, Carl T. Hall, Cicero A. Estrella, Chronicle Staff Writers

Thursday, March 30, 2006

Embarrassed BART officials are taking the blame for a computer crash that stranded 35,000 commuters for more than an hour at the height of the Wednesday evening commute.

The computerized baggage handling system in Heathrow Airport 4 was kicked offline for about two days due to software problem, carriers in the terminal were forced to sort baggage manually. 6000 passengers experienced delays, cancellations, and hassles.

A terse press release from BAA, which operates Heathrow, stated

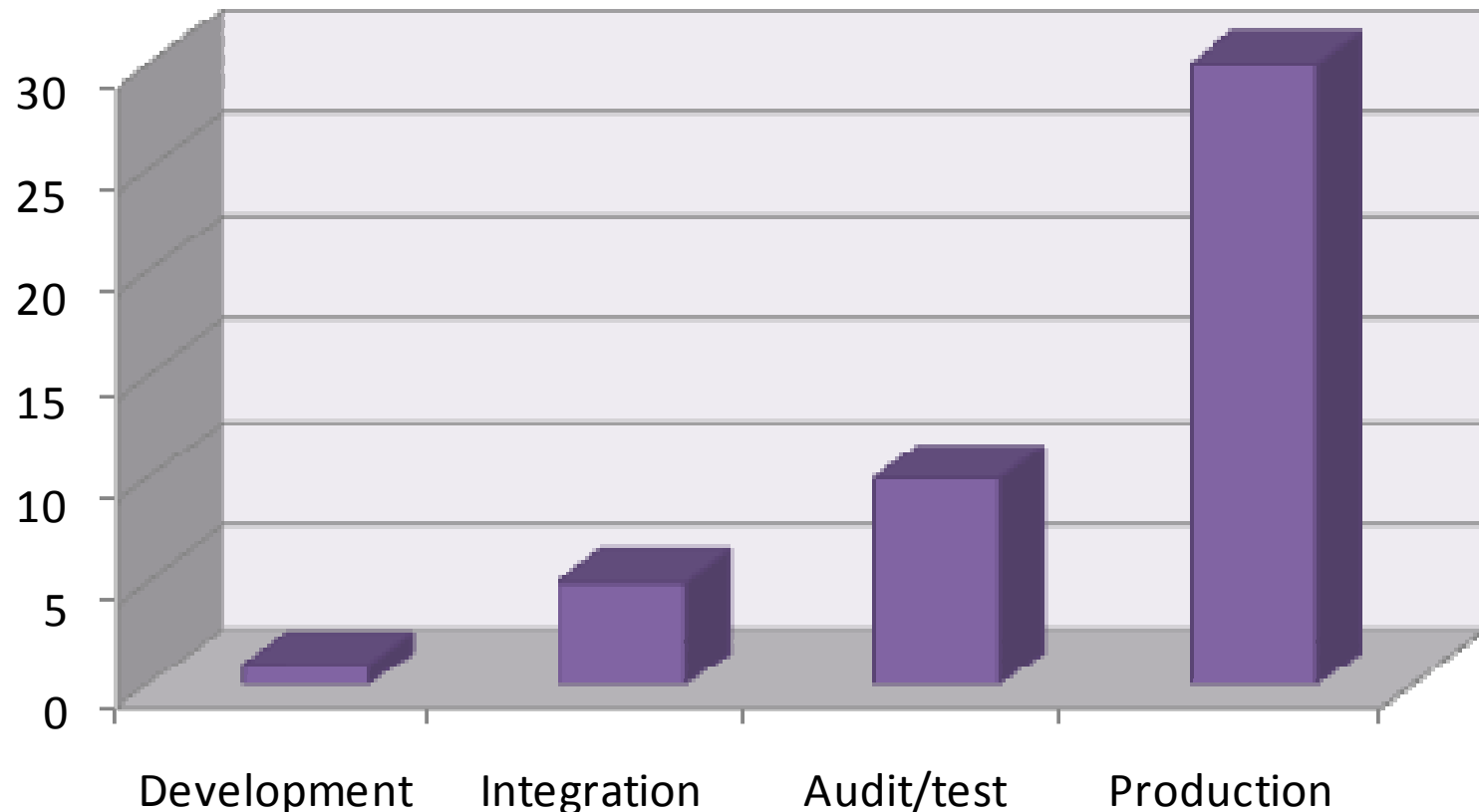
FREE ads and photos reaching millions of SFGate visitors.

MOST COMMENTED MOST READ MOST E-MAILED

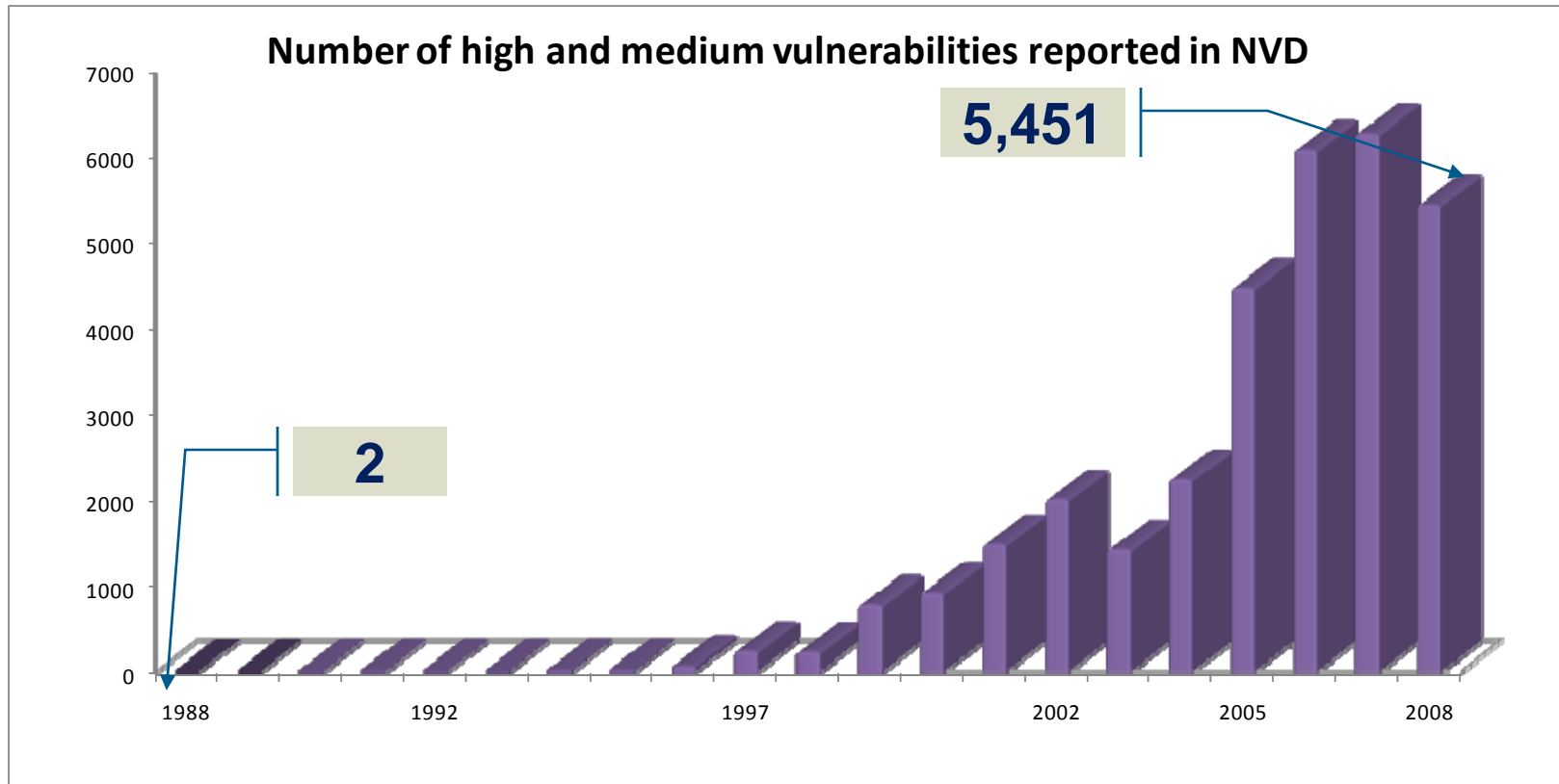
The earlier you find a defect...
...the cheaper it is to fix



Cost for defect fixes



Yet more defects are released in production each year



Source: NIST

Why software integrity is so important



Increased Complexity

- Codebases are growing
- Increased tension between development and QA; wasted cycles trying to reproduce bugs

Increased Mission Criticality

- Cost of failure is increasing, increasing the visibility and scrutiny of testing and defect management
- High-profile firefights on the rise

Increased Time to Market Pressure

- Development teams are under pressure to release higher quality software even faster
- Traditional development methodologies can't keep pace

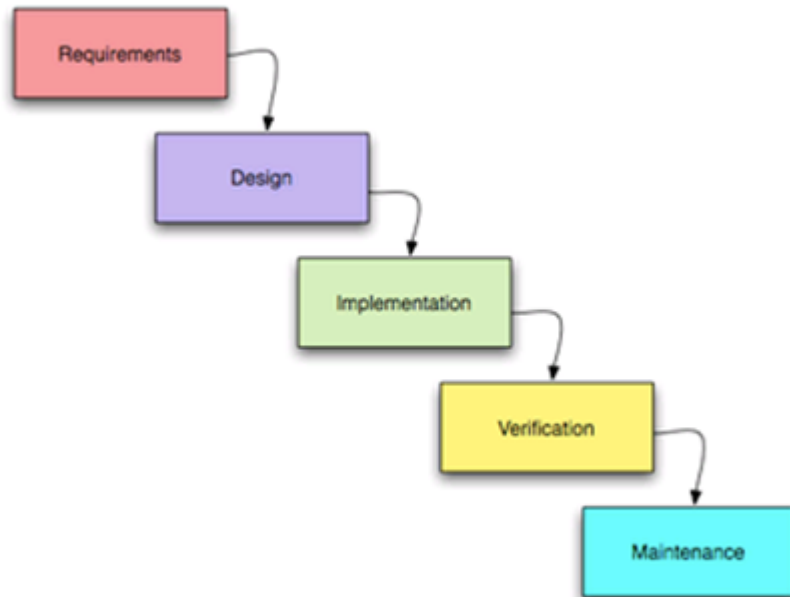
What is the biggest challenge to maintaining software integrity in your environment?

- A.** Code bases are increasing in size/complexity.
- B.** High employee turnover increases the risk of coding errors.
- C.** Manual processes make it hard to spot defects in development.

What is Agile?

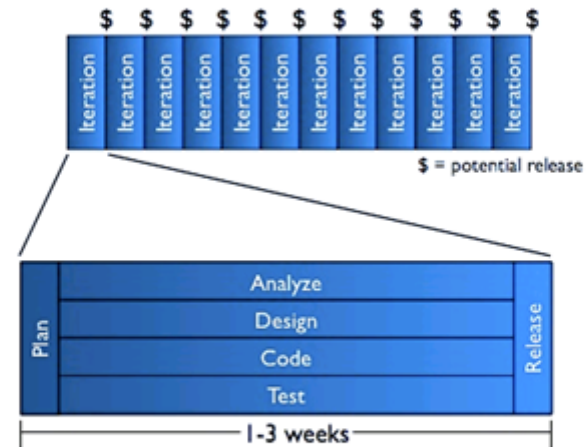
How is Agile development different?

Waterfall

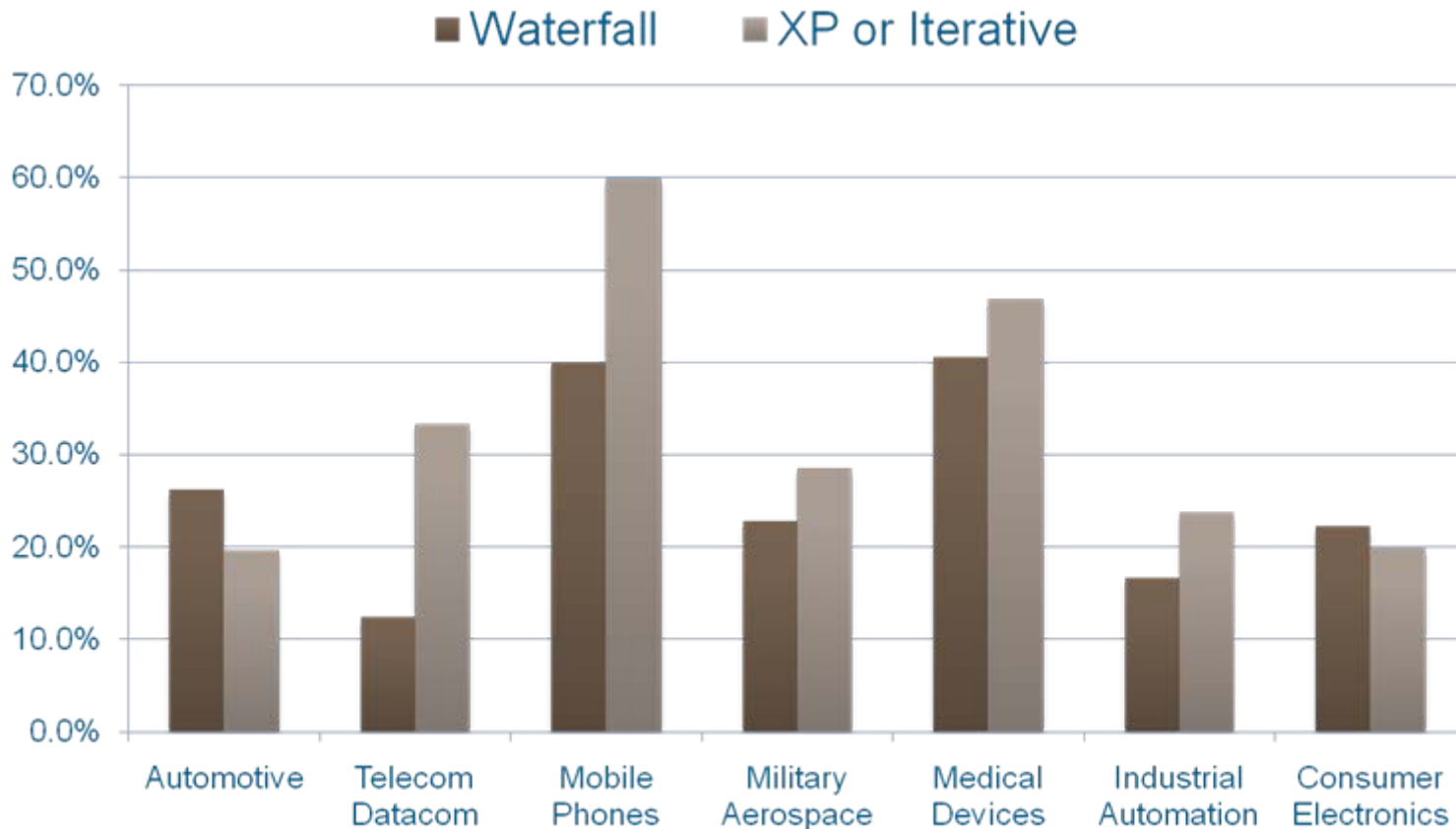


Agile (Scrum, XP, Lean, etc.)

The XP Lifecycle



Agile is becoming more popular



Agile requires software integrity to become everyone's responsibility!



Benefits of Agile

- Increased visibility
- Increased flexibility
- Increased quality

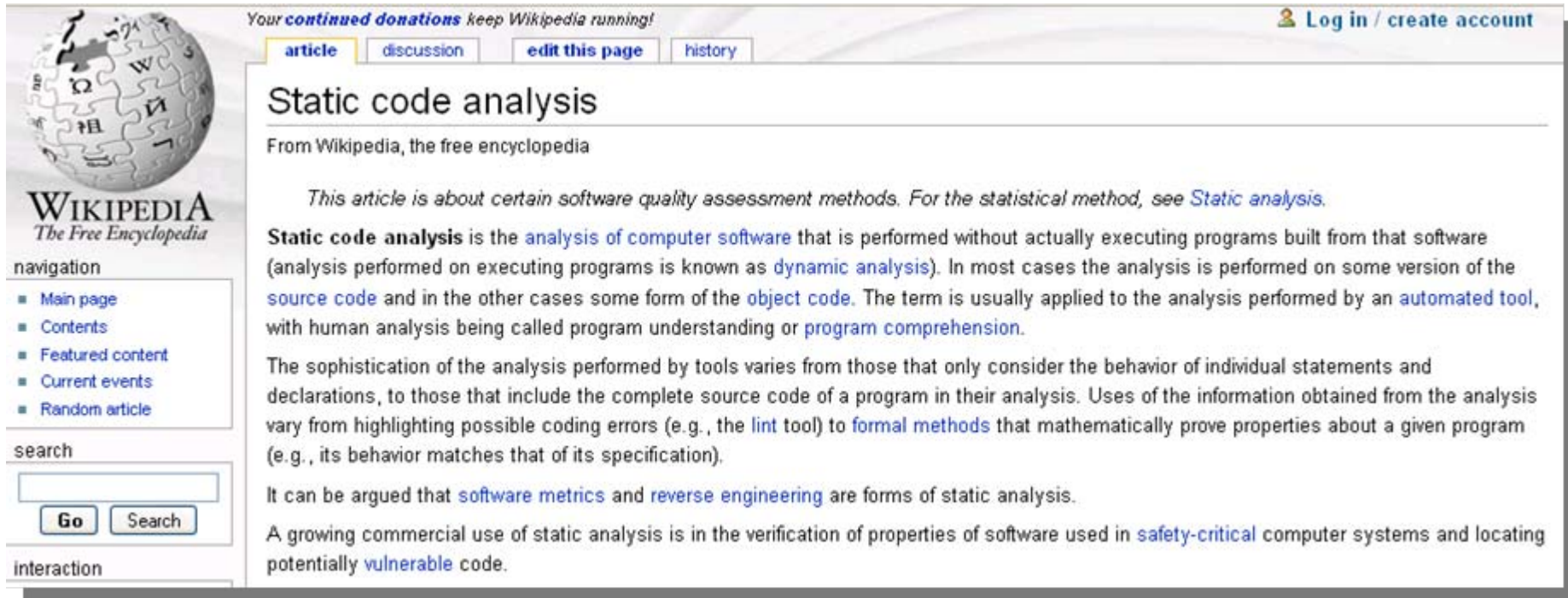
Risks to Software Integrity

- Frequent “potentially shippable” iterations, each requiring a QA cycle
- Shortened QA cycles – QA can't keep up!
- Increased pressure on full development team to deliver high-integrity code

How Static Analysis Increases Software Integrity



What is static analysis?



The screenshot shows the Wikipedia article for "Static code analysis". At the top, there is a navigation bar with tabs for "article", "discussion", "edit this page", and "history". The article title "Static code analysis" is prominently displayed. Below the title, it says "From Wikipedia, the free encyclopedia". A summary paragraph follows, stating that the article is about software quality assessment methods. The main body of the article defines static code analysis as the analysis of computer software performed without executing programs. It mentions that analysis can be performed on source code or object code, often using automated tools. The text also notes that the sophistication of analysis tools varies, from simple error highlighting to formal methods that prove properties about a program. Finally, it mentions the growing commercial use of static analysis in safety-critical systems to find vulnerable code.

Your *continued donations* keep Wikipedia running!

[article](#) [discussion](#) [edit this page](#) [history](#)

Static code analysis

From Wikipedia, the free encyclopedia

This article is about certain software quality assessment methods. For the statistical method, see [Static analysis](#).

Static code analysis is the [analysis of computer software](#) that is performed without actually executing programs built from that software (analysis performed on executing programs is known as [dynamic analysis](#)). In most cases the analysis is performed on some version of the [source code](#) and in the other cases some form of the [object code](#). The term is usually applied to the analysis performed by an [automated tool](#), with human analysis being called program understanding or [program comprehension](#).

The sophistication of the analysis performed by tools varies from those that only consider the behavior of individual statements and declarations, to those that include the complete source code of a program in their analysis. Uses of the information obtained from the analysis vary from highlighting possible coding errors (e.g., the [lint](#) tool) to [formal methods](#) that mathematically prove properties about a given program (e.g., its behavior matches that of its specification).

It can be argued that [software metrics](#) and [reverse engineering](#) are forms of static analysis.

A growing commercial use of static analysis is in the verification of properties of software used in [safety-critical](#) computer systems and locating potentially [vulnerable](#) code.

navigation

- [Main page](#)
- [Contents](#)
- [Featured content](#)
- [Current events](#)
- [Random article](#)

search

interaction

Think of it as a spell checker that finds your most difficult bugs!

- Nullable references were invented by [Hoare](#) in 1965 as part of the [Algol W](#) language.
- Hoare later (2009) described his invention as a 'billion dollar mistake':[\[4\]\[5\]](#)

“ I call it my billion-dollar mistake. At that time, I was designing the first comprehensive type system for references in an object oriented language ([ALGOL W](#)). My goal was to ensure that all use of references should be absolutely safe, with checking performed automatically by the compiler. But I couldn't resist the temptation to put in a null reference, simply because it was so easy to implement. This has led to innumerable errors, vulnerabilities, and system crashes, which have probably caused a billion dollars of pain and damage in the last forty years”

- Since a null-valued pointer does not refer to a meaningful object, an attempt to dereference a null pointer usually causes a run-time error.

Static analysis strengthens your existing testing approaches!



Traditional Testing	Static Analysis
<ul style="list-style-type: none">▪ Unit Tests▪ Integration Tests▪ Performance Tests▪ Code Reviews	<ul style="list-style-type: none">▪ Locking semantics▪ Class usage semantics▪ Performance exposures▪ Security exposures▪ and more!

Will a code review catch this?



```
#include <pthread.h>
pthread_mutex_t mtx;
int i;
void foo() {
    i++;
    if (i == 1) {
        pthread_mutex_lock(&mtx);
        i++;
        pthread_mutex_unlock(&mtx);
    }
}
```

i not initialized and i++ not locked

What kind of defects might be in *your* software?

- SQL Injection
- Improper Locking/Concurrency
- NULL Pointer Dereference
- Resource Leak
- Unintentional Ignored Expressions
- Use Before Test (NULL)
- Buffer Overrun (statically allocated)
- Unsafe use of Returned NULL

- Use After Free
- Uninitialized Values Read
- Unsafe use of Returned Negative
- Type and Allocation Size Mismatch
- Buffer Overrun (dynamically allocated)
- Use Before Test (negative)
- And more!



Benefits of static analysis in Agile environments

- **Finds defects sooner** - as soon as they're created, when they're cheapest to fix
- **Empowers development teams** to remain nimble and rapidly deliver quality software
- **Strengthens** existing testing methodologies with an objective lens



What software methodology most represents your codebase as it relates to Agile development?

- A.** We are standardized on Agile development.
- B.** We are looking to adopt Agile or have recently started utilizing Agile.
- C.** We use a hybrid of Agile and Waterfall methodologies.

About Coverity



Companies that have zero tolerance for software failures choose Coverity.

High Integrity Software



Software that
doesn't fail.



Software that
performs.



Software that
is secure.

600 Customers Worldwide

Over 600 customers. Billions of lines of code.



Five Steps To Mitigate Risk

```
long int SomeFunction();  
/* int OtherFunction();  
/* int */ CallingFunction(),  
{  
    long int test1;  
    register /* int */ test2;  
  
    test1 = SomeFunction();  
    if (test1 > 0)  
        test2 = 0;  
    else  
        test2 = OtherFunction();  
  
    return test2;  
}
```



Scan
your
software

- List of Defects
- _10001 critical
 - _10002 major
 - _10003 major
 - _10004 critical
 - _10005 major

Find
priority
defects

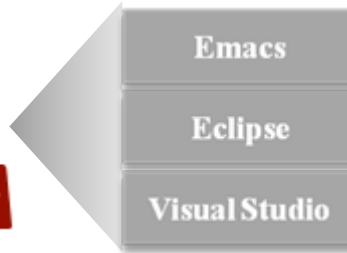
Browse code

```
long int SomeFunction();  
/* int OtherFunction(); */  
/* int */ CallingFunction()  
  
long int test1;  
register /* int */ test2;  
  
meFunction();  
> 0)  
2 = 0;
```

Impact Rankings



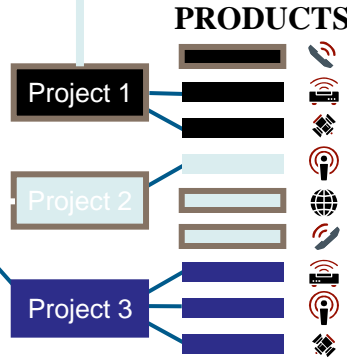
Fix
priority
defects



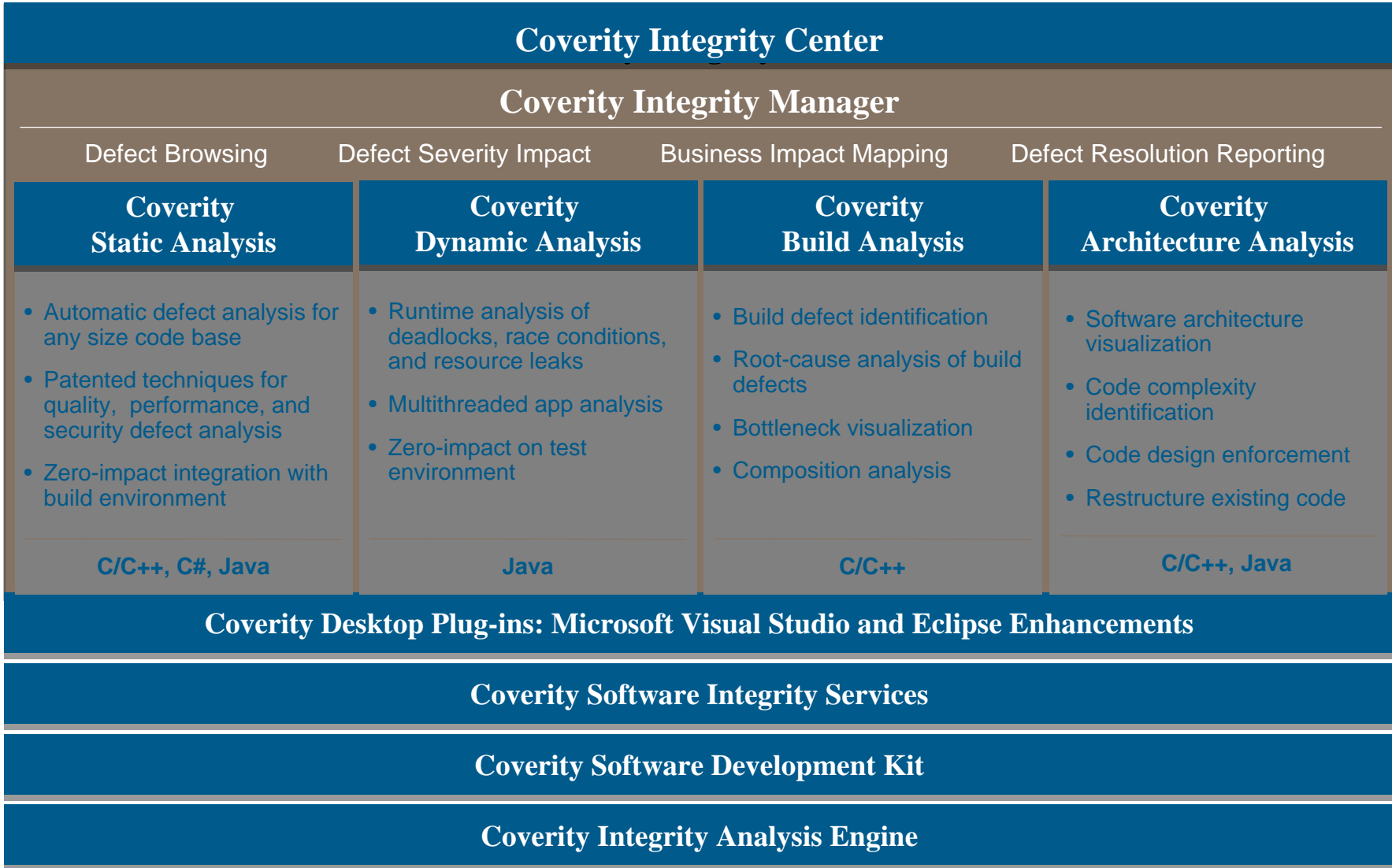
Report
defect
remediation

Map
business
impact

Code
base



PRODUCTS



Request a free trial today!



Please visit us at www.coverity.com
to request a free trial

The screenshot shows the Coverity website's 'Request a Free Trial' page for the Integrity Center. At the top, there is a navigation bar with the Coverity logo, a search bar, and a 'Request A Free Trial' button. Below the navigation bar, there is a circular graphic with the Coverity logo in the center and the text 'Integrity Center' below it. The circular graphic is surrounded by four segments: 'Architectural Analysis', 'Dynamic Analysis', 'Build Analysis', and 'Static Code Analysis'. To the right of the graphic, the text reads 'Coverity Integrity Center' followed by 'Improve Your Source Code Quality and Security'. Below this, there are three bullet points: 'Enhance Application Quality and Security', 'Lower Development Costs', and 'Improve Development Team Efficiency'. At the bottom, there is a 'Request a Free Trial' section with a paragraph of text and two input fields for 'First Name*' and 'Last Name*'. The text in the 'Request a Free Trial' section reads: 'Over 600 companies and 100,000 developers rely on Coverity products to identify, triage, and remediate critical software errors. See first hand how Coverity Software Architecture, Code, and Build Analysis can help improve the quality and security of your products.'

coverity 1-415-321-5200 or contact sales

Solutions Products Services Customers News and Events Company Research Library Request A Free Trial

coverity Integrity Center

Architectural Analysis
Dynamic Analysis
Build Analysis
Static Code Analysis

Coverity Integrity Center

Improve Your Source Code Quality and Security

- Enhance Application Quality and Security
- Lower Development Costs
- Improve Development Team Efficiency

Request a Free Trial

Over 600 companies and 100,000 developers rely on Coverity products to identify, triage, and remediate critical software errors. See first hand how Coverity Software Architecture, Code, and Build Analysis can help improve the quality and security of your products.

*Required

First Name*

Last Name*

Thank you!

