Pattern Insight™ Code Assurance

Stop Releasing Known Bugs

pattern insight™

Do you find your customers complaining that a previously fixed bug occurred again?

With Code Assurance, you can automate finding every instance of a bug and never release it again.

It's Critical to Find Every Instance of a Bug and Never Release it Again

Modern development processes inevitably result in a high degree of code replication caused by the increasing number of reusable components, branches, versions and parallel product lines with similar code, or due to copy-paste programming. You can fix a critical bug in one place, but miss other instances of it because you're simply not aware of other locations where the bug exists—especially if the code containing the bug was often reused and then modified.

Continuous integration development models, such as Agile, further amplify this problem as bugs spread quickly and broadly through the entire code base. Today's rapid release cycles make it impossible to manually check every module, version and branch. As a result, known bugs are often released to customers, leading to decreased customer satisfaction and retention. Finding and fixing bugs over and over again comes with a high price tag, dramatically reduces productivity, and can, at times, significantly increase the cycle time. Therefore it's critical to find every instance of a bug and never release it again.

A leading wireless telecom vendor relies on Code Assurance to eliminate all Priority 1 bugs from its entire code base. It does so by enforcing checks before every build and blocking a build if known bugs are present.

Why Current Solutions Don't Work

Most development organizations have policies around the propagation of bug fixes into other active branches and versions. But to implement those policies, they usually rely on either manual or in-house tools to track down other instances of the bug. To find every instance of a bug, these approaches aren't reliable. And here's why:

Tools that rely on simple search typically use keyword search or regular expression, which have many drawbacks. First, it's often hard to figure out the right query that gets the results you want. Second, the results are usually too noisy, since the search lacks context. Finally, you can't be certain that the results are complete.

For example, it's very likely that you'll miss cases where someone has copy-pasted the buggy code into another area, and then modified it—by changing a variable name that you've specified in the query keyword or expression, or by deleting or modifying a statement which contains the keyword or matches the regular expression.

So if you want to find all instances of a bug, current solutions are not the answer.

The Answer is Code Assurance—Based on Fast Fuzzy Search

In order to find every instance of a bug, across all versions and branches, you need Code Assurance, a solution based on Pattern Insight's patent-pending fast fuzzy snippet search capability.

pattern insight™

The fuzzy snippet search can:

- Enable searching for snippets of arbitrary size
- Tolerate differences at a lexical level, such as variable rename
- Tolerate differences at the syntactic level, such as statement additions, deletions, and modifications

Often a bug fix involves multiple snippet changes in a number of files. Code Assurance can intelligently determine the correct context and the right level of fuzziness to be considered for each snippet and consolidate the results from multiple snippet results to draw an accurate conclusion. It has the following characteristics:

- **Fast:** Always returns results in seconds
- **Accurate:** Extremely low false positives
- **Easy-to-use:** Fully integrates with all SCM systems and is capable of being used in any development, build or release process

With its out-of-the-box integration with most SCM systems and open APIs, Code Assurance can easily be built into the build process, not only to find every instance of a bug automatically, but also to ensure it is never released again.

How Code Assurance Works

Here are two typical use scenarios for Code Assurance:

Interactive Use. An engineer just fixed a bug in one place, and you want to use Code Assurance interactively to find out if it has other instances.

Batch Processing. A defect database containing all registered bugs is checked automatically for every daily build or every product release.

Let's start by discussing how to use Code Assurance interactively and then how it can be extended to check bugs automatically.

First, you'd fix a bug and commit the fix into your SCM system. Next, you create and register the patch through either the Code Assurance Web UI or command line interface. Code Assurance allows you to select one commit (or a list of commits if a fix involves multiple changes), automatically pull the change and its related files out of your SCM system, and then register it as a patch with Code Assurance's defect database. In addition, you have the option to specify attributes like a description of the patch, the bug IDs associated with the patch, bug priorities, and links to the bug tracking system.

Once the patch is registered, you can search it in a specific branch, across multiple branches or multiple versions, and across product lines—in a matter of seconds. Figure 1 shows an example of such a final result report. There are four kinds of possible results:

1. **Patched.** The patch has been fully applied in the expected location
2. **Partially Patched.** The patch has been partially applied in the expected location
3. **Unpatched.** The patch has not been applied in the expected location
4. **N/A.** The patch is not applicable in the expected location

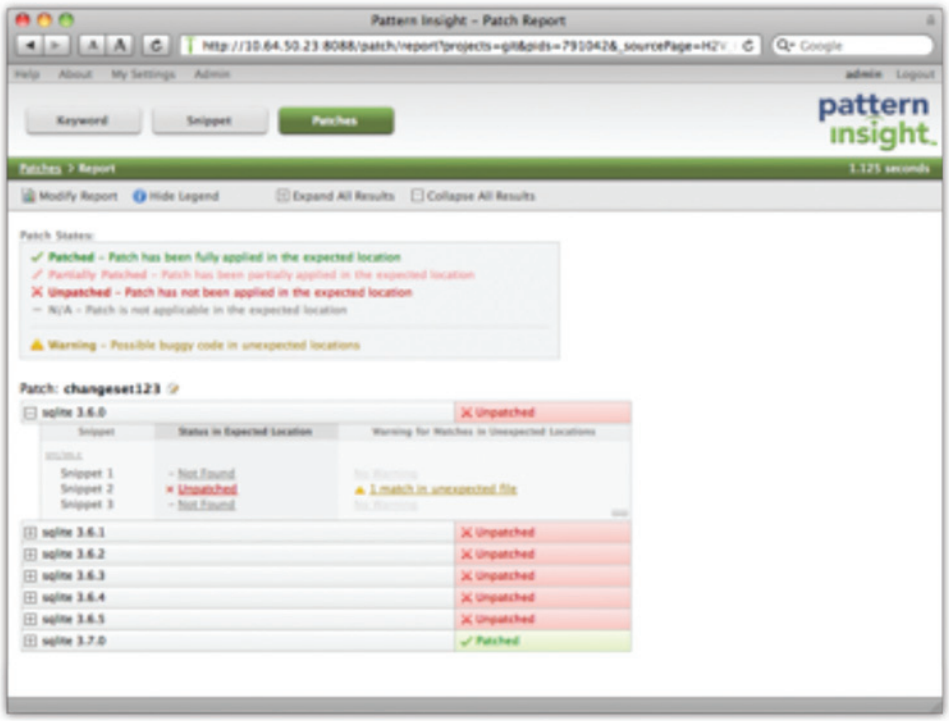

Figure 1: Code Assurance Report

In addition, if there are instances of the bug in other locations, Code Assurance flags a warning and shows you the location.

Once you locate all the matches of interest, you can be certain that you have found all the locations that match the original defect — regardless of modifications such as the code being moved to a different file, or copy-pasted into a new method.

You can further integrate Code Assurance into your build and release process with the use of its APIs. A defect database containing all your high impact bugs can be automatically checked against your daily build or product release and notify you when an occurrence of a known bug is found. With Code Assurance you can be confident that you will never release a know bug again.

Code Assurance Key Features

- **Easy-to-Use.** Install, configure, and get immediate value in hours
- **Fast.** Search takes seconds, and preprocessing takes ~1 min / MLOC initial, incremental update after
- **Accurate.** Extremely low false positives
- **Functionality Rich.** Equipped with numerous search capabilities, including advanced keyword search, fuzzy snippet search and intelligent patch search
- **Scalable.** Distributed architecture allows Code Assurance to scale to billions of lines of code
- **Seamless Integration with SCMs.** Out-of-the-box support for ClearCase, Perforce, SVN, Git, CVS, and Mercurial; scripting support for other SCM systems such as Borland StarTeam, Microsoft Team Foundation Server, and AccuRev
- **Highly Customizable.** Web services allow custom reporting and integration with any bug tracking system and build process
- **Open.** Supports C, C++ and Java out-of-the-box; extensible to support any other language

Server System Requirements	
Architecture	X86 and X86-64
Operating Systems	Linux (RHEL 4, RHEL 5, Ubuntu 7.10 and above)
Memory	4G minimum, 8G recommended
Disk Space	200 MB disk space for installation and 2-3X code base size for operational use and temporary storage
User Interface	Web browser (Chrome 8.0, IE 7.0, Firefox 3.0, Safari 3.0 and above)

pattern insight™

About Pattern Insight

Pattern Insight improves its customers' software quality and security. Its products are the first in the industry to make it significantly faster, easier and more accurate to find and address defects and security issues in source code. Designed for engineering teams, Pattern Insight's Code Assurance is a sophisticated software solution that identifies and removes all "known" (previously fixed) defects in code before it is released.

Pattern Insight was founded by a team of University of Illinois researchers with cutting-edge expertise in systems mining. Key investors include Venture Investors, John Lovitt and Dr. Kai Ki. Pattern Insight is headquartered in Mountain View, CA.

PatternInsight.com

Pattern Insight Headquarters
6540 Lusk Blvd Suite C200, San Diego, CA 92121 P: 866 582 2655 E: info@patterninsight.com