

The SIERRA Platform: Know Your Code

SIERRA is a bug management platform that helps developers deal with the often overwhelming number of findings produced by popular source code analysis tools when analyzing large codebases. For example, a typical 1MLOC app may yield close to 50,000 individual findings. Developers know that some are critical while others may be insignificant or spurious. SIERRA is designed to help you quickly zoom-in to the findings that matter to you, to handle them rapidly, to see what's new and different from the last scan, and to coordinate activity across a team, in harmony with existing ALM tools and software development processes.

It comes bundled with a suite of popular open-source analysis tools, along with a proprietary scanner and the capability to integrate additional analysis tools. It installs as a single IDE plug-in, providing a unified, multi-tool perspective on the code. It has a powerful GUI to enhance developer productivity. Additionally, it can be directly incorporated into an automated build process, with results easily browsed and acted on by team members. Finally, it generates reports and metrics to help team leaders gain visibility into their development process and better coordinate the handling of analysis findings.

SIERRA consists of an IDE plug-in and a Team Server. The plug-in is compatible with Eclipse 3.2+ and IBM RAD 7, with other IDEs on the way. The Team Server can be installed in three ways: (1) on a developer's local machine for smaller projects or for trial purposes, (2) on an on-site team server, or (3) hosted by SureLogic in a SaaS model.

IDE Plug-In Highlights

- **Bundled with multiple scanners.** Combines multiple analysis engines (including FindBugs™ and PMD) to scan code for bugs.
- **Quick search.** Makes it easy to filter and prioritize findings.
- **Configuration.** Lets you configure input rules for the analysis engines.
- **Queries/Diffs.** Persists analysis results, audit trails, and comments in an embedded database for fast and easy queries and “diffs.”
- **Easy start-up.** Installs in less than 10 minutes.

Team Server Highlights

- **Collaboration.** Team members can share audit trails, comments, scans, and findings.
- **Snapshots.** Reports on current code quality and contributions made by team members.
- **Time-series.** Team members can assess progress over time and set team goals to achieve.
- **Configuration.** Enables teams to centrally configure multiple IDE clients.

Words from our earliest early-adopters

“The tool’s UI is very responsive, and it’s easy to go through a lot of findings very quickly.”

– Vamsi S., Java Architect
healthcare industry

“I love it! It already caught one major problem and dozens of minors ones. I try to run it 2 or 3 times a week. It’s very easy to use. [...] I’m really excited about rolling out Sierra to our team.”

– Bob P., Java Team Leader
Web 2.0 / podcasting company

“I think I could get to like this tool VERY much.”

– John B., Senior Java Developer
speech recognition company

How can source code analysis help you?

Rule-based analysis tools quickly scan your code to find common faults that may affect performance, security, readability, and maintainability of your code.

Scans can be done within your IDE in the background while you code, or as part of your automated or nightly build to find bugs before they bite.

Analysis tools can help assess the quality of code that you receive from other teams and 3rd parties – open-source, outsourced, etc.

With explanations of each bug found and suggestions for how to fix, our analysis tools can also help you learn tips and tricks to produce cleaner, faster, and more understandable code.

Beyond “bug hunting,” SIERRA provides tracking, reporting, and collaboration functionality, so you can work effectively with your team, your management, your suppliers, and your clients.

For more information visit <http://www.surelogic.com/sierra>

JSure for Concurrency

Java on multi-core – speed with safety

JSure is a model-based static analysis tool that helps developers answer this question—

“Are my threads accessing shared state in a safe way?”

JSure provides positive assurance (sound analysis, not rule-based) that correct locks are held when shared state is accessed. This enables programmers to develop confidence that their code is “thread safe” – it satisfies state consistency requirements. Because it is composable, the JSure analysis is scalable to the largest Java systems, and has been proven on large existing code bases. It is supported by a lightweight developer-friendly annotation language that is compatible with JSR 305. JSure is an IDE plug-in, living with developers to provide rapid feedback concerning locking policies as code is written. JSure acts as a safety net as developers work to aggressively parallelize their code for multi-core processors, removing the fear of introducing hard-to-debug race conditions.

Flashlight Dynamic Analysis

Discover potential race-conditions and deadlocks

When developers are in the dark about why their application is experiencing intermittent failures, poor performance, or data corruption, Flashlight is a “special-purpose” runtime profiler that can provide visibility into concurrency issues within the app and illuminate the path to root causes. Flashlight monitors thread behavior, lock usage, and lock contention as threads access shared state over time. Flashlight then allows rich queries into the resulting multi-dimensional data set to determine where potential race-conditions, deadlocks, and performance bottlenecks exist in the code. This helps developers of threaded code better understand the realities of the runtime behavior of their code.

What our early users have said

“JSure can reduce the time required to detect and diagnose concurrency errors by several orders of magnitude, helping to quickly deliver a highly reliable and dependable software product.”

– Lockheed Martin ATL, TechBriefs article, December 2007.

http://www.atl.lmco.com/news/techbriefs/techbriefs_Fluid.php

“I can’t think of any of our Java code I wouldn’t want to run this tool [JSure] on.”

– Architecture Lead at a aerospace/defense company

“The tool can be used to increase concurrency which should lead to performance and scalability benefits.”

– Java developer at a major database vendor

The Multi-Core Challenge

It is reported that, as of February 2008, fewer than 10% of Intel processors shipped are single-core. Over the next few years, billions of multi-core processors will be embedded in mobile phones, TVs, PCs, and other appliances worldwide.

Writing concurrent applications that fully exploit the power of multi-core processors is hard to do correctly, but applications *must* be multi-threaded in order to keep up with rapidly increasing demand for greater performance.

“For the past 30 years, computer performance has been driven by Moore’s Law; from now on, it will be driven by Amdahl’s Law”

– Doron Rajwan, Intel

JSure and Flashlight were created to help Java developers deal with the multi-core challenge.

“Writing thread-safe code is, at its core, about managing access to state, and in particular, to shared, mutable state.”

– Java Concurrency in Practice, Brian Goetz et al.

“Sweeping changes (impacting thousands of locks in this case) would be risky and are not likely to be attempted without the assurance that the tool provides.”

– Senior Java developer at a major software/hardware vendor

For more information visit <http://www.surelogic.com/concurrency>