CodePro AnalytiX:

A Powerful Approach to Developing High Quality Code Using Eclipse

CodePro AnalytiX:

- Detects and corrects code quality issues...Automatically
- Defines, distributes and enforces code quality
- Distributes standards across development teams
- Contains 800+ audit rules and metrics / 350+ Quick Fixes
- Provides code metrics with drilldown and triggers
- Allows custom rules and rule sets
- Provides JUnit test case generation
- Generates code coverage analysis
- Provides dependency analysis and reporting
- Allows integrated team collaboration
- Provides Javadoc analysis and repair
- Integrates seamlessly into Eclipse

"We are impressed with the capabilities of CodePro AnalytiX. The dynamic audit facility in CodePro AnalytiX allows us to enforce programming standards and best practices at development time, instead of having to rely on extensive and lengthy code reviews. This results in cost savings and ensures that we deliver our software on time."

Rich Main, director, Java development environments at SAS

The Challenge

Software development organizations face the daunting task of creating software that meets best practice standards of code quality while still delivering the software product on time and within budget. Too often, valuable developer and engineering time is spent in code reviews and trying to find errors in the software code which often leads to both project and cost over-runs—while still producing software with errors. According to the Washington D.C. National Institute of Standards and Testing (NIST), “Software errors cost the U.S. economy $60 billion per year. There are many reasons that projects fail: poor management, unrealistic expectations, complex customization, market changes, etc. But more often than not, poor-quality software is to blame. Blame is directed at the software provider—whether it is a commercial software vendor, an outside tech vendor or a company's IT department—because they are the ones responsible for code quality.”

The Solution

This paper describes how Instantiations CodePro AnalytiX software can be used to solve the code quality, code review and code dependency issues typically faced in software development. Instantiations pioneered the ground-breaking principal of ***Continuous Collaborative Code Analysis (C^{3}A).*** C^{3}A is designed to aid developers in finding and fixing problems in their code earlier in the development process. CodePro AnalytiX embodies the principals of C^{3}A in an innovative Java-based tool suite that is a perfect complement to the Eclipse Integrated Development Environment (IDE).

Historically, early error detection has required software developers to change their processes, think harder, use better methodologies, create better architectures, and the like. The challenge of changing human behavior, the complexity induced by far-flung development teams, downsizing, outsourcing, open source, feature-rich software systems and complex development environments all increase the difficulty of building high-quality software systems on time and within budget.

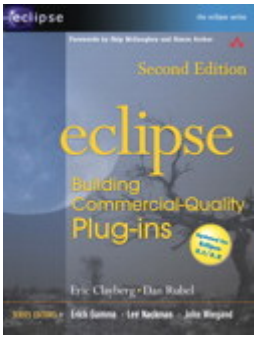
Pushing the resolution of software quality issues into a Quality Assurance (QA) or Testing organization, or worst case, onto customers, is at best expensive and can certainly be damaging to the software producer’s reputation. It’s common knowledge that if problems can be found and fixed early in the development process, it will result in better software, lower costs, faster time-to-market, and most importantly, happier users. With C^{3}A and CodePro AnalytiX, software development technology has advanced to the point where maximum quality can be automatically and efficiency built into software as it is being developed.

CodePro AnalytiX Integrated into Eclipse:

- CodePro AnalytiX functionality is seamlessly integrated in Eclipse
- CodePro AnalytiX menus are available from the Eclipse toolbar and various menus and submenus
- Concurrently supports all major versions of Eclipse including 2.0, 2.1, 3.0, 3.1 and 3.2

Instantiations Team—Eclipse Experts:

- CodePro AnalytiX was created by the development tool experts at Instantiations
- Development team with six years of Eclipse development experience
- CodePro AnalytiX developers are committers to the Eclipse open-source effort
- Development by authors of the popular book "*Eclipse: Building Commercial Quality Plugins*"



CodePro AnalytiX Improves Code Quality by Empowering Developers

CodePro AnalytiX was created by the development tool experts at Instantiations and incorporates the knowledge and expertise of the Instantiations team in an easy-to-use tool. CodePro AnalytiX, which is tightly integrated into Eclipse, ensures superior code quality and maximum developer productivity by adding powerful capabilities like code audit, metrics, testing, and team collaboration—while delivering continuous quality improvement throughout the entire code development cycle.

Using CodePro AnalytiX, less experienced developers can code like experts, while experts can quickly create tight, maintainable systems. CodePro AnalytiX lets companies define, distribute, and enforce corporate coding standards and quality measures across development teams—no matter where they are located.

Early identification and correction of code quality issues dramatically improves the application development process and reduces the time and money it takes to develop high performance Java software systems. Developers can use CodePro AnalytiX to easily find and fix code problems, conform to programming best practices, distribute standards and create high quality code. Companies can use CodePro AnalytiX to save time, money, and meet the increasingly stringent corporate quality standards.

CodePro AnalytiX Functionality Built for Seamless Integration into Eclipse

According to the Eclipse Technical Overview published by IBM and the Eclipse Foundation, "Eclipse is an open source community whose projects are focused on providing an extensible development platform and application frameworks for building software. Eclipse provides extensible tools and frameworks that span the software development lifecycle, including support for modeling, language development environments for Java, C/C++ and others, testing and performance, business intelligence, rich client applications and embedded development."

CodePro AnalytiX, designed to help developers create quality Java code, is the pre-eminent example of a value enhancing product that is designed for seamless integration into Eclipse. Few other products integrate as seamlessly into Eclipse as CodePro AnalytiX. For example, the Eclipse menu bar includes a menu for CodePro and the functionality built into CodePro AnalytiX is also available from the appropriate context menus within Eclipse, as shown in the following figure.

Benefits of Code Audit:

- Higher quality software is produced
- Adherence to coding conventions is proven to reduce the probability of errors
- Non-compliance can be detected in real-time so that corrections can be made immediately
- Dynamic code audit executes in real-time as the developer is coding
- Developers spend less time in code reviews and more time coding

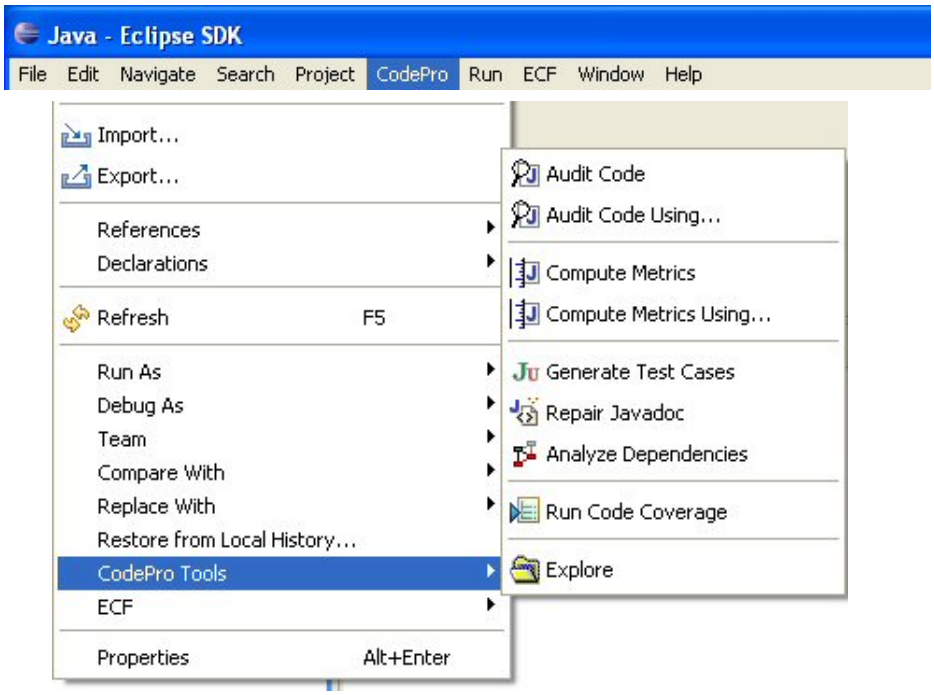


Figure 1: CodePro AnalytiX tools available in Eclipse

Code Audit

The code audit facility in CodePro AnalytiX is at the core of the product. There are over 770 Java-based coding rules in more than 30 categories built into the tool. Audit runs against these standards and determines areas where the code doesn't comply or has problems. Places in the code where issues are found are flagged in the editor's gutter and developers can select individual violations to see more detail on the code problem. Developers and managers can print a variety of reports to review what the tool finds. ***Using CodePro AnalytiX will improve the code quality because it finds problems while the code is being written.*** Companies can save both time and money because there will be less time spent searching for code errors and in time-consuming code reviews.

Dynamic Auditing

The dynamic code audit facility is invaluable in catching problems as the developer is writing code, rather than after the code is written. For example, a single naming violation might be propagated through dozens of source files before the nightly build's audit would flag the problem. Dynamic audit flags the problem as it occurs as close to the actual event as possible. Using the dynamic audit facility is like having a personal, private mentor or code reviewer sitting on your shoulder. In addition, dynamic auditing can save companies in coding costs and decrease project cost and time overruns since the cost to fix many coding problems is often in direct proportion to the time between when the problem was created and when it was discovered and resolved.

Code Audit Categories:

- **Coding Style**
- **Comments**
- **Dead Code**
- **Eclipse Plugin Development**
- **Exceptions**
- **Formatting**
- **Import Usage**
- **Inheritance**
- **Internationalization**
- **J2EE** (EJB, JSP, Portlets, Servlets)
- **Javadoc Conventions**
- **JDBC Usage**
- **JUnit Usage**
- **Logging** (JCL, log4j)
- **Manifest Files**
- **Miscellaneous**
- **Modifier Usage**
- **Naming Conventions**
- **Performance**
- **Portability**
- **Possible Errors**
- **Potential Refactorings**
- **Program Complexity**
- **Property Files**
- **Security**
- **Semantic Errors**
- **Spell Checking**
- **Threads & Synchronization**
- **UI Specific**
- **XML**

Flexibility is built into CodePro AnalytiX so that it can be modified to meet specific developer or company needs. Audit rules can be locally disabled for one or more lines in a file. Developers can use the dynamic audit feature to dynamically audit specific packages by using inclusion and exclusion patterns. When dynamic auditing is turned on for a package, opening or saving a class in that package will result in it being automatically scanned for audit violations. Developers can also establish project defaults that determine which files are included and excluded from the audit process and which audit rule sets are used for different packages or files.

Manual Auditing

The CodePro AnalytiX manual auditing facility lets developers control what code violations are reported. Developers may want to use manual auditing to be able to check their code for conformance to standards before checking it in. When manual auditing is selected, the tool will report violations only for the files that are selected (rather than all open files that are reported as with dynamic auditing).

Batch / Ant Script-based Code Auditing

Another powerful feature is the ability to run the CodePro AnalytiX code audit facility from an Ant script. This allows an organization to ensure that the established coding standards are being met by integrating code auditing into their build process. Failure to meet standards can optionally cause the build to fail. The results of the audit can also be sent (via e-mail) to the appropriate people.

Automate Code Standards using Audit Rule Sets

Companies often find that it is difficult to get developers to comply with company coding conventions, best practices and code formatting rules. Differences in coding style can lead to code errors and reduce code quality, costing companies both time and money spent trying to locate coding violations. CodePro AnalytiX provides the ideal solution to this dilemma. There are over 770 Java-based coding rules in 30 categories built into the tool (see the sidebar). The audit rules that ship with CodePro AnalytiX are based on industry standard books and best practices, as well as extensive customer feedback, and years of software development experience.

Developers can select the menu item  Audit Code from either the Eclipse menu or a context menu. The enabled audit rules in the default audit rule set will be run on the selected resources. This produces an *audit violation result set*, which is then displayed in the Audit view. Other audit rule sets, such as the pre-defined "The Elements of Java Style" and "Effective Java", may be also be run. With CodePro AnalytiX, the time-consuming task of locating code violations can be performed with a few keystrokes

Audit View Benefits:

- Displays the code audit violations found by the tool
- Shows the severity of the code violation
- Is configurable so that audit violations can be grouped by audit rule, category, severity, resource or author to provide more information to help troubleshoot the violation
- Allows double clicking on any violation to locate the specific violation
- Makes trouble-shooting a code violation easy

Customize Audit Rules to Conform to your Corporate Standards

CodePro AnalytiX audit rule sets are flexible—it is easy to modify or add audit rules to incorporate your best practices coding standards so that they become part of the tool. Developers access audit rule sets using the Preferences dialog where they can create new audit rule sets and share existing sets. They can also modify existing audit rule sets by turning rules on and off, setting the level of severity, or modifying the behavior of the audit rules. Most of the audit rules can be customized to further fine tune the behavior of a specific rule.

Use Project-Level Audit Rule Sets

Another powerful facility is the ability to define per-project audit rule sets to be used for specific projects. Once project-level audit rule sets are created, they can be shared with other teams or across locations to easily apply the same set of code audit rules for shared projects.

Displaying Code Violations using Audit View

Each time code audit runs, CodePro AnalytiX displays the results of the code audit in the Audit View as shown in the following figure. The name of each audit rule with a violation is shown at the root of the tree. Expanding any audit rule will show the list of individual violations (with their locations).

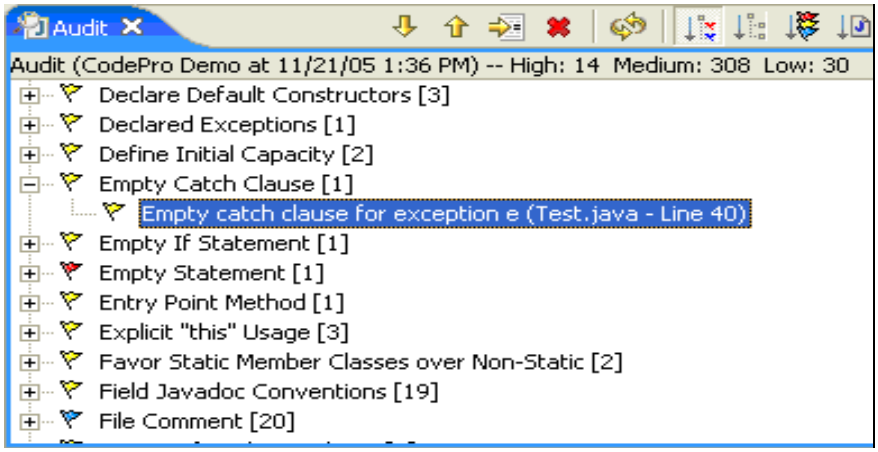


Figure 2: Audit View showing code violations

Double-clicking on any violation will open the corresponding compilation unit and position it at the location of the specific violation. Since CodePro AnalytiX is integrated into Eclipse, developers can see the color-coded severity flags from within the Eclipse Java Editor as shown in the following figure.

Code Repair Benefits:

- Displays Quick Fix hints associated with the code violations found by the tool
- Provides detailed explanations on how to fix the violation
- Lets you easily fix the violation
- Provides training tools to help developers learn how to code according to recommended best standards

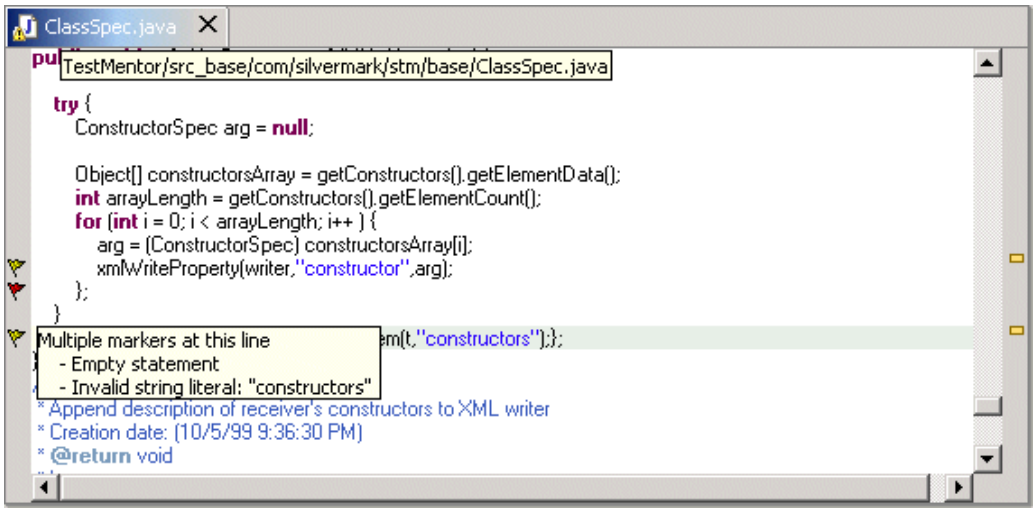


Figure 3: Displaying violation at the code location

Code Review Cost Savings

On small development projects, an organization would typically do weekly code reviews with a staff of four. The preparation and meeting time for each person is four hours. That translates to 16 hours of total time allocated to the weekly code review. With the CodePro AnalytiX audit, metrics and Javadoc repair that 16 hours can be reduced by 75% to a total of 4 hours. That translates into a total saving of 12 hours per week.

For larger development projects, weekly code reviews might involve up to 20 developers. The preparation and meeting time for each person is four hours. That translates to 80 hours of total time allocated to the weekly code review. With AnalytiX audit, metrics and Javadoc repair, that 80 hours can be reduced by 75% to a total of 20 hours. That translates into a total saving of 60 hours per week.

Assuming an annual fully burdened salary of $150,000 per developer this translates into a cost reduction of $900-$4,500 per week / project or $46,800-$229,500 per year / project.

Code Repair

CodePro AnalytiX has built-in features that can find and fix problems found in code or Javadocs. The tool can be set to automatically fix problems or display details on the problems so the developer can analyze the findings. ***The ability to find and repair code or Javadoc problems early in the process can result in dramatic cost saving for companies using CodePro AnalytiX*** as described in the sidebar example.

Explaining the Code Violation

Code Pro AnalytiX can provide an explanation about the code violation and give recommendations on how to correct it. The "**Explain**" command will show an explanation of what caused the selected audit violation or a description of the selected audit rule. If there are any recommendations for fixing the problem, they will be displayed as shown in the sample below.

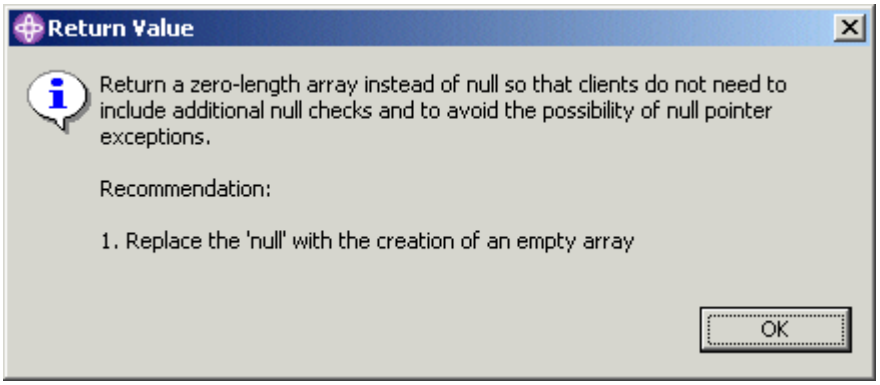


Figure 4: Explanation of a code violation with recommendations

Audit Reports Benefits:

- Valuable management tool
- Code Violation Reports can be generated in HTML, XML and text format and then printed or forwarded to other team members
- Developers can use the Audit Series Editor to customize a wide-range of management reports

Quick Fix Option

CodePro Audit tightly integrates with the built-in Eclipse **Quick Fix** option that can be used to fix the problems found during the code audit. Using the Quick Fix tool can save developers time by automating the process of fixing the violations.

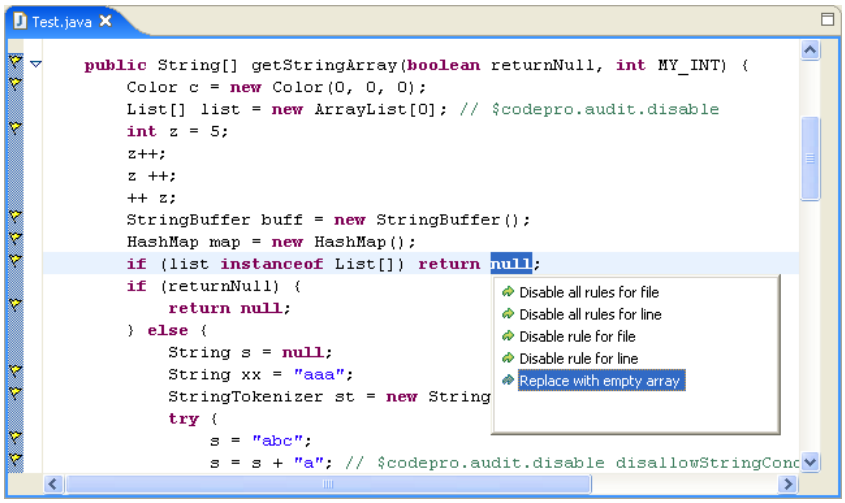


Figure 5: Quick Fix suggestions

Fixing the Code Violation

The "**Fix Violation**" command lists any specific resolutions supplied by the audit facility. Clicking on a fix will cause it to be applied to the selected violations. In some cases, ignoring the violation is the correct course of action, so several options for locally disabling the rule are also supplied. Some resolutions may be applied to multiple violations at one time, being able to apply a rule to multiple violations is a fast and easy way to apply best practices or fix code problems in multiple sites or with several development teams. Seeing what is marked as a code violation is a valuable tool in helping developers quickly learn how to write quality code

Audit Reporting

Application development managers benefit from receiving reports that characterize the code quality efforts of their development team. This includes measurements like lines of code, number of packages, methods, fields and level of documentation. CodePro AnalytiX meets this need by providing extensive management reporting capabilities. CodePro AnalytiX can generate a full range of reports in HTML, XLM, and text format that indicate what the tool found during a code audit. In addition, it is possible to generate e-mail and send individual reports to selected users or managers. The report feature is a valuable tool in tracking the problems in code, determining what needs to change, and providing the data needed to determine how to incorporate best practices features into a corporate coding structure. The following HTML report shows an example of what CodePro AnalytiX found during a code audit.

CodePro AnalytiX Lets you Customize Reports

CodePro AnalytiX can generate a wide variety of reports in a variety of formats. In addition, it is designed so that developers can easily set up reports that contain information that meet the reporting needs of their managers and company.

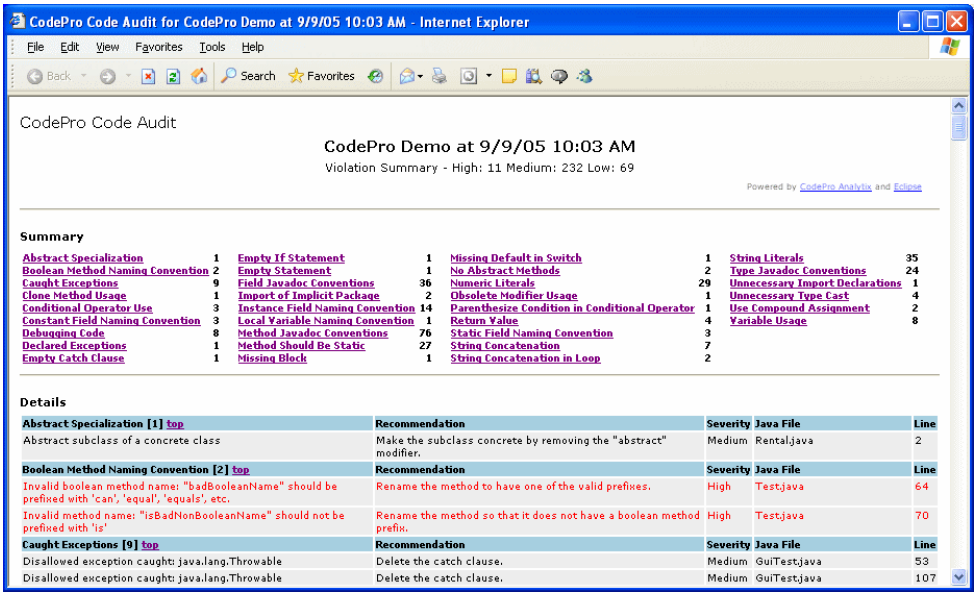


Figure 6: Audit Report in HTML

Define Reports using the Audit Series Editor

The **Audit Series Editor** is a multi-tabbed editor that is used to specify how audit violation set information is displayed on reports. This is a powerful tool that can be used to create management reports that show information in a meaningful manner in a variety of graphical and tabular report layouts. For example, you could use the Audit Series Editor to generate management reports displaying total violations over time, violations based on severity level, violations based on the project / package, violations occurring in certain parts of the system, or code violations by individual. Such information is critical for managers to see trends in the code audit data and determine where changes are needed.

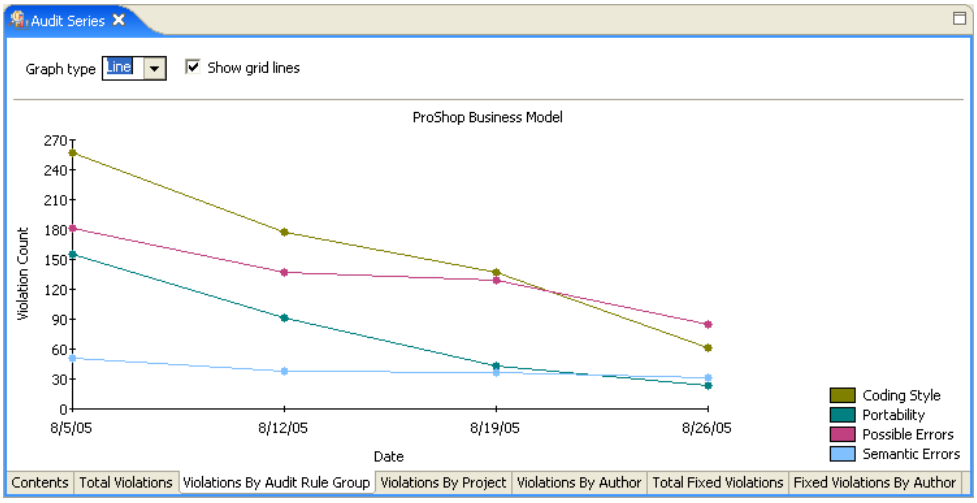


Figure 7: Audit Series report showing violations by category

Metrics Benefits:

- Developers can set metrics in a variety of categories letting them customize the tool to match site requirements
- Metrics exceeding their defined threshold show in a highlighted color
- Metric results can be copied into e-mail messages, text documents, HTML documents, XML documents, spreadsheet applications or even exported
- The metrics tool is valuable in sharing information across teams or locations

Metrics

Development managers find it important to evaluate software against coding benchmarks to assess code characteristics across projects or for individual developers. The CodePro AnalytiX metrics facility can provide managers with vital information about projects and developer performance. CodePro AnalytiX allows developers to tailor metrics preferences according to their project requirements. Developers can choose to capture metrics in basics, complexity, dependency, halstead, inheritance and ratio categories. When developers select Compute Metrics, the enabled metrics in the default metric set are run and displayed in a table. The individual metrics may be expanded (drill down) to show the value of that metric for project, package and type. Any metrics that have exceeded their user-defined thresholds and trigger points are shown in a highlighted color. Results can be captured to the clipboard so that they can be pasted into e-mail messages, text documents, HTML documents, XML documents or even a spreadsheet application. Developers can also export the results to a file in simple text, tab separated, comma separated, HTML or XML format. The resulting information can then be sent to development manager for review and analysis.

Dependency Analysis Benefits:

- Locating a dependency helps identify where code errors may occur across projects, packages or types
- This tool helps determine how a change can effect other code
- Dependencies are displayed in a graphical format
- Dependencies can be displayed at different levels of granularity
- Color-coding makes it easy to see where circular dependencies exist

Showing Dependencies (Dependency Analysis)

Complex software typically includes code that is invoked by or depends on the behavior of other code. Developers find that it is very time-consuming to find dependencies and determine how the effect of a change in one area will cause changes in other parts of the system. For example, changing code in one class might affect code in several seemingly unrelated classes. Developers need a tool that can show code dependencies and help them troubleshoot code problems. CodePro AnalytiX meets this need—it contains a facility that lets developers perform a dependency analysis on one or more projects or packages.

Selecting **Analyze Dependencies** from the CodePro Tools submenu lets developers show dependencies between projects, packages, or types. Finding dependencies is important because it helps developers determine how changes will impact other portions of the software and helps prevent propagating errors across multiple projects.

Dependency Reports can be displayed showing the results of a dependency analysis. The Dependency Analysis Report in an HTML format lists information associated with the projects, packages, types or classes used in the analysis. CodePro AnalytiX also displays dependencies in a graphical format. The elements visible at each level of granularity (projects, packages, or types) are displayed as rectangles labeled by both an icon indicating the kind of element being viewed and the name of the element. Dependencies between the elements are displayed as directed lines (lines with arrows at either one end or both). The elements are divided into three levels of granularity.

The following screen shows a sample of a dependency analysis graphic.

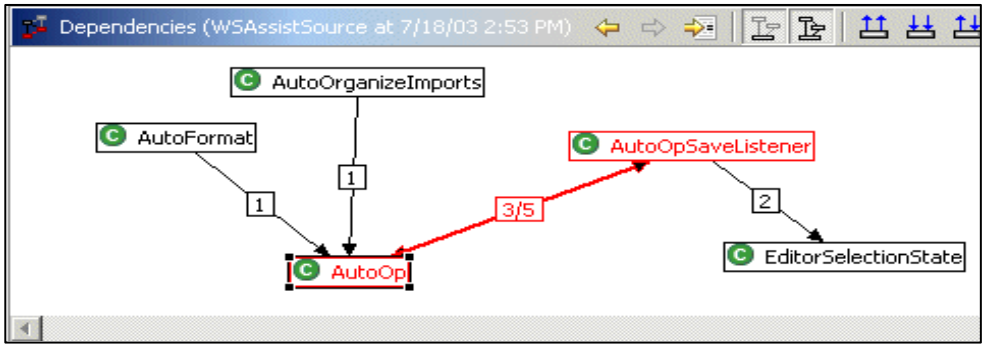


Figure 8: Dependency Analysis graph

Automatic Test Case Generation and Code Coverage

Most companies that create software have entire departments made up of Quality Assurance (QA) and testing teams whose job function is writing test cases and performing code coverage analysis looking for errors and verifying that every line of code has been tested. This task is a major cost for companies and often the QA and test teams still don't find all the errors in software. CodePro AnalytiX helps meet this need by generating unit tests and analyzing code coverage, helping drive out problems early in the development process.

The **CodePro JUnit Test Case Generation** facility lets developers automate the creation of comprehensive JUnit test cases. Given an input class, the tool creates a corresponding test class complete with multiple test methods for each input class method. The tool analyzes each method and input argument with the goal of generating test cases that exercise each line of code. The JUnit test cases created by CodePro AnalytiX provide the developer with a variety of unit-level test cases. Developers should review these test cases and use them as the starting point for writing integration tests.

The **CodePro Code Coverage** facility measures how much of the code is being executed. Perhaps the most common reason for measuring code coverage is to evaluate the effectiveness of test code at exercising all possible paths through the code. The tool measures the number of basic blocks of code that complete normally without exceptions and correlates it back to the source code on a line-by-line basis, providing information on code that was executed, partially executed, or not executed at all. Each time a particular class is measured, a coverage report is generated showing the results of the coverage. The developer should review the code coverage results and write new test cases to test any untested code. The following figure shows a sample of the Code Coverage View.

Benefits of Automatic Test Case Generation and Code Coverage:

- Higher quality software is produced
- Higher Code Quality translates into less QA/Testing time required
- Problems are much cheaper to correct at the point-of-origin rather than after-the-fact during QA or testing
- Generates JUnit tests that developers can use as the basis for writing integration test cases
- Code coverage results provide valuable information that can be used in troubleshooting
- Helps locate dead code and logic errors

Benefits of Collaboration:

- Allows distribution of Audit Rule sets to geographically distributed development teams
- Users and Groups can be created with assigned tasks
- Allows teams know the status of projects and who is assigned to various tasks
- Each user (assignee) can see all the tasks assigned to them
- Great for highly distributed teams and outsourced development

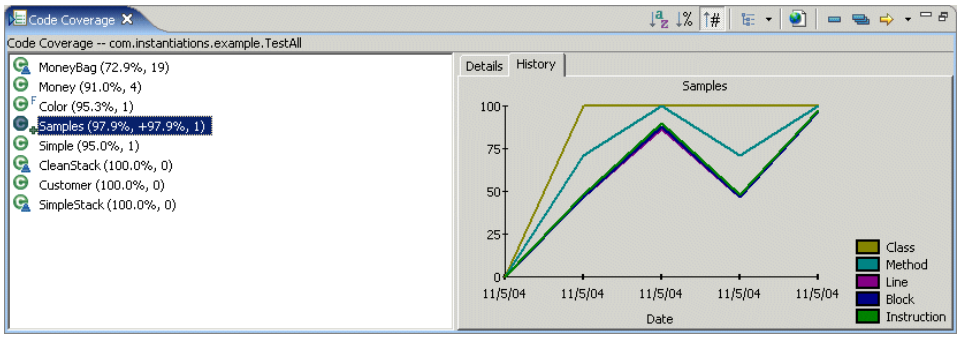


Figure 9: Code Coverage View

Collaboration—Share Information and Tasks

With the CodePro AnalytiX Collaboration facility, developers can share information, view and modify team assignments and send messages through a variety of methods. Developers can use the code audit preference pages to share audit rules by exporting rule sets to the file system, to an Eclipse project or via collaboration. This is a powerful feature because once the audit rules have been set and best practices incorporated in the rules, then the rules can be shared so that everyone is coding against the same standards. In addition, developers at all corporate locations can have access to the "mentoring tips" of CodePro AnalytiX so that even junior developers or those not familiar with the system can quickly write quality code with fewer errors.

Team Collaboration

Teams can collaborate and share information using the CodePro AnalytiX collaboration feature and Collaboration Server. The Collaboration Server acts as a "store and forward" mechanism which allows an administrator or other specified users to create user and group accounts and set up Team Tasks. The Team Tasks View shows the task identifier, summary information about a task, state (assigned, completed, postponed, etc.), and severity level of the task.

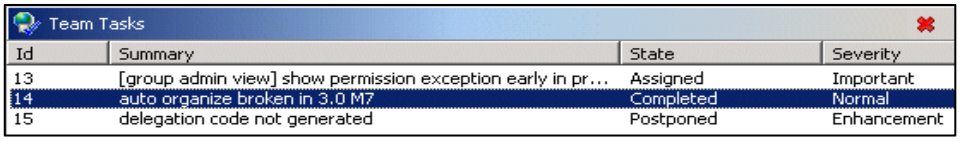


Figure 10: Team Tasks View

Once created, a team task automatically appears in the Team Task View in the assignees' workbench. Only group leaders and the administrator may assign a team task to users besides themselves. The collaboration features save companies both time and money by helping teams know the status of projects and developers understand their assigned tasks. ***Using the sharing and collaboration features lets companies define, distribute and enforce the coding quality measures across development teams—no matter where they are located.***

Benefits of using CodePro AnalytiX:

- Improves code quality
- Reduces development costs
- Shortens code review time
- Ensures that developers comply with coding standards and best practices
- Assists developers produce clean and efficient code
- Ensures predictable results
- Faster time to project delivery

"CodePro AnalytiX audit capability allows us to enforce many programming standards at development time, instead of having to rely on extensive and lengthy code walkthroughs. This is a huge benefit that allows Covansys to develop cutting edge solutions, while meeting and surpassing our client's expectations and critical time to market needs."

Randy Thomasson, Technology Director, Covansys

Summary

The traditional methods of developing and testing software too often results in poor quality code and project and cost overruns. What is needed is a new way to think about software development. Instantiations defined and uses the principal of *Continuous Collaborative Code Analysis* (C^3A) which is designed to aid developers in finding and fixing problems in their code earlier in the development process.

CodePro AnalytiX, which seamlessly integrates into the Eclipse IDE, is based on the C^3A concept and ensures superior code quality and maximum developer productivity by including code audit, metrics, testing, and team collaboration in one tool. Using CodePro AnalytiX, feedback is provided to the developer while code is being written helping to eliminate quality problems earlier in the development cycle. CodePro AnalytiX makes it easy for developers to add their own best practice rules and then distribute these rules to all their teams so all developers are using the same coding standards and receive feedback on their code. CodePro AnalytiX can automatically generate unit tests and analyze test code coverage which saves developer time and helps create tight, maintainable systems. Finally, the CodePro AnalytiX collaboration features make it easy to define, distribute and enforce corporate coding standards and quality measures, across individuals and teams—no matter where they are located.

Contact Us

http://www.instantiations.com

info@instantiations.com

(503) 598-4909

(800) 808-3737