



CASE STUDY:
**AgitarOne Ensures Quality
of Outsourced Software Development
at Major International Bank**

One of the major international Banks has adopted AgitarOne (www.agitar.com) technology for delivering generated unit tests for their Java software development. The Bank services millions of customers and is a leading provider of current accounts, savings, personal loans, credit cards, mortgages, etc. AgitarOne was able to substantially reduce the cost of software development and significantly improve the software quality process.

Through implementation, the bank discovered enormous financial gains in uncovering defects and reducing software complexity in the code base being maintained by their various outsourcers and contractors.

We tell the story here about why AgitarOne was purchased, what was found, and how it assists management with the software development and quality process today.

Reason for Purchase

In order to reduce the number of defects appearing in the integration testing and User Acceptance Testing (UAT) phases, a decision to adopt a more formal unit testing process was made. The “legacy” code base contained very few JUnit tests, and no real indicator of the efficacy of these tests.

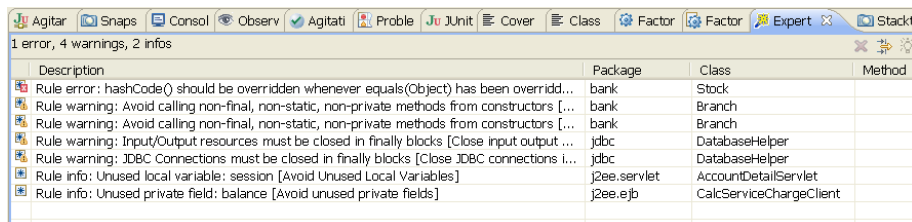
The immediate challenge was to generate the JUnit tests required to provide complete unit test coverage of the code. After reviewing the tools available, the Bank selected the AgitarOne solution for its classic automation application.

Volume/Scale

AgitarOne’s mission for the Bank’s software included scaling a massive code base. More than 5 Million lines of Java code (LOC) across various systems comprising more than 25,000 Java classes. The combined systems are primarily J2EE using a wide range of infrastructure products such as Hibernate, Struts, JSF, Trinidad, together with a market leading database and Internet framework. With this volume of code for such critical financial systems, there is no doubt that quality was a concern.

Quality Improvement Milestones

Quality is often described as a journey, but without measurement, it is a pretty pointless one, so the first stage was to quantify the issues. One of the components of AgitarOne is Code Rules, based on the widely used CheckStyle software. It includes the automated enforcement of customizable code rules, standards, and guidelines to ensure that a consistent and unified process is used.



The screenshot shows the Agitar IDE interface with a list of code quality issues. The issues are categorized by severity: 1 error, 4 warnings, and 2 infos. The table below represents the data shown in the screenshot.

| Description | Package | Class | Method |
|--|--------------|-------------------------|--------|
| Rule error: hashCode() should be overridden whenever equals(Object) has been overridden... | bank | Stock | |
| Rule warning: Avoid calling non-final, non-static, non-private methods from constructors [...] | bank | Branch | |
| Rule warning: Avoid calling non-final, non-static, non-private methods from constructors [...] | bank | Branch | |
| Rule warning: Input/Output resources must be closed in finally blocks [Close input output ...] | jdbc | DatabaseHelper | |
| Rule warning: JDBC Connections must be closed in finally blocks [Close JDBC connections i...] | jdbc | DatabaseHelper | |
| Rule info: Unused local variable: session [Avoid Unused Local Variables] | j2ee.servlet | AccountDetailServlet | |
| Rule info: Unused private field: balance [Avoid unused private fields] | j2ee.ejb | CalcServiceChargeClient | |

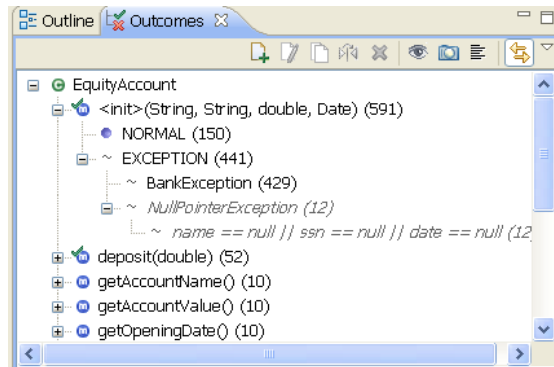
At the Bank, the in-house quality team reviewed the available AgitarOne rules and selected the options that, based on widely used industry norms, gave consistent feedback and reduced the potential for future

errors. AgitarOne evaluates these rules during builds and protects the software from error-prone coding patterns and unwanted bugs.

Initial scans with some of the packages revealed an astounding 5,000 code errors, indicating an unchecked maintenance process. A plan is in place to progressively reduce these totals over time as the classes undergo maintenance in line with the evolving requirements of the business.

Another major component of AgitarOne is Agitator, which assists developers in validating expected behavior of their code while writing new code. During the unit testing phase, to really unit test code, every line, every branch, and every outcome must be tested ... a daunting task. It is simply not practical to create such exhaustive tests manually. Agitator automatically creates dynamic test cases and analyzes results.

During the agitation process at the Bank, Agitator quickly revealed where potential Java run time errors could occur in the code, errors that could bring testing to a halt and require expensive re-work and delay. Initially, it was not uncommon to find as many as 10 potential Java runtime errors in classes being agitated. The sheer volume of errors discovered through the use of Agitator quickly became an indicator of how beneficial this tool would be to the Bank and in addition, shed light on the quality of software work that was being delivered.



Impact of Outsourcing

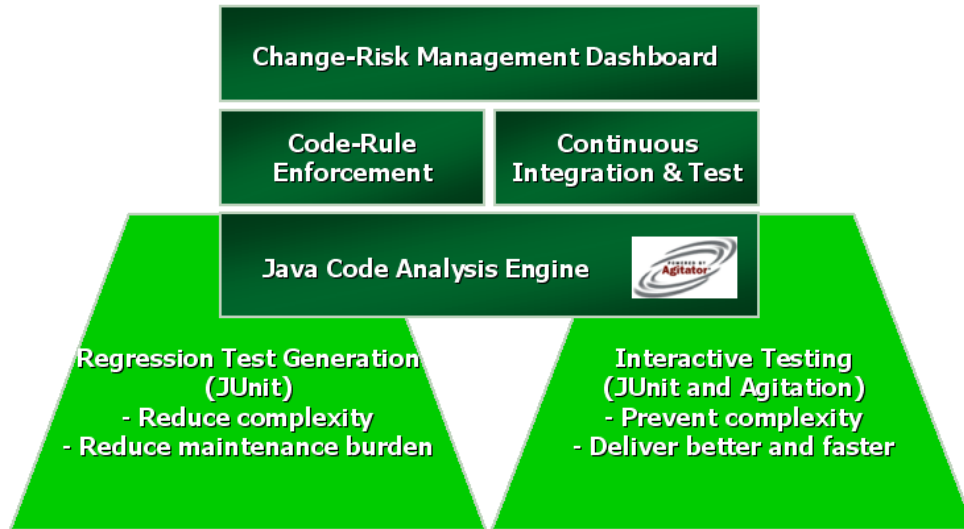
In the early days of outsourcing, the long term software quality impacts to development and maintenance processes were not always carefully considered. This forms one of the core reasons why quite often the savings envisaged at the outset have perhaps not materialized in the longer term. Adopting AgitarOne through the Bank's process has provided concrete data to quantify the alarming volume of high risk code.

Outsourcing and, more specifically, the control of quality in such an environment is one of the major challenges for modern IT departments. This Bank is no different except perhaps for the added layer of regulatory requirements based around SOX.

By not only providing productivity improvements in test generation and adding rigour to unit testing, AgitarOne provides an extensive measurement capability, together with a comprehensive drill down Dashboard, enabling the Bank to identify high risk code.

The AgitarOne Tool

The AgitarOne product family comprises five main components built round an advanced Java Code Analysis Engine.



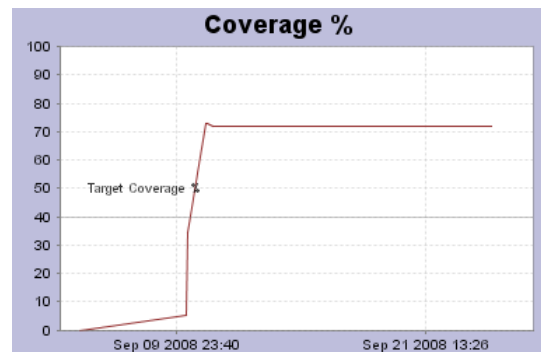
AgitarOne JUnit Generator creates high quality JUnit tests on code finding regressions thereby making it safer and easier to improve code and reduce the cost to maintain it. AgitarOne Agitator helps developers understand the behavior of code as it is written, preventing bugs and code complexity that become tomorrow's maintenance headache.

Additional functionality includes program code rule enforcement and integration into the CruiseControl continuous Integration engine. All of this feeds a management dashboard designed to provide comprehensive feedback to empower project leaders and show objective data on the unit-level quality and status of Java projects, providing continuous visibility into the unit tests.

The AgitarOne Process

Within the AgitarOne perspective, all the code that comes back for review at the Bank is examined for a number of key indicators:

- Ensuring generated tests for all classes
- Exercising 90% test coverage (Branch Coverage) for each method
- Validating no classes with zero coverage
- Recording no Java Runtime Exceptions
- Documenting all throwable exceptions
- Verifying at least 50% of expected outcomes have been asserted
- Proving no code errors or warnings in new code
- Confirming total number of code errors/warnings in the modified class has not risen and is falling



In this way the reviewers are able to satisfy themselves that the code has been developed in a rigorous manner and testing time will not be wasted uncovering 'silly coding mistakes'.

Cost of Ownership

When one considers the cost of equipping a modern Java developer with the full range of industrial strength development tools, we are typically looking at approximately £5K per seat. The cost of AgitarOne adds some £300 per developer for organizations such as this. Software development teams don't always realize how inexpensive it is to run automated testing tools.

The hardware required to support AgitarOne comprises both a Workstation (Eclipse family) component and a dedicated multi-Server component, however with powerful workstations the requirement for servers can be reduced. The Server component recognizes and makes use of the individual processors that are available in modern multi-core servers, thus further reducing the physical number of 'boxes' required.

Ease of Adoption

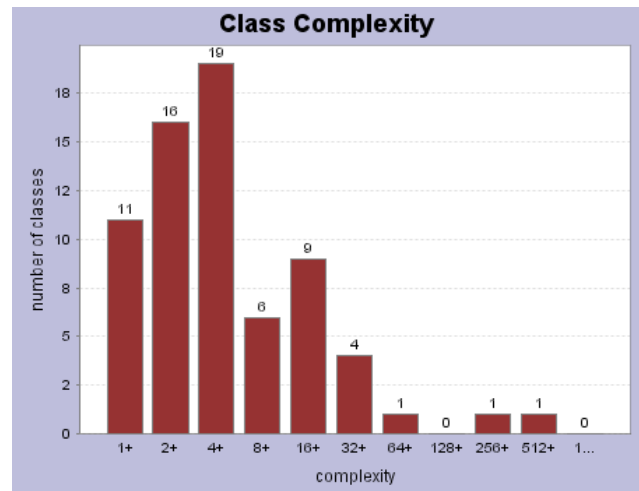
Software training courses should be provided when implementing AgitarOne. Training can be delivered in a variety of modes, ranging from the basic self tutorial that is delivered with AgitarOne, to the formal one day for basic developer usage, and going up to three days for a comprehensive understanding of all the principles. The training is modular and thus can span a number of days, as was the case at the Bank. Here a number of groups were trained, each in three sessions spread over three weeks, thus fitting in with their work commitments.

Summary

The UK bank was faced with a classic modern problem, namely legacy code written by a number of contractors on agreements where perhaps the long term maintenance costs were not discussed. To this mix was added the outsourcing of a substantial portion of the code base, together with new development to an offshore partner, without enough focus on long-term code quality. All testing is performed by teams of contractors and/or another offshore partner. This has resulted in a wide range of quality issues, delayed projects, mounting maintenance costs, and a loss of business agility.

The Bank chose AgitarOne as part of the fight back to bring its seemingly untamed code base under control. AgitarOne provides the Bank with:

- Generated JUnit characterization tests to monitor change
- Agitation to ensure new code has been unit tested
- Code rules to weed out non-standard programming practices
- Integration with ClearCase and Rational Application Developer
- Continuous Integration with reporting using CruiseControl
- Multi-tiered Dashboard to provide simple at-a-glance metrics and charts



About Agitar Technologies, Inc.

Agitar Technologies enables the enterprise to release Java applications faster, reduce the cost of bugs, and more easily change both new and legacy applications to meet changing business needs. The “AgitarOne” product family enables software teams to create, use, and manage the extensive set of unit tests needed to be truly agile. Customers have cut legacy maintenance efforts by 50%, released new applications 30% faster with fewer bugs, and cut the cost of finding and fixing post-release bugs by 90%. Agitar Technologies is a privately-held company headquartered in Cranston, RI, U.S.A. For more information visit www.agitar.com.