Develop Software with Confidence

Our Mission

Profoundly improve the **quality** of software
and the **economics**, **transparency** and **ility**
of softw...

**By making developer testing
effective for the enterprise**

Agitar's Mission

Agitar Software is focused
on **One** thing:

Helping software teams
succeed at developer testing

Goals of
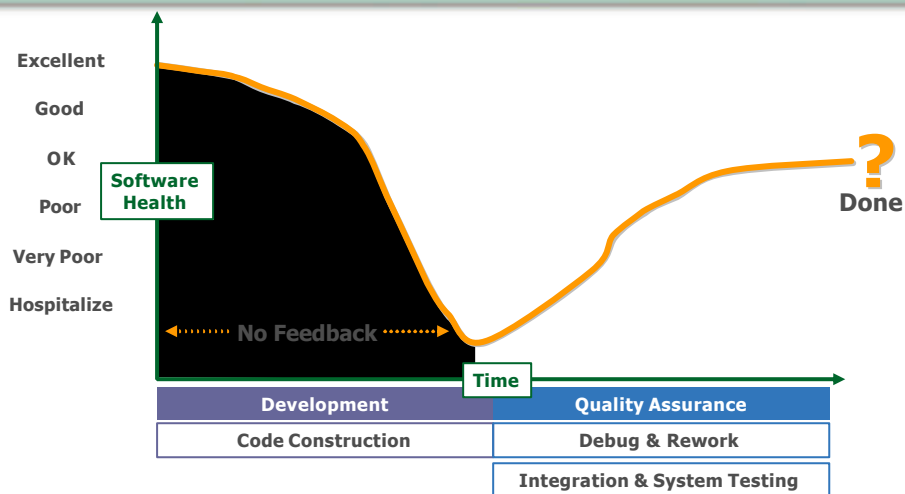## Software Development Organizations

Release software that works,

meeting the business need for

new functionality,

when it's needed,

at the budgeted cost

What Do They Need to Succeed?

- The organization as a whole
  - More free cycles to rapidly deliver new functionality
  - Less drag from fragile legacy applications
- Software developers
  - Confidence that the code they create is correct, and that it can be changed later without the fear of breaking something
- Quality Assurance and integration teams
  - Clean code that's easy to integrate and test at the system level
- Managers and team leaders
  - Visibility, with objective data, into the project's real status

---

The Traditional Approach Is Flawed

Source: "The Software Development Paradox", Alberto Savoia, 2004

Software Development
Is Still Immature . . .

$ 100+ Billion

Wasted annually on software bugs [1]

Only 29%

Of IT projects succeed [2]

50%

**Of all software projects
are total failures [3]**

84% Longer

Than originally scheduled [4]
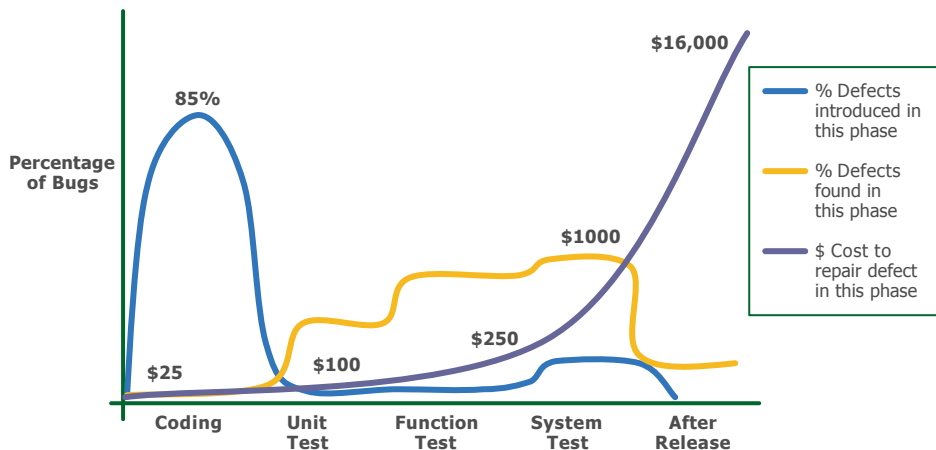
Sources: 1: US NIST study, 2002; 2 and 4: Standish Group data contrasting 2004 with 2002; 3: Cutter Consortium, May 2005

The Economics Are Compelling

$16,000

85%

**Percentage
of Bugs**

— % Defects introduced in this phase

— % Defects found in this phase

— $ Cost to repair defect in this phase

$1000

$250

$25   **$100**

Coding   **Unit Test**   **Function Test**   **System Test**   **After Release**

Source: Applied Software Measurement, Capers Jones, 1996

There Is a Better Approach

Test Bugs Out

Build Quality In

vs.

Typical software development process

- Give QA the main responsibility for quality
- Test late
- Use unverified components
- Deliver, and fix defects later
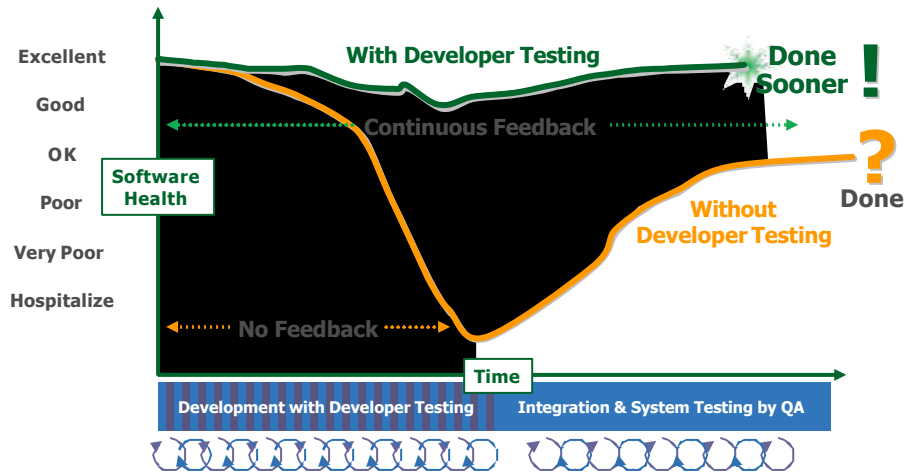
Mature world-class manufacturing process

- Hold everyone responsible for quality
- Test early and often
- Use verified components
- Stop the assembly line

Developer Testing Works

The **One** best thing you can
do to dramatically improve
your software is to help
developers test their code

Find and Fix Defects Sooner

Source: "The Software Development Paradox", Alberto Savoia, 2004

The Challenge

One of the hardest things to do
is to enable developers to start
to do developer testing well,
and to sustain it

A Few of Our Customers

What Our Customers Say They Want

Agitation, Dashboard reporting, code-rule enforcement

Largest
S...

Ma...
Mu...

"Using Agitar is the best way
to improve development"

...eduction in bugs
...aching QA

Major Fi...
S...

...% reduction in
...ing hours per project

Large...

Global S...
Hea...

1st 0% downtime quarter,
95% cut in cost of bugs

90% cut in
defect remediation cost

Our Customers Have Asked for Ways to:

Agitar

- Have faster, predictable test execution
- Get Java legacy applications under control
- Generate some basic tests, fast, with no overhead
- Educate novices about developer testing
- Deliver JUnit tests to meet contract deliverables

From Agitator to AgitarOne

Agitar

- Agitation, code rules, and Dashboard as before
- Use agitation to **automate generation of JUnit tests**, immediately
  - Regression tests on legacy code
  - Fast smoke tests
  - Contractually-required JUnit delivery
- More natural ways to train JUnit generation and agitation
  - Write **data helper** and **observation helper** methods, to complement traditional Agitator techniques of factories and mocks
- **Server-based** for easy deployment and superior scalability
  - Lower desktop resource usage
  - Easier sharing of artifacts across the organization
  - Distributed agitation and JUnit generation for faster results
- **Simplicity** of adoption, **ease** of usage
  - Adds value more easily at any time in the project lifecycle

Add AgitarOne to Your JUnit Testing

Agitar®

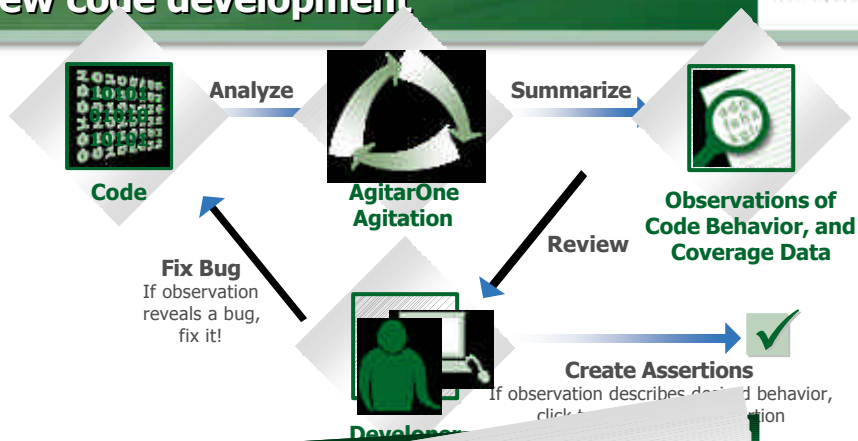**Kent Beck**   **Agitar Fellow**
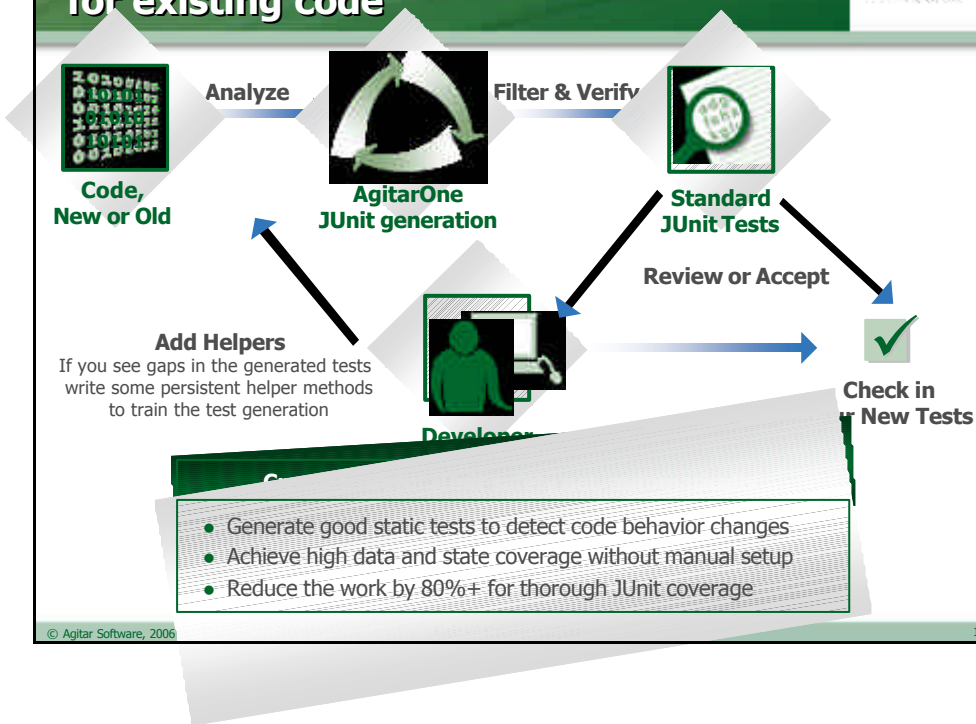
JUnit   **+**   **AgitarOne**

% of our customers start with JUnit

- AgitarOne and Dashboard understand and leverage existing JUnit

---

Software Agitation with Agitator for new code development

Agitar®

Analyze   **Summarize**

Code   **AgitarOne Agitation**   **Observations of Code Behavior, and Coverage Data**

Review

Fix Bug
If observation reveals a bug, fix it!

Create Assertions
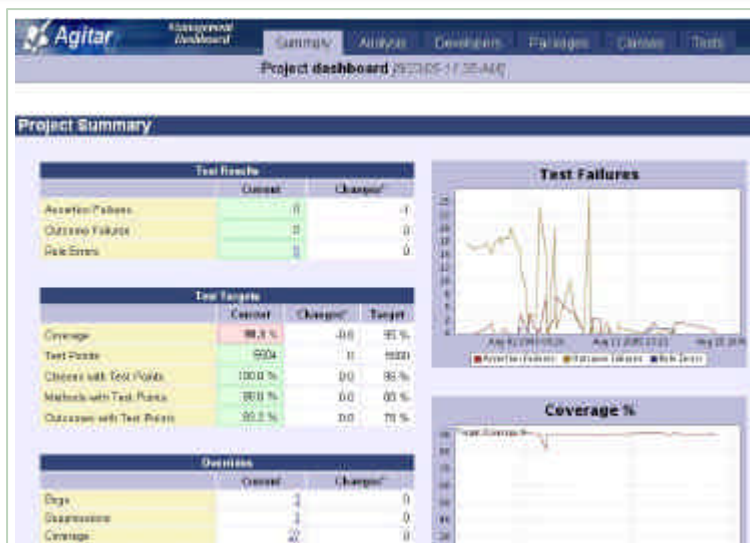If observation describes desired behavior, click to save assertion

Developer

- Discover invariants (or specify and validate them)
- Achieve high data and state coverage without manual setup
- Test without needing application server, database, etc.

Agitation Powers JUnit Test Generation for existing code

Analyze → **Filter & Verify** →

Code, New or Old

AgitarOne JUnit generation

Standard JUnit Tests

Review or Accept

Add Helpers
If you see gaps in the generated tests write some persistent helper methods to train the test generation

Developer

Check in New Tests

- Generate good static tests to detect code behavior changes
- Achieve high data and state coverage without manual setup
- Reduce the work by 80%+ for thorough JUnit coverage

Real Visibility into Unit-Level Quality

Metrics for High-Performance Teams

Project **dashboard**

Test Targets			
	Current	Changes*	Target
Coverage	89.8 %	0.0	95 %
Test Points	5273	5	5000
Classes with Test Points	100.0 %	0.0	95 %
Methods with Test Points	75.5 %	-0.1	80 %
Outcomes with Test Points	76.2 %	-0.0	75 %

Project Code Metrics		
	Current	Changes*
Project Classes	131	0
Test and Harness Classes	189	0
Project Methods	1355	1
Test Methods	416	0
Project Executable Lines	5279	1
Test Executable Lines	7070	0

High code coverage

Every class has assertions

High % of methods and outcomes have assertions
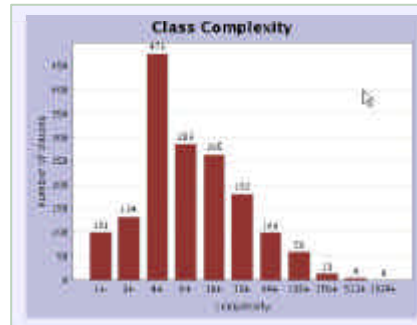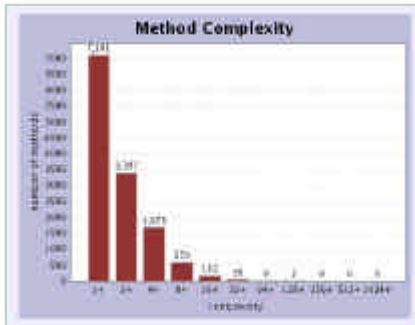
***1:1 ratio of assertions to lines of code
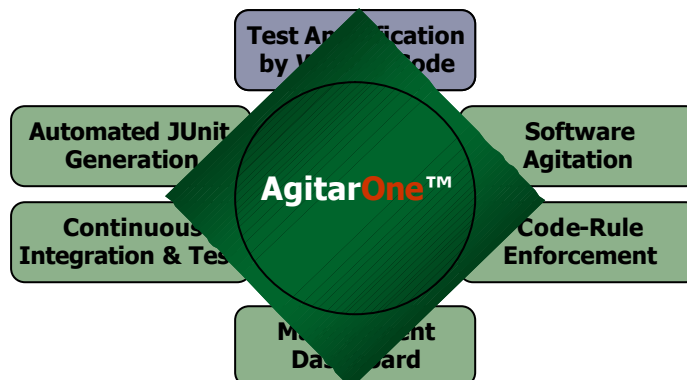

Manage Risk

All Classes

Class	Risk	Complexity	Usage	Methods	Coverage	Test Points Methods	Test Points Outcomes	Failures
com. agitar. analysis. Project	5829	188	31	86	94.5 %	94.2 %	91.3 %	0
com. agitar. analysis. Class	2490	131	19	77	94.2 %	84.4 %	81.0 %	0
com. agitar. dashboard. Config	1951	65	30	49	98.5 %	95.9 %	94.8 %	0
com. agitar. analysis. TestClass	1329	83	16	38	95.4 %	94.6 %	92.9 %	0
com. agitar. analysis. Package	766	85	9	42	96.4 %	100.0 %	100.0 %	0
com. agitar. dashboard. DashboardProject	631	70	9	35	93.4 %	32.3 %	36.8 %	0
com. agitar. analysis. NotBeclMethod	601	75	8	49	95.5 %	83.7 %	82.8 %	0
com. agitar. dashboard. owners. Owner	470	67	7	25	95.0 %	56.0 %	59.3 %	0
com. agitar. analysis. TestMethod	401	50	8	29	99.0 %	100.0 %	100.0 %	0
com. agitar. dashboard. data. CheckpointData	356	71	5	65	98.6 %	100.0 %	100.0 %	0
com. agitar. dashboard. DashboardWriter	337	48	7	24	84.7 %	5.0 %	13.0 %	0
com. agitar. dashboard. PageDataAdapter	313	13	24	10	100.0 %	90.0 %	94.7 %	0
com. agitar. dashboard. data. DataFileReader	273	68	4	27	98.0 %	46.2 %	52.9 %	0
com. agitar. common. utils. Path	225	28	8	14	93.0 %	83.3 %	85.0 %	0
com. agitar. dashboard. ClassDataAdapter	193	32	6	21	93.5 %	100.0 %	100.0 %	0
com. agitar. dashboard. IndexPageAdapter	179	89	2	76	97.8 %	95.9 %	92.1 %	0
com. agitar. dashboard. logging. DashboardLogger	177	8	22	6	100.0 %	66.7 %	83.3 %	0
com. agitar. dashboard. DashboardTemplate	161	10	16	5	91.7 %	66.7 %	66.7 %	0
com. agitator. agitator. AgitationResults	154	153	1	21	85.0 %	100.0 %	75.0 %	0
com. agitar. dashboard. DashboardChartGenerator	136	27	5	10	94.5 %	0.0 %	0.0 %	0
com. agitar. dashboard. TestResults	133	44	3	13	86.9 %	90.0 %	88.2 %	0
com. agitar. dashboard. OwnerDataAdapter	127	42	3	22	96.3 %	0.0 %	0.0 %	0

Manage Complexity

Method Complexity

Class Complexity

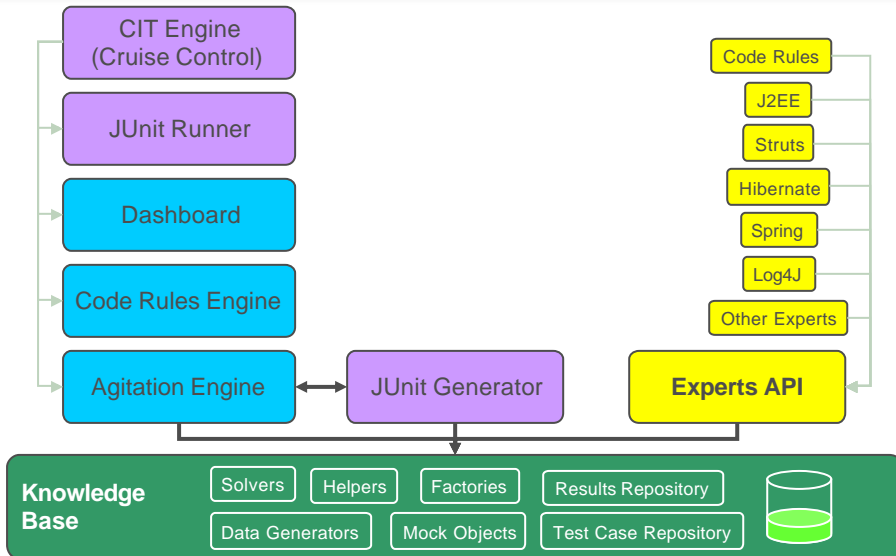
We Listened, and Created One Solution

AgitarOne™ brings unprecedented
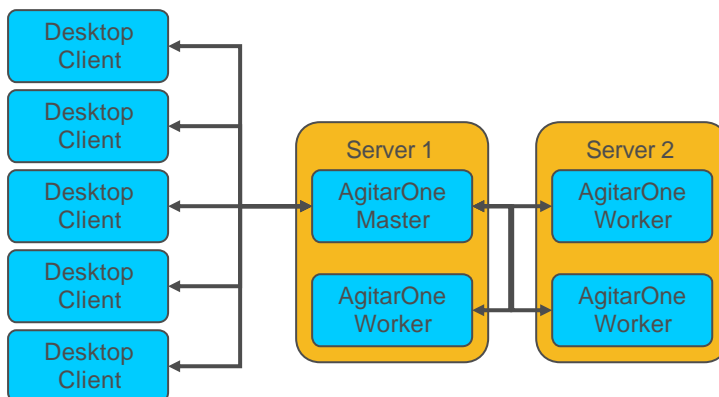thoroughness and ease-of-use to developer testing for Java

Test Amplification by Writing Code

Automated JUnit Generation

Software Agitation

AgitarOne™

Continuous Integration & Test

Code-Rule Enforcement

Management Dashboard

CIT Engine
(Cruise Control)

JUnit Runner

Dashboard

Code Rules Engine

Agitation Engine ⟷ JUnit Generator

Code Rules

J2EE

Struts

Hibernate

Spring

Log4J

Other Experts

Experts API

Knowledge Base

| Solvers | Helpers | Factories | Results Repository |
| Data Generators | Mock Objects | Test Case Repository |

Distributed Agitation and JUnit Creation

- Client(s) downloaded from Server
- Server distributed across one or more machines

Desktop Client

Desktop Client

Desktop Client

Desktop Client

Desktop Client

Server 1

AgitarOne Master

AgitarOne Worker

Server 2

AgitarOne Worker

AgitarOne Worker

One-Time Server Install

The Agitar**One**™ server is easily installed in less than
20 minutes on any Linux or Windows XP system

One Time

You Can Start In One Minute

The Agitar**One**™ client for Eclipse downloads and
installs in approximately **One** minute

One Minute

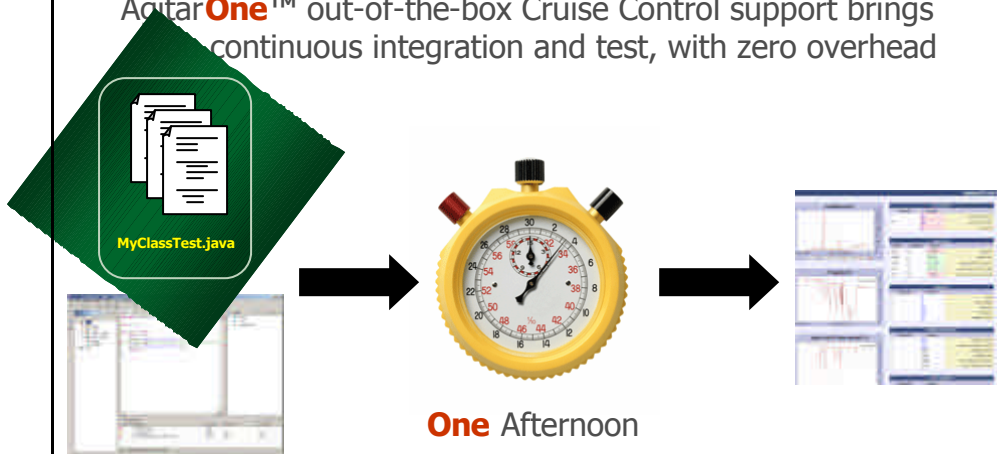**You've Got Tests!**

One minute later, you'll be looking at your first automatically generated regression tests

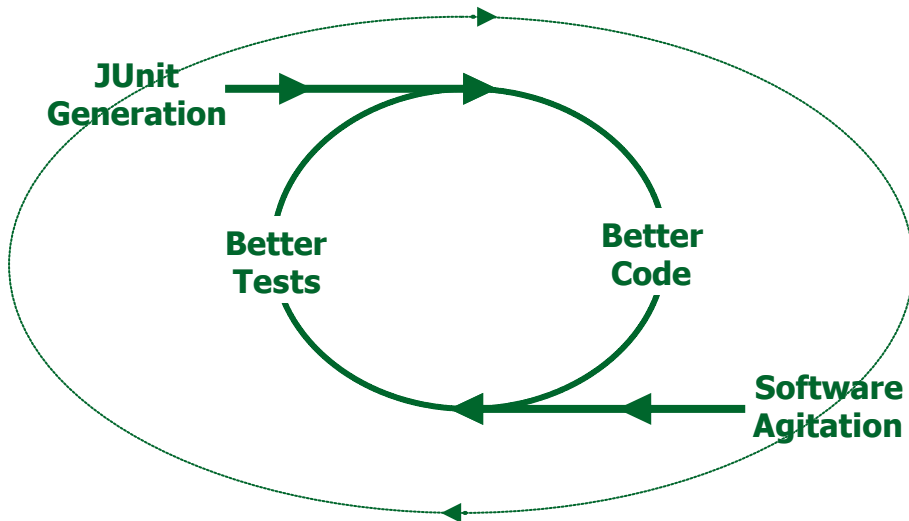MyClass.Java → **One** Minute → MyClassTest.java

One Afternoon to Full Visibility

Agitar**One**™ out-of-the-box Cruise Control support brings continuous integration and test, with zero overhead

MyClassTest.java → **One** Afternoon

Easy Introduction of AgitarOne™

JUnit Generation

Better Tests

Better Code

Software Agitation

Our Customers Have Asked for Ways to:

- Speed up deep large-scale agitation
- Reduce desktop CPU and memory requirements
- Simplify large-scale deployments and rollups
- Rapidly implement continuous integration & test
- Have faster, predictable test execution
- Get Java legacy applications under control
- Generate some basic tests, fast, with no overhead
- Educate novices about developer testing
- Deliver JUnit tests to meet contract deliverables

Agitar®

better code from better tests from

AgitarOne

Agitar®
SOFTWARE

Develop Software with Confidence