IntelliJ IDEA: Integrated Team Environment

Dmitry Jemerov

Software Developer
JetBrains, Inc.
http://www.jetbrains.com/

TS-5033

Integrated Team Environment

Develop with pleasure!

Find out how Team Server—a new product currently in development at JetBrains—improves team productivity

Agenda

Background and Overview

A Better Continuous Integration

- Problems and Solutions

Server-Side Code Analysis

Collaboration

Extensibility

Agenda

Background and Overview

A Better Continuous Integration

- Problems and Solutions

Server-Side Code Analysis

Collaboration

Extensibility

The Road to Team Server

- IntelliJ IDEA started as a productivity tool for an individual Java-based application developer
- Then we introduced support for more languages (XML/HTML, CSS, JavaScript™…)
- Now we move in another dimension—from an individual developer to the entire team

Goal

- Bring team productivity to a whole new level

  …as we've already done for Java-based development

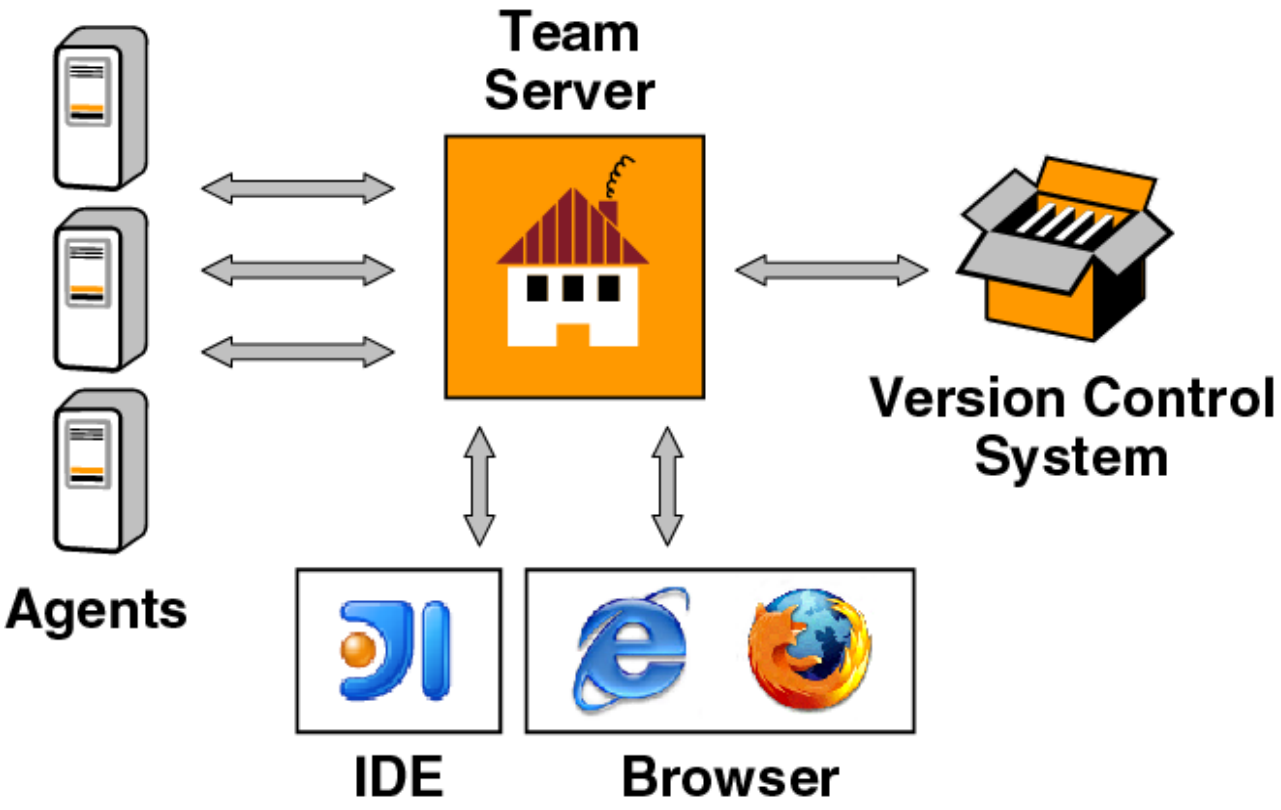
Don’t Work Against People

- People don’t always follow established practices
- Don’t try to police people and force them to obey rules
- Make it natural to follow established practices—as easy, or even easier, than breaking them

Dogfood as Always

- Our teams face the same problems as many other development teams
- Team Server used for development of itself
- IntelliJ IDEA development moved to Team Server as soon as it became possible

Product Overview



Product Overview

- Distributed build environment
  - Server coordinates builds run on multiple agents
  - Managed and monitored through Web interface
  - IDE independent
- Server-side code analysis
  - Results can be viewed in IDE or Web browser
- Collaboration support
  - Send messages, code pointers, changes

Agenda

Background and Overview

A Better Continuous Integration

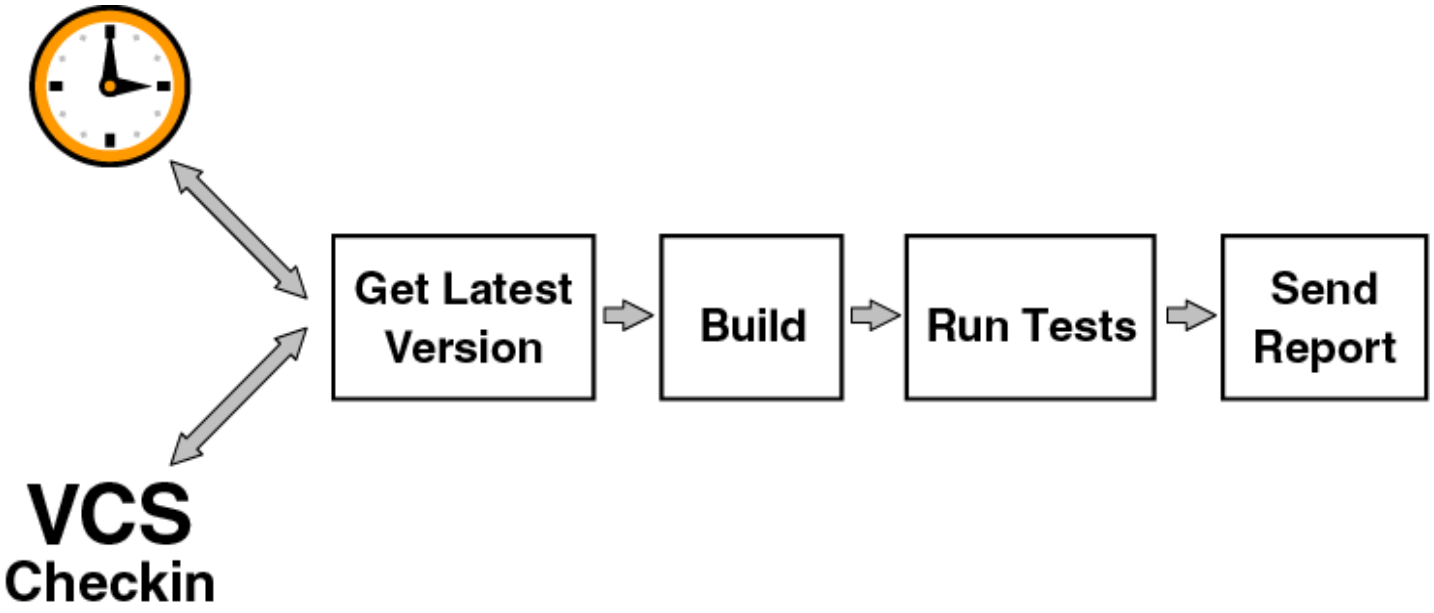
- Problems and Solutions

Server-Side Code Analysis

Collaboration

Extensibility

What Is Continuous Integration?

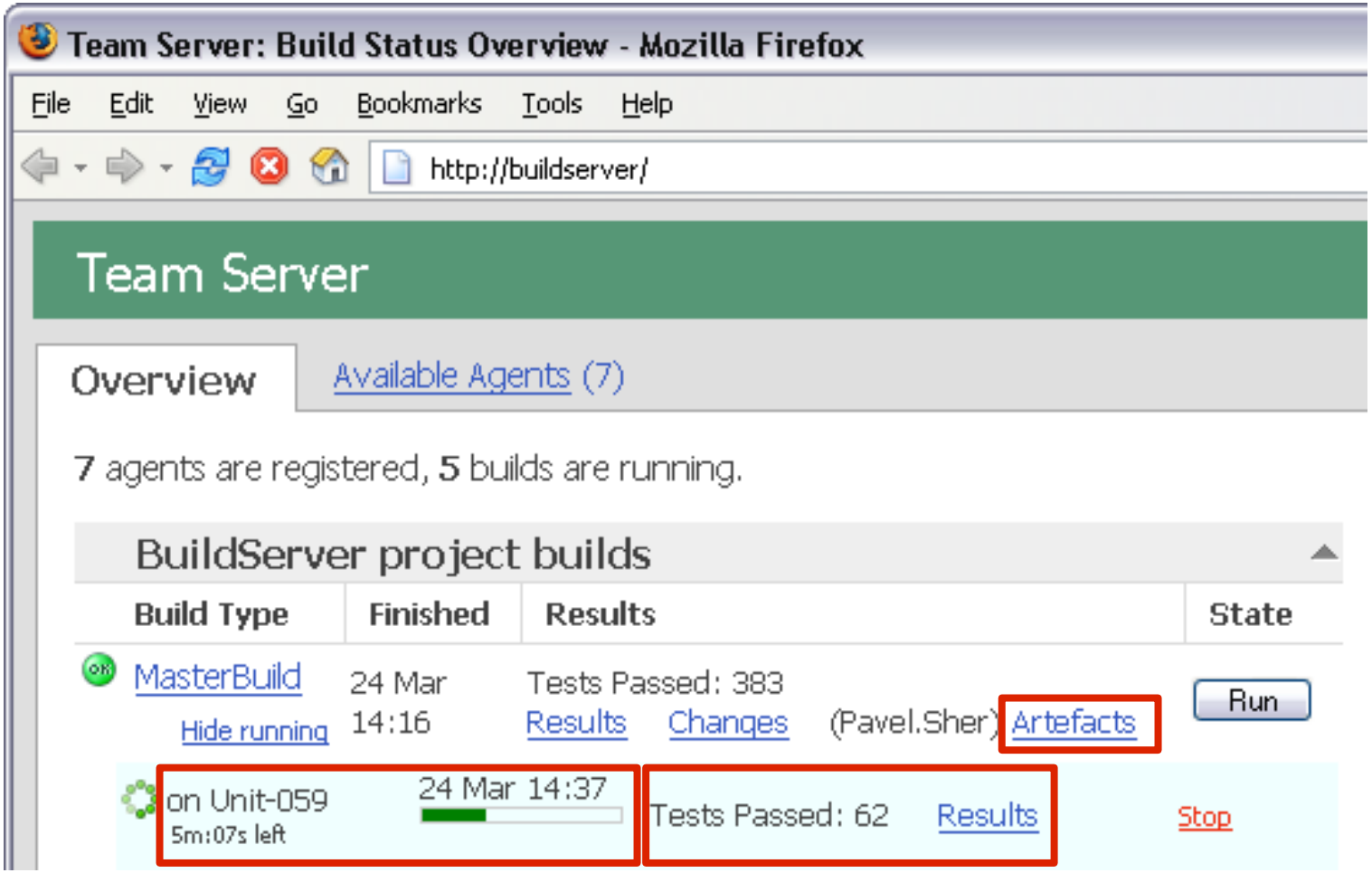


http://www.martinfowler.com/articles/continuousIntegration.html

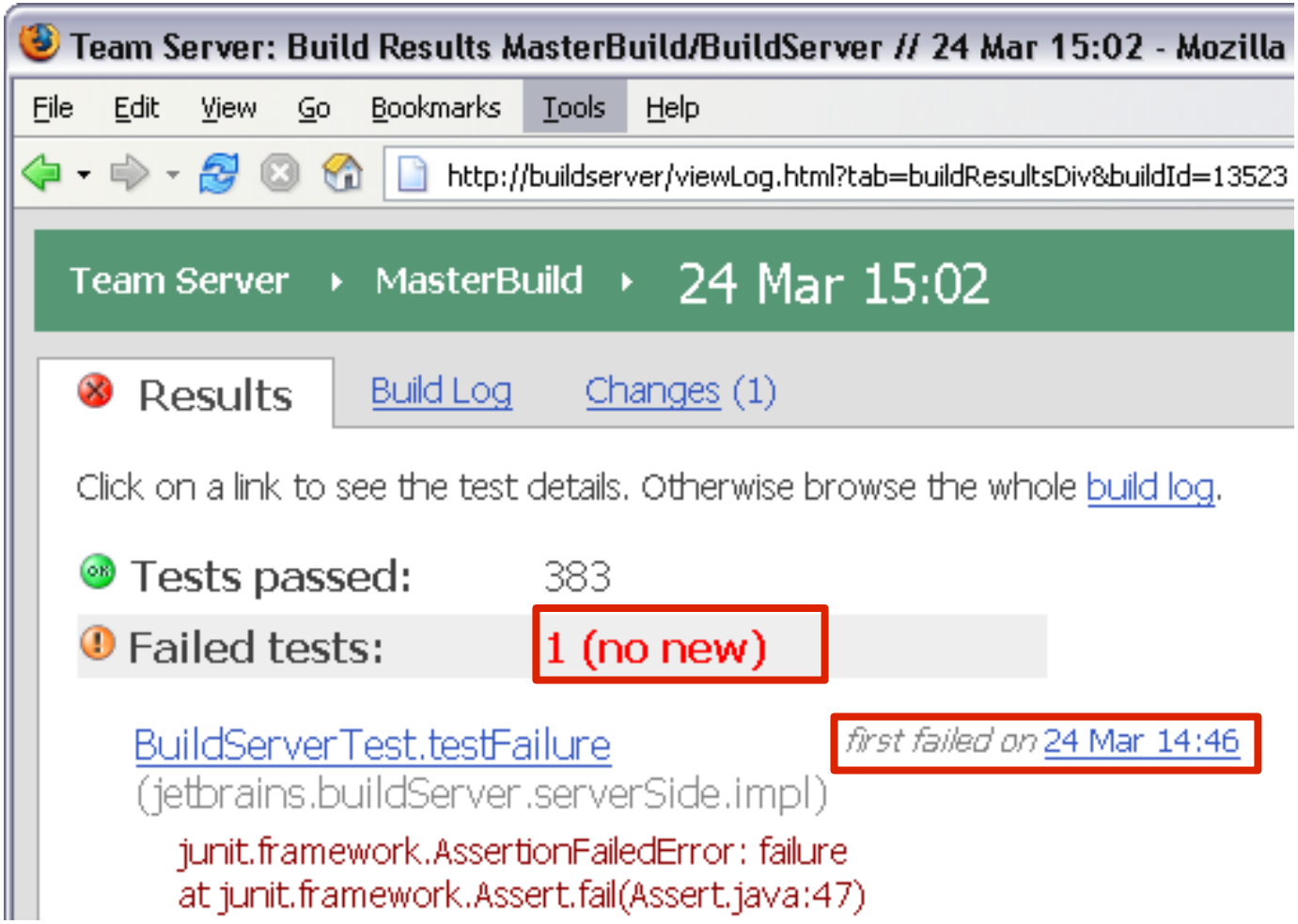
Continuous Integration Benefits

- Always aware of current status of the project
- Less time spent investigating integration bugs
- Less time wasted because of broken code in version control system

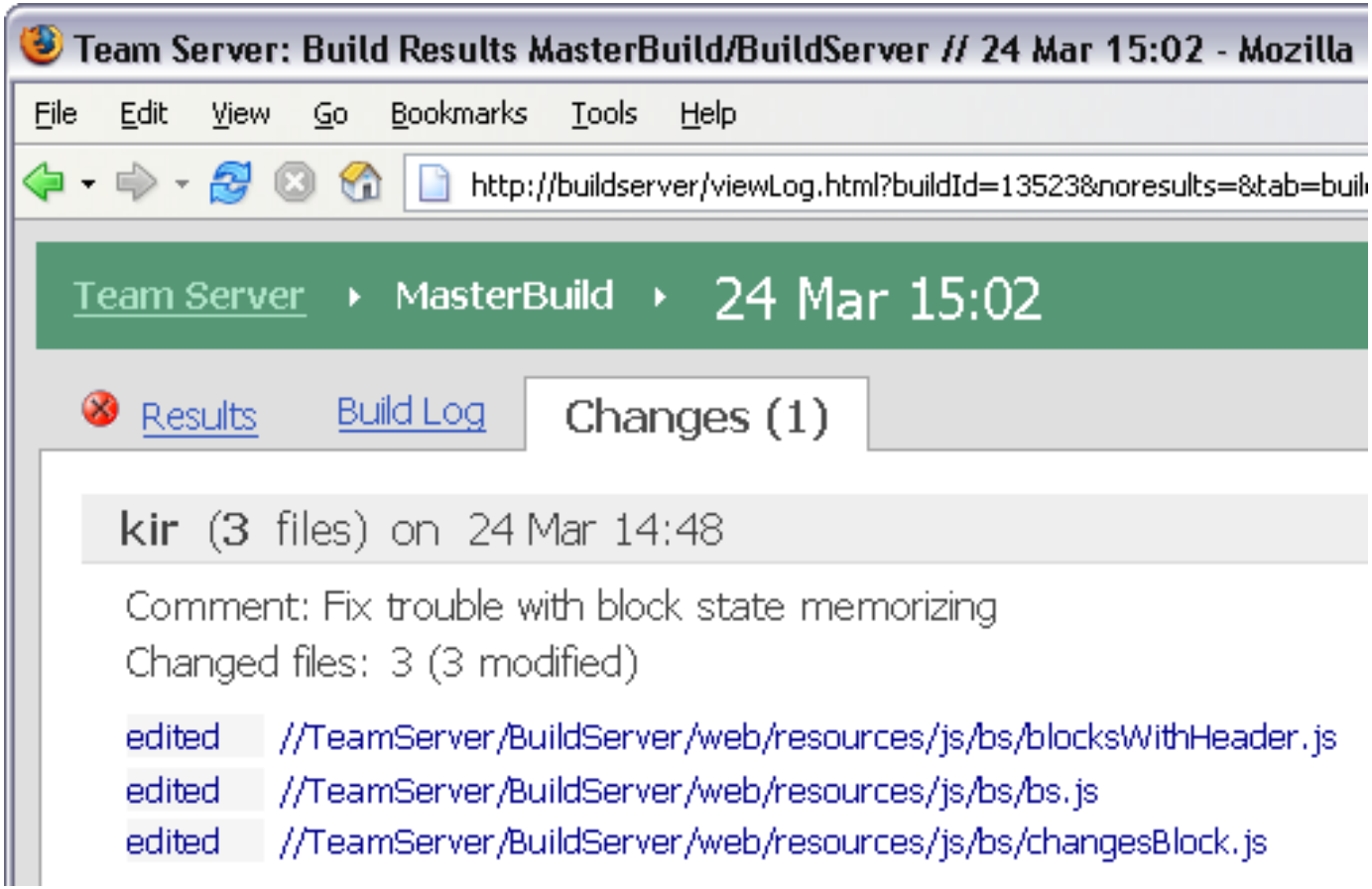
Team Server: First Look



Team Server: Build Results



Team Server: Changes in a Build



What We Build

- Java technology
- .NET
- Build systems for other platforms (through third-party plugins)
  - For example, ‘make’ or ‘scons’

Builds for Java Technology

- Industry-standard build tools supported:
  - Ant
  - Maven
- Easy set-up from IDE project
  - IntelliJ IDEA
  - NetBeans™
  - Eclipse

Builds for .NET

- NAnt
- MSBuild
- Visual Studio solutions

Easy to Get Started

- Unpack archive
- Specify version control server
  - CVS
  - Perforce
  - Subversion
- Specify build script location
- Run startup script and enjoy!
- Install more agents if required

Agenda

Background and Overview

A Better Continuous Integration

Problems and Solutions

Server-Side Code Analysis

Collaboration

Extensibility

Problem: Build Administration

"Anyone heard anything from our test machine?"

- Many builds to run
  - Several projects
  - Different branches
  - Multiple test suites
  - Continuous integration and release builds
- If one PC runs all builds sequentially, builds take too long
- If multiple PCs are used, administration becomes problematic

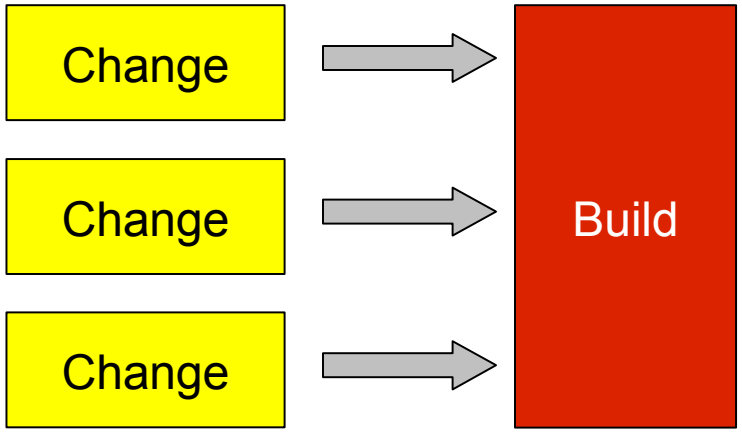
Solution: Build Grid

- Easy to add or remove agents
- Auto-update of agent software
- Can use idle developer PCs as build agents

Problem: Testing Granularity

"So which of those three changes broke the tests?"

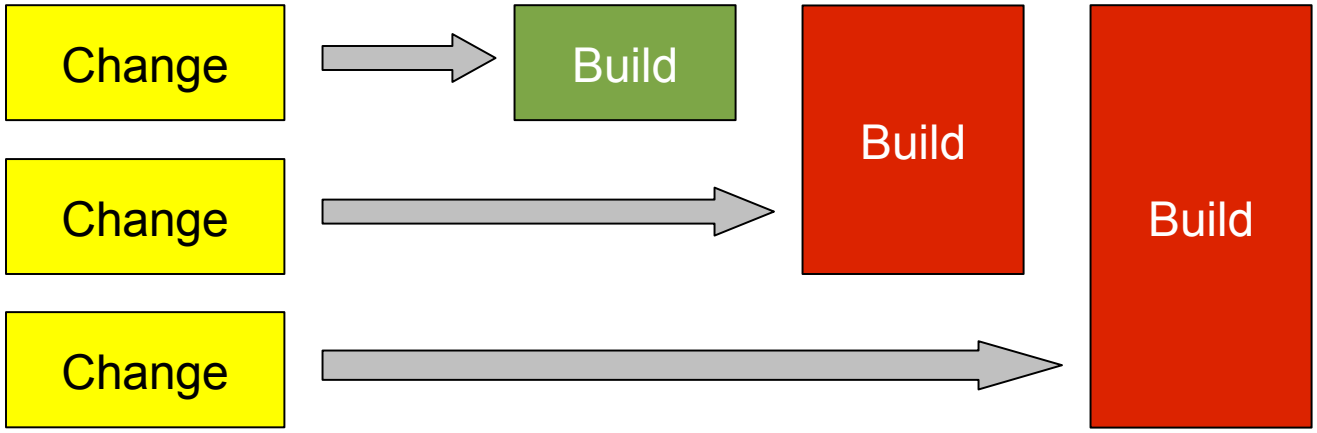
- Tests take a long time to run
- Checkins are more frequent than builds
- Several changes in build—which one broke it?



Solution: Distributed Testing

- Builds are distributed between agents
- If idle agents are available, build with new changes starts before previous is completed

Result: Faster and more specific feedback



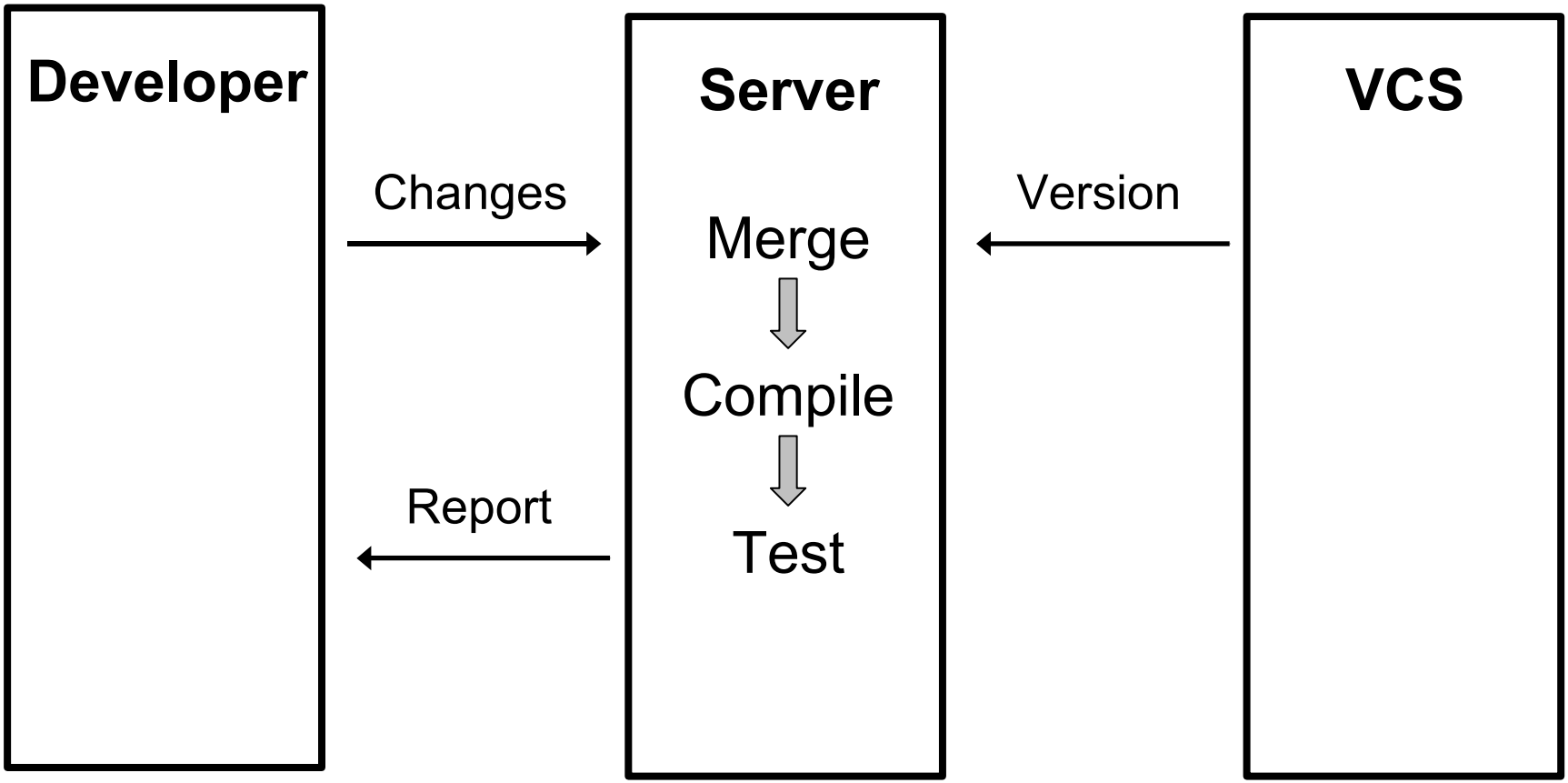
Problem: We Not Always Run Tests

"I can't run tests all day—I have work to do!"

- Tests take too long to run
- Complicated environment setup required
- Need to run tests under multiple operating systems

Result: Tests fail on build machine and break the build

Solution: Run Tests Remotely



Problem: 5 o'clock Check In

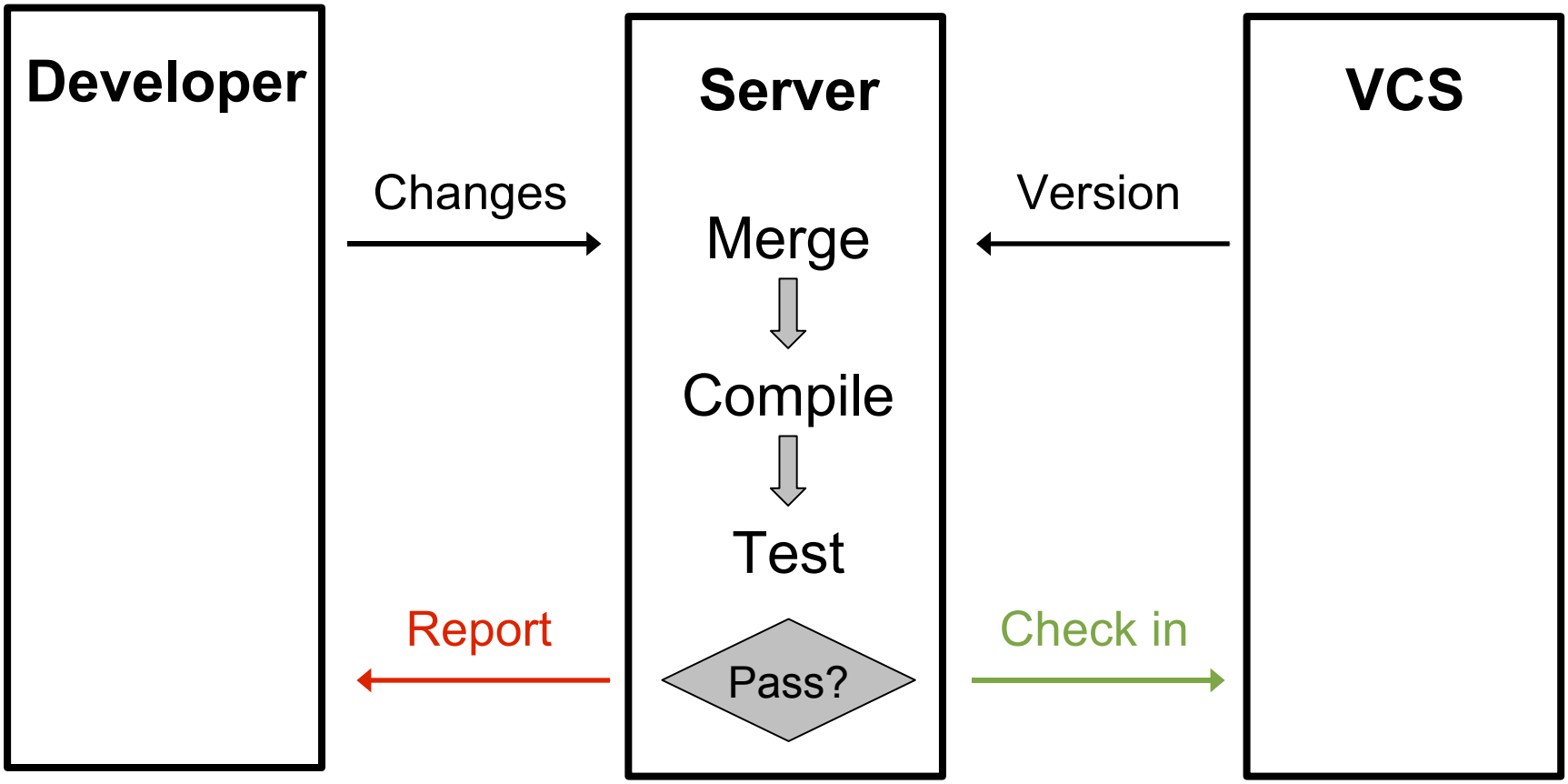
Check in, go home, build breaks

- Developer does not run tests
- …checks in broken code
- …leaves office

Result: Code in source control system is broken

- Particularly bad with distributed development (different time zones)

Solution: Delayed Check In



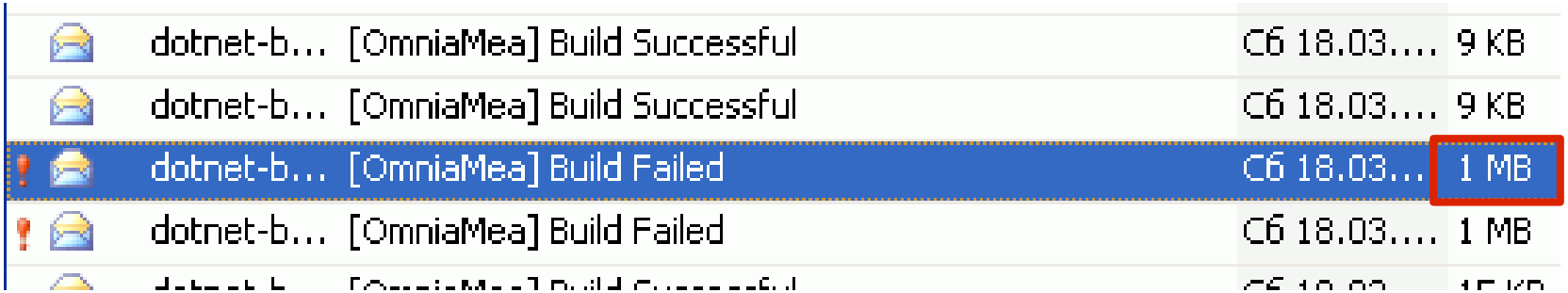
DEMO

Remote Run and Delayed Check In

Problem: Notification Spam

"Do you really expect me to read all of these?"

- Inbox is cluttered by e-mail notifications
- Hard to extract useful information from a failed build notification
- People stop reading notifications and don't notice when their tests fail



Solution: Clean Notifications

- Less frequent
  - Only failed builds
  - Only builds with my changes
  - Only if no one responsible for failure
- Earlier
  - As soon as failure is detected
- Simpler
  - By IM (Jabber) or by indicator in IDE status bar
  - Complete information clearly presented in Web UI

Problem: Nobody Fixes Build

"It's the other guy's fault!"

- Build starts to fail after multiple changes by different developers
- Everyone thinks that someone else is currently fixing it
- No fixing actually happens

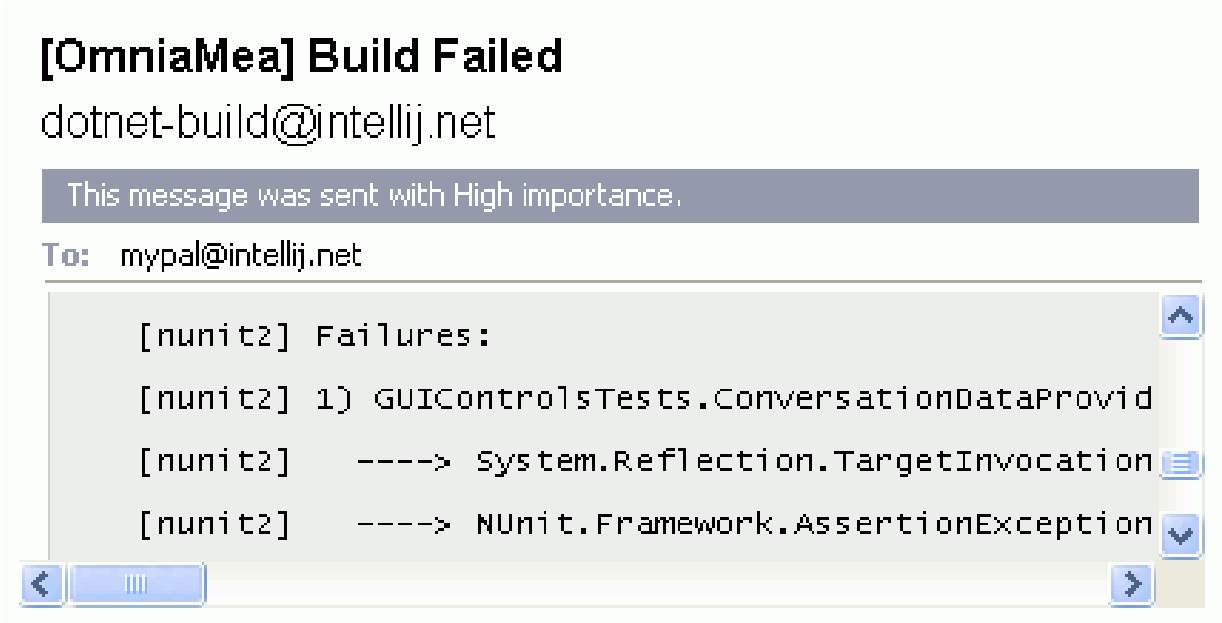
Solution: Take Responsibility

- Developer can take responsibility for build failure
- “Failure and no one responsible” is more severe than “failure with someone responsible”
- Responsibility cleared automatically on successful build

Problem: Locating Failures

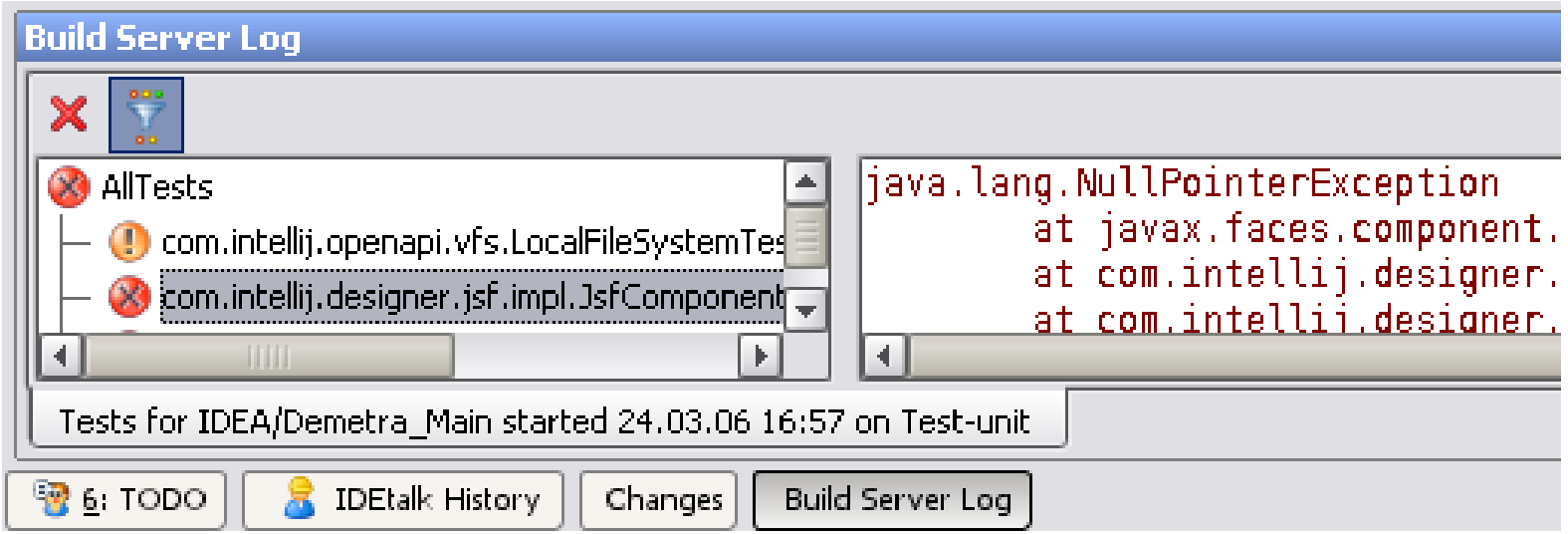
"I can see it failed—but why?"

- Reports in another application (browser or mail)
- Hard to find problem details in reports
- No live links to source code



Solution: IDE Integration

- View test results as if you've run the tests locally
  - Direct link to source code of failed test
  - Rerun failed test locally under debugger



Summary: What's Cool for Builds

- Distributed build environment allows for rapid and granular testing of changes
- Remote test run and delayed check ins help keep code in VCS clean
- Notifications are easy to understand and easy to investigate

Agenda

Background and Overview

A Better Continuous Integration

- Problems and Solutions

Server-Side Code Analysis

Collaboration

Extensibility

What We Have Now: Inspections

- Static code analysis in a single file
- Performed on-the-fly when editing code
- Results directly in editor
- Quick fixes available
- Used by almost every IntelliJ IDEA user

Other Analysis Tools

- Project-wide static code analysis
  - Unused code anywhere in the project
  - Method doesn't need to be public
  - …And many others
- Project-wide duplicate code search
- Code coverage
- More from third parties (code metrics, etc.)

… And Few Use Them!

- Breaks workflow
- Results often disconnected from code
- Too much time required for large projects

Solution: Server-Side Code Analysis

- Agent runs analysis and sends results to server
- IDE requests analysis results for current file from server
- Results are automatically adjusted if file was modified

Agenda

Background and Overview

A Better Continuous Integration

- Problems and Solutions

Server-Side Code Analysis

Collaboration

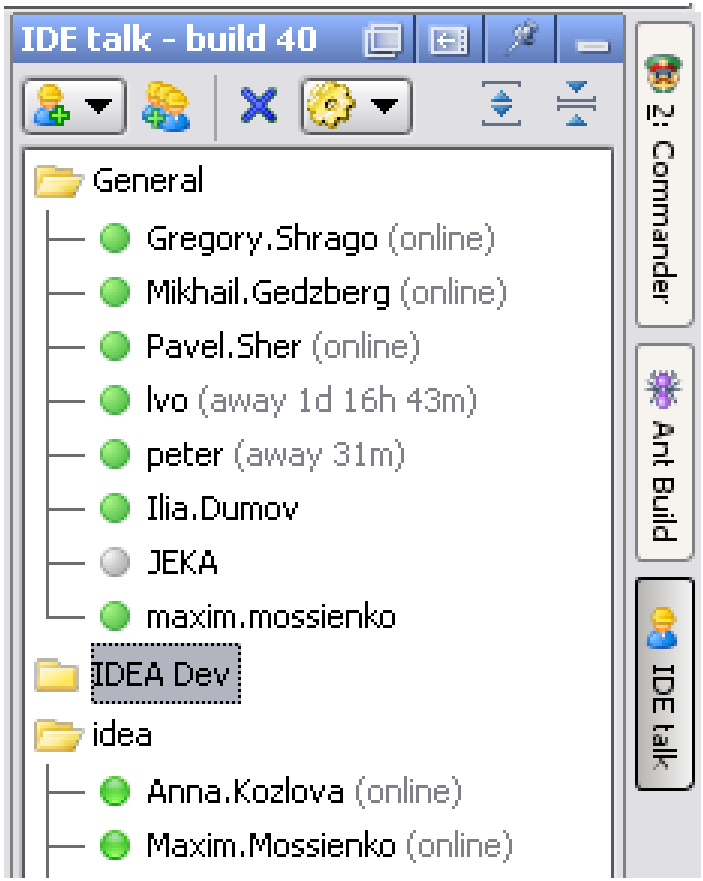
Extensibility

Collaboration: People Talk

- “Hey, it just crashed! I’ve emailed a stacktrace to you!”
- “Can you explain what the code at Foo.java line 230 is about?”
- “Is anyone else modifying Bar.java right now?”

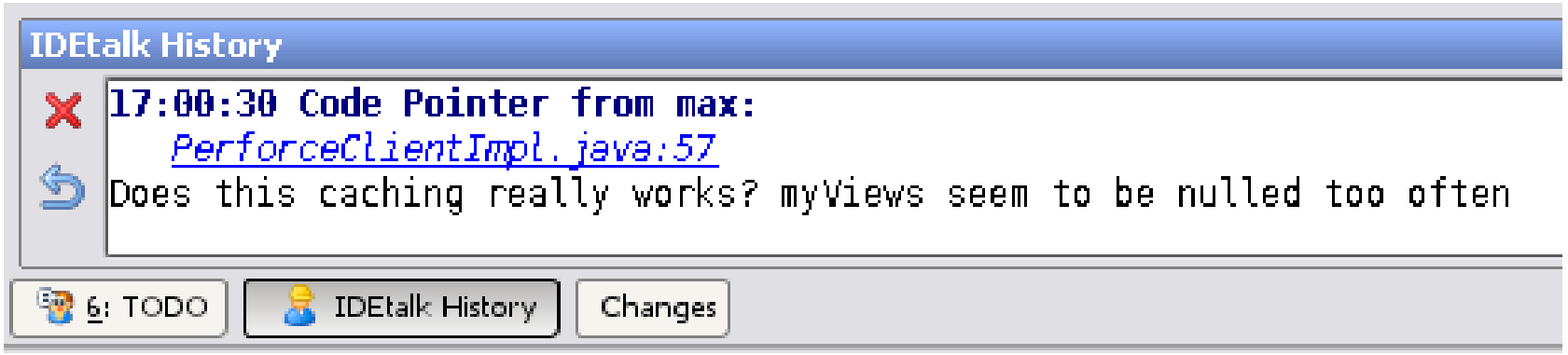
Solution: IDEs Talk

- Instant messaging integrated in IDE
- Project participants automatically added to contact list
- Industry-standard protocol (Jabber XMPP)



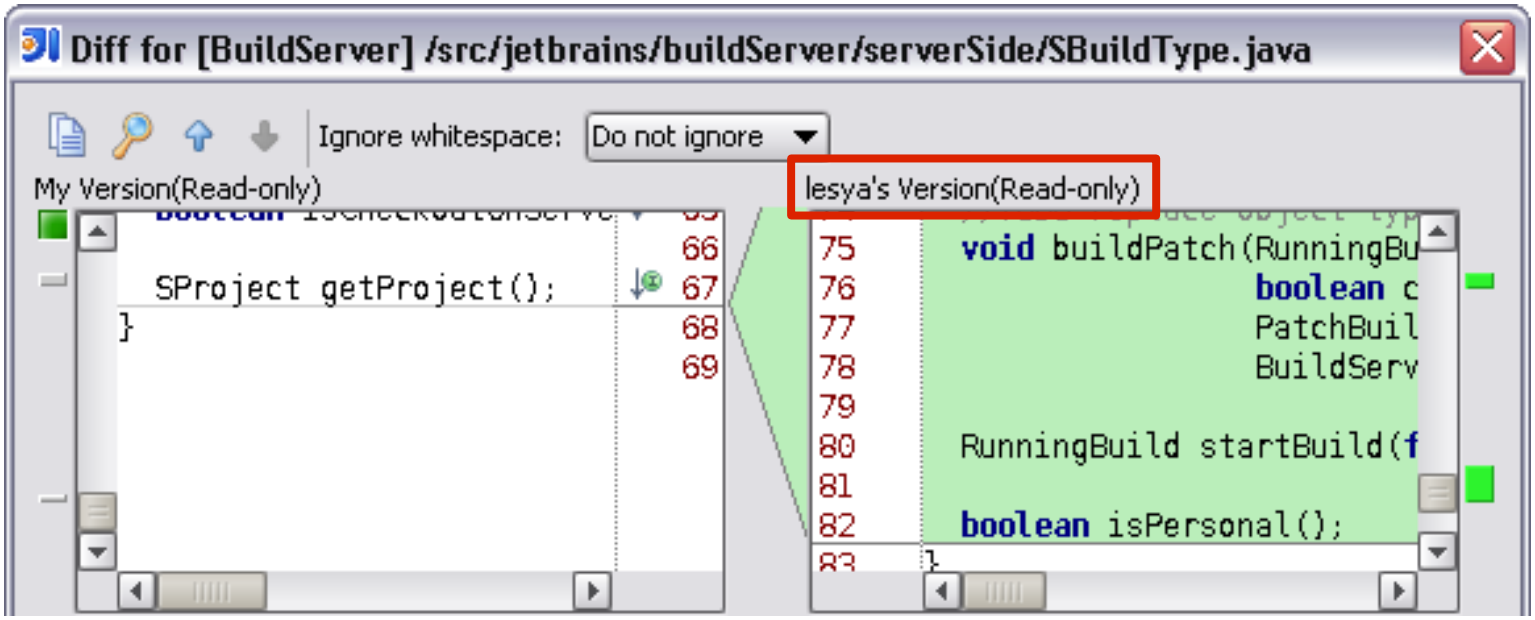
Collaboration: More Than Chat

- Send links to source code locations
- …Or entire navigatable stacktraces
- Auto-follow-me navigation (code review)



Collaboration: Diffs and Changes

- Diff with another developer's version of file
- Synchronize with changes made by other developer (no VCS involved)



Agenda

Background and Overview

A Better Continuous Integration

- Problems and Solutions

Server-Side Analysis

Collaboration

Extensibility

Core Extensibility Scenarios

- More notifiers (for example, ICQ or SMS)
- More build runners (for example, NMake or SCons)
- More version control systems (for example, ClearCase or Starteam)

Summary

- Team Server is an IDE independent solution for improving team productivity
- Solutions for a number of traditional continuous integration problems
- Results of powerful server-side code analysis are right at hand
- Collaboration features help team members communicate efficiently

For More Information

Early Access Preview program

- http://www.jetbrains.com/teamware/
- Public Team Server installation at http://teamserver.jetbrains.com/

Q&A

Dmitry Jemerov

IntelliJ IDEA: Integrated Team Environment

Dmitry Jemerov

Software Developer
JetBrains, Inc.
http://www.jetbrains.com/

TS-5033