TeamCity

TeamCity
A Professional Solution for Delivering Quality Software, on Time

Vaclav Pech
Senior Software Developer
JetBrains, Inc.

JetBRAINS
Develop with pleasure!

www.jetbrains.com

About Us

- Vaclav Pech
  - Professional software developer for 9 years
  - IntelliJ IDEA and TeamCity evangelist
- JetBrains
  - Makers of award winning productivity tools
    - IntelliJ IDEA, TeamCity, ReSharper, and more

TeamCity

About the presentation

- **Part 1:**
  - Continuous integration
- **Part 2:**
  - Solving problems from a developer’s perspective
- Questions and answers

TeamCity

TeamCity is a …

- Continuous integration tool
- Quality Control tool
- Tool for team cooperation

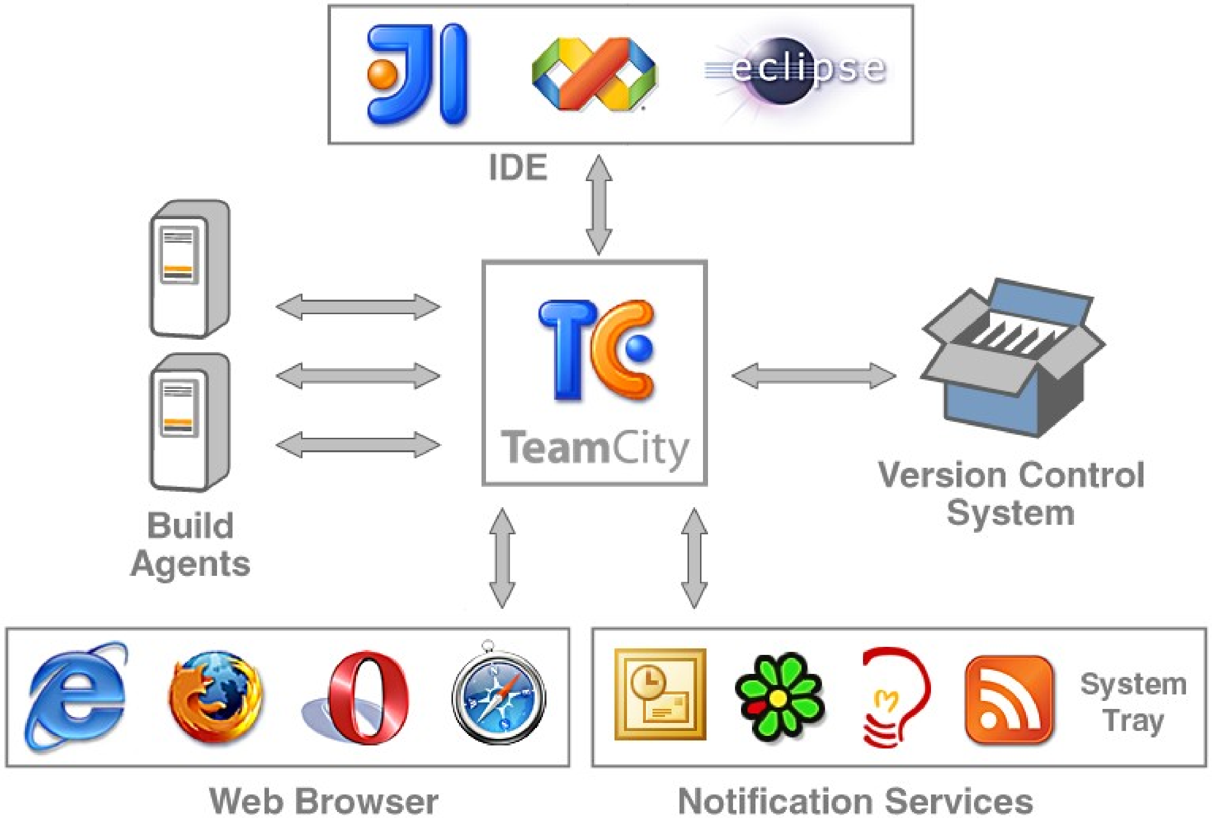
TeamCity

TeamCity is also …

- IDE-independent
- Platform-independent

Free Professional Edition available

Architecture



Part 1: Continuous integration

- Automatic build process
  - Triggered by
    - A timer
    - A VCS update
  - Builds the project
  - Runs tests
  - Generates artifacts
  - Notifies about the result

TeamCity

Continuous integration benefits

- You're always aware of the current project status
- Spend less time investigating integration bugs
- Spend less time fixing broken code

TeamCity

Part 2: TeamCity solving problems

- Build server administration
- Notification spam
- Nobody fixes the build
- Locating failures
- Integration
- Code quality
- Tests not run before commit

TeamCity

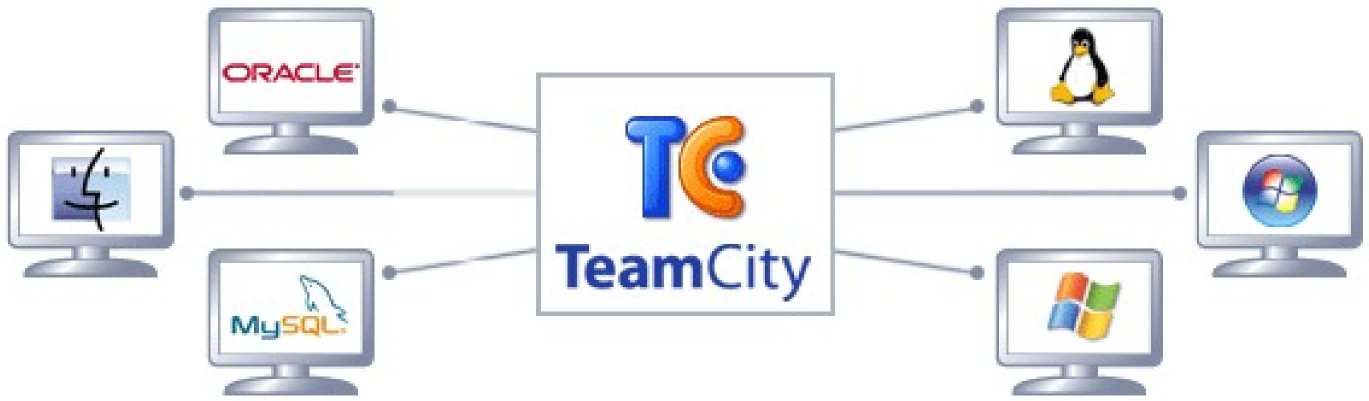
Problem:

- Build server administration

  - Many builds to run
    - Several projects
    - Different branches
    - Multiple test suites
    - Multiple platforms

  - Multiple machines to run the tasks

Solution: Distributed builds



TeamCity

Solution: Distributed builds

- Multi-platform testing
- Easy administration
  - Automatic update
- Optimized task distribution
  - Time estimates
  - Build queue
- Any computer can be used as an agent

Problem:

- Notification spam
  - Inbox is full of e-mail notifications
  - Hard to extract useful information from a failed build notification
  - People stop reading notifications

Solution: Clean notifications

- Less frequent
  - Only failed builds
  - Only builds with my changes
  - Failure after success, success after failure
- Earlier
  - As soon as a failure is detected
- Simpler
  - IM, IDE status bar
  - Complete details in IDE or through Web UI

Problem:

- Nobody fixes the build
  - Build starts failing after multiple changes by different developers
  - Everyone thinks that someone else is currently fixing it
  - No fixing actually happens

TeamCity

Solution: Take responsibility

- Easy detection of code changes included in the failed build

- Developer can take responsibility
- Different severity indicated for failures without responsibility set

TeamCity

Problem:

- Locating failures
  - Reports in another app
  - Hard to find the problem details in reports
  - No links to source code

TeamCity

Solution: IDE integration

- Show test results just as if they ran locally
- Direct links to the source code
- Hanging build detection and notification
    - Thread dump
- Intuitive UI
    - Optimized for daily use
    - Web
    - Integrated into IDEs

TeamCity

Problem:

- Integration
  - We cannot build with our build tool
  - We have problems with the VCS
  - Our test reports cannot be displayed

TeamCity

Solution: Integration

- Different VCSs – CVS, SVN, ClearCase, …
- Runners
    - Ant
    - Maven 2
    - IntelliJ IDEA projects
    - Command line
- Notification
    - Email, RSS, Jabber, IDE, windows Tray Notifier
- IDEs
    - Eclipse, IntelliJ IDEA, Visual Studio
- Test frameworks
    - JUnit, TestNG, EMMA, Cobertura (planned)

TeamCity

Solution: Extensibility

- Plugins
  - Custom statistics report tools
  - External status widgets
  - Notifiers
  - Runners
  - Triggers
  - User authentication
  - VCSs

Problem:

- Developers rarely use tools for code quality
  - Breaks workflow
  - Results often disconnected from code
  - Too much time required for large projects

Solution: Server checks for quality

- Inspections
  - 700+ rules for Java, JavaScript, HTML, XML, …
    - unused and unreachable code,
    - declaration redundancies,
    - performance issues
    - dependency rules
- Code coverage
- Code duplicates

Problem:

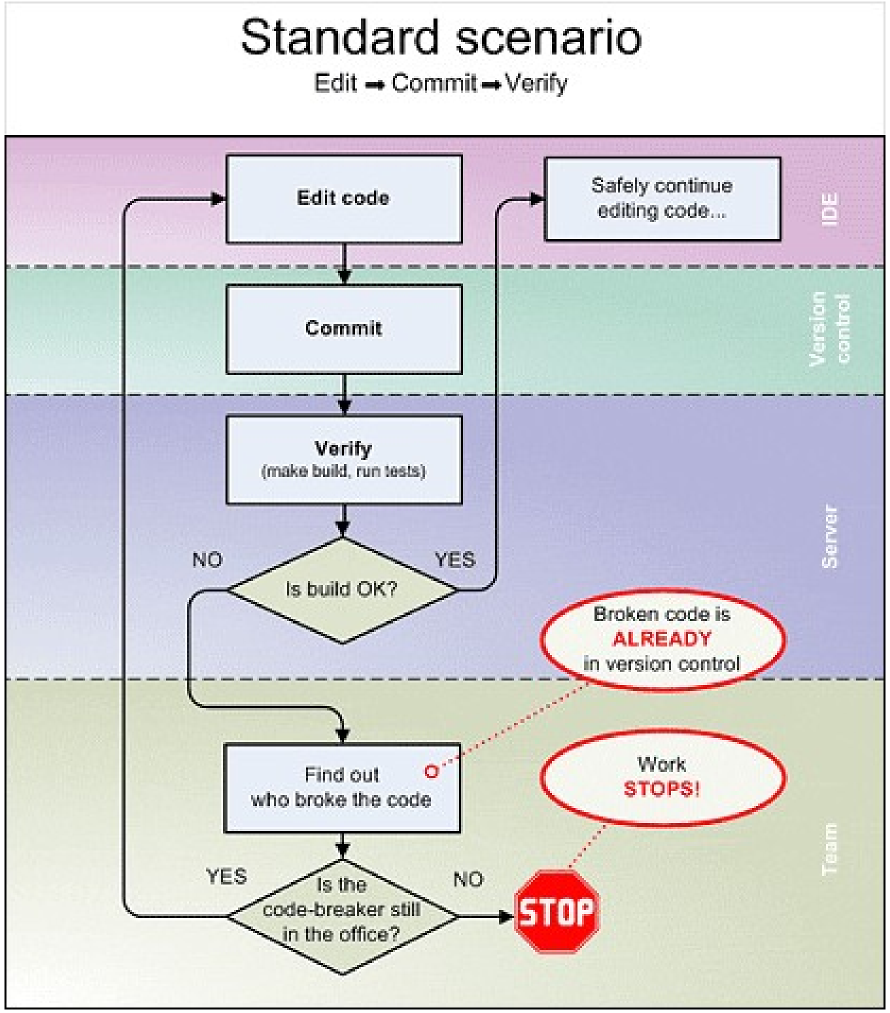
- We don’t always run the tests before commit
  - They take too long to run
  - Complicated environment setup
  - Need to run tests in different environments
- – 5 o’clock checkin
- Result
  - Broken code in VCS
  - Others cannot work
  - Particularly bad with distributed teams

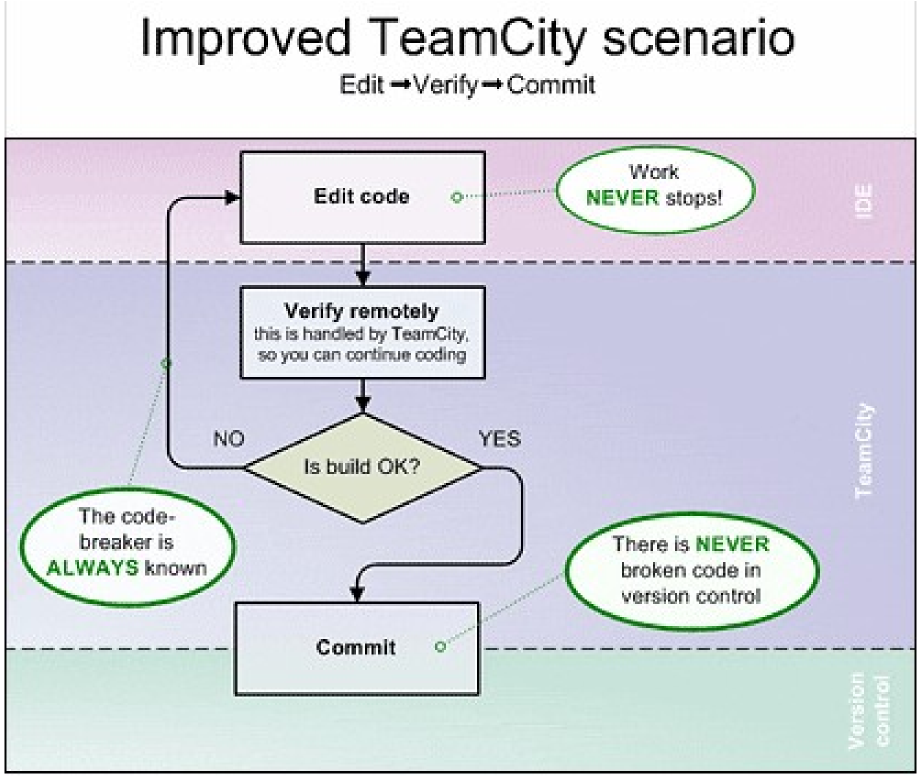
TeamCity

Solution: Pre-tested commit

- Pre-tested commit
  - Let TeamCity run the tests before your changes hit VCS
    - Your machine is available for further coding
    - No more broken builds
  - Run personal builds at any point in time to ensure you are still on track

TeamCity


Standard scenario
Edit → Commit → Verify
Edit code
Safely continue editing code...
IDE
Commit
Version control
Verify (make build, run tests)
Server
NO
Is build OK?
YES
Broken code is ALREADY in version control
Find out who broke the code
Work STOPS!
Team
YES
Is the code-breaker still in the office?
NO
STOP


Improved TeamCity scenario
Edit → Verify → Commit
Edit code
Work NEVER stops!
IDE
Verify remotely this is handled by TeamCity, so you can continue coding
NO
Is build OK?
YES
TeamCity
The code-breaker is ALWAYS known
There is NEVER broken code in version control
Commit
Version control

TeamCity

Summary

- Team-focused productivity
- IDE independent
- Eliminates a number of traditional continuous integration problems
- Free professional edition

  – **Contact me: vaclav@jetbrains.com**

Questions