Continuous Improvement via Continuous Integration

Anurup Joseph
Elegan Consulting

About Anurup

- - coding professionally since 1994
- - working with Java since 1996
- - different industries/sectors/geographies
- - loves to explore
- - enjoys fostering Agile development and Kaizen

About You

- - Agile? Waterfall? Other? Chaos?
- - Continuous Integration? Continuous Delivery?
- - Continuous Improvement process?

Kaizen

- “change for the better”
- Continuous Improvement
- inspired TPS to LSD

Continuous Improvement Importance

- - reduce tech debt
- - learn of emergent discoveries/vulnerabilities
- - identify and fix new issues
- - team education
- - CI teams yield competitive advantages

Continuous Integration Importance

- - non-trivial software consists of disparate components
- - components must be integrated
- - integration points tend to cause issues
- - Continuous Integration discovers issues early
- - early discovery = lower cost in time and money

But How?

Manual Review

- - human inspection of all existing and new code
- - requires varied expertise
- - slow, boring, error-prone
- - difficult with distributed teams
- - human time gets more expensive

Manual Tools

- - static analysis software (CLI or IDE)
- - regularly updated with new inspections
- - never gets bored or sloppy
- - must remember to run them — upon every change by anyone
- - tends to be episodic
- - no tool is as good as expert human inspectors
- - machine time gets cheaper

Automatic Tools

- - integrate static analysis software with Continuous Integration
- - each check-in/build results in full inspection
- - team informed of new issues
- - metrics tracked over time: “If you can’t measure it, you can’t manage it.”
- - next best thing to human experts always reviewing everything
- - machine time gets cheaper

TMTOWTDI

- - many tools available
- - will present those that worked for my teams
- - will not go in-depth into configuration specifics

Some Tools to Try

- - Maven & Jenkins
- - FindBugs & PMD
- - CPD
- - Cobertura
- - Checkstyle
- - Open Tasks

Workflow Part 1

- - checkin triggers Jenkins to run Maven build
- - Maven build runs tests
- - Maven build runs static analyzer via Maven plugins
- - static analyzers generate prioritized reports
- - Jenkins plugins present reports graphically

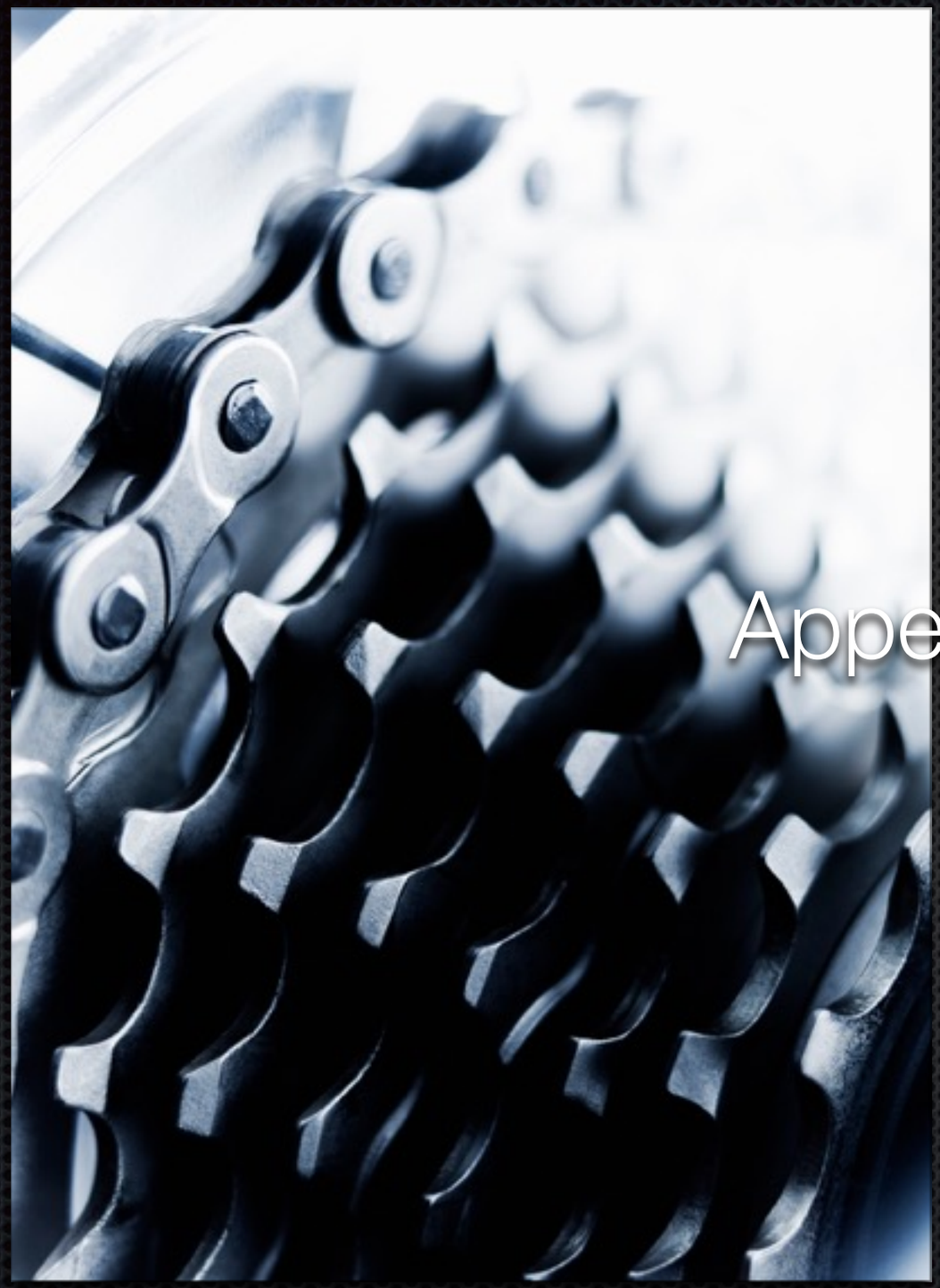
Workflow Part 2

- - team monitors analysis reports and emails
- - new issues fixed in Iteration
- - extant issues result in Stories/Defects in Backlog
- - team continually pulls from Backlog in priority order
- - leads/management monitors quality metrics
- - Retrospectives result in learnings
- - becoming a learning/improving organization

Fun with System.exit()

Continuous Integration
yields
Continuous Improvement

Appendix



TPS Principles

- - Continuous Improvement
- - Respect for People
- - Develop Long-term Vision (strategy)
- - Focus on Short-term Process (tactics)
- - Grow People
- - Create Learning & Improving Organization

LSD Principles

- - eliminate waste
- - Continuous Improvement to enhance learning
- - decide as late as possible
- - deliver early and iteratively
- - empower the team
- - Continuous Integration builds integrity
- - see the whole: “Think Big, Act Small, Fail Fast, Learn Rapidly”

Agile Principles

- - Customer satisfaction by early and continuous delivery of valuable software
- - Welcome changing requirements, even in late development
- - Working software is delivered frequently (weeks rather than months)
- - Close, daily cooperation between business people and developers
- - Projects are built around motivated individuals, who should be trusted
- - Face-to-face conversation is the best form of communication (co-location)
- - Working software is the principal measure of progress
- - Sustainable development, able to maintain a constant pace
- - Continuous attention to technical excellence and good design
- - Simplicity—the art of maximizing the amount of work not done—is essential
- - Best architectures, requirements, and designs emerge from self-organizing teams
- - Regularly, the team reflects on how to become more effective, and adjusts accordingly