Continuous Integration

Bill Buie Principal Consultant
Phone: 732.245.0984
email: bill.buie@cmi.com

- Who is CMI?
- What is Continuous Integration (CI)?
- How CI is Getting Done
- How Industry Trends Can Be Improved Upon
- Implementation Considerations

CMI Introduction

- 20 years focus Enterprise Software Configuration Mgmt.
- CMI focuses on delivering services to all elements of the Software Development Life Cycle (SDLC)
- CMI Global Company | over 350 employees
- CMI is a leading provider of Business Process Outsourcing, IT Consulting and IT Staffing Services
- Serving Fortune 500 business and leading mid-market companies across all industries

New Demands of Business Software

What's driving the need for "Continuous Integration"?

RADICALLY REDUCED TIME TO MARKET

UNASSAILABLE PRODUCT QUALITY

LOWER TOTAL IT COST

"THE ULTIMATE CUSTOMER IS THE BUSINESS CONSUMER"

What is Continuous Integration?

What is Continuous Integration?

- Basic Concept:
  - Integrate as frequently as possible
  - Avoid “death march” integrations
- Reason for Existence: Find & fix defects as early as possible
- There is a presumption that frequent commits to the code repository & frequent automated builds out of the code repository are the best way to do that

  CMI doesn’t necessarily agree, but more on that later

What is Continuous Integration?

- Famous Quote:

"Nightly builds mean that bugs lie undetected for *a whole day*[1] before anyone discovers them."

Martin Fowler

[1] Emphasis added

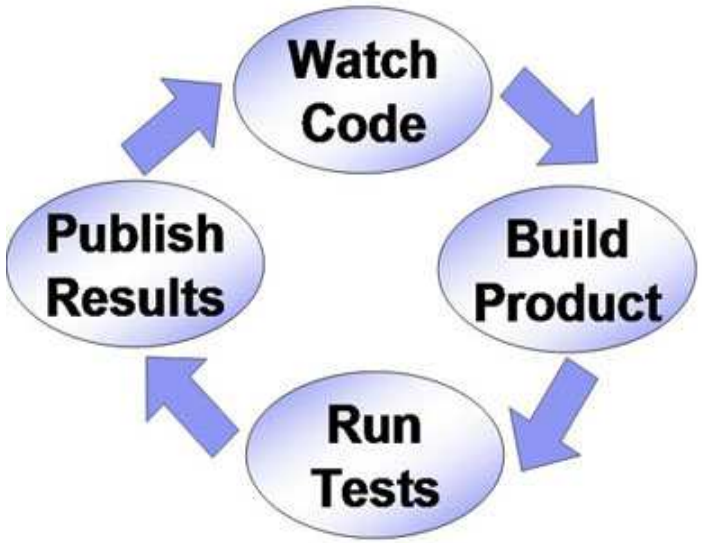
How CI is Getting Done

Let's look at some enabling techniques…

- **Granular and explicit "chunking out" of Tasks by PMs and/or Agile coaches**
  - A "Task" is something that needs doing in order to implement a User Story
  - A Task can be implemented in less than a day
  - A single developer may complete several tasks in a day
  - The work completing a Task is committed to source control immediately

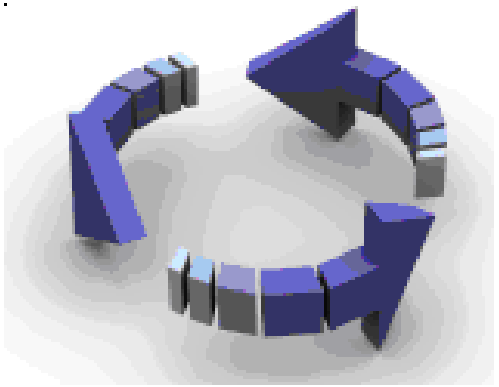
- **Source code control *(!!!)***

How CI is Getting Done cont.



- **Automated, *FAST* builds**
  - The industry expectation is a 10-minute build
  - Implies modular architecture
  - Fairly easy to achieve for well-designed systems, relatively small systems, and/or systems written compatibly with certain IDE expectations
  - Complex systems may need to be rearchitected *(which arguably ought to happen anyway)* before they can capitalize on this trend
- For large applications, this means reusable build components, so you're only rebuilding the component you changed

How CI is Getting Done



- Investment in Build and Test servers
  - The cost of metal is offset by the shortening of the development cycle
  - Virtualization
- Instant, automated notification
  - Email
  - Wiki pages
  - Clever Color Codes

How CI is Getting Done

- Prioritizing Bug Fixes
  - If the integration build breaks, it's **ALL HANDS ON DECK**
  - Team commitment: you can't go home until the integration build passes after your last commit to the source code repository

- Automated Deployments
  - Facilitated by some Integrated Development Environments (IDEs)
  - First and most critically for the integration test host
  - But then, that can be the basis for automating Production deployments…

How CI is Getting Done

- Code Signing & Verification

- Automated Testing
  - Software Test Tools
  - Parallel Test servers
  - Tiered testing
    - Longer tests go on dedicated servers for scheduled builds
    - Higher tier test failures are opportunities to enhance lower tier tests
  - *Note the very significant paradigm shift in the role of a test engineer!*

How CI is Getting Done

- **Software Tool Support**
  - Many tools in the market supporting this trend
    - Many IDEs are auto-generating the automation to build & deploy code
    - Build avoidance software
    - Build automation/facilitation software
  - Build Avoidance
    - Objective: cache component build product versions, and reuse them instead of rebuilding them
    - Clearmake used to do that…
    - Maven/Nexus
    - Ivy/Artifactory

How CI is Getting Done

- Software Tool Support *(cont'd)*
  - Build automation
    - IBM BuildForge
    - Cruise Control
    - Hudson/Jenkins
  - Test Tools
    - NUnit
    - Fitnesse
    - Quality Center
    - etc.

- Name three techniques used to implement Continuous Integration today

- Which trend do you believe will be easiest to implement in your organization?

- Which trend do you believe will bring your organization the most Bang for the Buck?

Improving Upon Industry Trends

- **“Emergent” software versus Governance**
  - Dynamic tension between popular development methodologies based on Trust and the need for “Outsider” interests to be protected
    - Trust is a fine perspective for members of the Agile team, but is it a good perspective for stakeholders outside the team?
    - There’s a reason some industries are regulated…
  - If we acknowledge the need for oversight then we can focus on making the oversight efficient & effective
    - Controls don’t have to kill productivity
    - Review expectations at start of Agile projects
    - Independent approver requires evidence

Improving Upon Industry Trends cont.

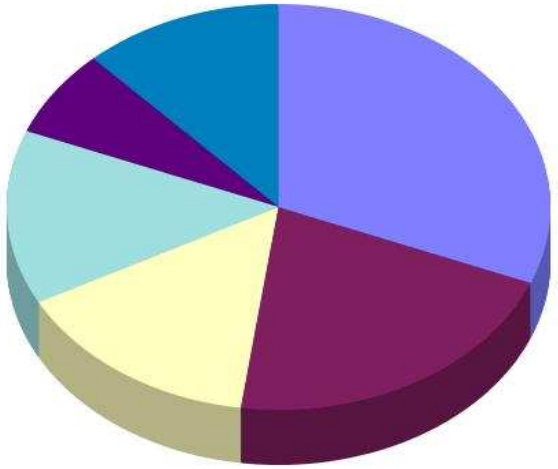
- Early Detection versus Prevention
  - Early detection impacts the whole team
  - Current practice very sensitive to team discipline/maturity

- Managed Continuous Integration offers the best of both worlds

Implementation Considerations

- Allocation of host platforms
  - “ownership” versus “allocation”
  - Dedicated build/test platforms
- Tool purchases/downloads
  - Many organizations maintain an approved applications list
  - Since these tools are for new functionality, there won’t be equivalents to remove from the list
- Administration policies w/r/t Maven-type behavior
- Access policies regarding consoles for managing automated, multi-platform builds and deployments

Implementation Considerations cont.



- Allocation of Organization Responsibilities
  - Test Engineering ➔ Development
  - Where do the SCM specialists go?

- Which two ideas are most critical to a regulated institution…?
  - Trust
  - Speed
  - Governance

- What are the pros and cons between early detection & defect prevention?

In summary

- Continuous Integration **radically shortens** the development lifecycle by virtually eliminating Integration Testing from the schedule
- Other companies are **“Getting to Done”** more reliably, with higher quality software and net reduced costs by adopting CI
- A certain investment in tools, training, and infrastructure is required to obtain these benefits
- **CMI Can help with CI pilots**

Questions?

Thank you!

Please contact me with any questions

Bill Buie

732.245.0984

bill.buie@cmi.com