Maturing agile teams and driving quality through architecture principles

Amine Chigani & Yun Freund
GE Software
San Ramon, CA

Imagination at work
Introducing the GE Software Center

Igniting the next industrial revolution by connecting minds & machines

$1B investment over past 3 years

- Launched in 2011 in Silicon Valley location
- Aggressive strategy for talent acquisition
- Founding member of Industrial Internet Consortium
- Introduce Predix™ platform to the world in 2015

Imagination at work
**Many Industries. Many Platforms.**

<table>
<thead>
<tr>
<th>Industry</th>
<th>Platforms</th>
<th>Optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>GE Aviation</td>
<td>Platform 1</td>
<td>Asset optimization</td>
</tr>
<tr>
<td></td>
<td>Platform 2</td>
<td>Operations optimization</td>
</tr>
<tr>
<td>GE Energy Management</td>
<td>Platform a</td>
<td>Asset optimization</td>
</tr>
<tr>
<td></td>
<td>Platform b</td>
<td>Operations optimization</td>
</tr>
<tr>
<td>GE Healthcare</td>
<td>Platform i</td>
<td>Asset optimization</td>
</tr>
<tr>
<td></td>
<td>Platform ii</td>
<td>Operations optimization</td>
</tr>
<tr>
<td>GE Oil &amp; Gas</td>
<td>Platform x</td>
<td>Asset optimization</td>
</tr>
<tr>
<td></td>
<td>Platform y</td>
<td>Operations optimization</td>
</tr>
<tr>
<td>GE Power &amp; Water</td>
<td>Platform A</td>
<td>Asset optimization</td>
</tr>
<tr>
<td></td>
<td>Platform B</td>
<td>Operations optimization</td>
</tr>
<tr>
<td>GE Transportation</td>
<td>Platform ...</td>
<td>...</td>
</tr>
</tbody>
</table>

**State of practice pre 2011...**
Many Industries. One Platform.

GE Aviation
GE Energy Management
GE Healthcare
GE Oil & Gas
GE Power & Water
GE Transportation

Industrial Internet Apps
Solutions for industrial asset and operation optimization

Predix™ Platform
Common scalable cloud architecture for machine, network, server and UX

Today’s direction...
Overview

• Context setting
• Architecture pillars for quality & maturity
  • Adopting an architecture-driven agile process
  • Defining with stakeholders key quality attributes
  • Building robust dev-test-deploy infrastructure
  • Aligning architecture with platform technology choices
  • Maintaining consistent code-level standards
  • Introducing new skillset to the teams
• Parting thoughts and conclusions
**Context Setting**

- **Teams**
  - Multiple scrum teams working on various projects from data science NTI to legacy integration

- **Customer**
  - GE Transportation; a 100 year market leader in Class 1 railroads; Several $B in backlog

- **Development methodology**
  - Agile; various flavors of scrum; six sigma traces; rigor-less to no architecture

---

**Objective**

To standardize quality practices across teams, projects, and product releases to deliver consistent quality within planned time and budget.
“Quality and architecture are two sides of the same coin.”

Regardless of development methodology (agile or otherwise), software quality is assessed from two complimentary perspectives:

• **Functional**: achieved through unit, functional, and user acceptance tests
• **Structural**: achieved through evaluating key quality attributes such as security, performance, maintainability, and user experience
Adopted an architecture-driven agile process...

Established clear Definition of Ready (DoR) and Definition of Done (DoD) for each phase
Defined with stakeholders key quality attributes...

Structural quality is concerned with cross-cutting concerns represented as key quality attributes:

- **Security**: Static analysis, dynamic analysis, defense-in-depth, and pen-testing
- **Compliance**: Sarbanes-Oxley
- **Performance**: Load testing
- **Functionality**: 100% implementation meeting acceptance & quality criteria
- **Usability**: User-centric design + consistent UI standards
- **Maintainability**: Coding standards + modular architecture + documentation
- **Testability**: Unit & functional testing + Automation
- **Integrate-ability**: 1.0/2.0 Integration + Continuous Integration (CI)
- **Deployability**: Deployment architecture + Continuous Deployment (CD)
<table>
<thead>
<tr>
<th>Quality Attribute</th>
<th>Supporting Indicator</th>
<th>Measurement</th>
<th>Target</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security / Compliance</td>
<td>Static analysis using Checkmarx &amp; JSHint</td>
<td>Identified security violations</td>
<td>0 Blocker and 0 Critical violations</td>
<td></td>
</tr>
<tr>
<td>GrayBox Testing</td>
<td></td>
<td>Identified security violations</td>
<td>Pass Requirements</td>
<td></td>
</tr>
<tr>
<td>Defense in Depth</td>
<td></td>
<td>Identified security violations</td>
<td>Pass Requirements</td>
<td></td>
</tr>
<tr>
<td>SOX compliance</td>
<td></td>
<td>Checklist of requirements</td>
<td>Pass Requirements</td>
<td></td>
</tr>
<tr>
<td>Performance</td>
<td>Performance testing - Release</td>
<td>Establish test cases and SLAs</td>
<td>&lt;= ? sec / step</td>
<td></td>
</tr>
<tr>
<td>Functionality</td>
<td>Functional coverage</td>
<td>Are all functionality implemented?</td>
<td>&gt;95%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Functionality acceptance</td>
<td>Do all functionality meet acceptance criteria?</td>
<td>&gt;95%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Defects</td>
<td># of open defects associated with functionality</td>
<td>No S1 or S2</td>
<td></td>
</tr>
<tr>
<td>Usability</td>
<td>UX Design standards followed</td>
<td>UX design review</td>
<td>Code to UX Design artifacts</td>
<td></td>
</tr>
<tr>
<td>Maintainability</td>
<td>Tech stack consistency</td>
<td>Architecture review</td>
<td>Code to architecture</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Code duplication</td>
<td>Sonar</td>
<td>&lt;= 10%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Complexity</td>
<td>Sonar</td>
<td>&lt;= 10%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Coding style</td>
<td>Coding style document</td>
<td>Code to standards</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Release documentation</td>
<td>Release document to ship with product</td>
<td>Complete Document</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Documentation</td>
<td>% in sonar</td>
<td>20%</td>
<td></td>
</tr>
<tr>
<td>Testability</td>
<td>Test coverage</td>
<td>% unit test coverage (back end)</td>
<td>80%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Test coverage</td>
<td>% unit test coverage (front end)</td>
<td>80%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Test coverage</td>
<td>% functional coverage</td>
<td>95%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Automation (Regression)</td>
<td>Whether testing is fully automated</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Integratability</td>
<td>Integration with legacy application</td>
<td>Ensure data integration is done</td>
<td>Code to architecture</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Automation (CI)</td>
<td>Whether integration is fully automated</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Deployability</td>
<td>Continuous Deployment</td>
<td>Deployment architecture followed</td>
<td>Deploy based on architecture</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Automation (CI)</td>
<td>Whether deployment is fully automated</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
Started from a defragmented infrastructure...

Automated CI / CD (Local Cluster)
- SR-DEV-T1
  - App 1
- SR-DEV-T2
  - App 2
- SR-DEV-T3
  - App 3
- SR-QA-T1
- SR-QA-T2
- SR-QA-T3

Commercial Private Cloud
- System X DEV
- System X STG
- System X PROD

Active Development & Testing
- Team 1
- Team 2
- Team 3

Sprint Drops

Ceremonial Deployment
Built robust dev-test-deploy infrastructure...

Automated CI / CD (Local Cluster)

- SR-DEV-T1
  - App 1

- SR-DEV-T2
  - App 2

- SR-DEV-T3
  - App 3

Active Development & Testing

- SR-QA
  - Legacy + New integration
    - Regression Testing
    - Technical Acceptance
    - QA Acceptance

Commercial Private Cloud

- System X DEV
- System X INT
- System X STG
- System X PROD

Stable Sprint Drops

Ceremonial Deployment

Team 1
Team 2
Team 3
System X Team
Implemented automated CI/CD process...

Plan 1: CI-Server
Plan 2: CI-UI
Plan 3: CI-Integration
Plan 4: CI-Regression
Plan 5: Code Analysis

Development

Jira
Antifactory
Bamboo
SVN / GIT
Crucible
CheckMarx
SVN / GIT

SQL Dev
Eclipse

Continuous Deployment

UAT
PO Acceptance

Integration
Deployment
Smoke Testing

Regression
Deployment
E2E Testing

Build
Compilation
Unit Test
Binary

Plan 4: CI-Regression
Aligned architecture with platform technology choices...

Insert any of your favorite architecture views here...
Maintained consistent code-level standards...

Maintenance metric: Code adherence to style

- Never copy any code from any place outside of company
- Brand your code
- Use descriptive names
- Organize source file consistently
- Enhance code readability
- Explain your code: Consistent JavaDoc
- Get a step ahead of debugging: Logging
- Write code in a consistent way
Introduced new practices and skillset to the teams...

- Code reviews: each story has an owner and reviewer in Jira
  - Encouraged teams to do as-needed pair programming
- A tech lead is designated to lead a team, and an SDM manages 1-3 teams
- Added QA engineers to teams to drive quality implementation
- Customers assigned dedicated Product Owners and IT Subject Matter Expert to each team
- Added Build & Release engineers as shared resources among teams
In summary...

- Adopting an architecture-driven agile process
- Defining with stakeholders key quality attributes
- Building robust dev-test-deploy infrastructure
- Aligning architecture with platform technology choices
- Maintaining consistent code-level standards
- Introducing new skillset to the teams
Parting thoughts...

- Critical role of product owner, and authority to make decisions:
  - Voice of the customer and big influence of architecture tradeoffs

- Architecture successful implementation:
  - Dependence on key skillset within agile teams

- Quality visibility:
  - Dependence on infrastructure for iterative development (CI/CD)

- Testing:
  - Different levels of testing; different stages of development
Contact Information

Amine Chigani
GE Software
San Ramon, CA

chigani@ge.com
www.gesoftware.com
Disclaimer

General Electric reserves the right to make changes in specifications and features, or discontinue the product or service described at any time, without notice or obligation. These materials do not constitute a representation, warranty or documentation regarding the product or service featured. Illustrations are provided for informational purposes, and your configuration may differ.

This information does not constitute legal, financial, coding, or regulatory advice in connection with your use of the product or service. Please consult your professional advisors for any such advice.

GE, Predix and the GE Monogram are trademarks of General Electric Company.

©2015 General Electric Company – All rights reserved.