Agile in Financial Services

A Framework in Focus

John B. Hudson, B.Sc, PMP, CSM

PMI-NJ Chapter February 19, 2013

Objectives

1. Agile Development - an Overview
2. The Agile Enterprise - Infrastructure and Objectives
3. Agile Requirements - a Vision for the future
4. Framework's for Success - Process and Data Model / SOA

Agile Alliance

www.agilealliance.com

The Manifesto for Agile Software Development (Feb 2001)

We are uncovering better ways of developing software by doing it and helping others do it.

Through this work we have come to value:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern , Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, Dave Thomas

eXtreme Programming (XP)

- Minimalistic (lightweight) approach based on the values of Simplicity, Communication, Feedback, Respect and Courage.
- Specific rules for Iteration Planning, Managing, Designing, Coding and Testing (within a framework similar to SCRUM).
- Embracing an overall strategy that is inclusive of such techniques as write the Unit Test first, rigid coding practices, Pair Programming and a no functionality added early philosophy.
- XP projects constantly cycle through **Plan-Design-Code-Test-Deliver** with continuous integration of small product increments.
- Automated Test Tools are a “must have” (constant regression testing)

Test Driven Development (TDD)

- TDD completely turns traditional development around.
- As its name suggests, this is an approach to software development, using small iterative steps, based on the concept of:
  - First writing a test for the functionality you propose to implement (TFD - Test First Development)
  - Running the test to demonstrate that it fails (noting how and where it fails)
  - Building just enough code to support the new functionality and re-testing it (automated testing is a "must have", tests are repeated frequently)
- Regression test suite becomes "executable specifications"

Feature Driven Development (FDD)

Based on Object and Data Modeling concepts following discrete process steps as follows:
#1 Develop an Overall Model - Define system scope, conduct detailed domain walkthroughs for each area to be modeled
#2 Build a Features List - Decompose this into Subject Areas, Business Activities and Steps
#3 Plan By Feature - Plan the order that the Features are to be implemented, based on feature dependencies
***#4 Design By Feature* -** Assign a Lead Developer to form a Feature Team inclusive of the owners of the object classes involved in the development
***#5 Build By Feature* -** Development class owners implement the items necessary for their class to support the design for this feature

SCRUM

- ***Product Owner*** creates a prioritized list of features, the ***Product Backlog***, defined by ***User Stories***
- In ***Sprint Planning*** the team selects work from the highest priority User Stories to create a ***Sprint Backlog*** - they decide how to implement those stories, the team is <u>self-organizing</u>, tasks are chosen, not assigned
- Team has a fixed amount of time, usually 2-4 weeks, in each development ***Sprint***, to complete its work (design, code, test); team meets each day to assess its progress in a ***Daily Scrum***
- ***ScrumMaster*** (Agile PM?) keeps the team focused on its goal; the Product Owner remains engaged throughout the Sprint
- At the end of the Sprint, the work should be ***potentially shippable***, as in ready to hand to a customer or show to a stakeholder
- Sprint ends with a ***Sprint Review*** *and* ***Sprint Retrospective*** before restarting with Sprint Planning for the next cycle.

Industry Acceptance

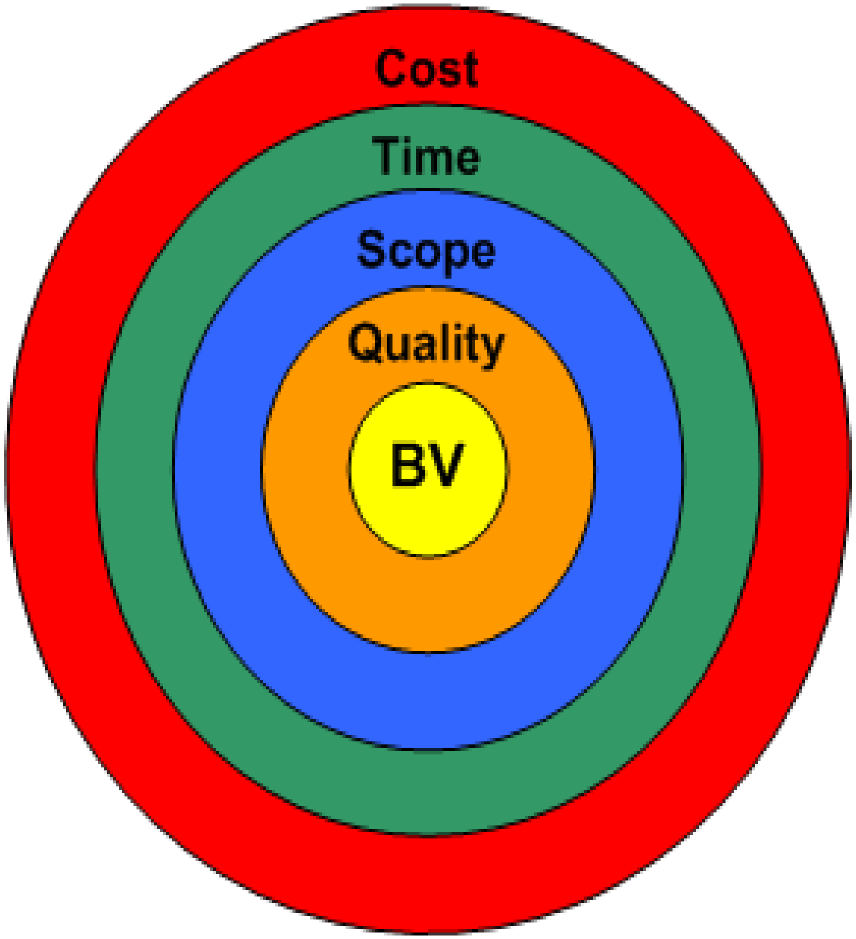
- Only **SCRUM** has emerged in widespread usage supported by software offerings from leading vendors (including CA Agile Vision, Rally Software, VersionOne and others)
- **Agile $ROI > Waterfall $ROI** since real product is delivered earlier in the schedule - this often yields improvements in CV (Cost Variance) and ETC (Estimate-To-Complete)
- **Project Management Institute (PMI)** now has an Agile Community of Practice (CoP) and, in 2011, began offering the PMI Agile Certified Practitioner (PMI-ACP) qualification
- **Software Engineering Institute (SEI)** at Carnegie Mellon University recognizes agile practices within it's Team Software Process (TSP) and Personal Software Practice (PSP) domains and notes good compliance up to Capability Maturity Model Integration (CMMI) L3

Agile Project Management Objectives

- Faster, more flexible, aggressively **Customer responsive** (delivering what you wish you had thought of at the outset)
- Delivering Business Value through **Technical excellence**
- Project Management as **Inspirational Leadership**, not administrative excellence (People, Product and Process - *in that order*)
- Opportunities for improved **Knowledge Management** and Personnel Development (increased team interaction, improved learning environment)
- Raising the bar on Project Management with a focus on **Business Value** (IT adopts the same goals and performance objectives of the Enterprise), **Quality** and **Constraints** (Scope/Time/Cost)

Agile Project Management **Target**

“the new and improved Iron Triangle?”



Agile Project Management Infrastructure

APM should be inclusive and supportive of:

+ IT/Business Integration strategy

+ Participant Roles and their Interactions

+ Required Project Artifacts, Deliverables and Performance metrics

+ Resources required to support the process (software, hardware)

+ Regulatory (Legal) and Compliance (Audit) needs

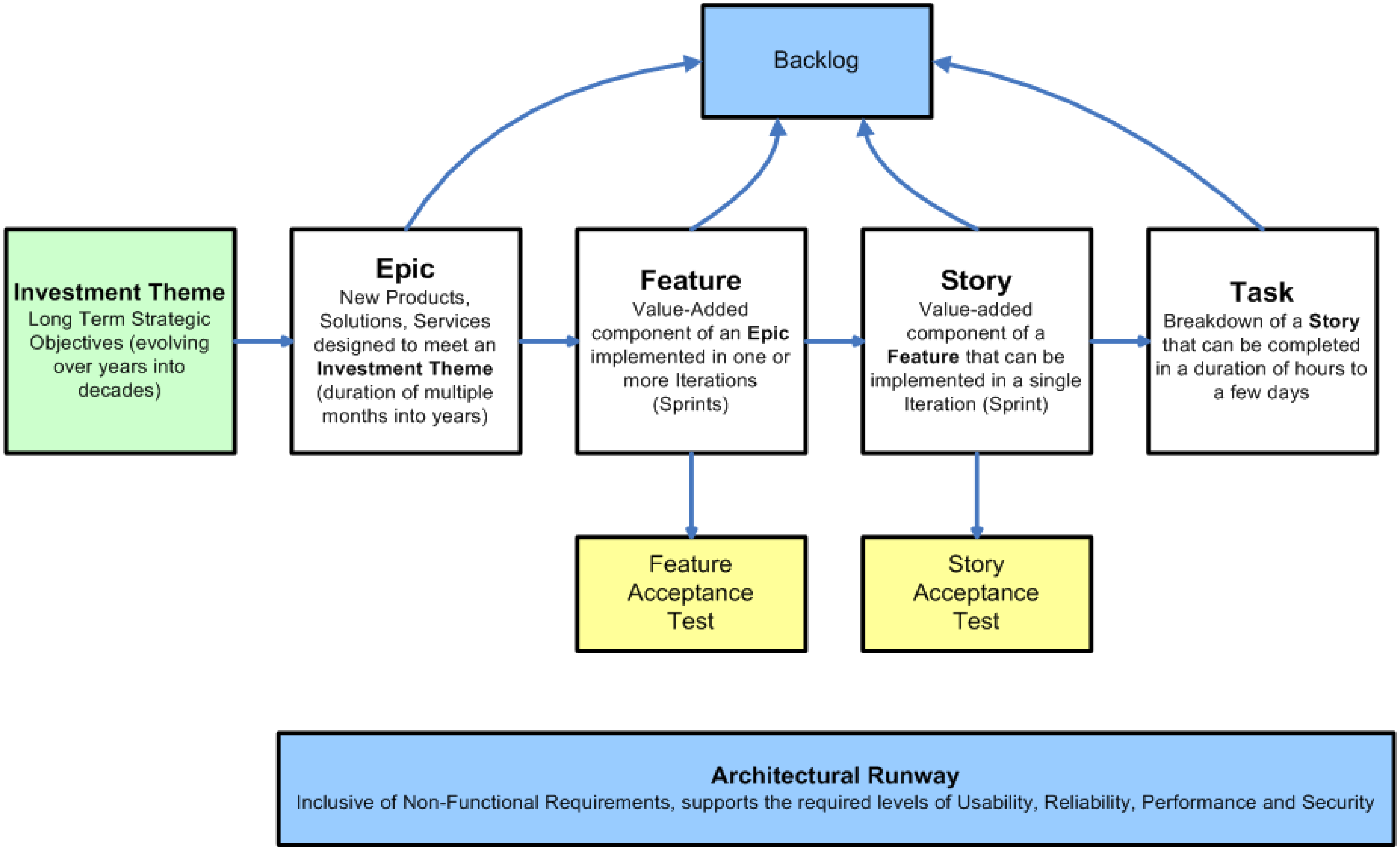
+ Data Security considerations

+ Continuous Risk Management

+ Continuous Change

+ Cultural Norms ("Lost in Translation")

Agile Enterprise Model



Backlog

Investment Theme
Long Term Strategic Objectives (evolving over years into decades)

Epic
New Products, Solutions, Services designed to meet an **Investment Theme** (duration of multiple months into years)

Feature
Value-Added component of an **Epic** implemented in one or more Iterations (Sprints)

Story
Value-added component of a **Feature** that can be implemented in a single Iteration (Sprint)

Task
Breakdown of a **Story** that can be completed in a duration of hours to a few days

Feature Acceptance Test

Story Acceptance Test

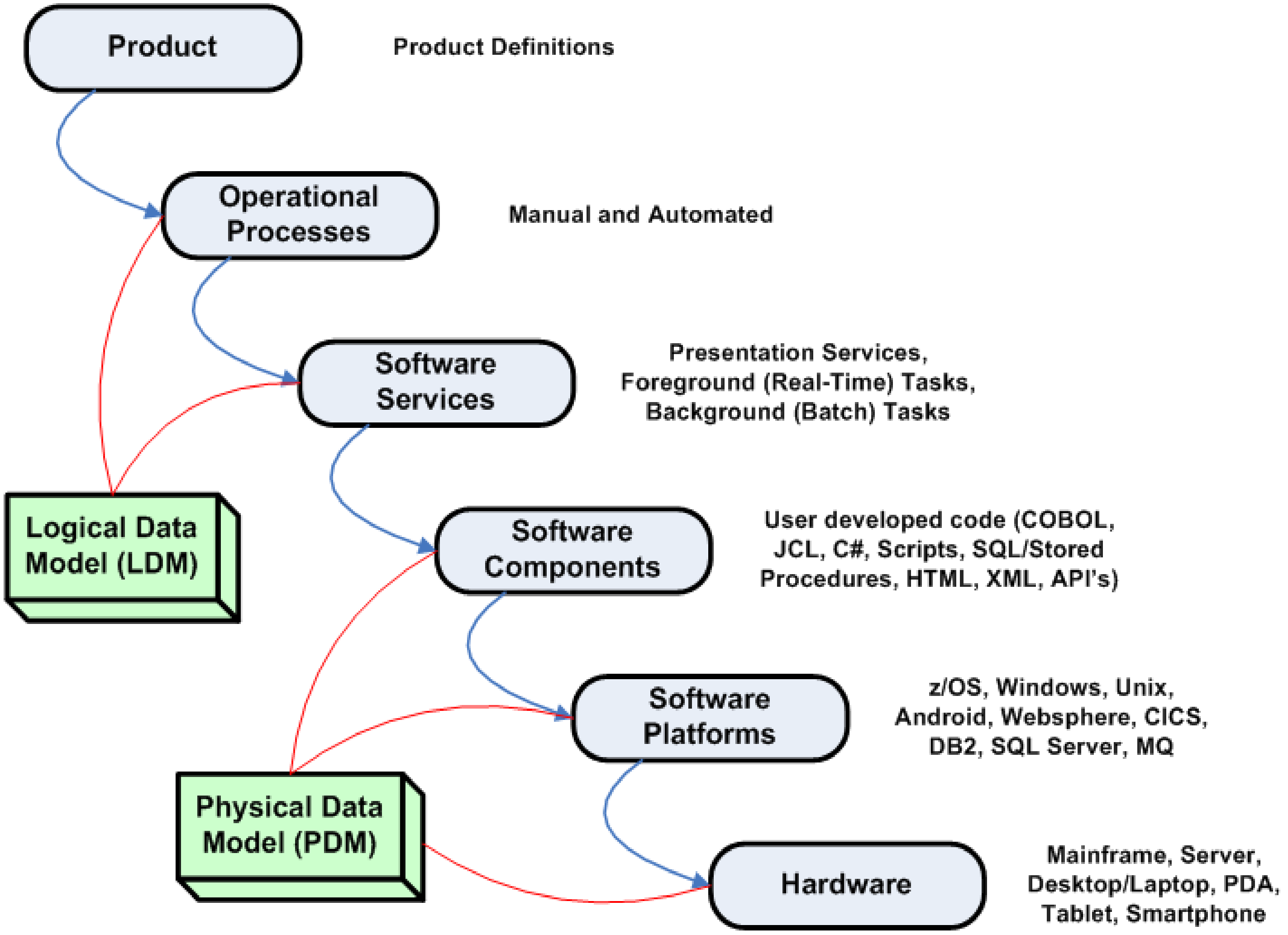
Architectural Runway
Inclusive of Non-Functional Requirements, supports the required levels of Usability, Reliability, Performance and Security

Agile Requirements

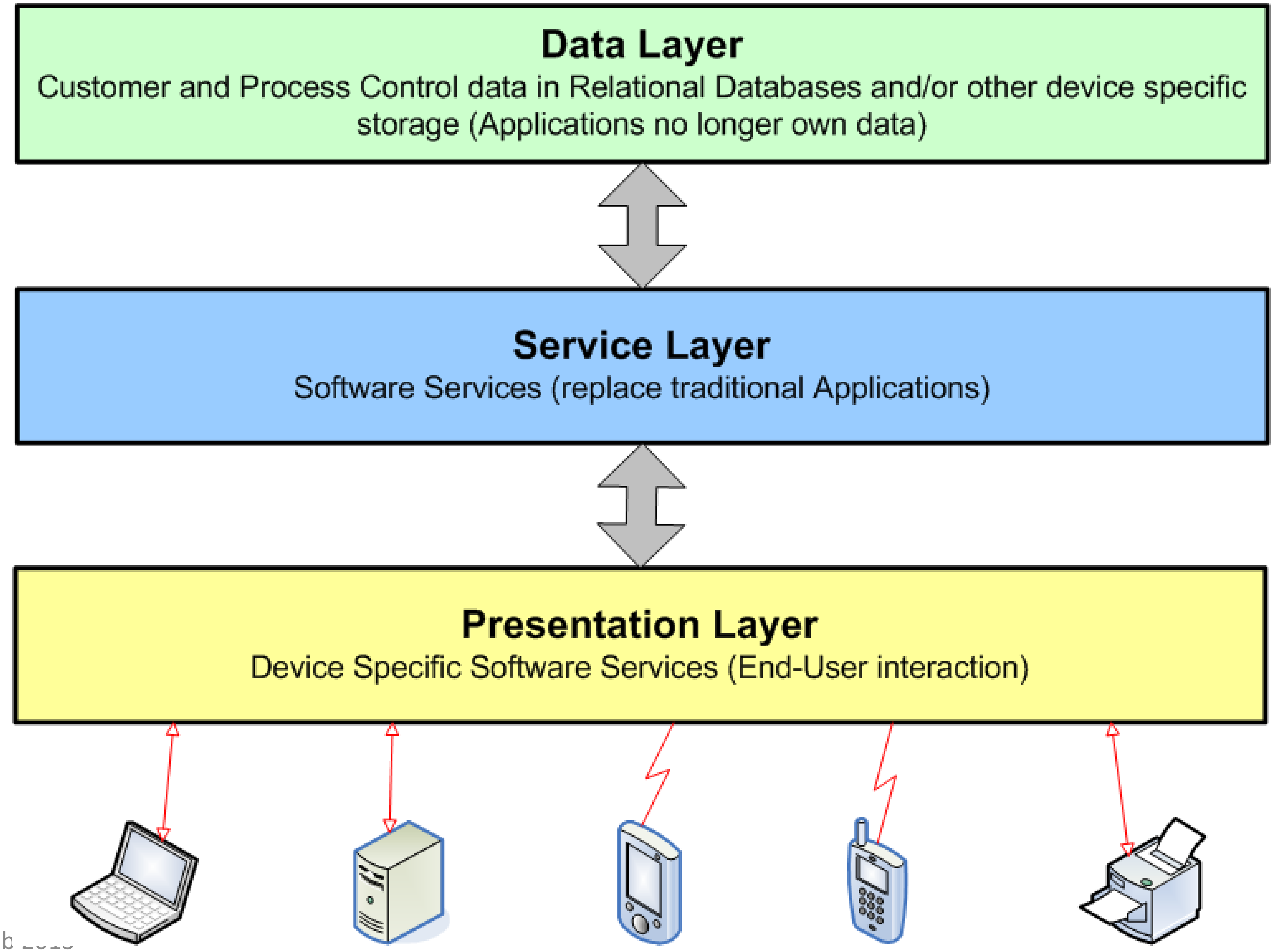
- Accommodating “I’ll know it when I see it”
- Defining Done (Approvals) - for now at least
- Functional Decomposition - Services, not Applications
- One-time use - throw-away Project documents
- Structured Writing - User Stories and Use Cases
- Back to School - “Educating” others on our Vision
- Video/Text links, Search and Indexing, Integrated Data Dictionary

Process and Data Model

Understanding IT Documentation Needs


Product
Product Definitions
Operational Processes
Manual and Automated
Software Services
Presentation Services, Foreground (Real-Time) Tasks, Background (Batch) Tasks
Logical Data Model (LDM)
Software Components
User developed code (COBOL, JCL, C#, Scripts, SQL/Stored Procedures, HTML, XML, API's)
Software Platforms
z/OS, Windows, Unix, Android, Websphere, CICS, DB2, SQL Server, MQ
Physical Data Model (PDM)
Hardware
Mainframe, Server, Desktop/Laptop, PDA, Tablet, Smartphone

Service Oriented Architecture (SOA) Model



Data Layer

Customer and Process Control data in Relational Databases and/or other device specific storage (Applications no longer own data)

Service Layer

Software Services (replace traditional Applications)

Presentation Layer

Device Specific Software Services (End-User interaction)

In Summary

- Introducing Agile Development into an environment that lacks good foundational frameworks will tend to make things worse, not better
- Agile requires not only an understanding of the processes involved but also a willingness of those participating to fully engage
- Agile primarily requires changes in People rather than Process
- Agile requires that you empower those who DO the work, not those who oversee the work
- Implementing Agile requires an effective IT/Business Integration Strategy

Acknowledgements and Recommended Reading

While all of the material in this presentation is original work (except where explicitly acknowledged otherwise), some of it has been inspired, at least in part, by content from and/or conversations with individuals from the following sources:

Agile Alliance (www.agilealliance.com)
Scrum Alliance (www.scrumalliance.org)
Software Engineering Institute (www.sei.cmu.edu)
 Software Engineering Information Repository (seir.sei.cmu.edu/seir)
Project Management Institute (www.pmi.org)
Agile Modeling (www.agilemodeling.com) - Scott W. Ambler and Associates
Agile Project Management (Author: Jim Highsmith ISBN: 978-0-321-65839-5)
Agile Project Management with SCRUM (Author: Ken Schwaber ISBN: 0-7356-1993-X)
Agile Software Requirements (Author: Dean Leffingwell ISBN: 978-0-321-63584-6)
Prudential Financial, Inc. (www.prudential.com)
Pramerica Systems Ireland Ltd. (www.pramerica.ie)