

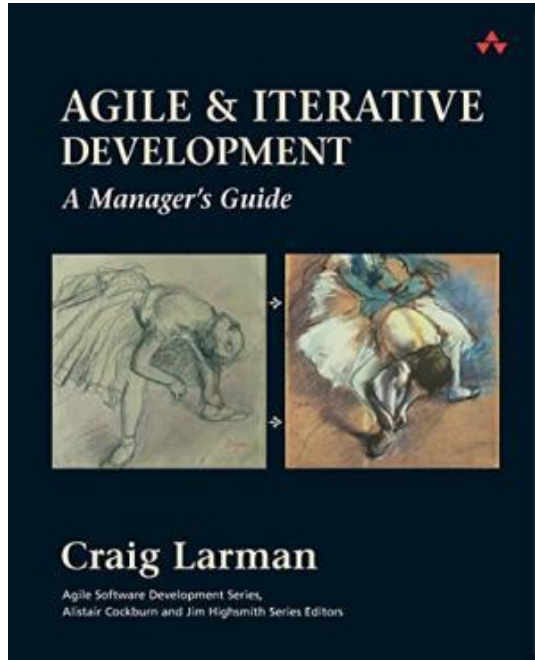


Agile Methods – An Introduction

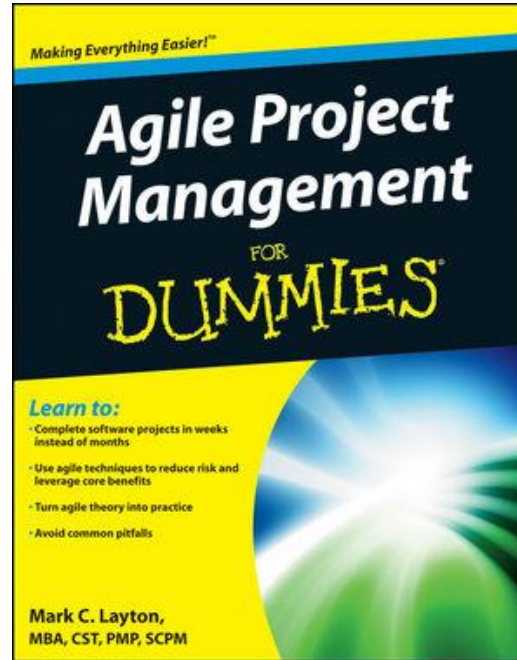
- Prof K G Krishna

Text/Reference Books

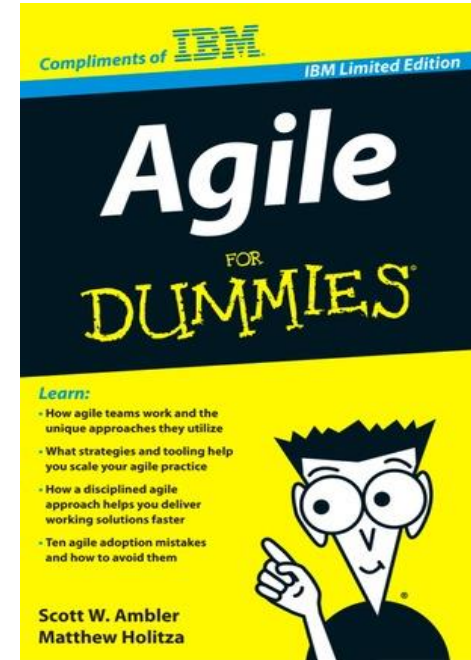
T1



T2



Compliments
of IBM



→ As this field is evolutionary, the student is advised to stay tuned to the current and emerging practices by referring to their own organization's documentation as well as Net sources

Topics

- Traditional software development practices
- Need for Agile Methods
- Benefits of Agile Methods



Traditional Software Development – The Backstory

- Large Software Projects (Mainframe era)
- Development Models based on Linear Models (Sequential/Waterfall)
- No Time-to-Market Constraints (Software is Expensive)
- Mostly Custom/Bespoke Software Development
- Programmer-Intensive Manual Activities

23/1/83 SPE

1. Replacing ENTER SPE & DIRECTIVE by *
2. Incorporating current line printer concept
3. On A25 mode display line numbers
4. Switching back to edit mode by 'NL'

A line can be added after line no 'n' by

1. POS n
2. DSP n
3. Any line no between termination line is 'n'

Ex: DSP m,n

SEL m,n

CRB 1 1 m,n

DEL m,n

REP m,n

PRT m,n *See Printing "NL"

1. *ADD n at any pos in SPE
2. *XNUM: controls the way printed at 'n' displaying the line
3. Defauld line no: 'SEL' (only user's line)
4. 'SEL m,n' (adding second line no)
5. Specifying anaphora for no's lines

Ex: SEL m @ n, after m, n lines

or SEL m,n

2. *END: bring from m
1. Specifying default 'A' Ex: SEL A (executable) and 'L' (storage) CHG /S/S/A

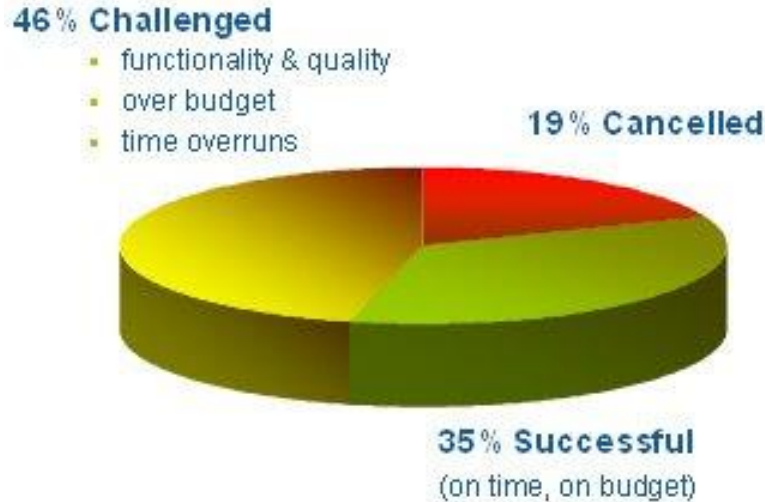
K G Krishna



K G Krishna

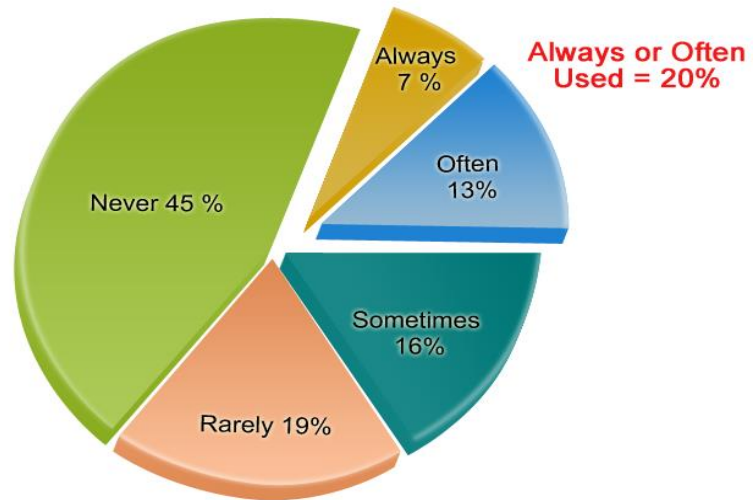
"A temporary endeavour undertaken to create a unique product, service, or result." -
A Guide to the Project Management Body of Knowledge (PMBOK® Guide), 4th Edition
(Project Management Institute, 2008)

Traditional Software Projects (circa 1990 ~ 2000)



Source: Standish Group Chaos Report [2006]

Features and Functions Used in a Typical System



Standish Group Study - Reported at XP2002 by Jim Johnson, Chairman
Copyright © 2011 luuduong.com

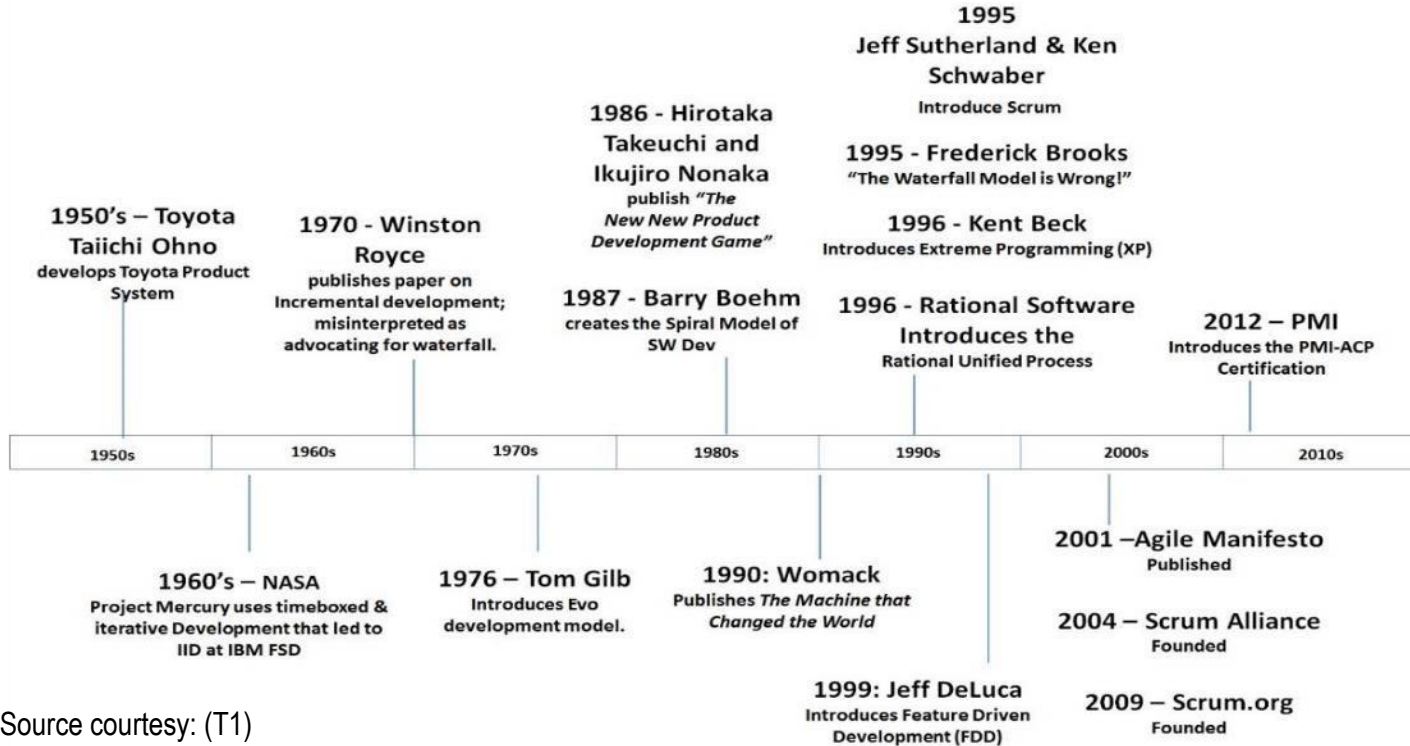
In 2009, companies and organizations in the U.S. spent \$491.2billion on application development. That means that more than \$103billion was wasted on failed projects.

Agile Turns Upside-down Project Constraints



Production-Line → Agile System – An Evolution


A Brief History of Agile



Source courtesy: (T1)

Being Agile Means...

agile

/ˈɑːɡaɪl/ 

adjective

1. able to move quickly and easily.

"Ruth was as agile as a monkey"

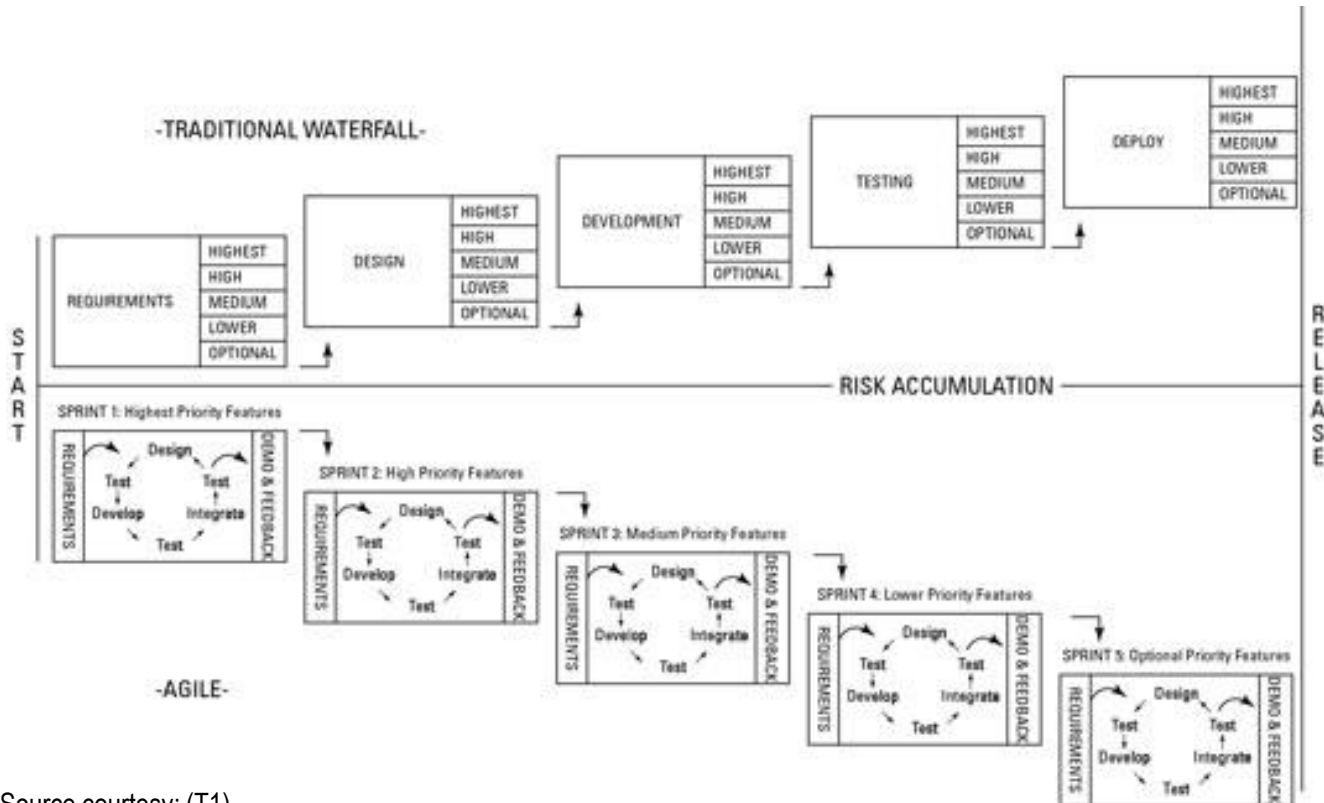
synonyms: nimble, lithe, spry, supple, limber, sprightly, acrobatic, dexterous, deft, willowy, graceful, light-footed, nimble-footed, light on one's feet, fleet-footed; [More](#)

- Agility

The ability to both create and respond to change in order to profit in a turbulent business environment

- Rigid Processes vs. Agile Frameworks
- Being Agile = Competitive, Responsive, Flexible,...

Agile Development – A Series of Short Sprints



Source courtesy: (T1)

Agile Development = Iterative Releases!

- **Agile software development** is a conceptual framework for software engineering that promotes development **iterations** throughout the life-cycle of the project.
- Software developed during one unit of time is referred to as an iteration (**sprint**), which may last from one to four weeks.
- Agile methods also emphasize **working software** as the primary measure of progress

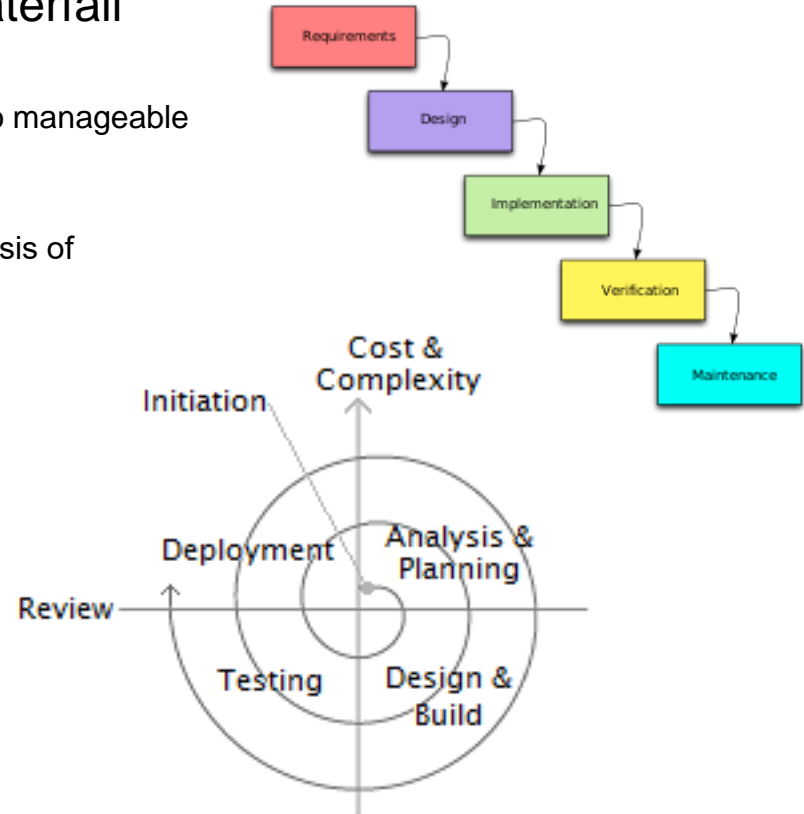
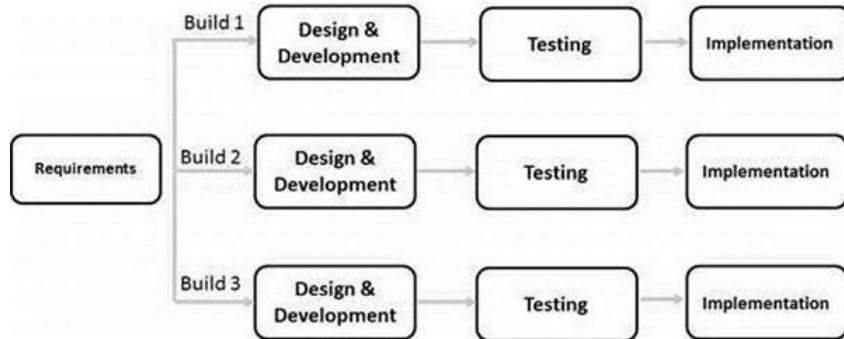
Need for Agile Methods ➔

Software Systems Today...

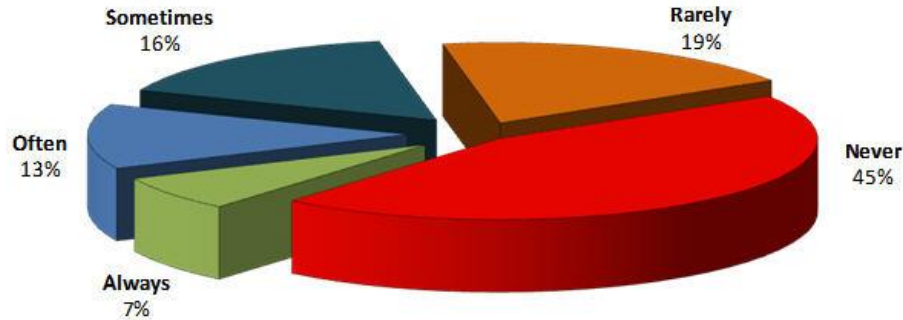
- Changing ('Unclear') Requirements / Customer Demands
"Requirements will be clear only at the end of the Project"
- Shrinking Development Cycles (Time-to-Market Releases)
- Entrepreneurial, Innovative Apps
- Shorter Shelf-life

Addressing Challenges of *Waterfall* Model

- Traditional Waterfall → Adaptations of Waterfall (Incremental, Iterative Prototyping,...)
 - Focus on Requirements Management (Breaking down into manageable 'Increments')
 - De-risking Large Development Effort
 - Capturing Requirements ('show-and-tell', 'prototype' as basis of discussion around Requirements)
 - "Quick-and-dirty" Working Prototype to Start with



Software Feature Overload

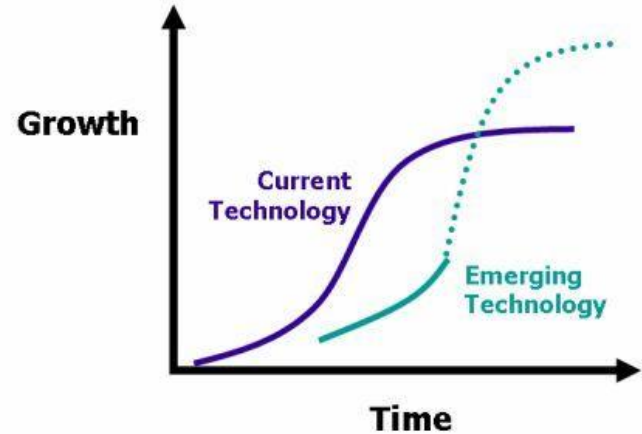


Source: Standish Group Study, 2002



Pace of Technology vs Project Duration

- Technology Life-cycles have shrunk (decade → 1..2..3 years)
- Bespoke Software → Product Customization
- Line-by-line Coding → Integrated Development Frameworks



De-risking Future Investments Upfront...

- “Fail-Safe” early rather than Failure at the end
- “Proof-of-concept” when adopting new Technologies
- “Testing the waters” before launching Big in the Marketplace
- Today’s Software Products are strategic levers--“Idea-driven” rather than mere automation of manual process

Benefits of Agile Methods ➔

Controlled Development

- Agile Products are based on empirical control method – decisions based on reality
- Adjustments on-the-go by Frequent Inspections
- **Transparency:** Everyone involved knows what is going in the project
- **Frequent Inspection:** Regular evaluation of the Product
- **Adaptation:** Make quick adjustments to minimize problems later

Agile Development → Agile Project Management

- Traditional Project Management Turned Upside-down!
- “Let’s wait till the Project completes to see the Product” →
“Several Min-projects with little visible successes”
- Transformation of Project Management by actively involving ALL the Stakeholders; Organization Structures & Communication; Time-Boxing of Deliveries

“Standish Group Study on Software project success and failure: In 2009, 26 percent of projects failed outright — but in 2011, that number fell by 5 percent. The decrease in failure has, in part, been attributed to wider adoption of agile approaches

Benefits of Agile Project Management

- *Almost Zero* Risk of Catastrophic Project Failure
- Prioritization of Business Value over 'Good or Nice-to-have' features
- Agile Testing (Continuous Testing) ensures Problems are discovered early
- Down-plays 'Scope-creep' as Requirement Changes are managed throughout Product Development Life-cycle
 - Prioritizing Features in early Iterations
 - Managing Evolving Requirements
- Continuous Inspection and Adaptation: Improvement of Processes and Products based on Prior Experience of the 'Completed' Product

Agile Methods - Summary

- Traditional Software Development Methods involving Large One-time Projects are yielding to Low-risk High-turnaround Incremental Deliverables in Agile Methods
- Involvement of All Stakeholders (including Customer) early on in the Process enhances Collaboration and minimizes Scope-creep
- Full Transparency and High-visibility of Deliverables in Short Iterations (Sprints)
- Continuous Quality Monitoring with embedded Agile Testing across all Iterations

“Agile is the Great Leap Forward in Software Development Methodology with its associated Transformation in Agile Project Management”

Thank You

© Copyrights of original Authors are duly acknowledged

™ ® All Trademarks, Registered Trademarks referred in this document are the property of their respective owners