


---

---

---

---

---

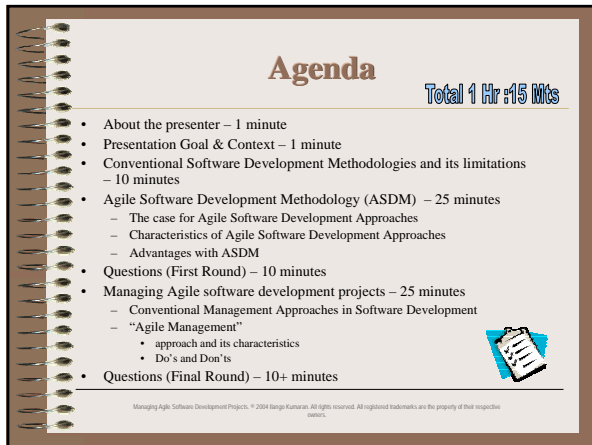
---

---

---

---

---




---

---

---

---

---

---

---

---

---

---




---

---

---

---

---

---

---

---

---

---

## About the presenter


**S. ILANGO KUMARAN**  
[WWW.ILANGOKUMARAN.COM](http://WWW.ILANGOKUMARAN.COM)  
Project Manager-and-Enterprise Architect

**Academic Qualification**

- Master in Engineering (ME)
- Master in Business Administration (MBA).

**Publications**

- Author of the Amazon's best selling and highly reviewed book "*Jini Technology: An Overview*" published Nov, 2001 by Prentice Hall, Technology Series.
- Author of numerous published articles in leading technical Journals such as *XML Journal* and *Java Report*



Managing Agile Software Development Projects. © 2004 Ilango Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---


---

---

## Presentation Goal & Context

The goal of this presentation is to

1. Define and build a case for Agile software development methodologies
  - As a part of this goal, I will highlight the differences between classic software development approaches and agile software development approaches
2. Highlight the challenges in managing an agile software development project
  - As a part of this goal,
    - I will highlight the differences between conventional management techniques and agile management techniques
    - I will provide a list of do's and don'ts while managing Agile projects



Managing Agile Software Development Projects. © 2004 Ilango Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---


---

---

---

---

# Conventional Software Development Approaches



Managing Agile Software Development Projects. © 2004 Ilango Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

## Classic Software Development Life Cycle (SDLC)

Classic SDLC is composed of the following phases

- Feasibility Study
- Requirements
- Design
- Development & Integration
- Testing

**Good News: Everyone agrees on the phases involved in a software development life cycle**

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

## Conventional Software Development Approaches/Models

**Not-so Good News: There are more than one way to sequence or arrange the SDLC phases - a.k.a process/methodology**

- Waterfall Model
- Fountain Model
- Spiral model
- Build and fix
- Rapid Application Development (RAD)
- Incremental model
- Etc etc .. (including proprietary methodologies)

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

## Conventional Software Development Models

Lifecycle Model	Characteristics
<b>Waterfall Model</b>	<ul style="list-style-type: none"> <li>• <b>"Most well understood"</b></li> <li>• Most Rigid, The next phase cannot be started until the previous phase is completed. All tasks has a Finish-to-Start (F-S) relationship</li> <li>• Assumes that all requirements can be well specified in advance</li> <li>• Provides excellent project control/progress parameters for managing the process</li> </ul>
<b>Fountain Model</b>	<ul style="list-style-type: none"> <li>• <b>"Not well known"</b></li> <li>• Methodology is based on the realization that while some phases have an interdependency(F-S relationship), there are considerable overlap among other activities(S-S or S+lag-S)</li> <li>• Need skilled project managers to manage this type of process</li> </ul>

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

## Conventional Software Development Models

Lifecycle Model	Characteristics
<b>Spiral Model</b>	<ul style="list-style-type: none"> <li>• A "series of waterfall" representing a part of the entire project.</li> <li>• The spiral model emphasizes the need to go back and reiterate earlier stages a number of times as the project progresses.</li> <li>• Helpful when you want to demonstrate a proof of concept early in the cycle.</li> <li>• Due to its repetitive characteristics, project can become chaotic if improperly managed.</li> <li>• Project control and progress values can be confusing if not well understood.</li> </ul>
<b>Build and Fix</b>	<ul style="list-style-type: none"> <li>• "Crudest of the methodology"</li> <li>• Write some code, and then keep modifying it until the customer is happy.</li> <li>• Without planning, the technique is very open-ended and can be risky.</li> <li>• Can result in poor design and/or poor quality product.</li> </ul>

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Conventional Software Development Models

Lifecycle Model	Characteristics
<b>Rapid prototyping</b>	<ul style="list-style-type: none"> <li>• "Demonstrate and Discard"</li> <li>• Goal is to               <ul style="list-style-type: none"> <li>• Elicit and Validate Requirements</li> <li>• Prove a project or technology feasibility</li> </ul> </li> <li>• Usually when the project or prototype is approved, the prototype is usually discarded and real software is written.</li> <li>• Most success reported in UI projects</li> </ul>
<b>Incremental Model</b>	<ul style="list-style-type: none"> <li>• "Divide and Build"</li> <li>• The incremental model divides the product into builds, where sections of the project are created and tested separately.</li> <li>• Advantage is, this approach will likely find errors in user requirements quickly, since user feedback is solicited for each stage and because code is tested sooner after it's written.</li> </ul>

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Software Development Models Vs Software Development Organization Maturity

Organization lean towards more rigid and conventional development models as they pursue higher maturity standards

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Conventional Software Development Approaches : Challenges

**Challenges in following rigid methodologies such as Waterfall**

- Adopting Rigid Methodologies helps
  - To ensure repeatability
  - To have more project controls
  - To create compliance check lists
- Such rigid methodologies
  - Do not take the fact into account that all the requirements are NOT known upfront
  - Do not take advantage of the possible phase overlaps during software development
  - As tangible results are available only after the testing phase, these methodologies keep building the risks till the end of the project. This can lead to catastrophic failure for the project.

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Conventional Software Development Approaches : Challenges

- As more staff and more efforts are needed to comply with the rigid process, it increases project cost. Such increase in project cost reduces project profit margin
  - *Makes a strong case for offshore development if the organization prefers to be process-centric ☺*

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Conventional Software Development Approaches : Reality Check



- Not all the development organization are positioned well enough to absorb an increase in the process cost
- Not all the projects are critical enough to go through such rigorous process
- Not all organizations are planned well enough to leverage offshore development
- Not all the project can be off-shored. The reason could be the project's sensitivity or complexity or criticality or others

Are there alternative approaches?

Managing Agile Software Development Projects, © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---




---

---

---

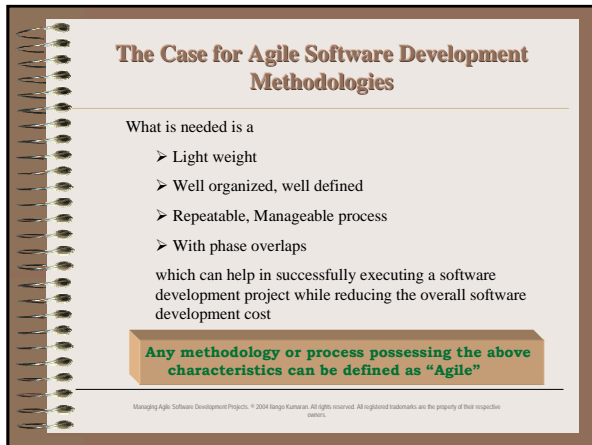
---

---

---

---

---




---

---

---

---

---

---

---

---




---

---

---

---

---

---


---

---

## Agile Software Development Methodology – Current List

**Today more than a dozen agile approaches are known/published**

[Aspect Oriented Programming](#)  
[Big Design Upfront](#)  
[Crystal](#)  
[DSDM](#)  
[Extreme Programming](#)  
[Feature-Driven Development](#)  
[Lean Development](#)  
  
[Pair Programming](#)  
[Rational Unified Process](#)  
[Scrum](#)  
[Test Driven Development](#)  
[Usability Design](#)



Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

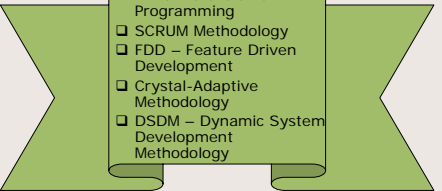
---

---

---

## Agile Software Development Methodology – Widely Known/Adopted List

**The widely known and popular Agile methodologies are:**



XP a.k.a extreme Programming  
 SCRUM Methodology  
 FDD – Feature Driven Development  
 Crystal-Adaptive Methodology  
 DSDM – Dynamic System Development Methodology

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Agile Software Development Methodology - Philosophy

- Agile Software Development is based on the philosophy that
  - Requirements are ALWAYS unclear at the beginning of a development, hence instead of waiting for them, it would be prudent to initiate the development of the system with
    - the known requirements
    - in small iterations (4-6-8 weeks)
    - with a small team (3-20)

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Agile Software Development Methodologies – Characteristics

- In classic SDLC terms Agile methodologies follow a combination of incremental, spiral and fountain.
- All Agile methodologies are Incremental & Iterative
  - Each Iteration
    - Begins with a planning phase
    - Followed by a Design, Development, Integration & Testing phase
    - A stakeholder evaluation is scheduled at the end of each iteration.
    - Every stakeholder evaluation results in a Go/No-Go" decision
      - A GO moves to the next iteration while a No-Go results in project termination
- From development perspective
  - Small Self managed team
  - Complete customer participation
  - Adopts team building techniques such as pair programming, scrum meeting etc

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

## Extreme Programming (XP)

Copyright 2001 E. Thomas Wirth

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

## SCRUM Approach

Pre - Game	Mid - Game	Post - Game
<div style="border: 1px solid black; padding: 5px; width: 80%; margin: auto;">           Planning and High-Level Design         </div>	<div style="border: 1px solid black; padding: 5px; width: 80%; margin: auto;"> </div>	<div style="border: 1px solid black; padding: 5px; width: 80%; margin: auto;">           Closure         </div>

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

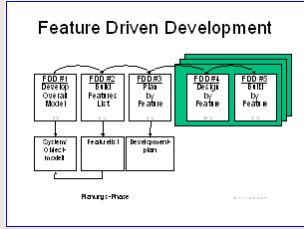
---

---

---



## Feature Driven Development (FDD)



Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

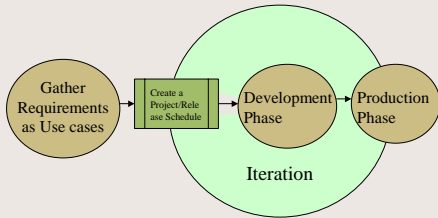
---

---

---

---

## Crystal-Adaptive Approach



Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

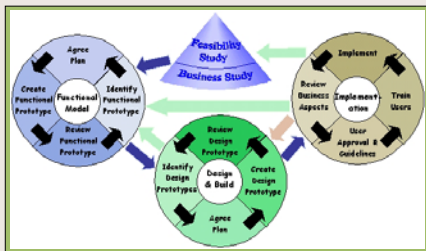
---

---

---

---

## Dynamic Systems Development Methodology (DSDM)



Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Agile Software Development Methodology - Advantages

**Advantages**

- A small iteration of 4-8 weeks means
  - **Tangible results are available** to evaluate every 4 to 8 Weeks
  - **Project risk is completely minimized** as errors can be identified early in the life cycle.
  - **Better project controls** – as projects can be evaluated every 4 to 8 weeks.

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Agile Software Development Methodology - Advantages

**Advantages**

- A small team of 4-20 means
  - **Less communication overhead** ( $n(n-1)/2$  channels)
  - **Lower resource cost**
  - **Lower operating cost** (Expenses, Travel etc)
  - **Less "burn rate"**
- A light weight process means
  - **Less overhead for process-compliance**
  - **Higher margin in project execution**

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---


---

---

---

---

# Management Approaches in a conventional Software Development



Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Conventional Management Approaches in a Software Development Project

- Conventional Management believes in
  - Adopting rigid procedures and organizational structure
    - **Rationale:**
      - Rigid and detail procedure/process provides greater control
      - Increased control results in increased order
  - Adopting hierarchical Teams
    - **Rationale:**
      - Best team model to provide greater control
      - Increased control results in increased order
  - Project Managers performing the role of Task Masters. To Manage, Track, Control and Report on Tasks
    - **Rationale :**
      - Managing detail level tasks help to identify variances very early

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Conventional Management Approaches in a Software Development Project

- Decompose work based on work packets (WBS). Breakdown work packets further to a detail level task that are meaningful and manageable
  - **Rationale :**
    - Easy to assign a resource
    - Easy to track any variance
- Team members are “replaceable” resources
  - **Rationale :**
    - Most of the time, resources are identical in skills
- Operationally, most of the time, resources are un-evenly loaded
- Complete scope and planning must precede any design and development task
  - **Rationale :**
    - Scope can be “nailed down” for the project
    - Scope change can be “stone walled”

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---


## Conventional Management Approaches in a Software Development Project - Summary

**In Short**

Conventional Management is a

- Rigid,
- Hierarchical,
- Task focused
- Supervisory

model of management



Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---


---

---

---

---

# Management Approaches in a Agile Software Development



Managing Agile Software Development Projects © 2004 Bangi Kurnian. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---


---

---

---

## Managing Agile Software Development Projects

- Agile project needs “Agile Management”
- “Agile Management” relies on “Leadership style” management
  - Agile Manager’s role is to manage and react to changes instead of resisting to changes.
  - The role need to be people-driven than process-driven
- Agile Manger must be goal-oriented, adaptable and flexible in order to succeed



Managing Agile Software Development Projects © 2004 Bangi Kurnian. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---



---

---

## Managing Agile Software Development Projects

### Who is a Leader?

- There is enough literature on leadership.
- A definition stated by Stephen Covey is what I would like to state here:
  - “A leader is the one who climbs the tallest tree, surveys the entire situation and yells, “Wrong Jungle!””

Managing Agile Software Development Projects © 2004 Bangi Kurnian. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

## Managing Agile Software Development Projects

- Agile Management
  - Needs Leaders and not Task Masters
    - **Rationale :**
      - Leaders understand that activity is not necessarily accomplishment
  - Adopts loosely coupled procedures and organizational structure
    - **Rationale :**
      - Believes in spontaneous self-organization of teams
      - People naturally align themselves and follow leaders stronger than themselves
  - Adopts Flat team structure
    - **Rationale :**
      - In order to facilitate "emergent order" instead of "imposed order"

Managing Agile Software Development Projects. © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

## Managing Agile Software Development Projects

- Decompose effort based on goals (tangible deliverables) and drill down to a level which can be identified as sub-goals (phased tangible deliverable)
  - **Rationale :**
    - Easy to organize, manage and track the team's progress in terms of goals and sub-goals instead of tasks
- As goals are team based, each team-member becomes critical in the achieving the results
  - **Rationale :**
    - Achievement is based on team-collaboration and collective skills
    - Need to properly build the team with right mix of specialists and generalists
- Planning is restricted to an iteration
  - **Rationale :**
    - As only that much of requirement is clearly known and hence forms the scope

Managing Agile Software Development Projects. © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

# Management Approaches in a Agile Software Development

Managing Agile Software Development Projects. © 2004 Bangor Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

### Managing Agile Software Development Projects - Do's and Don'ts

- **Time :**
  - Do not have an iteration spanning more than 6-8 weeks. The benefits of Agile approach will be lost, when the duration becomes longer
- **Resource :**
  - Do not have a team size larger than 7. If more than 7 break it down into different teams with sub-deliverables
- **Scope :**
  - Do not include requirements which are not finalized at the iteration planning time. This can minimize the extent of rework in the subsequent iteration.
- **Change Control :**
  - Avoid change control during an iteration. Any new requirements identified after an iteration has begun, must be queued for the next iteration.

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

### Managing Agile Software Development Projects - Do's and Don'ts

- **Team :**
  - Identify a core skill for the team
  - Create equally important roles within the team. Avoid hierarchical roles, unless needed for a training.
  - Balance the team with specialized and overlapping skill sets
- **Management :**
  - Do not use conventional supervisory style management
  - Allow emergent order within the team
  - Encourage self-management among the team members with clearly defined SMART (Simple, Measurable, Agreeable, Reasonable, Timed) goals
  - Keep short, frequent meetings with team members to track progress, plan out tactics or change strategies.
  - Recognize and Reward the team member's contribution and team's achievement.

Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---


---

---

---

### Managing Agile Software Development Projects - Parting thought

Any one can steer the ship, but it takes a leader to chart the course



Managing Agile Software Development Projects, © 2004 Banga Kumar. All rights reserved. All registered trademarks are the property of their respective owners.

---

---

---

---

---

---

---

---

---

---

---

---

*"Management of Java-based projects and development in other project cases is the foundation to take a look at Java technology, then after a while working on implementing their objectives based on a project."*  
*— from the Foreword by W. Scott Eckhardt, editor of Java, 2nd Edition, and the Director of Oracle, Java 2000, 2001*

**JINI**  
**TECHNOLOGY**  
*An Overview*


• Jini technology for developers

• Concepts, applications, advantages, and limitations

• Solutions for device and enterprise connectivity

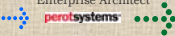
**S. ILANGO KUMARAN**  
 Software Engineer - Product Sales, Inc. (Microsoft, Inc.)

**Questions**  
**Managing Agile Software Development Projects**



**S. ILANGO KUMARAN**  
[WWW.ILANGOKUMARAN.COM](http://WWW.ILANGOKUMARAN.COM)  
 ILANGO@ILANGOKUMARAN.COM

Project Manager-and-Enterprise Architect



Managing Agile Software Development Projects © 2004 Ilango Kumaran. All rights reserved. All registered trademarks are the property of their respective owners.

---



---



---



---



---



---



---



---