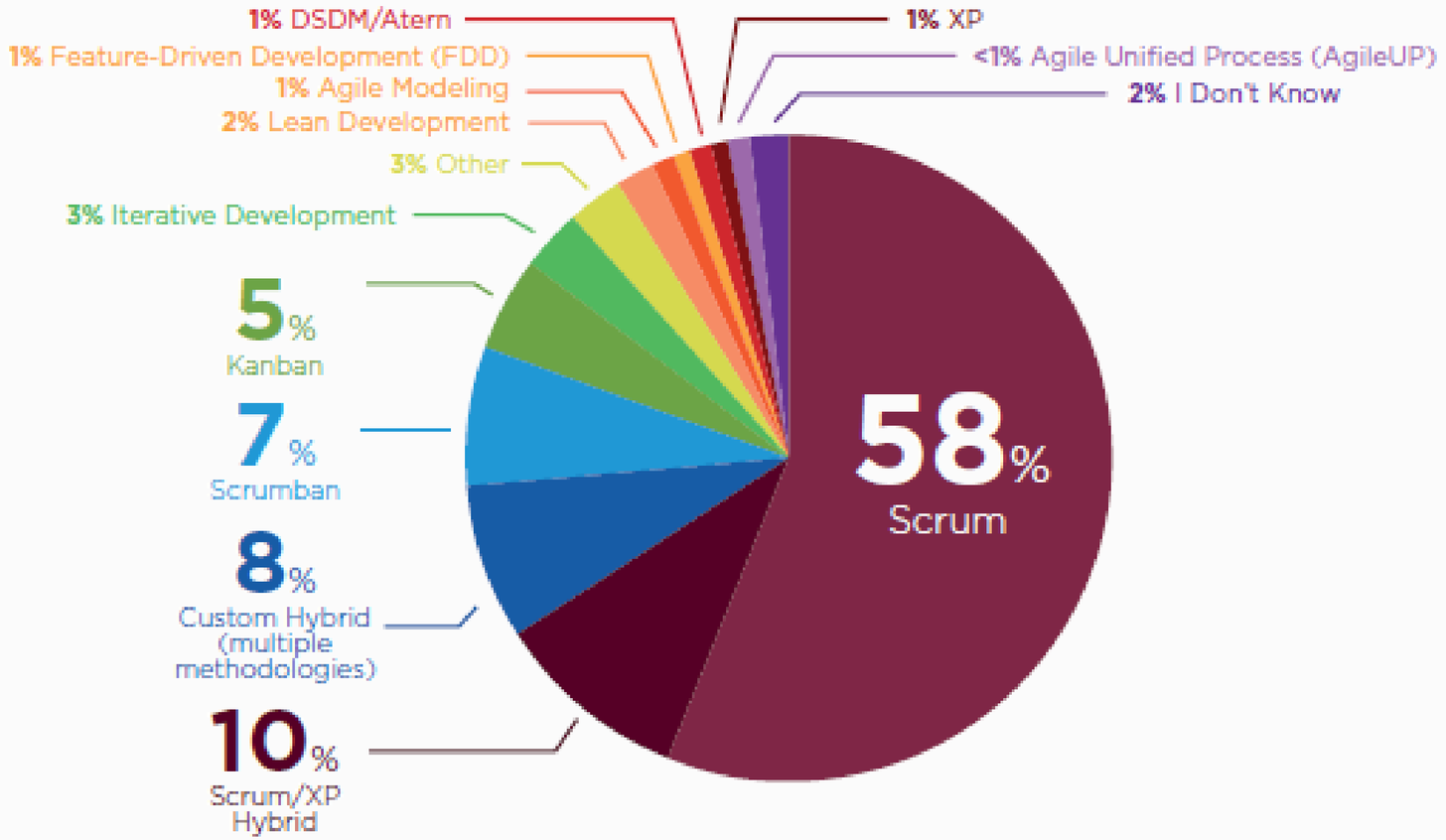
Other Agile Approaches & Methodologies

Most common Agile Methodologies

- Scrum
- XP
- Kanban

  => Lean



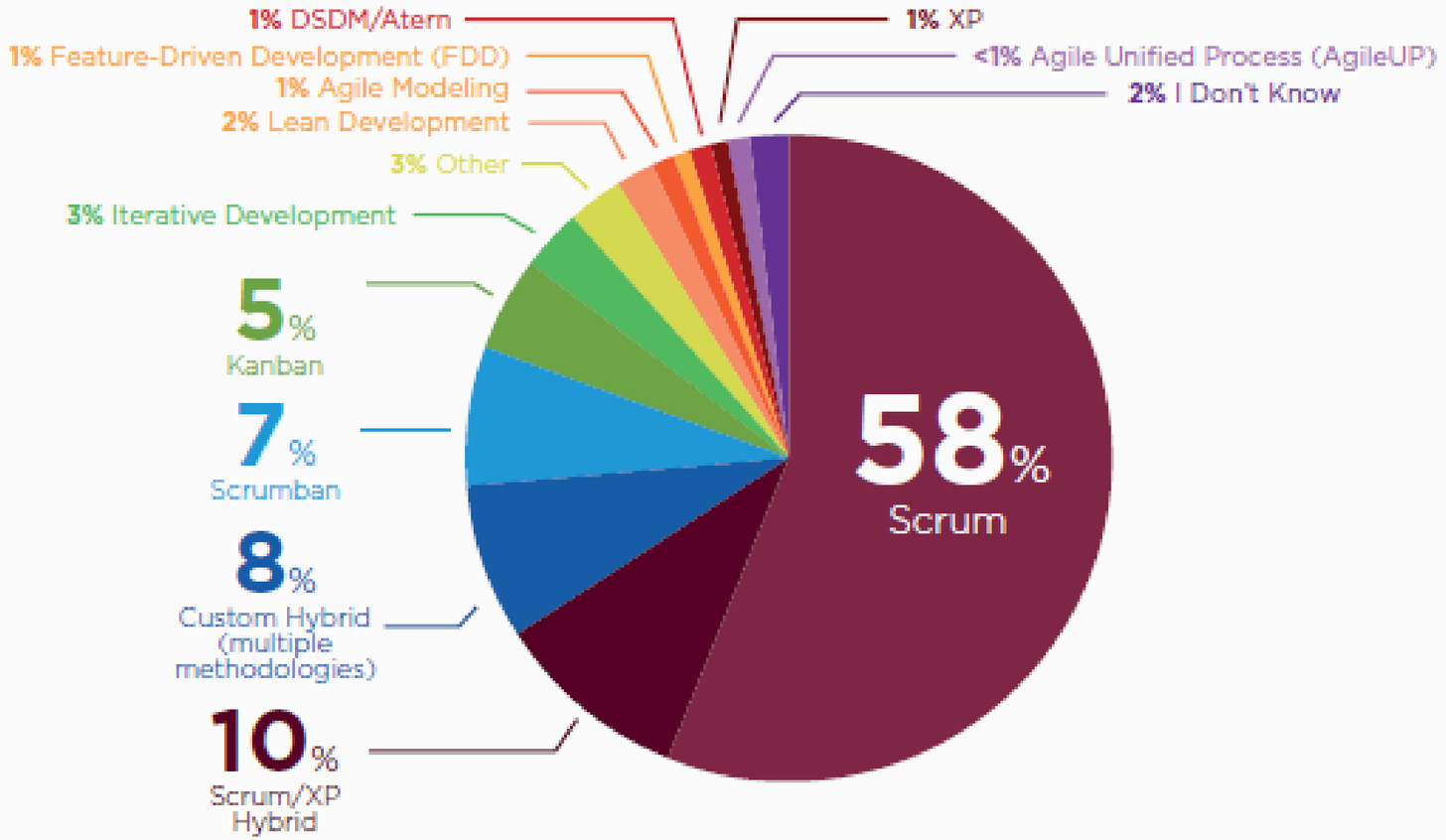
10th Annual State of Agile Report, 2016

Most common Agile Methodologies

- Feature-Driven Development
- Agile Unified Process

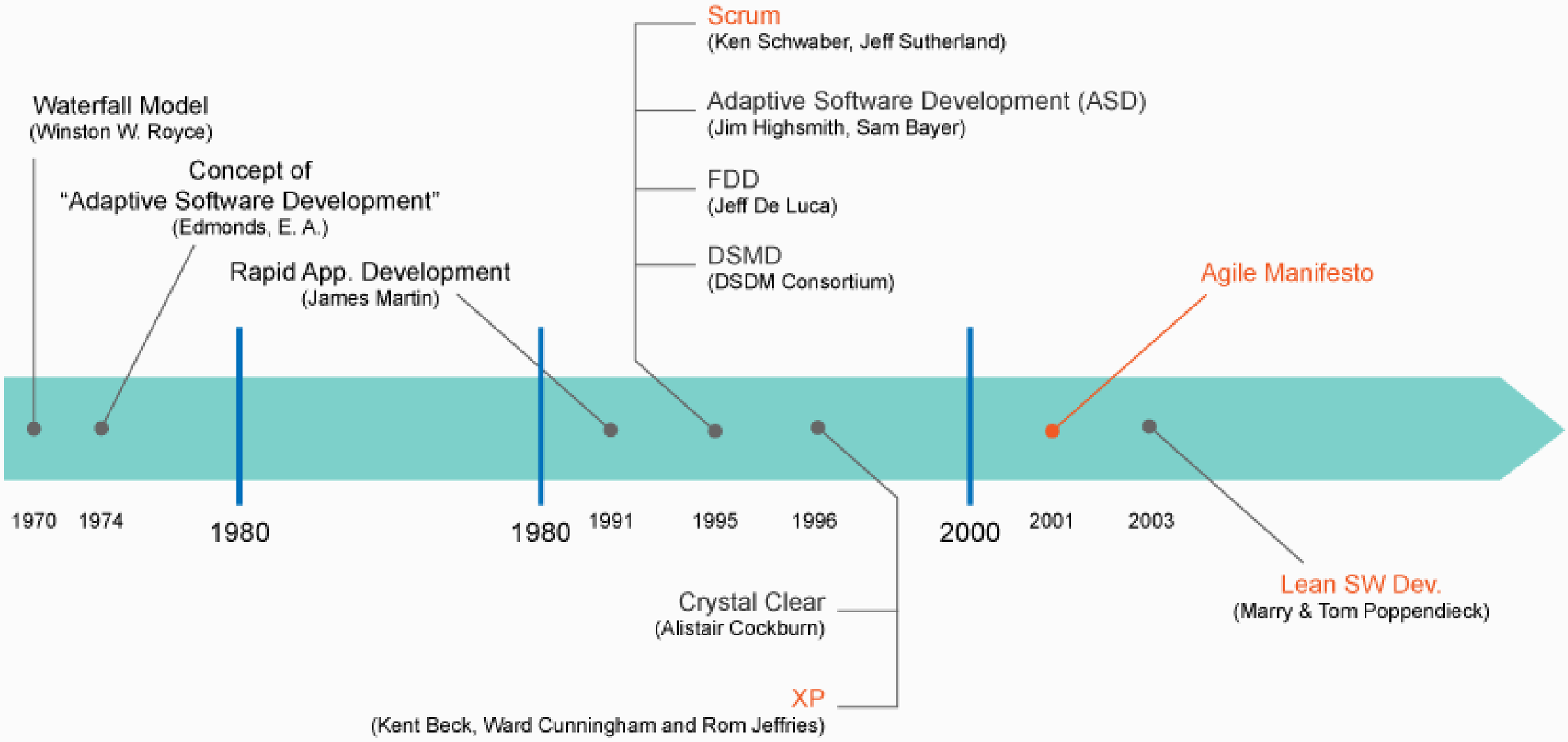
/ Essential Unified Process

- Crystal
- Dynamic Systems Development Method
- Modern Agile



10th Annual State of Agile Report, 2016

History of Agile



40 Agile Methods

Craig Smith

Open your eyes to other methods

Modern Management Methods and Lean Software Development

- PRODUCT DEVELOPMENT FLOW
- KANBAN
- DEMING — PROFOUND KNOWLEDGE SYSTEM
- SCRUM — Scrumbut/ Scrumand
- LEAN
- XP
- CRYSTAL
- DSDM
- FDD
- ASD
- PERSONAL KANBAN
- SCRUM PLOP — PATTERN LANGUAGES OF PROGRAMS
- ETF — Enterprise Transition Framework
- LEAN STARTUP
- CERTIFICATIONS
- ACCELERATED AGILE
- HYBRID — SCRUMBAN, XANPAN, NONBAN, WATER SCRUM FALL
- AGILE UNIFIED PROCESS
- EXTREME (WIKISPEED) MANUFACTURING
- DISCIPLINED AGILE DELIVERY (DAD)
- TDD/ATD/BDD/SBE — Begin with the end in mind
- VANGUARD METHOD
- CONTEXT DRIVEN TESTING
- BEYOND BUDGETING
- HOLOCRACY
- RIGHTSHIFTING
- MANAGEMENT 3.0
- STOOS NETWORK
- RADICAL MANAGEMENT
- DRIVE
- CYNEFIN
- THEORY of CONSTRAINTS
- ENTERPRISE UNIFIED PROCESS (EUP)
- LARGE SCALE SCRUM (LeSS)
- ENTERPRISE SCRUM
- XSCALE (AGILE TNG)
- SCALED AGILE FRAMEWORK (SaFe)
- DEVOPS
- SPOTIFY/SQUADIFICATION
- MIKADO METHOD
- PROGRAMMER ANARCHY
- MOB PROGRAMMING

by www.lynnecazaly.com

scrum australia

flavor of

One Team - One Methodology

Declaration of Interdependence

Agile Leadership Network

- *David Anderson*
- Sanjiv Augustine
- Christopher Avery
- *Alistair Cockburn*
- *Mike Cohn*
- Doug DeCarlo
- Donna Fitzgerald
- *Jim Highsmith*
- Ole Jepsen
- Lowell Lindstrom
- Todd Little
- Kent McDonald
- Pollyanna Pixton
- Preston Smith
- Robert Wysocki

Declaration of interdependence

- Created in 2005
- Focuses on project management side of agile projects
- Aimed at leaders

pmdoi.org/ & www.agileleadershipnetwork.org/

Declaration of Interdependence

Agile and adaptive approaches for linking people, projects and value

We are a community of project leaders that are highly successful at delivering results. To achieve these results:

- We **increase return on investment** by making continuous flow of value our focus.
- We **deliver reliable results** by engaging customers in frequent interactions and shared ownership.
- We **expect uncertainty** and manage for it through iterations, anticipation, and adaptation.
- We **unleash creativity and innovation** by recognizing that individuals are the ultimate source of value, and creating an environment where they can make a difference.
- We **boost performance** through group accountability for results and shared responsibility for team effectiveness.
- We **improve effectiveness and reliability** through situationally specific strategies, processes and practices.

- We **increase return on investment** by making continuous flow of value our focus.

- Concentrate efforts on developing features that the business asks for
- When projects consistently deliver business results, they are hard to ignore or cancel

=> business are more likely to approve requests from your project

- We **deliver reliable results** by engaging customers in frequent interactions and shared ownership.

try to be more like

the good neighbor whom you see frequently and can easily call on

rather than

the intrusive relative who moves in for a while and then disappears for a year

- We **expect uncertainty** and manage for it through iterations, anticipation, and adaptation.

instead of

trying to create and follow a rigid plan that is likely to break,

it is better

to plan and develop in short chunks and adapt to changing requirements

- We **unleash creativity and innovation** by recognizing that individuals are the ultimate source of value, and creating an environment where they can make a difference.

- “We manage property and lead people; if you try to manage people, they feel like property”

- If you want the best results from people, provide the best environment

- We **boost performance** through group accountability for results and shared responsibility for team effectiveness.

- Empowered teams are:
  - Happier
  - More productive
  - More likely to take ownership of problems
  - Trying hard to solve problems

- We **improve effectiveness and reliability** through situationally specific strategies, processes and practices.

- There is no single cookbook for how to run successful projects
- We need to adjust our approach to best fit the project ingredients and the environment

Feature Driven Development (FDD)

Feature Driven Development (FDD)

- Iterative/incremental software process
- Developed in 1997 by Jeff De Luca
- Domain Model is the core of FDD
  (no specific values / principles defined)

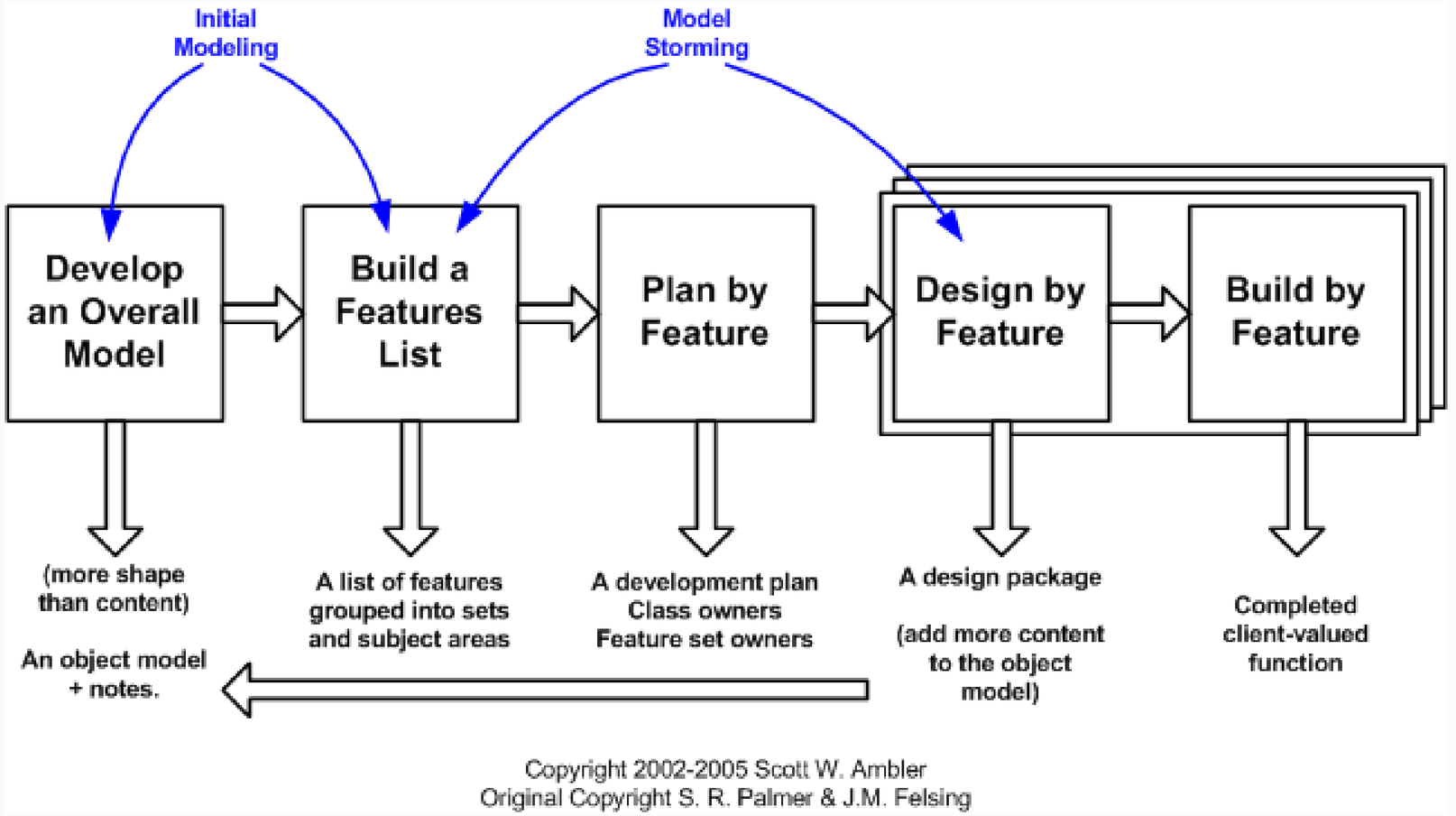
Feature Driven Development (FDD)

Requirements are gathered using a top-down approach

- Subject Areas (*general business practices*)
    - Feature Sets (*business activities*)
        - Features (*tasks*)

Typically 2 weeks iterations

Feature Driven Development (FDD)



Feature Driven Development (FDD)

Everything is

planned,

designed,

built,

managed

at the feature level.

Feature Driven Development (FDD)

- Formula for defining features:

  < action > < result > [of | to | for | from] < object >

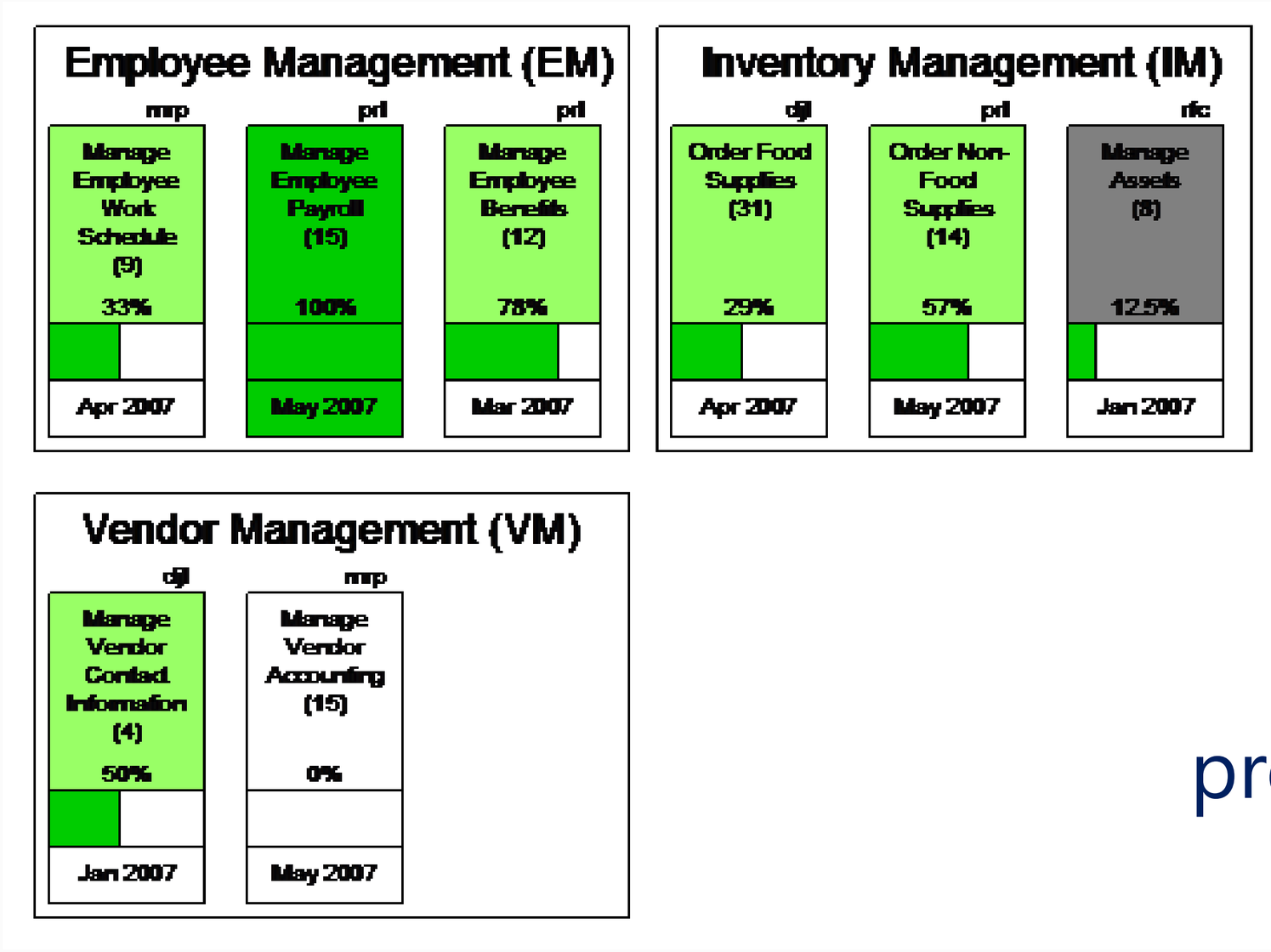
"Calculate monthly payment for car loan."

Feature Driven Development (FDD)

Roles:

- **Project Manager** –administrative, financial, reporting responsibilities
- **Chief Architect** –controls the design of the Domain Model manages the technical architecture, design sessions, and code reviews.
- **Development Manager** – manages daily development activities, coordinates the development team
- **Chief Programmer** –senior developer who is responsible for a specific Feature Set and manages their design and development activities.
- **Class Owner** –developer who reports to the CP and designs, codes, tests, and documents features
- **Domain Expert** – defines requirements as features that the solution must provide. Business analysts are the most common Des
- **Tester** – is responsible for validating that features perform as defined.
- **Deployer** – manages the data definitions and conversions and supports the deployment of code to the various platforms.
- **Technical Writer** – creates and maintains the documentation for users.

Feature Driven Development (FDD)



Feature set progress report

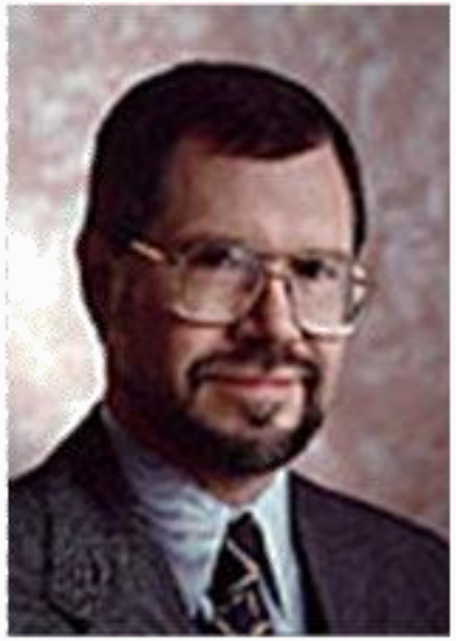
Feature Driven Development (FDD)

The exact state of each feature is documented in a table with six specific milestones

- Domain Walkthrough
- Design
- Design Inspection
- Code
- Code Inspection
- Promote to Build

Agile Unified Process

Agile Unified Process


"*Three Amigos*" ": Grady **Booch**, James **Rumbaugh**, Ivar **Jacobson**

Early 90's: Unified Modeling Language (UML)

Founders of Rational Software Corporation (today division of IBM)

Agile Unified Process

= simplified version of *Rational Unified Process*

(EssUP – *Essential Unified Process* – first attempt to simplify RUP by I. Jacobson)

- "high ceremony" framework
- based on integration of different agile concepts and techniques

Agile Unified Process

6 philosophies

- **Competence** – The team knows what it's doing. They won't read detailed process documentation, instead will apply high-level guidance and standards.
- **Simplicity** – Describe things concisely on a few pages, not reams of pages.
- **Agility** – Conforms to the values and principles of the Agile Alliance.
- **Activity** – Focus on only the high-value activities that count. Ignore the noise.
- **Tools** – Simple tools are often the best. Recommends using the tools best suited for the job.
- **Tailor** – AUP works best when tailored to the needs defined by the context.

Agile Unified Process

Phases

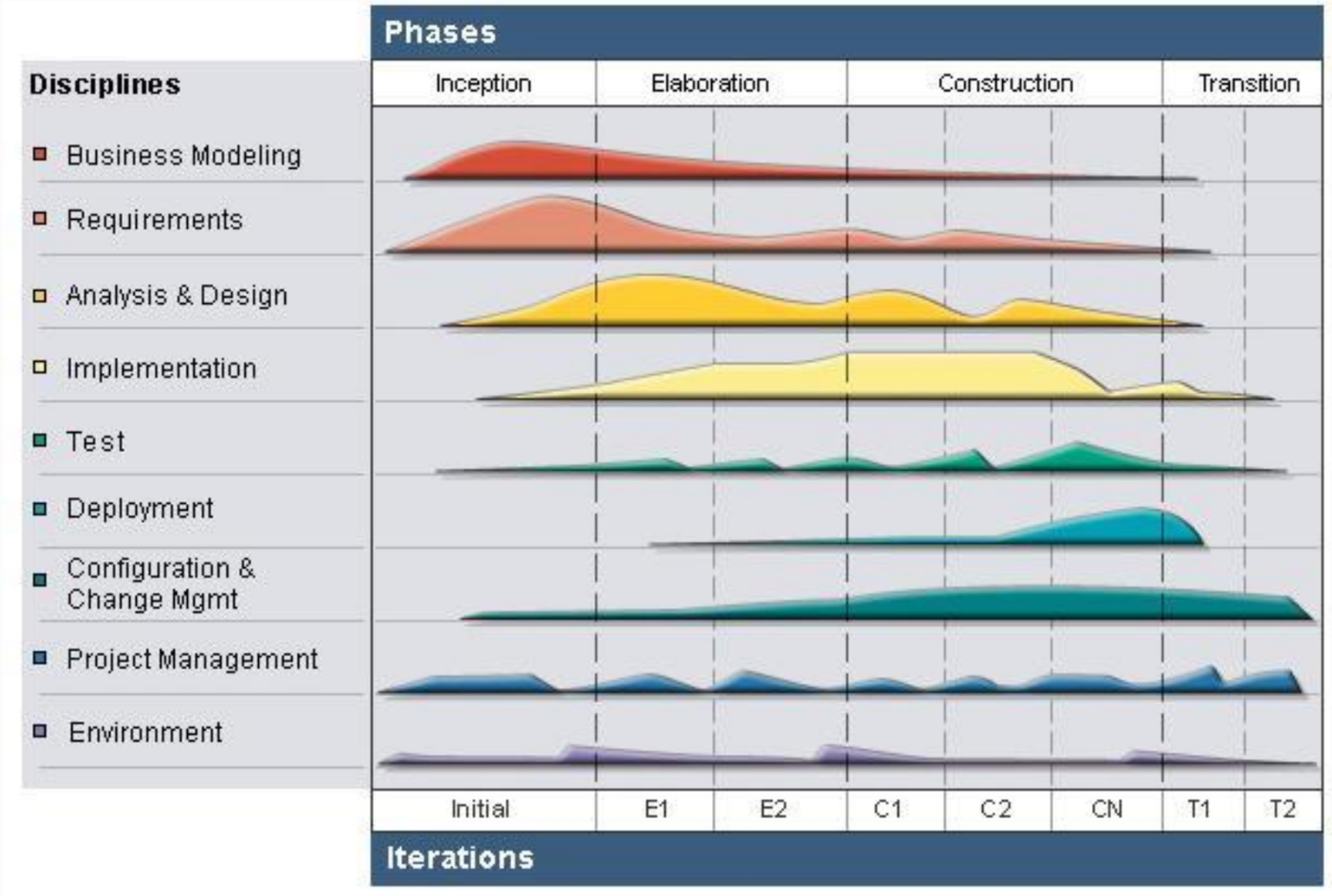
- **Inception** – cultivates a shared understanding of the project scope and defines architectural choices.
- **Elaboration** – develops the understanding of the system into requirements and validates architectural choices.
- **Construction** – occurs until system development is completed.
- **Transition** – all testing and system deployment to production.

Agile Unified Process

Disciplines

- **Model** – Use a model to represent the organization's business approach, the problem domain, and any viable solution to solve the problem.
- **Implement** – Code the model(s) into executable code and perform unit testing.
- **Test** – Apply additional tests to find defects, validate the system design works, verify the requirements are satisfied, and ensure code quality.
- **Deploy** – Plan and deliver the system for end users.
- **Configuration Management** – Control all project artifacts, including version tracking and change management.
- **Project Management** – Provide project management, including scope, resource, risk and progress management, and coordination of external interfaces, to achieve an on time, on budget completion.
- **Environment** – Provide process guidance standards and ensure needed tools are available for the team

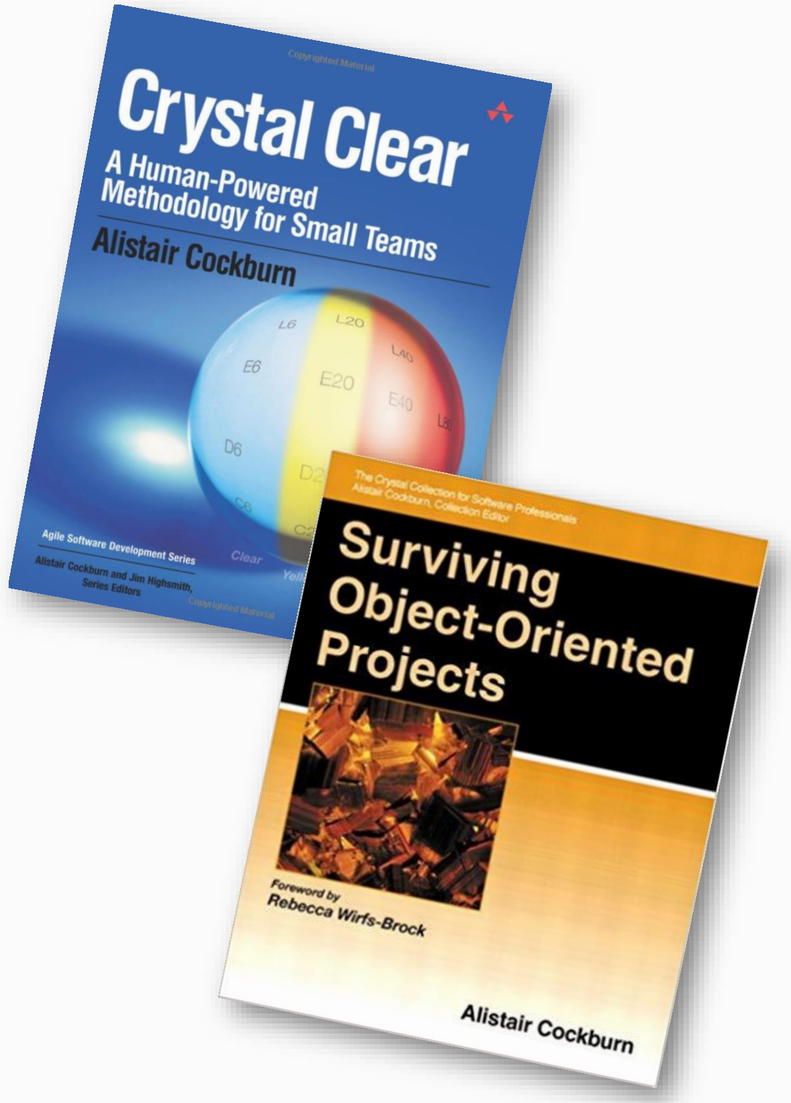
Agile Unified Process



Crystal

Crystal

- Alistair Cockburn, 2004
- Family of frameworks
  - based on size & criticality
  - not upward/downward compatible



Crystal

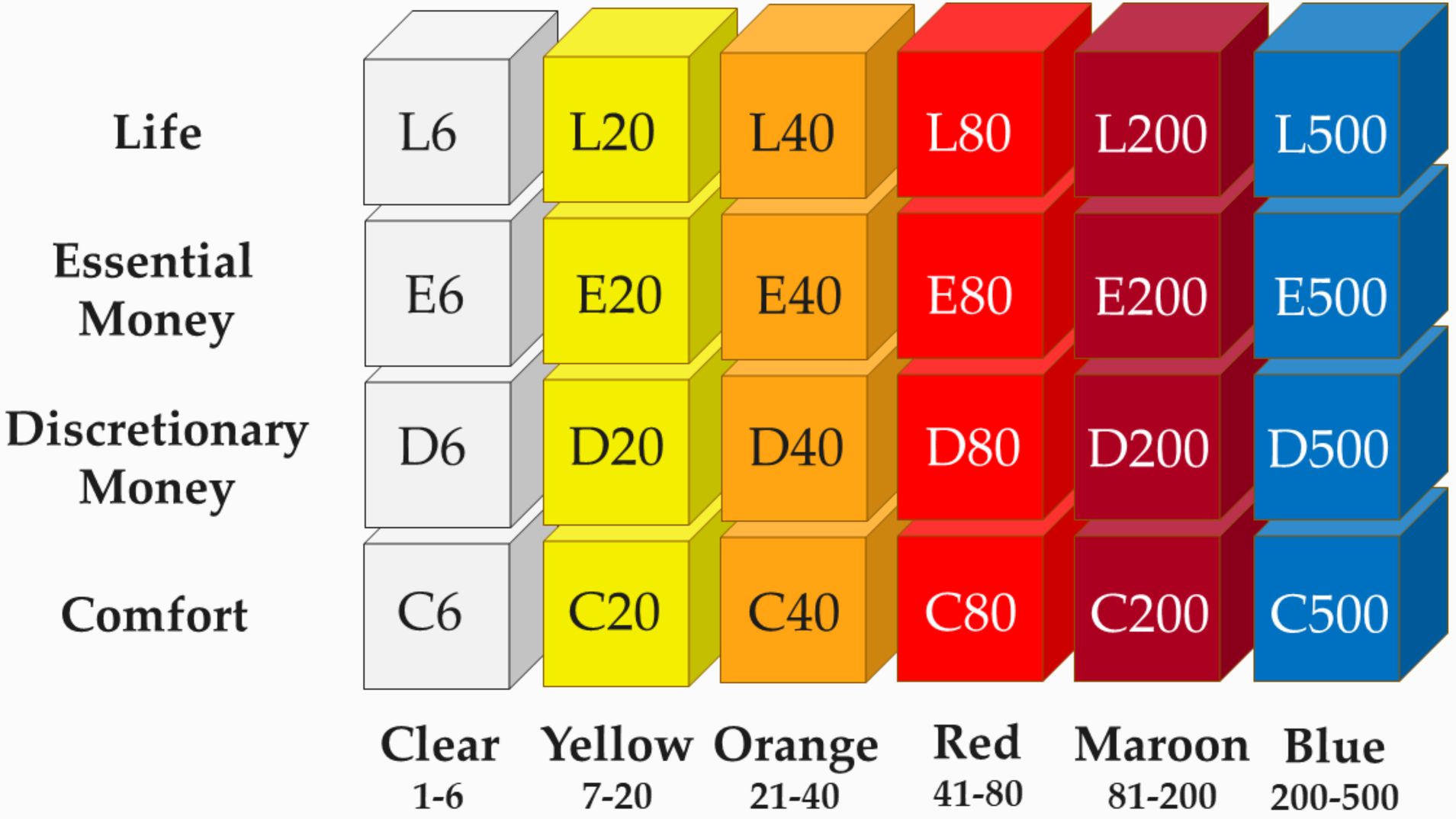
Key Principles

- **Frequent Delivery**: Project owners/customers can expect deliverables from the team(s) every couple of months.
- **Continual Feedback**: The entire project team & stakeholders meets on a regular basis to discuss project activities.
- **Constant Communication**: Teams co-located in the same room/facility. All projects expect to have frequent access to the person(s) defining the requirements.
- **Safety**: 1. The safe zone that team members must have to be effective and to communicate truth during the project.
  2. Evaluate how software projects affect the safety of their end-users.
- **Focus**: There should be enough time to complete priority items each without interruption.
- **Access to Users**: Project team will have access to one or more users of the system being built.
- **Automated Tests and Integration**: Controls must be put in place to support versioning, automated testing, and frequent integration of system components.

Crystal

- **Size**: number of people involved in the project.
  - Bigger size – more formality to the structure, artifacts and management
- **Criticality**: the potential for the system to cause damage
  - More critical: increase the rigidity of the project needs

Crystal


Life
Essential Money
Discretionary Money
Comfort
L6
L20
L40
L80
L200
L500
E6
E20
E40
E80
E200
E500
D6
D20
D40
D80
D200
D500
C6
C20
C40
C80
C200
C500
Clear
1-6
Yellow
7-20
Orange
21-40
Red
41-80
Maroon
81-200
Blue
200-500

Crystal Clear

- Has the fewest defined roles:
  - Sponsor
  - Senior Designer
  - Programmer
- Roles of *project manager*, *business analyst*, *tester*, etc. are shared among all team members.
- The expected release is every 60 or 90 days
- Minimal documentation (*project milestones*)

Crystal Orange

- Roles:
  - Sponsor
  - Project Manager
  - Business Analyst
  - Architect
  - Senior Designer
  - Programmer
  - Tester
- The expected release is every 90 or 120 days

Crystal Orange (cont)

- Specific Deliverables:
  - Requirements Document
  - Release Sequence (Schedule)
  - Project Schedule
  - Status Reports
  - UI Design Document (*if needed*)
  - Object Model
  - User Manual
  - Test Cases

Dynamic Systems Development Method DSDM

Dynamic Systems Development Method

- UK, 1990,
- DSDM Consortium (manages DSDM framework versions)
- Mot popular Agile methodology practice in UK
- Developed as an extension for RAD (Rapid Application Development)
- One of the *heavier* Agile approaches

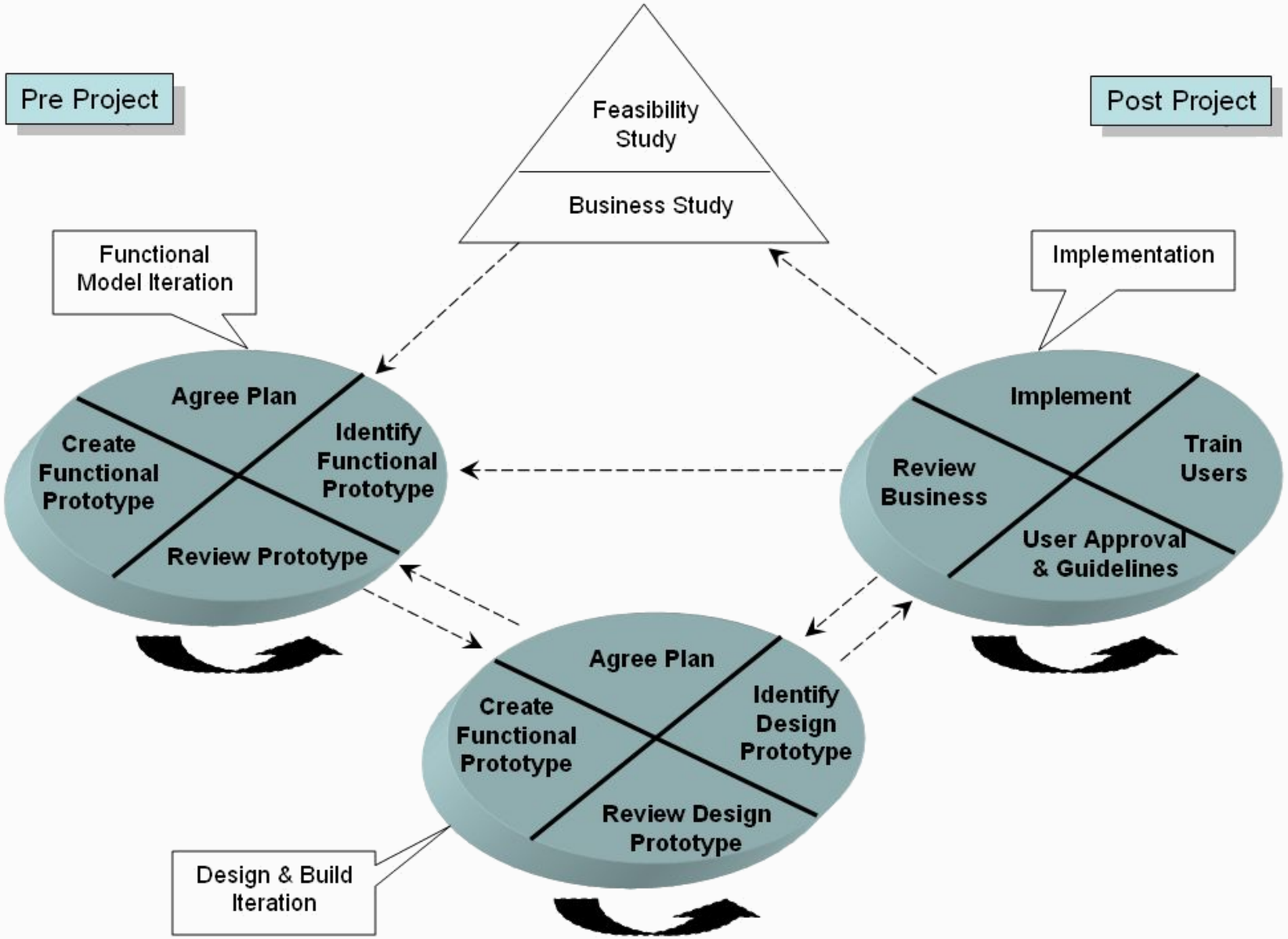
Dynamic Systems Development Method

- Principles
  - Focus on the business need
  - Deliver on time
  - Collaborate
  - Never compromise quality
  - Build incrementally from firm foundations
  - Develop iteratively
  - Communicate continuously and clearly
  - Demonstrate control

Dynamic Systems Development Method

- Phases:
  - **Pre Project**: Things that need to occur before the project begins.
  - **Project Lifecycle**: The actual project occurs. This phase is broken into five stages:
    - Feasibility Study
    - Business Study
    - Functional Model Iteration
    - Design and Build Iteration
    - Implementation
  - **Post Project**: Things that need to occur after the project has been completed.

Dynamic Systems Development Method



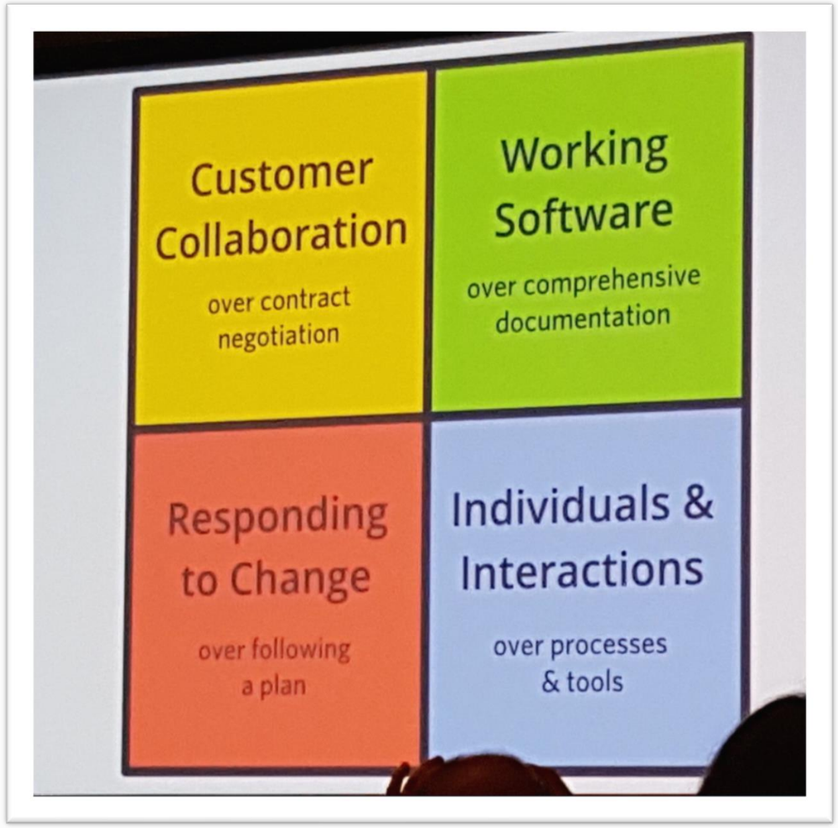
Modern Agile

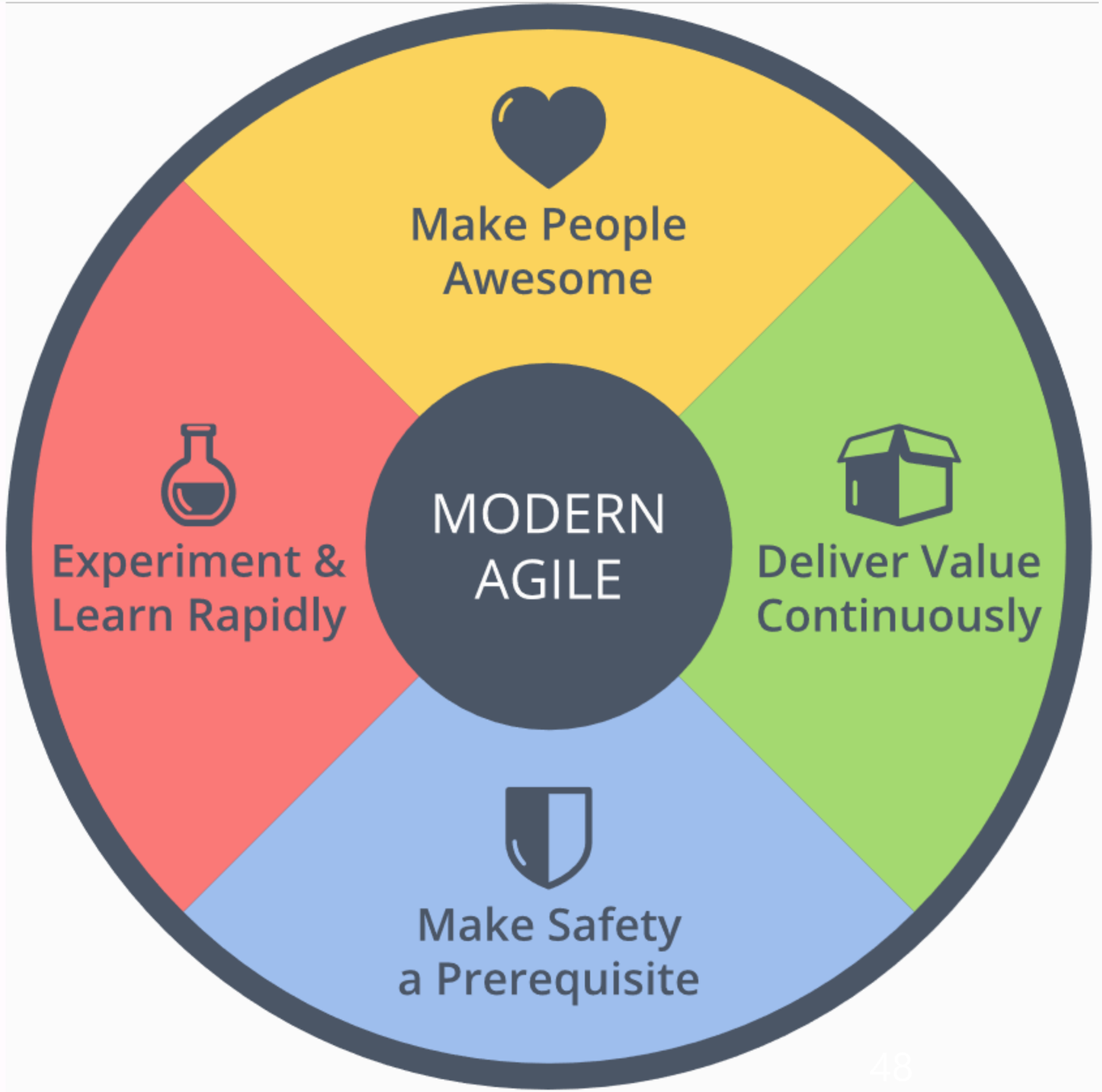
Modern Agile *is a community for people interested in uncovering better ways of getting awesome results. It leverages wisdom from many industries, is principle driven and framework free.*

Joshua Kerievsky, *CEO, Industrial Logic*

2015 – Joshua Kerievsky

modernagile.org/

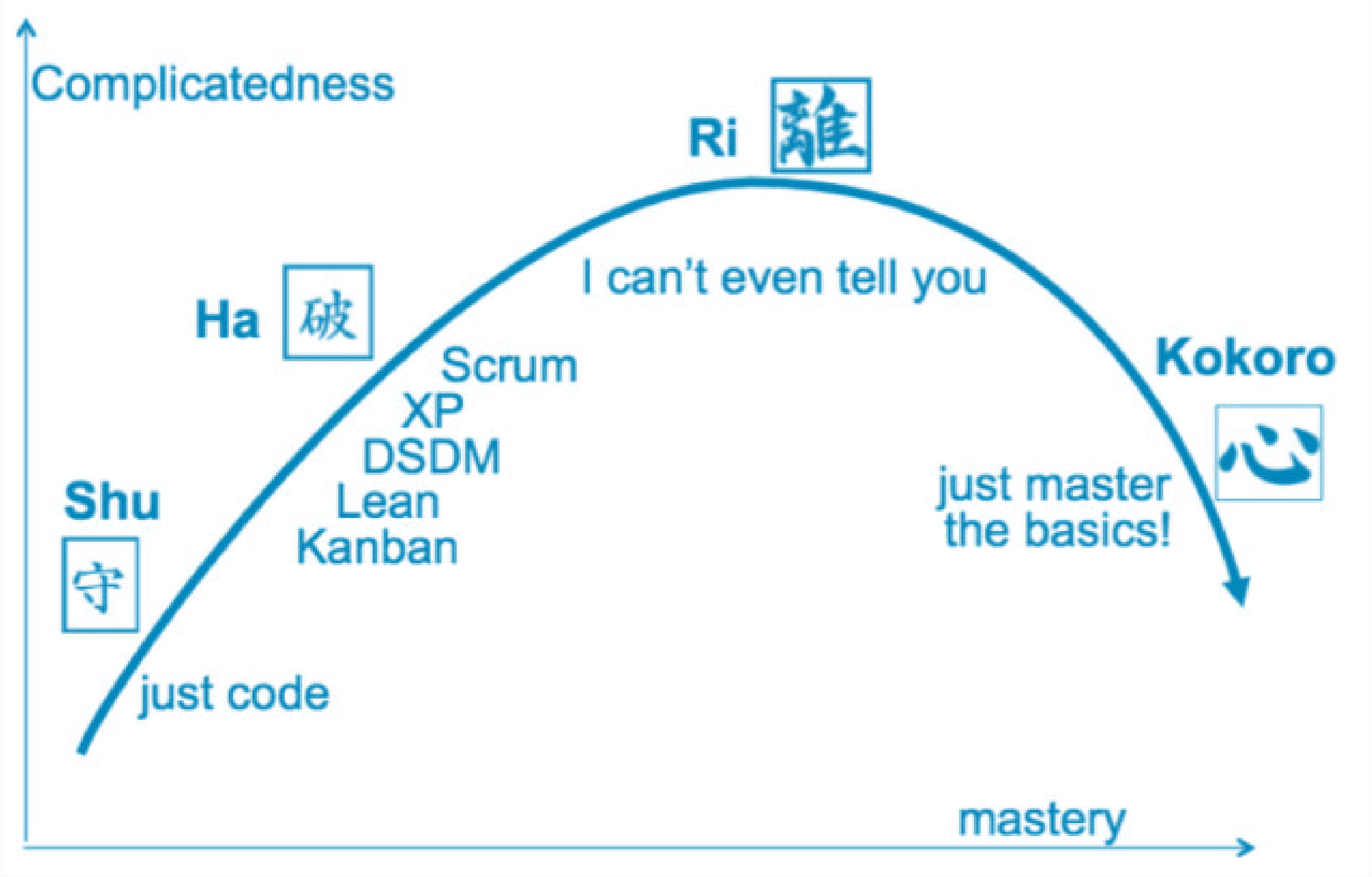




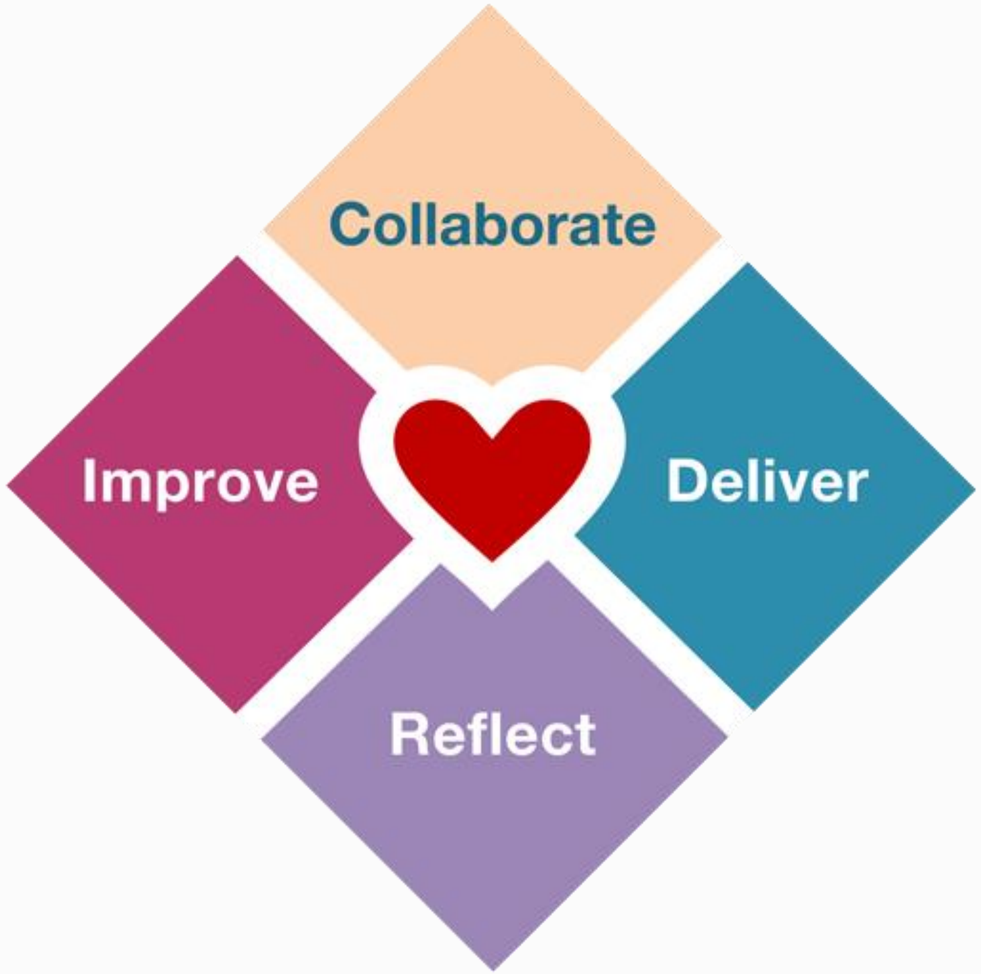
Modern Agile

The "*Oath of Non-Allegiance*"

"I promise not to exclude from consideration any idea based on its source, but to consider ideas across schools and heritages in order to find the ones that best suit the current situation."


Complicatedness
Ri
離
I can't even tell you
Ha
破
Scrum
XP
DSDM
Lean
Kanban
Kokoro
心
Shu
守
just master
the basics!
just code
mastery

The Heart of Agile



Agile Methodologies Strengths & Weaknesses

Agile Methodologies Strengths & Weaknesses

XP	• Strong technical practices. • Customer ownership of feature priority, developer ownership of estimates. • Frequent feedback opportunities. • Most widely known and adopted approach, at least in the U.S.	• Requires onsite customer. • Documentation primarily through verbal communication and code. For some teams these are the only artifacts created, others create minimal design and user documentation. • Difficult for new adopters to determine how to accommodate architectural and design concerns.

Agile Methodologies Strengths & Weaknesses

Scrum	• Self organizing teams and feedback. • Customer participation and steering. • Priorities based on business value. • Only approach here that has a certification process.	• Only provides project management support, other disciplines are out of scope. • Does not specify technical practices. • Can take some time to get the business to provide unique priorities for each requirement.

Agile Methodologies Strengths & Weaknesses

Lean	• Complements existing practices. • Focuses on project ROI. • Eliminates all project waste. • Cross-functional teams.	• Does not specify technical practices -> do something wrong efficiently. • Requires constant gathering of metrics which may be difficult for some environments to accommodate. • Theory of Constraints can be a complex and difficult aspect to adopt. •

Agile Methodologies Strengths & Weaknesses

FDD	• Supports multiple teams working in parallel. • All aspects of a project tracked by feature. • Design by feature and build by feature aspects are easy to understand and adopt. • Scales to large teams or projects well.	• Promotes individual code ownership as opposed to shared/team ownership. • Iterations are not as well defined by the process as other Agile methodologies. • The model-centric aspects can have huge impacts when working on existing systems that have no models.

Agile Methodologies Strengths & Weaknesses

AUP	• Robust methodology with many artifacts and disciplines to choose from. • Scales up very well. • Documentation helps communicate in distributed environments. • Priorities set based on highest risk. Risk can be a business or technical risk.	• Higher levels of ceremony may be a hindrance in smaller projects. • Minimal attention to team dynamics. • Documentation is much more formal than most approaches mentioned here.

Agile Methodologies Strengths & Weaknesses

Crystal	• Family of methodologies designed to scale by project size and criticality. • Only methodology that specifically accounts for life critical projects. • As project size grows, cross-functional teams are utilized to ensure consistency. • The “human” component has been considered for every aspect of the project support structure. • An emphasis on testing is so strong that at least one tester is expected to be on each project team.	• Expects all team members to be co-located. May not work well for distributed teams. • Adjustments are required from one project size/structure to another in order to follow the prescribed flavor of Crystal for that project size/criticality. • Moving from one flavor of Crystal to another in mid project doesn’t work, as Crystal was not designed to be upward or downward compatible.

Agile Methodologies Strengths & Weaknesses

DSDM	• An emphasis on testing is so strong that at least one tester is expected to be on each project team. • Designed from the ground up by business people, so business value is identified and expected to be the highest priority deliverable. • Has specific approach to determining how important each requirement is to an iteration. • Sets stakeholder expectations from the start of the project that not all requirements will make it into the final deliverable.	• Probably the most heavyweight project compared in this survey. • Expects continuous user involvement. • Defines several artifacts and work products for each phase of the project; heavier documentation. • Access to material is controlled by a Consortium, and fees may be charged just to access the reference material.