Choosing The Right Agile Methodology For Your Drupal Project

Prabhat Sinha

Shani Memfy

Speakers

Prabhat Sinha

He has been managing product and project since 8 years. After work Prabhat can be found jogging and socializing at local parks.

Prabhat lives in the bustling city New Delhi with his wife and 2 kids.

Shani Memfy

She's been managing product and project deliveries since 1999. After work you can find her on the court shooting hoops with a local Netball league.

Shani lives in a suburban city in Israel with her husband and 4 children.

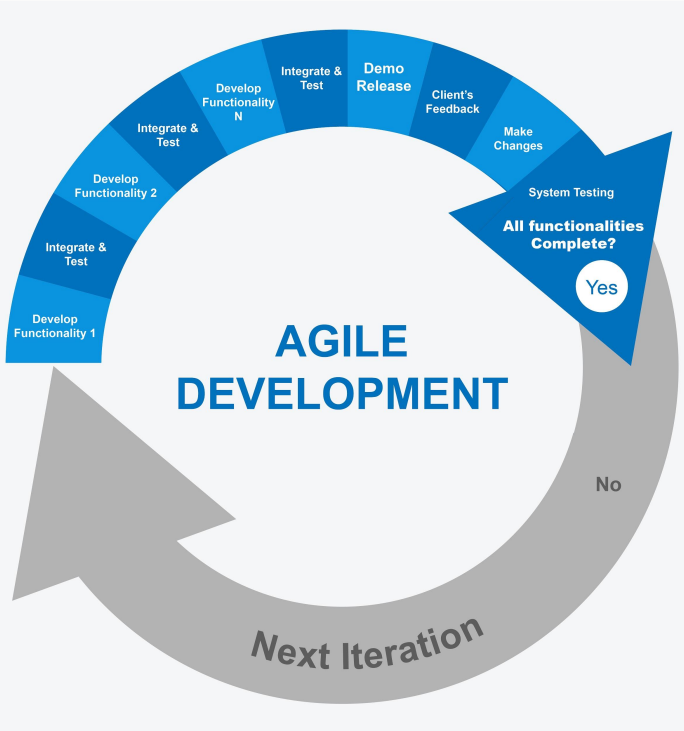
Agenda

1. What is Agile
2. Agile Frameworks
    - Scrum
    - KANBAN
    - Scrumban
    - Extreme Programming
    - Lean
    - Feature Driven development
3. Comparison
4. CYNEFIN

What is Agile?

Better ways of developing software is by doing it and helping others do it.
Agile gives importance to:

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



12 Principles of Agile Manifesto

01 Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

02 Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

03 Deliver working software frequently, from a couple of weeks lo a couple of months, with a preference to the shorter timescale.

04 Business people and developers must work together daily throughout the project.

05 Build projects around motivated individuals. Give them the environment & support they need, and trust them to get the job done.

06 Agile processes promote **sustainable** development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

07 Working software is the primary measure of progress.

08 The most efficient and effective method of Conveying information to and within a development team is face-to-face conversation.

09 Continuous attention to technical excellence and good design enhances agility.

10 Simplify--the art of maximizing the amount of work not done--is essential.

11 The best architectures, requirements & designs emerge from self-organizing teams.

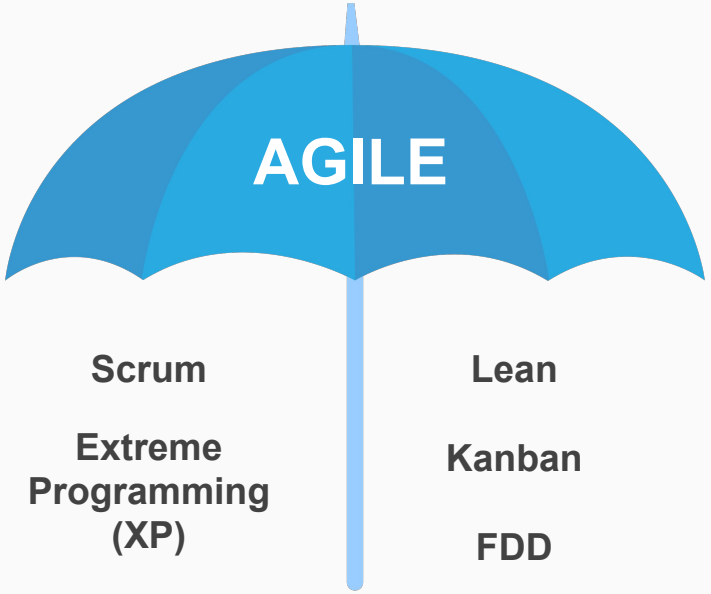
12 At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile Is a Methodology Only

There are several agile project management frameworks being successfully used for delivering Drupal projects. Let's evaluate few popular agile frameworks:

1. Scrum
2. Kanban
3. Scrumban
4. Extreme Programming
5. Lean Development
6. Feature driven development

In next few slides we will discuss each frameworks one by one and Help you choose the most fitting for your projects.



Scrum

Scrum is an Agile framework that brings order to chaos.

Scrum comes from Rugby:

An ordered formation of players in which the forwards of a team form up with arms interlocked and heads down, and push against a similar group from the opposing team.

An ordered formation of players, used to restart play, in which the forwards of a team form up with arms interlocked and heads down, and push against a similar group from the opposing team.

The cross functioning team

Sprint

PO, SM & the team

No distractions

Scrum doesn't work when the team is not working to a shared goal

Scrum in a Nutshell

Processes & Tools

- Sessions
  - Scrum Meeting (daily)
  - Backlog Refinement grooming (1 or 2 per Sprint)
  - Sprint planning 1 or 2 per Sprint)
  - Demo or Review (end of Sprint)
  - Retrospective (end of Sprint)

- Product Backlog
- Tools
  - Scrum Board
  - User Stories
  - Burn Down Chart
  - Timeboxing
  - Icebox

The Team

Product Owner, Scrummaster and Team

- Well-trained
- Specialized
- Capable of Self-management
- Communicative
- Ability to Make Decisions
- Common Goal
- Self Improving

The project requirements

- Clear vision of the end product
- A clear set of requirements (user stories)
  - Product Backlog at least for the upcoming 2 sprints

- Objective to add small marketable values in each increment
  - PSP: potentially shippable product
  - MMF: minimal marketable feature

When to choose Scrum

Tools:

- A clear vision of the end product
  - There needs to be enough of a product backlog for a few sprints (2-3)
    - There can be changes but the team shouldn't stop working to adapt to change
  - Scrum projects can change direction as long as the product backlog is maintained and groomed

Team:

- The WHOLE scrum team is onboard and committed
- Stakeholders have the project as a priority—not necessarily a top priority, but available for answers
- Team is focused: don't get many interruptions from everyday business
- The team is fully skilled team to match the project
- The team has the ability to communicate frequently and easily (daily stand ups, etc.)

Output:

- Something to show at the end of the sprint
- The team gets a full sprint to produce value without any interruptions
- Anything after that may be subject to change, this is where a sprint really lives up to its name

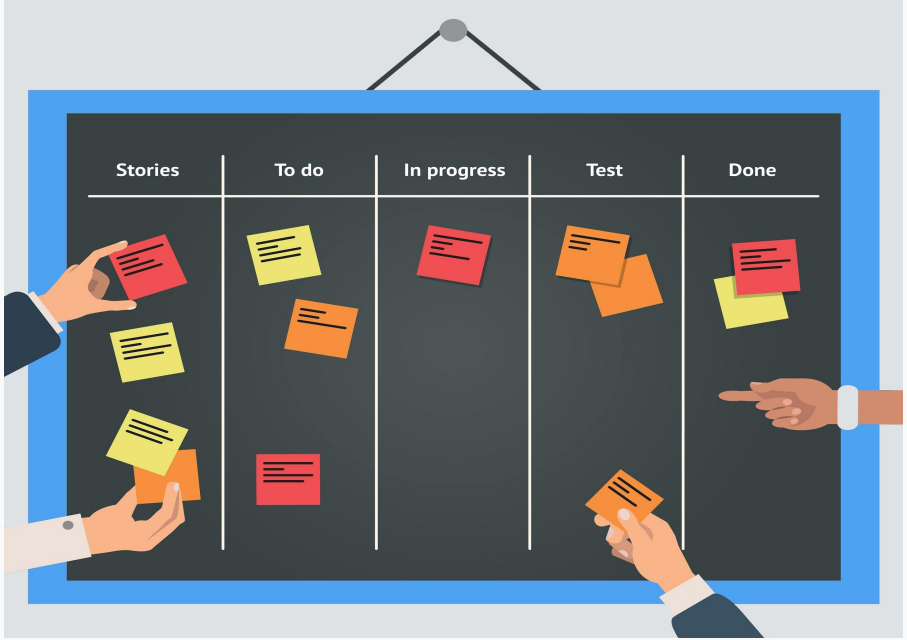
When should Scrum turn to Kanban and Scrumban

- Sprint scope frequently changed due to “high priority”/ emergency situations
- Scope isn’t clear or is blocked
- Basic Scrum rules not fully enforce
- Releases start to be ad hoc
- No clear distinction between Project and Product
- Addressing maintenance and support work as main sprint goal
- Team dedication is erratic

KANBAN

Kanban is a visual signal that's used to trigger an action. The word **kanban** is **Japanese** and roughly translated means: “card you can see.” Toyota introduced and refined the use of **kanban** in a relay system to standardize the flow of parts in their just-in-time (JIT) production lines in the 1950s.

"Kanban promotes flow and reduced cycle-time by limiting WIP and pulling value through in a visible manner."
-Torbjörn Gyllebring



KanBan in a Nutshell

Processes & Tools

- Sessions
  - Meeting (daily)
    - Often those that have converted from Scrum to Kanban,are inclined to keep the daily stand-up
- Product Backlog
  - Team to **pull** the tickets
- Tools
  - Kanban board
  - **WIP limits**
    - Limiting the amount of WIP, at each step in the process, prevents overproduction and reveals **bottlenecks** dynamically so that you can address them before they get out of hand.

The Team

- Technical needs as per requirement of the project

Kanban recognizes that there may be value in the existing process, roles and responsibilities

Take what is working and preserve it.

The project philosophy

- Kanban doesn’t prohibit change, but it doesn’t prescribe it either.
- Kanban encourages making incremental changes to avoid drastic decrease in productivity
- Small course corrections are also just easier than altering the complete process
- Vision of the end product

When to choose Kanban

- Operation support
  - Running a production system as a whole,
- Maintenance work and support projects
- Continuous flow
  - No nee0d to stop to reassess just keep on going and deploying
- When the project requires the maximum flexibility and frequent change of priorities and scope
  - When the goals are not clearly defined
  - Risk of scope creep
- Ad hoc releases required
- Team dedication is erratic
  - When team isn't focused - responsibilities elsewhere.
- If management isn't 'into scrum'
  - Management isn't giving the needed time and attention to the project

Focus

Courage

Openness

Respect

Commitment

SCRUMBAN

Mix of scrum and Kanban

Scrumban is a management framework that emerges when teams employ Scrum as their chosen way of working to understand and continuously improve how they work and use the Kanban Method as a lens through which to view (kanban board)

- Scrumban comes into play to help teams with the transition from Scrum to Kanban. Can stay here a while
-
- Scrum-ban is still unclear. Is it an improved Scrum or and improved Kanban
- Scrumban is not unique to the software development process

ScrumBan in a Nutshell

Processes & Tools

- Sessions (The Scrum)
  - Stand up for daily check in
  - Retro for team improvement
  - Review- for incremental releases to demo to the client
- Product Backlog

- Tools (The Ban)
  - Kanban board
  - Ad hoc release
  - Continuous flow
  - WIP limits
  - Pulling tickets
  - Grooming on demand

The Team

Like Scrum

- Cross functional
- Allowing for specialized teams and functions
- Dedicated team

The project philosophy

Improvement on;

- Roles
- Work processes
- Better flows
- Flexibility

When to choose Scrumban

- Any Scrum with an unstable and/or moving vision
  - When the project requires the maximum flexibility and frequent change of priorities and scope
  - When the goals are not clearly defined
  - Constantly evolving product
  - Sprint planning isn’t happening
- If management isn’t ‘into scrum’
  - Management isn’t giving the needed time and attention to the project
- Any Kanban project that needs the structure of Scrum
  - Hard deadline
    - Still working to the objective
  - Team not focused

Extreme Programming (XP)

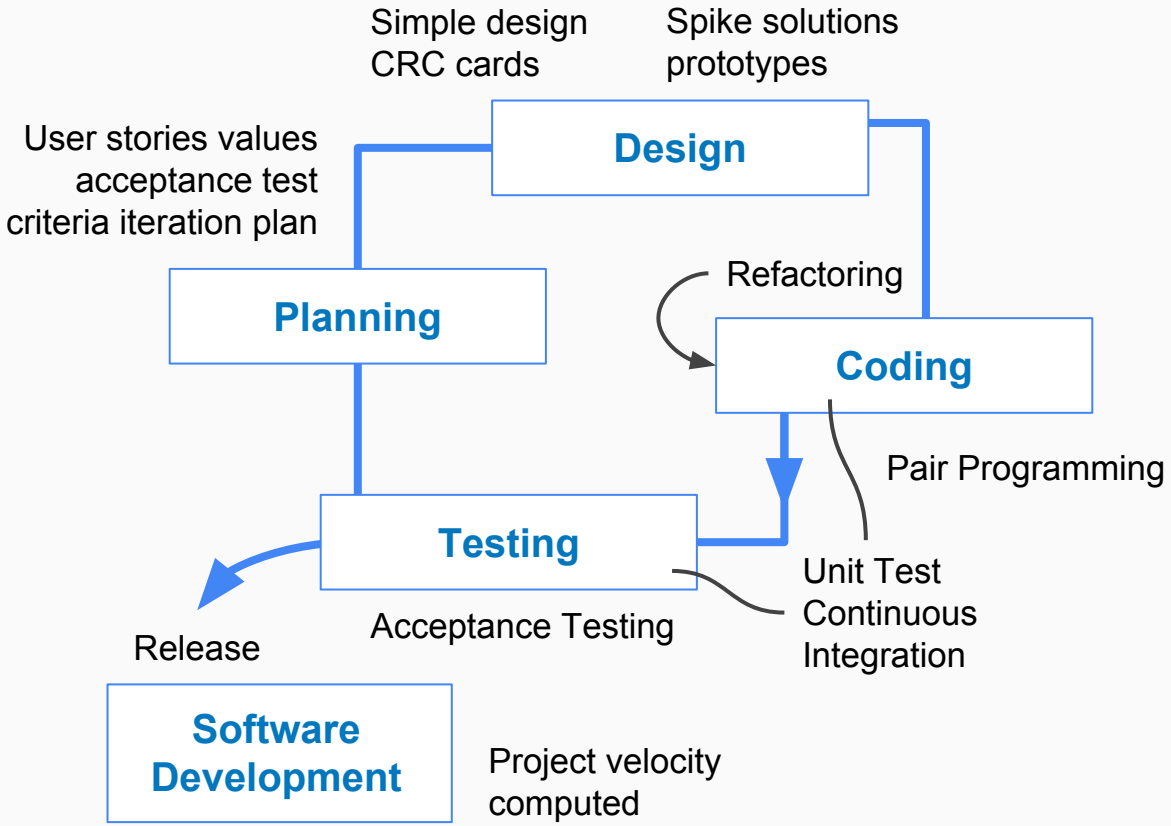
Extreme Programming (XP) is another agile development framework, focussed on improving software quality and responsiveness to evolving client requirements. Extreme programming advocates frequent releases in short development cycles in order to improve productivity and introduce checkpoints so that new requirements can be accommodated.

Planning Game **Small Releases** **Metaphor**	**Simple Design** **Testing** **Refactoring**	**Pair Programming** **Integration** **Collective Ownership**	**On–site Customer** **40–Hour Work Week** **Coding Standards**

Other elements of extreme programming are: programming in **pairs, code reviews, unit testing** of all codes, avoiding programming of features—until they are actually needed, a flat management structure, simplicity, and clarity in code.

Extreme Programming XP (Prabhat)

- Extreme programming is code first approach to software delivery and emphasizes on four basic activities: coding, testing, listening, designing.
- Extreme programming brought testing at the forefront of the delivery process which helped with evolution of core software engineering.
- Practices like: automated testing, refactoring, continuous integration, and test driven development.



Advantages & Disadvantages of Extreme Programming

Advantages

- Pair programming under XP helps in writing better codes
- Increased team accountability
- Extreme Programming manages risks in a better way
- Source code is always robust as simplicity helps in faster development and less defects
- Easy to accommodate changes

Disadvantages

Detailed planning is required right since the inception of XP due to changing scope and associated cost

- XP does not have a set measurement plan or quality assurance for coding.
- Pair programming may lead to too much duplication of codes and data.
- XP is more code centric than design which may cause UI/UX issues in larger projects.

XP in a Nutshell

Processes & Tools

- Pair programming
  - Planning game
- Release Plan
  - Iteration plan
- Project velocity

- **Iteration** - Usually 1 week long

Practices

- Test driven development
- Continuous Integration
- Collective code ownership

The Team

Team Size - 5 or less

- Tracker, Customer, Programmer, Tester, Coach
- Well-trained
- Specialized
- **Co-located**

The project philosophy

- Code first
- Refactoring
- Simple designs
- Spike solutions

When to use XP

When to use

- Have reached a certain level of maturity as usual tasks, defects and smaller unrelated user stories
- Require maximum flexibility and frequent change of priorities
- Have multiple releases per week or per day
- Have many unscheduled releases
- Have less cross functional teams

Maturity

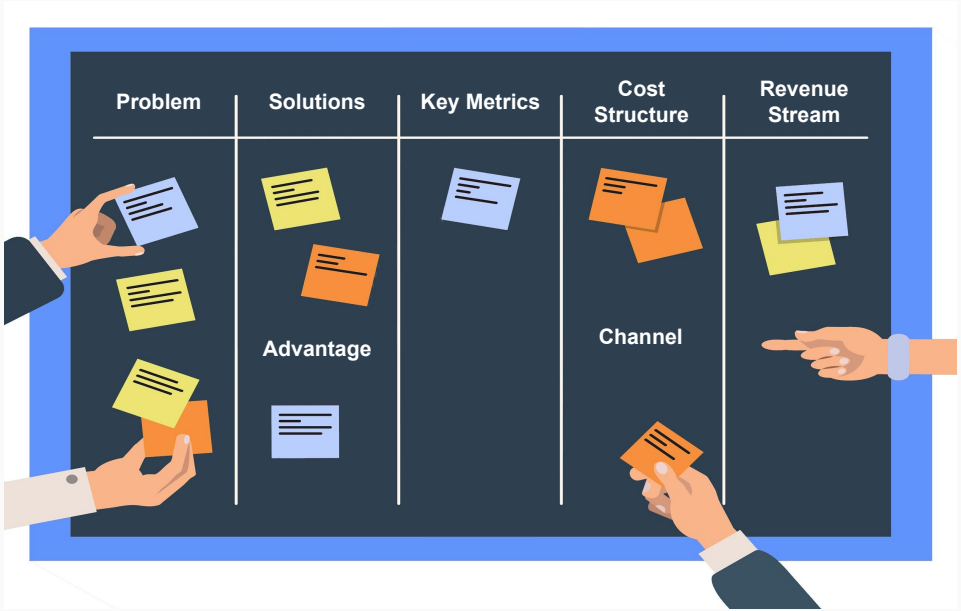
Flexibility

Unscheduled Releases

Less cross functional team

Lean Development

- Lean was adopted for the software industry in the year 2000
- Later on lean was applied for startups in a book “Lean Startups” by Eric Reis as a way of developing new product and services in circumstances of extreme uncertainty.
- A typical lean process: Learn-Measure-Build cycle, performs quality analysis, and testing, frequently connects with clients to understand the business value and focuses on continuous improvement
- The continuous cycle leads to sustainability, smart development, and success



Advantages and Disadvantages of LEAN

Advantages

- Complements existing practices.
- Focuses on project ROI.
- Eliminates all project waste.
- Cross-functional teams.

Disadvantages

- Does not specify technical practices.
- 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.

Lean in a Nutshell

Processes and Tools

- Sessions
  - Daily Stand up meetings
  - Operations Review
- Tools
  - Cumulative Flow Diagrams
  - Visual Controls
  - Virtual Kanban Systems
- Iterations
  - Small Batch sizes
  - Automation

The Team

- Well-trained
- Specialized
- capable of self-management
- Communicative
- Ability to make decisions

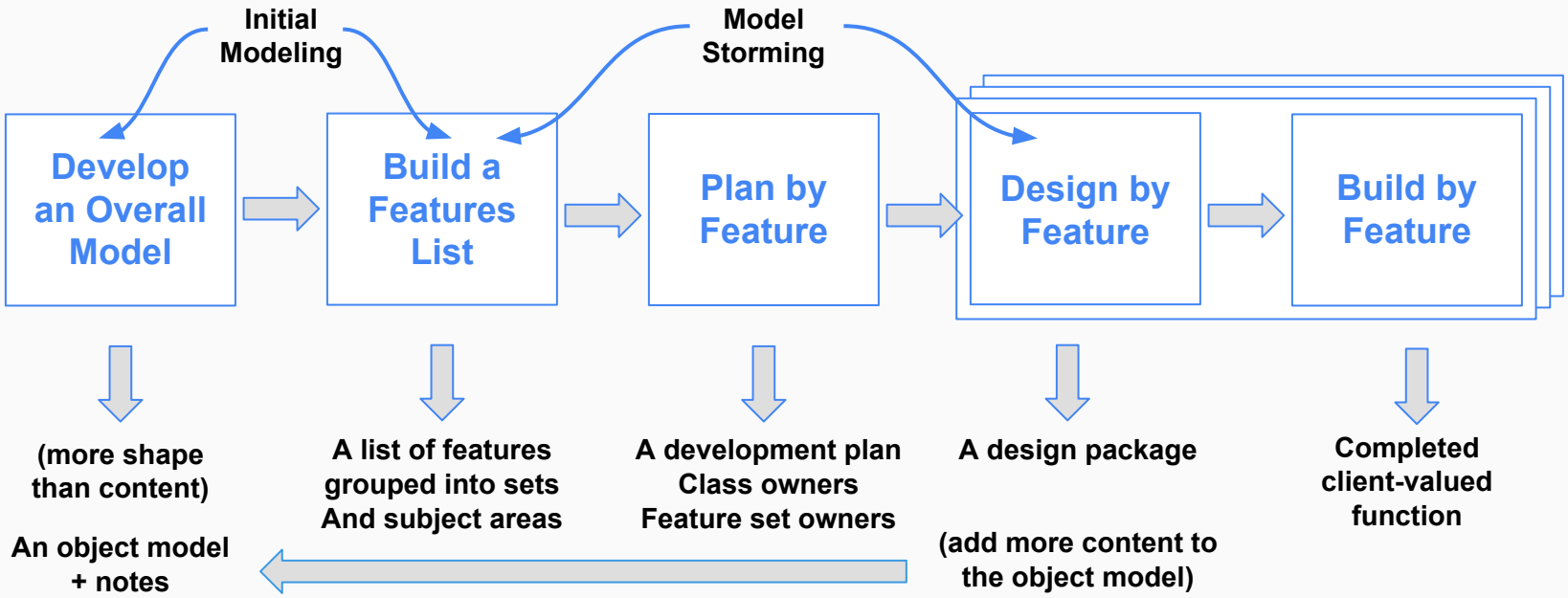
Key Principles

1. Eliminate Waste
2. Build Quality “in”
3. Create Knowledge
4. Defer Commitment
5. Deliver Fast
6. Respect People
7. Optimize the Whole

Feature Driven Development

- Feature Driven Development is a pragmatic software process which is architecture centric and focused to achieve the goal of client.
- Feature Driven Development was introduced in 1999 in a book named **“Java modeling in color with UML.”**
- As the name suggests, features are an important aspect of FDD. These are the primary source of requirements and primary input into the planning efforts in FDD.
- A feature is a small, client-valued function expressed in the form: <action><result><object>.

Feature Driven Development



Advantages & Disadvantages of Feature Driven Development

Advantages

- 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

Disadvantages

- Promotes individual code ownership as opposed to shared 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

FDD in a Nutshell

Processes & Tools

- Develop an overall model
- Build a feature list
- Plan by feature
- Design by feature
- Build by feature

Practices

- Domain Object Modeling
- High level class diagrams
- Developing by feature
- Individual class ownership
- Reporting visibility of reports

The Team

- Project Manager
- Chief Architect
- Development Manager
- Chief Programmer
- Class Owner
- Domain Expert

Supporting roles

- Domain manager
- Release Manager
- Language Guru
- Build Engineer
- Toolsmith
- System Administrator
- Tester
- Deployer
- Technical Writer

The project philosophy

- Multiple teams working on the same project.
- Tracking through features
- Model Centric Approach

Comparison table

	Scrum	Kanban	Scrumban
Artifacts	simple board , product backlog, sprint backlog, product increment, burndown chart	mapped on the process board	mapped on the process board
Board	Clean board for start of sprint	same board continually used	same board continually used
Ceremonies	daily scrum, sprint planning sprint review sprint retrospective	on demand and optional	Defined by the team
Teams & Roles	SM, PO, Team, Small to Medium	Cross functional / specialized teams with a possibility Agile coach, Small to Medium	Cross functional / specialized teams with a possibility Agile coach, Small to Medium
Iterations	Incremental Improvements	Constant Stream (not have predefined)	Incremental Improvements OR Constant Stream
Task Assignment	Assigned to team	Pulled by the team member	Taken by each team member
prioritization	Part of backlog grooming, done by PO	Out of the process. Backlog should be prioritized	Out of the process. Backlog should be prioritized
performance metrics	Burn down chart	lead and cycle time and cumulative flow	lead and cycle time and avg cycle time

Comparison table

	Extreme Programming	FDD	Lean Development
Artifacts	Release Plan, Metaphor, Iteration plan	Features, high level class diagrams	Cumulative Flow Diagram, Virtual Kanban Board
Board	Clean board for start of sprint	Feature board	Same board continually used
Ceremonies	daily meeting, planning game on demand	on demand and optional	Daily Stand up and operation review
Teams & Roles	Tracker, Customer, Programmer Coach	Project Manager, Chief Manager, Architect, Class owner and so on	cross functional / specialized Agile coach
Iterations	Incremental Improvements	Feature Development	Incremental Improvements OR Constant Stream
Task Assignment	Assigned by the customer	Assigned to the team member	Pulled by the team member
prioritization	Done by Customer	Done by Project Manager or Architect	Done based on the business value
performance metrics	Release plan	lead and cycle time and cumulative flow	lead and cycle time and avg cycle time

Which Methodology is right for you

We know about all the methodologies, advantages and disadvantages now.

Now comes the question - which methodology is right for you?

Let’s use popular decision making framework CYNEFIN for it.

CYNEFIN

CYNEFIN describes the **problem, situation, and systems**—with the help of research—into adaptive system theory, cognitive science, narrative patterns, and evolutionary psychology

Further it explores relationship between **context, experience, and the person** to propose new approaches to communication, decision-making, policy making, and knowledge management

- The term CYNEFIN was first coined by Welsh Scholar “Dave Snowden”
- George W. Bush administration used CYNEFIN for analysing policy and the impact of religion in process
- CYNEFIN provides a typology of contexts that guides what kind of explanation or solution may apply
- CYNEFIN has 5 domains: **obvious, complicated, complex, chaotic, disorder**

CYNEFIN

Complex

Enabling Constraints Loosely Coupled

probe–sense–respond

Emergent Practice

Complicated

Governing Constraints Tightly Coupled

sense–analyse–respond

Good Practice

Chaotic

Lacking Constraint De–Coupled

act–sense–respond

Novel Practice

Obvious

Tightly Constrained No Degrees of Freedom

sense–categorise–respond

Best Practice

Obvious

Sense, Categorize, Respond

- No analysis required
- Cause and effect repeatable, known and predictable
- We use best practices for it
- Need to follow standard operating procedures
- Automation can also do the job
- Data provides answers

Complicated

Sense, Analyze, Respond

- Cause and effect separated over time and space
- We use our good practices
- We do predictive planning and expert analysis to reach the solution
- Data provides options, experts analyze and measure goodness

Complex

Probe-Sense-Respond

- Cause and effect seen in retrospect and do not repeat
- We use emergent practices for it
- We do sensemaking, stories and monitor coherence to find the solution
- We need to do pattern management and heuristics.
  - For example: more stories like this, less like this

Chaos

Act, Sense, Respond

- Cause and effect not usefully perceivable
- Act to bring stability and crises management
- Experience informs decision, action is required
- Noble practices are discovered

Q & A

Thank you!!!