Understand Key Agile Frameworks: Learn Essentials of Scrum, Kanban, and more.

Agile methodologies emphasize adaptability, collaboration, and continuous improvement in software development.

Various Agile frameworks exist beyond Scrum, offering tailored approaches for different team structures and project scopes. Familiarity with these frameworks helps in selecting the best fit for a project.

BRIEF OVERVIEW OF COMMON AGILE FRAMEWORKS

SCRUM, KANBAN, LEAN, DSMD, XP, FDD, CRYSTAL

KEY AGILE FRAMEWORKS

1) **Scrum**: A highly structured framework designed to manage complex projects, emphasizing teamwork, accountability, and iterative progress.

2) **Kanban**: Visualizes workflow, allowing teams to optimize flow and reduce waste by focusing on continuous delivery.

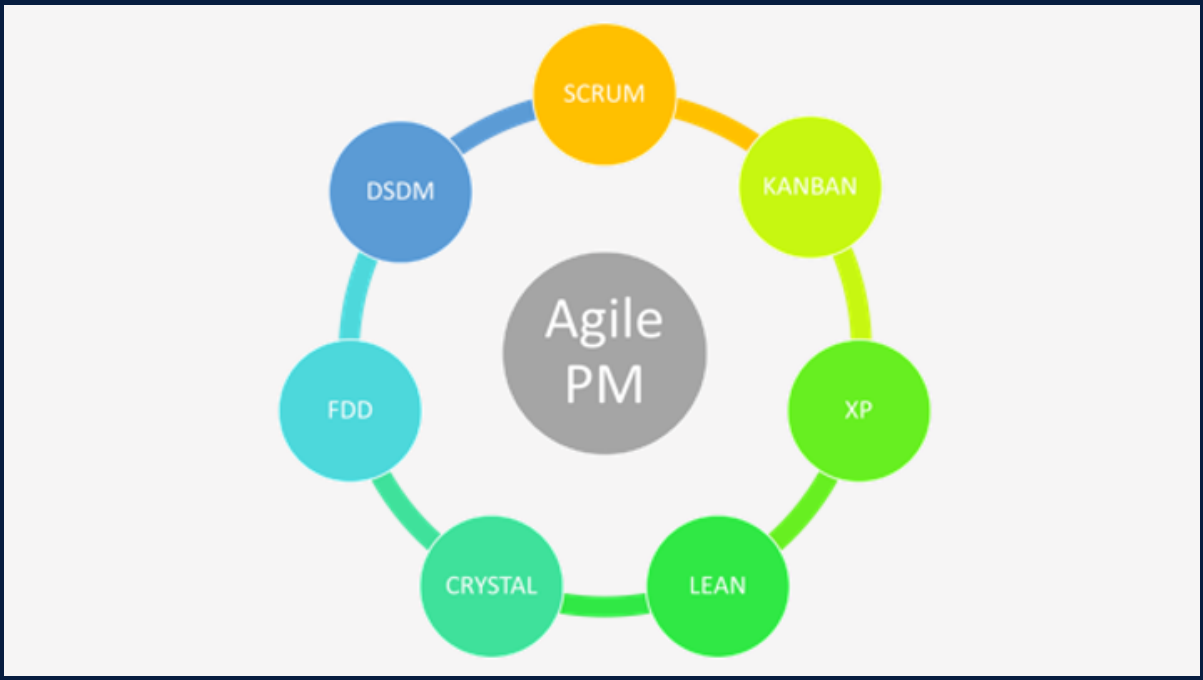
3) **Lean**: Rooted in manufacturing, this approach aims to enhance efficiency by minimizing waste and streamlining| processes.

4) **DSDM** (Dynamic Systems Development Method): A robust Agile project delivery framework that ensures all aspects of a project align with business goals.

5) **XP** (Extreme Programming): Prioritizes customer satisfaction, emphasizing flexibility to adapt to changing requirements.

6) **FDD** (Feature Driven Development): Focuses on building features, with a detailed modeling phase to capture and analyze requirements.

7) **Crystal**: A family of Agile methodologies emphasizing collaboration and adaptability. It tailors processes to team size and project needs.



KANBAN

KANBAN

Kanban, a term that originates from Japanese, translates to "signboard" or "visual board."

Kanban emphasizes visualization of work processes to enhance transparency and efficiency.

The framework employs visual tools, most prominently the "Kanban Board", to manage and track the progress of tasks.

KEY PRINCIPLES:

1) **Visualize Work:** Display tasks visually to provide an at-a-glance understanding of work status.

2) **Limit Work in Progress (WIP)**: By setting clear boundaries on how many tasks can be in a particular stage, teams prevent over-commitment and maintain a sustainable pace.

3) **Flow Management:** As tasks near completion, ensure the smooth transition of new tasks into the workflow, promoting a steady flow.

APPLICATIONS OF KANBAN

Kanban is a highly versatile framework used in various domains and contexts, not limited to software development.

Its primary application is to improve workflow efficiency and transparency.

In essence, any process or task that can benefit from visualization, flow optimization, and continuous improvement can leverage the principles of Kanban.

SCRUM

Scrum stands as one of the most widely adopted Agile frameworks.

Distinct from Kanban, Scrum focuses on **iterative** development, dividing projects into short phases termed as "Sprints."

CORE ELEMENTS

1) Roles:

- **Product Owner:** Represents stakeholders and manages the product backlog.
- **Scrum Master:** Ensures the team follows Scrum practices and removes impediments.
- **Developers:** Professionals who do the work of delivering a potentially releasable increment of the product at the end of each Sprint.

CORE ELEMENTS

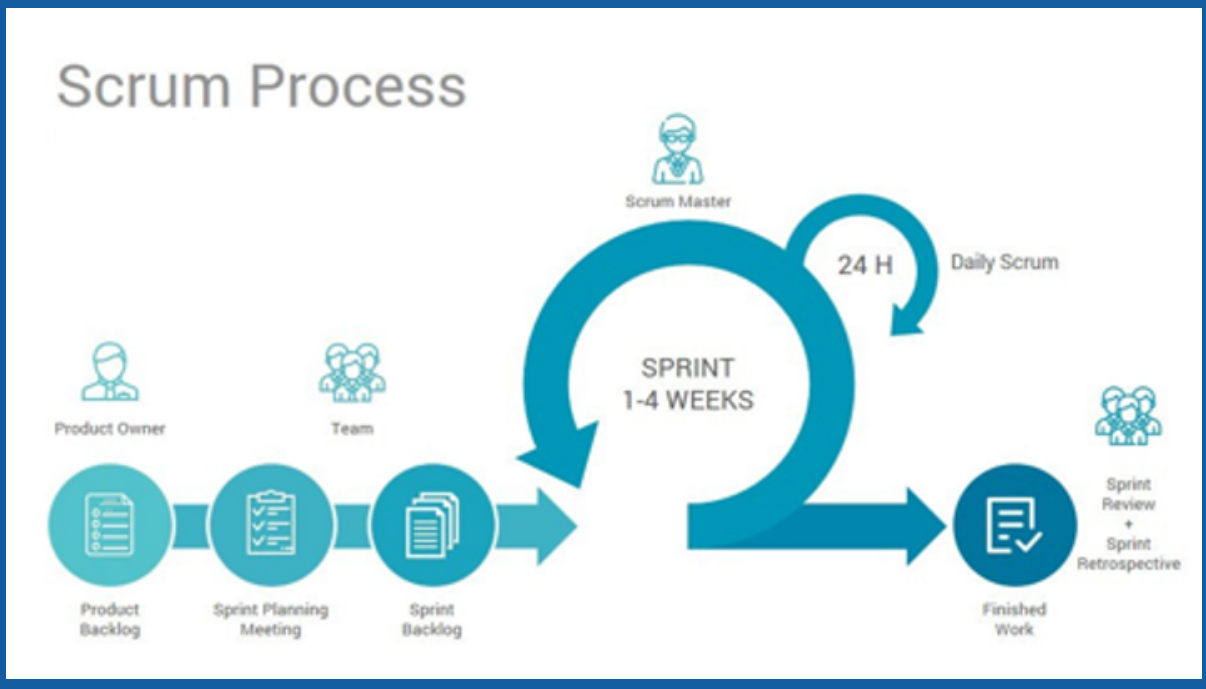
2) Artifacts:

- **Product Backlog:** A prioritized wish list maintained by the Product Owner.
- **Sprint Backlog**: A set of items from the Product Backlog selected for the Sprint, along with a plan to deliver them.
- **Increment:** The sum of all items completed during a Sprint combined with the value of increments from previous Sprints.

CORE ELEMENTS

3) Ceremonies:

- **Daily Scrum:** A daily meeting where the team inspects its progress and plans the work for the day.
- **Sprint Review**: Held at the end of the Sprint to inspect the increment and adapt the Product Backlog.
- **Sprint Retrospective**: Reflects on the past Sprint and identifies improvements for the next Sprint.



APPLICATION

Scrum is embraced by cross-functional teams focusing on product development.

With its iterative nature, work is divided into 2-4 week periods, or Sprints.

The framework's adaptability means teams can integrate practices from other methodologies as needed, continually refining their approach based on real-world feedback.

LEAN

BACKGROUND & ORIGIN:

Lean is distinct from Agile, but there are some philosophical overlaps.

This methodology, originally known as **"Lean Manufacturing,"** dates back to the twentieth century in Japan, with the goal of ensuring efficiency and value in production systems.

5 Principles of Lean

LEAN SOFTWARE DEVELOPMENT

Incorporating the essence of Lean manufacturing, Lean Software Development distilled the methodology into a set of principles specifically tailored for the software industry:

1) **Deliver Fast**: Accelerate delivery to promptly meet customer needs.

2) **Eliminate Waste**: Remove non-value-adding processes and artifacts.

3) **Respect for People**: Acknowledge the pivotal role of people in delivering value.

4) **Create Knowledge**: Emphasize learning and knowledge sharing.

5) **Build Quality**: Embed quality in every step rather than adding it later.

6) **Optimize the Whole**: Consider the entire value stream rather than local optimizations.

7) **Defer Commitment**: Keep options open and make decisions when they're most informed.

Lean is a holistic approach that seeks to optimize processes, minimize waste, and maximize value.

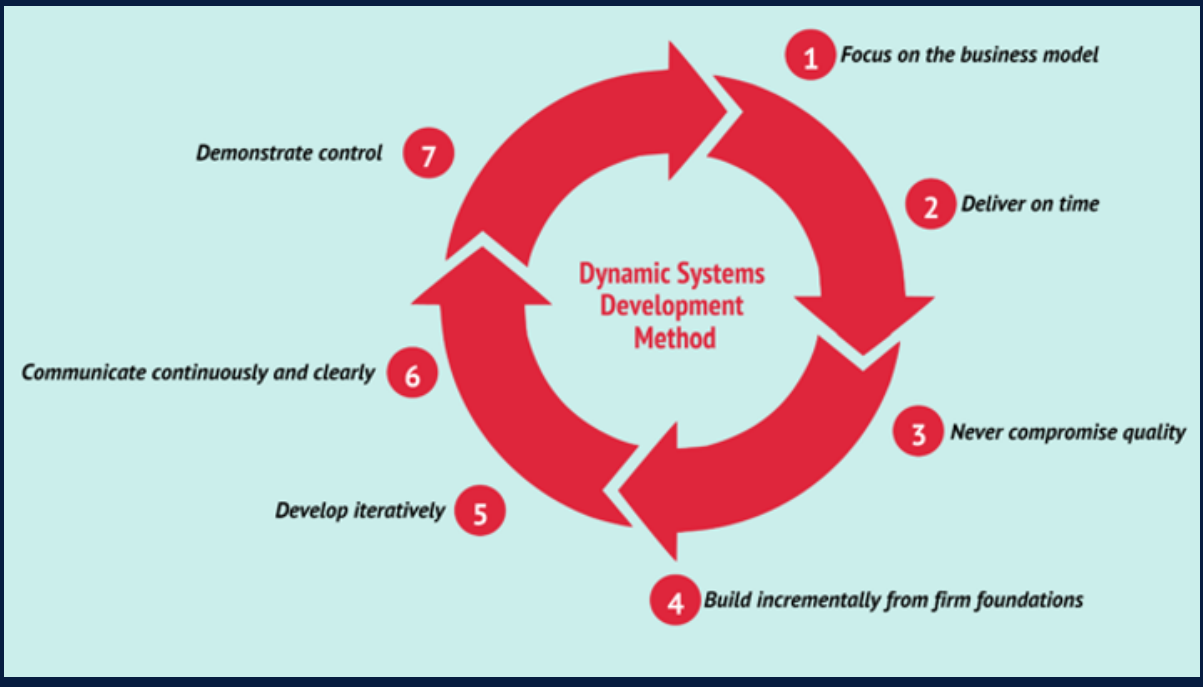
It has been adopted across various industries and serves as a guiding light for organizations striving for efficiency and excellence.

DYNAMIC SYSTEMS DEVELOPMENT METHOD (DSDM)

ORIGIN & BACKGROUND

Recognizing that post-delivery improvements are unavoidable, DSDM promotes adaptability by allowing for product changes even after delivery.

Furthermore, any additional developments should be easily reversible.



EXTREME PROGRAMMING (XP)

BACKGROUND & ESSENCE:

Extreme Programming (XP) is a unique Agile framework that prioritizes close collaboration between developers and customers.

Customers play a proactive role in the XP approach, through continuous feedback.

Developers tailor their software updates based on this feedback.

A framework for creating high-quality software that is closely aligned with user needs.

It combines rigorous software development practices with the flexibility of Agile, making it particularly useful when dealing with changing software requirements, working with small development teams, or leveraging technologies that support automated testing.

FEATURE DRIVEN DEVELOPMENT (FDD)

BACKGROUND & OVERVIEW:

FDD prioritizes detailed planning and design, making it particularly effective for teams with robust planning and design skills.

The essence of FDD is its focus on iterative development centered around distinct software features.

Typically, these feature-centric models are developed and refined bi-weekly.

SIGNIFICANCE

FDD's strength is its methodical approach to software development.

By concentrating on specific features, it ensures that each component of the software receives adequate attention.

This results in a unified product with meticulously planned, designed, and developed features.

CRYSTAL

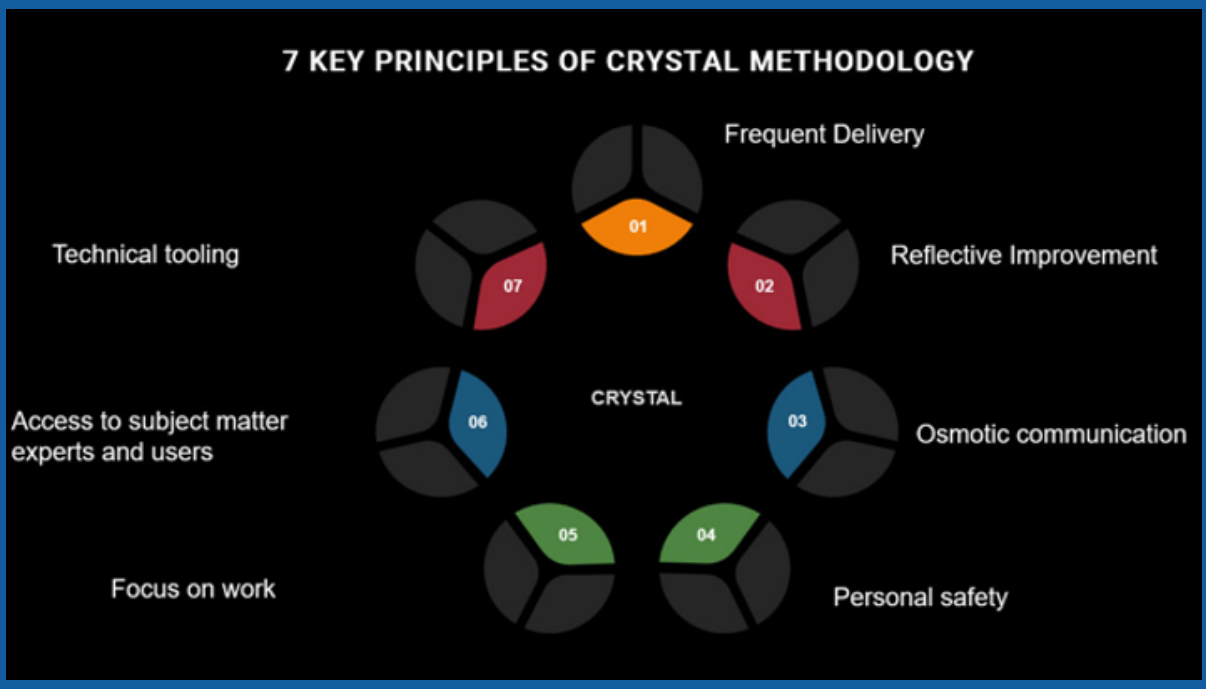
BACKGROUND & CLASSIFICATION

Crystal isn't a singular methodology but rather a family of Agile frameworks, each tailored to different team sizes. These frameworks include:

- Crystal clear: Teams of six or fewer employees
- Crystal yellow: Teams between 7 and 20 employees
- Crystal Orange: Teams between 21 and 40 employees
- Crystal red: Teams between 41 and 80 employees

- Crystal maroon: Teams between 81 and 200 employees
- Crystal diamond and Crystal sapphire: For larger projects

Each variant of Crystal is distinct, with practices and processes tailored to the team's size.



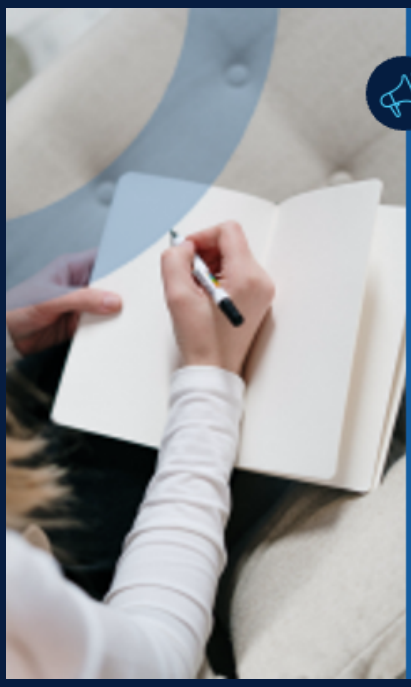
IN SUMMARY

Crystal sets itself apart by focusing on people and their interactions over rigid tools and processes.

It's especially useful for projects that prioritize team communication, active user involvement, continuous integration, and adaptable processes.

FROM WATERFALL TO AGILE:

HOW TO DETERMINE IF IT'S THE RIGHT MOVE FOR YOUR PROJECT



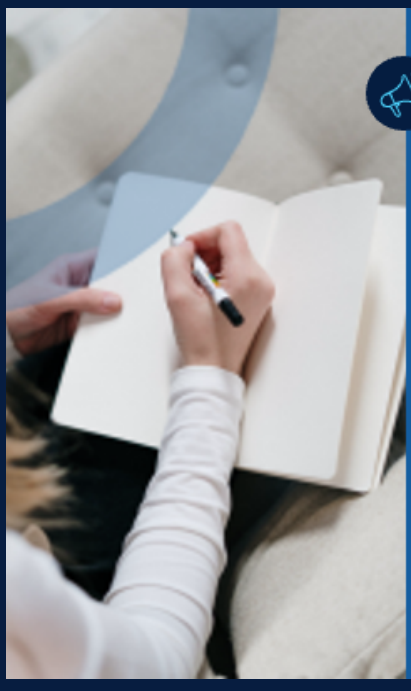
CONSIDER THE FOLLOWING CRITERIA

1. **Project Complexity and Uncertainty:**
   - *Is the project complicated, with a high level of uncertainty and frequent changes?*

2. **Stakeholder Engagement:**
   - *Are stakeholders willing to participate in the iterative process and engage on a regular basis?*

3. **Team Dynamics:**
   - *Is your team eager to collaborate, share responsibilities, and self-organize?*

4. **Delivery Timeline:**
   - *Is it acceptable to deliver in stages or in increments?*



5. **Organizational Support:**
   - *Is the culture and leadership of the organization supportive of a more adaptive approach?*

6. **Customer Involvement:**
   - *Is the customer or end-user willing to take an active role in the process and provide frequent feedback?*

7. **Risk Tolerance:**
   - *Is the organization willing to accept the risks associated with a new approach?*

8. **Tool and Resource Availability:**
   - *Is the organization equipped with, or willing to invest in, tools to support Agile processes (e.g., project management software)?*

9. **Training and Skillset:**
   - *Is the team willing to go through training, and is the organization willing to invest in this training, in order to effectively understand Agile principles and practices?*

10. **Regulatory Constraints:**
   - *Are there any strict laws or regulations that would be incompatible with an Agile approach?*

If your project and organization meet the majority of these criteria, Agile could be a good fit. It is critical to obtain buy-in from your team and key stakeholders, as well as to be prepared for the cultural shift .