**FDD**

**1. Geschichte zum Modell und Namensherkunft**

FDD (Feature-Driven Development) wurde 1997 von Jeff De Luca entwickelt, um ein großes, zeitkritisches Projekt durchzuführen. Es stellt den Feature-Begriff in den Mittelpunkt, organisiert die Entwicklung anhand eines Feature-Plans und definiert ein Prozess- und Rollenmodell, das gut mit klassischen Strukturen harmoniert.

**2. Funktionsweise von FDD:**

* Iterativ und inkrementell: Entwicklung in kurzen Iterationen mit Fokus auf Features.
* **Fünf Aktivitäten:** 
  1. Domänenstruktur entwickeln (Funktionen identifizieren und kategorisieren).
  2. Funktionsliste erstellen (Features detailliert aufschlüsseln und schätzen).
  3. Planschätzung (Ressourcen und Zeitaufwand bestimmen).
  4. Entwicklung von Features (kurze Iterationen).
  5. Fortschritt überprüfen (Ergebnisse bewerten, Probleme frühzeitig erkennen).

**Klassisch oder agil?**

* Agil: FDD ist iterativ, flexibel und ermöglicht eine systematische Vorgehensweise bei Softwareprojekten.

**3. Wofür geeignet?**

FDD ist für große und komplexe Softwareprojekte geeignet. Es zerlegt die Entwicklung in kleinere, beherrschbare Arbeitspakete und funktioniert gut mit großen Teams durch klare Rollen. Die strukturierte Vorgehensweise hilft bei zeitkritischen Projekten und Teams mit unterschiedlicher Erfahrung.

**4.** **Vorteile:**

* **Strukturierte und systematische Vorgehensweise** → Geeignet für große und komplexe Softwareprojekte.
* **Zerlegung in kleinere, beherrschbare Arbeitspakete** → Verbesserte Kontrolle über die Entwicklung.
* **Kurze Iterationen** → Kritische Bugs frühzeitig erkannt und behoben.
* **Gut mit klassischen Projektstrukturen kompatibel** → Erleichtert die Einführung.
* **Klare Rollen wie Chefarchitekt und Chefprogrammierer** → Effiziente Teamführung bei großen Teams.

**Nachteile:**

* **Feature-Begriff im Mittelpunkt** → Weniger Flexibilität bei Änderungen außerhalb des Feature-Plans.
* **Planschätzung und detaillierte Aufschlüsselung** → Erfordert initialen Aufwand.
* **Für große und komplexe Softwareprojekte** → Eventuell nicht für kleinere Vorhaben geeignet.

**5. Jira** → Für die Verwaltung von Feature-Listen und Planung.

**Azure DevOps** → Für die Versionskontrolle und iterative Entwicklung.

**GitLab / GitHub** → Für Code-Management und Zusammenarbeit in großen Teams.

**Enterprise Architect** → Für Modellierung und Strukturierung der Domänenmodelle.

**Trello** → Für einfache, featurebasierte Aufgabenverwaltung.

Ein Bild, das Diagramm, Reihe, Origami enthält.

Automatisch generierte Beschreibung

Quellen:

<https://de.wikipedia.org/wiki/Feature_Driven_Development>

<https://www.alleaktien.com/lexikon/feature-driven-development>

<https://www.computerwoche.de/article/2656955/feature-driven-development.html>