it-agile
die agilen Experten der akquinet AG

Feature Driven Development

Die andere agile Methode

Dipl.-Inform. Henning Wolf
henning.wolf@it-agile.de

FDD™
Feature Driven Development

Agile Alliance
CORPORATE MEMBER

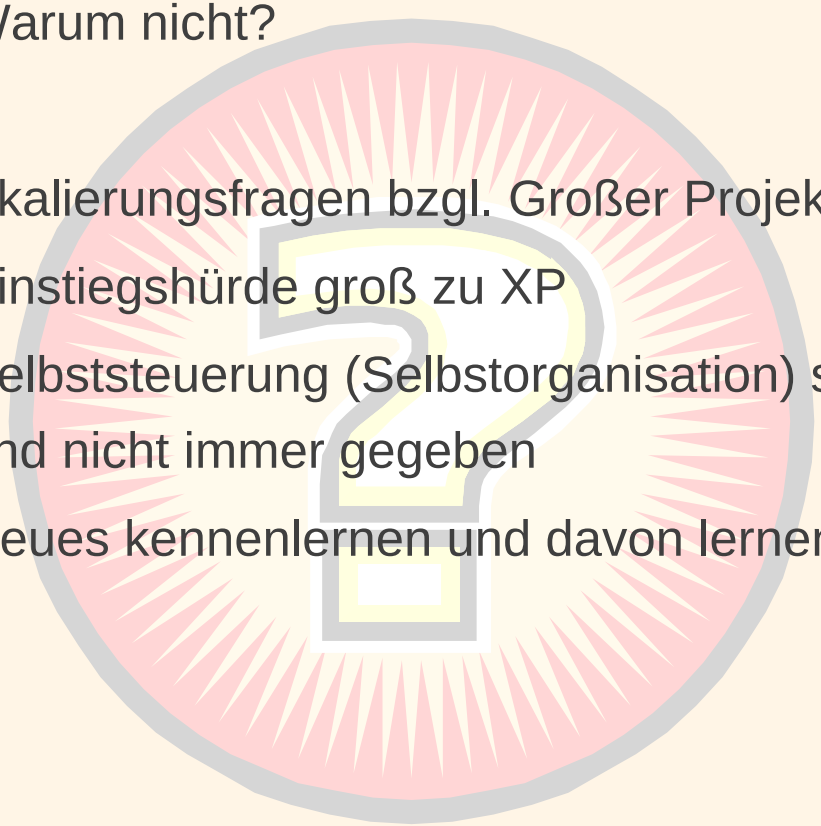
Überblick

- Warum mit FDD beschäftigen?
- Woher kommt FDD?
- Was ist FDD?
  - 5 (Teil-)Prozesse
  - Rollenmodell
- Vorteile von FDD
- Diskussion: Ist das (noch/schon) agil?
- Fazit

Warum mit FDD beschäftigen?

- Warum nicht?

- Skalierungsfragen bzgl. Großer Projekte
- Einstiegshürde groß zu XP
- Selbststeuerung (Selbstorganisation) schwierig und nicht immer gegeben
- Neues kennenlernen und davon lernen



Woher kommt FDD?

- Jeff De Luca
- 1997 Singapore-Projekt
  - 17 Monate, 50 Entwickler
  - mit Peter Coad
- Kontinuierliche Weiterentwicklung
- http://www.featuredrivendevelopment.com

Jeff De Luca

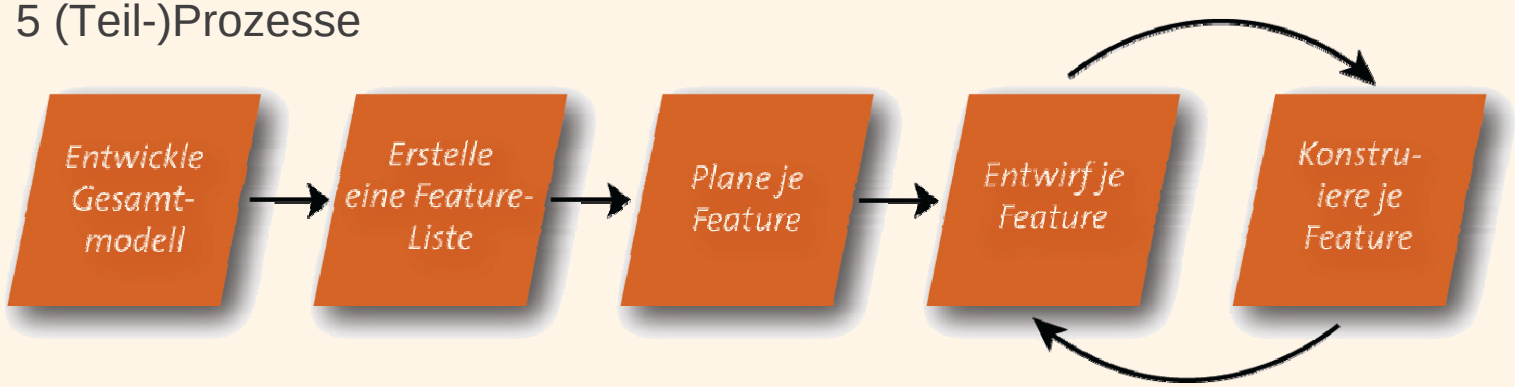
- Beschreibung auf 10 Seiten:
http://www.nebulon.com/articles/fdd/download/fddprocessesA4.pdf

Vorwort von Jeff

- „Managers must first learn to see, hear, and think about human systems before they can hope to control them."
- „Software projects are human systems."

Was ist FDD?

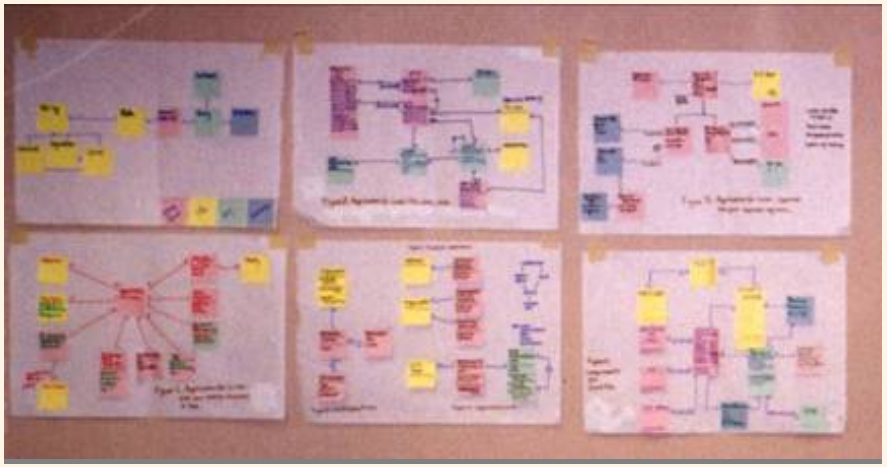
- 5 (Teil-)Prozesse

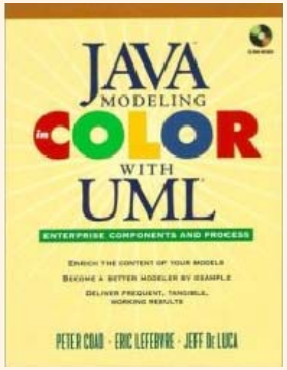


- Rollenmodell
  - Projektleiter
  - Entwicklungsleiter
  - Chefarchitekt (Chefmodellierer)
  - Chef-Programmierer
  - Entwickler
  - Fachexperten

Entwickle Gesamtmodell

- Ziele:
  - Domäne kennenlernen und verstehen
  - Fachliches Klassenmodell
  - Ggf. auch Sequenzdiagramme
- Beteiligte:
  - Chefarchitekt = Moderator
  - (Chef-)Programmierer
  - Fachexperten
- Vorgehen:
  - Meist Modeling In Color





Exkurs: Modeling in Color 1/2

- Fachliche Konzepte werden in 4 Archetypen klassifiziert.
- Ein Archetype ist ein Prototyp, ein typischer Vertreter, dessen Form alle Dinge **mehr oder weniger** folgen.

1. Ist es ein Zeitpunkt oder Zeitraum im Hinblick auf einen Geschäftsvorfall?

<Moment-Interval>

2. Ist es eine Rolle, die eingenommen wird?

<Role>

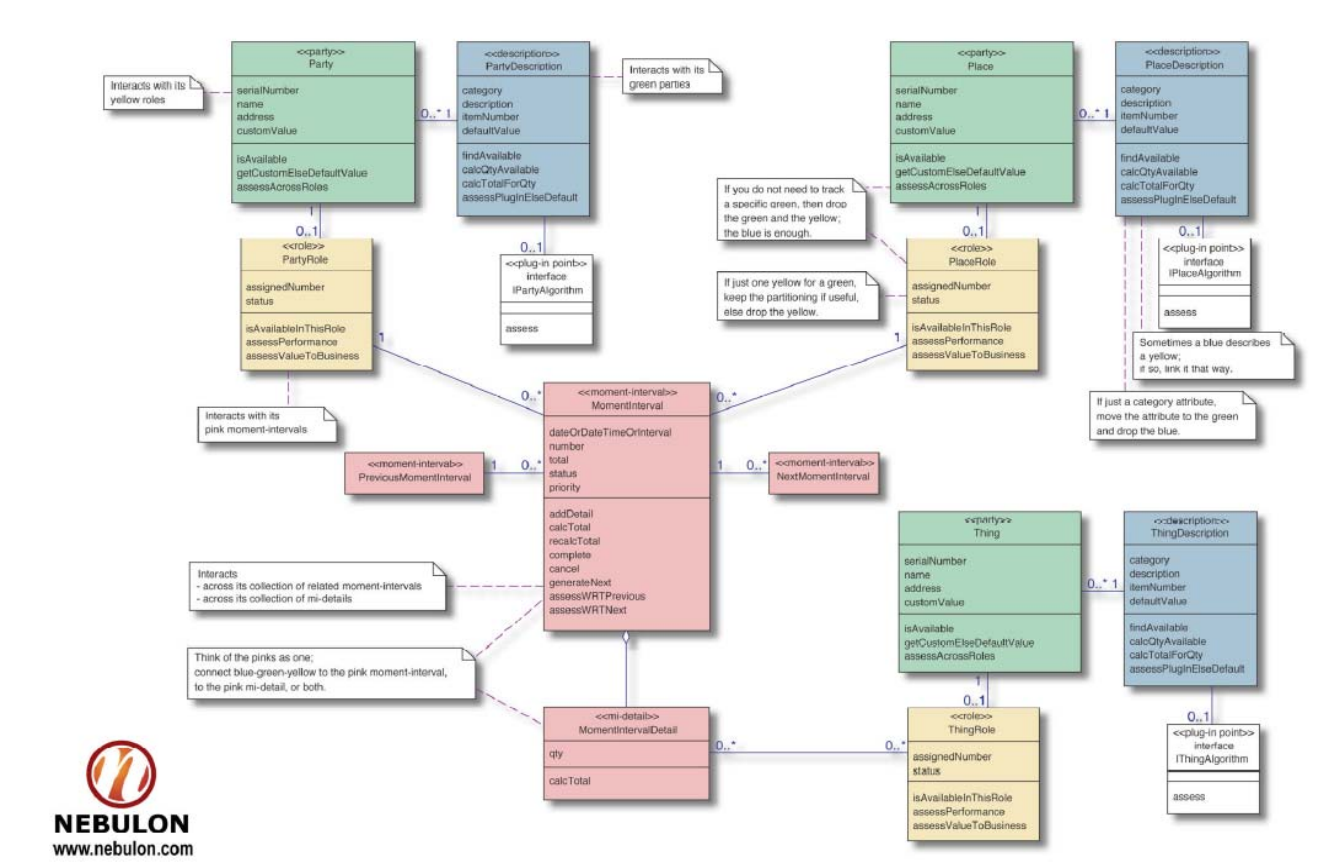
3. Ist es eine Beschreibung wie ein Katalog-Eintrag?

<Description>

4. Ansonsten ist es eine Partei, ein Ort oder ein Ding

<PPT>

Exkurs: Modeling in Color 2/2



Erstelle Featureliste

- Ziele:
  - Alle Anforderungen in Form von Features liegen vor
  - Basis für weiteres Vorgehen und Schätzung
- Beteiligte:
  - Chef-Programmierer
  - Anschließend Abstimmung mit Fachexperten

- Feature-Schema:

<Aktion> <Ergebnis> <Objekt>

- Beispiel:

„Berechne Summe der Rechnungspositionen.“

Hierarchie der Features:

- Major Feature Set (Geschäftsbereich)
  - Feature Set (Geschäftstätigkeit)
    - Feature (Systemfunktion)

Plane je Feature

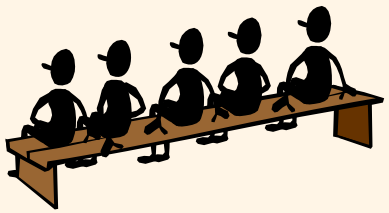
- Ziele:
  - Projektplan erstellen
  - Aufwände und Termine klären
  - „Flow“ für alle Entwickler herstellen

Beteiligte:

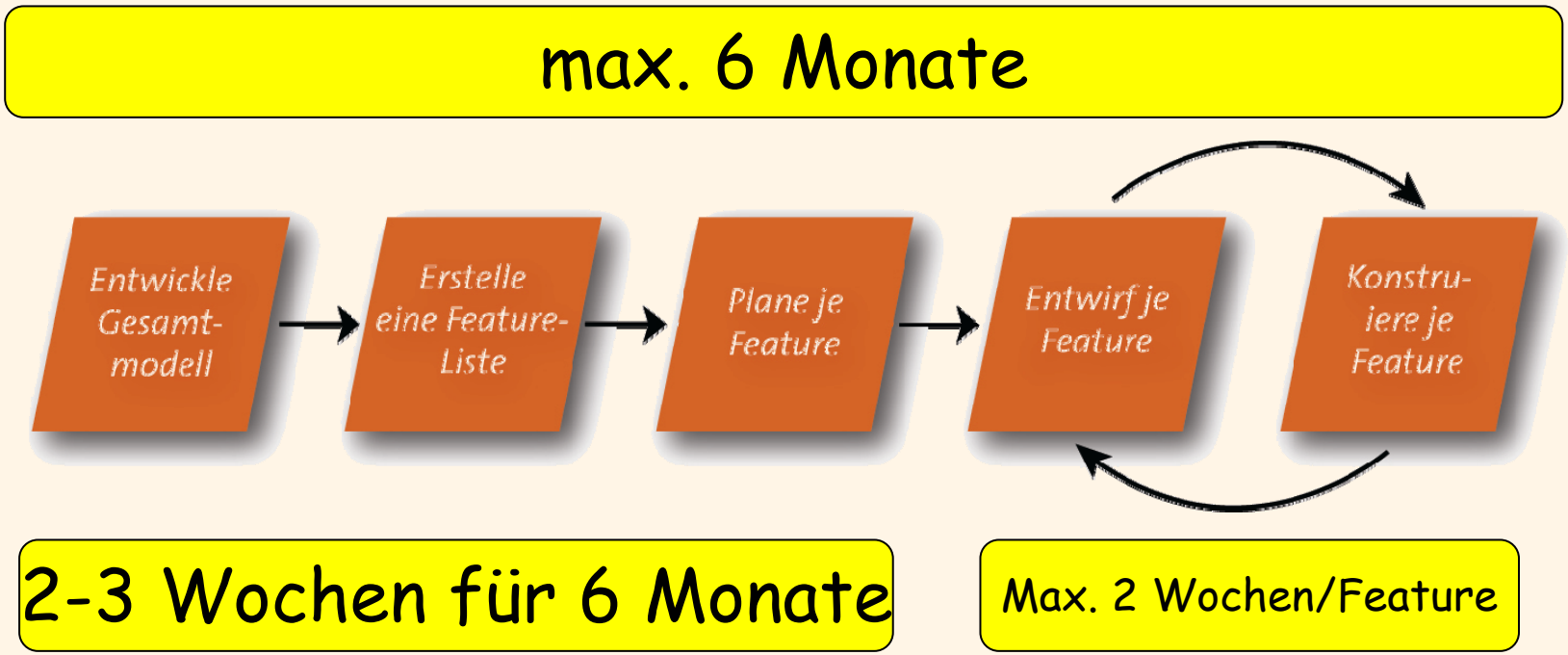
- Projektleiter
- Entwicklungsleiter
- Chef-Programmierer

- Vorgehen:
  - Feature-Klassen-Beziehungen ermitteln
  - Class-Owner festlegen
  - Arbeitsbelastung ausbalancieren

Es ergeben sich Feature-Teams:
Alle Class-Owner der beteiligten Klassen.



Zeitliche Abläufe



Entwirf je Feature

- Ziele:
  - Gemeinsamen Entwurf erstellen
  - Aus gemeinsamen Entwürfen lernen
- Beteiligte:
  - Feature-Team
  - ggf. Chefarchitekt
  - ggf. Chef-Programmierer

- Vorgehen:
  - Im Team
  - Klassen- und Sequenzdiagramme erstellen
  - Design-Inspektionen

Nach „Entwirf je Feature“ folgt immer direkt „Konstruiere je Feature“ desselben Features!

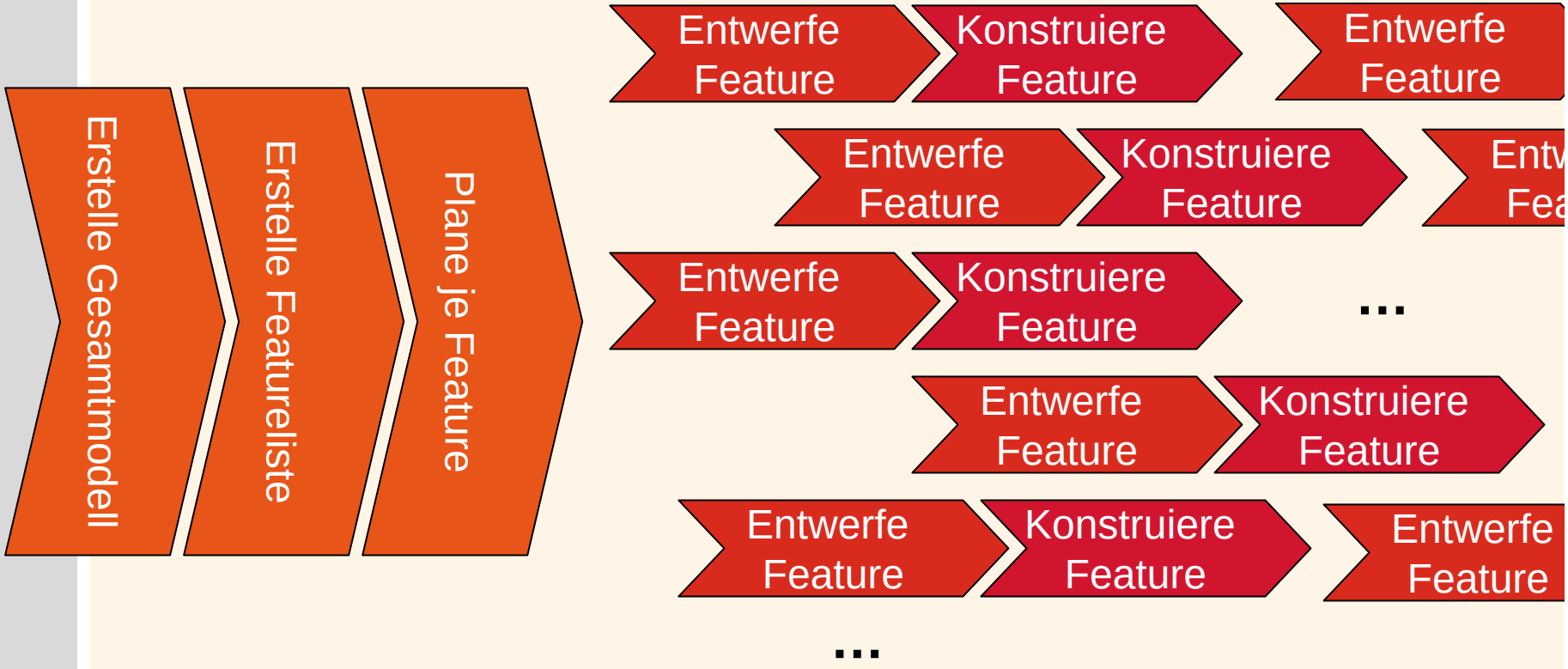
Konstruiere je Feature

- Ziele:
  - Produktivklassen erstellen
  - Tests erstellen
  - Programmieren verbessern/lernen
- Beteiligte:
  - Feature-Team
  - ggf. Chefarchitekt
  - ggf. Chef-Programmierer

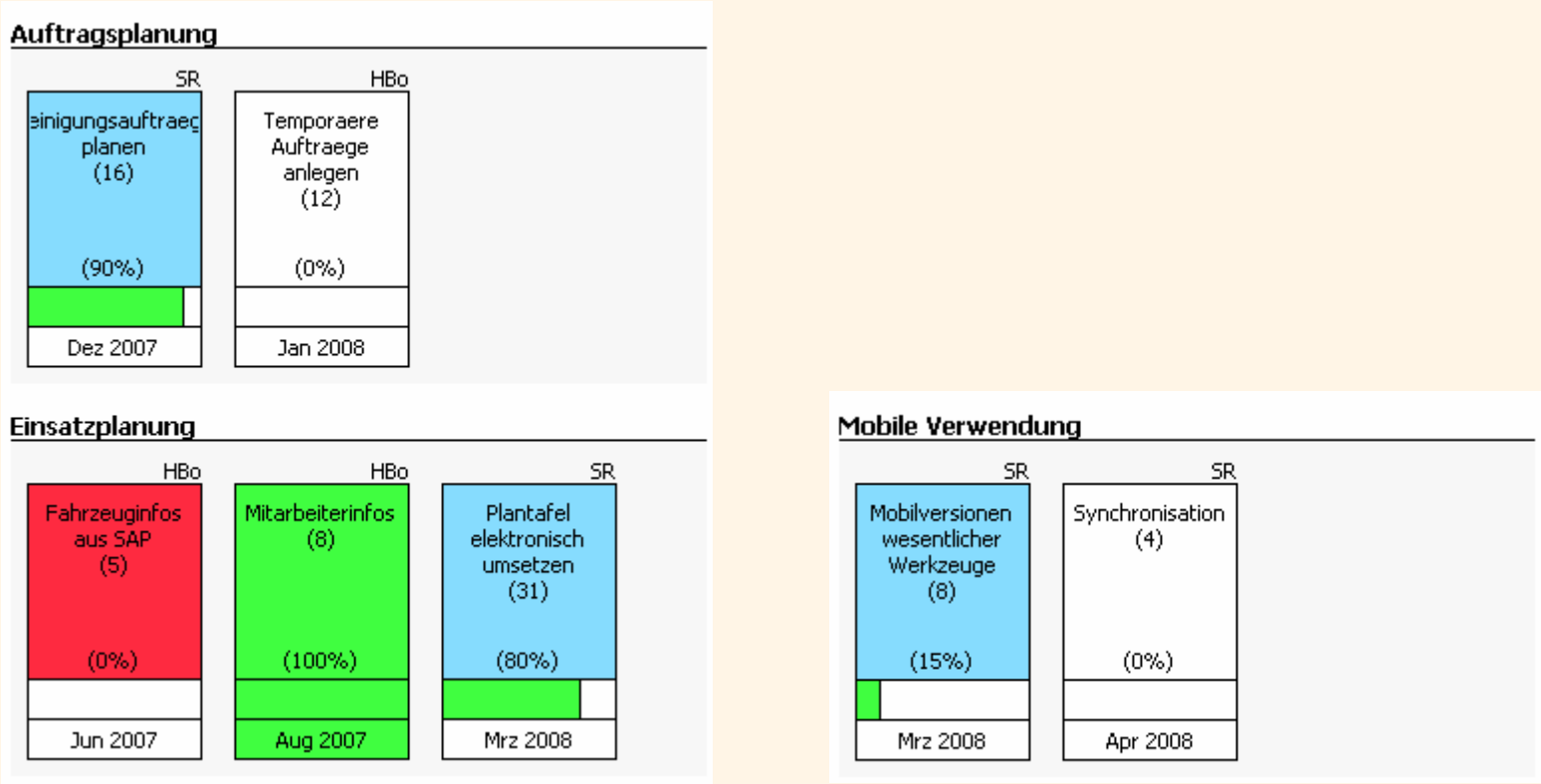
- Vorgehen:
  - Class-Owner programmieren Produktivklassen
  - Class-Owner erstellen Tests
  - Im Team erfolgen Code-Inspektionen

Je Feature dauert
„Entwirf je Feature“
und „Konstruiere je Feature“
nicht länger als 2 Wochen!

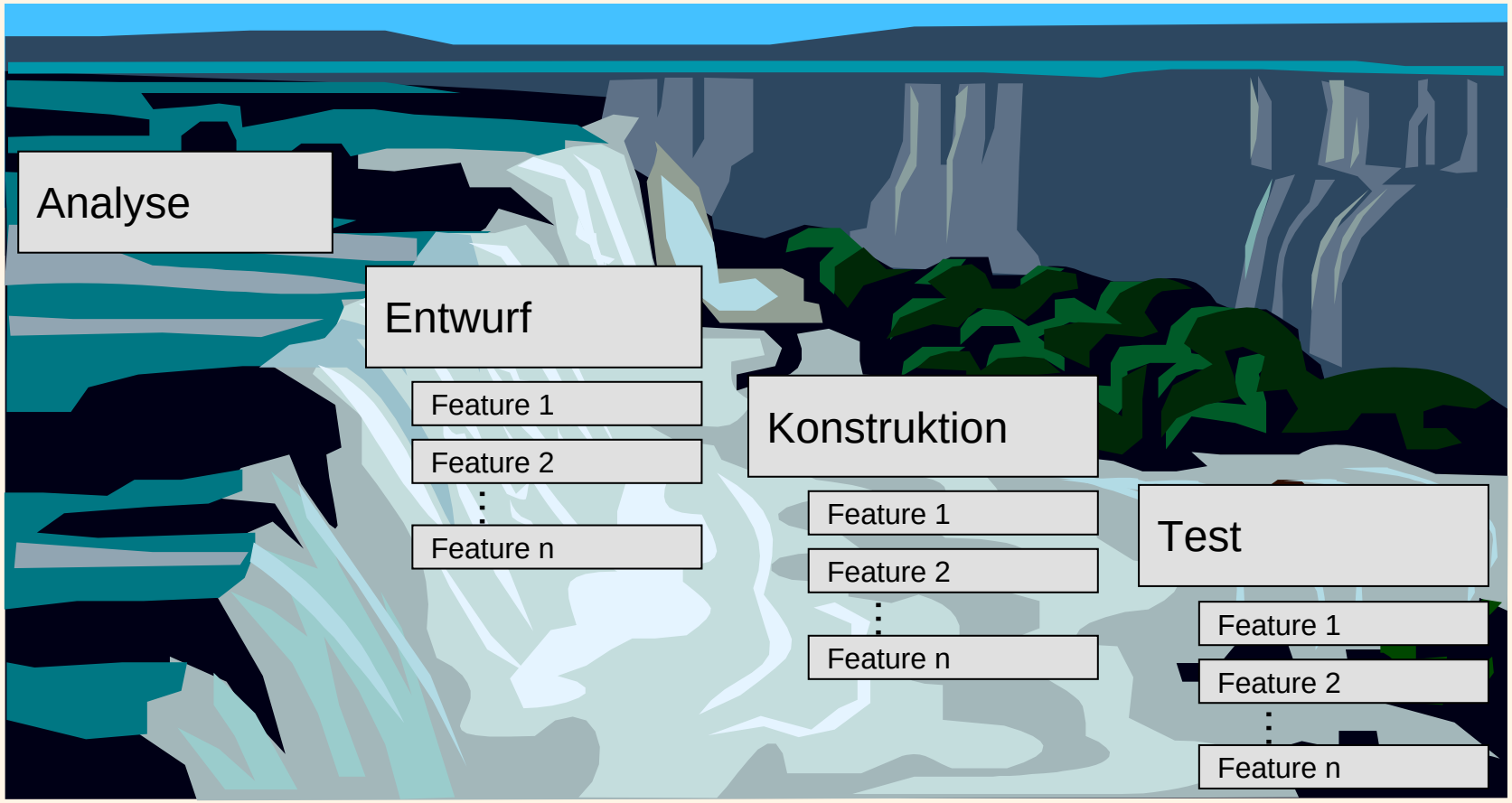
Zeitlicher Ablauf: Parallele Teilprozesse



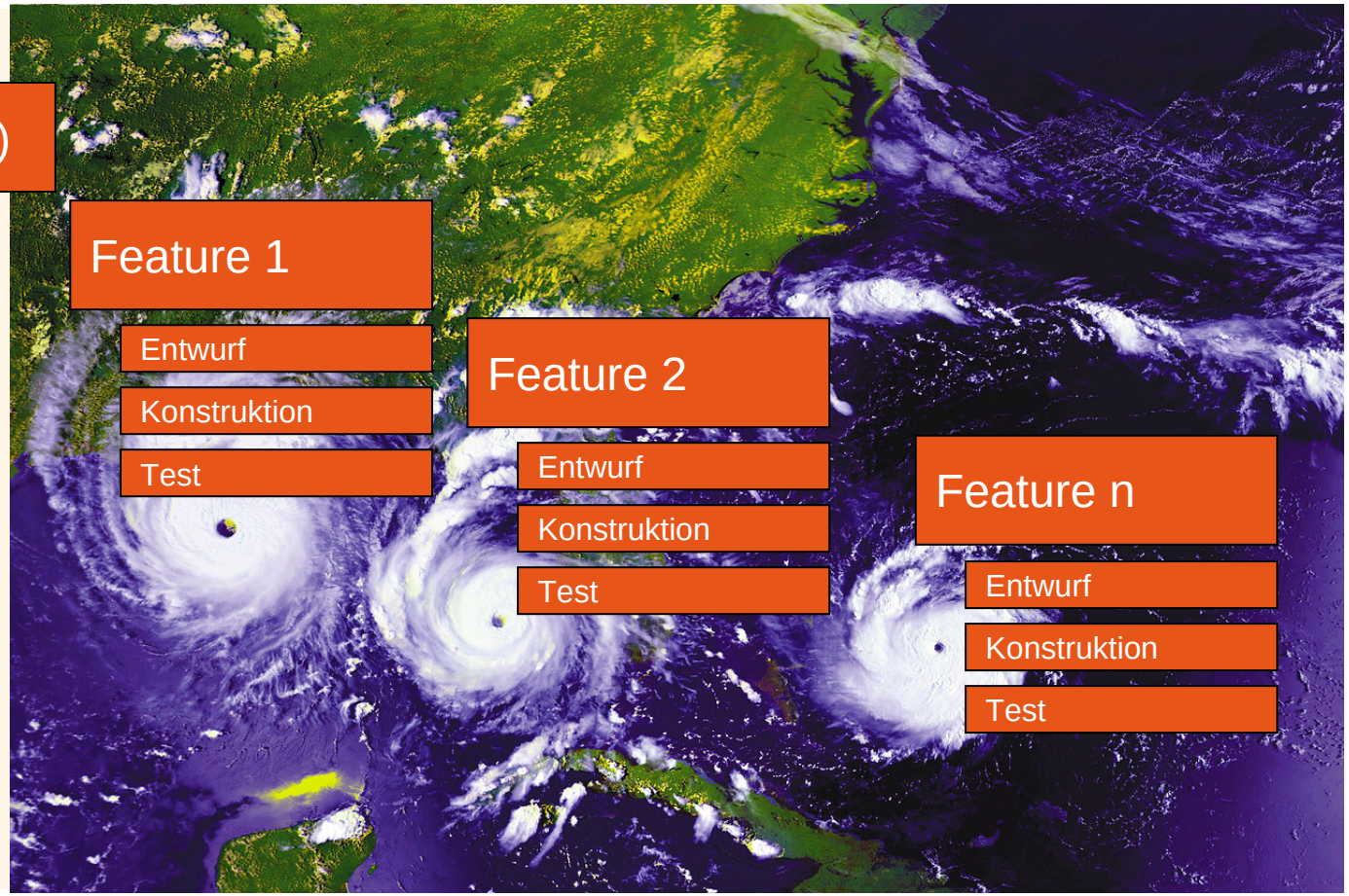
Projektfortschrittsmessung mit Parking Lot Chart



Wasserfall vs. Feature-Wirbel



Wasserfall vs. Feature-Wirbel



Vorteile von FDD

- Klar beschrieben
- Weniger Hürden für viele Organisationen
  (im Vergleich zu anderen agilen Methoden)
- Skaliert gut
- Auch für große Projekte geeignet
- Initiale Modellierungsphase explizit vorgesehen
  (und auf angenehmer Abstraktionsebene)
- Passt gut zu Festpreiskonstellationen

Diskussion: Ist das (noch/schon) agil?

- Agile Werte:
  - Rückkopplung/Feedback — ja, an vielen Stellen
  - Einfachheit — ja
- Agiles Manifest:
  - Menschen wichtiger als Prozesse — ja, aber anders
  - Laufende Software wichtiger als Doku — ja
  - Zusammenarbeit mit dem Kunden wichtiger als Vertragsverhandlungen — ja
  - Veränderungen begrüßen statt Planverfolgung — ja, aber anderer Takt

Fazit

- Wir lernen von FDD (und können auch in andere Projekte übernehmen)
  - Feature-Beschreibungsschema, Feature-Strukturierung
  - Leichtgewichtige Modellierung (in Color)
  - Code-Ownership und Inspektionen als Alternative zu Collective-Ownership/TDD/Pair-Programming
- Leichter einzuführen
- Klares Rollenmodell

- Für manches Projektsetting (Team, Kunde, Größe) kann FDD eine Alternative sein!

Vielen Dank für die Aufmerksamkeit

Noch Fragen?