

Domain-Driven Design

What is it, and what are the benefits?

Prof. Dr. Stefan Bente



Technology
Arts Sciences
TH Köln

What you can expect

1. DDD – what is it?
2. What are the benefits?
3. What impact can it have on your architecture?

1. DDD – what is it?

DDD Series

Domain-Driven

DESIGN

Tackling Complexity in the Heart of Software



Eric Evans
Foreword by Martin Fowler

DDD Community

IMPLEMENTING DOMAIN-DRIVEN DESIGN

VAUGHN VERNON
FOREWORD BY ERIC EVANS



- „Der entscheidende Punkt, um den es mir geht, ist auch der Grund, warum das Konzept „Domain-driven Design“ heißt:
- Wenn wir Software entwickeln, sollte unser Fokus **nicht primär auf den Technologien** liegen, die wir verwenden.
- Stattdessen sollte sich unser Hauptaugenmerk auf die geschäftliche Seite richten, also auf den **fachlichen Bereich**, den wir mit unserer Software unterstützen wollen – kurz: **auf die Domäne**.
- Das ist es, was ich mit Domain-driven Design meinte.“

- »The crucial point that I am concerned with is also the reason why the concept is called "domain-driven design":
- When we develop software, our **focus should not be primarily on the technologies** we use.
- Instead, our main focus should be on the **business side**, i.e. on the specialist area that we want to support with our software - in short: **on the domain**.
- That's what I meant by domain-driven design.«

Technical Complexity vs. Domain Complexity

„Car insurance is easy, that's always the same, we'll manage this.

But life insurance is tricky, can we skip this?“

1. Domain = **Starting Point** for the whole design process
2. Handling complexity **in iterations**
3. „**Ubiquitous Language**“
based on domain model

**2. What are the
benefits?**

Fast, agile development



...

Design in Iterations

(1)

When you start looking at a problem and it seems really simple, you don't really understand the complexity of the problem.



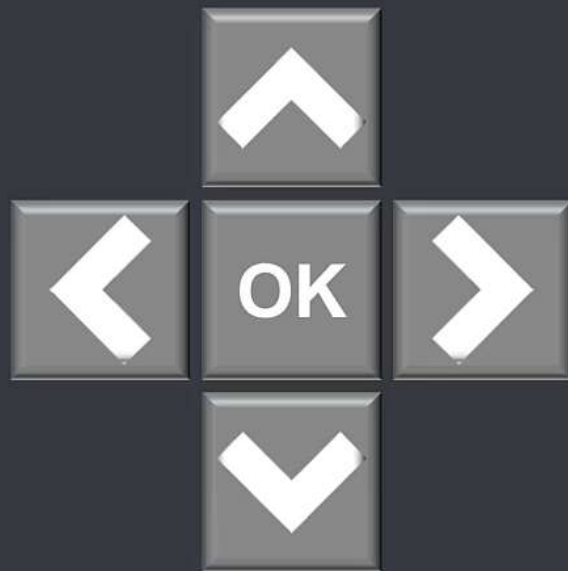
The Problem

- Let's go back to 2004 ...
- Music player
- Navigation through large amounts of tree-shaped data
- Handheld device without touch

(1) Complexity Not Really Understood



(1) Complexity Not Really Understood



Song	Artist	Year
(Don't Fear) The Reaper	Blue Oyster Cult	1976
(I Can't Get No) Satisfaction	Rolling Stones	1965
(I Just) Died in Your Arms	Cutting Crew	1986
(Oh) Pretty Woman	Van Halen	1982
(Sittin' On) The Dock of the Bay	Otis Redding	1968
(You Can Still) Rock in America	Night Ranger	1983
(You Gotta) Fight for Your Right (To Pa	Beastie Boys	1986
18 And Life	Skid Row	1989
1984/jump	Van Halen	1984
19th Nervous Breakdown	Rolling Stones	1966
2112: Overture / The Temples of Syrin	Rush	1976
25 Or 6 To 4	Chicago	1970
3 Strange Days	School Of Fish	
30 Days In the Hole	Humble Pie	1972
50 Ways to Leave Your Lover	Paul Simon	1975
867-5309 / Jenny	Tommy Tutone	1982
A Conspiracy	The Black Crowes	
A Day In The Life	The Beatles	1967
A Face In The Crowd	Tom Petty & The Heartbreakers	

(2)

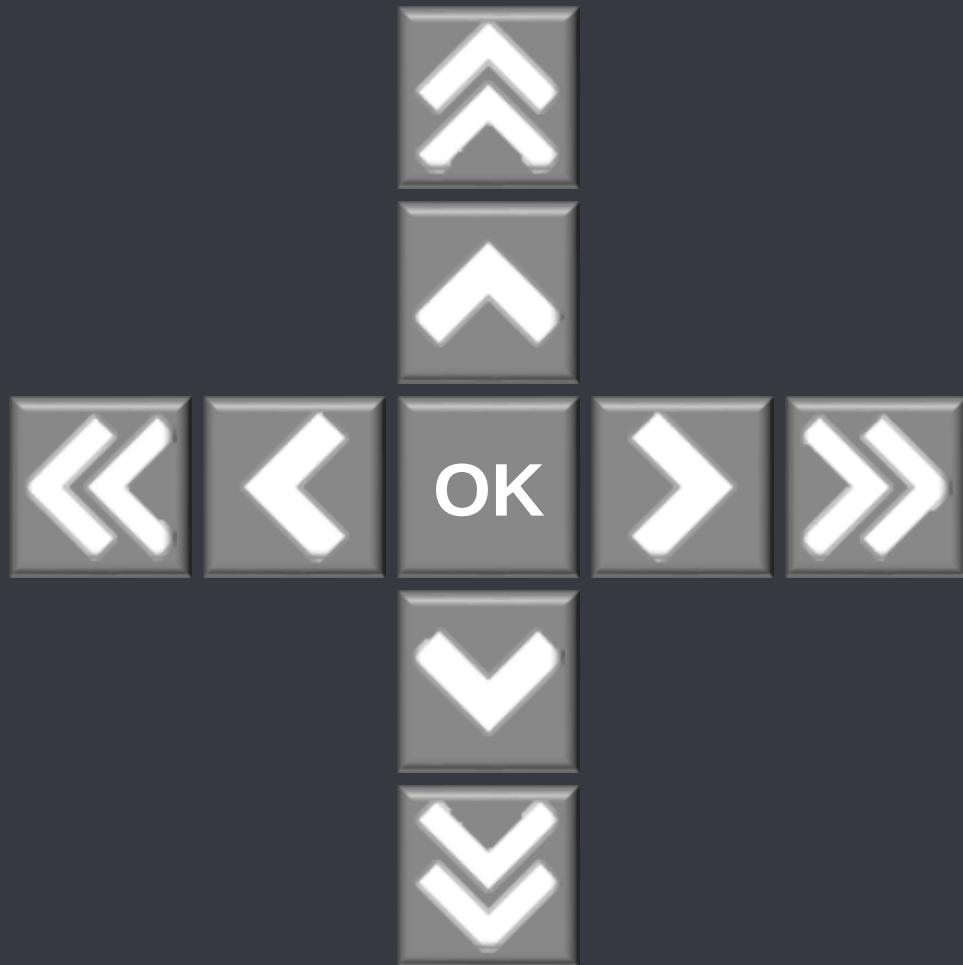
Then you get into the problem, and you see that it's really complicated, and you come up with all these convoluted solutions.



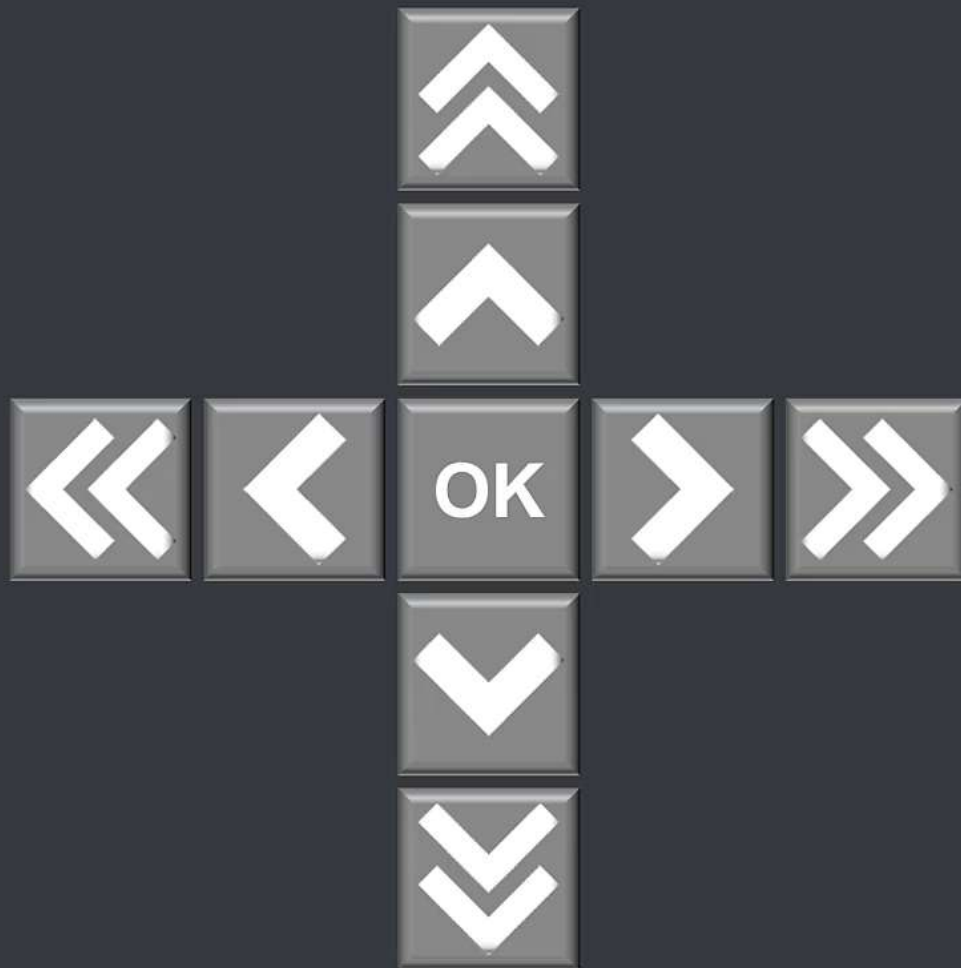
(2) Convoluted Solution



(2) Convoluted Solution

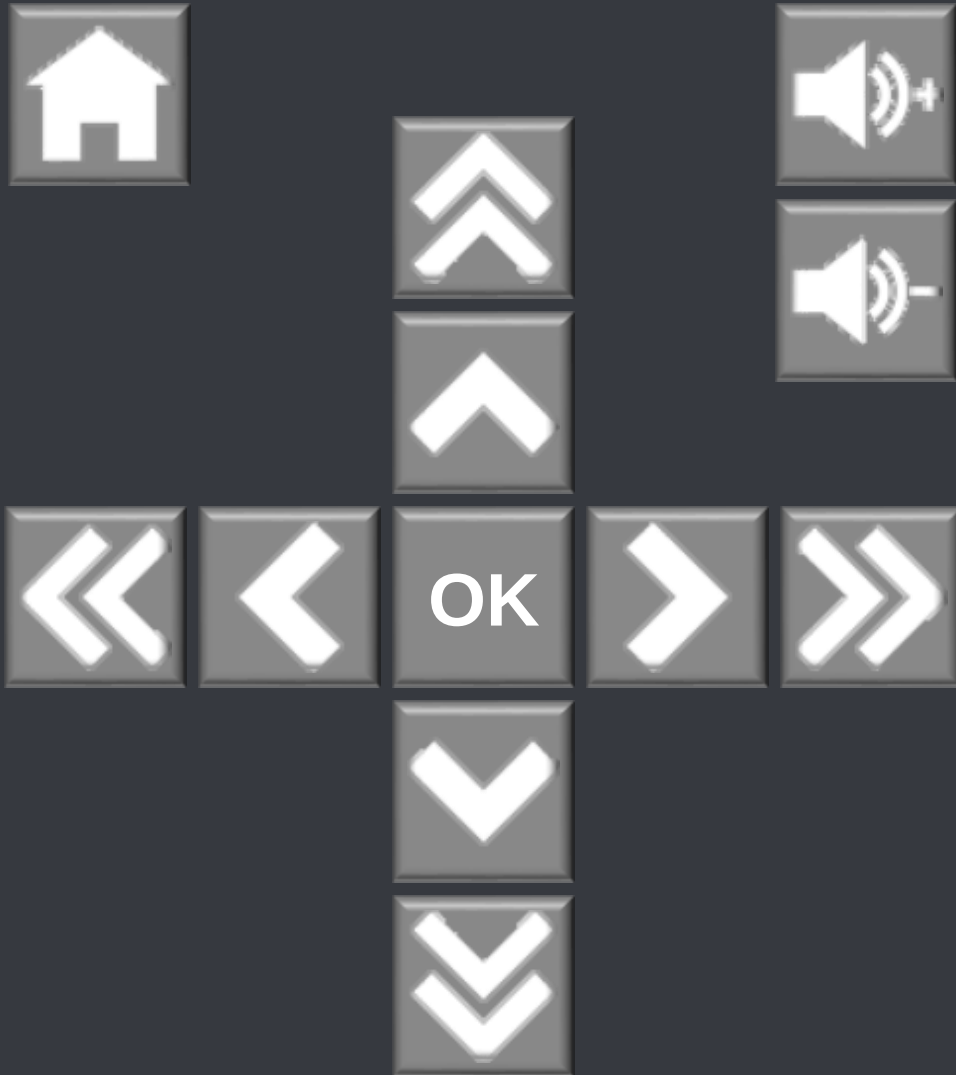


(2) Convoluted Solution



Song	Artist	Year
#9 Dream	John Lennon	1974
(Don't Fear) The Reaper	Blue Oyster Cult	1976
(I Can't Get No) Satisfaction	Rolling Stones	1965
(I Just) Died in Your Arms	Cutting Crew	1986
(Oh) Pretty Woman	Van Halen	1982
(Sittin' On) The Dock of the Bay	Otis Redding	1968
(You Can Still) Rock in America	Night Ranger	1983
(You Gotta) Fight for Your Right (To	Beastie Boys	1986
18 And Life	Skid Row	1989
1984/jump	Van Halen	1984
19th Nervous Breakdown	Rolling Stones	1966
2112: Overture / The Temples of Syr	Rush	1976
25 Or 6 To 4	Chicago	1970
3 Strange Days	School Of Fish	
30 Days In the Hole	Humble Pie	1972

(2) Convoluted Solution



That's sort of the
middle, and
that's where
most people
stop...



(3)

But the really great person will keep on going and find the key, the underlying principle of the problem — and come up with an elegant, really beautiful solution that works.“



(3) Really Elegant Solution



Fast, agile development



Autonomous teams



Decoupled / loosely coupled business contexts („Bounded Contexts“)





Grade
management

Enrollment

Course Catalogue

Lecture Scheduling

Project Market
PROX

„Find your
Learning Group“



Raum

Grade
management

- Study regulations
- Examinations
- Exam registration

Study
Program

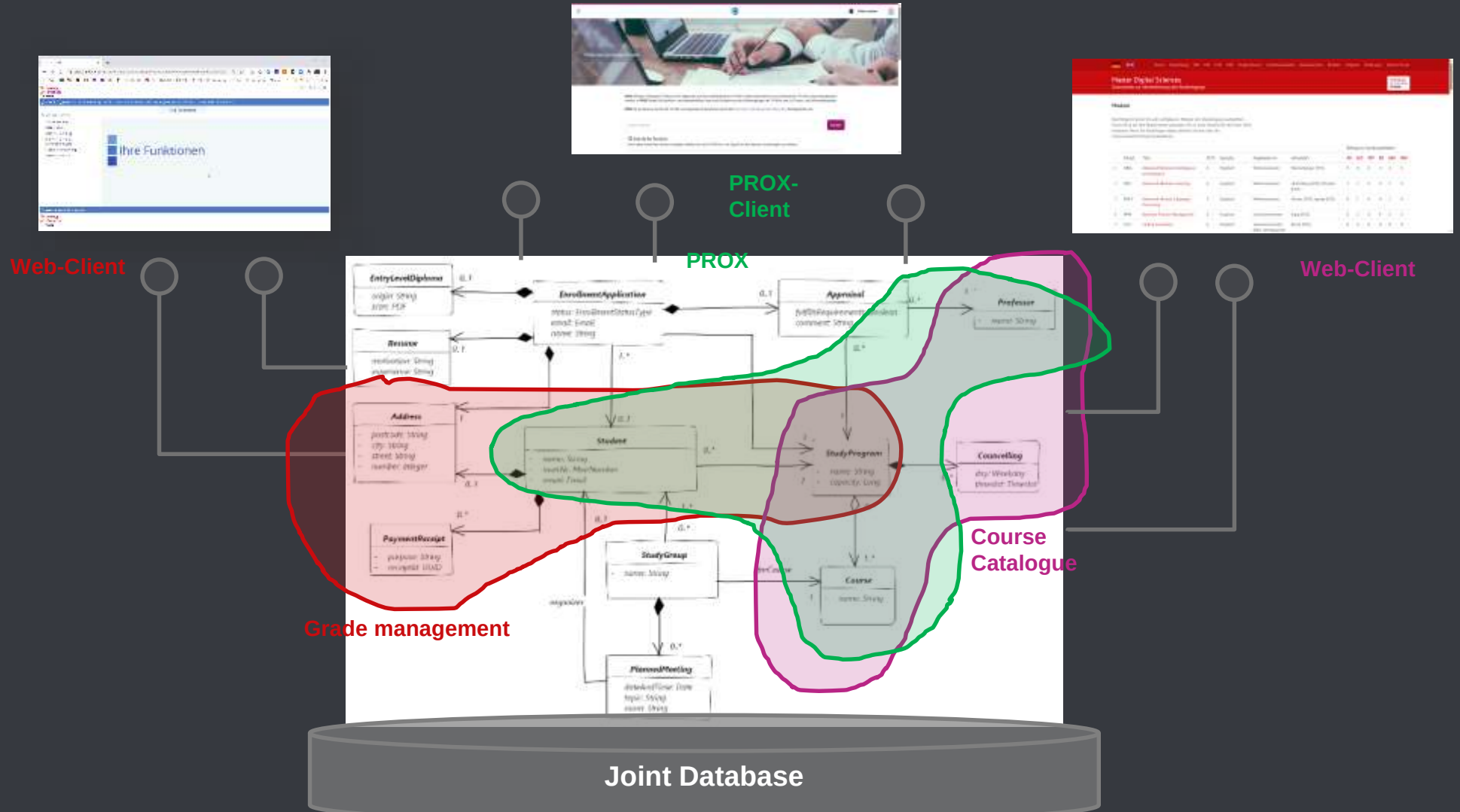
Project Market
PROX

Course Catalogue

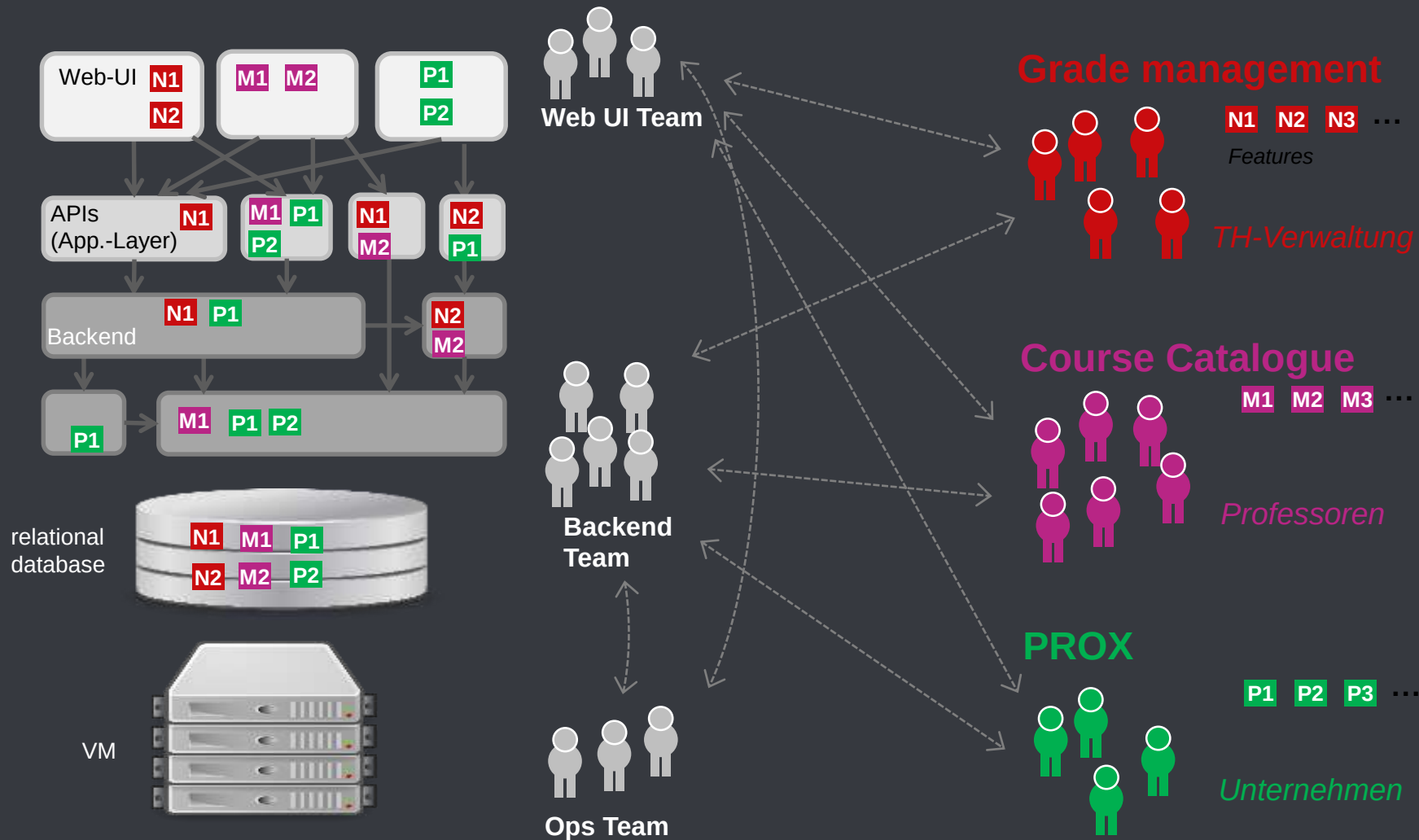
- Goals of the degree program
- Graduate profiles
- Courses

- Study projects
- BA / MA Theses
- Opportunities for internships and practical projects
- Mapping of local companies to study program

Classical Monolith ...



Classical Monolith ...



Risk 1: Big Ball of Mud

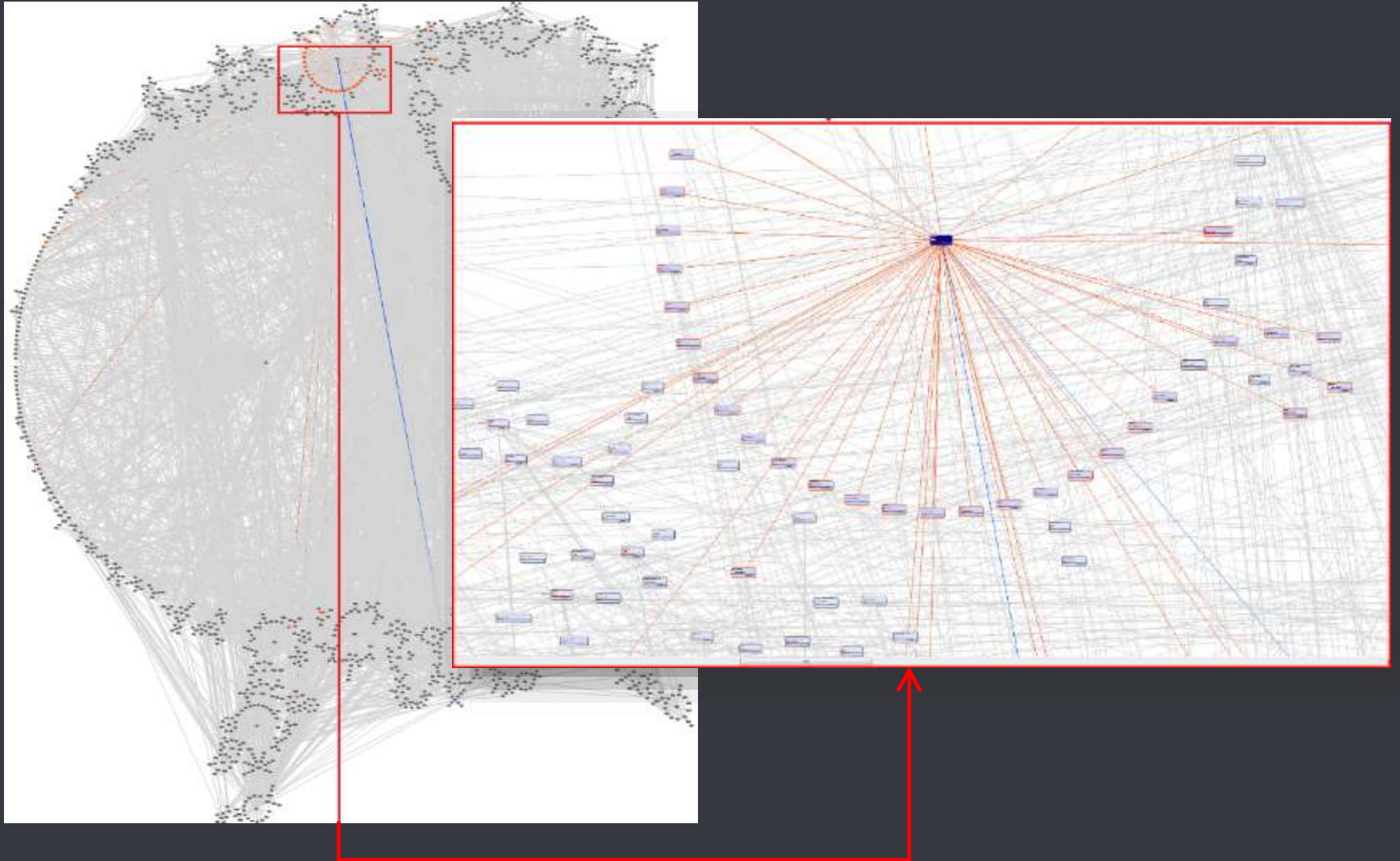
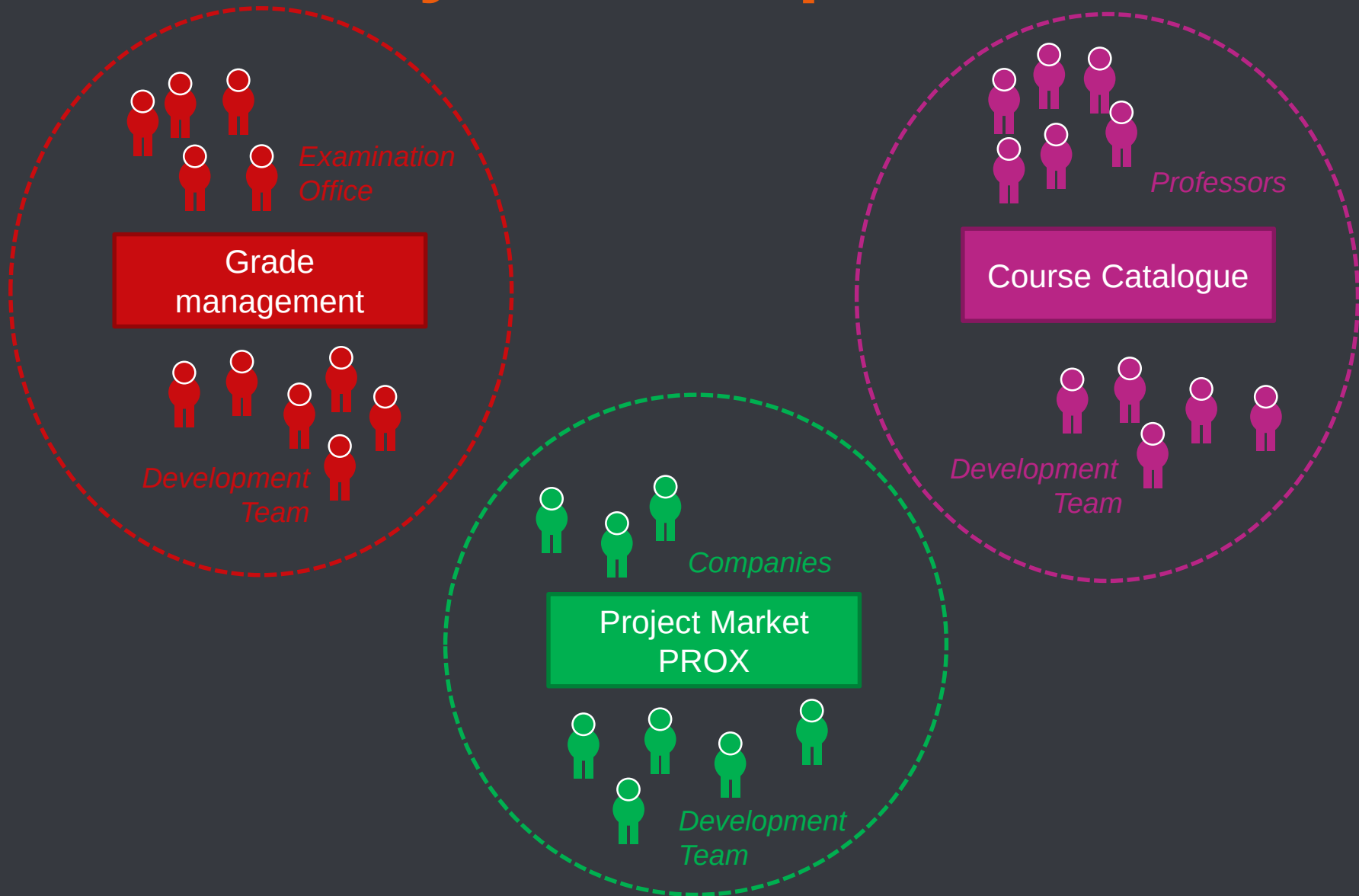


Bild: Remy Porter, <https://thedailywtf.com/articles/the-big-balls-of>

Risk 2: Too slow for Innovation

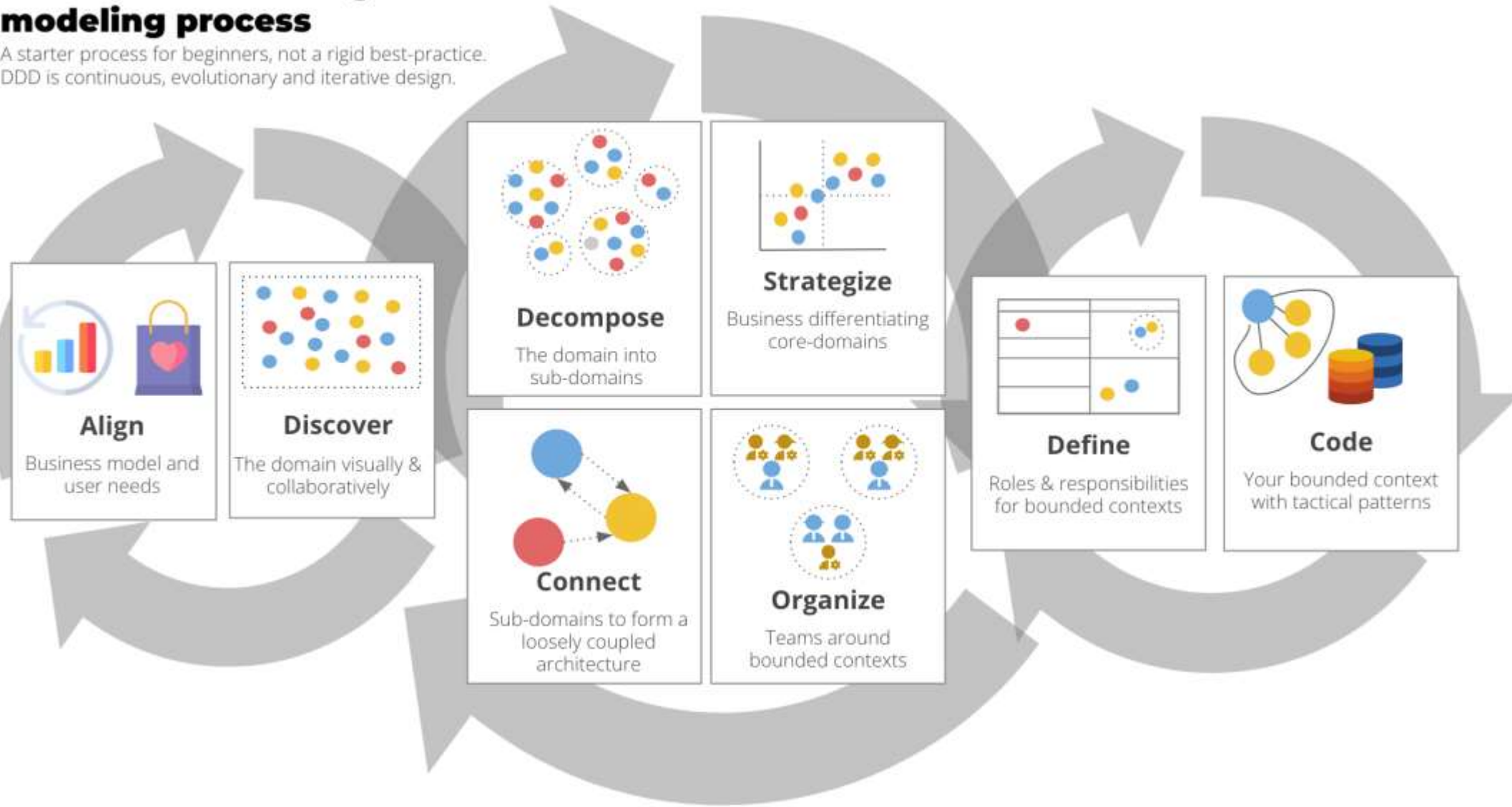
- Lots of coordination required for each feature
- Long release cycles (usually $\leq 4-6$ main releases per year)
- Complex integration tests
- Little flexibility in the development and change process

Goal: Loosely / de-coupled contexts



Domain-Driven Design starter modeling process

A starter process for beginners, not a rigid best-practice.
DDD is continuous, evolutionary and iterative design.



EVENT STORMING

Alberto Brandolini



Workshop 1: Event Storming

Domain-Driven Design starter modeling process

A starter process for beginners, not a rigid best-practice.
DDD is continuous, evolutionary and iterative design.

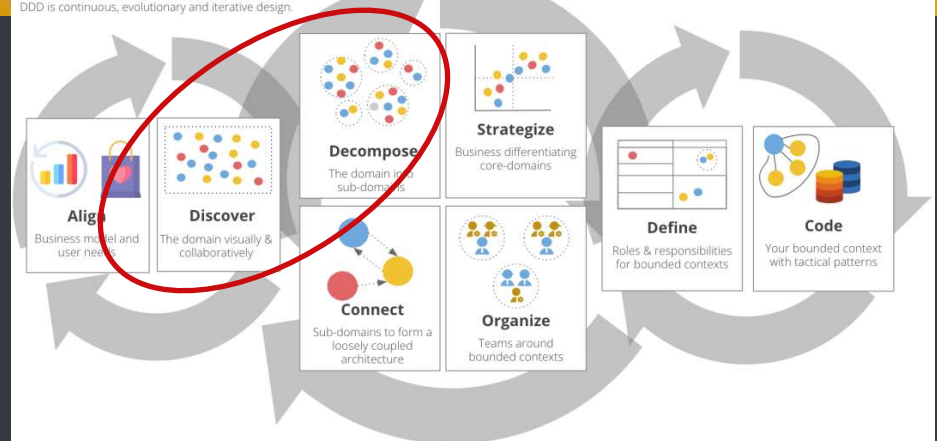
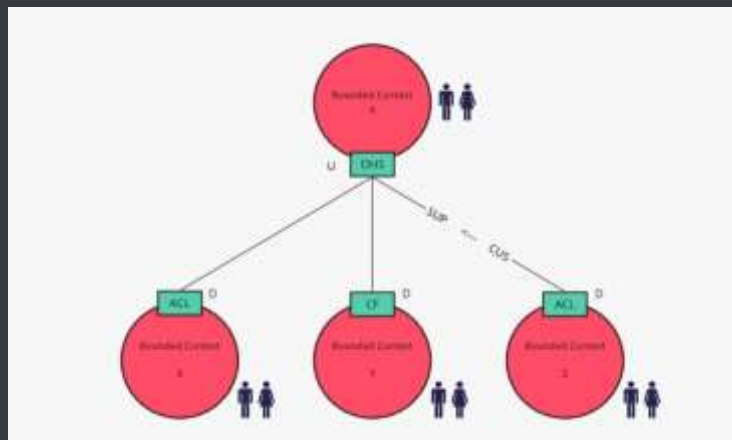
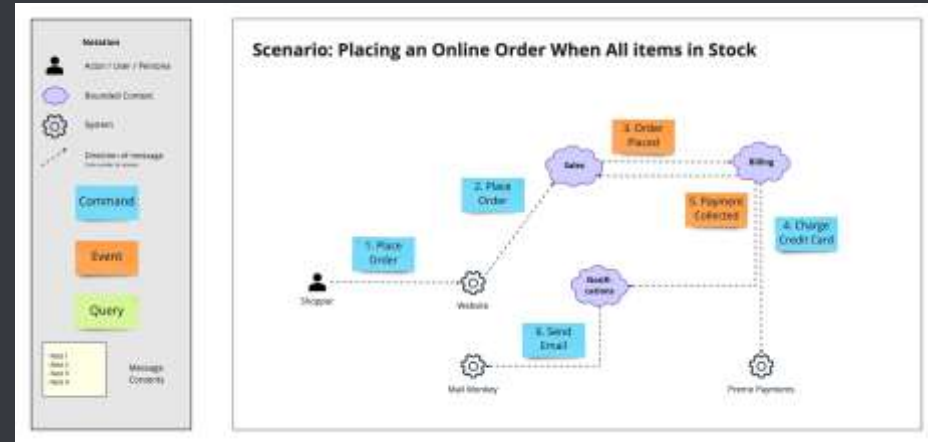
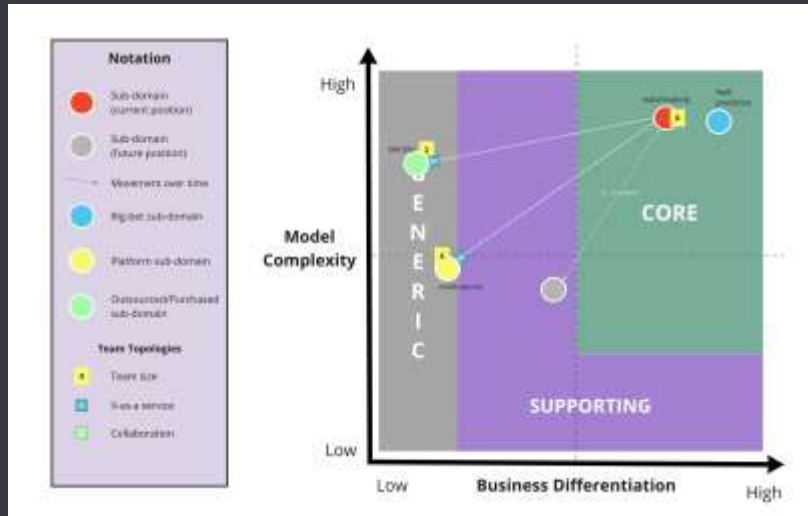


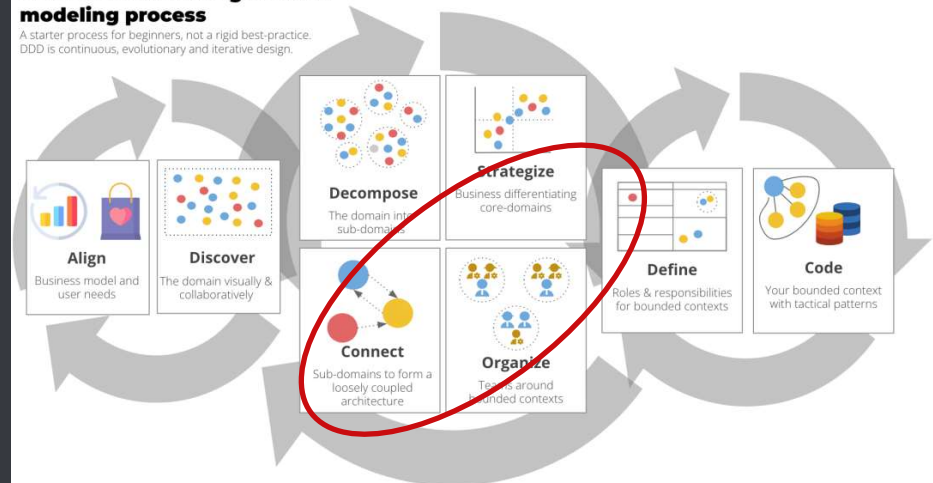
Bild: © Mariusz Gil, <https://raw.githubusercontent.com/mariuszgil/awesome-eventstorming/master/assets/timelapses/timelapse-1.gif>

Workshop 2: Bounded Contexts / selected DDD Crew methods

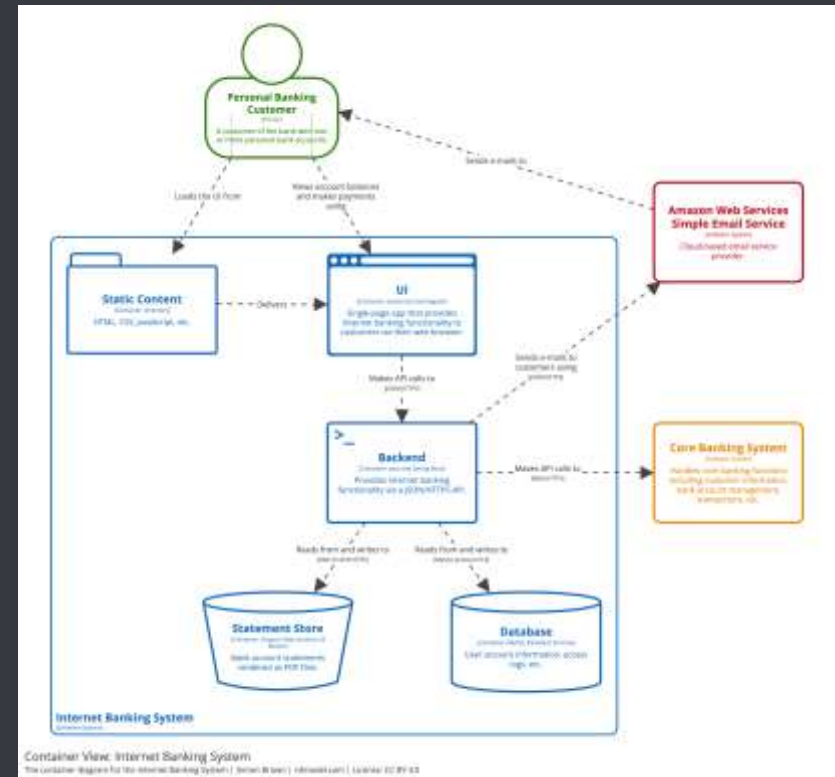
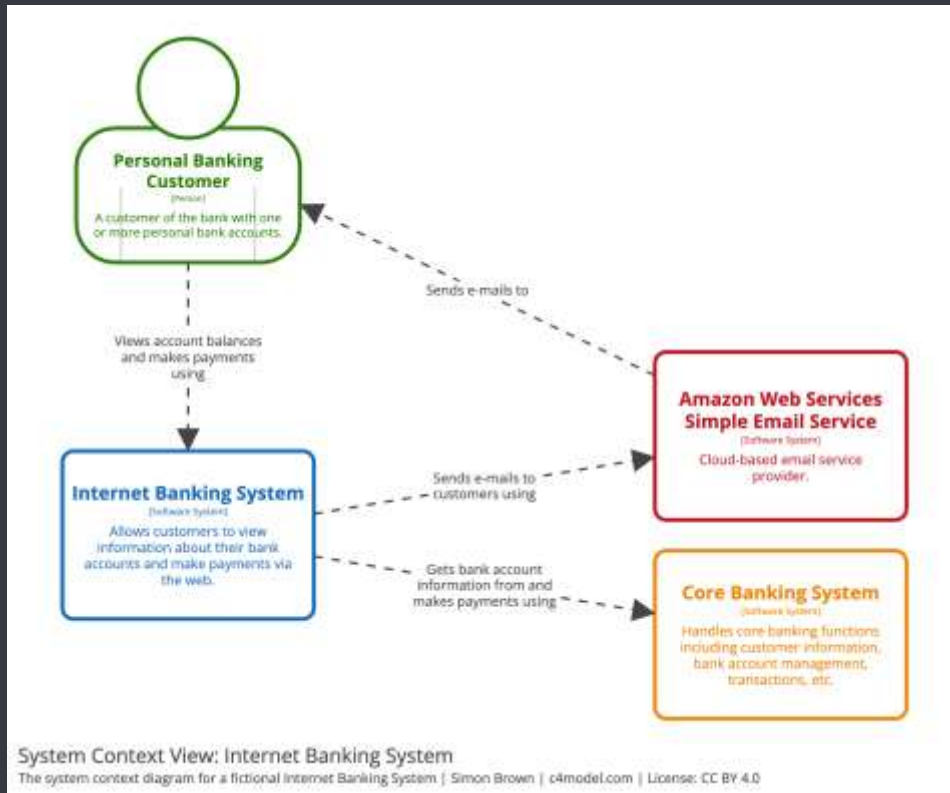


Domain-Driven Design starter modeling process

A starter process for beginners, not a rigid best-practice. DDD is continuous, evolutionary and iterative design.

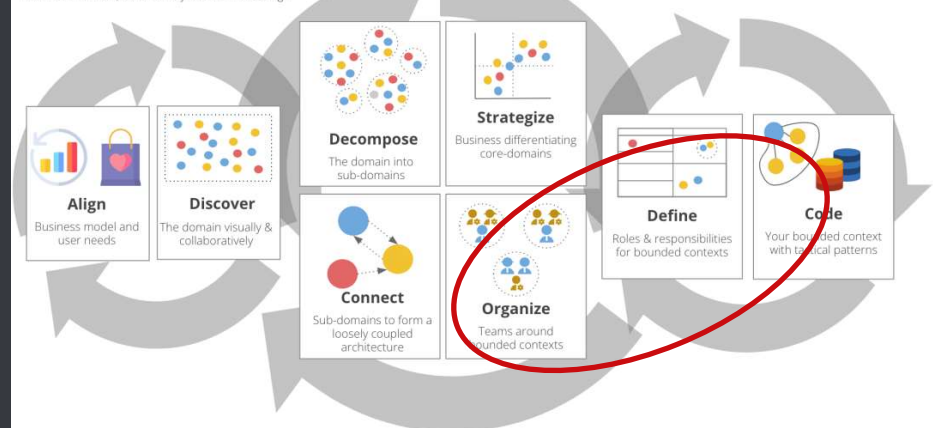


Workshop 3: C4 Model

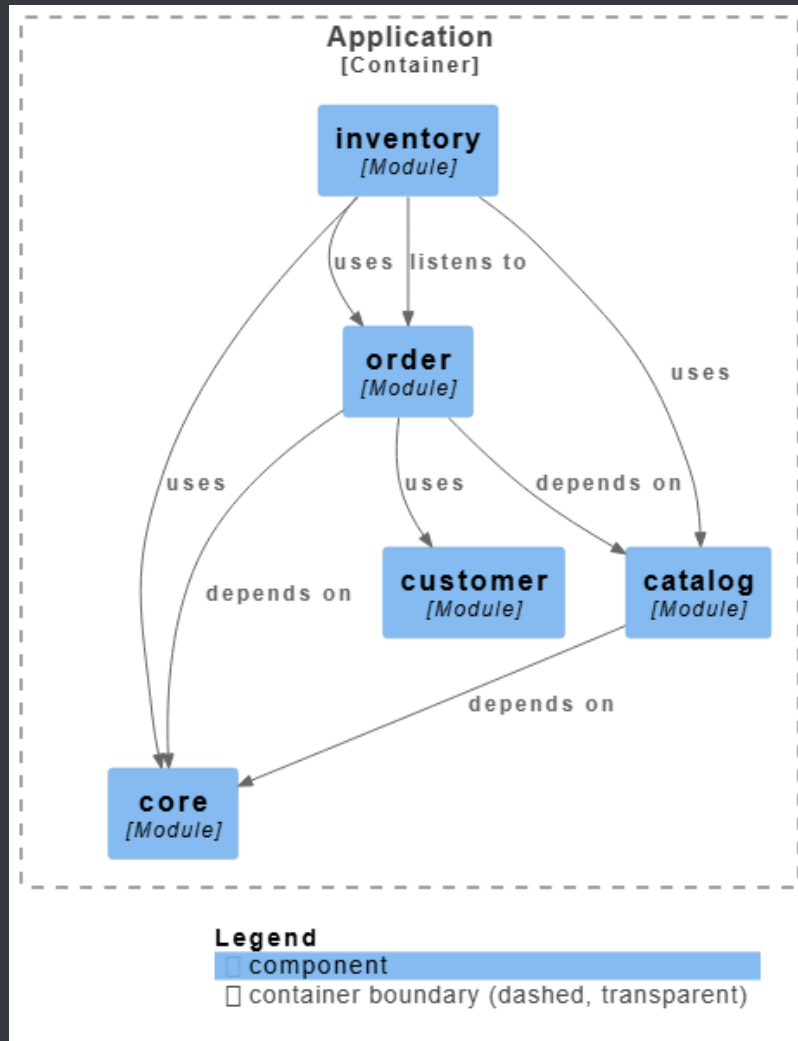


Domain-Driven Design starter modeling process

A starter process for beginners, not a rigid best-practice. DDD is continuous, evolutionary and iterative design.



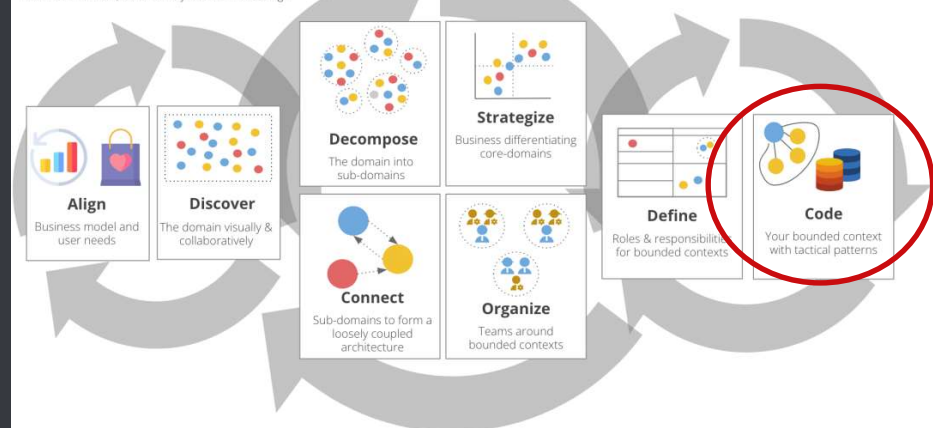
Workshop 4 & following: Implementation



Spring
Modulith

Domain-Driven Design starter modeling process

A starter process for beginners, not a rigid best-practice. DDD is continuous, evolutionary and iterative design.





ArchiLab

www.archi-lab.io

© 2025 Prof. Dr. Stefan Bente