

Sven Müller

Legacy-Modernisierung

mit DDD und Event-Storming

Sven Müller	×
* Architekt * DDD-Enthusiast * Geschäftsführer	

Agenda

synyx

1. Motivation

- 1.1 Was ist Legacy-Software?
- 1.2 Warum DDD?
- 1.3 Stakeholder überzeugen

2. Big-Picture-Event-Storming

- 2.1 Einführung & Vorbereitung
- 2.2 Durchführung Workshops

3. Umsetzung

- 3.1 Strategie
- 3.2 Pitfalls

1.

Motivation

Warum DDD für Legacy-
Modernisierung?

Motivation



1.1 Was ist Legacy-Software?

**Aus dem Leben eines Software-Systems
(Jahrgang ca. 2000)**

Motivation

synyx



Lastenheft

Motivation

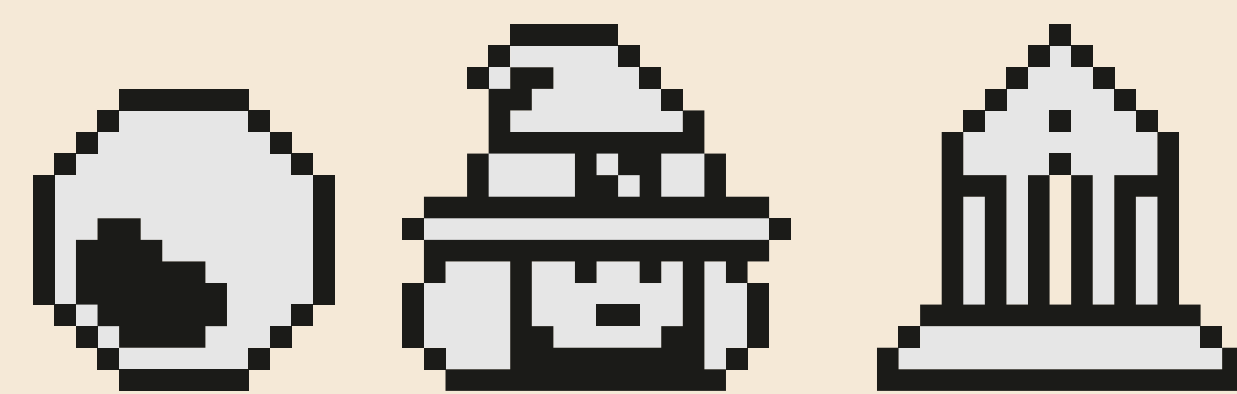
synyx



Pflichtenheft

Motivation

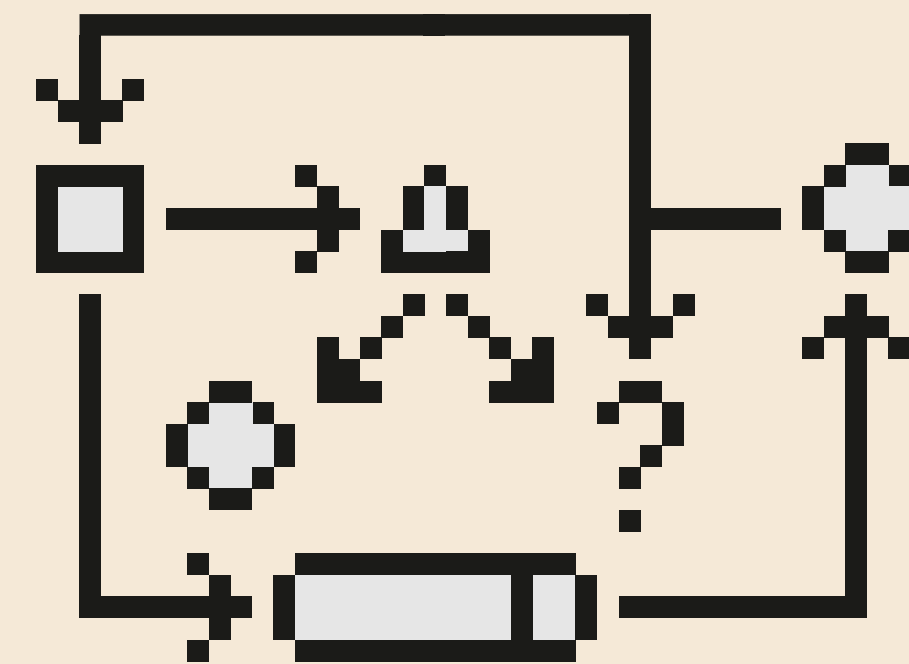
synyx



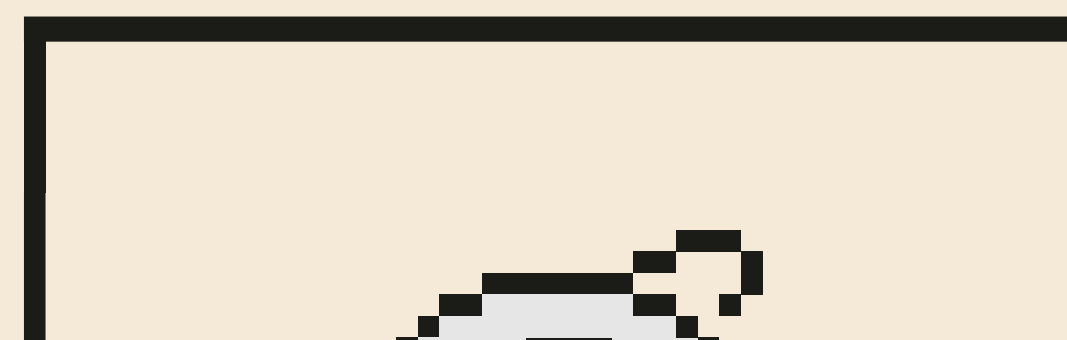
Architekt im
Elfenbeinturm

Motivation

synyx

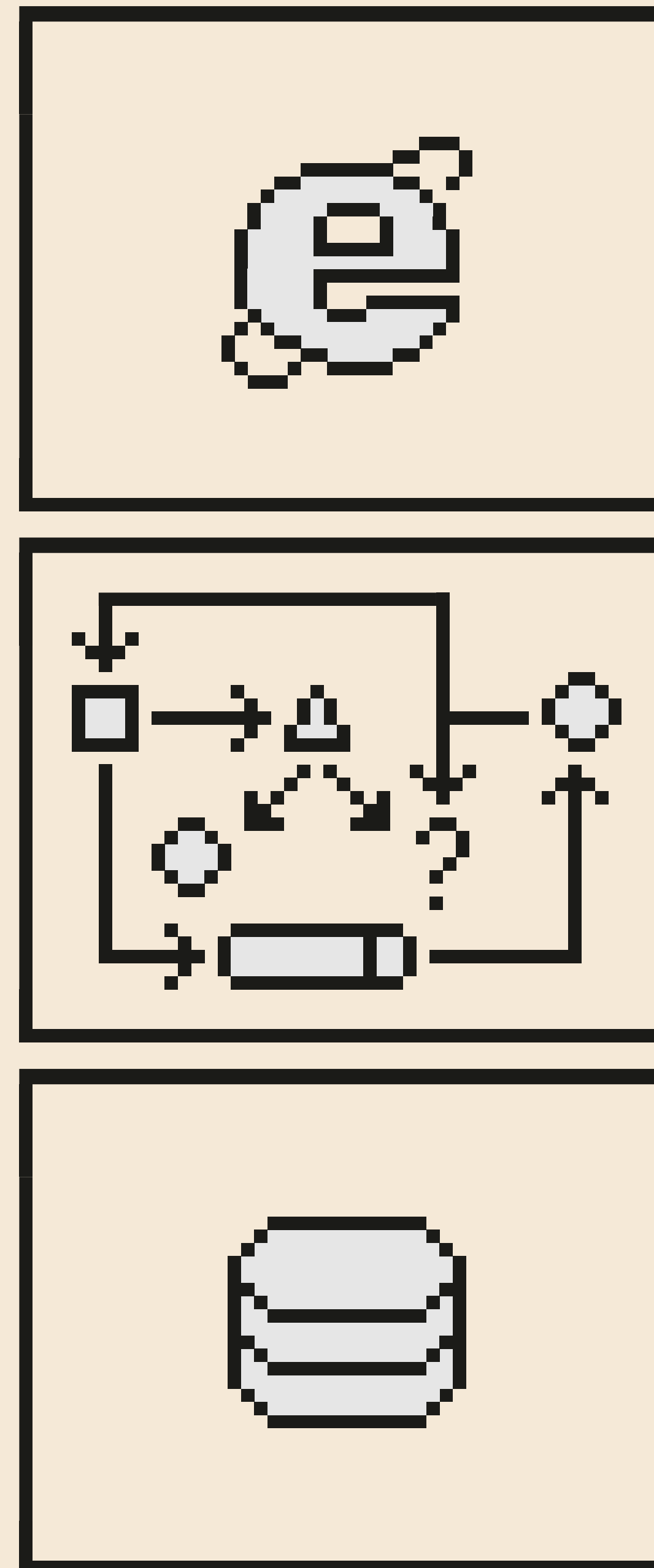


UML



Motivation

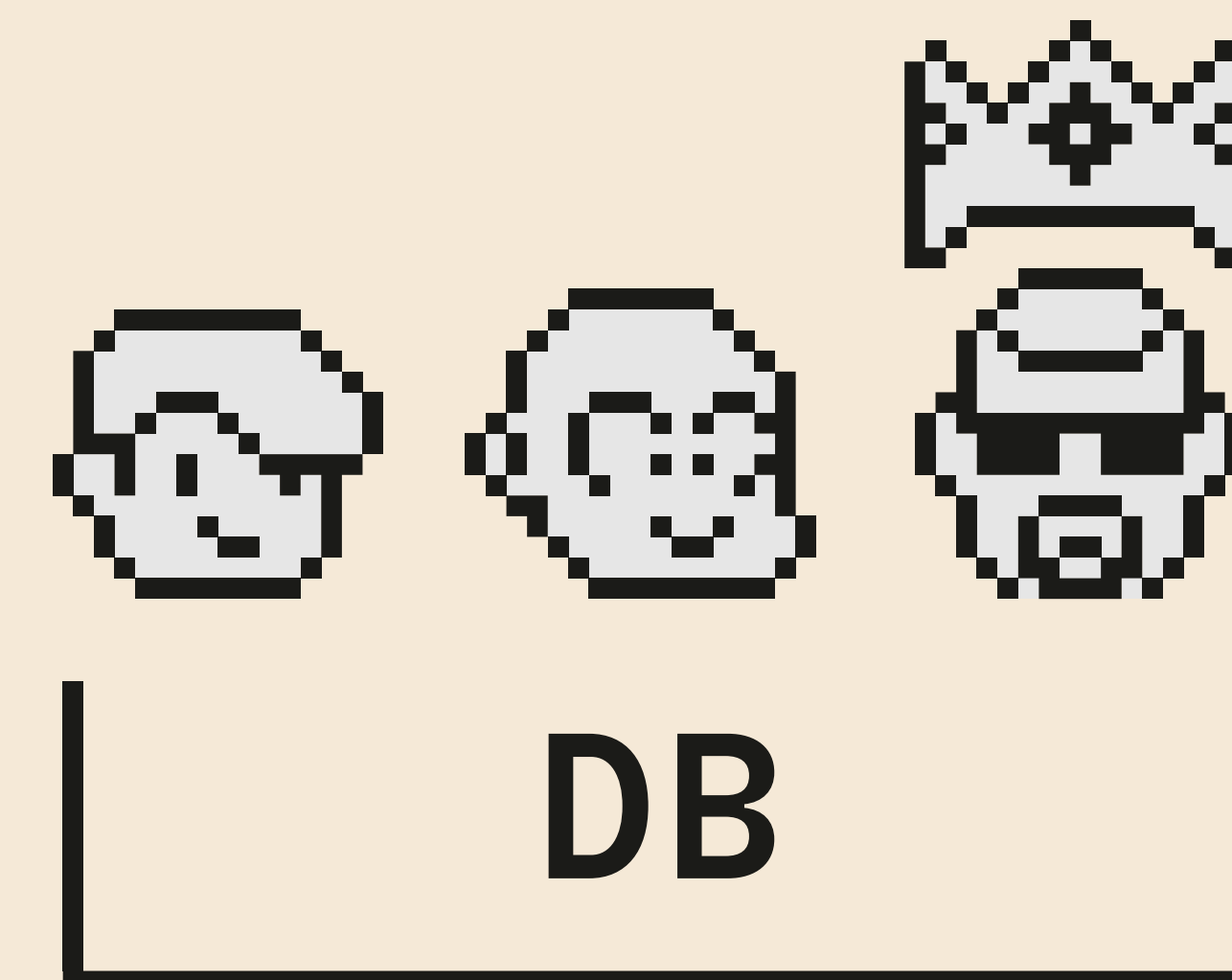
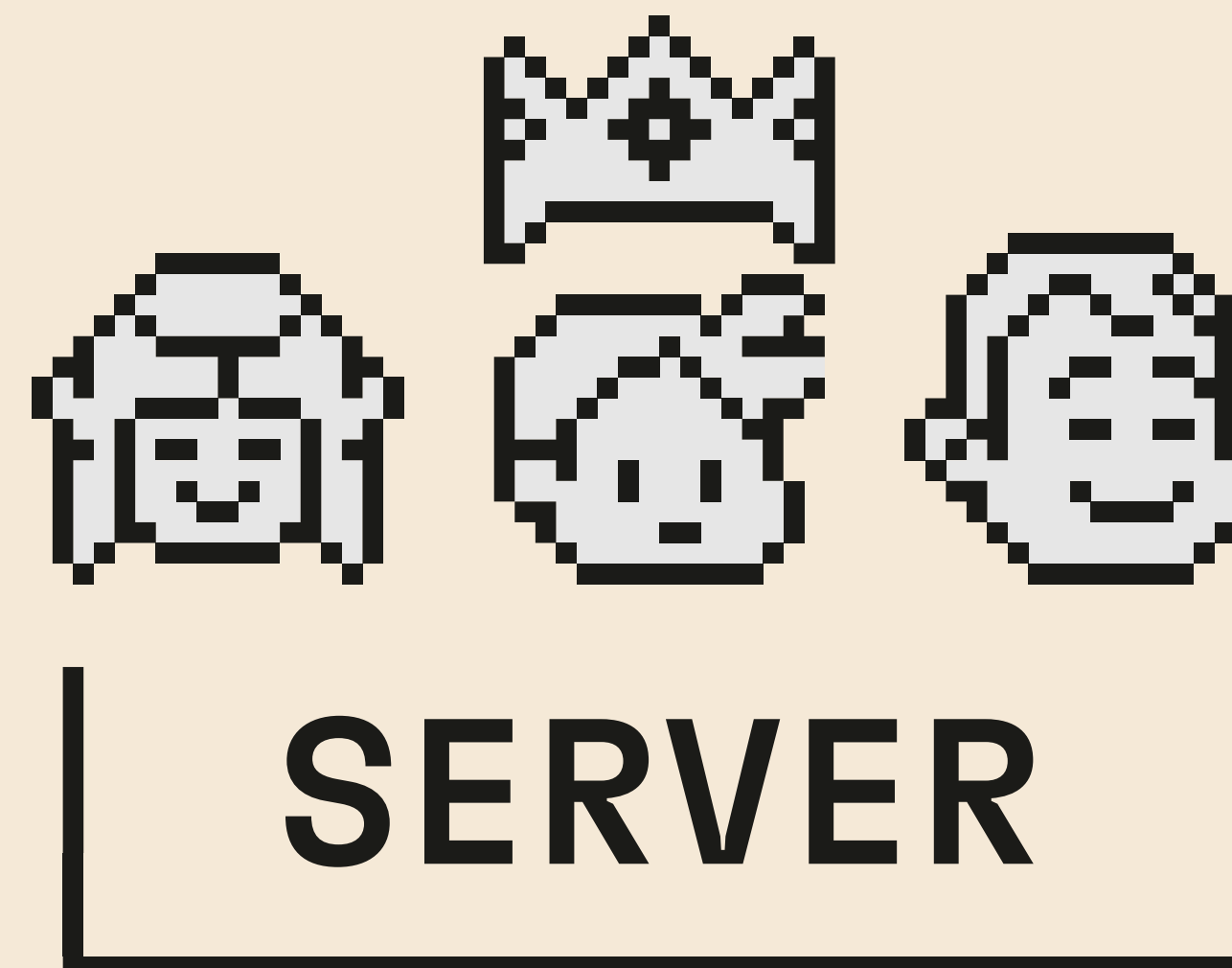
synyx



Schicht- Architektur

Motivation

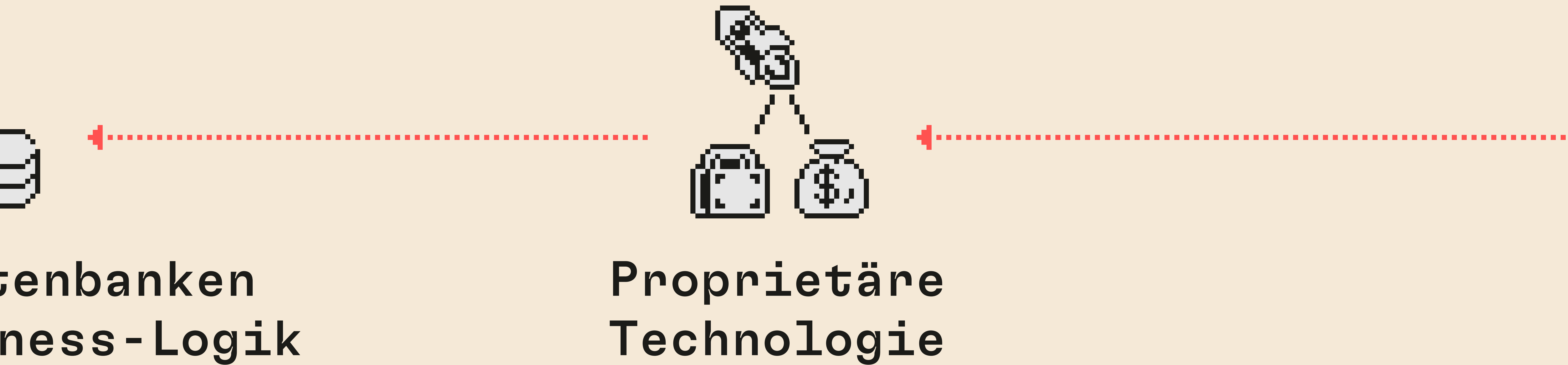
synyx



Silo-Teams

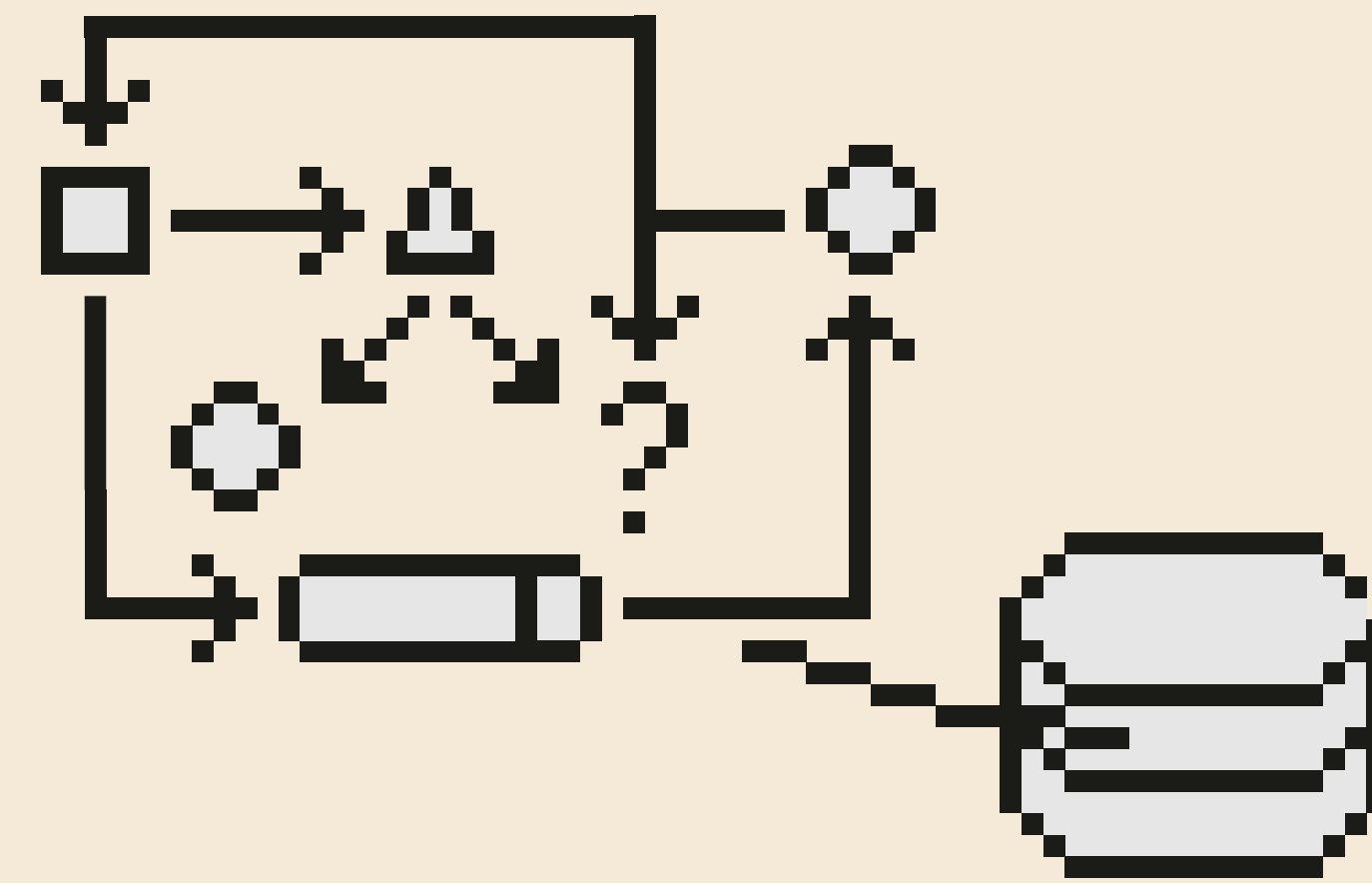
Motivation

synyx

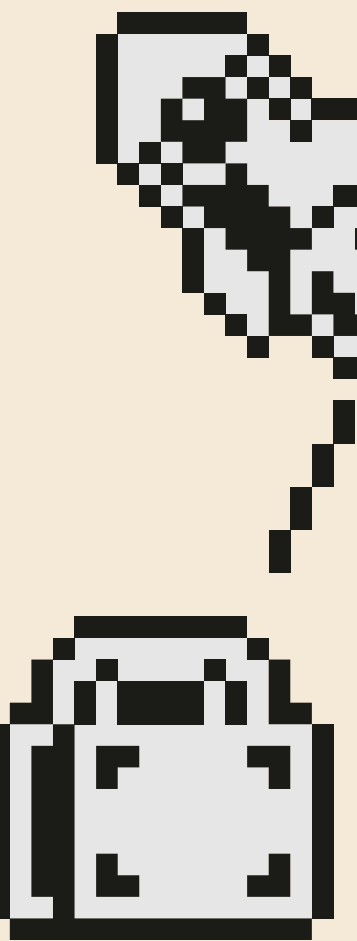
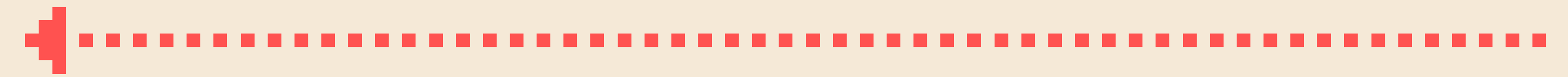


Motivation

synyx



Große Datenbanken
inkl. Business-Logik

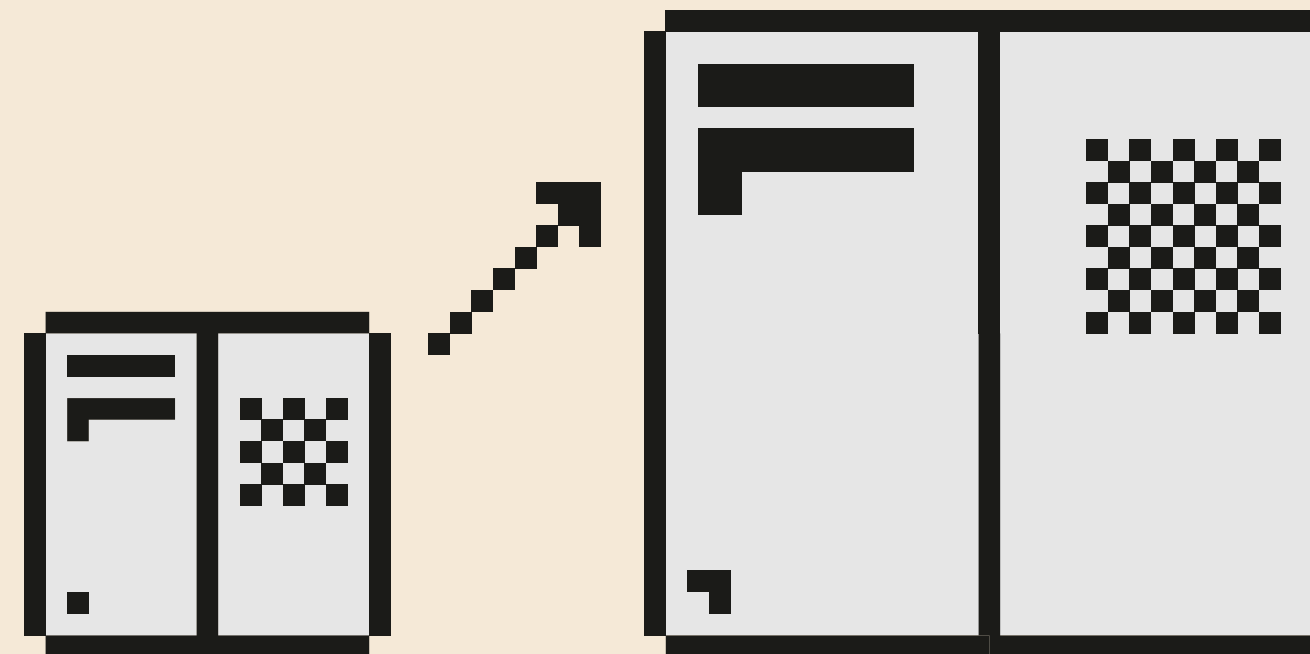


Propriäre
Technologie



Motivation

synyx

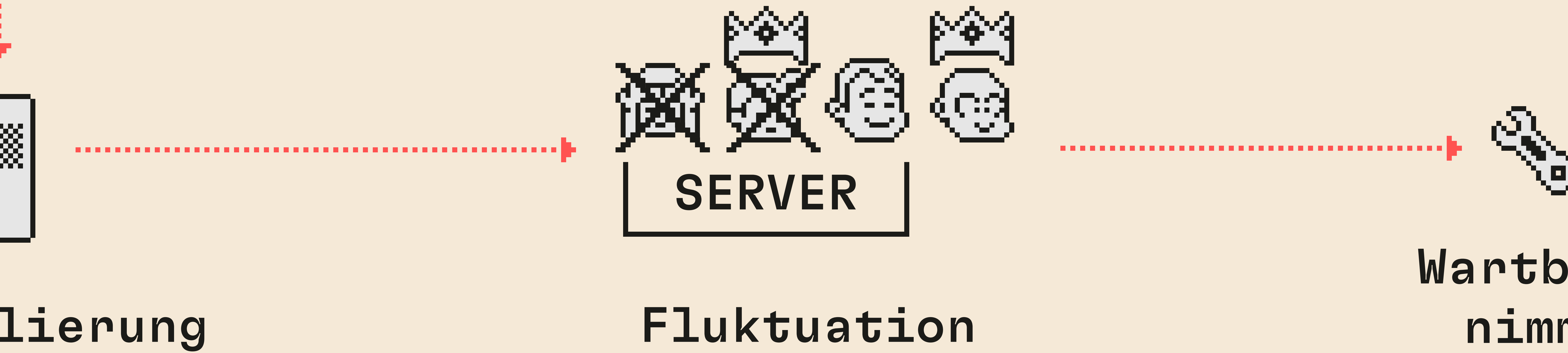


Vertikale Skalierung

F1

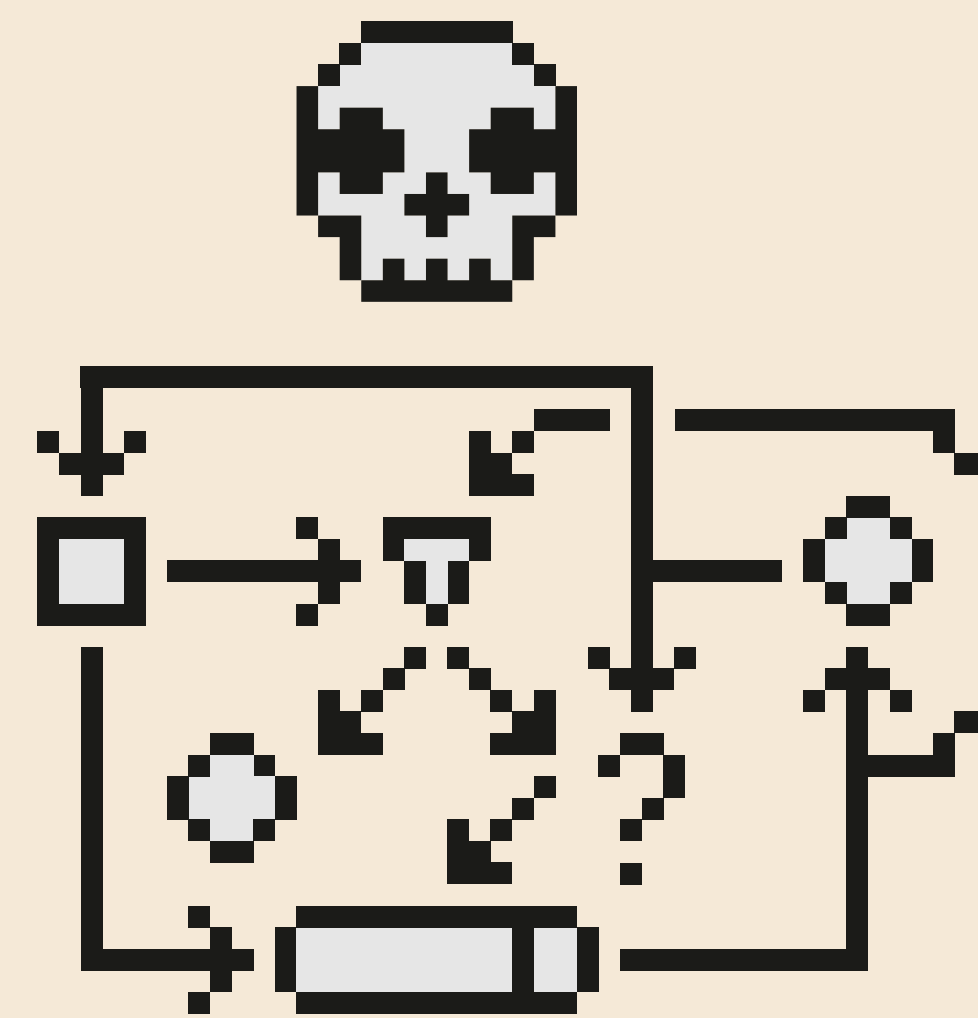
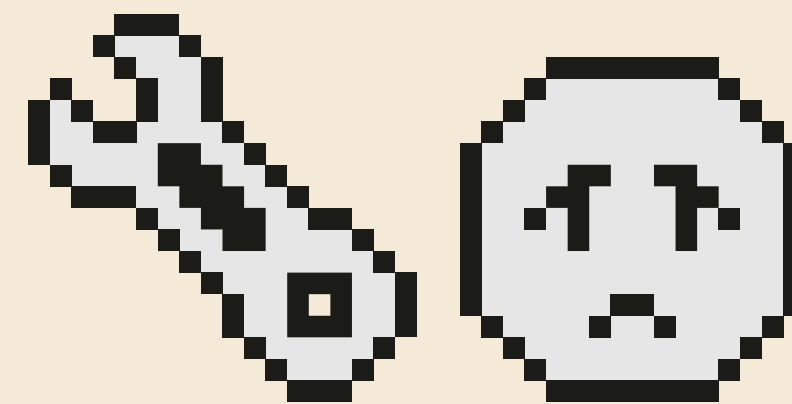
Motivation

synyx



Motivation

synyx



Wartbarkeit
nimmt ab

Architektur ver

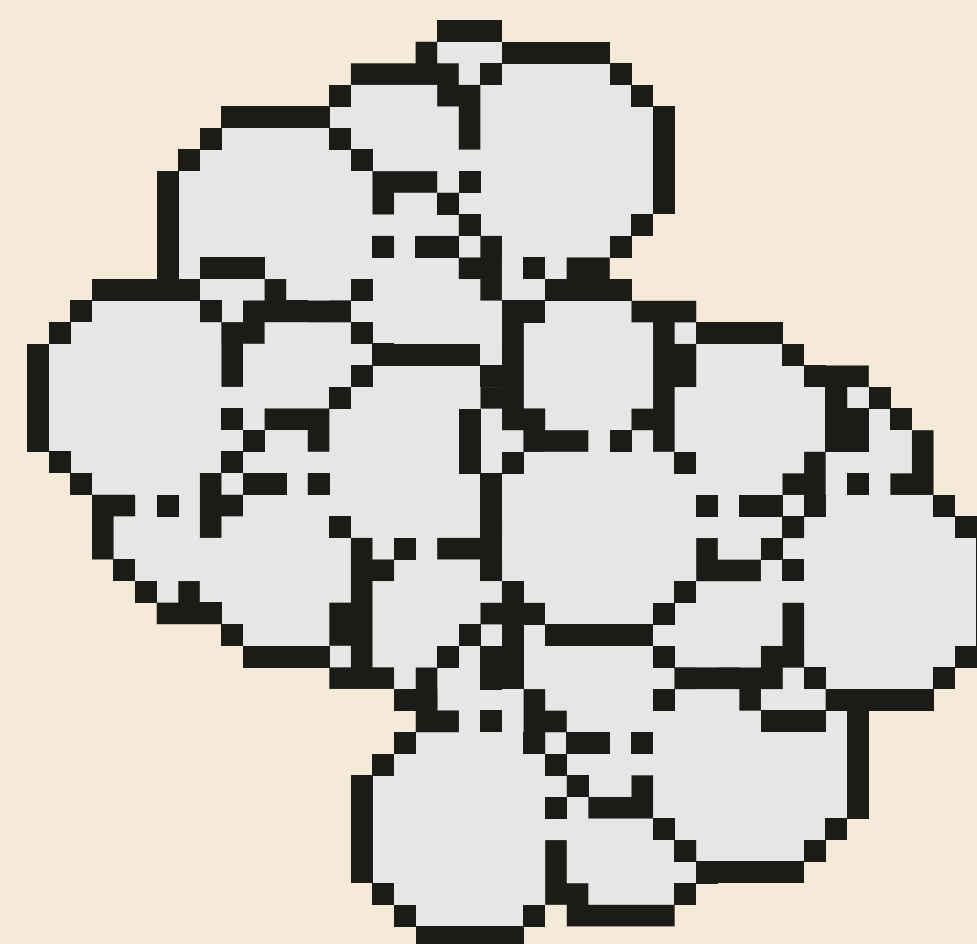
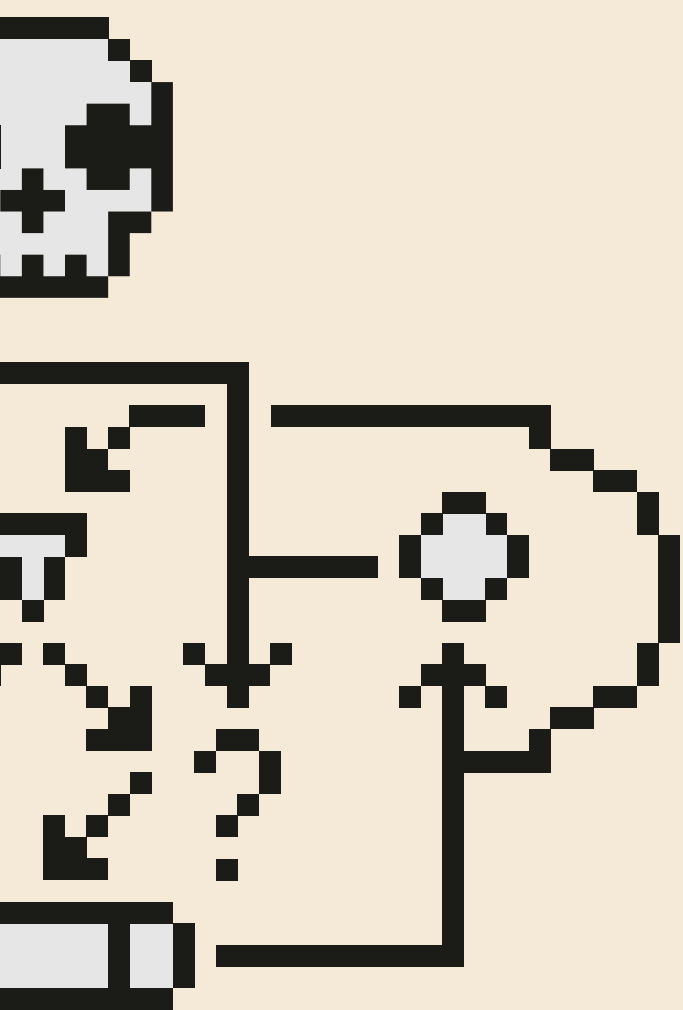
Motivation

synyx



Motivation

synyx



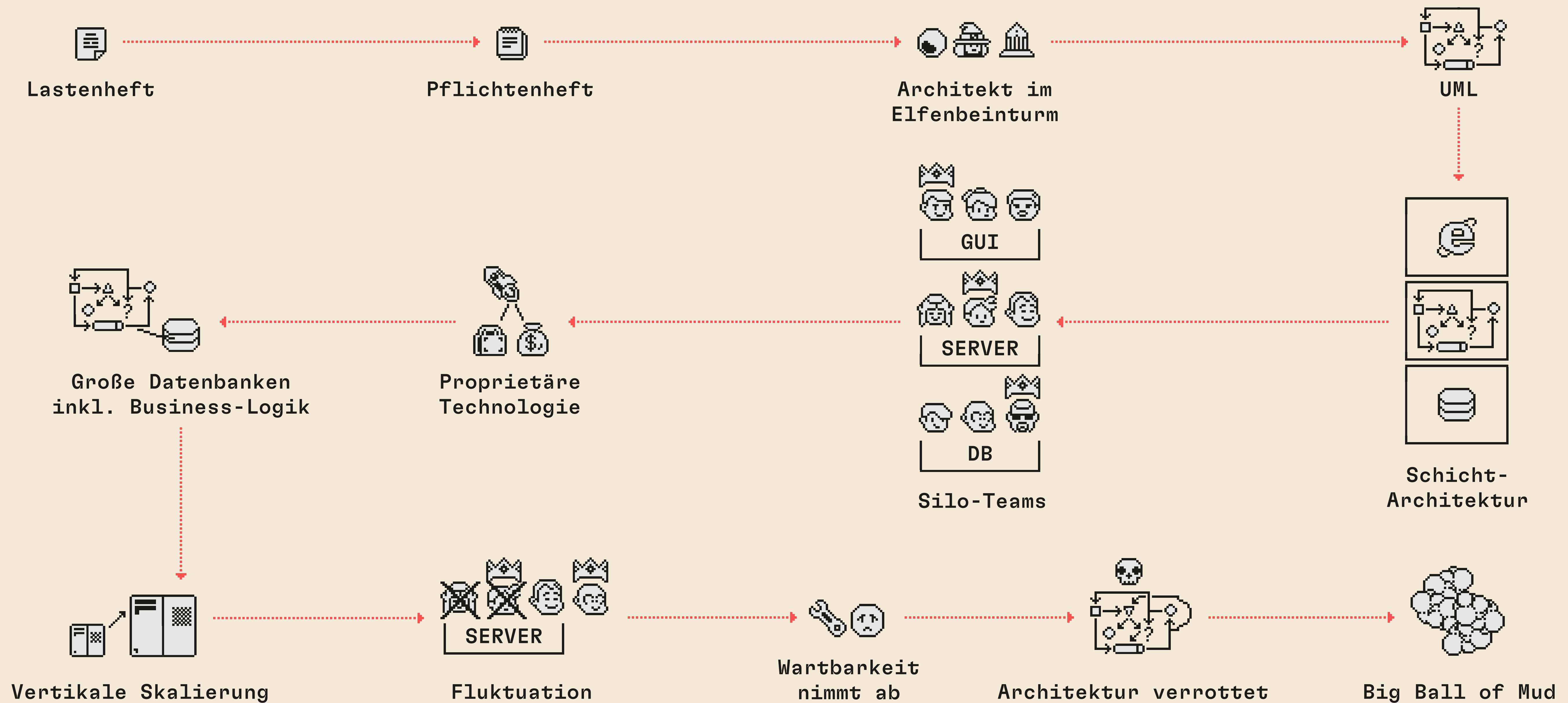
r verrottet

Big Ball of Mud

Motivation

synyx

Aus dem Leben eines Software-Systems
(Jahrgang ca. 2000)



1.2 Definition DDD



- * Fachlichkeit soll Treiber von Software-Design und Architektur sein
- * Create collaboration of software experts and domain experts
- * Domain-Driven Design is an approach to the development of complex software in which we:
 - Focus on the core domain.
 - Explore models in a creative collaboration of domain practitioners and software practitioners.
 - Speak a ubiquitous language within an explicitly bounded context.

1.2 Warum DDD?



- * Oftmals keine Original-Entwickler & keine Fachexperten mehr da
 - Fachlichkeit UND Code wird von Menschen maintained, die bei der Entstehung nicht dabei waren
- * Gemeinsames Domain-Wissen und Ziel-Architektur notwendig!

1.3 Stakeholder überzeugen

×

Management überzeugen

- * Weniger Wartungsaufwand
- * Schnellere Feature-Entwicklung
- * Moderner Technik-Stack besser verkäuflich
- * Klare Domänen-Schnitte ermöglichen...
 - fundierte Make-or-Buy-Entscheidungen
 - gezielten Ressourcen-Einsatz am Kern der Wertschöpfungskette

1.3 Stakeholder überzeugen

×

Entwickler überzeugen

- * Technische Schulden abbauen
- * Legacy-Tech-Stack loswerden
- * Kollaboratives Software-Design sorgt für...
 - UNFASSBAR VIEL mehr Spaß, Software zu entwickeln, wenn Entwicklung und Fachbereich das gleiche Verständnis von der Domain haben
 - Keinen Frust beim rumwursteln in Alt-Code, den keiner kapiert

1.3 Stakeholder überzeugen

×

Fachexperten überzeugen

- * Herausfordernd, aber essenziell
- * Fachexperten lernen...
 - wie moderne Software-Entwicklung funktioniert
 - wie sie unmittelbar Einfluss auf das entstehende Software-System nehmen können

2.

Big Picture Event Storming

Big Picture Event Storming

synyx

2.1 Einführung & Vorbereitung

×

- * Leichtgewichtige Methode zum Modellieren von Geschäftsprozessen
- * Sehr inklusiv: Menschen aus allen Bereichen der Organisation können gemeinsam einen ersten Entwurf für die Architektur eines Software-Systems gestalten
- * Bonus: Auch sehr gut geeignet, um ein gemeinsames Verständnis der Abläufe in einem bestehenden System (oder auch Geschäftsprozess!) zu erarbeiten
- * Lebende Dokumentation, die nie fertig ist: muss agil und iterativ entstehen!

Big Picture Event Storming

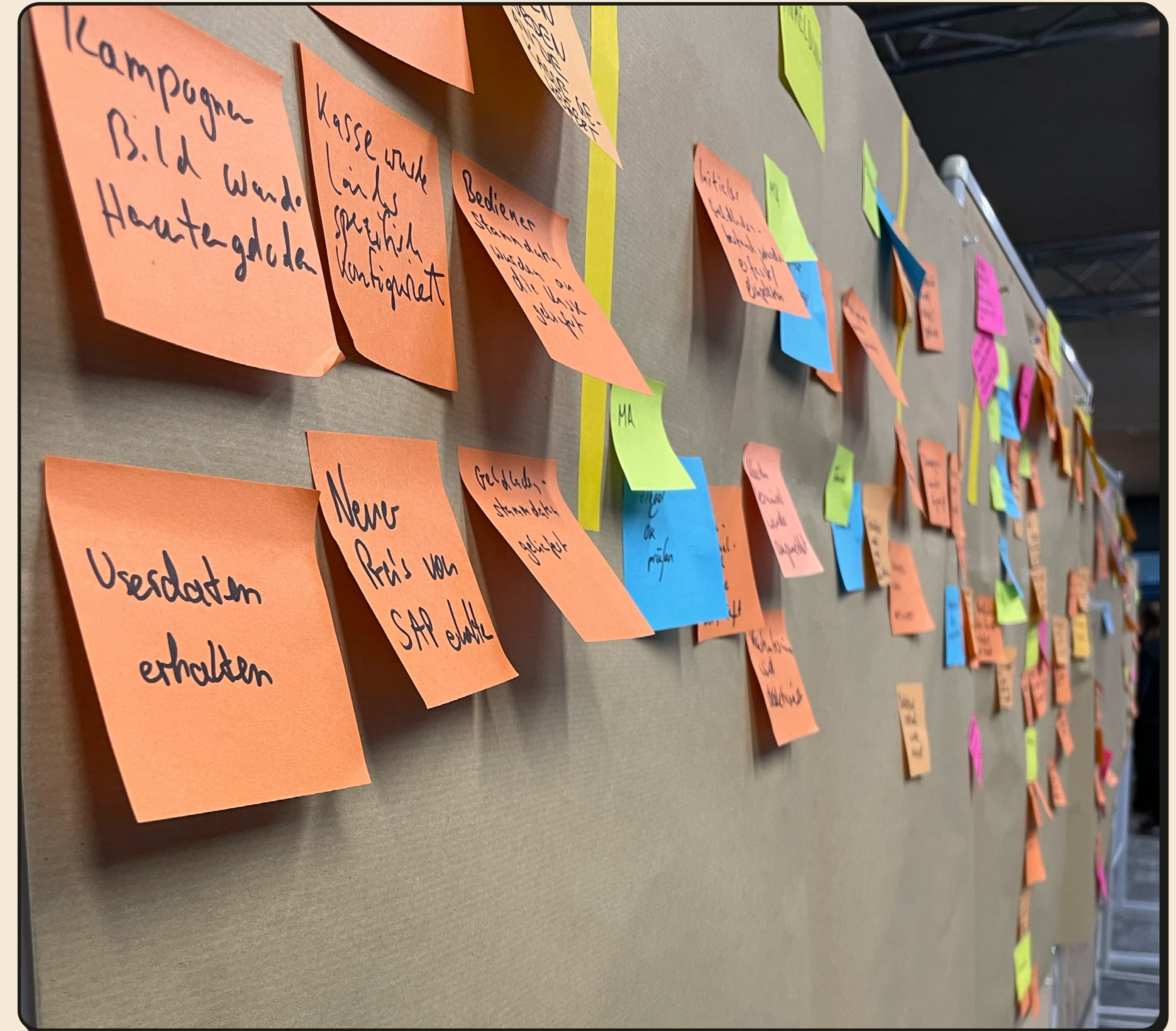
synyx

2.1 Einführung & Vorbereitung

x

Zeitplanung

- * 5 Halbtages-Sessions für große Software-Systeme
- * Mehrere Tage Pause zwischen den Sessions!
- * Möglichst alle Termine im Vorfeld planen



Big Picture Event Storming

synyx

2.1 Einführung & Vorbereitung

×

Teilnehmer

- * Optimal ca. 10-15 Personen
- * Idealerweise mindestens 50% Fachexperten!



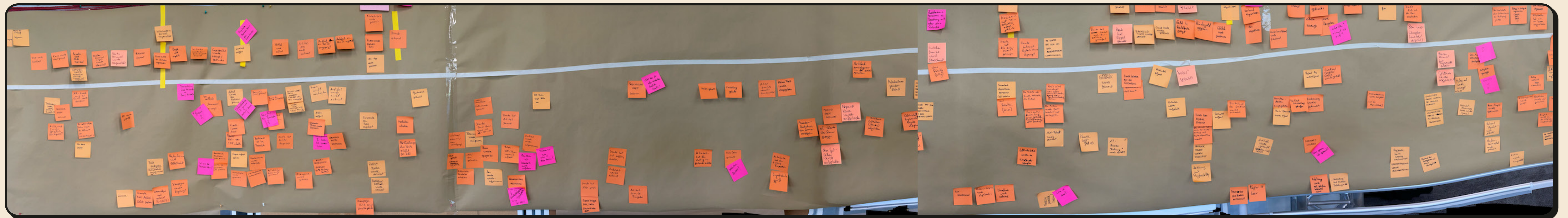
Big Picture Event Storming

2.1 Einführung & Vorbereitung

×

Location

- * Mindestens 5m, besser 10m breite Wand
- * Notfalls Whiteboards zusammenstellen



Big Picture Event Storming

synyx

2.1 Einführung & Vorbereitung

×

Material

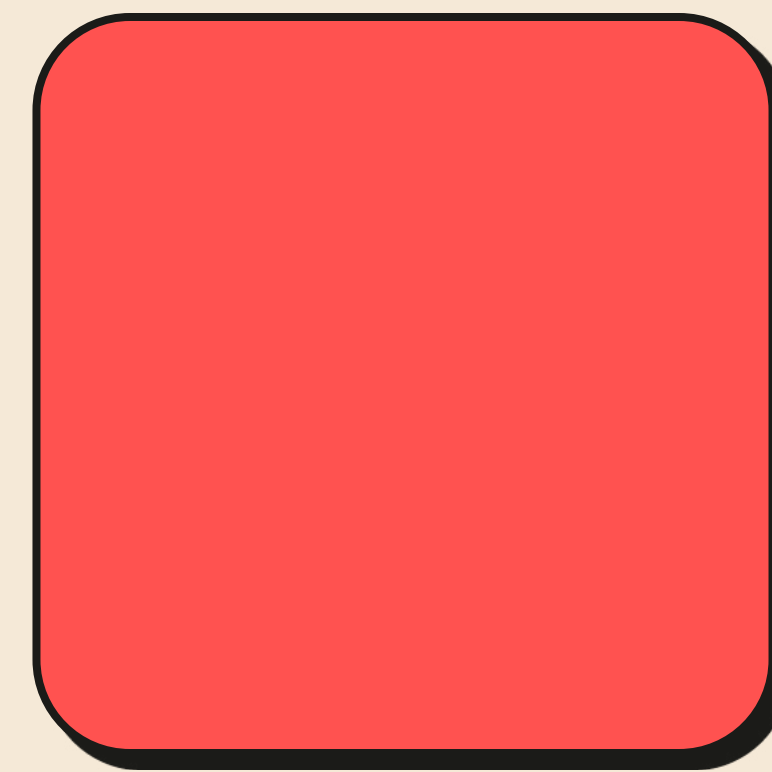
- * Große Rolle Plotterpapier als Modelling Surface
- * Super Sticky Post-Its (Viel Orange, Pink, Blau, Grün, ...)
- * Stabilo Pen 68 (schwarz)
- * Krepp-Band

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 1



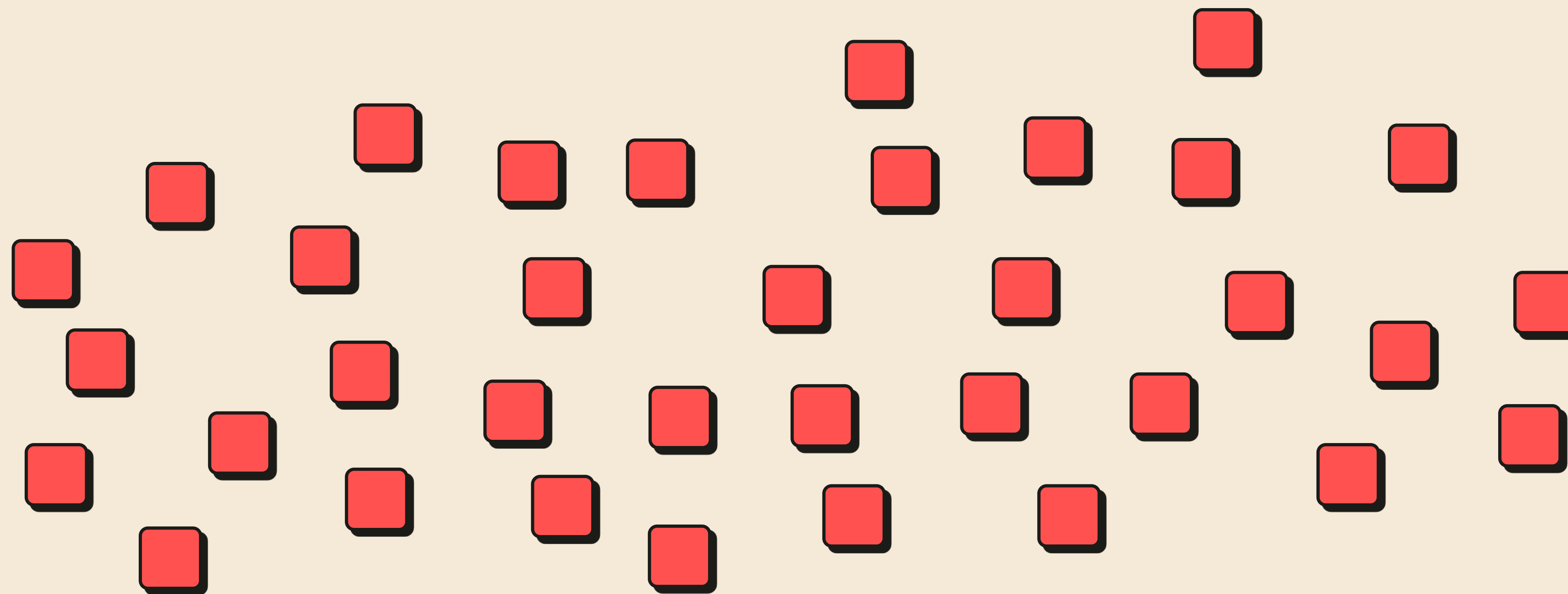
Getting started

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 1



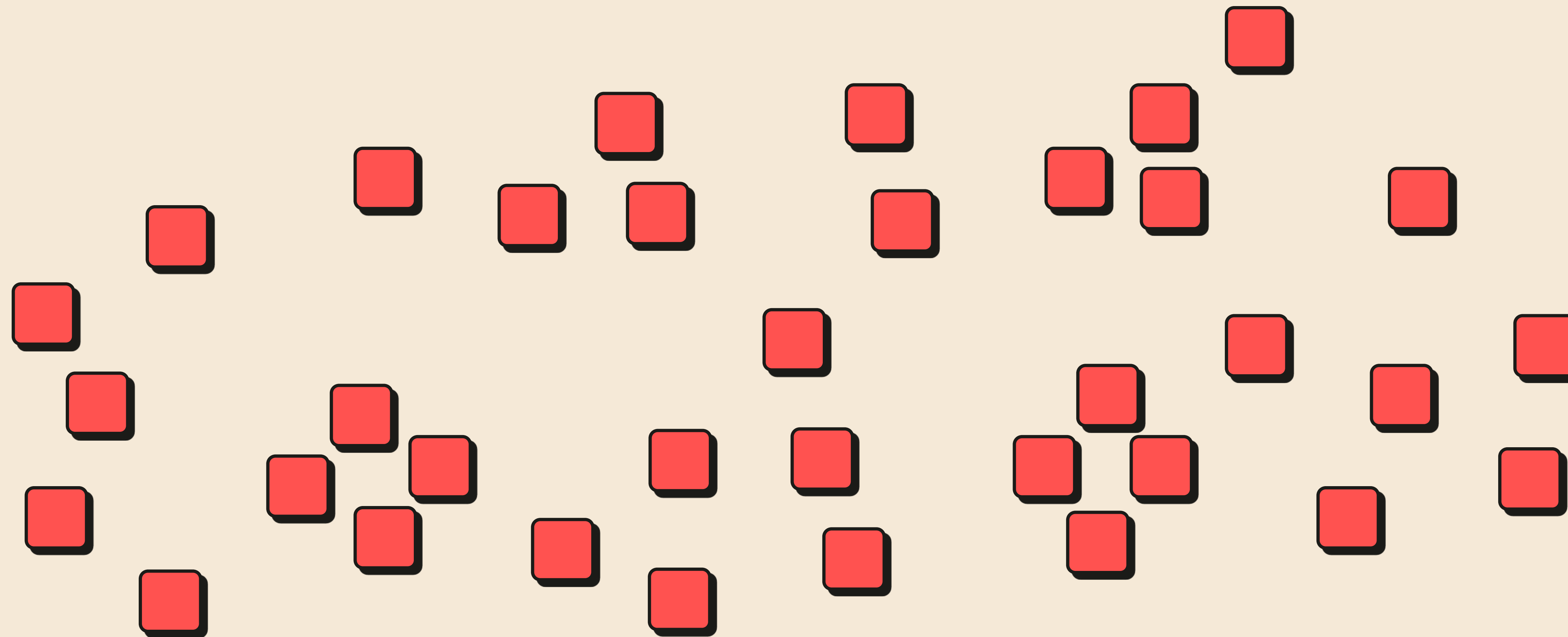
Chaotic Exploration

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 1



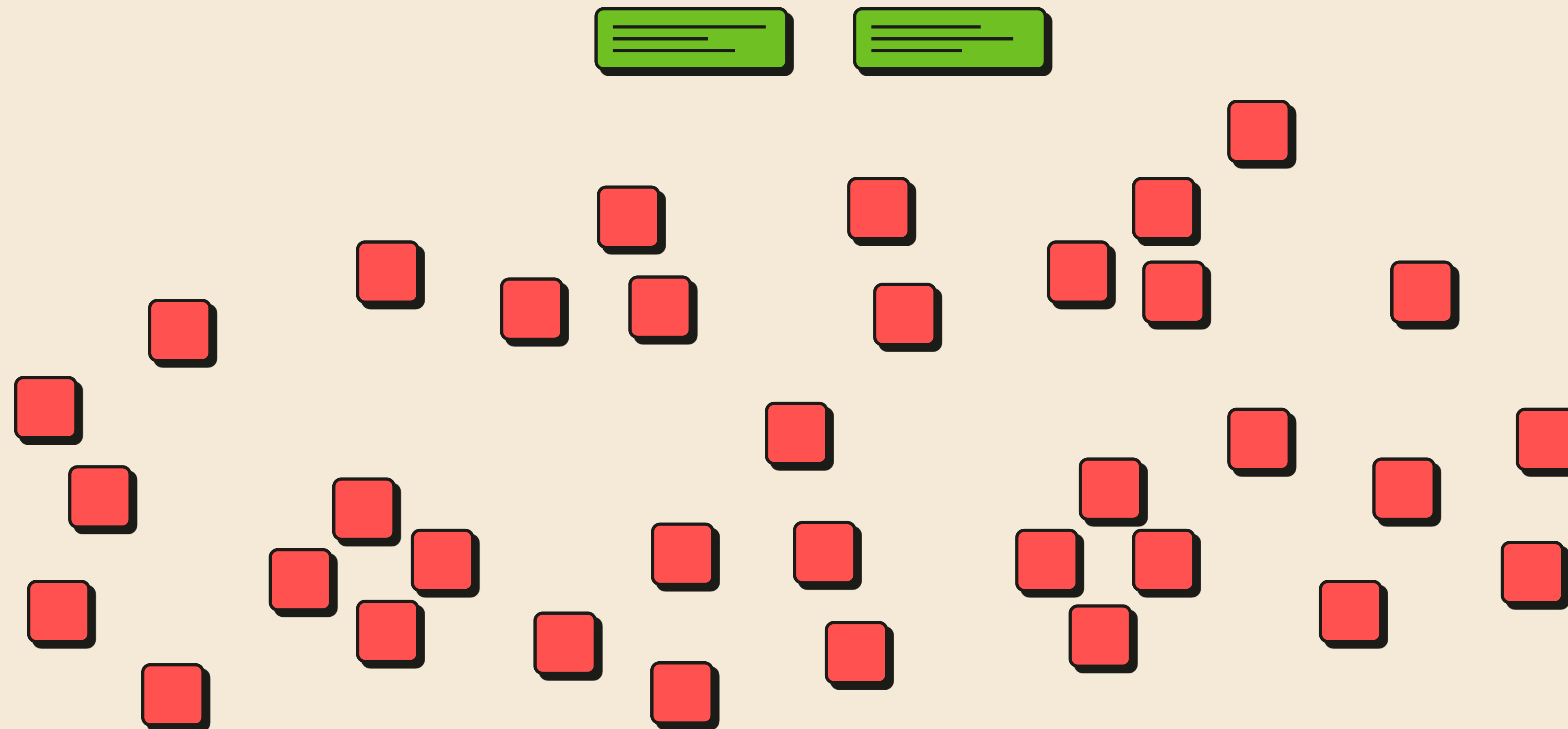
Duplikate auflösen

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 1



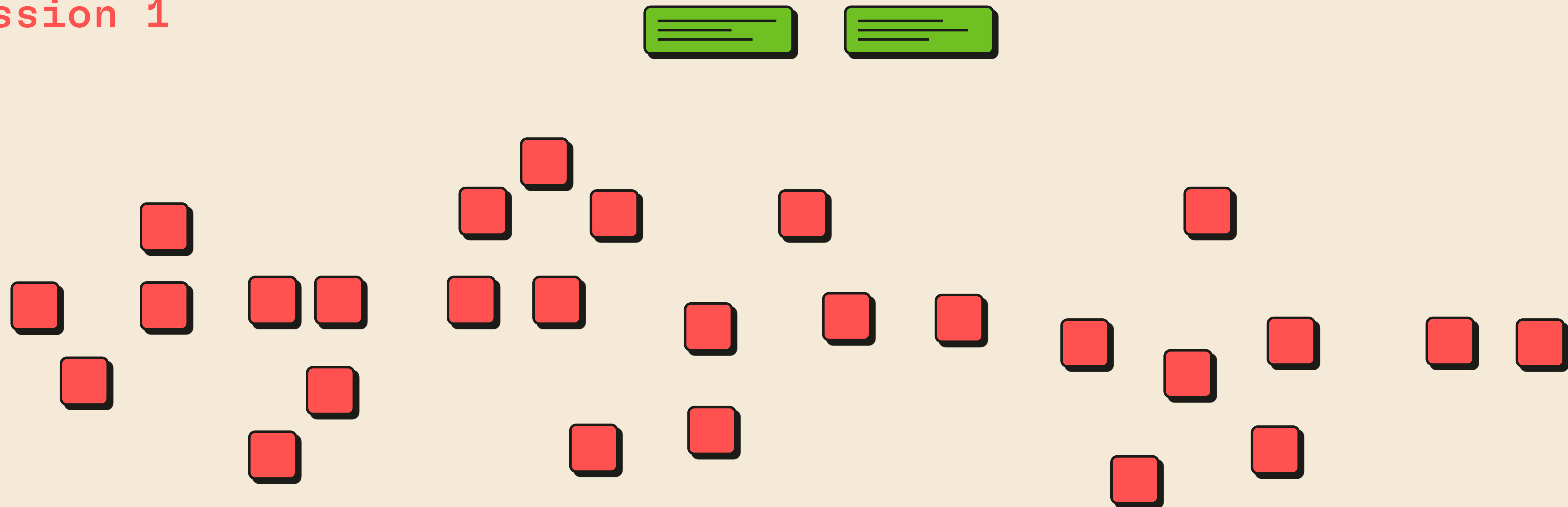
Begriffsdefinitionen festhalten

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 1



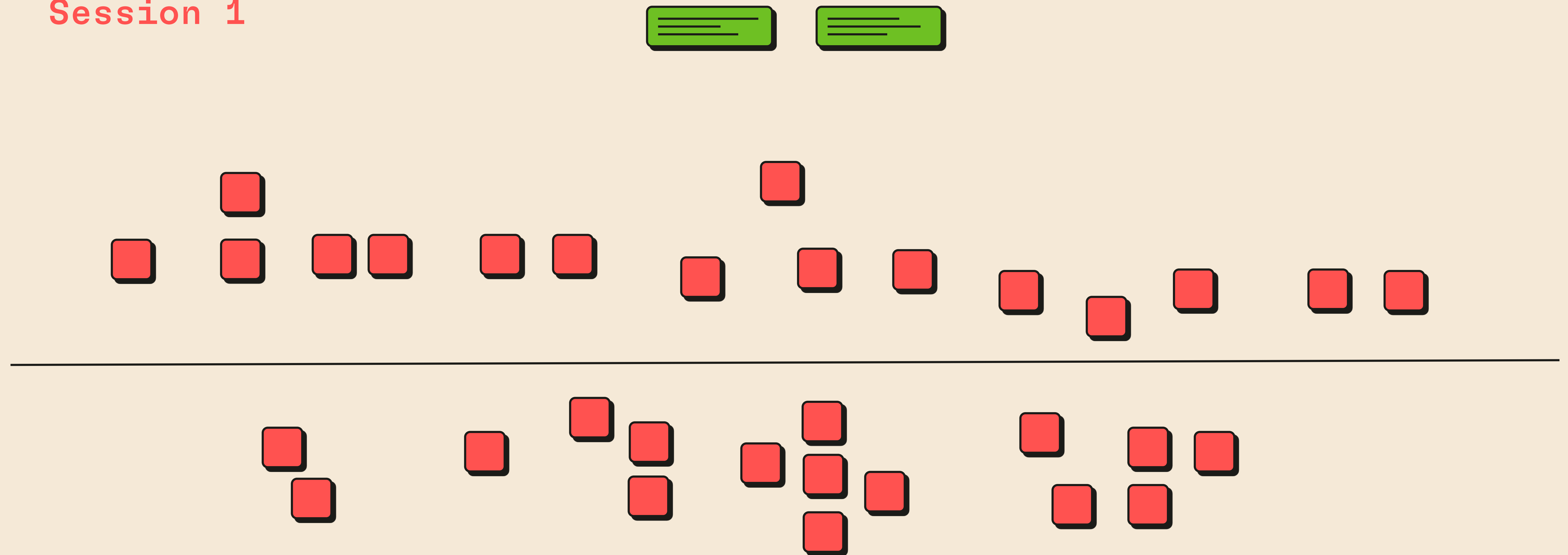
Enforcing the timeline

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 1



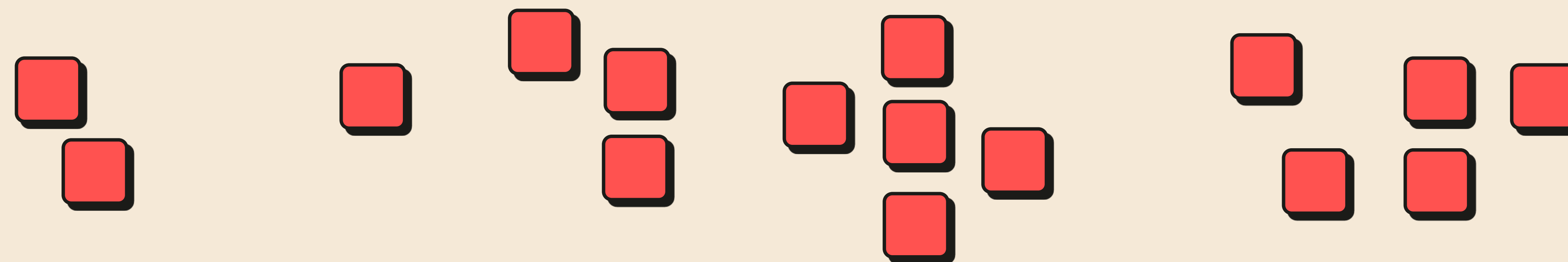
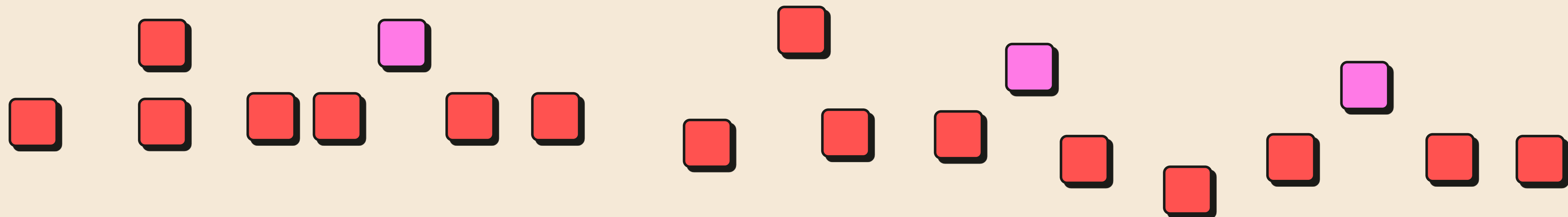
Happy Path

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 1



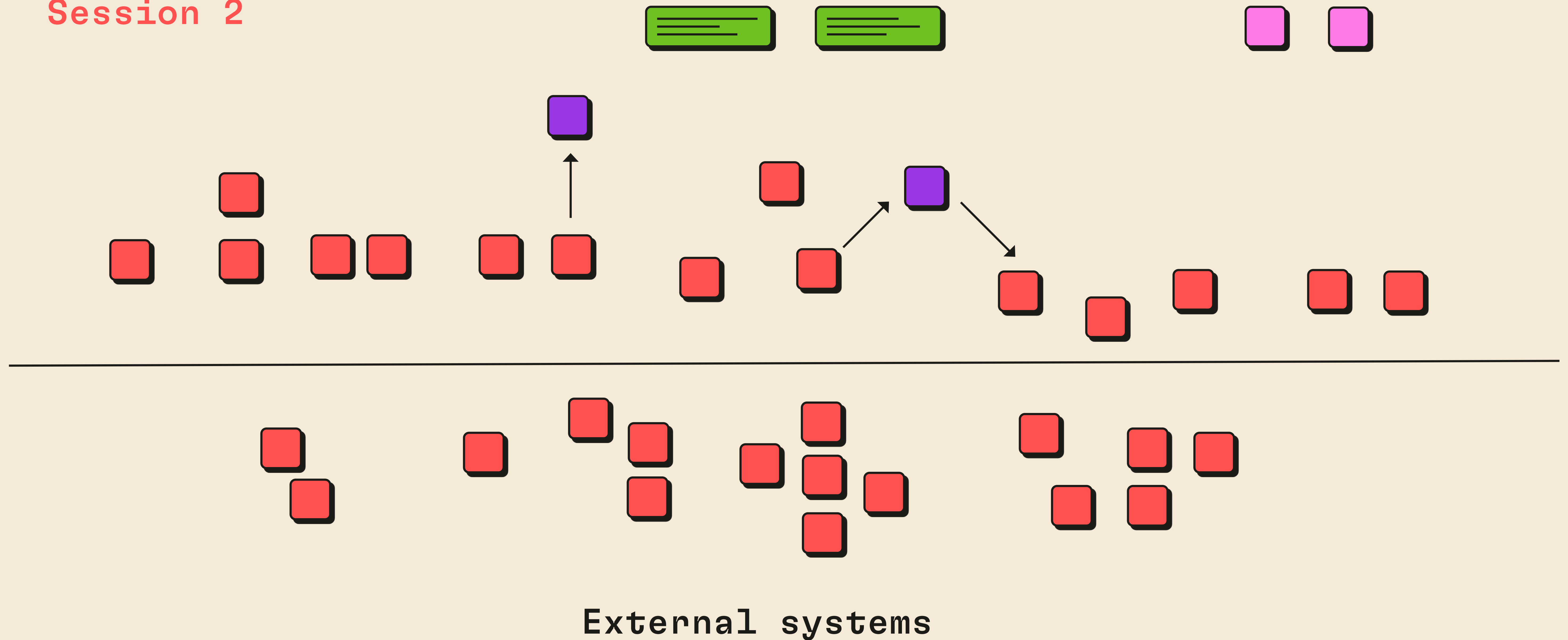
Hotspots

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 2

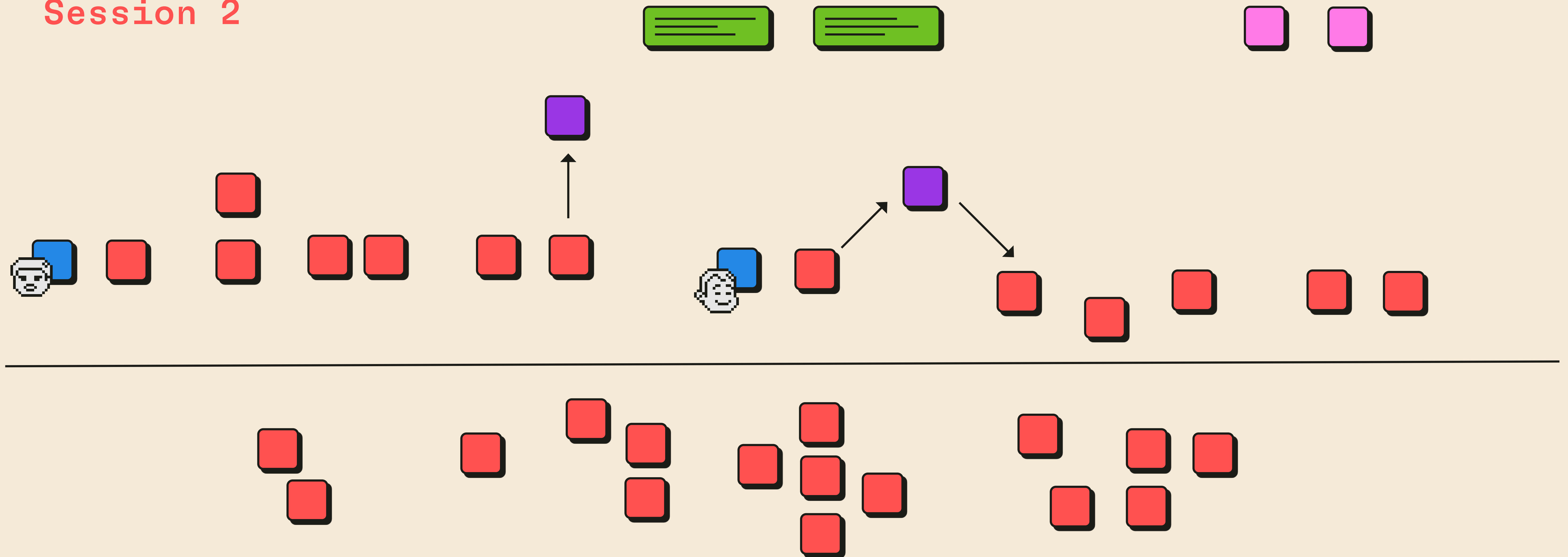


Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 2



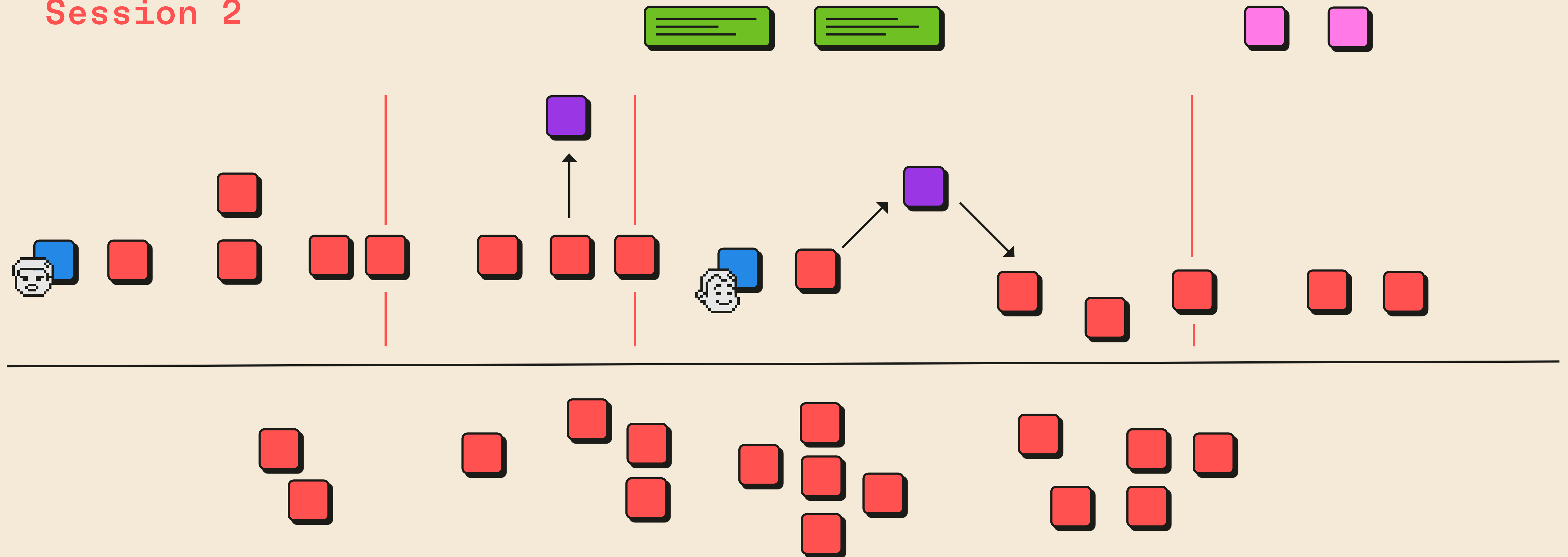
Commands & Actors

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 2



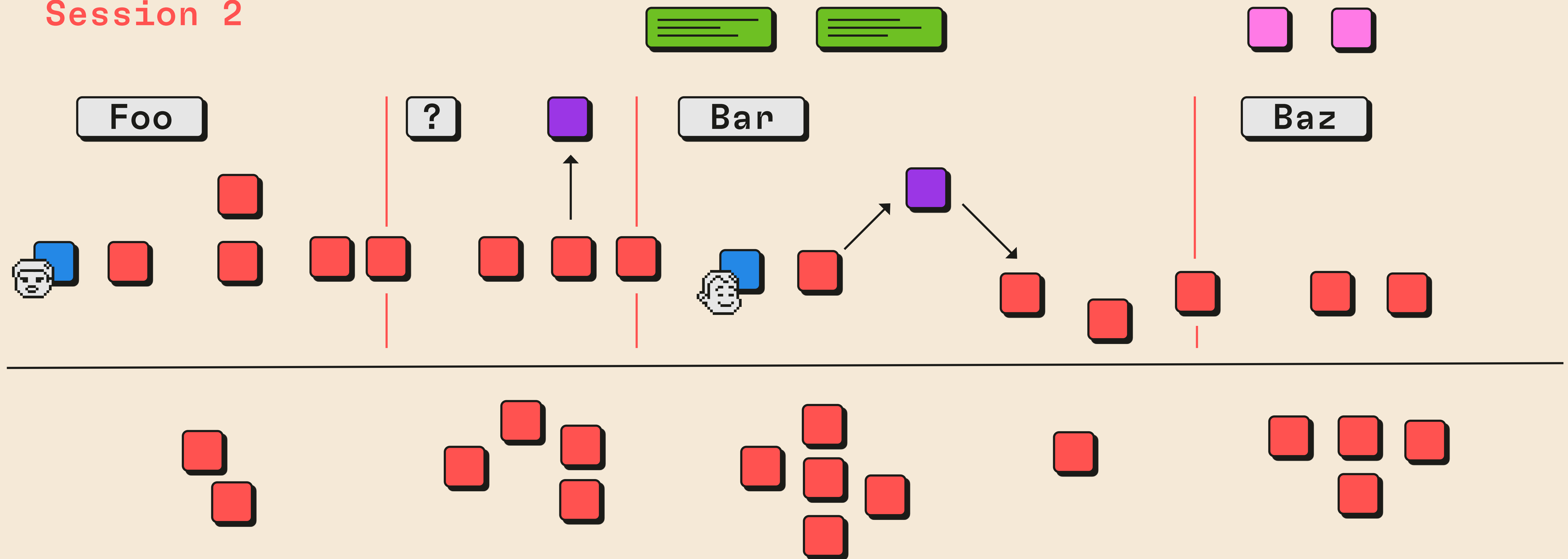
Pivotal Events

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 2



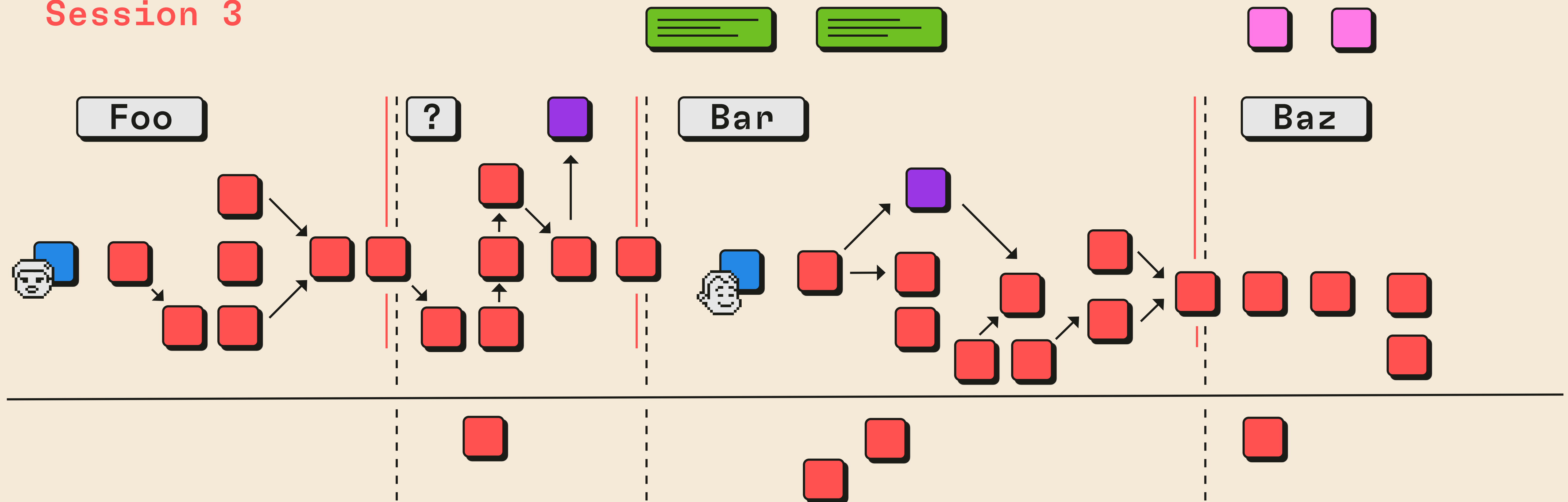
Kandidaten für Bounded Contexts

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 3



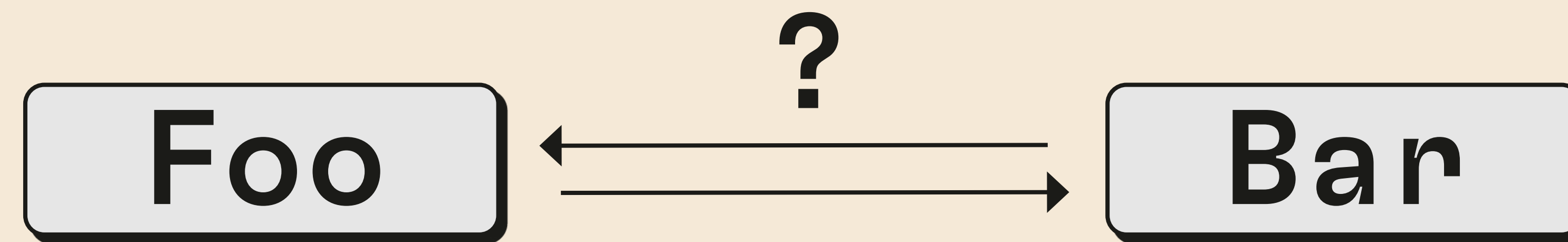
Bounded Contexts schärfen

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 3



Kommunikationsbeziehungen betrachten

Big Picture Event Storming



2.2 Durchführung Workshops

Session 3

Name		✕
Purpose	Classification	
Inbound Communication Ubiquitous Language Business Decisions Outbound Communication		
...		

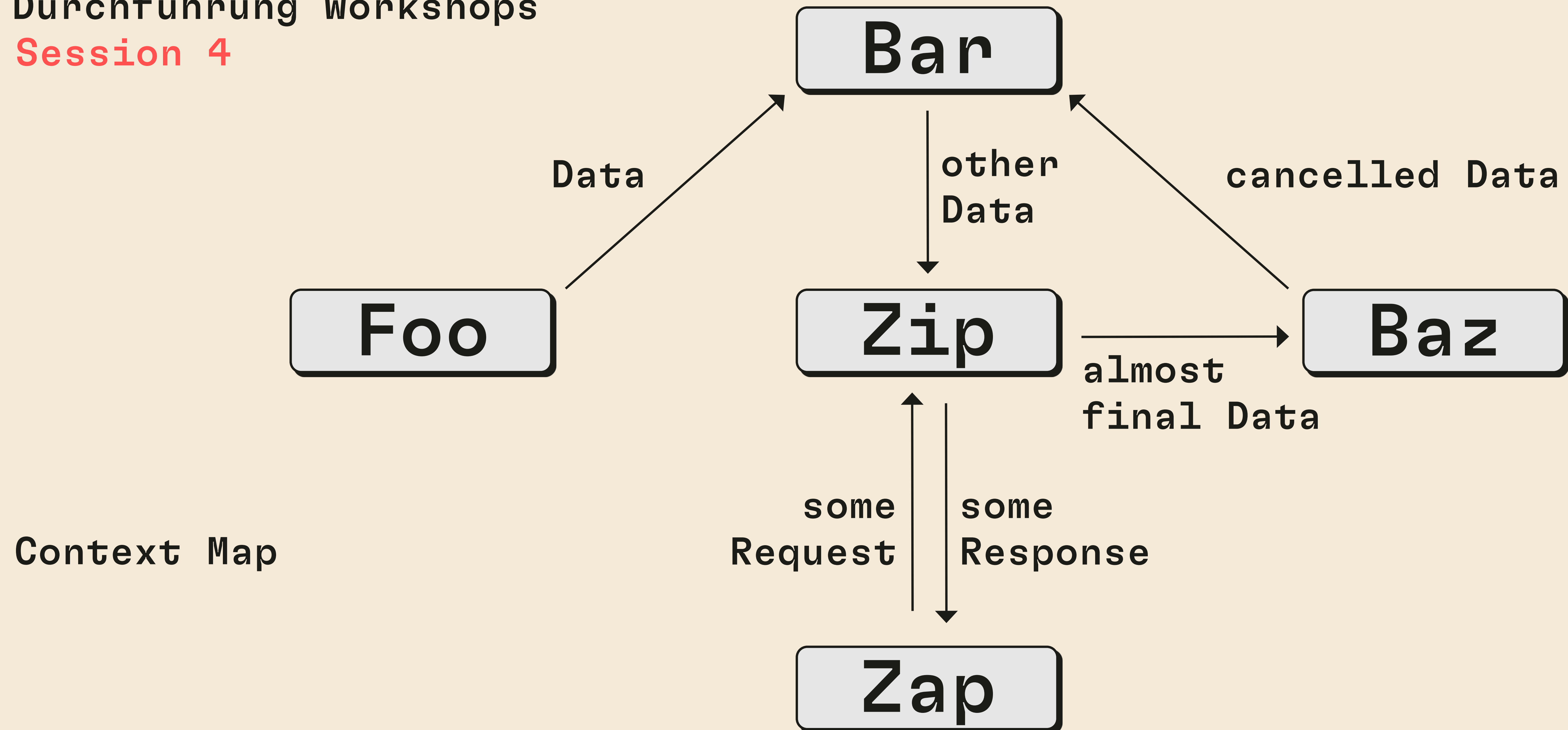
Context
Canvases

Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 4



Big Picture Event Storming

synyx

2.2 Durchführung Workshops

Session 5

	×
* Bei Bedarf: Restarbeiten vom Big Picture Event Storming * Umsetzungs-Strategie erarbeiten	

3.

Umsetzung

3.1 Strategie

×

- * Big-Bang-Neuentwicklung
 - Vermeintlich einfacher, weil Greenfield
 - Lange Entwicklung -> spätes Nutzer-Feedback -> hohes Risiko, unpassendes System zu entwickeln
 - Erzwungene Aufteilung in Legacy- und Next-Gen-Teams ist oft politisch schwierig
- * Iterative Ablösung
 - Frühzeitig und kontinuierlich in Produktion ausliefern minimiert Risiken
 - Anti-Corruption-Layer/Adapter-Ring um neue Services notwendig -> erhöhte Komplexität
 - Idealzustand: Ein Produkt, ein Backlog

Umsetzung

synyx

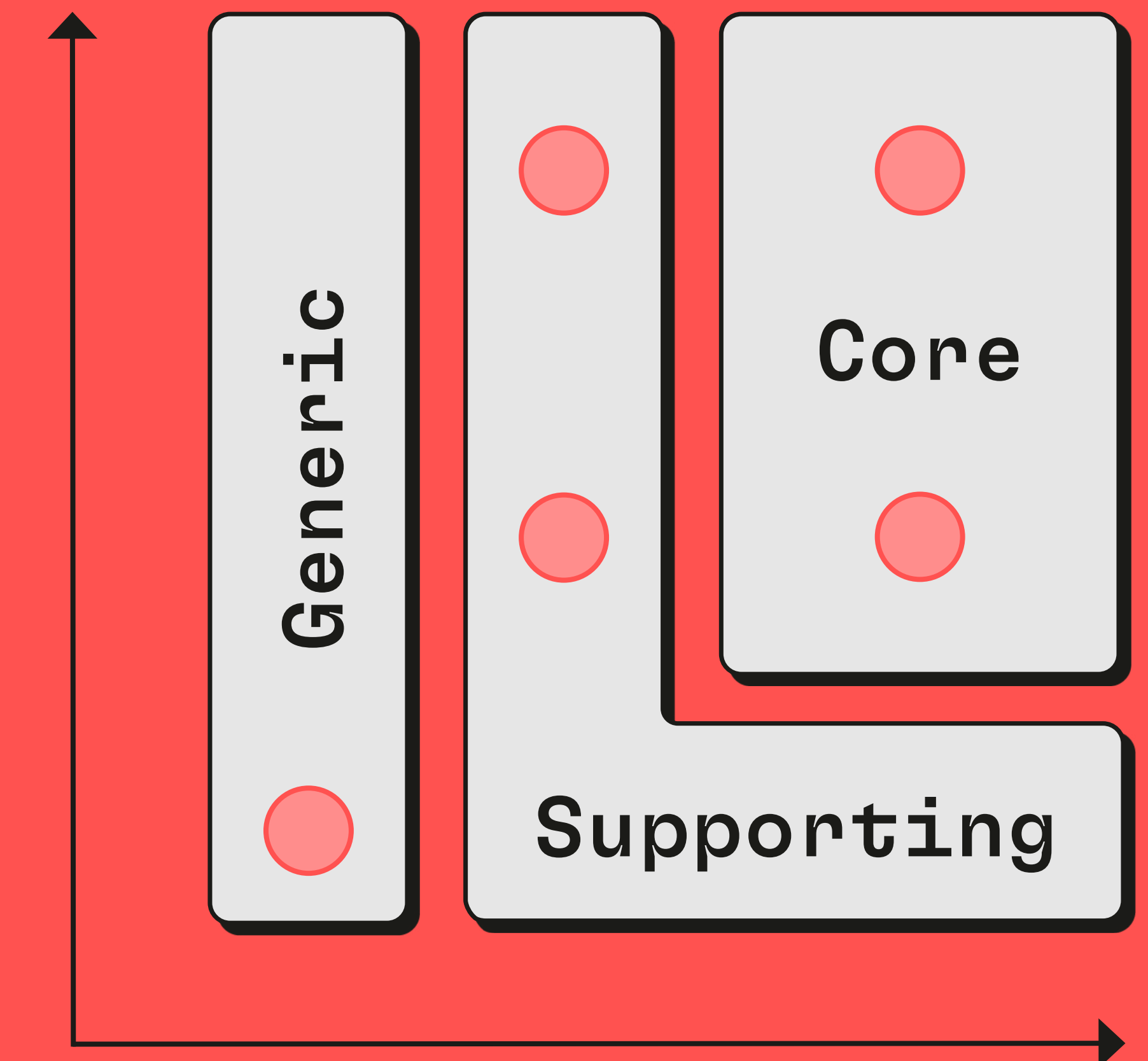
3.1 Strategie

×

Core Domain Chart

- * Bounded Contexts bewerten
- * Roadmap für die Umsetzung
 - Reihenfolge abhängig von Strategie:
 - MVP für auslieferbares Gesamtsystem
 - oder
 - Risiko-Minimierung bei schrittweiser Ablösung

Complexity



Business
Differentiation

3.1 Strategie



Leitplanken

* Grundlegende Architekturmuster

- Kommunikationsmuster
 - Event notification vs. Event carried state transfer vs. Event sourcing
 - REST vs. RPC
- Modularisierung
 - Self Contained Systems vs. Modulith
 - Frontend-Monolith vs. Microfrontends

3.1 Strategie

×

Leitplanken

- * Technologie-Empfehlungen
 - Datenbank
 - Message-Broker
 - Laufzeitumgebung
 - Frameworks

3.1 Strategie

×

Proof of Concept

- * Kleiner Ausschnitt der Context Map (2-3 Bounded Contexts) als PoC umsetzen
- * Nicht zu komplex, aber auch nicht trivial!

3.1 Strategie



Team-Zusammenstellung / Skalierung

- * Empfehlung: 50% Entwickler aus Alt-System, 50% Experten für neuen Technik-Stack
- * Langsam skalieren, max. ein neues Team pro Quartal
- * Outsourcing (mindestens von Core Domain) vermeiden

3.2 Pitfalls



Zu wenige Fachexperten im Event-Storming

- * „Internen Projektleiter“ finden, der gut in der Organisation vernetzt ist
- * 1-2 Fachexperten können ausreichen, wenn sie als Multiplikator agieren

3.2 Pitfalls



„Modellieren wir jetzt das Alt-System oder etwas Neues?“

- * Wir modellieren zunächst (noch) gar kein System, sondern erschaffen ein gemeinsames Verständnis für unsere Geschäftsprozesse!
- * Möglichst von Technik lösen

3.2 Pitfalls



Outsourcing

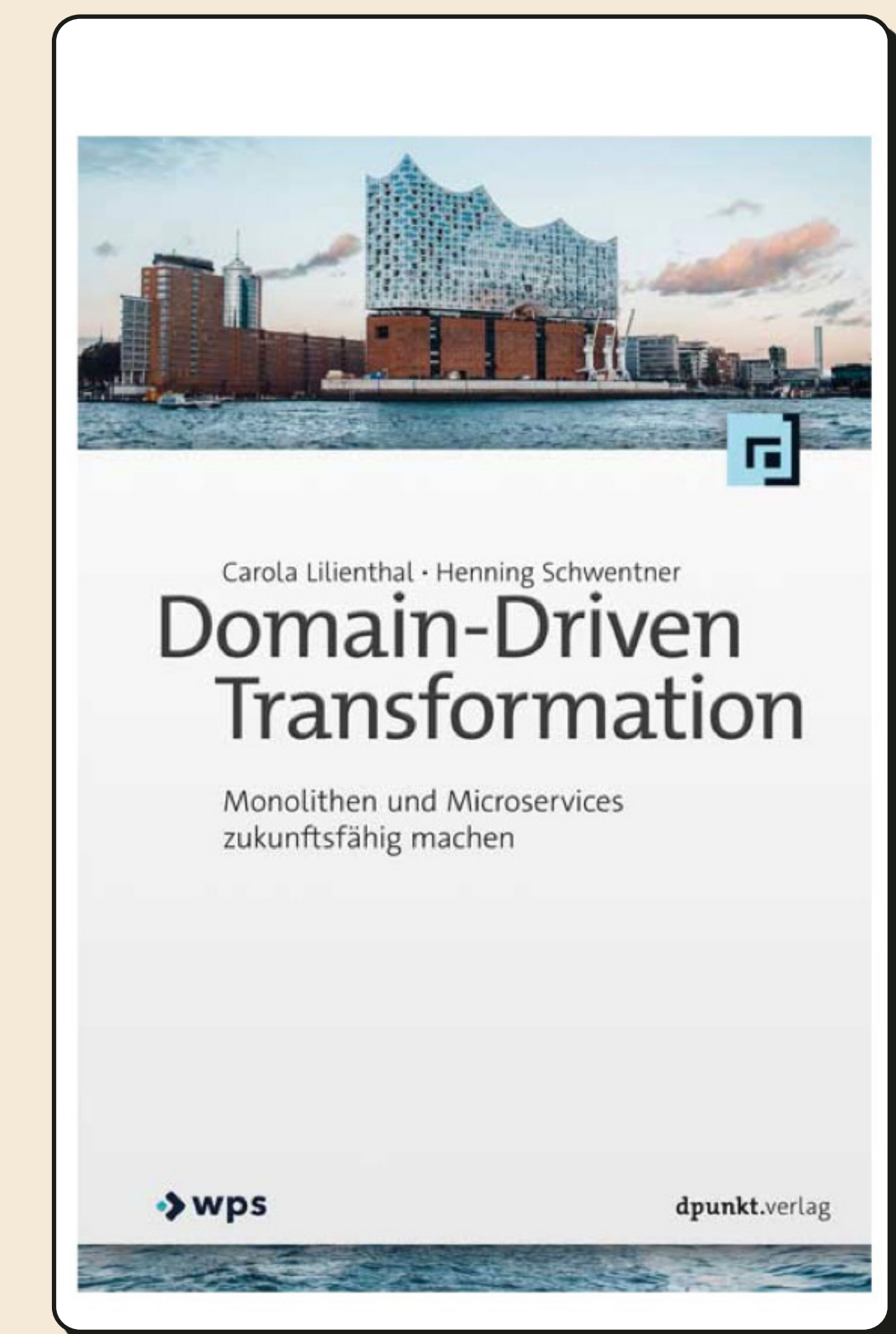
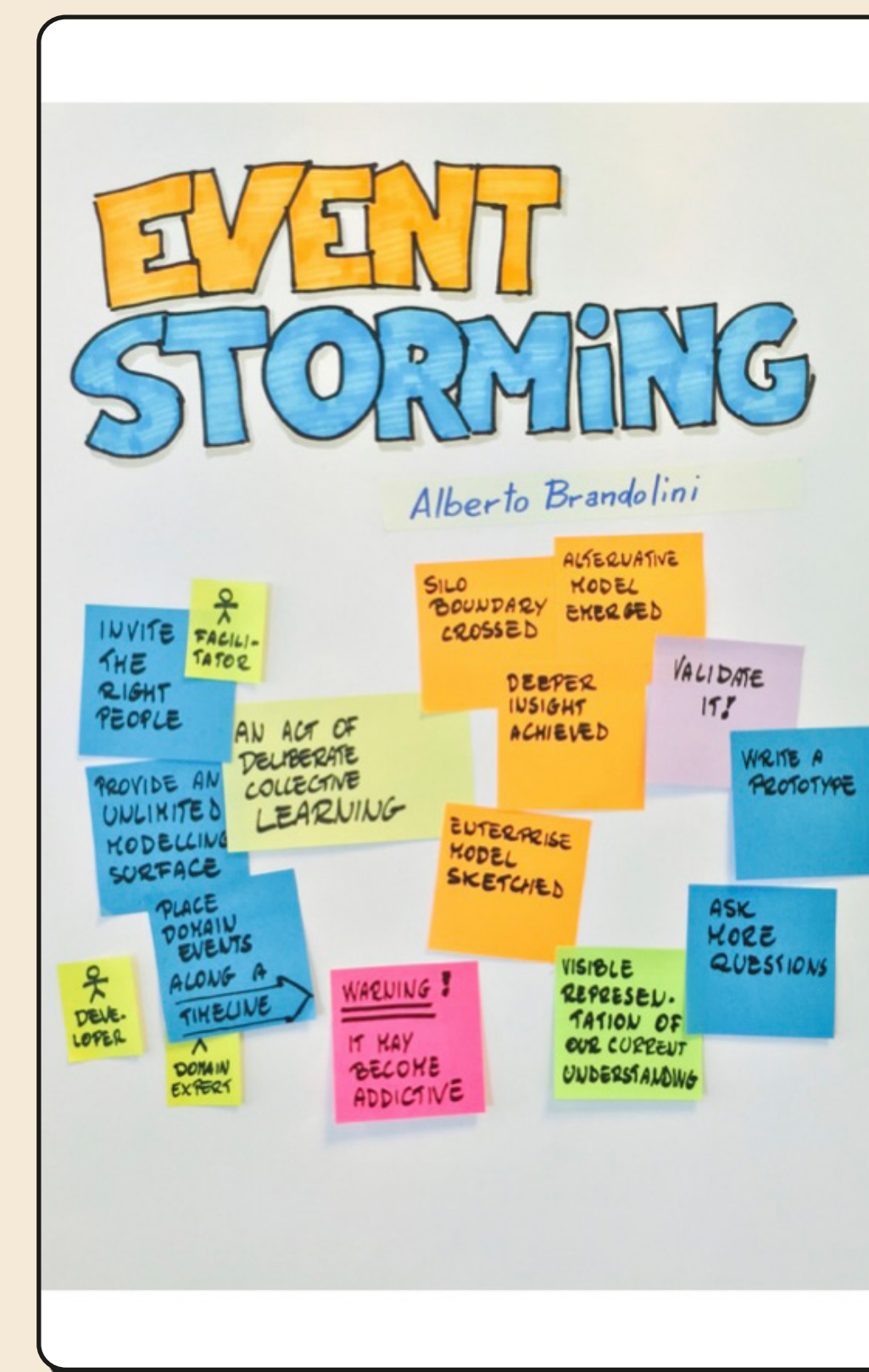
- * Saubere Bounded Context-Beschreibungen können Manager verleiten, diese als komplette Arbeitspakete extern zu vergeben
- * Standard-Software für Generic Domain ist völlig in Ordnung
- * Wenn externe Entwicklungs-Teams eingesetzt werden:
Zusammenarbeit an Supporting Domain erproben
- * Zur Not: Laufen lassen. Wahrscheinlich merkt das Management nach ein paar Monaten, dass die Idee nicht besonders gut war.

Fazit

synyx

×

- * Gemeinsam Geschäftsprozesse wiederentdecken ist eine faszinierende Erfahrung
- * Die Architektur von Software-Systemen muss gepflegt werden, sonst wird sie (wieder) verrotten



Vielen Dank!



<https://synyx.de>