ANWENDUNGSMODERNISIERUNG?

PRAGMATISCH, PRAKTISCH, ERFOLGREICH MIT ARCHITECTURE DECISION RECORDS!

JUG Saxony Day

25.09.2020

Stephan Pirnbaum @spirnbaum

BUSCHMAIS

BUSCHMAIS ist ein Dresdner IT-Beratungsunternehmen, gegründet im Jahre 2008. Unsere Schwerpunkte liegen in der Architekturberatung und der Entwicklung moderner Geschäftsanwendungen.

Gemeinsam mit unseren Kunden analysieren, planen und optimieren wir IT-gestützte Prozesse und unterstützen sie bei der Umsetzung neuer Anforderungen. Dabei arbeiten wir branchenunabhänig und technologiefokussiert in effizienten Teams.

BUSCHMAIS GbR

Leipziger Straße 93

01127 Dresden

Tel. +49 351 3209230

info@buschmais.com

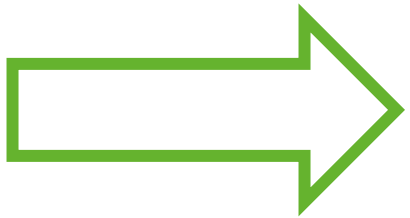
www.buschmais.de

ANWENDUNGSMODERNISIERUNG

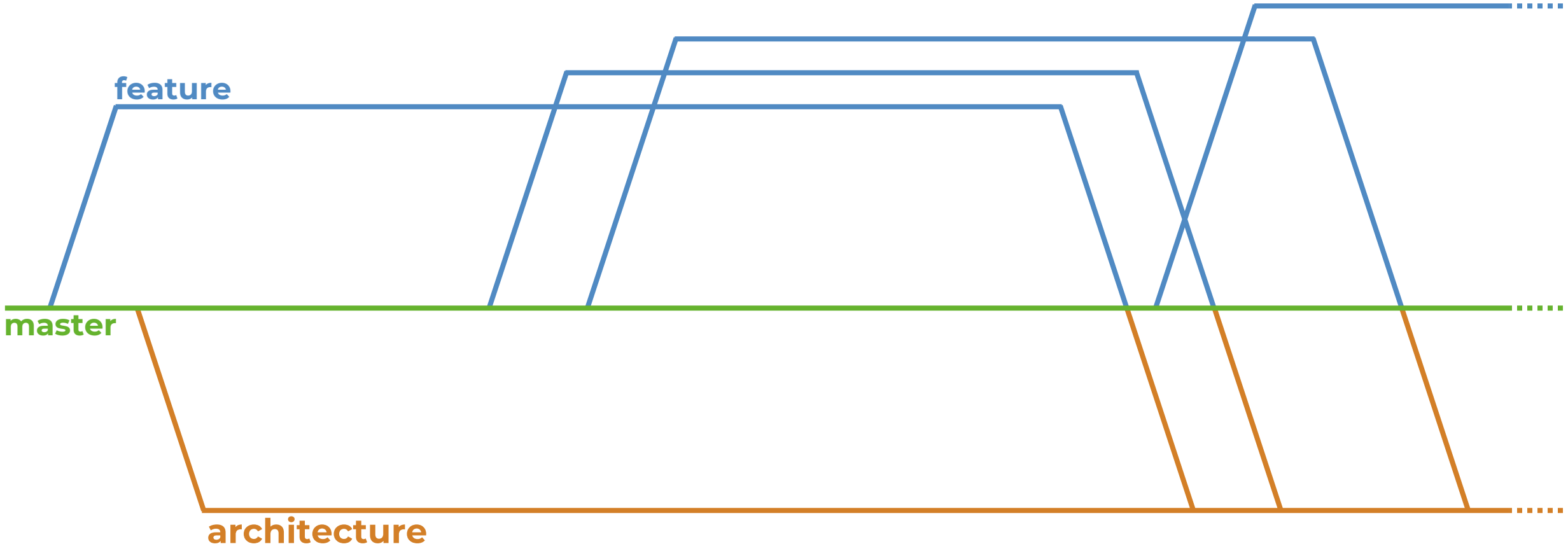
- Modernisierungsdruck durch ext. und int. Faktoren
  - Neue Anforderungen
  - Neue Technologien
  - Neue Konkurrenten
  - Kostenoptimierung
  - Prozessoptimierung

- Kontinuierliche Modernisierung der Architektur notwendig

ABER: Funktionale Weiterentwicklung kann nicht ausgesetzt werden

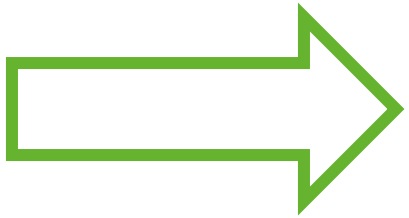


Parallele Modernisierung

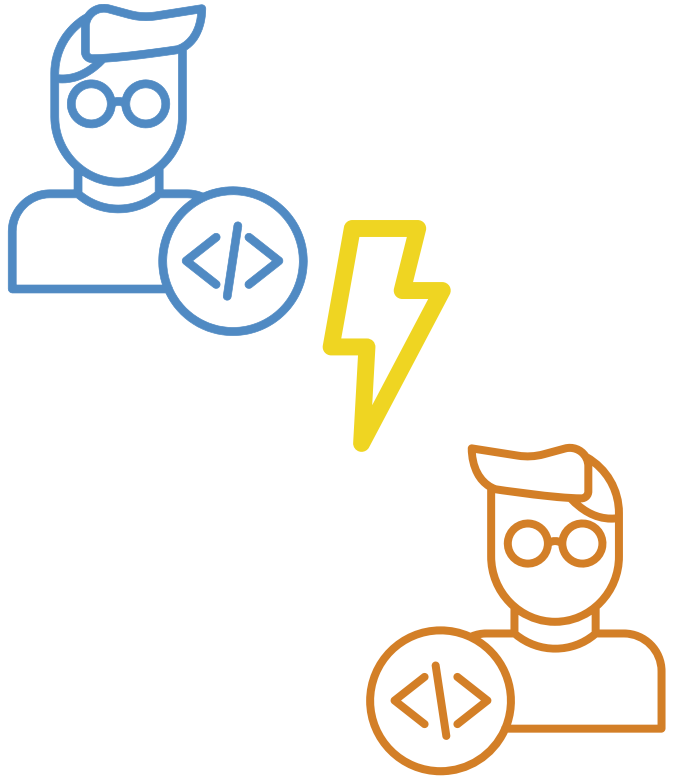


Ein typischer Projektverlauf

- Parallele Entwicklung an Features und Architektur
  - Entwickler arbeiten gegeneinander
  - Steigender Integrationsaufwand



Endlose Geschichte



Anwendungsmodernisierung

= Chance für (Re-)Dokumentation

→ AUCH: Notwendigkeit für klare Dokumentation und Kommunikation

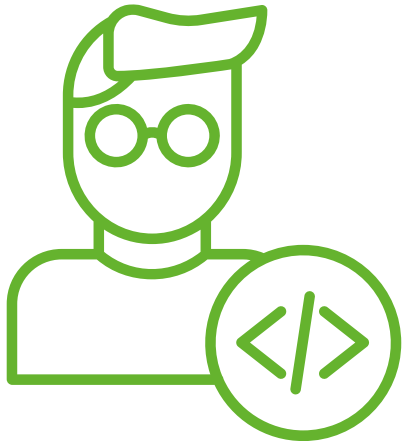
AD WAS?

Michael Nygard (2011)

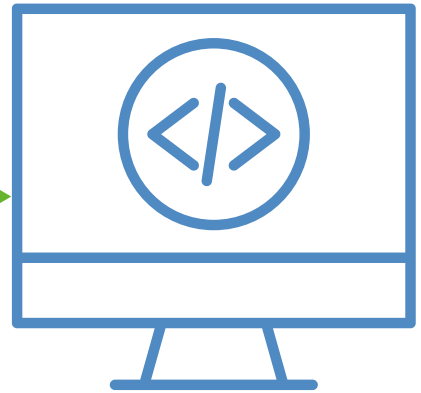
“Documents that assist the team itself can have value, but only if they are kept up to date.”

“Large documents are never kept up to date.”

“Nobody ever reads large documents, either.”



Warum?

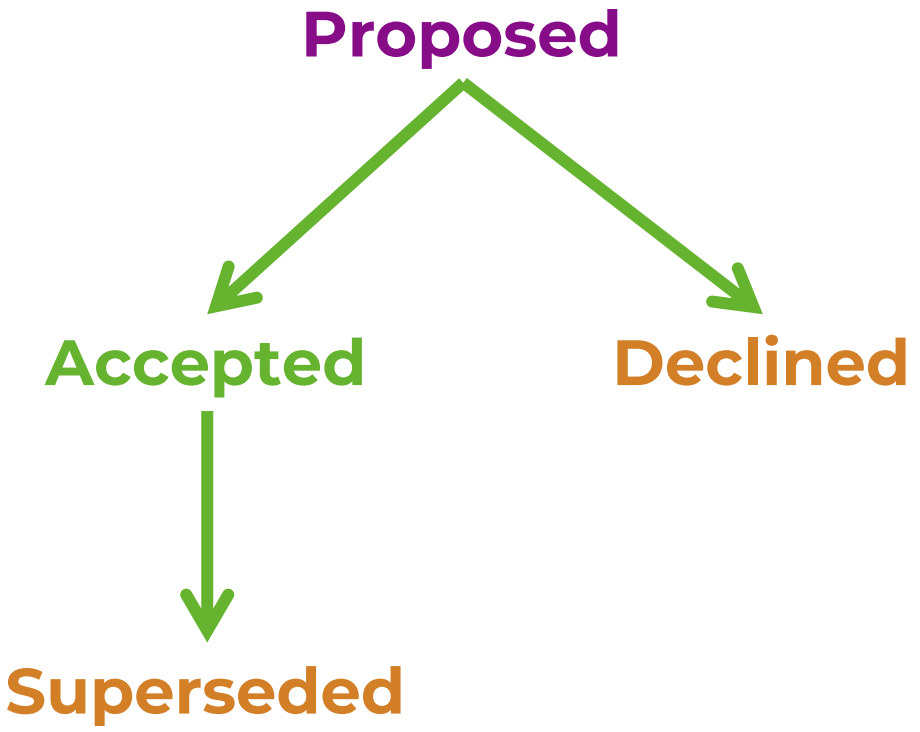


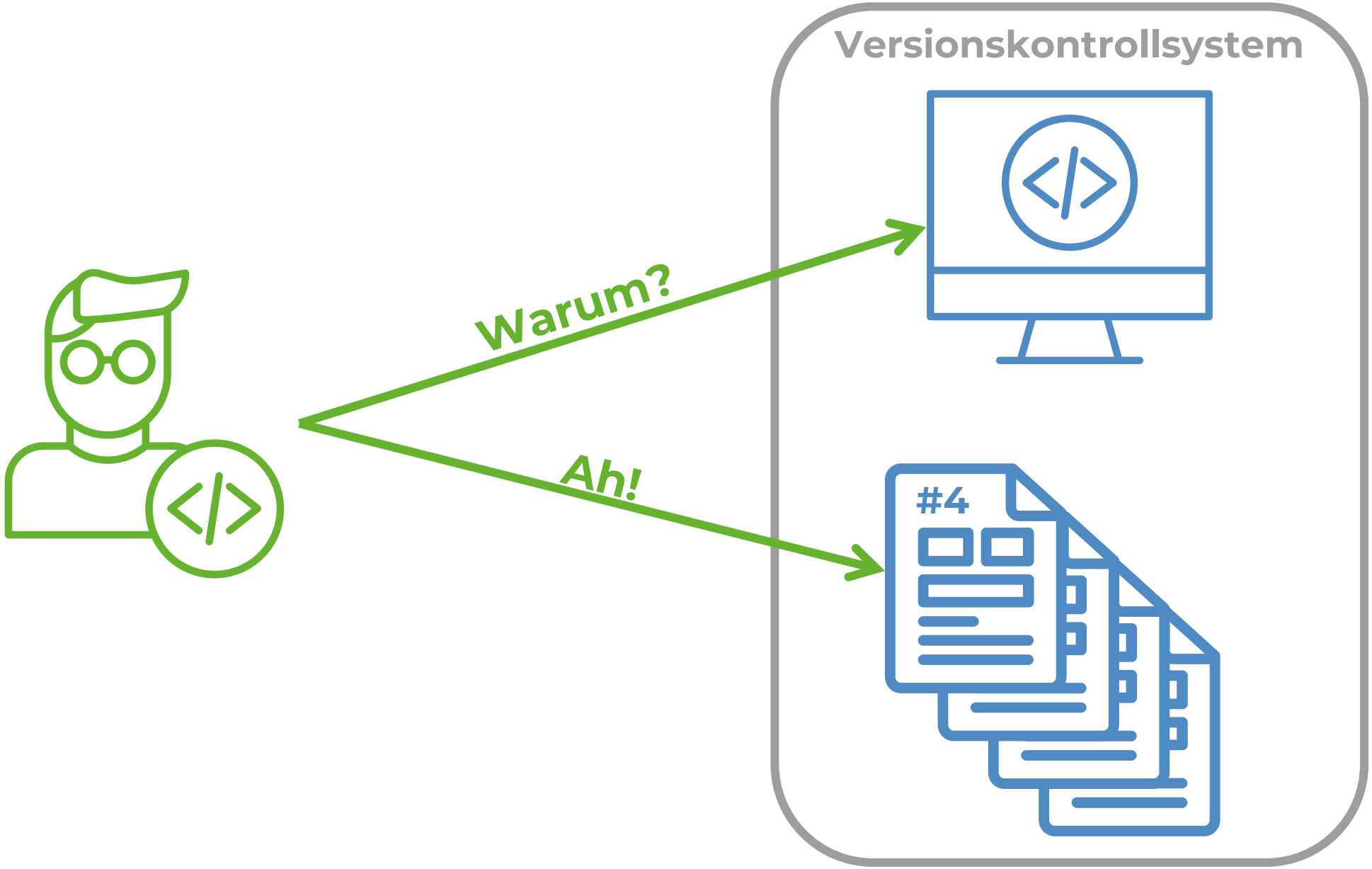
Blind akzeptieren oder ändern ist riskant

- Dokumentation muss:
  - Entscheidungen explizit festhalten
  - Kontext der Entscheidung darlegen
  - Auswirkungen auf Entwicklung aufzeigen
  - Leichtgewichtig sein

- Architecture Decision Records
  - Ein Dokument je Entscheidung
    - ✓ Klein
    - ✓ Übersichtlich
    - ✓ Fokussiert

- Architecture Decision Records
  - Version mit Status
    - ✓ Nachvollziehbar
    - ✓ Relevant
    - ✓ Aktuell





ADRS ♡ MODERNISIERUNG

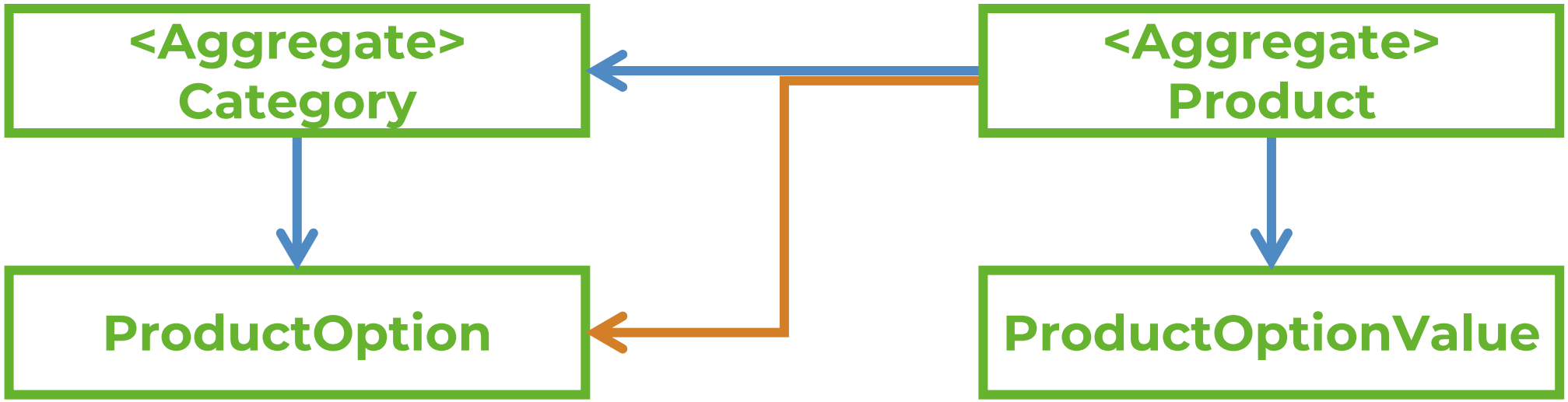
ADRS ♡ MODERNISIERUNG

Ein Beispiel

- Online-Shop
  - Produktkatalog
  - Warenkorb
  - Nutzerkonten
  - Bestellabwicklung
  - ...

- Bevorstehende Änderung der DB für den Produktkatalog
  - (Aufgrund Performanz, Pflege, …)
  - Relationale DB → Dokumentenbasierte DB
    - Evaluation und Proof of Concept ist bereits erfolgreich umgesetzt
    - Shop soll entsprechend vorbereitet werden

- Dokumentenbasierte DB erfordert Aggregate im Modell
  - Deep Linking zwischen Aggregaten ist verboten



- Herausforderungen
  - Weiterentwicklung vs. Umsetzung der Architekturentscheidung
  - Planbarkeit der Umsetzung → Aufwand?
  - Monitoring der Fortschritte

ADRS ♡ MODERNISIERUNG

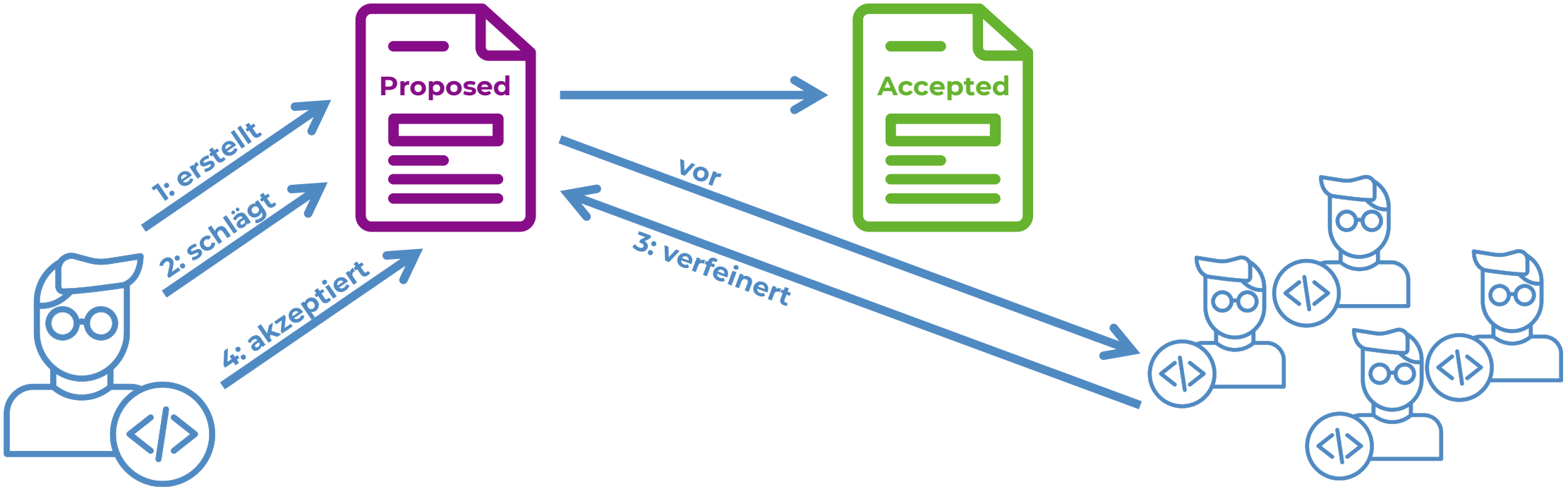
Pragmatisch!

- Dokumentation mit AsciiDoc
  - Leichtgewichtig
  - Versionierbar
  - Verschiedene Ausgabeformate

```
= 001 – ...

== Status Proposed

== Kontext
* Migration der DB des
  Produktkatalogs

== Entscheidung
* Aggregatbildung
** Keine Deep Links

== Konsequenzen
* Alle Entitäten nur durch 1
Aggregat referenziert
** Existierende Deep Links
auflösen
```

Nicht alle im Team müssen an der Entscheidung beteiligt sein

- Prozess möglichst schmal halten
  - Wichtig: Jeden Beteiligten informieren

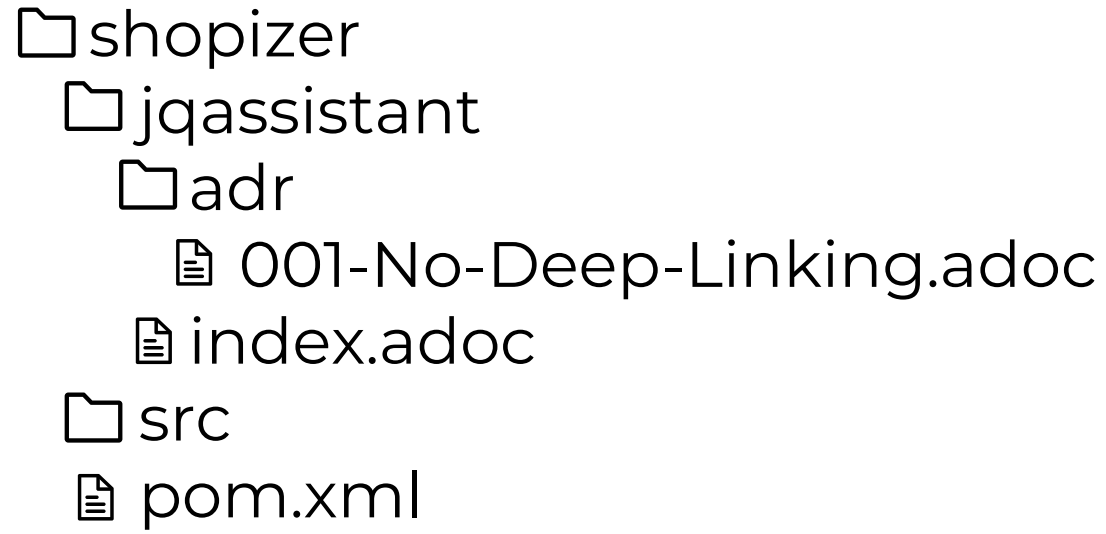


- Vereinfachung der Arbeit mit ADRs durch
  - Bereitgestelltes ADR-Template
  - Thematische Unterordner (Struktur, Persistenz, Logging, ...)
  - Klarer Prozess von der Idee zum finalen ADR
  - Gute IDE-Integration (AsciiDoc-Plugin)

ADRS ♡ MODERNISIERUNG

Praktisch!

- Ablage im Repository
  - Code-nah
  - Versioniert
    - Nachvollziehbar
  - Integration mit z.B.
    - Maven (Rendering-Plugins)
    - jQAssistant ☺

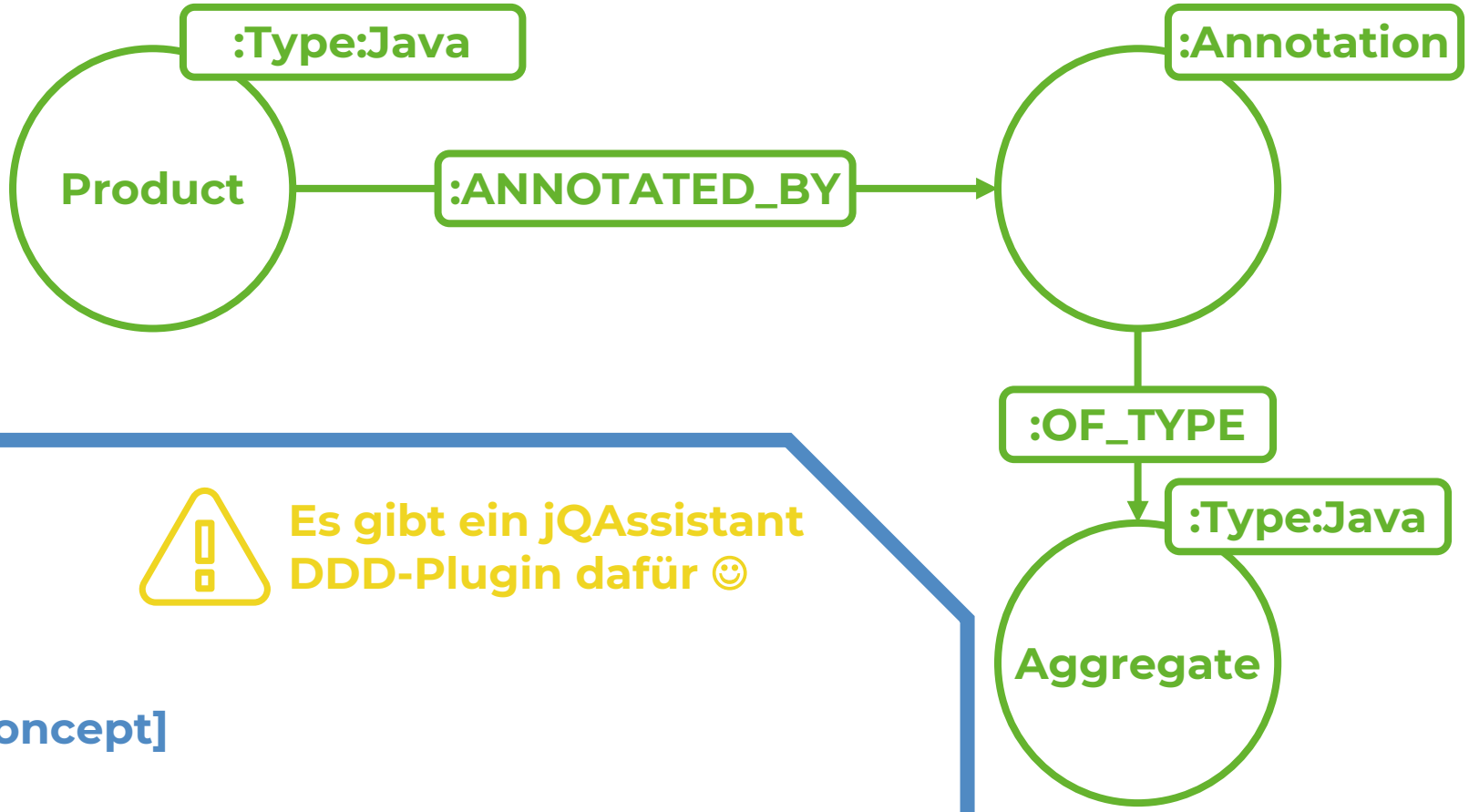


- ADR ist nah am Code, ABER:
  - Wer prüft die Aktualität und Relevanz der Regeln?
  - Wer weist den Entwickler auf Regelverletzungen hin?
  - Wer überprüft Fortschritte und zeigt verbleibende Aufwände?


jQAssistant
Your Software. Your Structures. Your Rules.
+
=
scan
document
validate

- Integration mit jQAssistant ermöglicht
  - Identifikation von Architekturkonzepten im Code
    - z. B. Aggregate
  - Überprüfung von definierten Architekturregeln
    - z. B. kein Deep Linking zwischen Aggregaten

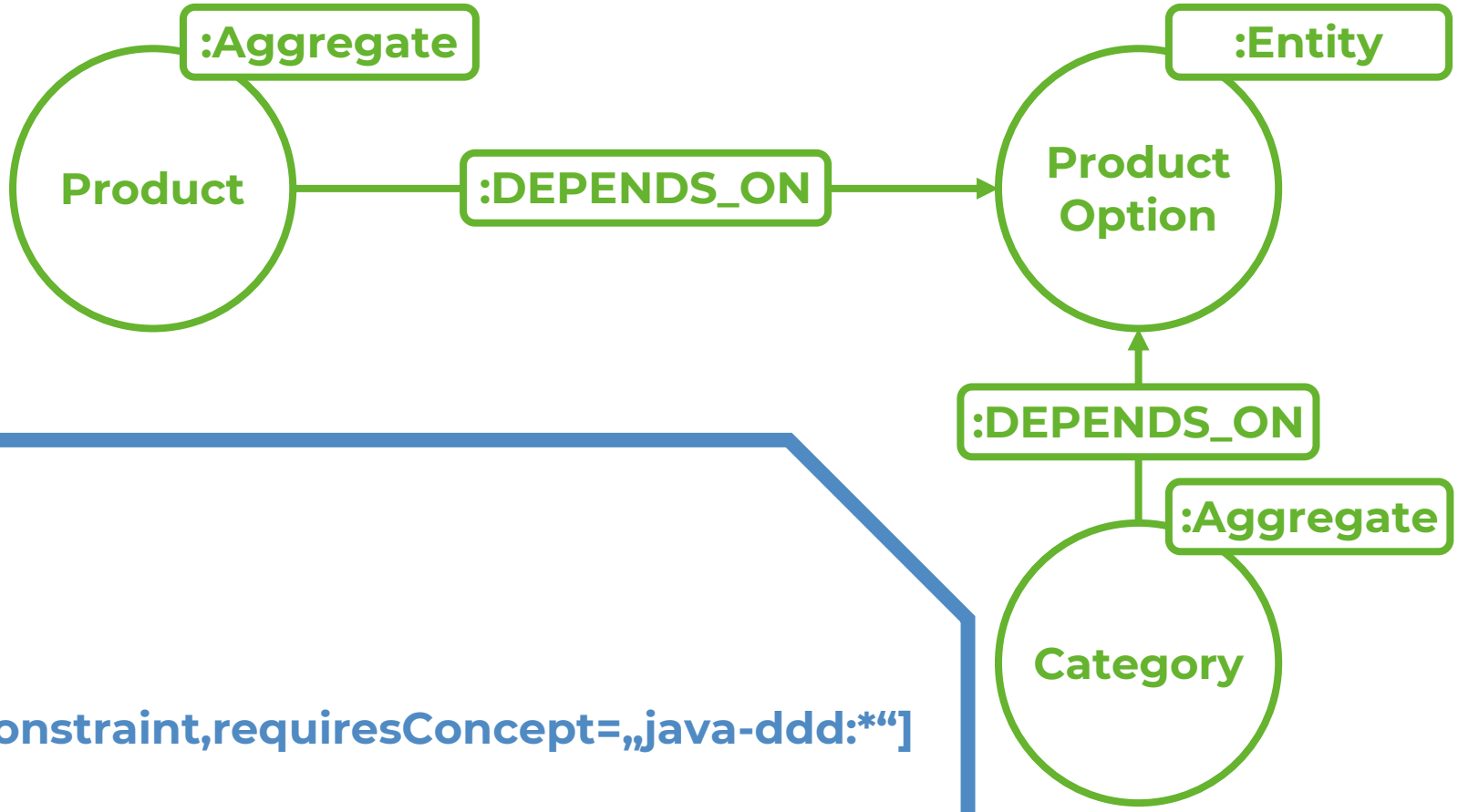
∧ Konzepte




```
= 001 – ...
...
== Konsequenzen
[[adr:Aggregates]]
[source,cypher,role=concept]
----
MATCH (t:Type)-[:ANNOTATED_BY]->()-[:OF_TYPE]->(a:Type)
WHERE a.name = „Aggregate“
SET t:Aggregate
RETURN t.fqn
----
```

Es gibt ein jQAssistant DDD-Plugin dafür ☺

Regeln




```
= 001 – ...
...
== Konsequenzen
[[adr:Aggregates]]
[source,cypher,role=constraint,requiresConcept=„java-ddd:*“]
----
MATCH (a:Aggregate)-[:DEPENDS_ON]->(e:Entity)
WITH e, collect(a.fqn) AS referencingAggregates
WHERE size(referencingAggregates) > 1
RETURN e.fqn AS Entity, referencingAggregates
----
```

ADRS ♡ MODERNISIERUNG

Erfolgreich!

- Leichte Nachvollziehbarkeit der Entscheidungen
  - Besonders für neue Entwickler
- Leichtgewichtige Dokumentation einfach verteilbar
- Manifestation von Wissen

- Definition von Architekturkonzepten und -regeln
  - Kontinuierliches Monitoring der Fortschritte
  - Genaue Planung der Umsetzung (Abschätzung)
  - Automatisierte Absicherung der Umsetzung der Entscheidungen

Beispiele

https://github.com/buschmais/adr-starter/tree/master/jqassistant/adr

DANKE !

NOCH FRAGEN?

WAR'S DAS?


BUSCHMAIS
UNSERE VORTRÄGE
UND WORKSHOPS
Online-Vortrag "Softwaremodernisierung als Katalysator für die Digitale Transformation"
25. November 2020
Workshop "Lasst uns einen Monolithen (z)erlegen!"
20. - 21. Januar 2021 in Dresden
Jetzt anmelden unter buschmais.eventbrite.com!
www.buschmais.de

BUSCHMAIS

KONTAKT

Stephan Pirnbaum

stephan.pirnbaum@buschmais.com

Tel. +49 351 320923-22

Twitter: @spirnbaum

BUSCHMAIS GbR

Leipziger Straße 93

01127 Dresden

Tel. +49 351 3209230

info@buschmais.com

www.buschmais.de