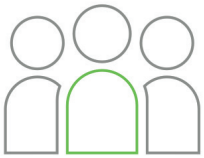
# Projektstart-Workshop mit Domain-Driven Design

## Software-Design anhand der geplanten Nutzung



## Die Situation

Sie planen ein neues Produkt, etwa eine neue Dienstleistung. Oder Sie möchten ein bestehendes Produkt mit einer neuen Software verbessern. Also geht Ihr Entwicklerteam an den Start. Nur: Entwickler denken meistens in Technologien und Daten. Für sie hängt die Geschäftslogik oft von Daten, Tabellen und Datenbanken ab. Früh festgelegte Architekturentscheidungen lassen sich jedoch teilweise nur schwer anpassen. Weder an Erweiterungen, noch, wenn sich zeigt, dass der Fachbereich andere Intentionen hat, als die Entwicklung verstanden hatte.

## Warum Domain-Driven Design?

Domain-Driven Design dreht die Perspektive um: Vereinfacht gesagt gliedern wir bei DDD die Geschäftslogik in fachliche Einheiten und entwerfen die Software so, dass sie dazu passt. Der Datenzugriff hängt ab von der Fachlichkeit, nicht umgekehrt. Anders gesagt: DDD ist das konsequente Design der Software anhand von Nutzungsfällen. Damit lässt sich DDD beispielsweise besonders gut mit Microservices kombinieren: DDD illustriert, wie sich Einheiten sinnvoll „schneiden“, also abgrenzen lassen. Dieses „Schneiden“ ist eine wichtige Grundlage für Microservices.

### Der Workshop

Direkt auf Ihr konkretes Projekt bezogen …

- bringen wir Fachbereich und Entwicklung zusammen
- vermitteln wir kurz & kompakt den Ansatz des Domain-Driven Design und seine Werkzeuge
- beginnen wir damit, gemeinsam die Geschäftsprozesse abzubilden
- untersuchen und clustern wir die einzelnen Ereignisse und die damit verbundenen Aktionen
- entwickeln dabei eine gemeinsame Sprache
- und erarbeiten ein grobes erstes Set-up

Das Ergebnis:

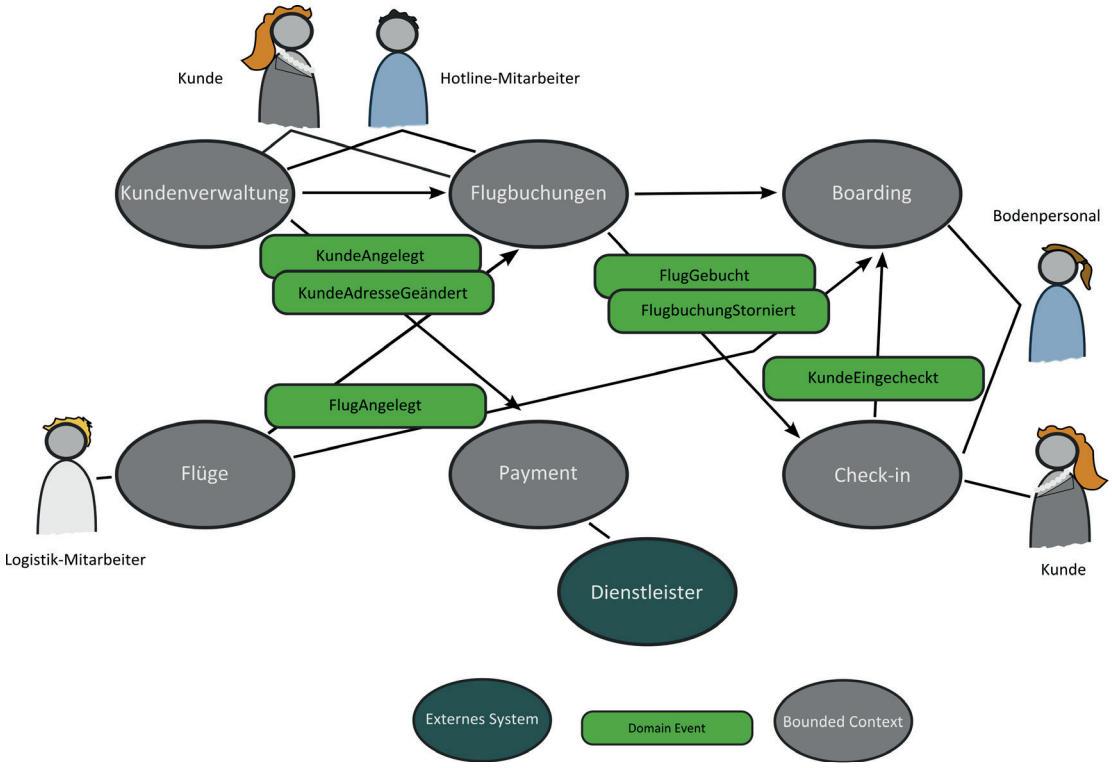
- ein gemeinsames Verständnis der Aufgaben und Prozesse von Fachbereich und Entwicklung
- Grundlagen, um selbständig weiter mit DDD zu arbeiten
- eine Basis, auf die das Entwicklerteam aufbauen kann – direkt auf das Projekt bezogen und nicht auf ein künstliches Beispiel
- die Möglichkeit, DDD selbst auszuprobieren, mit der Unterstützung engagierter Coaches
- ein erster Schritt zur Zielarchitektur – zum Beispiel Microservices

Die Rahmendaten:

- für Gruppen bis maximal 15 Teilnehmende
- Zielgruppe: Ein gemischtes Team mit Teilnehmenden aus Fachbereich und Entwicklung

## Im Detail: Warum ergänzen sich Agile, DDD und Microservices so gut?

Weil DDD ein Set von Werkzeugen und Mustern zur Verfügung stellt, um zunächst einen Geschäftsprozess zu beschreiben und damit eine Software zu designen. Dieses Design ist das Ergebnis der Zusammenarbeit zwischen Entwicklung und Fachbereich. Dabei basiert das Design darauf, fachliche Einheiten, die „bounded contexts“, voneinander zu differenzieren. Damit liefert DDD einen möglichen „Schnitt“ für Microservices - kleine Softwareeinheiten, die in sich eine geschlossene, kleine („micro“) Anwendung darstellen. Außerdem geht DDD iterativ vor - der einmal modellierte Geschäftsprozess wird kontinuierlich geprüft und verbessert. Das entspicht den agilen Prinzipien. Dabei sorgt die kontinuierliche Lieferung kleiner Software-Inkremente dafür, dass der Fachbereich sofort testen kann, ob die Software die gewünschte Intention erfüllt. Und wenn das mal nicht der Fall sein sollte? Dann erleichtert es die via DDD entwickelte gemeinsame Sprache zwischen Entwicklern und Fachbereich, ein gemeinsames Verständnis der Anforderungen zu erlangen – Ausgangsbasis für ein verbessertes (Teil-)Produkt.




Die Abbildung zeigt ein vereinfachtes Beispiel für eine Darstellung der einzelnen Einheiten (hier: Bounded Context) und der verbundenen Events sowie der beteiligten Personen – eine wertvolle Grundlage für das Design der Software!

## Kontakt

andrena. Wir sind Experten für Agile Software Engineering.
Lassen Sie sich von uns unverbindlich beraten!

### Ihre Ansprechpartner finden Sie unter:

https://www.andrena.de/kontakt



andrena objects ag • Albert-Nestler-Straße 9 • 76131 Karlsruhe • Tel. +49 721 6105-122 • info@andrena.de • www.andrena.de