Whitepaper

Radikaler Wertschöpfungsfokus mit fluiden Teams

Meist arbeiten mehrere Teams gemeinsam an einem Produkt. Durch die notwendige Abstimmung und Koordination fühlt sich die Entwicklung oft nach „Behörde für Agilität“ an.

Das geht besser. Dieses Whitepaper zeigt innovative Ansätze zu fluiden Teamstrukturen, die deutlich mehr Drive entwickeln können.

Wertschöpfung: eingeschränkt einsatzbereit

Agile Teams sind nicht so stark an Wertschöpfung ausgerichtet, wie wir gerne glauben.

Betrachten wir das fiktive Beispiel einer Dating-Plattform für Katzen. Hört sich lustig an, ist aber letztlich nur eine Webanwendung, über die Katzenzüchter passende Paarungspartner für ihre Rassekatze finden können. Für solche eBusiness-Anwendungen werden meist Teams entlang der Customer Journey gebildet (Team Topologies, Domain Driven Design etc. führen zu vergleichbaren Teamsstrukturen).

Jedes Team verantwortet einen Bereich auf dem Bildschirm und ist damit Ende-Zu-Ende-verantwortlich für kundensichtbare Funktionalität:

- Customer: Anlage von Benutzerkonten, Login etc.
- Offer Management: Einstellen von Angeboten
- Cat: Eingabe, Anzeige und Verwaltung der Katzendaten
- Search: Suchkriterien und Ausführen der Suche
- Search Result: Anzeige der Suchergebnisse
- Monetization: Monetarisieren, z. B. über Werbeeinblendungen

Businessrelevante Features

Nach der initialen Entwicklung des Systems entsteht meist der Wunsch nach business-relevanten Features, die die Mitarbeit mehrerer Teams benötigen. Wollen wir beispielweise zusätzliche Einnahmequellen durch Prämium-Angebote erschließen, müssen wir Anpassungen bei Offer Management, Search, Search Result, Monetization und Customer vornehmen.

Wenn wir es regelmäßig mit solchen übergreifenden Features zu tun haben, sind die Teams nicht so radikal an der Wertschöpfung ausgerichtet, wie es möglich wäre.

Die oben gezeigte Struktur schafft kleine, stabile Teams mit einem festen Themenfokus. Jedes Team verantwortet kundensichtbare Funktionalität. Business-relevante Funktionalitäten können aber selten von einem Team allein umgesetzt werden. Der Wertschöpfungsfokus ist also eingeschränkt. Wir nennen solche Teams *statische Teams*, deren Charakteristik in Abbildung 1 dargestellt ist.

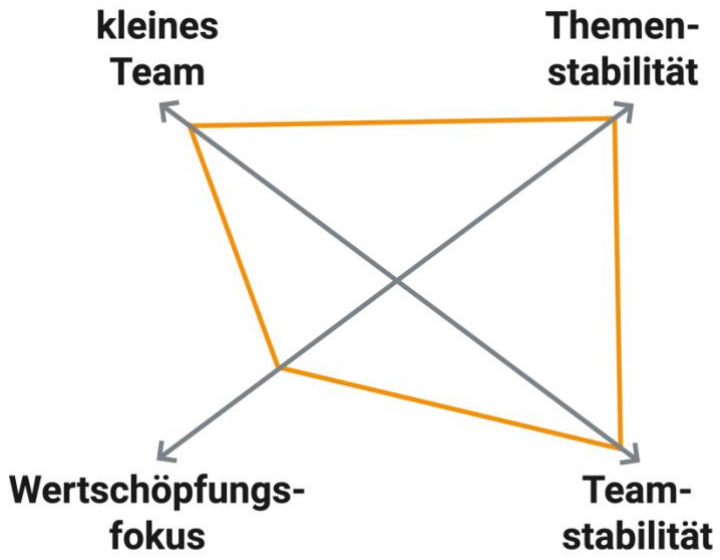


Abbildung 1: Statische Teams

Unsere Teams sind nicht so sehr an der Wertschöpfung ausgerichtet, wie wir gerne glauben möchten!

Ziele und High Performance

High Impact Teams können nur dann entstehen, wenn die Teammitglieder ein gemeinsames motivierendes Ziel verfolgen.

Wenn mehrere Teams beteiligt sind, ist es besonders herausfordernd, ein gemeinsames motivierendes Ziel zu schaffen. Hier stellt sich die wichtige Frage, ob die Teams **gemeinsam ein Ziel** verfolgen oder ob jedes Team **sein eigenes Ziel** hat. Beide Wege sind – auf ihre eigene Art und Weise – anspruchsvoll.

Ein gemeinsames Ziel für mehrere Teams

Arbeiten mehrere Teams auf ein gemeinsames Ziel hin, dann leidet oft das Gefühl der Selbstwirksamkeit. Der eigene Einfluss auf die Zielerreichung ist für ein einzelnes Team, geschweige denn ein einzelnes Teammitglied kaum noch erkennbar. Damit leidet die Fähigkeit zur Selbststeuerung: Wenn wir unseren Beitrag zum Ziel nicht einschätzen können, können wir auch nicht gezielt unser Verhalten so anpassen, dass wir das Ziel besser erreichen können.

Oft wird versucht, dieses Problem durch Fremdsteuerung zu lösen. Dann gibt man dem Team Arbeitsanweisungen (heute oft in Form detaillierter Listen von User Stories) und überprüft Fortschritt anhand der Abarbeitung. Das wird dann „agil“ genannt, ist aber nur eine entschärfte Variante von Command&Control. Das ist nicht per se verwerflich. Es erzeugt aber kein High Impact Team.

High Impact Teams mit teamübergreifenden Zielen sind möglich! Es ist aber anspruchsvoll, sie herzustellen. Man muss es schaffen, dass der Beitrag zum übergreifenden Ziel erfahrbar wird und die Fähigkeit zur Selbststeuerung in den Teams entwickelt wird. Dazu ist es essenziell, dass gemeinsame Ziele schnell erreicht werden. Wenn übergreifende Funktionalitäten sequenziell von Team zu Team geschoben werden, fühlt sich das häufig träge an und es entsteht kein echter Drive. Wenn hingegen die Teams alle gleichzeitig an der übergreifenden Funktionalität arbeiten, haben eine komplett andere Situation.

Damit diese Parallelisierung der Arbeit möglich wird, braucht es Vertrauen, enge Abstimmung und die passenden technische Skills im Team (Test Driven Development und der Umgang mit Mocks, Stubs und Dummies gehören zum Handwerkszeug).

Teams mit eigenen Zielen

Wenn die Teams eigene Ziele verfolgen, ist das Gefühl von Selbstwirksamkeit und folglich die Selbststeuerung viel leichter herstellbar. Man hat es dafür mit zwei anderen Herausforderungen zu tun:

1. Die Teamziele müssen so gestaltet werden, dass sie in Summe dazu führen, dass das übergeordnete Ziel erreicht wird.
2. Oft müssen die statischen Teamstrukturen aufgebrochen werden und wir brauchen fluidere Teamkonzepte.

Fluide Teams

Statische Teams haben unbestreitbare Vorteile. Durch ihre langfristige Stabilität können die Teams zusammenwachsen und durch ihren Themenfokus tiefes Fachwissen erwerben. Beides kann dazu führen, dass die Teams immer leistungsfähiger werden. Die Einschränkungen beim Wertschöpfungsfokus sind der Preis, den wir dafür bezahlen (zur Erinnerung nochmal die Visualisierung in Abbildung 2).

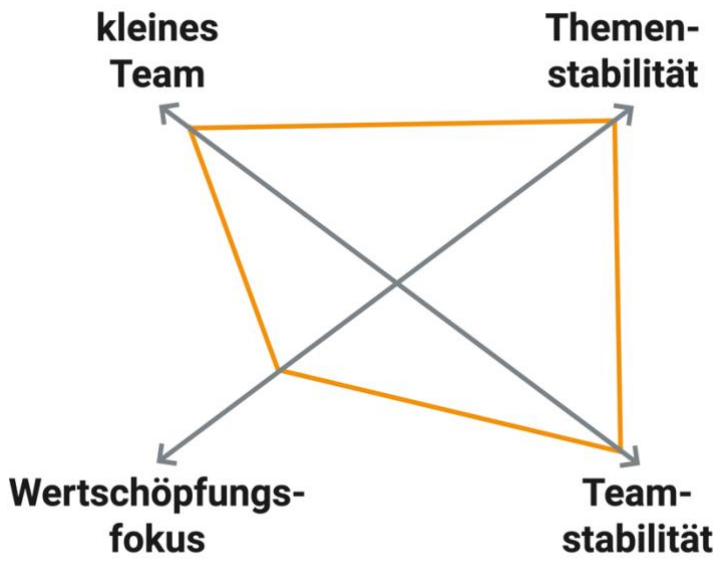


Abbildung 2: Statische Teams

Fluide Teamkonzepte erhöhen den Wertschöpfungsfokus und nehmen dafür Einschränkungen bei den anderen Dimensionen in Kauf: Mobile Teams, Mission Teams, Floating Teams. Das ideale Team existiert leider nicht.

Mobile Teams

Mobile Teams haben meist einen thematischen Schwerpunkt, ähnlich dem bei statischen Teams. Allerdings beschränken sie ihre Arbeit nicht auf diesen thematischen Schwerpunkt, sondern setzen business-relevante Funktionalitäten komplett um. Das bedeutet, dass sie Änderungen in Systemkomponenten anderer Teams vornehmen, wenn das für ihre Funktionalität notwendig ist. Sie sind mobil im ganzen Produkt unterwegs. Die langfristige Verantwortung für den Code und ggfs. Betrieb bleibt beim jeweiligen Schwerpunktteam.

Natürlich wird man die Funktionalitäten so auf Teams verteilen, dass die Teams möglichst in ihren Spezialisierungen arbeiten. Schließlich werden sie bei Änderungen in anderen Systemkomponenten nicht so produktiv sein, wie das jeweilige Spezialistenteam. Ähnlich wie auf der Ebene individueller Teammitglieder bilden sich dann für die Teams T-Shaped-Skill-Sets heraus (siehe Abbildung 3). Jedes Team hat eine thematische Spezialisierung, ist in anderen Themen aber auch handlungsfähig.

Abbildung 3: T-Shaped Skill Sets für Teams

So hat ein Team unserer Katzen-Dating-Plattform vielleicht seinen Schwerpunkt bei Search, ist aber auch in bei Offer Management und Search Result arbeitsfähig.

Mobile Teams vereinfachen die Planung und reduzieren Koordinationsaufwände zwischen Teams. Sie benötigen aber zwei Voraussetzungen:

1. Das Gesamtsystem braucht ein Mindestmaß an Einheitlichkeit. Wenn jede Systemkomponenten in einer anderen Programmiersprache geschrieben ist, dürfte das die meisten Teams überfordern.
2. Es braucht ein Mindestmaß an Vertrauen zwischen den Teams. Schließlich müssen die Teams ihre Systemkomponente verantworten, obwohl andere Teams in dem Code ändern dürfen. (Automatisierte Tests können als Basis einen erheblichen Beitrag leisten.)

Im Gegensatz zu statischen Teams schränken Mobile Teams die Themenstabilität ein: Die Teams arbeiten nicht nur an ihrem Thema. Dafür bekommen wir eine stärkere Ausrichtung der Teams an der Wertschöpfung (also den business-relevanten Funktionalitäten).

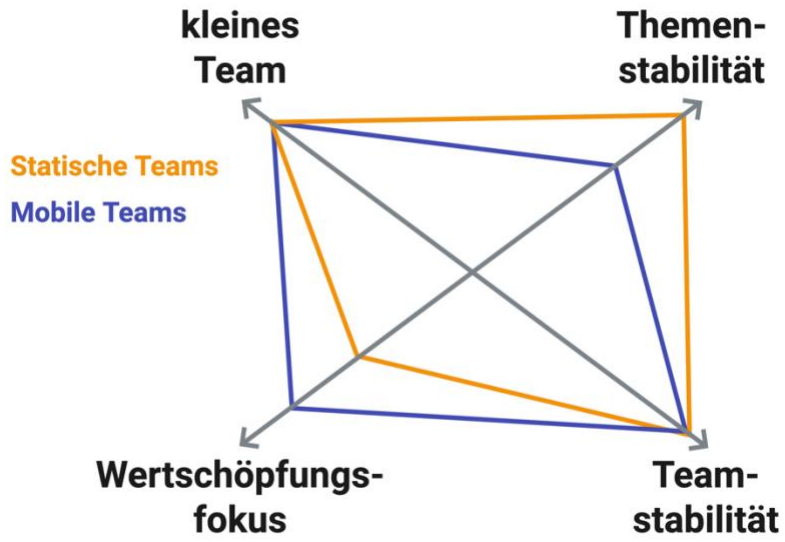
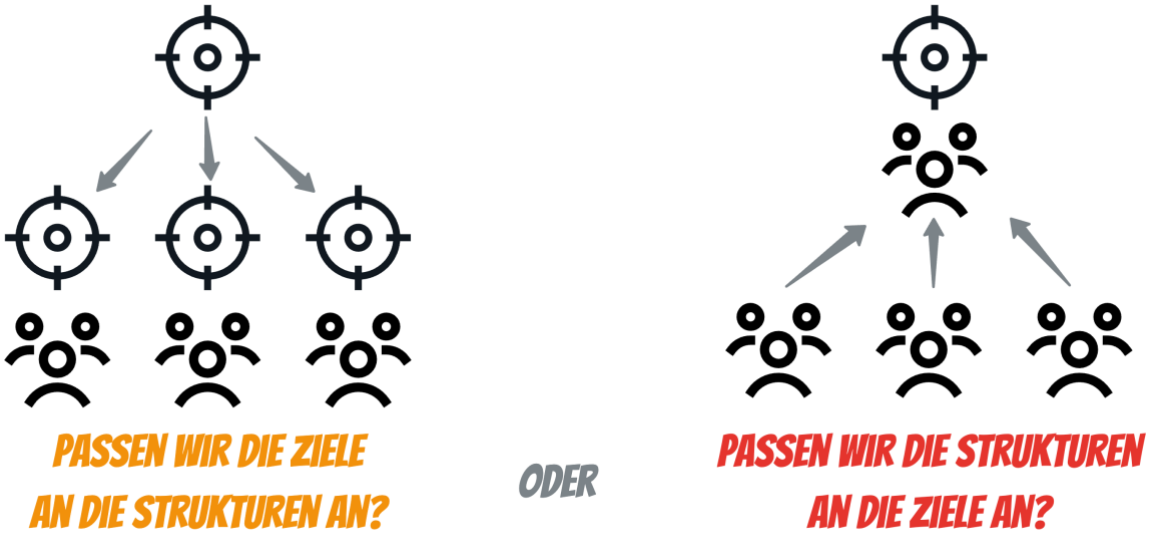


Abbildung 4: Mobile Teams

Mission Teams

Es gibt die deutliche Tendenz, Ziele und Aufgaben an die existierenden Strukturen anzupassen. Das führt oft zu vielen Abhängigkeiten. Wir fragen uns viel zu selten, ob wir die Strukturen (z. B. die Teams) an die Ziele anpassen sollten.



Wenn wir die Teams an die Ziele anpassen, entstehen Mission Teams. **Mission Teams** verfolgen eine Mission und lösen sich wieder auf, wenn sie ihre Mission erreicht haben. Missionen von 3-4 Monaten, in Ausnahmefällen auch länger, scheinen eine gute Dauer zu sein.

Die Mission Teams werden aus statischen Teams rekrutiert und nach Abschluss der Mission gehen die Mitglieder wieder zurück in ihre Stammteams.

Mission Teams existieren also deutlich kürzer als statische oder mobile Teams. Folglich sind sie auch kürzer mit ihrem Thema verbunden. Team- und Themenstabilität sind also geringer als bei statischen und mobilen Teams. Dafür ist der Wertschöpfungsfokus maximal (siehe Abbildung 5).

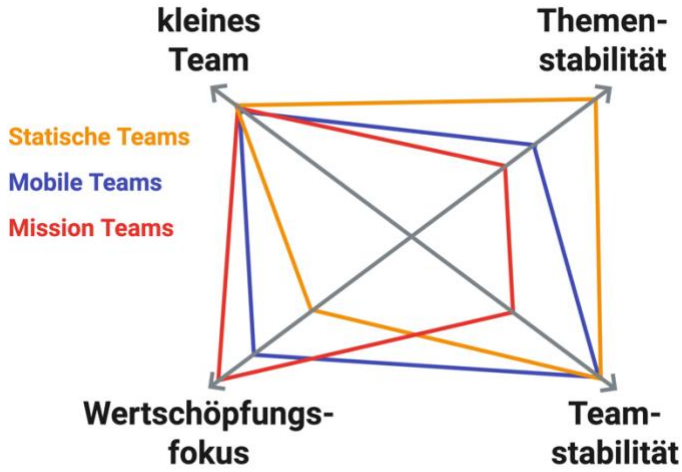


Abbildung 5: Mission Teams

Mission Teams brauchen weniger Einheitlichkeit der Systemkomponenten als Mobile Teams. Durch die Besetzung des Mission Teams lässt sich sicherstellen, dass die notwendigen Fähigkeiten zur Änderung der betroffenen Systemkomponenten im Mission Team vorhanden sind.

Damit Mission Teams gut funktionieren, sollten die Mitglieder exklusiv in ihrer Mission arbeiten und nicht parallel noch in anderen Teams oder an anderen Themen.

Floating Teams

Floating Teams sind in gewisser Weise Mission Teams auf Speed.

Wir bilden ein (zu großes) Produktteam – oft 20-50 Personen -, das ein Produkt oder eigenständigen Produktteil verantwortet. In diesem großen Team bilden sich je User Story temporäre Story Teams, die meist 3-4 Personen groß sind und nur wenige Tage zusammenbleiben.

Das Produktteam hat Wertschöpfungsfokus (es verantwortet ja das ganze Produkt), Themenstabilität (das Produkt) und ist langfristig stabil. Dafür ist es ganz und gar nicht klein. Die Story Teams sind sehr klein, haben einen hohen Wertschöpfungsfokus (keine Abhängigkeiten bei der Umsetzung von User Stories), aber nur sehr geringe Team- und Themenstabilität (siehe Abbildung 6).

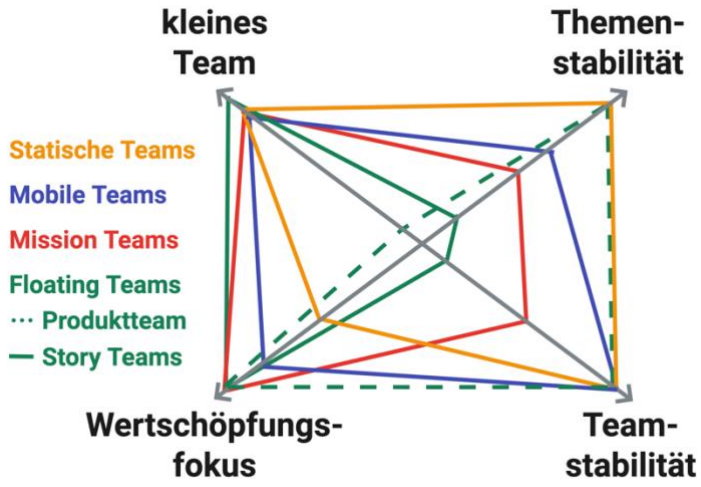


Abbildung 6: Floating Teams

Eine besondere Herausforderung besteht darin, das große Produktteam so zu gestalten, dass die Story Teams sofort produktiv arbeitsfähig sind. Dazu ist ein großes Maß an Vertrauen und psychologischer Sicherheit innerhalb des Produktteams notwendig. Und das wird mit steigender Teamgröße immer schwieriger. Vermutlich sind Floating Teams mit deutlich mehr als 50 Personen zu aufwändig herzustellen.

Neben Vertrauen und psychologischer Sicherheit sind zwei weitere Herausforderungen für Floating Teams zu meisten: Heimatgefühl und langfristige Verantwortung.

Durch die Größe des Produktteams besteht die Gefahr, dass den Teammitgliedern eine Heimat fehlt. Dieses Problem kann durch eine große Nähe zu den Kunden und ihren Problemen gelöst werden. Die Teammitglieder finden ihre Identifikation nicht primär im Team, sondern darin, Kundenprobleme zu lösen.

Darüber hinaus besteht bei Floating Teams die Gefahr, dass sich eine „Fire and Forget"-Haltung einstellt. Schließlich hat man als Mitglied eines Story Teams nach der initialen Entwicklung einer User Story möglicherweise nie wieder mit dem erstellten Code zu tun. Die Mitglieder das Floating Teams sollten vorher erlebt haben, wie es ist, gemeinsam im Team Verantwortung zu übernehmen.

Referenzen

- Mobile Teams sind im LeSS-Framework vorgesehen (und heißen dort Feature Teams).
- Mission Teams finden sich z. B. im Goal#1-Ansatz (siehe https://www.hamburg-startups.net/jimdo-macht-alles-neu-interview-mit-fridtjof-detzner/) oder bei Pipedrive (siehe https://unfix.com/blog/pipedrive-unfixed).
- Floating Teams werden in diesem Vortrag diskutiert: https://youtu.be/rAn0BP9L0ls?si=J4d0I00AELXcHbsX

Kontakt

Stefan Roock, sr@it-agile.de

Gesprächstermin:

https://to.it-agile.eu/termin-stefan-25m