



VOL. 24

Der Softwerker

EXPEDITION
IN DIE TIEFE

Der Ozean

IST DAS AM WENIGSTEN ERFORSCHTE ÖKOSYSTEM DER ERDE, OBWOHL ER MEHR ALS 60 % DER ERDOBERFLÄCHE BEDECKT. DAS LEBEN IN DER TIEFSEE, EINSCHLISSLICH MIKROBEN UND MEERESGRUNDBEWOHNERN, IST AUCH IM ZEITALTER DES TECHNOLOGISCHEN FORTSCHRITTS NOCH IMMER EIN MYSTERIUM.

Lasst uns gemeinsam in diese geheimnisvolle Welt eintauchen.

Liebe Leserinnen und Leser,

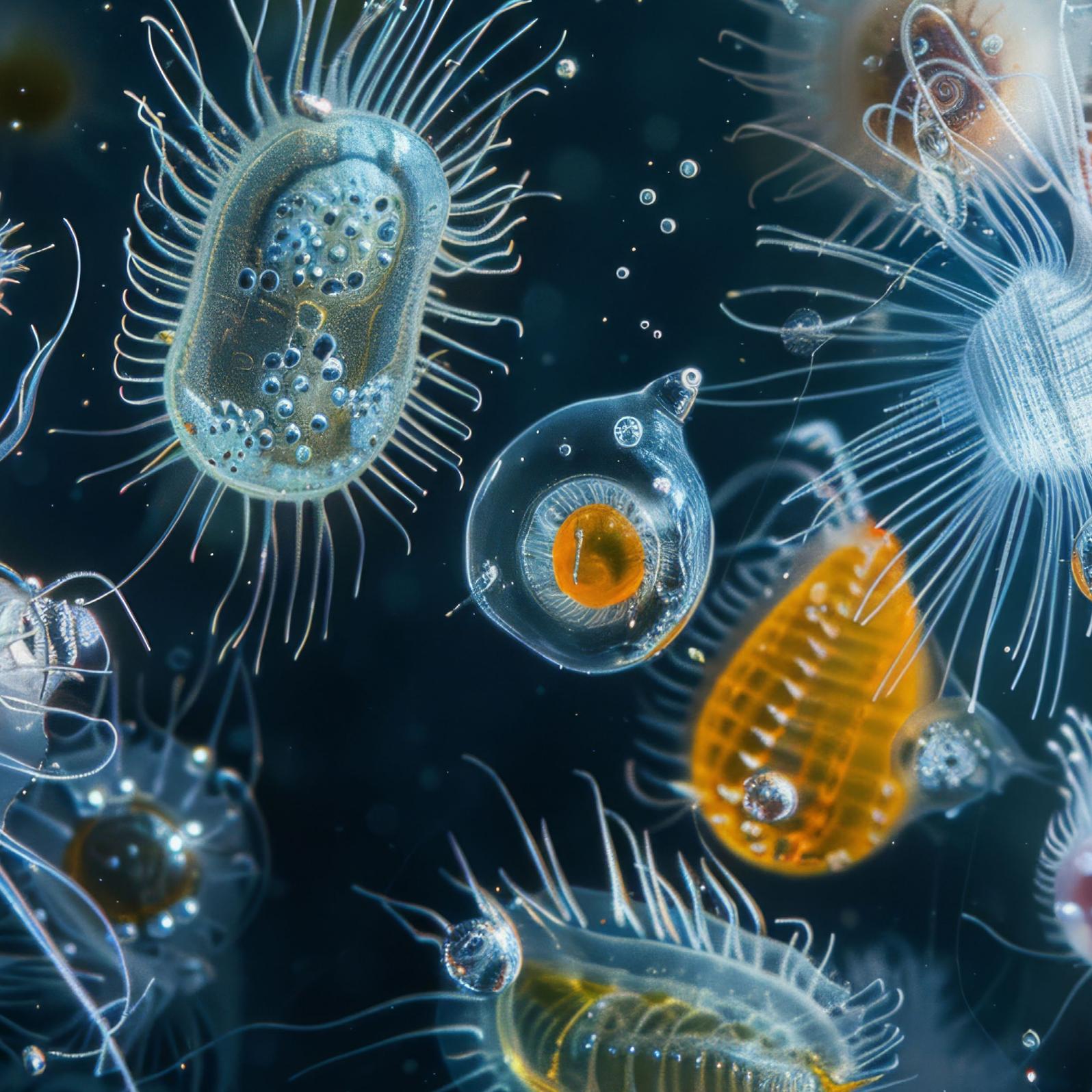
„I'd like to be / Under the sea / In an octopus's garden / In the shade“. Diese Zeilen sangen 1969 – na, wer erinnert sich? – die Beatles. Heute, 55 Jahre später, übt der Ozean noch immer dieselbe Faszination auf den Menschen aus. Und noch immer steckt die Erkundung des größten Ökosystems der Erde in den Kinderschuhen. Die Tiefsee, die den weitaus größten Anteil der Meere ausmacht, ist weniger erforscht als der Mond. Dabei spielen die Bereiche unterhalb von 200 Metern Meerestiefe eine enorme Rolle in unser aller Leben. Tiefsee-Meereströmungen und dort beheimatete Lebewesen beeinflussen beispielsweise das Klima. Und nicht zu vergessen: Sie müssen mit den gewaltigen Mengen Mikroplastik fertig werden, die wir Menschen täglich produzieren und achtlos auf den Grund des Ozeans sinken lassen.

Und was hat das alles mit IT zu tun? Nun, spätestens, wenn ihr in dieser Ausgabe die Artikel von Marco Paga (ab S. 26), Elena Rüdener und Shirin Elsinghorst (ab S. 60) lest, werdet ihr feststellen, dass alles irgendwie miteinander zusammenhängt: Digitale Lösungen haben einen direkten Einfluss auf die Umwelt. Denn sie beanspruchen Ressourcen – Energie, Wasser, Luft, Materialien –, die unser Planet nicht unbegrenzt zur Verfügung stellt. Die gute Nachricht: Ökonomische und ökologische Nachhaltigkeit gehen meist Hand in Hand. Wiederverwendbarkeit und Effizienz (s. die Artikel über Platform Engineering am Anfang dieser Ausgabe), aber auch eine sinnvolle API-Strategie und entsprechende Aufteilung von Software-Teams (s. Artikel von Miriam Greis ab S. 20) sind nicht nur Erfolgsfaktoren für Unternehmen, sondern möglicherweise auch überlebenswichtig für den einen oder anderen Zeitgenossen unter der Meeresoberfläche, den wir euch in dieser Ausgabe vorstellen.

In diesem Sinne wünschen wir euch einen erkenntnisreichen „Deep Dive“ in die IT und die faszinierende Unterwasserwelt!

@codecentric





ATMEN SIE TIEF EIN UND TAUCHEN SIE
EIN IN DEN UNBEKANNTEN OZEAN

Inhalt



Take me Backstage!----- 6

Ein Blick hinter die Kulissen des aufgehenden
Sterns am Platform-Engineering-Himmel
von Manuel Zapf

Platform Engineering mit GitLab----- 12

Der DevSecOps-Tausendsassa als Ausgangspunkt
für interne Developer-Plattformen
von Sujeevan Vijayakumaran, Nils Bauroth und Marc Schnitzius

API Enabling als Erfolgsfaktor----- 20

Damit Schnittstellen wirklich auf die
Geschäftsstrategie einzahlen
von Miriam Greis

Sustainable Software----- 26

Nachhaltigkeit als Chance in der IT
von Marco Paga

Data Products, Data Contracts und API Contracts----- 32

Grundpfeiler moderner Datenarchitekturen als
Erfolgsfaktoren für Organisationen
von Daniel Kocot

Von der Domain Story zum Interface----- 38

Elegante APIs entstehen, wenn Experten auf der
technischen und der geschäftlichen Seite
zusammenarbeiten
von Annegret Junker

Das Product Operating Model----- 48

Die Zauberformel, um neue Technologien
zum Markterfolg zu machen
von Marc Pudelski und Ralf Westbrock

Sechs Säulen für nachhaltige Digitalisierung----- 60

Responsible Digital Solutions als ganzheitlicher
Ansatz für Innovation
von Elena Rüdener und Shirin Elsinghorst

Who's afraid of AI?----- 70

Wie Unternehmen Ängste und Vorbehalte ihrer
Mitarbeitenden gegenüber GenAI überwinden –
und Vertrauen aufbauen können
von Janina Lerner

Mehr als nur Ergebnisse----- 80

Warum Reproduzierbarkeit in Machine Learning
der Goldstandard ist
von Denis Stalz-John

Gut integriert----- 84

Wie GPT und iPaaS Geschäftsmodelle transformieren
von René Rother und Kamil Braschke

Vom Daten-Silo zum Daten-Zoo----- 92

Datenaustausch in Anwendungslandschaften
in den Griff bekommen
von Roger Butenuth

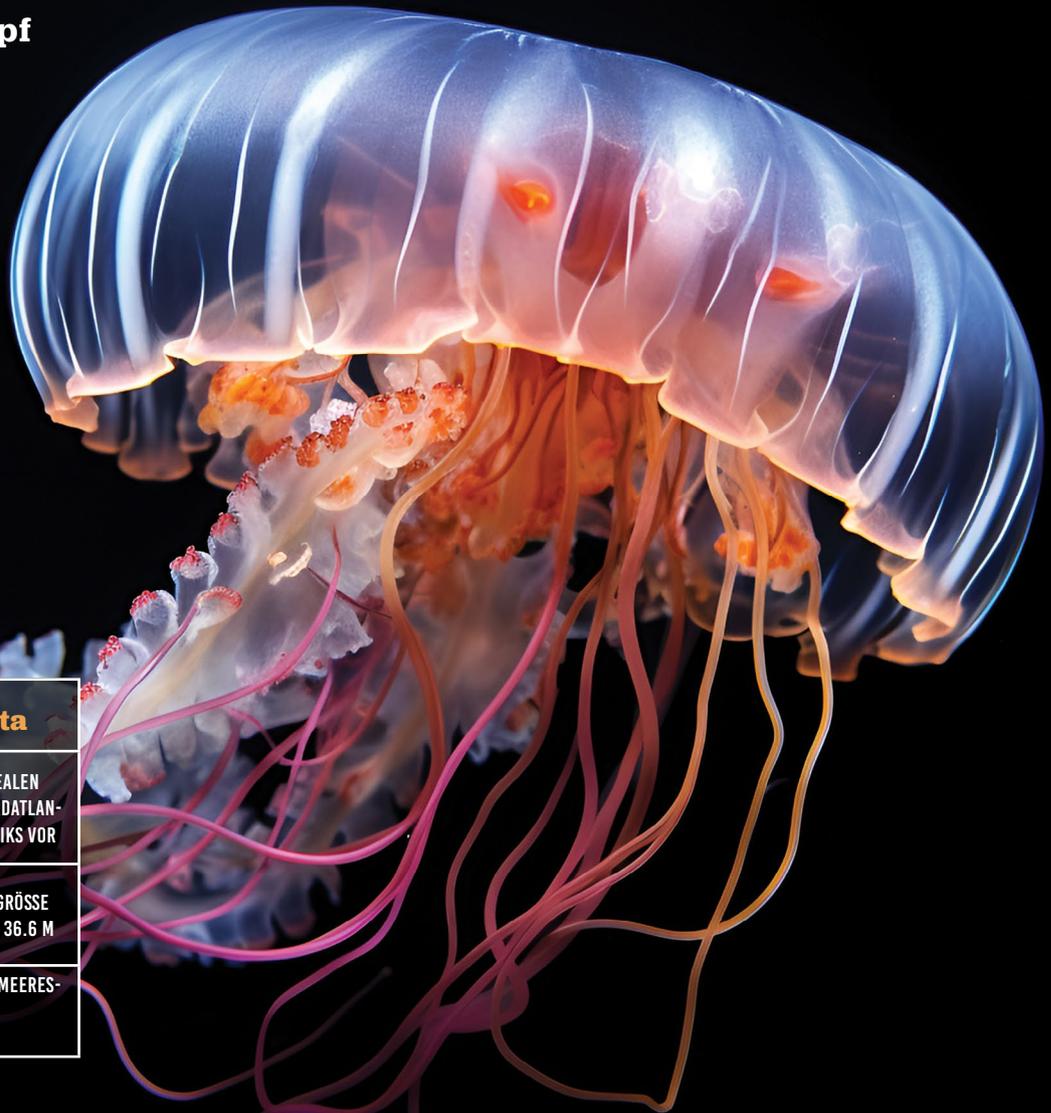
Damit die IT-Forensik nicht weint----- 102

Wie logge ich für den Ernstfall?
von Antonia Schmalstieg

Take me Backstage!

Ein Blick hinter die Kulissen des aufgehenden
Sterns am Platform-Engineering-Himmel

von **Manuel Zapf**



Cyanea capillata

KOMMEN IN DEN KALTEN, BOREALEN
GEWÄSSERN DER ARKTIS, DES NORDATLAN-
TIKS UND DES NÖRDLICHEN PAZIFIKS VOR

LEBEN IN DER NÄHE
DER OBERFLÄCHE
< 20 M TIEFE

GRÖSSE
< 36.6 M

FRESSEN FISCHÉ, ZOOPLANKTON, MEERES-
TIERE UND
KLEINERE QUALLEN

In der letzten Dekade gab es nur wenige Begriffe, die die aktuelle IT-Landschaft so geprägt haben wie DevOps. Lange Zeit waren DevOps und Themen wie Containerisierung, Kubernetes, Cloud-Native etc. so eng miteinander verknüpft, dass die Grenzen hier oft fließend waren. Doch mit wachsenden Anforderungen und komplexen Themenfeldern hat sich eine neue Strömung herausgebildet: Platform Engineering. Letzteres konzentriert sich darauf, eine solide technische Grundlage für die Entwicklung, Bereitstellung und Skalierung von Anwendungen zu schaffen. Backstage hilft dabei, eine Plattform bereitzustellen, die die Verwaltung und Automatisierung von Entwicklungs- und Betriebsaufgaben für große, skalierbare Software-Infrastrukturen erleichtert.

Platform Engineering ist aktuell so etwas wie der heilige Gral oder der „eine Ring“ – sämtliche Probleme in der Bereitstellung von Software in mittleren bis größeren Umgebungen bzw. Unternehmensgrößen sollen sich durch seinen Einsatz beheben lassen. Doch was ist Platform Engineering eigentlich? Wie unterscheidet es sich von dem allseits bekannten DevOps? Warum und wann sollte man Platform Engineering überhaupt einsetzen? Wie kann das Open-Source-Framework Backstage hier Abhilfe schaffen? Fragen über Fragen, die es nun zu beantworten gilt.

Platform Engineering im Überblick

Platform Engineering könnte man als eine Art Evolutionsstufe von DevOps ansehen. Kein Begriff hat die letzten Jahre so getrieben wie DevOps – eine Kultur, Praktik und

Philosophie, die darauf abzielt, die Zusammenarbeit zwischen Entwicklungsteams (Dev) und Betriebsteams (Ops) zu verbessern, um die Softwarebereitstellung und Infrastrukturverwaltung effizienter, kontinuierlicher und automatisierter zu gestalten. Die Ziele von DevOps umfassen die Förderung einer Kultur der Zusammenarbeit, Automatisierung von Prozessen, kontinuierliche Bereitstellung von Software, um die Time to Market zu verkürzen und gleichzeitig die Qualität sowie Stabilität zu erhöhen. Platform Engineering könnte man als eine natürliche Evolution dieses Bestrebens ansehen. Gleichzeitig schließen sich Platform Engineering und DevOps aber nicht gegenseitig aus. Sie gehen eher Hand in Hand. Das Ziel von Platform Engineering ist es, eine Plattform bereitzustellen, die dann selbstständig von Entwicklungsteams genutzt werden kann, um Anwendungen zu erstellen und bereitzustellen. Diese Plattformen können verschiedene Dienste, Tools und Infrastrukturen umfassen, die es Entwicklern ermöglichen, Anwendungen effizienter zu entwickeln, zu testen und bereitzustellen.

Ab wann lohnt sich Platform Engineering?

Diese Frage ist natürlich schwierig zu beantworten, denn die ehrliche Antwort ist, wie so oft: „It depends.“ Platform Engineering lohnt sich der Erfahrung nach am meisten, sobald es mehrere Entwicklerteams gibt, die möglichst frei arbeiten können; dann sind die Teams nämlich in der Lage, am meisten Value beizutragen und sich auf ihr Kernthema zu fokussieren, zum Beispiel die Weiterentwicklung des

Produktes. Um sich hier zu fokussieren, kann es sich dann anbieten, ein eigenes Plattform-Team zu etablieren, das die Internal Developer Platform (IDP) als eigenes, internes Produkt ansieht.

Plattform als Produkt

Plattform als Produkt ist die logische Folge von Plattform Engineering. Da die Plattform nun ein eigenes Produkt ist, ergeben sich hier diverse weitere Aufgaben, die das Plattform-Team abarbeiten muss. Nicht nur muss die Plattform die Funktionen bieten, die die Teams benötigen (Scope); diese muss auch ausgebaut und um weitere Funktionalitäten ergänzt werden (Roadmap). Gleichzeitig muss Support geleistet werden können, denn die Plattform ist nun zentraler Bestandteil des Entwicklungsprozesses. Klassischerweise enthält die Plattform Funktionen wie das Erstellen neuer Infrastruktur, Bereitstellung neuer Services, Dokumentation u. v. m. Durch die getrennten Verantwortlichkeiten und Abhängigkeiten zwischen den diversen Entwicklerteams und dem Plattform-Team benötigt man einen standardisierten Weg, ein Interface für den Umgang mit der IDP. Hierbei ist es wichtig, dass dieses Interface, das vom Plattform-Team bereitgestellt wird, dem entspricht, was die Entwicklerteams heute und in der Zukunft brauchen. Oft wird für dieses Interface heute das Open-Source-Framework Backstage eingesetzt.

Backstage: Erst Spotify, jetzt CNCF

Im Jahr 2020 hat der Musikstreaming Anbieter Spotify Backstage das erste Mal der Öffentlichkeit vorgestellt.

Backstage selbst wurde bei Spotify aus der Not heraus geboren. Spotify hat festgestellt, dass seine diversen Entwicklerteams mehr Zeit damit verschwenden, Informationen zu finden und immer die gleichen Abläufe zu tun, als wirklich am Produkt zu arbeiten. Daher entschied sich Spotify, dieses Problem anzugehen. Die Idee war es, End-to-End-Softwareentwicklung zu zentralisieren und zu vereinfachen, indem man eine Abstraktionsebene schaffte, die über der gesamten Infrastruktur und Entwicklerwerkzeugen liegt, was zur Veröffentlichung von Backstage führte. Da Spotify nicht allein mit diesen Problemen war, erlangte Backstage schnell große Bekanntheit und der Hype um Backstage nahm seinen Lauf. Bereits kurze Zeit nach Veröffentlichung übergab Spotify Backstage in die Hände der Cloud Native Computing Foundation (CNCF) als ein Sandbox-Projekt. Mittlerweile hat Backstage über 25.000 Sterne auf GitHub und der Hype ist ungebrochen.

Backstage: Architektur

Backstage besteht aus drei Komponenten: Core, App und Plug-ins. Den Core von Backstage bilden Basis-Funktionalitäten, die direkt in der Open-Source-Variante verfügbar sind und von der Open-Source-Community implementiert und gepflegt werden bzw. wurden. Eine App ist eine wirkliche Instanz von Backstage. Backstage selbst ist eine TypeScript-Anwendung, die dann in den jeweiligen Umgebungen deployt wird. Eine App, also eine Instanz, kann durch Plug-ins erweitert werden. Plug-ins fügen also immer Funktionalität in genau eine App hinzu.

Welches Core-Feature-Set gibt es?

Backstages Feature-Set ist über die Zeit beachtlich gewachsen. Nicht zuletzt dadurch, dass durch die breite Adoption von Backstage immer mehr Learnings in das Tool einfließen, aber auch diverse Vendors, die sich im Kreis des CNCF bewegen, haben Integration für Backstage bereitgestellt. *Out of the box* stellt Backstage drei Kern-Features bereit:

- Einen Software-Katalog, der sich über alle Komponenten erstreckt (Services, Libraries ...)
- Software-Templates (wiederverwendbare Komponenten, die automatisiert sind), um schnell neue Services etc. bereitzustellen
- Tech-Dokumentation „as Code“
- Kubernetes-Integration

Die Kombination aus diesen drei Funktionalitäten erlaubt bereits die Beantwortung diverser Fragen, z. B.: Welche Ressourcen (Services, API...) sind in meiner Systemlandschaft vorhanden? Wie sind diese dokumentiert oder wo werden bestimmte Services gerade ausgeführt? Dazu gesellen sich Commodity-Features, wie User-Management oder ein Permission-Management. Die Kubernetes-Integration ist neu als Kern-Feature angekommen. Dieses erkennt Services in Kubernetes-Clustern und spiegelt die Informationen in den Service Catalog.

Plug-ins: Customize until it fits

Richtig stark wird Backstage aber über das bereits breite Angebot an Plug-ins. Wie bereits erwähnt, ist Backstage selbst eine TypeScript-Applikation, die durch das integ-

rierte Plug-in-System nahezu beliebig erweitert werden kann, um den mannigfaltigen Anforderungen gerecht zu werden. Stand heute gibt es nun sogar ein neues Plug-in-System, das sich in Backend- und Frontend-Plug-ins unterscheidet, dieses ist aber noch im Alpha-Stadium. Plug-ins in Backstage können genutzt werden, um diverse Funktionalitäten zu erweitern. Gern genutzte Plug-ins sind z. B. die CI/CD-Plug-ins (CircleCI, Argo CD) oder API Docs, um API-Dokumentation direkt im Backstage sichtbar zu machen. Plug-ins selbst sind technisch TypeScript-Libraries, die wiederum auch quelloffen zur Verfügung gestellt werden können.

Einen neuen Service erstellen

Wie kann man sich einen geänderten Ablauf mit Backstage nun vorstellen? Als Beispiel nehmen wir einmal die Situation an, dass das Entwicklungsteam eine neue Datenbank für einen neu zu erstellenden Service benötigt. Der Service soll dabei den geltenden Standards im Unternehmen folgen, was Formatierung und Co. betrifft, sowie in einem Git-Repository existieren und auf einem K8s Cluster deployt werden können. Dies ließe sich in Backstage gut abbilden. In einer Welt ohne Automatisierung müsste sich das Entwicklerteam jetzt diverse Gedanken machen:

- Wie konfiguriere ich meine Datenbank in Bezug auf Backups, Cluster etc.?
- Wie bekomme ich ein Git-Repo?
- Wie richte ich die Pipelines ein?

Backstage kann hier helfen, den Prozess deutlich zu vereinfachen. Das Plattform-Team könnte zwei Software-Templates

bereitstellen: ein Template für die Datenbank sowie eines für eine neue Applikation. Im Datenbank-Template wären all die wichtigen Aspekte wie Backups, Cluster Mode etc. bereits vorkonfiguriert, sodass die Datenbank direkt produktionsreif konfiguriert wäre. Gleichzeitig könnte das Template eine wirkliche Datenbank-Instanz direkt ausrollen, z. B. im Kubernetes-Cluster. Über den Service Catalog wäre diese Datenbank dann auch direkt für das Dev-Team ersichtlich. Im zweiten Schritt könnte nun das Template für Applikationen genutzt werden und diesem die Datenbank als Endpunkt mitgegeben werden. Dieses Template würde dann ein Git-Repository erstellen, vorkonfiguriert mit einer Barbone-Applikation sowie diversen Settings, was den Qualitätsstandard angeht. Darüber hinaus könnte diese Dummy-Applikation ebenfalls direkt im Kubernetes-Cluster deployt werden, über generierte CI/CD-Pipelines.

Ausblick/Fazit

Mit all den Funktionalitäten bietet sich Backstage an, ein zentraler Teil des Platform Engineerings zu werden. Durch die „Single Pane of Glass“ können alle Beteiligten im Unternehmen, egal ob Entwickler, Platform Engineer oder sogar Engineering Manager, ihre benötigten Informationen einsehen. Gleichzeitig kann die Automatisierung dabei helfen, manuellen Aufwand zu reduzieren und den Qualitätsstandard mindestens beizubehalten. Natürlich ist Backstage aber nur ein Tool auf der Reise des Platform Engineerings. Speziell im Kontext von Kubernetes gibt es diverse Tools, die hier auch eine wichtige Rolle spielen können, wie z. B. Crossplane. Abschließend sei gesagt, dass

Tools natürlich einem Vorgehen folgen sollten und nicht umgekehrt. Backstage kann helfen, Platform Engineering zu meistern. Es muss aber auch entsprechend gelebt werden.

*Seit März 2024 ist **Manuel Zapf** als Solution Architect wieder Teil des codecentric-Teams. Thematisch liegt sein Hauptaugenmerk weiterhin mit Leidenschaft auf API-Gateways, Service Meshes und Kubernetes im Allgemeinen. Darüber hinaus ist Manuel weiterhin ein Maintainer des bekannten Cloud-Native-Application-Proxys Traefik. manuel.zapf@codecentric.de*

@codecentric

Kultur & Karriere Podcast

Jetzt neu!
Überall zu hören,
wo es Podcasts gibt.



Hier Reinhören

Platform Engineering mit GitLab

Der DevSecOps-Tausendsassa als Ausgangspunkt
für interne Developer-Plattformen

von **Sujeevan Vijayakumaran, Nils Bauroth und Marc Schnitzius**



| | |
|--|------------------|
| Siphonophorae | |
| LEBEN MEIST IN DER TIEFSEE UND KOMMEN IN ALLEN OZEANEN VOR | |
| LEBEN IN TIEFEN ZWISCHEN 1.600 UND 2.300 METERN | GRÖSSE < 47 M |
| VERWENDEN IHRE TENTAKELN, UM ZOOPLANKTON UND RUDERFUSSKREBSE ZU FANGEN | |

Interne Developer-Plattformen (IDP) sind komplex. Dieser Artikel zeigt, wie eine vorhandene GitLab-Installation als Basis für eine gute IDP dienen kann.

Um Platform Engineering und DevOps voneinander abzugrenzen, lohnt sich ein Blick zurück in die Zeit vor der Cloud-Native-Ära. Etwa um das Jahr 2014, als der Begriff DevOps zunehmend an Aufmerksamkeit gewann, sahen Systemlandschaften grundlegend anders aus als heute in Cloud-nativen Umgebungen. Statt verteilter Service-Architekturen wurden hauptsächlich monolithische Software-Systeme entwickelt und die Infrastruktur war aus Sicht der Entwickler*innen deutlich weniger komplex als heute. In der Regel wurden fertige Software-Builds an IT-Operations übergeben, die sie dann auf Systeme ausrollten.

Über die Jahre hat sich DevOps leider zu einer Praktik entwickelt, die durch ständig neue Tools immer mehr Komplexität im Software-Entwicklungsprozess schafft. Häufig genug liegt die Verantwortung für diese Tools in den Händen der Entwickler*innen. Mit DevOps sollten Barrieren zwischen Operations und Development abgebaut werden, so Komplexität reduziert und kontinuierliche Feedback-Zyklen etabliert werden, um die Zusammenarbeit zwischen beiden Teams voranzutreiben. Wir beobachten in der Realität, dass oft das Gegenteil der Fall ist. Es wird heutzutage oft angenommen, dass tiefgehendes Wissen bspw. in CI/CD, Kubernetes, Helm-Charts, Terraform-Modulen und Cloud-nativen Tools im Repertoire eines Entwicklers vorhanden ist. Das ist jedoch nicht die Regel und auf dem Bewerbermarkt kaum zu finden. DevOps,

so wie es sich verbreitet hat, skaliert also nicht. Die AirBnBs, Netflix und Spotifys dieser Welt haben dieses Problem schon vor einigen Jahren erkannt. Ihre Lösung war es, die Technologien und Werkzeuge im Entwicklungsprozess zu klaren Pfaden zu verbinden, die Developer Self-Service ermöglichen und die kognitive Belastung aller Beitragenden verringern. Heute nennen wir das Platform Engineering. Die Idee dahinter ist, die neu entstandene Komplexität durch Automatisierung zu abstrahieren, so die Standardisierung im Tooling und der Systemlandschaft voranzutreiben und in einheitliche Pfade, auch *Golden Path* genannt, zu gießen. Die Summe dieser Pfade sind dann eine interne Developer-Plattform. Diese wird von einem Plattform-Team nach dem Platform-as-a-Product-Ansatz bereitgestellt. Dabei betrachtet das Plattform-Team die Entwickler*innen als seine Kunden, denen es Self-Service-Angebote für alle Tools und Werkzeuge im Entwicklungsprozess bereitstellt.

Herausforderungen in der Praxis

In vielen Kundenprojekten konnten wir folgende Beobachtungen machen:

1. **Komplexität:** Bei der Betrachtung der Platform Tooling Landscape (siehe **Abb. 1**) fallen schnell die Herausforderungen auf: Komplexität der Anwendungslandschaft inklusive der damit verbundenen Entscheidungsfindung für ein geeignetes Tool.
2. **Betrieb und Wartung:** Die Pflege der zuvor erwähnten Anwendungen etc. erfordert kontinuierliche Aufmerksamkeit und viel Aufwand. Kunden müssen sicherstel-

len, dass eine bereitgestellte Plattform zuverlässig, sicher und skalierbar bleibt. Dies erfordert ein dediziertes Team, das sich bspw. um Aktualisierung und Optimierung der Plattform kümmert.

3. **Standardisierung vs. Flexibilität:** Viele Kunden stehen vor der Herausforderung, die richtige Balance zwischen Standardisierung und Flexibilität zu finden. Zu viel Standardisierung kann die Innovationsfähigkeit und

Agilität von Teams beeinträchtigen. Zu wenig Flexibilität kann Komplexität erhöhen.

4. **Kulturelle Anpassung:** Kunden vergessen oft, dass Platform Engineering auch einen kulturellen Wandel bedeutet. Die Plattform sollte als strategisches Produkt betrachtet werden, einschließlich enger Zusammenarbeit zwischen Entwicklungs-, Betriebs- und Infrastrukturteam.

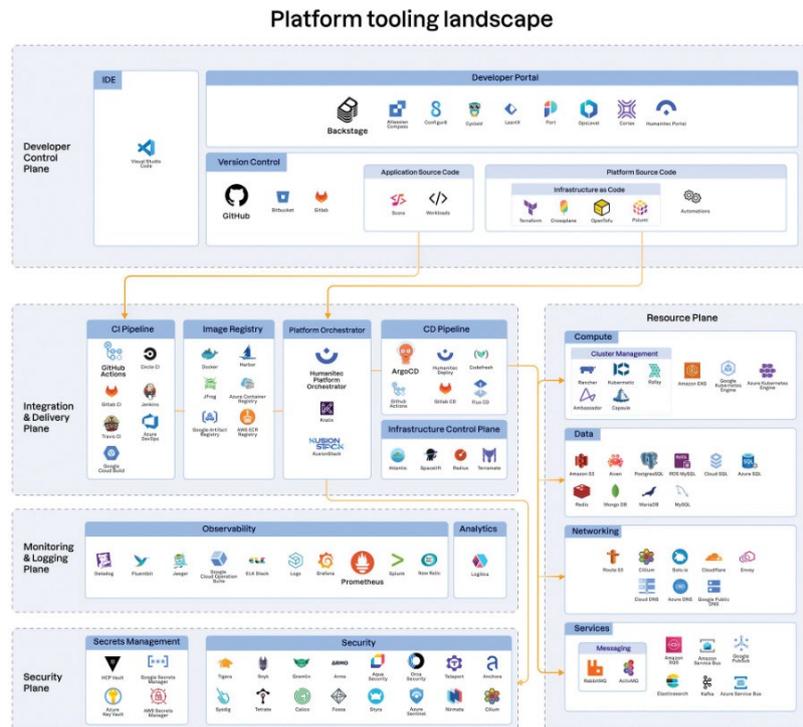


Abb. 1: Platform Engineering Tooling Landscape
(Quelle: <https://platformengineering.org/platform-tooling>)

Um die negativen Auswirkungen zu reduzieren, kann auf ein bereits bestehendes Tooling gesetzt werden.

GitLab Features für IDPs

In vielen Firmen ist GitLab schon die zentrale DevSecOps-Plattform, wo heutzutage nicht nur Sourcecode-Management und CI/CD zusammenkommen, sondern eben auch weitere Aspekte wie Security und Compliance, Package und Container-Registries sowie Observability. Schließlich deckt GitLab den gesamten DevOps-Lifecycle ab.

Spannend ist es daher zu betrachten, welche GitLab-Features auf die Eigenschaften einer IDP einzahlen. Das geht vom Coden über die Web-IDE und die Nutzung von CI/CD Components

im Rahmen von CI/CD über Deployments nach dem GitOps Pattern mit Flux bis hin zu Security- und Compliance-Funktionen.

Project Templates

Software-Projekt werden meistens nur phasenweise frequentiert parallel gestartet. Nach einiger Zeit befindet sich ein Projekt in der Regel in einem Wartungsmodus oder wird nur noch um weitere Features ergänzt. Trotzdem kann es sinnvoll sein, Projekt-Templates anzulegen, um ein vordefiniertes Grundgerüst zu nutzen. In GitLab kann solch ein Projekt angelegt werden, worin eine Basisstruktur definiert ist, die mit allen Dateien daherkommt, die für den Start von neuen Projekten benötigt werden. Dies verkürzt die Ramp-up-Phase, insbesondere, wenn es spezielle Anforderungen im Unternehmen gibt, die in vielen Projekten behandelt werden müssen. Viel wichtiger ist es allerdings eher, den Onboarding-Prozess für Neulinge im Projekt anzuschauen.

Verständlich wird das Ganze mit einem konkreten Beispiel. Ein neuer Microservice soll in Go geschrieben werden. Das Platform-Engineering-Team legt dazu ein Projekt-Template an, worin diverse Dateien und Konfigurationen bereits gesetzt sind. Das ist zum einen ein Go-typisches Boilerplate eines Microservice; aber auch die weiteren üblichen Funktionen von GitLab sind direkt vorkonfiguriert. Etwa eine Pipeline-Definition, die direkt die Komponenten aus dem CI/CD Catalog nutzt – mehr dazu später – oder Issue Templates und Merge Request Templates, um typische Workflows dahingehend vorzugeben.

Web IDE und Remote Workspaces

Nachdem das Projekt also aus dem Template angelegt worden ist, kann das Programmieren ja beginnen. Üblicherweise klonst sich die Person das Repository lokal und fängt zunächst an, die Entwicklungsumgebung einzurichten.

Das birgt aber immer wieder einige Probleme, denn der neue Microservice soll direkt mit anderen Microservices sprechen können, die dafür lokal laufen müssen. Da hierfür zunächst weitere Dokumentation aller relevanter Microservices benötigt wird, war das lange sehr umständlich. Das wurde auch immer wieder deutlich, wenn Entwickler*innen schnell einen Fehler in einem Projekt aus anderen Teams beheben wollten. Das war aus den gleichen Gründen nur schwer möglich, da das Setup recht komplex und – wenn überhaupt – nur teilautomatisiert war.

Das Platform-Engineering-Team hat es sich zur Aufgabe gemacht, auch hier zu unterstützen. Abhilfe schafft jetzt die Web IDE, zusammen mit GitLab Workspaces. Die Idee ist dabei recht simpel: Die Web IDE basiert bei GitLab auf Visual Studio Code und kann direkt im Browser aufgerufen und genutzt werden. Für die Entwicklung des neuen Microservice müssen allerdings ein paar Abhängigkeiten in Containern hochgefahren werden. Hierfür wurde bereits zuvor im Projekt-Template ein `.devfile.yaml` angelegt, worin definiert wird, welche Container für die Entwicklungszwecke benötigt werden. Diese laufen dann in einer isolierten Umgebung, in der die Container mit der Web IDE verbunden sind – das funktioniert ganz unabhängig

vom Rechner des jeweiligen Entwicklers. Eine Änderung beizutragen dauert dann für Projektfremde nur noch wenige Minuten statt einige Stunden.

CI/CD Components im Catalog

Ein großes Thema ist auch die Wiederverwendbarkeit von CI/CD-Pipelines. Denn häufig wurde die Pipeline für die gleichen Probleme immer wieder neu erfunden. Und genau darauf zielt der CI/CD Catalog ab. Denn Probleme, die schon gelöst wurden, sollten idealerweise nicht nur wiederverwendet werden, sondern auch auffindbar sein.

Der CI/CD Catalog fungiert als zentrale Anlaufstelle für Entwickler*innen und Organisationen, indem die wiederverwendbaren Konfigurationen in Form von CI/CD Components UI-gestützt durchsuchbar sind. Eine CI/CD-Komponente stellt dabei einen wiederverwendbaren Teil der Pipeline dar. Dies kann sowohl ein einzelner, für sich abgeschlossener Baustein sein oder eine komplette Konfiguration darstellen.

Für das Go Microservice heißt es etwa, dass im importierten Template bereits die CI/CD-Komponenten aus dem CI/CD-Katalog verwendet worden sind. Nach dem Import muss man also nicht nochmals eine neue Pipeline schreiben, sondern in der Pipeline werden direkt die Komponenten referenziert und somit eingebunden.

GitOps mit Terraform und Flux

Neben dem reinen Bauen und Testen der Anwendung ist das Deployment wichtig. Problematisch ist – und das sieht man leider häufig –, dass ein reines Entwicklungsteam den Betrieb der Anwendung mit übernimmt, wofür vor allem die Zeit fehlt und teilweise auch die Kenntnisse.

Aber selbst wenn ein echtes cross-funktionales Team aus Dev und Ops existiert, gibt es in vielen Firmen diverse Anforderungen, wenn es um das Management von Deployments geht. Hier spielt insbesondere das Management von Infrastrukturen mit Terraform eine bedeutende Rolle, aber auch das Deployment der Anwendung selbst – etwa in einem Kubernetes-Cluster mittels Flux.

GitLab bietet für beide Tools entsprechende Integrationen an. Seitens des Platform-Engineering-Teams können so Abstraktionen für die firmenintern relevanten Terraform Manifests erstellt werden, die von den Teams eingebunden und genutzt werden können. Auch hier ist es elementar, dass die Teams Änderungen per Merge Requests beitragen können, die dann vom entsprechenden Platform-Engineering-Team reviewt werden.

Das gleiche Konzept gilt dann auch dem Deployment der Anwendung selbst, etwa nach dem GitOps-Pattern mittels Flux. Auch hier lassen sich die Integrationen so einrichten, dass alles Hand in Hand geht und mit der korrekten Umsetzung die kognitive Last der einzelnen Teams idealerweise reduziert wird.

Security und Compliance

Sowohl aus DevSecOps-Gesichtspunkten als auch aus Platform-Engineering-Sicht ist die effiziente Handhabung von Security und Compliance essenziell. Wichtig sind hier zwei Punkte: Zunächst müssen die Security-Scans definiert werden und des Weiteren müssen diese auch im Rahmen von Compliance-Richtlinien erzwungen werden.

Der Fokus vom Entwicklungsteam ist die effiziente Weiterentwicklung der Anwendung. Durch das frühe Einbinden

automatischer Security-Scans sollte es dem Team einfach gemacht werden, potenzielle Sicherheitslücken frühzeitig zu finden und zu korrigieren, idealerweise spätestens dann, wenn in einem Merge Request die Änderung zum Review bereitsteht. Das Security-Team ist hingegen für die Auswahl der richtigen Scanner und für die Einbindung zuständig.

Durch die Definition von Compliance-Frameworks in die Projekte kann so sichergestellt werden, dass bestimmte Regularien eingehalten werden müssen. Scan Execution

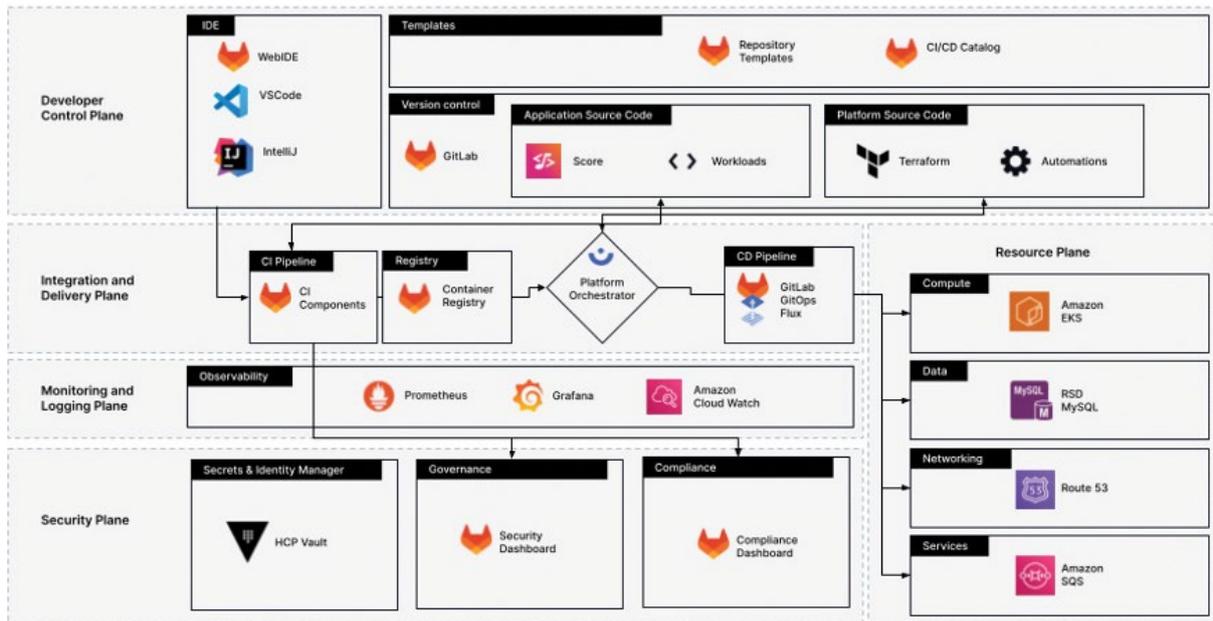


Abb. 2: Referenzarchitektur mit GitLab und Humanitec

Policies sind etwa dazu hilfreich, die Konfiguration zu ermöglichen und auch zu erzwingen. Mittels Merge Request Approval Policies wird es zudem möglich, dass Merge Requests ein zusätzliches Approval vom Security-Team benötigen, für den Fall, dass neue Sicherheitslücken eingeführt werden. Wenn keine Sicherheitslücken eingeführt werden, wird ein solches Approval nicht benötigt. Hier kommt dann automatisch eine engere Kollaboration zustande, denn beide Teams sehen frühzeitig etwaige Probleme und können gemeinsam an der Lösung arbeiten. Gleichzeitig werden Compliance-Anforderungen eingehalten.

Fazit

Stellt man **Abb. 1** und **2** gegenüber, lässt sich Folgendes recht schnell erkennen: GitLab als Ausgangspunkt für eine interne Developer-Plattform stellt eine solide Basis dar, um die Komplexität und Vielschichtigkeit interner Entwicklungsplattformen zu reduzieren. Die meisten Kunden stehen nicht vor Herausforderungen wie beispielsweise Netflix oder Spotify. Ein vollumfänglicher Plattform-Ansatz ist oftmals überdimensioniert. GitLab ermöglicht eine effektive Umsetzung von Platform-Engineering-Prinzipien, darunter die Automatisierung und Standardisierung von Tools und Prozessen mit Features wie Projektvorlagen, CI/CD-Kataloge, GitOps mit Terraform und Flux sowie integrierte Security- und Compliance-Funktionen. Insgesamt kann eine vorhandene GitLab-Installation schon dazu beitragen, die kognitive Belastung der Entwicklungsteams zu verringern und eine effizientere und produktivere Entwicklungsumgebung zu schaffen.

***Sujevan Vijayakumaran** arbeitet als Senior Solutions Architect bei GitLab. Seit einigen Jahren unterstützt er große Konzerne aus Deutschland, Österreich und der Schweiz beim Ein- und Umstieg in die DevOps-Welt. Er ist zudem Autor von Büchern über DevOps und Git. svijayakumaran@gitlab.com*

***Nils Bauroth** arbeitet als Cloud Consultant bei der cc cloud GmbH. Er begleitet Unternehmen bei der Transformation in Richtung GitLab, häufig im regulierten Umfeld. nils.bauroth@codecentric.cloud*

***Marc Schnitzius** ist IT Consultant mit Fokus auf DevOps-Methodiken und Infrastrukturthemen. Er hat einen mehrjährigen Hintergrund in der klassischen Linux-Systemintegration und berät für codecentric rund um die Themen Cloud Infrastructure, Automatisierung und Platform Engineering. Vor allem Fragen rund um skalierbare Entwicklungsprozesse und die dazugehörige Infrastruktur sind genau sein Metier. marc.schnitzius@codecentric.de*

ONLINE
SOFTWAREWERKER
KONF



Auf den Schirm!

Softwarewerker-Wissen in
bewegten Bildern



Alle Vorträge von den
SoftwarewerkerKonfs



www.youtube.com/codecentricAG

API Enabling als Erfolgsfaktor

Damit Schnittstellen wirklich auf die
Geschäftsstrategie einzahlen

von **Miriam Greis**

Copepoda

KOMMEN IN FAST JEDEM SÜSS- UND
SALZWASSERLEBENSRAUM VOR: VON DER
NORDSEE UND DEM NORWEGISCHEN MEER
BIS ZUR BARENTSSEE

ERNÄHREN SICH VON
PHYTOPLANKTON

GRÖSSE
< 2 MM

SIND WICHTIG FÜR DIE GLOBALE ÖKOLOGIE
UND DEN KOHLENSTOFFKREISLAUF



Ein Enabling Team ist einer der vier Teamtypen des Frameworks Team Topologies. Wie lässt sich dieses Muster erfolgreich auf das Design und die Entwicklung von APIs anwenden?

APIs sind ein wichtiger Baustein der Digitalisierung und so wächst der Druck auf Entwicklungsteams, APIs zur Verfügung zu stellen, stetig. Doch gerade wenn Entwicklungsteams hauptsächlich im ERP-Umfeld oder in Backend-Systemen unterwegs sind, fehlt ihnen meist das Wissen und die nötige Erfahrung dazu. Unter dem oftmals herrschenden Zeitdruck entstehen dann APIs, die nicht wiederverwendbar, schlecht wartbar, unsicher und schlecht dokumentiert sind. Dies frustriert und verlangsamt die Entwicklungsteams, die auf APIs angewiesen sind, erheblich und hat damit auch negative Auswirkungen auf die Innovationsfähigkeit einer Organisation. Ein API Enabling Team, bestehend aus Expertinnen und Experten, kann Teams bei ihren ersten Gehversuchen unterstützen und so sicherstellen, dass qualitativ hochwertige APIs entstehen, die auf die Geschäftsstrategie einzahlen und den Erfolg der Organisation fördern. Dieser Artikel beleuchtet die Problematik, die Voraussetzungen und auch die Chancen von API Enabling als wichtigen Erfolgsfaktor, um APIs als Bausteine der Digitalisierung zu etablieren.

Das Problemfeld

Warum tun sich viele Entwicklungsteams überhaupt schwer damit, qualitativ hochwertige APIs zu bauen? Gerade Entwicklungsteams, die hauptsächlich mit Altsystemen arbei-

ten oder ein sehr spezialisiertes Skillset haben, fehlt das benötigte Wissen. Viele Praktiken der modernen Softwareentwicklung sind bei ihnen noch nicht angekommen. Genau diese sind für die Entwicklung qualitativ hochwertiger APIs aber unerlässlich.

In großen Organisationen ist zusätzlich eine Vielzahl an Teams für unterschiedliche Services zur Bereitstellung von Schnittstellen zuständig. Das betrifft beispielsweise Services wie Integration, Middleware wie API Management oder Autorisierung. Sich in diesem Dschungel zurechtzufinden, ist bei noch nicht entwickeltem Skillset besonders schwierig, stellt aber auch erfahrene Entwicklungsteams auf die Probe. Zusätzlich werden den Entwicklungsteams oft Anforderungen „über den Zaun geworfen“. Das heißt, sie kennen die Stakeholder ihrer APIs gar nicht oder kaum und haben so auch keine Möglichkeit, sich mit ihnen zu identifizieren oder ein Verständnis davon aufzubauen, was sie ihren Stakeholdern liefern müssen, um diese zufriedenzustellen. Die Nutzer*innen werden so nicht als Kunden wahrgenommen, sondern als Personen, die Mehraufwand schaffen und dem Team eine Tätigkeit abverlangen – die Entwicklung von APIs – die das Team nicht zu seinen Kernaufgaben zählt und es vor große Herausforderungen stellt.

Team Topologies als Grundlage

Das Framework Team Topologies [1] liefert die Grundlage für eine effiziente Entwicklung von Software. Es unterscheidet zwischen vier verschiedenen Typen von Teams: *Stream-Aligned* (geschäftorientiert), *Enabling* (unterstützend), *Complicated Subsystem* (spezialisiert) und *Platform*

(beschleunigend). Die Teams interagieren anhand dreier verschiedener Interaktionsmuster entweder kollaborativ, *as a Service* oder durch Facilitating. **Abb. 1** zeigt beispielhaft die Interaktion zwischen verschiedenen Team-Typen. Ein Enabling Team besteht aus Expertinnen und Experten, die Wissen vermitteln und das Skillset der Stream-Aligned Teams erweitern, indem sie diesen die Evaluierung und Erprobung von neuen Werkzeugen, Methoden, Frameworks u. ä. abnehmen. Dabei unterstützen sie Stream-Aligned Teams nur über einen bestimmten Zeitraum hinweg und nicht dauerhaft.

Dieser Artikel beleuchtet, inwieweit ein Enabling Team speziell für APIs, im Folgenden „API Enabling Team“ genannt, Stream-Aligned Teams dabei unterstützen kann, ihr Skillset zu erweitern, um sie fit für das Design und die Implementierung von APIs zu machen.

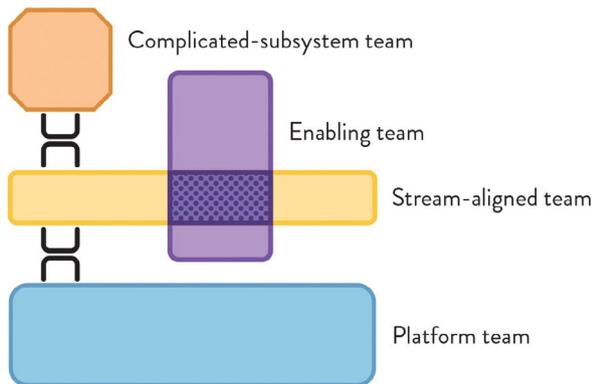


Abb. 1: Beispielhafte Interaktion der Teamtypen des Frameworks Team Topologies. Aus dem Buch Team Topologies von Matthew Skelton und Manuel Pais [1]. Hier veröffentlicht mit freundlicher Genehmigung der Autoren.

Aufgabengebiete eines Enabling Teams für API

Ein API Enabling Team hat primär die Aufgabe, andere Teams zu unterstützen und diese bzgl. APIs zu schulen. Dafür muss das Team ein Workshop-Konzept entwickeln, das sowohl moderne Standards als auch die Gegebenheiten und die API-Strategie der jeweiligen Organisation berücksichtigt. Bestenfalls unterstützt das Team die Stream-Aligned Teams beim Design und der Entwicklung ihres ersten API. Wichtig ist hierbei, dass das Enabling Team nur unterstützende Tätigkeiten leistet und nicht selbst entwickelt. Da das API Enabling Team nicht jegliche Backend-Technologie in der Organisation kennen kann, müssen die Stream-Aligned Teams ihre Technologie ausreichend beherrschen. Nach der Unterstützung durch das Enabling Team stellt es Dokumentation zur Verfügung, die es den Stream-Aligned Teams ermöglicht, denselben Prozess beim nächsten Mal eigenständig zu durchlaufen. Natürlich bleibt es aber auch zukünftig für Spezialfragen, Reviews u. ä. ansprechbar.

Ein API Enabling Team ist am wirkungsvollsten, wenn es zusätzlich bei weiterführenden Themen eingebunden wird und Einfluss auf die API-Strategie, Governance und Operations nehmen kann, damit die Erfahrungen mit den einzelnen Teams dort aufgegriffen und bestenfalls abgebildet werden können.

Ein API Enabling Team trägt außerdem dazu bei, eine lebendige API-Community aufzubauen, da es die nötigen Kontakte zu den einzelnen Entwicklungsteams aufbaut und sie so untereinander vernetzen kann.

Einer Philosophie folgen

Um als API Enabling Team erfolgreich zu sein, muss sich das Team an den Architekturvorgaben und der API-Strategie der Organisation orientieren und daraus seine Philosophie ableiten.

Basierend auf den Architekturvorgaben der Organisation leitet das API Enabling Team einen *goldenen Pfad* ab und identifiziert die Vorgehensweisen und Werkzeuge, die von den Stream-Aligned Teams beachtet und verwendet werden müssen. Dies können beispielsweise bestimmte Architekturmuster, ein vorgeschriebenes API-Management-Tool oder vorgeschriebene Qualitätsmerkmale sein.

Basierend auf der API-Strategie der Organisation ergänzt das Enabling Team seine Philosophie mit konkreten API-Themen. Dabei stehen vor allem Aspekte wie Wiederverwendbarkeit, APIs as a Product, nutzerzentrierte Entwicklung und API Design first vs. Code first im Vordergrund.

Wichtig ist dabei, dass das API Enabling Team seine Philosophie transparent macht und für alle seine Handlungen und Workshops zu Grunde legt.

Basierend auf der Philosophie ein Workshop-Konzept entwickeln

Basierend auf seiner aus der Organisation abgeleiteten Philosophie entwickelt das API Enabling Team, wie in Abb. 2 dargestellt, dann ein passgenaues Workshop-Konzept.

Das Workshop-Konzept besteht aus einzelnen Bausteinen und muss die unterschiedlichen Wissensstände verschiedener Stream-Aligned Teams berücksichtigen. Folgende

Themen sollten mindestens durch das Workshop-Konzept abgedeckt sein:

- Grundsätzliche Funktionsweise eines API
- nutzerzentrierte API-Entwicklung und Einbeziehung von Stakeholder sowie Developer Experience
- verschiedene Interaktionsmuster und deren Einsatzgebiete
- Erstellung eines Datenmodells
- API-Design und Standardformate (z. B. OpenAPI)
- API-Richtlinien, Standards und hilfreiche Werkzeuge in der Organisation und darüber hinaus
- API-Management in Verbindung mit den Themen Sicherheit und Authentifizierung/Autorisierung
- Backend-Entwicklung (soweit möglich)
- Testen

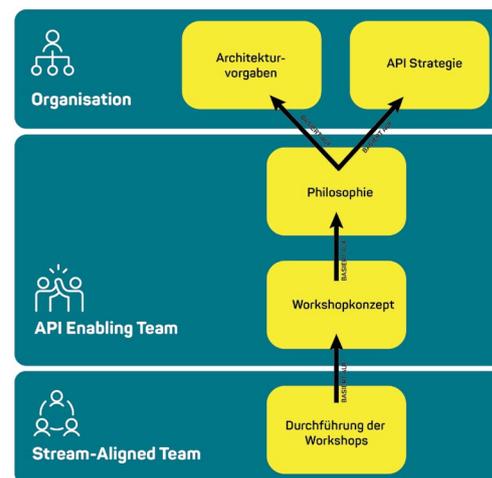


Abb. 2: Ableitung des Workshop-Konzepts über die Philosophie [1]

Die Entwicklung eines solch umfassenden Workshop-Konzepts benötigt Zeit und Expertise. Die Weiterentwicklung und Verbesserung des Konzepts basierend auf den Erfahrungen mit den Stream-Aligned Teams ist unerlässlich.

Das Workshop-Konzept umsetzen

Um das Workshop-Konzept zu verproben, geht das API Enabling Team zunächst gezielt auf ausgewählte Stream-Aligned Teams zu, die offen für Neues und direkten Austausch sind. Nach der Verprobung bekommen alle Stream-Aligned Teams die Möglichkeit, das API Enabling Team in Anspruch zu nehmen, um ihr erstes qualitativ hochwertiges API zu entwickeln. Zunächst klärt das API Enabling Team immer die Voraussetzungen und Erwartungen mit dem jeweiligen Stream-Aligned Team ab und macht seine Philosophie transparent. Im Anschluss muss geklärt werden, dass APIs wirklich die richtige technische Lösung für die Aufgaben des Stream-Aligned Teams sind. Oftmals stellt sich dabei heraus, dass das gar nicht der Fall ist. Ist dies jedoch gegeben, können im Folgenden gemeinsam die nötigen Workshops und Arbeits-Sessions geplant werden, so wie es für das Stream-Aligned Team am besten in den Arbeitsalltag passt. Nach der Durchführung der Workshops holt das Enabling Team systematisch Feedback ein und integriert das Stream-Aligned Team in eine API-Community, wo es sich zukünftig mit anderen Entwicklungsteams zu Problemen, Herausforderungen und Erfolgen austauschen kann.

Wichtig ist dabei jedoch, dass kein Stream-Aligned Team zu einem Enabling gezwungen wird, wenn es das nicht

möchte. Die Erfahrung zeigt jedoch, dass sich gutes Enabling herumspricht und durch Mund-zu-Mund-Propaganda unter den Teams verbreitet wird, sodass die Nachfrage nach dem Enabling automatisch steigt.

Plattform-Aspekte

Um als API Enabling Team effektiv und effizient zu arbeiten, muss der generelle Prozess, wie APIs in der Organisation erstellt werden, so schlank und klar wie möglich sein. Das Team muss in die Lage versetzt werden, mit anderen Teams gemeinsam den Prozess zu verbessern. Es bietet sich an, auch ein API Platform Team einzusetzen, wenn viele Teile des bisherigen Prozesses nicht automatisiert sind. Automatisierte API Operations unterstützen das Enabling Team dabei, Wissen schneller zu vermitteln und die Services den Stream-Aligned Teams bekannt zu machen, als für jedes einzelne Stream-Aligned Team das Rad neu zu erfinden. Die Services tragen gleichzeitig dazu bei, die kognitive Last für die Stream-Aligned Teams zu reduzieren, die während des Enablings sowieso schon sehr viel Neues lernen müssen.

Voraussetzungen für den Erfolg

Um die Einführung eines API Enabling Teams erfolgreich zu gestalten, müssen einige Voraussetzungen erfüllt sein. Zunächst ist es wichtig, dass das Team den Support des Managements hat. Da das Team selbst keine Software schreibt und somit keine vorzeigbaren Produkte liefert, muss das Management von dessen Wichtigkeit überzeugt sein und auch die richtigen Metriken zur Beurteilung heranziehen. Die Anzahl der erstellten APIs, die dabei

gerne als Metrik herangezogen wird, ist beispielsweise nur bedingt aussagekräftig, da sie weder etwas über die Qualität der APIs aussagt noch Teams berücksichtigt, für die ein API gar keine sinnvolle Lösung war.

Ein erfolgreiches API Enabling Team arbeitet auch mit vielen anderen Teams zusammen, hauptsächlich betrifft dies Platform, Complicated Subsystem und andere Enabling Teams (z. B. Autorisierungsteams, Integration Teams etc.). Diese Teams müssen zum Austausch mit dem Enabling Team bereit sein und ihnen die nötigen Informationen zur Verfügung stellen, die sie dann kondensiert und aufbereitet mit den Stream-Aligned Teams teilen.

Eine weitere wichtige Voraussetzung für den Erfolg eines API Enabling Teams ist es, die Stream-Aligned Teams zur Zusammenarbeit mit dem Team zu motivieren. Dies kann beispielsweise nur schwer gelingen, wenn die Kosten für das Enabling Team intern auf die Stream-Aligned Teams umgelegt werden. Der Service, den das API Enabling Team bietet, sollte für alle Teams kostenfrei und so einfach wie möglich in Anspruch zu nehmen sein.

Fazit

Ein API Enabling Team, basierend auf den in *Team Topologies* [1] vorgestellten Team-Typen, unterstützt Stream-Aligned Teams erfolgreich darin, ihr erstes API zu veröffentlichen. Eine Organisation, die auf ein API Enabling Team setzt, kann sicherstellen, dass qualitativ hochwertige APIs entstehen, die auf die Geschäftsstrategie und den Erfolg der Organisation einzahlen. Dies erreicht das Enabling Team durch ein passgenau zugeschnittenes Work-

shop-Konzept, das dafür sorgt, dass Stream-Aligned Teams alle wichtigen Aufgaben rund um die Erstellung, Entwicklung und Veröffentlichung von APIs erlernen und zukünftig selbst beherrschen. Der Erfolg eines API Enabling Teams ist eng gekoppelt an den Rückhalt im Management, die Bereitschaft für die Zusammenarbeit anderer zentraler Teams und die Motivation der Stream-Aligned Teams. Nur wenn diese Voraussetzungen erfüllt sind, kann das API Enabling Team zum entscheidenden Erfolgsfaktor werden.

Durch ihre Promotion in der Mensch-Computer-Interaktion lag Dr. Miriam Greis' technischer Schwerpunkt ursprünglich auf der Frontend-Entwicklung. Mit der Zeit hat sie jedoch ihre Begeisterung für APIs entdeckt. Deshalb ist sie aktuell im API Consulting als Enablerin mit dem Schwerpunkt API Experience & Operations unterwegs. Dabei legt sie ein besonderes Augenmerk auf die kontinuierliche Verbesserung und Automatisierung von Prozessen. Nicht nur im Projekt, sondern auch darüber hinaus gibt sie ihr Wissen sehr gerne in Schulungen oder Workshops weiter und liebt es, Talks zu halten. Auch beim Girls'Day ist sie jedes Jahr dabei, um Mädchen für einen Beruf in der IT zu begeistern.
miriam.greis@codecentric.de

Referenzen

[1] Matthew Skelton, Manuel Pais (2023). *Team Topologies: Organisation von Business- und IT-Teams für einen schnellen Arbeitsfluss*. O'Reilly

Sustainable Software

Nachhaltigkeit als Chance in der IT

von **Marco Paga**

Leptocephalus

LEBT IN ALLEN OZEANEN WELTWEIT

ERNÄHRT SICH VON
WINZIGEN PARTIKELN, DIE
FREI IM MEER SCHWIMMEN

GRÖSSE
5-30 CM

LEBT HAUPTSÄCHLICH IN DEN OBEREN 100
METERN DES OZEANS



Nachhaltigkeit wird schnell mit Einschränkungen gleichgesetzt. Das aktuell viel diskutierte Thema hat jedoch ein riesiges Potenzial als Nordstern für positive Veränderungen, gerade in der IT, und erlaubt es, auch in Zukunft erfolgreich zu sein – Stichwort: Sustainable Software.

Nachhaltigkeit wird immer relevanter und gehört zu einer langfristigen Unternehmensstrategie für erfolgreiche Unternehmen. Innerhalb der codecentric AG haben alle Berater*innen die Möglichkeit, ca. 20 % ihrer Arbeitszeit in relevante Themen der Weiterentwicklung zu investieren. Diese Zeit erlaubt es, wichtige Themen zu finden und diese für uns und unsere Kunden mit Leben zu füllen. Nachhaltigkeit – mit all ihren Facetten – war ein Thema, das immer wieder verfolgt wurde. Von Charity- über Klima-Initiativen bis hin zu internen Mental-Health-Programmen. Mit der erfolgreichen B-Corp™-Zertifizierung [1], die Aufnahme in ein renommiertes Nachhaltigkeitsprogramm, haben wir ein gemeinsames Dach für diese Initiativen bekommen. Diese Zertifizierung stellt sicher, dass wir die wichtigen Bereiche beleuchten und, aus meiner Sicht viel wichtiger, bei jeder regelmäßig stattfindenden Rezertifizierung immer besser werden.

Sehr oft wird der Begriff der Nachhaltigkeit gleichgesetzt mit ökologischer Nachhaltigkeit, was für viele einen Verzicht bedeutet. Das Thema ist jedoch größer und bietet Chancen für positive Auswirkungen für die Firma, die Mitarbeiterinnen und Mitarbeiter, sowie die Gesellschaft. Der Begriff der Nachhaltigkeit beschreibt die Idee, ein

System nicht auszubeuten, sondern für die Ewigkeit zu erhalten. Wichtig hervorzuheben ist das damit verbundene Mindset. Für mich bedeutet es, dass das Thema permanent mitgedacht wird und auch über die vermeintlichen Grenzen hinweg geschaut wird.

Bei vielen mittelständischen Unternehmen sehe ich den Aspekt des langfristigen Blicks, denn auch hier herrscht ein generationsübergreifendes Denken mit einem Blick über Dekaden hinweg. Das ist ein guter Ansatz, um auch diese Langfristigkeit mit IT-Landschaften und Systemen in Verbindung zu bringen.

Das bekannteste Modell der Nachhaltigkeit ist das Drei-Säulen-Modell [2], das ökologische, ökonomische und soziale Nachhaltigkeit betrachtet. Beispiele für die drei Säulen aus der Praxis einer Firma:

- Die **ökonomische Nachhaltigkeit** betrachtet die Felder, die den wirtschaftlichen Erfolg langfristig beeinflussen. Fragen sind hier z. B.: Wie muss die Firma auf veränderte technische Rahmenbedingungen reagieren? Muss das Geschäftsmodell angepasst werden? Aktuelle Auslöser sind z. B. die Weiterentwicklungen im Bereich der künstlichen Intelligenz oder auch der gesellschaftlich absehbare Faktor der Fachkräfteknappheit.
- **Ökologische Nachhaltigkeit** betrachtet die Reduktion der eigenen negativen Emissionen wie CO₂ und Verbräuche von nicht nachwachsenden Rohstoffen bzw. die Wiederverwendung von Stoffen und Produkten und Code.
- Die **soziale Nachhaltigkeit** betrachtet die Auswirkungen auf die Gesellschaft und die eigenen Mitarbeiterin-

nen und Mitarbeiter. Themen sind hier z. B., welche Arbeitsmodelle angeboten werden und wie eine Zusammenarbeit insgesamt aufgebaut ist. Weiterhin sind in Bezug auf die Gesellschaft positive Engagements zu betrachten.

Eine Erweiterung um technische Nachhaltigkeit ist in diesem Zusammenhang sehr wichtig. Diese wird in verschiedenen Modellen beschrieben. Da ich an dieser Stelle pragmatisch sein möchte, beziehe ich mich auf das arc42 Quality Model, das an dieser Stelle Quality Requirements definiert, die einen guten Überblick geben. Attribute wie Wartbarkeit, Ausfallsicherheit, Verfügbarkeit sind hier die wichtigsten Kriterien, um Lösungen dahingehend zu bewerten.

Neben diesem Modell bilden die Kriterien der UN eine praxisrelevante Einordnung für die zu erreichenden Ziele. Die 13 Sustainable Development Goals (SDG) sind innerhalb der bunten Kacheldarstellung oft zu sehen [3]. Jede einzelne Kachel beschreibt ein Ziel. Diese Ziele gilt es für Unternehmen im Rahmen der CSRD zu beleuchten und zu optimieren, was im Rahmen eines Nachhaltigkeitsberichts erfolgt. Die Zusammenhänge wären ein eigener Artikel, daher an dieser Stelle nur dieser Hinweis.

Sustainable Software

Ein Begriff, der ebenfalls in diesem Zusammenhang sichtbar wird, ist die *Twin Transformation*, denn hierbei geht es um die digitale Transformation mit dem Nordstern der Nachhaltigkeit. Wobei die digitale Transformation die erste anstehende Transformation ist und im weiteren Ver-

lauf die weiteren Umstellungen vorgenommen werden mit dem gleichbleibenden Leitmotiv der Nachhaltigkeit. Die Twin Transformation bezieht sich auf die gesamte Firma, hingegen bezieht sich die Idee der Sustainable Software konkret auf die IT-Abteilungen. Die Handlungsfelder und Patterns sind auf diese zugeschnitten. Der Anspruch ist die große Idee, konkrete Ansätze zu schaffen, um Lösungen anzubieten.

Die Frage, die sich stellt, ist, wie eine nachhaltige Entwicklung digitaler Systeme in der Praxis aussieht. Es beeinflusst alle Aspekte des Software Development Lifecycle von der Umsetzungsidee bis zur produktiven Nutzung in allen Phasen. Der Aufbau und die Zusammenarbeit innerhalb der IT-Abteilungen und das Zusammenspiel mit den Fachabteilungen sind in dieser Betrachtung ebenfalls eine wichtige Dimension.

Im Folgenden beschreibe ich beispielhaft Ideen aus dem Entwicklungsprozess von Software, zeige den Vorteil für die Firma auf und ordne diese in die SDGs ein, um die Relevanz zu zeigen. Dies ist eine erste Inspiration für das Thema, um zu zeigen, an wie vielen Stellen es relevant und hilfreich ist.

Das große Ganze

Welche Themen fallen also in die Betrachtung? Softwarequalität gehört für mich dazu. Durch eine sinnvolle Architektur und passende Tests baue ich ein Fundament, um weitere Entwicklungen risikoarm zu ermöglichen. Tests können komplexe Logik verifizieren und Systeminteraktionen automatisiert sicherstellen.

Mit vorhandenen Tests hat ein Team die Möglichkeit, von Automatisierungen zu profitieren. Damit ist es möglich, mit Zuversicht schnell am Markt zu agieren. Die Automatisierung ist im Idealfall über Infrastructure-as-Code(IaC)-Tools bereitgestellt, damit die Vorteile der Reproduzierbarkeit und Versionskontrolle genutzt werden können.

Darüber hinaus ist es wichtig, das Zusammenarbeitsmodell der Software-Teams zu hinterfragen. Sind die Ziele und der Qualitätsrahmen bekannt, können Entscheidungen schnell getroffen werden? Machen wir das Bild umfassender: Welche Maßnahmen gibt es, um Ausgrenzungen zu vermeiden und eine möglichst breite Gruppe von Menschen anzusprechen, um im Unternehmen zu arbeiten?

Viele der notwendigen Schritte sind uns bekannt, jedoch hilft die Idee der Nachhaltigkeit, die Schritte konsequent zu verfolgen und auf lange Sicht erfolgreich zu sein. Wir kennen leider alle die Hacks, die uns kurzfristig schneller machen, jedoch auch schon mittelfristig zur Belastung werden.

Diese Betrachtung und Herangehensweise zahlt auf SDG 9 – „Industry, Innovation and Infrastructure“ [3] – ein. Denn durch diese wird die Innovationskraft erhöht, da der Fokus auf den konkreten fachlichen Themen liegt.

Sustainable Platform Engineering

Aus meiner Sicht ist ein sehr großer Hebel das Sustainable Platform Engineering (s. die drei ersten beiden Artikel dieser Ausgabe). Schauen wir einmal kurz hinein: Wie sieht die Entwicklung aus, wenn wir keine definierte Plattform

mit vorhandenen Lösungsbausteinen haben und möglichst schnell eine fachlich motivierte Lösung aufbauen wollen? Wir können uns nicht sofort mit den fachlichen Herausforderungen beschäftigen, sondern müssen erstmal die technischen Hürden meistern. In welcher Laufzeitumgebung soll die Applikation laufen? Welche weiteren Systeme werden benötigt und welche APIs sind dafür verfügbar? Wie binden wir das IAM an? Welche Datenbanken? Welche Polycys? Und dann geht es in die einzelnen Systeme, um Code-Repositorys und Pipelines anzulegen.

Eine Plattform adressiert diese Schritte durch einen *Golden Path*, der möglichst automatisiert die Schritte abnimmt und vieles bereitstellt. Eine etablierte Plattform automatisiert die Bereitstellung von Repositorys und Pipelines. Darüber hinaus werden Building Blocks für die Infrastruktur angeboten, was z. B. Crossplane Compositions oder CDK Constructs sind. Diese Komponenten sind auf die eigene Organisation optimiert und enthalten die Best Practices für z. B. Security und Compliance genauso wie für die reduzierten CO₂-Emissionen oder Kostenkontrolle. Die Informationen zur Integration sind ebenfalls darin vorhanden, sodass auch dies ohne viele Meetings machbar ist.

Es ist ein erweiterbares Modell mit klaren Zuständigkeiten. Teil dieser Plattform sind in einer weiteren Ausbaustufe zusätzliche Elemente wie eine Pattern Library mit Frontend-Komponenten, die Best Practices aus den Bereichen der Accessibility und Green Software enthalten. Genauso können Service Tools und Librarys bereitgestellt werden. Die Weiterentwicklung und Pflege dieser Plattform ist inner-

halb des Vorgehens ebenfalls adressiert. Die Plattform und alle nutzenden Applikationen haben einen aktuellen Stand basierend auf dem in Code gegossenen verteilten Wissen innerhalb der gesamten Organisation.

Damit habe ich das Thema angerissen. Wer einen tieferen Eindruck erhalten möchte, findet diesen z. B. im Artikel von Manuel Zapf zum Thema (ab S. 6) in diesem Softwerker. Aber wie zählt dies auf Nachhaltigkeit ein? Beleuchten wir einmal die Aspekte:

Die ökonomische Nachhaltigkeit betrifft dies ab einer gewissen Größe des Entwicklungsteams. Durch dieses geschickte Vorgehen werden Synergien ab ca. fünf Teams schnell erreicht. Eingesparte Zeit durch klare Lösungsbausteine und Automatisierung kann in die Entwicklung von konkreten Fachlichkeiten fließen. In der Praxis reduzieren sich Onboarding-Aufwände massiv, was auf SDG 9 – „Industry, Innovation and Infrastructure“ [3] – einzahlt. Die technische Nachhaltigkeit wird durch eine einheitliche Plattform erhöht. Unterschiedliche Applikationen haben eine gemeinsame Basis, die optimiert und gepflegt wird. Für die Entwickler*innen wird damit der Fokus auf wirklich wertschöpfende Tätigkeiten frei gemacht und eine höhere Wertschöpfung erlaubt. Damit zählt das Plattform Engineering auf die SDG 8 – „Decent Work and Economic Growth“ und das bereits erwähnte SDG 9 ein.

Die ökologischen Best Practices zur Klimafreundlichkeit sind direkt in die Plattform eingebaut und müssen nicht jedes Mal nachgepflegt werden, sodass automatisch SDG 13 – „Climate Action“ – und SDG 14 – „Life Below Water“ – erfüllt sind. Eine Plattform bringt automatisch Vorgaben

mit, sodass z. B. Cluster und Services im Normalfall elastisch sind und auch eine Carbon Awareness für Workloads anbieten. Carbon Awareness ist ein Begriff, der durch die Green Software Foundation geprägt ist, und bezeichnet die Verschiebung von zeitunabhängigen Arbeitsblöcken in eine Zeit, in der so wenig CO₂ wie möglich verursacht wird. Dies muss nur einmal bereitgestellt werden und wird dann automatisch überall als Teil der Plattform genutzt.

Ein Marathon

Mit diesem Artikel lade ich zum Nachdenken und zum Austausch ein. Die Themen und Ideen habe ich bewusst skizziert, in der Hoffnung, dass wir sie schrittweise mit Leben füllen. Ich halte das Thema für eine Chance und denke, das Leitmotiv findet an sehr vielen Stellen Anknüpfungspunkte, die für alle Beteiligten positiv sind. In der Praxis gibt es sehr viele Möglichkeiten, die Softwareentwicklung qualitativ zu verbessern und eine ganzheitliche Perspektive einzunehmen. Abschließend der Hinweis von mir: Nehmt das Thema in Gespräche und Entscheidungen mit. Das Bekenntnis und die Strategie sind wichtig, doch nur, wenn es Teil eures Alltags wird, wird das Anliegen auch gelebt.

Marco Paga (marco.paga@codecentric.de) arbeitet seit mehr als 20 Jahren in wechselnden IT-Rollen. Er hat ein breites Wissen in der Softwareentwicklung und -architektur, immer mit Blick auf das große Ganze. Ihn beschäftigt das Thema Nachhaltigkeit/Green Software, um konkrete positive Schritte zu gehen.

Referenzen

- [1] <https://www.codecentric.de/ueber-uns/b-corporation>
- [2] [https://de.wikipedia.org/wiki/Drei-S%C3%A4ulen-Modell_\(Nachhaltigkeit\)](https://de.wikipedia.org/wiki/Drei-S%C3%A4ulen-Modell_(Nachhaltigkeit))
- [3] <https://www.un.org/sustainabledevelopment/>

Data Products, Data Contracts und API Contracts

Grundpfeiler moderner Datenarchitekturen
als Erfolgsfaktoren für Organisationen

von **Daniel Kocot**



Chauliodus sloani

RÄUBERISCHER MESOPELAGISCHER
DRACHENFISCH, DER IN NAHRUNGSREICHEN
GEWÄSSERN AUF DER GANZEN WELT
VORKOMMT

FRISST LATERNENFISCHE,
ANDERE KNOCHENFISCHE,
SCHALENTIERE

GRÖSSE
6-26 CM

ÖFFNET SEINEN KIEFER BIS ZU 90 GRAD UND
WEITET SEINEN MAGEN, UM BEUTE ZU
SCHLUCKEN

In der digitalen Wirtschaft sind eine durchdachte Datenarchitektur und die effiziente Nutzung von Daten entscheidend für den Organisationserfolg. Data Products, Data Contracts und API Contracts gewinnen an Bedeutung, da sie helfen, Daten sinnvoll zu nutzen, um innovative Produkte zu entwickeln und die betriebliche Effizienz zu steigern.

Data Products transformieren Rohdaten in wertvolle Erkenntnisse, während Data Contracts die Datenqualität und -nutzung regulieren. API Contracts sichern die Kommunikation zwischen Anwendungen und unterstützen die Skalierbarkeit der IT-Infrastrukturen. Dieser Artikel erläutert, wie diese Elemente zusammenwirken, um den Organisationserfolg zu fördern.

Data Products

Data Products oder Datenprodukte sind zentrale Bestandteile moderner Datenarchitekturen, die spezifische Geschäftsprobleme durch datengetriebene Dienste oder Lösungen adressieren. Im Kern eines jeden Data Products steht die Fähigkeit, Daten so zu nutzen, dass sie einen direkten Mehrwert für die Organisation schaffen, sei es durch die Optimierung interner Prozesse oder die Verbesserung der Kundeninteraktion. Der Wert eines Data Products liegt in seiner Spezifität und seiner direkten Ausrichtung auf messbare Geschäftsergebnisse.

Die Entwicklung eines Data Products erfordert eine sorgfältige Planung und ein tiefes Verständnis sowohl der technischen Möglichkeiten als auch der geschäftlichen

Anforderungen. Der Data Product Canvas [1] ist hierbei ein entscheidendes Werkzeug, das Teams dabei unterstützt, die Schlüsselemente eines Data Products zu definieren – von der Identifizierung der Zielbenutzer über die Datenerfassung bis hin zur finalen Wertgenerierung. Dieses Framework fördert einen systematischen Ansatz, um sicherzustellen, dass alle Aspekte des Data Products zur Unterstützung der Organisationsziele beitragen.

Die strategische Bedeutung von Data Products im Organisationskontext kann nicht hoch genug eingeschätzt werden, da sie nicht nur zur Effizienzsteigerung und Kostensenkung beitragen, sondern auch signifikante Wettbewerbsvorteile schaffen können. Um jedoch nachhaltig erfolgreich zu sein, muss die Entwicklung von Data Products innerhalb einer klar definierten Datenstrategie erfolgen. Es gilt, sowohl die technischen Anforderungen als auch die geschäftlichen Erwartungen genau zu balancieren und regelmäßig zu überprüfen, ob das Datenprodukt den sich ändernden Marktanforderungen noch gerecht wird. Die kontinuierliche Anpassung und Verbesserung des Produkts in Zusammenarbeit mit den Endnutzern ist dabei essenziell für den langfristigen Erfolg.

Data Contracts

Data Contracts oder Datenverträge sind essenziell für die erfolgreiche Implementierung und das Management von Data Products, wie im vorangehenden Abschnitt über diese dargelegt wurde. Sie definieren präzise die Qualitätsanforderungen, das Format und die Nutzungsbedingungen der Daten, die innerhalb einer Organisation oder zwischen

Geschäftspartnern ausgetauscht werden. Diese Verträge gewährleisten, dass Daten konsistent, zuverlässig und gemäß den festgelegten Spezifikationen gehandhabt werden.

Der Open Data Contract Standard [2] spielt eine zentrale Rolle in der Formulierung von Data Contracts, indem er eine standardisierte Vorlage bietet, die die Interoperabilität zwischen verschiedenen Datenquellen und -systemen erleichtert. Dieser Standard hilft Organisationen, ihre Datenverträge klar zu strukturieren und rechtliche sowie technische Missverständnisse zu minimieren.

Im Gegensatz dazu bietet die Open Data Product Specification [3] eine umfassendere Sichtweise, die nicht nur die vertraglichen Aspekte der Datennutzung abdeckt, sondern auch Richtlinien zur Gestaltung des gesamten Data Products einschließlich seiner technischen und funktionalen Merkmale. Während sich der Open-Data-Contract-Standard hauptsächlich auf die Definition der Datenaustauschparameter konzentriert, umfasst die Open Data Product Specification breitere Aspekte der Datenproduktentwicklung und -verwaltung.

Darüber hinaus ermöglicht der Data Product Canvas [1], der bereits im Kontext der Datenproduktentwicklung erwähnt wurde, eine detaillierte und nutzerzentrierte Planung von Datenprodukten. Er unterscheidet sich von den Data Contracts dadurch, dass er nicht nur die technischen und datenspezifischen Aspekte abdeckt, sondern auch die Identifizierung der Stakeholder, sowie die Nutzeranforderungen und die Wertschöpfungspfade aufzeigt. Data Contracts sind somit ein unverzichtbarer Bestandteil des Ökosystems der Datenprodukte, der sicherstellt, dass

alle Beteiligten ein gemeinsames Verständnis haben und die Daten in einer Weise genutzt werden, die allen beteiligten Parteien gerecht wird. Ihre Implementierung und fortlaufende Anpassung sind entscheidend für die Aufrechterhaltung der Datenintegrität und -sicherheit im gesamten Lebenszyklus eines Data Products.

API Contracts

API Contracts sind entscheidend für die Interaktion zwischen Softwareanwendungen. Sie definieren Anfragen, Strukturen und erwartete Antworten eines API und sind daher unverzichtbar für die Integration interner und externer Services. Diese Verträge garantieren Funktionalität, Sicherheit und Leistung der APIs, indem sie klare, maschinenlesbare Spezifikationen bieten und somit konsistente Funktionen und reibungslose Interaktion von Diensten sicherstellen. Ein gut definierter API Contract verhindert Missverständnisse und Fehler in der Softwareentwicklung. Der Entwurf von API Contracts erfordert Kenntnis der Best Practices, wie bspw. die Nutzung des OpenAPI-Standards, der klare Richtlinien für Endpunkte, Methoden und Parameter festlegt. Diese Standards erleichtern anderen Entwicklern die Integration und Nutzung der APIs. Sicherheitsrichtlinien in den Verträgen schützen die Datenübertragung und definieren Zugriffskontrollen, während Leistungsoptimierungen durch effiziente Spezifikationen Engpässe vermeiden.

Die Wartung und Versionierung von API Contracts ist essenziell, um auf Änderungen in Geschäftsanforderungen und Technologien reagieren zu können. Regelmäßige Updates

und eine klare Versionierungsstrategie sichern die Unterstützung älterer API-Versionen und schützen bestehende Nutzer vor Beeinträchtigungen durch Änderungen.

Zusammenfassend ist die sorgfältige Gestaltung und Pflege von API Contracts zentral für eine stabile und skalierbare IT-Infrastruktur, sichert die Kommunikation zwischen Systemkomponenten und unterstützt die digitale Transformation von Organisationen.

Integration und Synergien

Die effektive Integration von Data Products, Data Contracts und API Contracts ist entscheidend für den Erfolg einer modernen Datenstrategie. Durch die nahtlose Verbindung dieser drei Kernkonzepte können Organisationen eine kohärente Datenarchitektur schaffen, die es ermöglicht, Daten effizienter zu nutzen und einen echten Geschäftswert zu generieren.

Integration beginnt mit dem Verständnis, dass Data Products, Data Contracts und API Contracts zwar unabhängige Entitäten sind, jedoch in einem symbiotischen Verhältnis stehen. Data Products liefern datengetriebene Lösungen, die für den internen und externen Gebrauch durch Organisationen entwickelt werden. Als Endprodukte basieren sie auf Datenflüssen und -diensten, die durch Data Contracts definiert und durch API Contracts zugäng-



Abb. 1: Das Verhältnis von Data Product, Data Contract und API Contract

lich gemacht werden. Data Contracts sichern dabei die Qualität und das Format der Daten, während API Contracts die technischen Schnittstellen definieren, die für den Zugriff und die Integration dieser Daten in verschiedene Anwendungen und Systeme erforderlich sind.

Die Synergie zwischen diesen Elementen entsteht durch die Erstellung einer Datenpipeline, die reibungslos funktioniert und den Datenfluss von der Quelle bis zum Endprodukt nahtlos koordiniert. Dies erhöht nicht nur die Effizienz, sondern stellt auch sicher, dass alle datenbezogenen Aktivitäten den festgelegten Organisationsstandards und Compliance-Anforderungen entsprechen. Beispielsweise kann ein gut strukturierter Data Contract die Integrität der Daten sicherstellen, die durch APIs übertragen werden, was wiederum die Zuverlässigkeit der darauf aufbauenden Data Products erhöht.

Ein weiteres wichtiges Element der Integration ist die Fähigkeit, verschiedene Datensätze und Anwendungen zu verbinden, die in unterschiedlichen Bereichen der Organisation verwendet werden. Durch die Verwendung von API Contracts, die klare und konsistente Schnittstellendefinitionen bieten, können IT-Abteilungen verschiedene Datenquellen und Anwendungen nahtlos integrieren, ohne dass es zu Konflikten oder Dateninkonsistenzen kommt. Jedoch birgt die Integration von Data Products, Data Contracts und API Contracts auch Herausforderungen. Zu diesen gehört insbesondere die Komplexität der Datenverwaltung über mehrere Plattformen und Systeme hinweg. Viele Organisationen kämpfen mit fragmentierten Daten-

landschaften und isolierten Systemen, was eine konsistente Datenintegration erschwert.

Um diese Herausforderungen zu überwinden, müssen Organisationen häufig umfangreiche Anpassungen vornehmen und investieren in Technologien, die eine bessere Datenorchestrierung ermöglichen. Empfohlene Ansätze zur Überwindung von Integrationsbarrieren umfassen die Implementierung zentraler Datenmanagementplattformen, die Nutzung von Cloud-Diensten zur Förderung der Skalierbarkeit und Flexibilität sowie die Förderung einer Kultur der Datenkompetenz in der gesamten Organisation. Zudem ist eine enge Zusammenarbeit zwischen den IT-Teams und den Geschäftseinheiten erforderlich, um sicherzustellen, dass alle Technologien und Prozesse auf die Geschäftsziele abgestimmt sind.

Indem sie diese Strategien umsetzen, können Organisationen nicht nur die technischen Herausforderungen bewältigen, sondern auch die Synergien zwischen Data Products, Data Contracts und API Contracts voll ausschöpfen, um letztlich eine robuste, effiziente und zukunftssichere Datenarchitektur zu schaffen.

Fazit

In diesem Artikel haben wir die Bedeutung von Data Products, Data Contracts und API Contracts für die Optimierung der Datenarchitektur in Organisationen dargelegt. Ihre sorgfältige Integration steigert nicht nur Effizienz und Datenqualität, sondern trägt auch zur Sicherheit und Einhaltung von Compliance bei. Diese Elemente sind zentral, um interne Prozesse zu optimieren und neue Marktchan-

cen zu erschließen. Ein tiefgehendes Verständnis dieser Konzepte ist entscheidend, um technische, organisatorische und strategische Herausforderungen effektiv zu managen. Mit den Fortschritten in Technologien wie KI und maschinellem Lernen ist die Fähigkeit zur schnellen Anpassung unerlässlich, um die Wettbewerbsfähigkeit und Innovationskraft zu stärken.

Daniel Kocot ist seit Oktober 2016 ein Mitglied des codecentric-Teams und hat seit Anfang 2022 die Rolle des Senior Solution Architect am Standort Dortmund inne. Anfänglich fokussiert auf das Application Lifecycle Management als Consultant, hat er seine Expertise schrittweise auf APIs ausgeweitet. Neben seiner aktiven Beteiligung an diversen Kundenprojekten und seiner Hingabe zur Open-Source-Welt in Bezug auf APIs hat er sich auch in seiner Position als Head of API Consulting als ein häufig angefragter Sprecher etabliert.

daniel.kocot@codecentric.de

Referenzen

- [1] <https://www.codecentric.de/wissens-hub/blog/mit-applied-data-products-zum-datengetriebenen-unternehmen>
- [2] <https://bitol.io>
- [3] <https://opendataproducs.org/v2.1/#open-data-product-specification-2-1>

Der IT-Podcast von Techies für Techies



Mit

Marco Paga

& seinen kompetenten Gästen

Neue Themen u. a.:

- Digitale Produktentwicklung
- Green Serverless Batch Jobs
- KI in der Softwareentwicklung



Alle Folgen unter
codecentric.de/softwarekercast
oder auf Spotify, Apple Podcasts etc.



Von der Domain Story zum Interface

Elegante APIs entstehen, wenn Experten
auf der technischen und der geschäftlichen
Seite zusammenarbeiten

von **Annegret Junker**

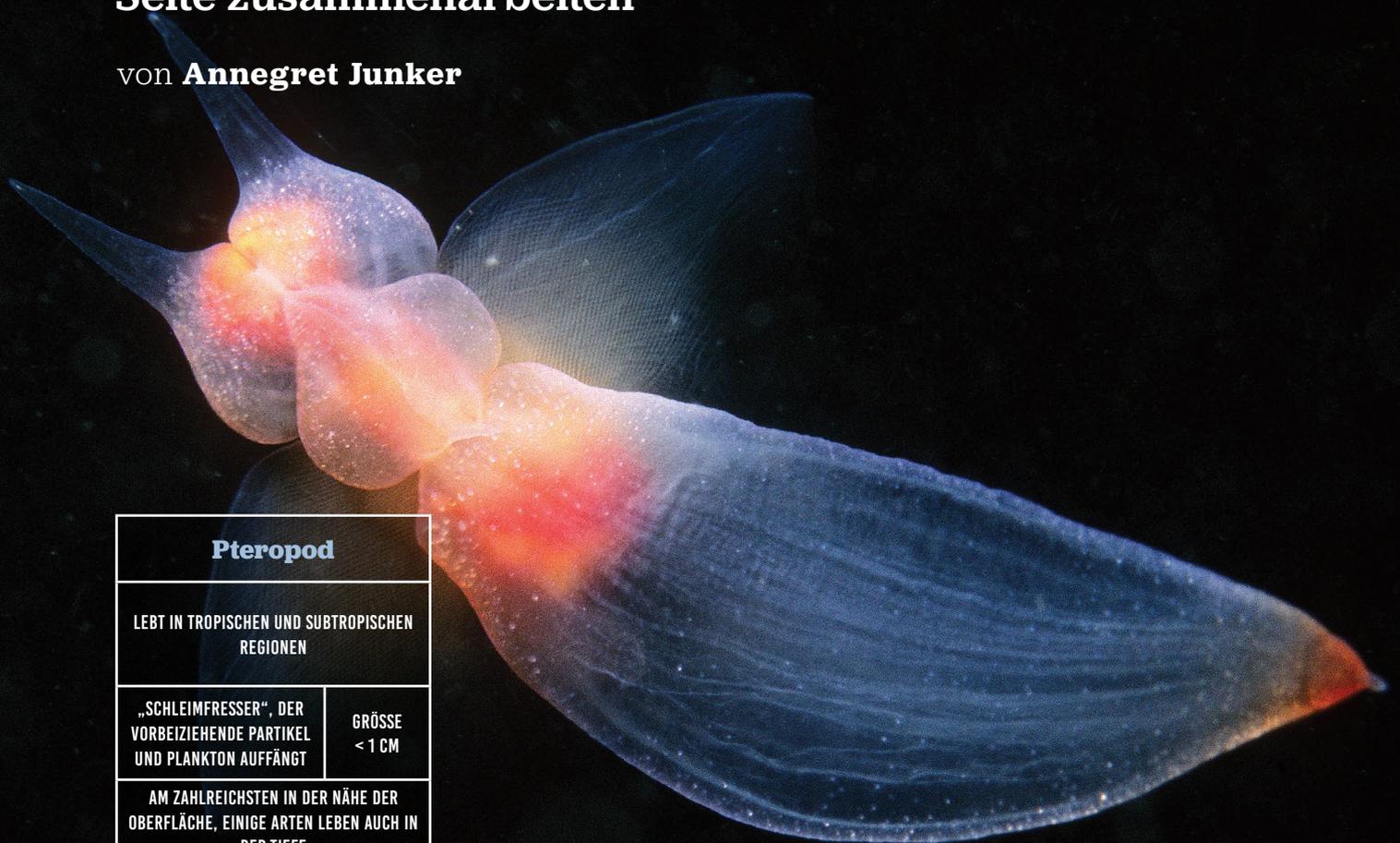
Pteropod

LEBT IN TROPISCHEN UND SUBTROPISCHEN
REGIONEN

„SCHLEIMFRESSER“, DER
VORBEIZIEHENDE PARTIKEL
UND PLANKTON AUFFÄNGT

GRÖSSE
< 1 CM

AM ZAHLREICHSTEN IN DER NÄHE DER
OBERFLÄCHE, EINIGE ARTEN LEBEN AUCH IN
DER TIEFE



Wie kann man aus Business- und technischer Sicht kollaborativ asynchrone und synchrone APIs definieren? Der folgende Artikel beschreibt einen kollaborativen Entwicklungsprozess, in dem Business- und IT-Experten gemeinsam Anforderungen aufnehmen, modellieren und APIs formulieren.

Schnittstellendesign ist eine der größten Herausforderungen in modernen Softwarearchitekturen. Schnittstellen müssen nicht nur technisch funktionieren, sondern vielmehr gut beschrieben sein. Zudem muss die Beschreibung sowohl für Techniker als auch für Business-Experten verständlich sein. Ein solches Design kann kollaborativ in gemeinsamen Workshops zwischen Business-Experten und IT-Spezialisten entwickelt werden.

Die erste Idee

Erste Ideen werden häufig als kurze Skizzen ohne Formalisierung gezeichnet und beschrieben. Ein Beispiel ist in **Abb. 1** (S. 40) dargestellt. Es ist die Idee, eine Verwaltung für eine Privat-Bibliothek zu schaffen.

Die Bibliothek enthält eine Übersicht der vorhandenen Bücher und in welchem Regal sie stehen. Die Übersicht enthält hierbei auch einen Lesestatus. Aus diesen ersten Ideen können die Stakeholder eine Skizze mit groben Geschäftsfähigkeiten erstellen.

Die in **Abb. 2** dargestellten Geschäftsfähigkeiten verfeinern die groben Skizzen des Anfangs. Folgende Fähigkeiten sollen durch das zu entwickelnde System möglich sein:

- Bücher kaufen
- Bücher einsortieren
- Bücher suchen
- Standort verwalten
- Bücher lesen
- Bücher ausleihen

In diesem ersten Schritt werden auch schon Vorüberlegungen diskutiert, die die zu übergebenden Daten enthalten, die ausgetauscht werden müssen. Diese werden durch die Pfeile zwischen den Ellipsen angedeutet. So muss zwischen den Geschäftsfähigkeiten „Bücher kaufen“ und „Bücher einsortieren“ das Buch als Datum übergeben werden.

Auf dieser Basis können die Experten von der geschäftlichen Seite und der technischen Seite einen gemeinsamen Workshop mit Domain Storytelling durchführen [1].

Domain Storytelling

Domain Storytelling ist eine Workshop-Methode. Business-Experten erzählen eine Geschichte, die grafisch durch einen Moderator aufgenommen wird. Die Geschichte wird Satz für Satz aufgenommen. Dabei enthält ein Satz einen Akteur, eine Aktion, die der Akteur ausführt, und ein Objekt, auf das die Aktion ausgeführt wird. Die Sätze werden mit Zahlen versehen, um die Sequenz der Geschichte zu verdeutlichen. **Abb. 3** zeigt eine Domain Story, die für eine Privatbibliothek entwickelt wurde. Die Geschichte enthält folgende Sätze:

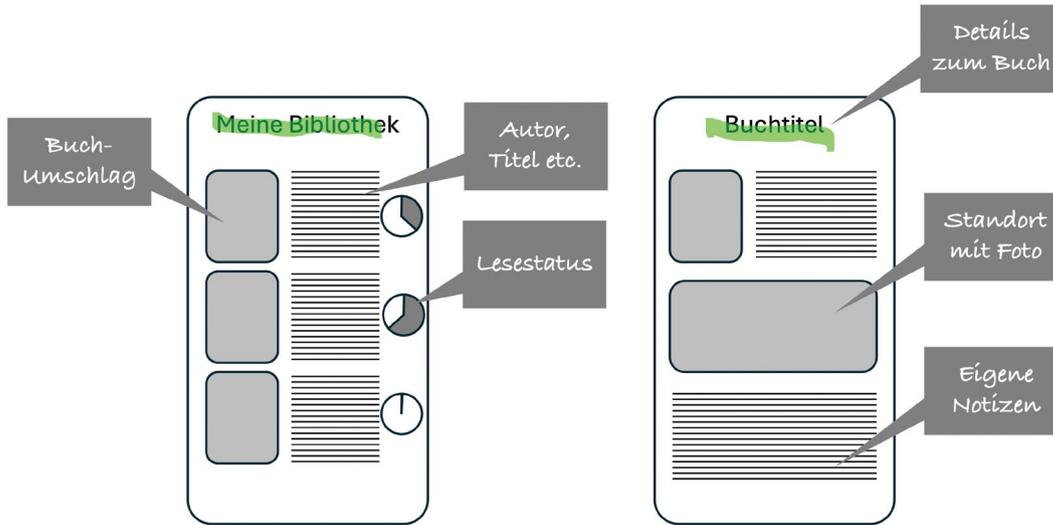


Abb. 1: Skizze einer Verwaltung einer Privatbibliothek

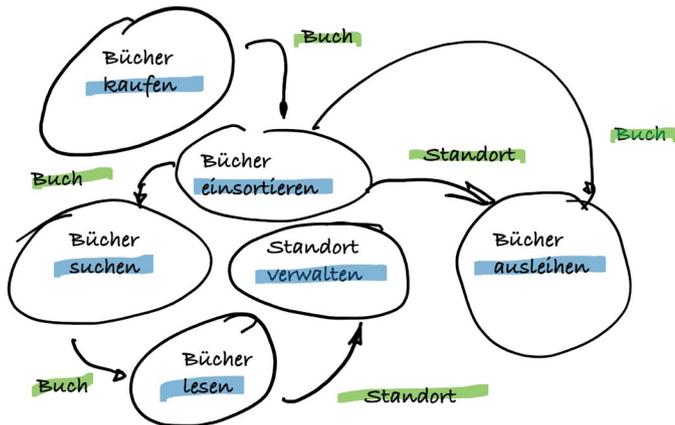


Abb. 2: Geschäftsfähigkeiten einer Privatbibliothek

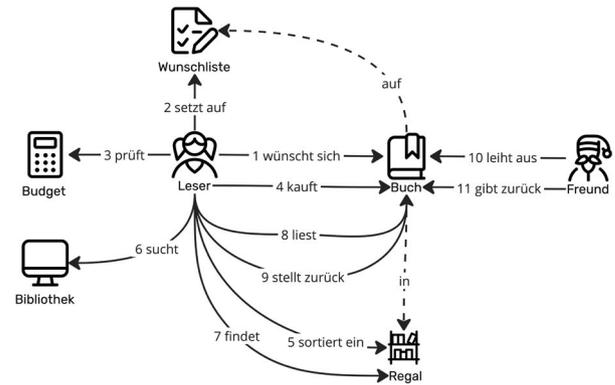


Abb. 3: Domain Story einer Privatbibliothek

1. Der Leser wünscht sich ein Buch.
2. Der Leser setzt das Buch auf die Wunschliste.
3. Der Leser prüft den Preis des Buches gegen sein Budget.
4. Der Leser kauft das Buch.
5. Der Leser sortiert das Buch in das Regal ein.
6. Der Leser sucht das Buch in der Bibliothek.
7. Der Leser findet das Buch im Regal.
8. Der Leser liest das Buch.
9. Der Leser stellt das Buch in das Regal zurück.
10. Der Freund borgt das Buch.
11. Der Freund gibt das Buch zurück.

Die Geschichte definiert schon die Begriffe „Buch“, „Wunschliste“, „Budget“ und „Regal“. Die Akteure „Leser“ und „Freund“ agieren auf diesen Objekten. Die statischen Zusammenhänge zwischen den Begriffen lassen sich gut über ein visuelles Glossar darstellen [2]. Ein visuelles Glossar enthält die Begriffe und erläutert sie durch die Beziehungen zueinander. **Abb. 4** zeigt ein solches visuelles Glossar für die Privatbibliothek.

Mit einem visuellen Glossar wird nicht versucht, für jeden Begriff eine allgemeingültige Definition zu finden. Vielmehr sind die Begriffe und ihre Zusammenhänge im jeweiligen Kontext gültig. Mit der Methodik des visuellen Glossars, das schon im Domain-Storytelling-Workshop erstellt wird, lässt sich zu einem sehr frühen Zeitpunkt eine *Ubiquitous Language* entwickeln, die für den jeweiligen Kontext gültig ist.

Im Domain Storytelling wird auch schon Status des Buches mit *gewünscht*, *gekauft*, *ausgeliehen* angedeutet. Auch diese Status können in einem einfachen Statusdiagramm im Workshop festgehalten und diskutiert werden. **Abb. 5** zeigt ein solches vereinfachtes Statusdiagramm.

Auf dieser Basis können Business-Experten und IT-Spezialisten diskutieren, wo im System die Bounded Contexts liegen und wo welche Daten übergeben werden müssen. Dies kann auch in einem Workshop mit der Verwendung der Workshop-Methode Event Storming erfolgen [3].

Event Storming

Event Storming wurde entwickelt, um anhand von Domain Events Bounded Contexts und zugehörige Aggregate zu definieren. Im ersten Schritt sammeln die Workshop-Teilnehmer Events. Im zweiten Schritt werden diese konsolidiert und in eine logische Reihenfolge gebracht. Die Diskussion enthält die Bewertung, wo in der Reihenfolge die Events einzusortieren sind oder ob das betrachtete Event schon in der Reihenfolge vorhanden ist und gelöscht werden kann. Gleichzeitig kann die Benennung von Events bewertet werden und die Workshop-Teilnehmer können sich für die bessere Bezeichnung entscheiden. Das entsprechende Ergebnis zeigt **Abb. 6**. Events sind hierbei in orange dargestellt. Um die einzelnen Karten besser unterscheiden zu können, sind die Events mit einem Feuerwerk-Icon gekennzeichnet.

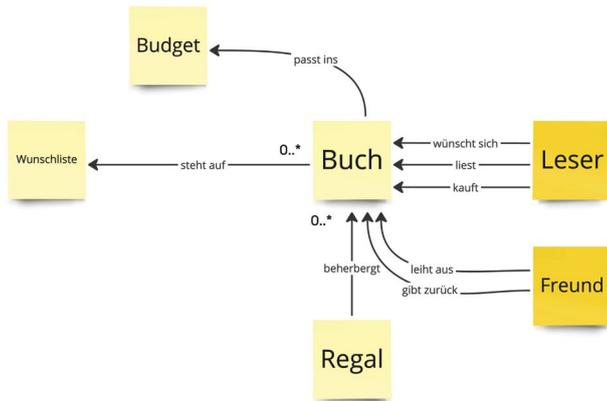


Abb. 4: Visuelles Glossar für die Privatbibliothek



Abb. 5: Vereinfachtes Statusdiagramm eines Buches



Abb. 6: Domänen-Events entlang des Prozesses einer Privatbibliothek

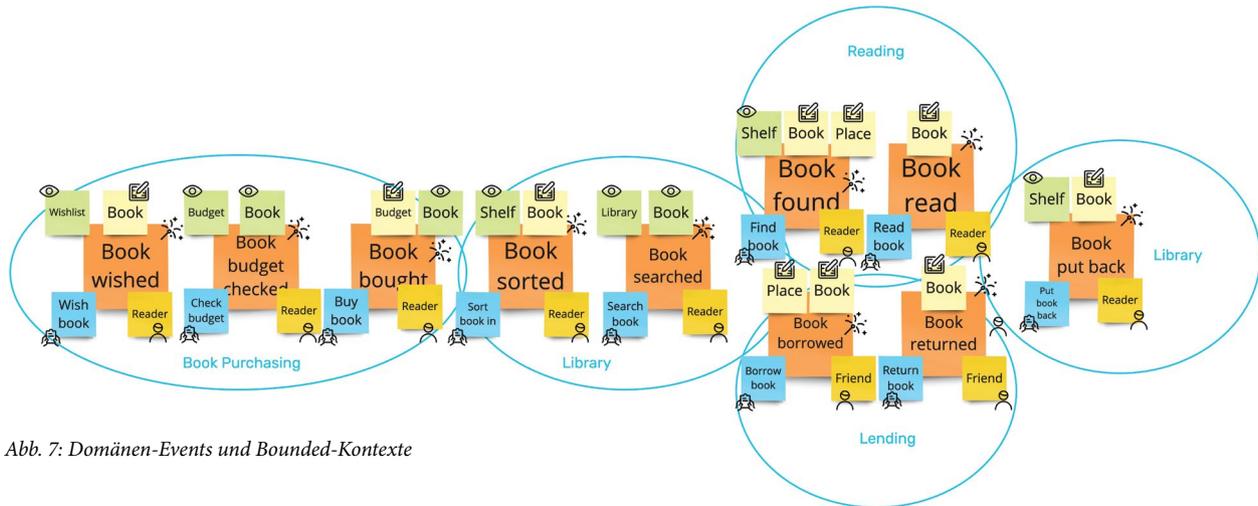


Abb. 7: Domänen-Events und Bounded-Kontexte

Die Events „Book found“, „Book read“, „Book borrowed“ und „Book returned“ sind parallele Zweige im Prozess, die einander ausschließen. Parallele Zweige, unabhängig davon, ob sie einander ausschließen oder nicht, können untereinander dargestellt werden. Das Event „Book put back“ führt beide Zweige wieder zusammen.

Im nächsten Schritt eines Event-Storming-Workshops können die einzelnen Events mit Kommandos und Rollen erweitert werden. Kommandos sind hierbei die Trigger, die das jeweilige Event herbeiführen. Rollen sind menschliche Benutzer, die das jeweilige Kommando erteilen. Auch automatische Prozesse können in solchen Event Stormings auftreten. Im vorliegenden Beispiel tritt allerdings ein solches Event nicht auf. In Workshops werden Rollen mit dunkelgelben Karten gekennzeichnet. Hier sind die Karten mit einem User-Icon gekennzeichnet. Kommandos sind in den Workshops blau. In unserem Beispiel sind sie mit einem Auftragsbuch-Icon dargestellt. Das Ergebnis der Kommando-Rollen-Zuweisung zu Events ist in **Abb. 7** dargestellt.

Im anschließenden Schritt können die Events durch die Aggregate erweitert werden, die durch sie verändert wurden. Zusätzlich kann es sein, dass Daten benötigt werden, um das Event zu erzeugen, die aber nicht verändert werden. Solche Daten werden *Views* genannt und die Events werden durch sie ebenfalls erweitert. Das Ergebnis dieser Diskussion ist **Abb. 7** dargestellt.

Normalerweise werden die Aggregate mit hellgelben und die Views mit hellgrünen Karten gekennzeichnet. Hier sind die Views zusätzlich mit einem Auge und die Aggregate mit einem Notizbuch und Stift gekennzeichnet.

Diese Darstellung hilft, die Bounded-Kontexte zu erkennen. Man sucht von Event zu Event nach einer Änderung des Aggregates oder einer Änderung der Rolle. Eine solche Änderung ist ein Hinweis für die Grenze von Bounded-Kontexten. Das Ergebnis dieser Überlegungen zeigt **Abb. 7**.

Ubiquitous Language

„Übergreifende Sprache“ – eine gemeinsame Sprache für Business-Experten und IT-Spezialisten, die innerhalb eines Bounded Kontext gültig ist.

Bounded-Kontext

Abgeschlossene Funktion oder Service, in dem ein Modell gültig ist. Idealerweise wird es durch einen Service und ein Team implementiert.

Capability oder Geschäftsfähigkeit

Fähigkeit eines Unternehmens, seine Tätigkeit auszuführen. Geschäftsfähigkeiten können Kerndomänen, unterstützenden Domänen oder generischen Domänen zugeordnet werden.

Context Map

„Karte“ der Bounded-Kontexte mit Abhängigkeiten und notwendigem Datenaustausch.

Folgende Kontexte wurden gefunden:

- **Book Purchasing / Bücher kaufen**
Die Grenze dieses Bounded-Kontextes ist die Grenze zwischen „Budget checked“ und „Book bought“, da hier der Übergang in die eigene Bibliothek stattfindet.
- **Library / Bibliothek**
Die nächste Grenze ist zwischen „Book searched“ und „Book found“ bzw. „Book borrowed“, da sich zwischen diesen Events der Standort ändert. Es wurde diskutiert, ob es sich beim Standort um einen separaten Bounded-Kontext handelt. Es wurde sich aber dagegen entschieden, da der Standort von den Kontexten „Library / Bibliothek“ und „Reading / Lesen“ benutzt wird.
- **Reading / Lesen**
Dieser Kontext grenzt sich gegen Bibliothek sowohl am Eingang über „Book found“ als auch am Ende über „Book put back ab“. „Standort / Place“ als Eigenschaft von „Book / Buch“ ändert sich in seiner Aussage. In der Bibliothek ist es der Platz im Regal, beim Lesen ist es der Platz im Wohnzimmer oder am Bett, im Koffer etc.
- **Lending / Ausleihe**
Dieser Kontext grenzt sich wie auch „Reading / Lesen“ über „Book found“ und „Book put back“ von der Bibliothek ab. Auch hier hat „Standort / Platz“ eine andere Bedeutung und umfasst vielmehr den Ausleiher.

Sind die Bounded-Kontexte gefunden, kann die Darstellung in eine Context Map überführt werden.

Context Map

Eine Context Map wird benutzt, um die Abhängigkeiten zwischen Bounded-Kontexten und eventuellem Datenaustausch zu zeigen [4]. Sie kann auf der Basis eines Event Stormings sehr einfach erzeugt werden, indem man für jeden gefundenen Bounded-Kontext ein Rechteck zeichnet. Diese Rechtecke enthalten die gefundenen Events und Aggregate.

Nun kann man anhand der Übergänge diskutieren, welche Daten zwischen den Bounded-Kontexten übertragen werden müssen. Die zu übertragenden Daten werden in Datenflussrichtung mit Pfeilen angedeutet. Dabei hat es sich bewährt, für synchronen Datenaustausch durchgezogene und für asynchronen Datenaustausch gestrichelte Linien zu verwenden. Die Pfeile werden beim synchronen Datenaustausch mit dem entsprechenden Aggregat und bei einem asynchronen Datenaustausch mit dem entsprechenden Event und mit dem zu übertragenden Aggregat bezeichnet. Zusätzlich können die Workshopteilnehmer die synchronen Schnittstellen diskutieren, die für eine Darstellung in einer Benutzerschnittstelle z. B. einer mobilen Anwendung benötigt werden.

Das Ergebnis einer solchen Diskussion für das Beispiel Privatbibliothek ist in **Abb. 8** dargestellt.

Während der Erstellung der Context Map können die Teilnehmer des Workshops auch das visuelle Glossar verfeinern und die Strukturen für den jeweiligen Bounded-Kontext definieren. Diese Diskussion muss zwingend zusammen mit den Business-Experten erfolgen, da das

visuelle Glossar die Dokumentation für die Ubiquitous Language – die gemeinsame Sprache – darstellt. Die Verfeinerung des visuellen Glossars mit Gültigkeitsbereichen für die Bounded-Kontexte zeigt **Abb. 9**.

Erstellung der APIs

Aus den Darstellungen des verfeinerten visuellen Glossars und der Context Map können die APIs formuliert werden. Um eine erste Version der APIs zu erzeugen, können die Bilder für Bounded-Kontext an eine generative Intelligenz

übergeben werden. Zusätzlich empfiehlt es sich, dem Automaten ein Skeleton, ein Gerippe der zu erzeugenden API-Definition, zu übergeben. Die Autorin stellt solche Skeletons für OpenAPI für REST-APIs [5] und AsyncAPIs für Events [6] in GitHub zur Verfügung.

Diese erste Version muss um gute Beschreibungen und Beispiele erweitert werden. Security-Anforderungen müssen ergänzt werden. Aber die Eleganz und Verständlichkeit ist durch die Verwendung der Ubiquitous Language und der definierten Bounded-Kontexte sichergestellt.

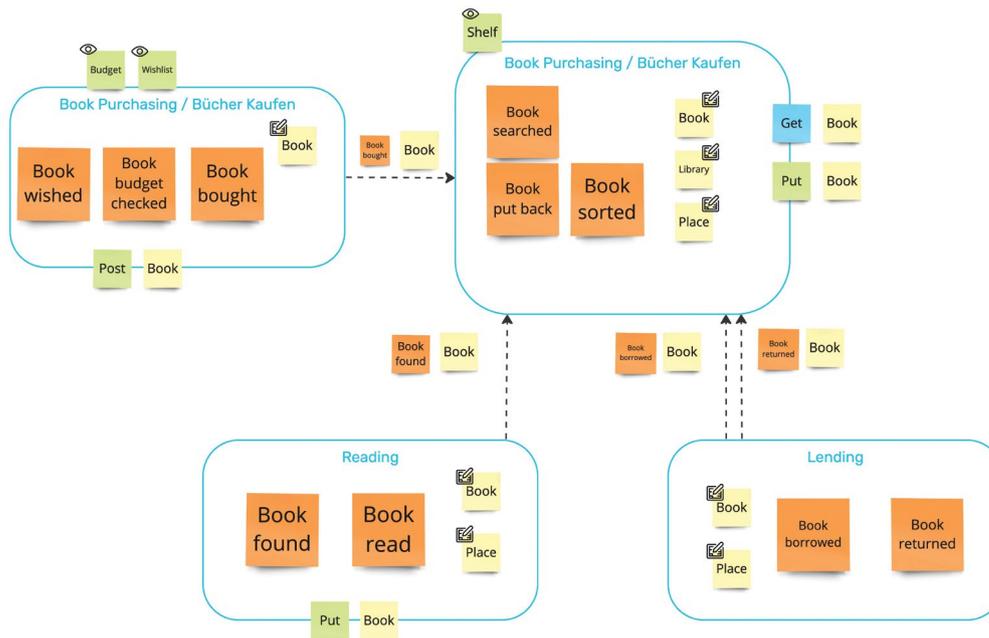


Abb. 8: Context Map der Privatbibliothek

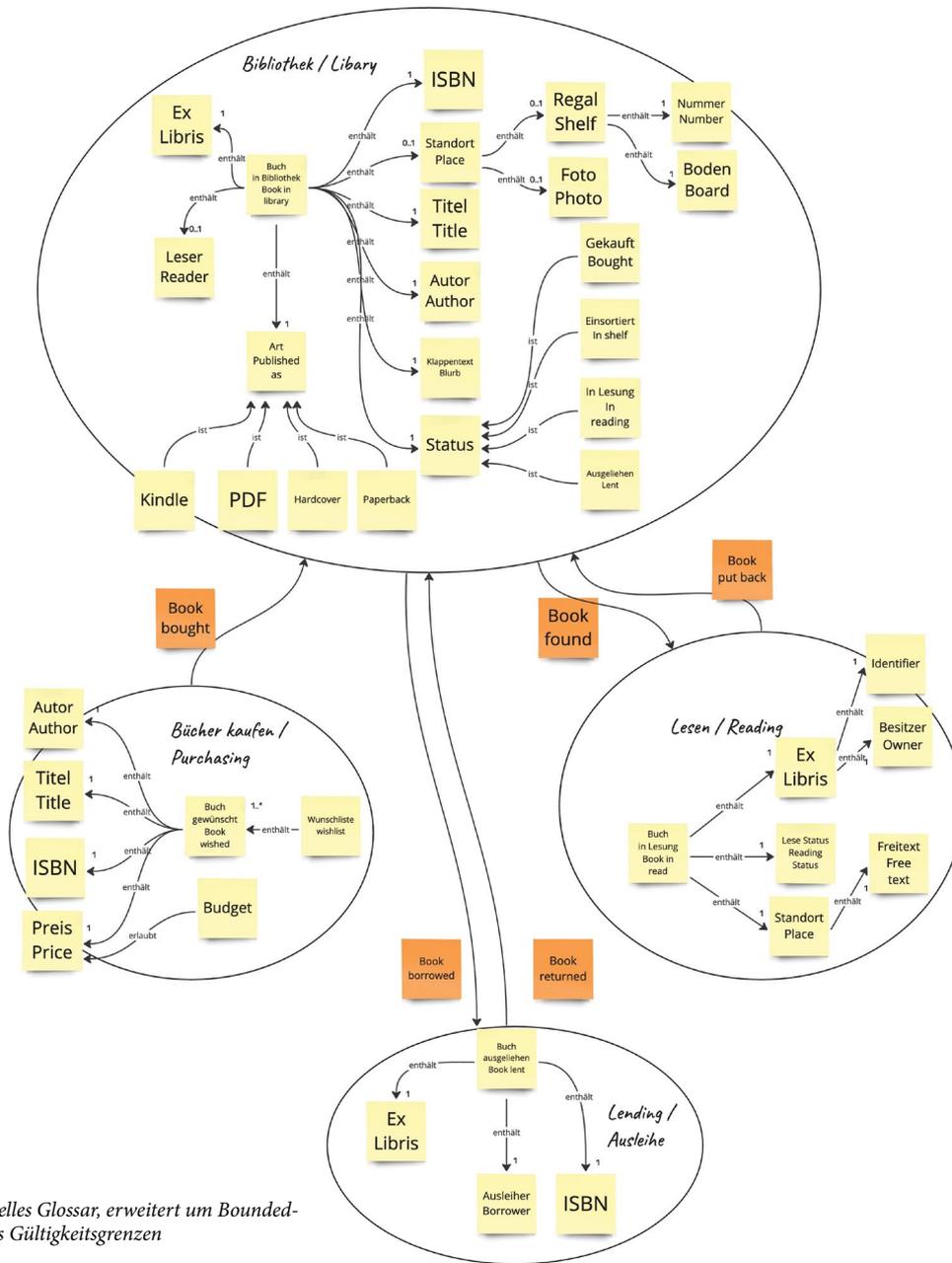


Abb. 9 Visuelles Glossar, erweitert um Bounded-Kontexte als Gültigkeitsgrenzen

Sprache

In diesem Artikel werden deutsche und englische Begriffe gemischt verwendet. Diese Vermischung wird absichtlich so gezeigt, da man in reinen deutschsprachigen Projekten die beschriebenen Workshop-Formate Domain Storytelling und Event Storming in deutscher Sprache durchführen kann. Allerdings sagt die Erfahrung der Autorin, dass in späteren Projektphase häufig nicht-deutschsprachige Projektmitglieder hinzukommen. Dann sind Dokumentationen in Englisch zwingend. Mehr noch: APIs als Fenster zur (Entwickler-)Welt müssen in Englisch sein, da man APIs als Produkt zu einer breiteren Verwendung zur Verfügung stellen möchte als ein rein deutschsprachige Verwendung.

Ausblick und Fazit

Der Artikel stellt eine durchgehende Vorgehensweise vor, wie Business-Experten und IT-Spezialisten gemeinsam elegante und verständliche APIs für Bounded-Kontexte entwerfen können. Dabei stehen den Experten Workshop-Formate wie Domain Storytelling und Event Storming sowie Dokumentations-Formate wie ein visuelles Glossar zur Verfügung. Die Verwendung der erarbeiteten Ubiquitous Language stellt die Verständlichkeit und Angemessenheit des API sicher.

Dr. Annegret Junker ist Chief Software Architect bei der codecentric AG. Sie arbeitet seit mehr als 30 Jahren in der Softwareentwicklung in unterschiedlichen Rollen und unterschiedlichen Domänen wie Automotive, Versicherungen und Finanzdienstleistungen. Besonders interessiert sie sich für DDD, Microservices und alles, was damit zusammenhängt. Derzeit arbeitet sie in einem großen Integrationsprojekt und definiert APIs für Partner.

annegret.junker@codecentric.de

Referenzen

- [1] S. Hofer and Schwentner, Henning, Domain Storytelling: A Collaborative, Visual, and Agile Way to Build Domain-Driven Software, Pearson International, 2021.
- [2] S. Zörner, Softwarearchitekturen dokumentieren und kommunizieren, Entwürfe, Entscheidungen und Lösungen nachvollziehbar und wirkungsvoll festhalten, München: Carl Hanser Verlag, 2015.
- [3] A. Brandolini, „EventStorming“, 2023: <https://www.eventstorming.com/>. [Accessed 04 2024].
- [4] E. Evans, Domain-Driven Design: Tackling Complexity in the Heart of Software, Boston: Addison-Wesley, 2003.
- [5] A. Junker, „REST API Skeleton“, 3. März 2024: <https://github.com/Grinsetddy/DomainDrivenApiDesign/blob/main/OnlineLibrary/RestSkeleton.yaml>
- [6] A. Junker, „AsyncAPI“, 10. März 2024: <https://github.com/Grinsetddy/DomainDrivenApiDesign/blob/main/OnlineLibrary/AsyncSkeleton.yaml>

Das Product Operating Model

Die Zauberformel, um neue Technologien
zum Markterfolg zu machen

von **Marc Pudelski** und **Ralf Westbrook**

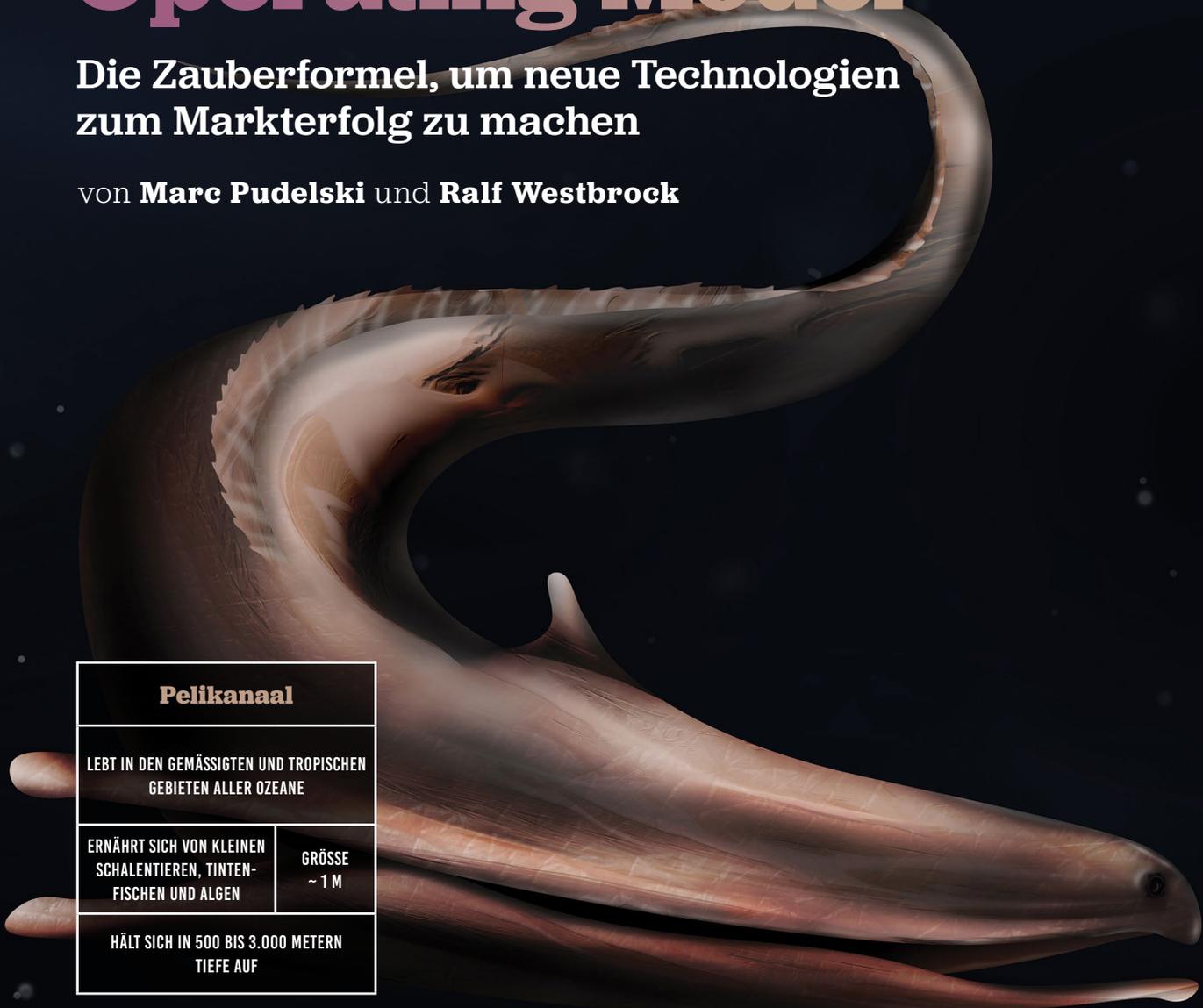
Pelikanaal

LEBT IN DEN GEMÄSSIGTEN UND TROPISCHEN
GEBIETEN ALLER OZEANE

ERNÄHRT SICH VON KLEINEN
SCHALENTIEREN, TINTEN-
FISCHEN UND ALGEN

GRÖSSE
~ 1 M

HÄLT SICH IN 500 BIS 3.000 METERN
TIEFE AUF



Habt ihr es mitbekommen? Ein Raumfahrtunternehmen konnte dank neuer Software, KI und Sensorik die ersten Raketenstufen autonom unbeschädigt landen und wiederverwenden lassen, wodurch es große Kostenvorteile erzielte. Ein weiteres Unternehmen transportiert Passagiere in fahrerlosen Autos und ist nun schon in vier Großstädten mit überdurchschnittlicher Sicherheit im Einsatz. Eine neue Softwareplattform konnte innerhalb von zwei Monaten 100 Millionen Nutzer akquirieren.

Innovationserfolge wie diese erregen große Aufmerksamkeit. Tech-Unternehmen überraschen ihre Branchen oft, wie sie technische Neuerungen einsetzen und über die geschaffene Lösung sowohl erfolgreich etablierte als auch bisher unberücksichtigte Kundengruppen gewinnen. Die ständige Branchendisruption führt viele Unternehmen zu der Frage: „Wie können wir es aus eigener Kraft schaffen, im Markt mittels eigener Investitionen in Technologie im Wettbewerb weiter mitzuhalten und erfolgreich unsere Zukunft zu gestalten?“ Die aktuell rasante Veränderungsgeschwindigkeit durch die Nutzung generativer KI schafft erneut Veränderungsdruck und -bereitschaft.

Marty Cagan und die Silicon Valley Product Group bieten mit ihrem Product Operating Model eine Lösung. Sie beschreiben bereits seit einigen Jahren die Arbeitsweisen, wie technologiegetriebene Firmen nachhaltig erfolgreiche Geschäftsmodelle entwickeln und beraten Unternehmen in der Anwendung. Mit seinem neu erschienenen Buch *Transformed* hat Marty zum ersten Mal das gesamte

Arbeitsmodell dokumentiert und Einblicke in die Transformation gegeben.

In unserem Artikel beschreiben wir das Modell, erläutern seine Merkmale, diskutieren typische Herausforderungen und skizzieren mögliche erste Schritte.

Was ist das Product Operating Model?

Das Product Operating Model, auch als Product Model bekannt, bietet Organisationen Konzepte und Prinzipien, wie sie ein tiefes Verständnis für Kunden und Märkte entwickeln können und dies mit den Potenzialen technologischer Entwicklungen zu nachhaltigem Geschäftserfolg verbinden können. Dabei beschreibt das Modell nicht nur Konzepte der Arbeitsweise, sondern benennt auch Prinzipien zu Organisation, Zusammenspiel und Kultur des gesamten Unternehmens.

Ein Beispiel für ein alternatives Modell zur Abgrenzung wäre das sogenannte „IT Model“, das uns in in vielen Organisationen begegnet. Das Modell leidet darunter, dass IT und Business nicht holistisch gedacht werden. Die IT erhält meist Lösungswünsche und lernt die dahinterliegenden Probleme nicht in der Tiefe kennen. Teams sind operativ erst verbunden, wenn es an die Umsetzung geht. Dadurch können sich technologische Neuerungen und Einblicke zu Marktchancen nicht gegenseitig befruchten. Ein typisches Symptom besteht darin, dass durch die strikte Fokussierung auf die Lieferung fester Outputs zu bestimmten Deadlines die Flexibilität und Anpassungsfähigkeit an Marktveränderungen und unvorhergesehene

Herausforderungen eingeschränkt wird, während gleichzeitig Verantwortung für den Markterfolg und kontinuierliche Innovation verloren gehen.

Es ist wichtig zu verstehen, dass das Product Operating Model keinen neu erfundenen Prozess oder ein Prozessframework darstellt, das mit einem Anwendungs- und Migrationsvorgehen verkauft wird. Vielmehr beschreibt es einen Zielzustand, der über Konzepte und Prinzipien beschrieben ist, die uns Orientierung geben. Ein Beispiel ist das Prinzip, dass die Notwendigkeit zum Probieren mehrerer Lösungsideen gegenüber einer Vorhersagbarkeit der Lösung wichtiger ist.

Für diese Konzepte und Prinzipien gibt es zahlreiche Möglichkeiten zur Abbildung auf ein Unternehmen, seine Mitarbeitenden und Geschäftstätigkeiten. Besonders span-



Abb. 1: Product Model Concepts [2]

nend ist es dabei zu bedenken, dass diese Konzepte und Prinzipien wiederum in vielen der erfolgreichsten Tech-Unternehmen der Welt wiederzufinden sind und von deren Leadership meist selbst als erfolgskritisch erachtet werden.

Das Product Operating Model gliedert sich in fünf Konzepte, die jeweils mit mehreren Prinzipien verbunden werden (Abb. 1). Im Folgenden stellen wir die Konzepte kurz vor und geben einen Überblick über die aus unserer Erfahrung wichtigsten Punkte.

Teams mit hoher Mitverantwortung

Ein fundamentales Konzept des Product Models ist die Zusammenarbeit in cross-funktionalen Produktteams mit Ergebnisverantwortung für Produkte oder Teile davon. Die Teams unter der inhaltlichen Führung von Product Manager, Designer und Tech Lead werden mit den Kompetenzen und Handlungsfreiheiten ausgestattet, um in direkter Partnerschaft mit den Kunden zu arbeiten und gleichzeitig auf Augenhöhe mit den anderen Unternehmensbereichen agieren zu können.

In der Kundeninteraktion stellen die cross-funktionalen Teams im Product Model sicher, dass sie den Kontext und die Aufgabenstellungen der Nutzer und Kunden verstehen. Dann verproben sie Lösungsideen, um den Wert der Produkte für Kunden und Unternehmen weiter zu erhöhen. Gleichzeitig stellen sie in Zusammenarbeit mit den internen Bereichen sicher, dass das Produkt mit der Organisation und ihren Wertschöpfungsprozessen abgestimmt bleibt. Die Aufgabenstellungen und der Erfolg der Teams

werden dabei an den Messgrößen der durch die Strategie gesetzten Ziele (Outcomes) und nicht an der fristgerechten Lieferung festgelegter Funktionen (Outputs) gemessen.

Strategische Entwicklung von Produkten

Produktstrategie nimmt eine zentrale Rolle ein und befasst sich damit, die kritischsten Probleme, Chancen und Bedrohungen des Unternehmens zu identifizieren, zu fokussieren und passende Handlungen umzusetzen. Ein zentrales Prinzip ist dabei, dass die Produktstrategie von Erkenntnissen über den Markt getrieben wird. Dabei ist besonders wichtig, wie sich die möglichen Zielsegmente, Technologien, Wettbewerber, Branchen und regulatorischen Rahmenbedingungen verändern. Aus der Strategiearbeit kennt man hier z. B. *Porter's 5 Forces* oder im modernen Remix auch das Business Model Environment von Strategyzer. Die Quellen für erkenntnisgetriebene Strategiearbeit sind vor allem Kundeninterviews und Nutzungsdaten, ergänzt um Wettbewerbsanalysen und Trendforschung. Verankert wird die Strategie durch die Produktvision, die als zukünftiges Zielbild fungiert, das allen beteiligten Produktteams einen gemeinsamen Rahmen gibt und das Zusammenspiel reguliert.

Cagan grenzt das Modell ab gegenüber einer Organisation, die hauptsächlich reaktiv bzgl. Anfragen einzelner Kunden oder den Druck interner Bereiche handelt. Diese Organisationen riskieren extreme Entwicklungskosten oder verlieren schlichtweg den Anschluss an Wettbewerber, die

stärker auf Wertschöpfung für Kunden und Innovation in Verbindung mit schlagkräftigen Geschäftsmodellen ausgerichtet sind.

Discovery

Einer der wichtigsten Paradigmenwechsel in der digitalen Produktentwicklung ist das Konzept der Product Discovery. Es spiegelt die Einsicht wider, dass ein Großteil der Ideen für die Produktentwicklung bei Kunden weniger erfolgreich ist als zuvor erhofft. In der Discovery erarbeitet das Team in engem Kontakt mit Kunden und durch Einsatz unterschiedlicher Verfahren wie Prototypen in schnellen Zyklen eine für Kunden und das eigene Unternehmen gewinnbringende Lösung. Es besteht eine wechselseitige Beziehung aus Strategie und Discovery. Die Aufgabenstellungen der Discovery entstehen aus der Strategie, und die gewonnenen Erkenntnisse aus der Discovery beeinflussen die Strategie.

Die kritische Veränderung stellt an dieser Stelle dar, dass die Entwicklungskosten bezogen auf den geschaffenen Wert reduziert werden. Die Umsetzung von Ideen ist in der Entwicklung relativ teuer. Die nachträgliche Verbesserung wird oft wegen der benötigten zusätzlichen Zeit vermieden und so sprechen Teams häufig davon, dass man bestimmte Funktionen „eigentlich nochmal besser machen müsste“. Discovery hilft uns, die Mehrzahl der Fragestellungen vor der kostspieligen Entwicklung zu lösen.

Delivery

Geht es letztlich in die Umsetzung, streben Teams nach schnellen Feedbackschleifen und suchen nach kleinen unabhängigen Paketen, die nicht nur bezüglich der technischen Funktionsfähigkeit beobachtet werden. Auch in der Delivery wird im Betrieb wieder die Rückkopplung zu den ursprünglichen Zielen für die Kunden und das eigene Geschäft geprüft. Wurden diese nicht erreicht, bleibt die neue Variante so nicht erhalten und es muss verbessert oder ein anderer Weg gefunden werden.

Ebenso wichtig wie die Prüfung auf die Zielerreichung ist hier, dass auch auf mögliche Seiteneffekte auf Kunden oder das eigene Geschäft geschaut wird. Leiden andere kritische Kennzahlen unter der neuen Funktion oder entstehen Kunden durch den neuen Vorteil andere Nachteile? Dann ist die Arbeit ebenfalls noch nicht abgeschlossen und muss fortgesetzt werden.

Kultur als Verankerung

Damit die beschriebenen Konzepte erfolgreich sein können, braucht es die passende Kultur. Um die Übernahme von Verantwortung zu fördern, müssen gängige Kontrollmechanismen reduziert werden. Das gilt dabei nicht nur für Product Leadership, sondern für die gesamte Organisation. Statt in jährlichen Planungsrunden die Wünsche aller Unternehmensbereiche an die Produktentwicklung zu verhandeln und dann die Einhaltung der versprochenen Liefertermine nachzuverfolgen, verfolgt die Organisation in enger Zusammenarbeit mit den Produktteams die Vision und Strategie. Der Fortschritt wird anhand der Strategie-

umsetzung und der erhofften Outcomes gemessen. Zusätzlich benötigt die Organisation ein Verständnis dafür, dass einzelne Schritte der Lösungsansätze auch fehlschlagen können. Sie werden als Wetten verstanden. Das Testen von Lösungen wird dabei möglichst so aufgesetzt, dass nicht nur die Fehlschläge als solche erkannt, sondern auch die Gründe dafür verstanden werden. So kann das Unternehmen zielgerichtet lernen und zunehmend treffsicherer agieren.

Aus der Praxis – Kritische Hürden für die Transformation

Kunde oder interne Prozesse – wer steht im Vordergrund?

Eine klassische Frage, die wir Organisationen stellen, ist: „Habt ihr eine Vision, was euer Angebot in einigen Jahren leisten soll, um erfolgreich zu sein? Und wenn ja – wie habt ihr dieses Zielbild erarbeitet und wie verfolgt ihr dessen Erreichung?“ Diese Fragen decken oft auf, dass es zwar Unternehmensziele wie „Europas Nr. 1 in unserer Branche“ gibt, die Produktstrategie dagegen fehlt. Die Entwicklung wird oft durch aktuelle Kundenanfragen und interne Vorschläge getrieben. Führungskräfte und die Unternehmenskultur sind darauf ausgerichtet, in jährlichen Offsites durch Konsens eine Roadmap zu gestalten. Bei den Anforderungen für die resultierende Planung wird also die Mehrheitsfähigkeit zwischen den Führungskräften über die Erfolgsaussichten am Markt priorisiert.

Im Gegensatz dazu zielt das Product Model darauf ab, die eigenen Angebote so voranzutreiben, dass sie den maximalen Wert für die Kunden schaffen und das Unternehmen

im Wettbewerb behaupten. Die Erkenntnisse zu Kundenbedürfnissen und Segmentstrukturen aus einer Vielzahl von Gesprächen und ausführlicher Datenerhebung aus Kundengewinnung und Produktnutzung bilden hierbei eine zentrale Grundlage für Zielbild, Strategie und operative Detailentscheidungen in den Teams. Die Bedürfnisse und Anforderungen der verschiedenen Abteilungen werden dagegen von den Produktteams während der Umsetzung in direkter Zusammenarbeit abgewogen und integriert.

Führungskräfte könnten hierbei einen Statusverlust empfinden, da sie ihre Wünsche nicht mehr so direkt einbringen können. Eine kontinuierliche Evangelisierung und klare Unterstützung der Veränderung durch die Unternehmensspitze sind hier entscheidend. Die größte Überzeugung leisten hier Erfolge am Markt, wenn die Fokussierung für schnellere Fortschritte und spürbare Reaktionen des Markts sorgt.

Outputs versus Outcomes

Ein besonders häufiger Ausgangspunkt ist die Fokussierung der Organisation auf Features und Liefertermine. Fokussiert auf die Frage, ob eine Produktveränderung am Tag X abgeschlossen ist, verpasst das Unternehmen die wichtigere Frage: Wie hat sich durch die Entwicklung unser Produkt und Geschäft verbessert? Im Gegensatz zum Projektgeschäft verdient das Unternehmen nicht durch die Lieferung vorher definierter Merkmale, sondern ist darauf angewiesen, dass das Produkt für Kunden möglichst attraktiv und für das Geschäftsmodell möglichst wertschöpfend

bleibt. Obwohl diese Logik außenstehend schlüssig ist, wird hier die Pulsmessung und Taktgebung des Zusammenspiels zwischen Produktbereich und Unternehmen radikal verändert. Eine naive Umstellung kann ein Vakuum entstehen lassen, in dem keine Liefertermine mehr existieren, das Unternehmen aber auch keinen anderen Fortschritt mehr erlebt und so schnell an der Transformation zweifelt.

Während der Umstellung von Outputs auf Outcomes kann zum ersten Mal die Quote der Misserfolge sichtbar werden. Ohne Vorwarnung kann dies schnell für Unruhe sorgen und die Befürchtung entstehen, dass das Modell nicht funktioniert. Tatsächlich werden diese nur transparent und Team und Organisation können endlich dazulernen.

Produktteams im Elfenbeinturm

Um erfolgreiche und wirksame Lösungen zu entwickeln, benötigen Produktteams ein tiefes Verständnis über Kontext, Ziele und Aufgaben von Kunden, verbunden mit detaillierten Nutzungsdaten. Oft fehlt ihnen jedoch der direkte Kontakt zur Zielgruppe und Ausbildung und Erfahrung in effektiven Befragungs- und Testmethoden. Ebenso mangelt es häufig an Erfahrung und professionellen Werkzeugen für die Erhebung und Analyse von Nutzungsdaten. Erst durch die Kombination von qualitativen Methoden (Kundenbefragungen und Tests von Prototypen) und quantitativen Methoden (Analyse von Verhaltensmustern und Konversionsraten) können Teams effektive Lösungen entwickeln und schnell verbessern.

Direkter Kundenkontakt ist dabei eine besondere Herausforderung. In vielen Firmen gibt es spezifische Rollen wie Vertriebsmitarbeiter oder technische Betreuer, die den Kundenkontakt übernehmen. Diese leben oft in der Überzeugung, dass das Unternehmen nur mit einer fertigen Lösung zum Kunden gehen sollte und ihre Rolle exklusiv dazu bemächtigt ist, Kundenbedürfnisse zu erfassen und zu kommunizieren. Für schnelle Entscheidungen in der Produktentwicklung ist jedoch direkter Kundenkontakt essenziell, um Erkenntnisse schnell zu integrieren, Prototypen zu testen und ein tiefes Verständnis für das Produkt zu entwickeln. Klare Motivationen und Rollenverteilungen sind nötig, um Vertrauen und Zusammenarbeit zu stärken.

Zugang zu Produktionsdaten wie Business Intelligence und Product Analytics ist entscheidend, um Erfolge und mögliche Nebeneffekte festzustellen und den Lernzyklus zu vervollständigen. Ohne diese Daten bleibt die Ergebnisorientierung unvollständig.

Neue Rollen, neue Qualifikationen

Letztlich muss noch der Elefant im Raum benannt werden: Die Anforderungen an Qualifikation und Persönlichkeit der Mitarbeitenden sind häufig radikal verschieden. Insbesondere die Produktmanagement-Rolle des Modells unterscheidet sich beispielsweise grundsätzlich von den Rollen eines Projektleiters oder Scrum Product Owners. Verantwortung für die Lieferung von Ergebnissen zu übernehmen unterscheidet sich beispielsweise deutlich davon, die Umsetzung von internen Anforderungen wie beispiels-

weise dem Vertrieb in lauffähigen Implementierungen zu verwandeln.

Transformation: Einführung des Product Operating Models

Vor einer Transformation sind zwei essenzielle Punkte zu beachten:

1. Klares Zielverständnis und Umfang der Veränderung

Das Product Operating Model ist nur erfolgreich, wenn die Geschäftsleitung voll dahintersteht. Es ist entscheidend, ein gemeinsames Verständnis der Mechanismen und nötigen Prinzipien des Modells zu entwickeln und die Unterschiede zum aktuellen System deutlich zu machen. Nur durch eine klare Abgrenzung zum Status quo wird das Ausmaß der Veränderung erkennbar, was Überraschungen und Widerstände während der Transformation minimiert. Sollten dennoch Widerstände auftreten und es nicht gelingen, die Vorteile des neuen Ansatzes zu vermitteln, muss das Transformationsziel überdacht werden. Es könnte möglich sein, einzelne effektive Praktiken und Konzepte schrittweise zu integrieren.

2. Erarbeitung einer individuellen

Transformationsstrategie

Wie Marty Cagan betont, erfordert die Vielfalt der Ausgangssituationen eine detaillierte Einzelfallbetrachtung. Unsere Erfahrung bestätigt, dass die Elemente des Product Operating Models vielfach voneinander abhängig sind, sodass das Fehlen eines Elements den Erfolg eines anderen beeinträchtigen kann. Eine falsche oder unbe-

dachte Umsetzungsreihenfolge kann daher zu Misserfolgen, Irritationen und Vertrauensverlust in das Modell oder Personen in zentralen Rollen führen.

Wenn das Unternehmen entschlossen ist, die Transformation fortzusetzen, sollte man erfahrene Unterstützung hinzuziehen. Dies kann externe Beratung durch Interimsmanger oder Coaches sein oder auch die Einstellung von Führungskräften, die bereits erfolgreiche Implementierungen durchgeführt haben. In größeren Organisationen können diese Ansätze kombiniert werden.

Vorbereitende Schritte

Obwohl eine Transformation zum Product Model bewusst durchgeführt werden muss, möchten wir ein paar grundlegende Fähigkeiten aufgreifen, die für das Modell essenziell sind, jedoch auch für viele andere Konstellationen Wert schaffen können. Dazu gehören:

- **Kulturwandel**

Teams sollen von Lieferterminen zu ergebnisorientiertem Arbeiten wechseln („Outcomes“ statt „Output“). Produktteams sollten nur noch an Features arbeiten, wenn sie erklären können, welches Problem damit gelöst wird und welche messbare Wirkung davon erwartet wird. Dies erfordert ein besseres Kundenverständnis und verlagert die Aufmerksamkeit der Gespräche mit internen Stakeholdern. Dabei entstehen oft Lösungen, die die ursprüngliche Idee übertreffen und die Zusammenarbeit verbessern. Die Planung fokussiert sich auf die erwartete Wirkung, was interne Priorisierungen erleichtert.

- **Direkter Kundenkontakt**

Wenn Teams direkt mit Kunden interagieren, verbessert sich das Verständnis für die Aufgabenstellung und den Kontext. Mittels Coaching können Teams hier schnell ihre Wertschöpfung steigern und Stolperfallen wie das Wecken falscher Erwartungen bei Kunden vermeiden. Die Rollen im Kundenkontakt, besonders bei Vertriebspersonal, müssen klar gemeinsam definiert und das Zusammenspiel gelernt werden, um Konflikte zu vermeiden.

- **Schnelle und kurze Feedbackschleifen**

Auch in termingetriebenen Organisationen fördern Prototypen und die Messung der Wirksamkeit von Veränderungen mittels kurzer Releasezyklen die Reduzierung fehlgeschlagener Feature Launches.

- **Einführung von Product Analytics**

Analytics-Tools wie Amplitude, Mixpanel o. ä. sowie ein Kennzahlensystem sind kritisch, um die Wirksamkeit von Änderungen zu messen und den Fortschritt sichtbar zu machen. Ein Auslassen dieser Werkzeuge ähnelt der Nutzung eines Navigationssystems ohne GPS-Positionierung.

Fazit

Das Product Operating Model bietet Unternehmen eine Perspektive, die eigene Organisation selbst in ein erfolgreiches Innovationsunternehmen zu verwandeln. Die vorgestellten Konzepte und dahinterliegenden Prinzipien geben dabei nicht nur ein Zielbild vor, sondern können bereits für die Standortbestimmung des eigenen Unter-

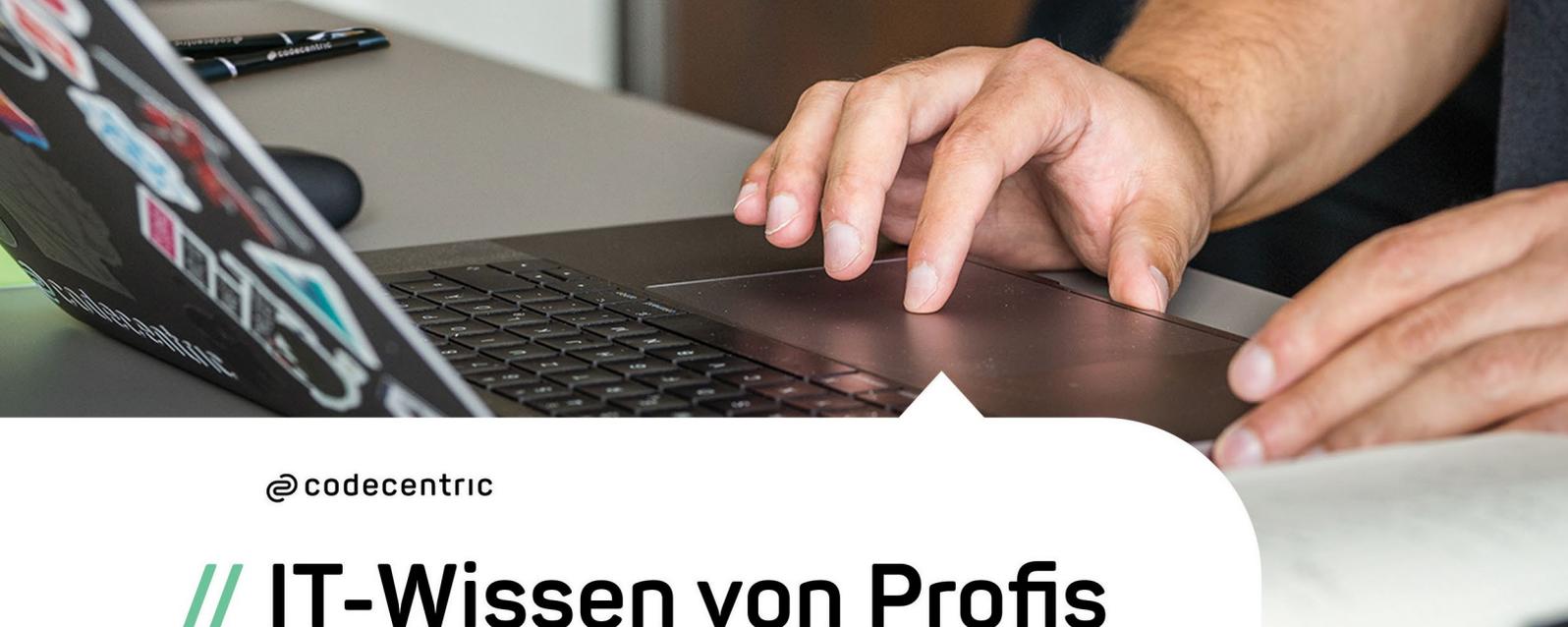
nehmens und zum Aufzeigen von Potenzialen eingesetzt werden. Aufgrund der Vielfalt möglicher Ausgangsszenarien sollten sich Unternehmen stets mit erfahrenen Anbietern in Verbindung setzen, um sich auf der Reise begleiten zu lassen. Um sich ein erstes Bild zu verschaffen, wie das Product Operating Model im Detail funktioniert und welche Potenziale sich dabei vielleicht für euch ergeben könnten, sind wir gerne zum Austausch bereit.

***Marc Pudelski** ist als Product Consultant bei der codecentric AG in Solingen tätig. Er hilft Unternehmen dabei, Produkte mit mehr Markterfolg zu entwickeln. Sein Schwerpunkt liegt im Bereich Produktstrategie und Discovery.
marc.pudelski@codecentric.de*

***Ralf Westbrock** unterstützt seit mehr als 15 Jahren Unternehmen dabei, neue und bestehende digitale Geschäftsmodelle mittels Wertschaffung für Kunden erfolgreicher zu machen. Als Transformation Coach und Interim Manager vermittelt er dabei die Konzepte und Prinzipien des vorgestellten Modells.
ralf.westbrock@codecentric.de*

Referenzen

- [1] Marty Cagan, *Transformed* (2024)
- [2] Eigene Darstellung basierend auf Marty Cagan, *Transformed* (2024)



@codecentric

// IT-Wissen von Profis für Profis

Unser Tech-Blog zu aktuellen
IT-Themen

- ▶ Leichte Suche nach aktuellen und beliebten Themen
- ▶ Ein Blogpost gefällt euch? Hinterlasst ein <♥>!
- ▶ Featured und beliebte Posts
- ▶ Einfache Kontaktmöglichkeit mit den Blog-Autor*innen

Suche nach Thema oder Technologie



THEMENFILTER

Agilität API Big Data CI/CD Cloud Container DevOps Digitalisierung
IoT IT-Security KI Kubernetes Microservices Produktmanagement

▶ Schaut vorbei und hinterlasst ein <♥> auf [blog.codecentric.de!](https://blog.codecentric.de)



JETZT KOSTENLOS REGISTRIEREN UND KEINE AUSGABE VERPASSEN:

www.dersoftwerker.de



System (Ops) Integration und Automatisierung mit DevOps, Intelligenz, Auto-Management, Anomalieerkennung und Predictive Analytics zu höherer Plattformen. Es lohnt sich, sich als eine natürliche Evolution der Cloud zu sehen, die sich gegenseitig unterstützen. Das Ziel von Plattformen ist es, die Arbeit zu erleichtern, die dann Plattformen betreiben werden können. Diese Plattformen ermöglichen es, verschiedene Dienste, Tools und Services, die es Entwicklern ermöglichen, Plattformen zu entwickeln, zu testen und

Wohin geht die Plattform?

Es ist natürlich schwierig zu beantworten, denn die Zukunft ist, wie so oft, „It depends“. Plattformen, die sich der Erfahrung nach am meisten lohnen, sind die, die am meisten in der Lage sind, mehrere Entwicklungsteams zu unterstützen. Die meisten Väter betonen, dass sie sich auf ihr Kernthema konzentrieren, um Beispiele für die Weiterentwicklung des Produktes. Um sich hier zu fokussieren, kann es sich dann

Die Autorinnen und Autoren dieser Ausgabe



Nils Bauroth
Cloud Consultant (cc Cloud GmbH)



Kamil Braschke
IT Consultant



Roger Butenuth
Senior Integration Architect



Shirin Elsinghorst
Team Lead | Consultant Data & AI



Miriam Greis
Lead API Consultant



Annegret Junker
Chief Software Architect



Daniel Kocot
Senior Solution Architect |
Head of API Consulting



Janina Lerner
Organisationsentwicklung |
Agile Coach



Marco Paga
Senior Solution Architect |
Senior IT Consultant



Marc Pudelski
Product Consultant



René Rother
IT Consultant



Elena Rüdener
Product Consultant



Antonia Schmalstieg
IT Security Consultant



Marc Schnitzius
Platform Engineering Lead



Denis Stalz-John
Machine Learning Specialist



Sujeevan Vijayakumaran
Senior Solutions Architect (GitLab)



Ralf Westbrook
Principal Product &
Innovation Consultant



Manuel Zapf
Solution Architect

Sechs Säulen für nachhaltige Digitalisierung

Responsible Digital Solutions als ganzheitlicher Ansatz für Innovation

von **Elena Rüdener** und **Shirin Elsinghorst**



| | |
|--|------------------|
| Salpidae | |
| KOMMEN IN ÄQUATORNAHEM, GEMÄSSIGTEM UND KALTEM MEER VOR | |
| ERNÄHREN SICH VON PHYTOPLANKTON | GRÖSSE 8 MM-5 CM |
| SINKENDE SALPENAUSSCHEIDUNGEN UND -KÖRPER TRANSPORTIEREN KOHLENSTOFF ZUM MEERESBODEN | |

In einer Welt, in der Digitalisierung die Wertschöpfungsketten durchdringt und unser Leben grundlegend verändert, sind Software, Daten und digitale Technologien wie künstliche Intelligenz nicht länger nur unterstützende Werkzeuge. Sie dienen als zentrale Grundlage für Innovation und Effizienz, die tiefgreifende Veränderungen ermöglichen. Gleichzeitig erfordert die steigende Komplexität globaler Herausforderungen ein Umdenken in der digitalen Produktentwicklung. Verantwortungsvolle digitale Lösungen sind unser Ansatz, um Unternehmen zu helfen, innovativ im Wettbewerb zu bestehen und zukunftsfähig zu sein.

Verantwortungsvolle digitale Lösungen (engl. Responsible Digital Solutions, kurz RDS) richten sich strategisch an einem Markt aus, der nachhaltigen und ethischen Prinzipien folgt. Um visionäre digitale Produkte und Lösungen zu entwickeln, die die Welt positiv verändern, haben wir ein Assessment entwickelt. Damit befähigen wir Unternehmen, strategische Handlungsfelder für die Umsetzung von RDS abzuleiten. Dieses stellen wir im Folgenden vor und verdeutlichen die Anwendung anhand eines Use Case mit Responsible und Explainable AI.

Zentrale Herausforderungen für Unternehmen

Unternehmen stehen vor vielfältigen Herausforderungen, um im Wettbewerb zu bestehen: Die Komplexität steigt, technologische Innovationen werden immer schneller entwickelt und die Digitalisierung ist in nahezu jeden

Aspekt der Wertschöpfungen und des privaten Lebens vorgedrungen. Globale Krisen, wie geopolitische Auseinandersetzungen, Rezession, Fachkräftemangel, soziale Ungerechtigkeiten und der Klimawandel, setzen Unternehmen zusätzlich unter Druck. Wie können Firmen diesen Herausforderungen begegnen?

Zur Kostenreduktion steht bei den meisten Unternehmen die Effizienzsteigerung im Fokus. Erfolgversprechend ist für sie, die Betriebs-, Weiterentwicklungs- und Wartungskosten der IT zu reduzieren und ihre Geschäftsprozesse zu automatisieren [1]. Neue Technologien wie künstliche Intelligenz (KI) helfen dabei. Allerdings sind Vertrauen und Akzeptanz für den erfolgreichen Einsatz von KI erforderlich. Das fehlt bisher jeder zweiten Person und hemmt den Einsatz [2]. Auch begegnen Unternehmen veränderten Märkten. Der Fachkräftemangel ist stark zu spüren: Die Menschen, die benötigt werden, um Visionen umzusetzen, sind schwer zu finden. Ca. 1,8 Millionen Stellen blieben im Jahr 2023 in der deutschen Gesamtwirtschaft unbesetzt [3]. Um rares qualifiziertes Personal zu gewinnen, müssen Unternehmen soziale und ökologische Nachhaltigkeit auf ihre Agenda setzen. Sie ist für über 80 Prozent der Berufseinsteigenden und für über 60 Prozent der Arbeitssuchenden relevant bei der Arbeitgeberwahl [4]. Auch fast 60 Prozent der Konsument*innen wählen ihre Produkte und Mobilität nach diesem Kriterium aus [5]. Verantwortungsvolle Lösungen und Produkte werden nachgefragt und helfen Unternehmen dabei, zukunftsfähig zu sein. Vorschriften wie z. B. der EU AI Act, der EU Accessibility Act und die EU Corporate Sustainability Reporting Directive

(CSRD) verpflichten Unternehmen, Transparenz und Verantwortung in den Mittelpunkt ihrer Strategien zu stellen. Dabei geht es nicht nur darum, dass sie Bußgelder vermeiden oder Imageschäden abwenden; im Kern geht es um die Entwicklung von Lösungen, die echten Mehrwert bieten, und um den Aufbau einer Welt, in der digitale Innovationen nachhaltig und inklusiv sind [6].

Unternehmen müssen also Antworten auf drei zentrale Fragen finden:

1. Wie können sie innovativ und wirtschaftlich **erfolgreich bleiben** – und das alles mit **weniger Menschen**?
2. Wie können sie das **Vertrauen** in neue Technologien stärken und Akzeptanz gewinnen, um das **Potenzial** für sich **nutzbar zu machen**?
3. Wie können sie digitale Lösungen entwickeln, die sowohl **wirtschaftlich** als auch **ökologisch** und **sozial** sind?

Ganzheitlich denken, nachhaltig handeln: Die sechs Säulen für nachhaltigen Erfolg

Die Antwort liegt für uns in der Ganzheitlichkeit und der engen interdisziplinären Zusammenarbeit von Fachbereichen in IT und Business. Auch wenn kurzfristige Maßnahmen als „schnelle Pille“ verlockend erscheinen, bieten sie in der Regel keine nachhaltige Lösung für die tiefgreifenden Herausforderungen, denen Unternehmen gegenüberstehen. Was sie brauchen, ist eine grundlegende Veränderung ihrer Herangehensweise. Die Bereitschaft,

Arbeitsweisen zu überdenken, neue Methoden und Technologien zu adaptieren, die Zusammenarbeit zu intensivieren und eine Kultur der kontinuierlichen Verbesserung zu etablieren, sind essenziell. Wie sollten Unternehmen jedoch beginnen?

Wir streben danach, Unternehmen dabei zu unterstützen, Lösungen zu entwickeln, die auch morgen wertvoll sind. Software spielt dabei die befähigende Rolle: Sie ermöglicht Lösungen, die gleichzeitig wirtschaftlich erfolgreich sind und einen positiven Einfluss auf Gesellschaft und Umwelt haben. Wir glauben, dass Unternehmen mit sechs Säulen

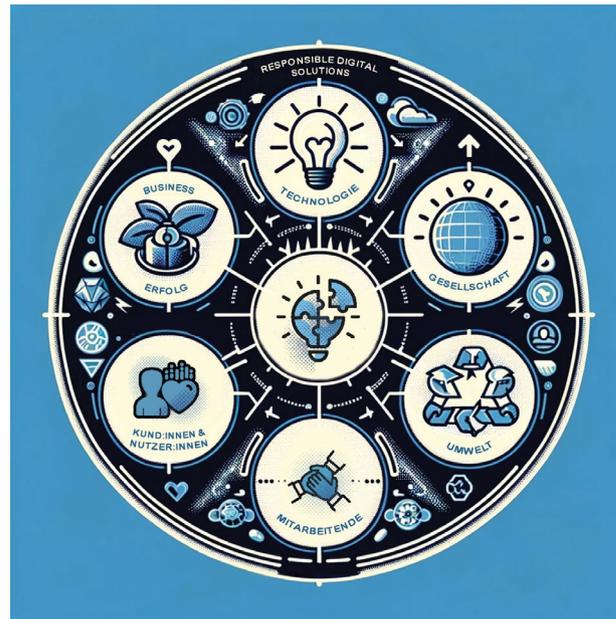


Abb. 1: Die sechs Säulen für nachhaltigen Erfolg | Responsible Digital Solutions

die Prinzipien der Nachhaltigkeit in verantwortungsvolle digitale Lösungen umsetzen können. Sie weisen den Weg zu nachhaltigem Erfolg: *Business-Erfolg, Kund*innen & Nutzer*innen, Mitarbeitende, Umwelt, Gesellschaft und Technologie* (Abb. 1). Dadurch sind Unternehmen in der Lage, Nachhaltigkeit und Digitalisierung in Einklang zu bringen und strategisch voranzutreiben.

Das Responsible Digital Solutions Assessment

Unternehmen identifizieren mithilfe der sechs Säulen Handlungsfelder, um digitale Lösungen verantwortungsvoll zu gestalten. Gezielte Fragestellungen je Säule dienen als Instrument, das aufzeigt, wo Unternehmen aktuell stehen und in welchen Bereichen es Verbesserungspotenziale gibt. Das Ergebnis ermöglicht ihnen, dort anzusetzen, wo sie im Kontext ihrer spezifischen Situation die größte Wirkung absehen.

Business-Erfolg: RDS tragen unmittelbar zum Unternehmenserfolg bei. Eine ganzheitliche Betrachtung ist dabei essenziell: die Verankerung von (digitaler) Verantwortung in der Unternehmensstrategie. Dabei ist es wichtig, klare Businessziele zu definieren, die RDS-Initiativen und -Produkte fördern, und deren Wirkung sichtbar zu machen. Eine Kultur des Lernens ermöglicht Unternehmen, sich flexibel an veränderte Gegebenheiten anzupassen und frühzeitig neue Ideen auszuprobieren. Durch RDS können neue Marktsegmente erschlossen, Effizienz erhöht und die Marktposition gestärkt werden.

Auszug der Fragen, die sich Unternehmen in der Säule „Business-Erfolg“ stellen können:

- Wie können wir unsere Geschäftsprozesse optimieren und unser Geschäft verantwortungsvoll skalieren?
- Weshalb existieren wir als Unternehmen, welchen Beitrag wollen wir leisten, wie wollen wir die Zukunft mitgestalten?
- Wie können wir einen Raum für Innovation schaffen, der RDS-Produkte und Initiativen fördert?

Kund*innen & Nutzer*innen: Über 80 Prozent der neu eingeführten Produkte scheitern [7]. Nur, wenn wir Produkte entwickeln, die echte Probleme lösen, werden sie auch genutzt. Ressourcen wie Zeit und Budget sind so nachhaltig investiert. Daher ist es essenziell, den Nutzungskontext, samt der Bedürfnisse und Probleme der Kund*innen und Nutzer*innen, kontinuierlich zu verstehen. Lösungsideen werden in kleinen Experimenten früh getestet. So können RDS auf Evidenz basierend gestaltet und adaptiert sowie das Risiko ihres Scheiterns minimiert werden. Dann wird nachhaltiger Mehrwert für das Unternehmen, Kund*innen und Nutzer*innen geschaffen.

Auszug der Fragen, die sich Unternehmen in der Säule „Kund*innen und Nutzer*innen“ stellen können:

- Wie können wir die relevanten Ziele und Probleme der Nutzer*innen in ihrem Kontext kontinuierlich verstehen und lösen?

- Wie können wir Produkte und Lösungen partizipativ gestalten?
- Wie können wir Produkte und Lösungen entwickeln, die die Privatsphäre und Daten der Nutzer*innen schützen?

Mitarbeitende: In Zeiten des Fachkräftemangels müssen Unternehmen attraktive Arbeitsumgebungen schaffen, um qualifizierte Mitarbeitende zu gewinnen und zu halten. Soziale und ökologische Verantwortung spielen eine wichtige Rolle in der Arbeitgeberattraktivität. Ein Umfeld, das die Entwicklung nachhaltiger Lösungen fördert, zieht Talente an und hält sie langfristig im Unternehmen. Cross-funktionale Zusammenarbeit und ein offener, vertrauensvoller Umgang, auch mit Fehlern, verbessern den Wissensaustausch und die Entscheidungsfindung. Vielfältig besetzte Teams sind innovativer und tragen zum langfristigen Erfolg des Unternehmens bei. Sie führen zu höherer Zufriedenheit der Mitarbeitenden und damit zur Bindung ans Unternehmen [6].

Auszug der Fragen, die sich Unternehmen in der Säule „Mitarbeitende“ stellen können:

- Wie können wir ein Umfeld schaffen, in dem sich die Mitarbeitende beruflich und persönlich weiterentwickeln können?
- Wie bleiben Mitarbeiter*innen mental und physisch gesund sowie leistungsfähig?
- Wie können wir mit Unterbesetzung umgehen, ohne die Mitarbeitenden zu belasten?

Umwelt: In Anbetracht der drängenden Umweltprobleme unserer Zeit können Unternehmen nicht ignorieren, welchen Einfluss ihre digitalen Lösungen auf die Umwelt haben. Verantwortungsvolles Handeln in diesem Bereich ermöglicht Unternehmen nicht nur, ihre Kosten zu senken: Die Unternehmen verbessern ihre Reputation sowie Compliance und leisten einen Beitrag zum Umweltschutz. Ziel ist es, durch nachhaltige Praktiken wie Energieeffizienz, Abfallreduktion mittels Kreislaufwirtschaft und den Einsatz erneuerbarer Ressourcen den ökologischen Fußabdruck zu reduzieren.

Auszug der Fragen, die sich Unternehmen in der Säule „Umwelt“ stellen können:

- Wie können wir die ökologischen Auswirkungen unserer digitalen Lösungen, Produkte und Dienstleistungen messen und transparent machen?
- Auf welche Weise kann Digitalisierung helfen, in der Entwicklung und im Betrieb Ressourcen (z. B. Energie, Wasser, Luft, Materialien, Geldmittel, Reserven, Daten usw.) effizienter zu nutzen?
- Wie können wir Lösungen und Produkte entwickeln, die Bestehendes wiederverwenden und den Einsatz neuer Ressourcen minimieren?

Gesellschaft: Unternehmen übernehmen durch ihre Praktiken gesellschaftliche Verantwortung, da ihr Einfluss weit über den eigenen Kontext hinausgeht [8, 9, 10]. Das ist nicht nur eine Frage von Werten und Moral,

sondern hat auch klare wirtschaftliche Aspekte. Für Kund*innen, Nutzer*innen sowie Mitarbeitende ist gelebte Verantwortung immer mehr die Voraussetzung. Für die Entwicklung inklusiver (KI-)Technologien und -Lösungen ist die Berücksichtigung gesellschaftlicher Diversität unerlässlich. Damit entgegnen sie (systematischer) Diskriminierung und ermöglichen gesellschaftliche Teilhabe [6].

Auszug der Fragen, die sich Unternehmen in der Säule „Gesellschaft“ stellen können:

- Welche Werte haben wir als Unternehmen und wie können wir diese leben?
- Wie können wir (systematische) Diskriminierung sowie Bias in Daten und KI vermeiden?
- Wie können wir die gesellschaftlichen Auswirkungen unserer digitalen Lösungen erkennen und sichtbar machen?

Technologie: Unternehmen setzen Software und Technologien als befähigende Funktion ein, um RDS zu gestalten. Dabei geht es um die Integration fortschrittlicher Technologien in alle Aspekte der Geschäftstätigkeit. Unternehmen müssen sicherstellen, dass sie Technologien auswählen, die nicht nur innovativ sind. Sie sollten für den Kontext funktionieren und flexibel sein. Dazu gehören eine mögliche Skalierung, Weiterentwicklung und der Betrieb. Dabei sollen sie ethische Grundsätze widerspiegeln.

Auszug der Fragen, die sich Unternehmen in der Säule „Technologie“ stellen können:

- Anhand welcher Kriterien wählen wir die geeignete Technologie für die Umsetzung von RDS aus?
- Wie bleiben wir auf dem neuesten Stand der technologischen Entwicklungen und Standards?
- Wie können wir digitale Lösungen entwickeln, die flexibel erweiter- und anpassbar sind?

Software mit Sinn – Unser Modulbaukasten für Responsible Digital Solutions

Durch das Assessment werden Handlungsfelder in Unternehmen sichtbar für die Umsetzung von RDS. Eine gleichzeitige Priorisierung und Realisierung auf allen Ebenen ist nicht zwingend erforderlich. Unser Vorgehen, um RDS umsetzbar zu machen, ist wie ein Modulbaukasten flexibel und erweiterbar konzipiert. Einzelne Bausteine befähigen Unternehmen, maßgeschneiderte Lösungen zu entwickeln, die genau auf ihre Bedürfnisse und Ziele abgestimmt sind (Abb. 2). Wir möchten Unternehmen dort abholen, wo sie stehen und sie bestmöglich auf ihrer Reise zu verantwortungsvoller digitaler Lösungs- und Produktentwicklung begleiten.

Use Case Responsible und Explainable AI

Mit unserer Kundin, der Medizin GmbH, entwickeln wir eine verantwortungsvolle digitale Lösung basierend auf Responsible und Explainable AI (Abb 3). Der fiktive Use

Case basiert auf realen Erfahrungen aus Kundenprojekten und kann an andere Branchen, Unternehmen und Modulbausteine angepasst werden. Damit zeigen wir, wie wir RDS mithilfe des Modulbaukastens und der sechs Säulen umsetzen.

Die Medizin GmbH ist ein Hersteller von Produkten und Lösungen zur Unterstützung von Ärzt*innen in der radiologischen Diagnostik und Befundung. In unserer Zusammenarbeit geht es im Kern um die folgenden Aspekte, um eine sichere Diagnostik und effektive Patient*innen-Versorgung zu ermöglichen:

- Entscheidungsunterstützung,
- Steigerung von Vertrauen und Akzeptanz in (KI-) Technologie,
- Reduktion von Bias in Daten

Business-Erfolg: Für die Medizin GmbH ist es erfolgskritisch, dass sie sichere und fehlerfreie medizinische Produkte anbietet, denen ihre Kund*innen vertrauen. Daher ist Verantwortung Teil der Geschäftsstrategie. Die Entscheidungsunterstützung mittels KI in der radiologischen Diagnostik wird als erfolgversprechende Initiative identifiziert. Um das Potenzial von KI nutzen zu können, ist das Vertrauen der Ärzt*innen in die Technologie essenziell. Durch die Initiative wird die Qualitätssteigerung der Diagnostik und folgenden Therapien fokussiert. Die Zeitreduktion der Diagnostik durch Prozessoptimierung zahlt ebenfalls darauf ein, da Ärzt*innen mehr Fokus auf die Patient*innen-Interaktion ermöglicht wird. Zusätzlich ist die Einhaltung von Gesetzen und Richtlinien erfolgskritisch, was bei der Lösungsentwicklung gewährleistet wird.

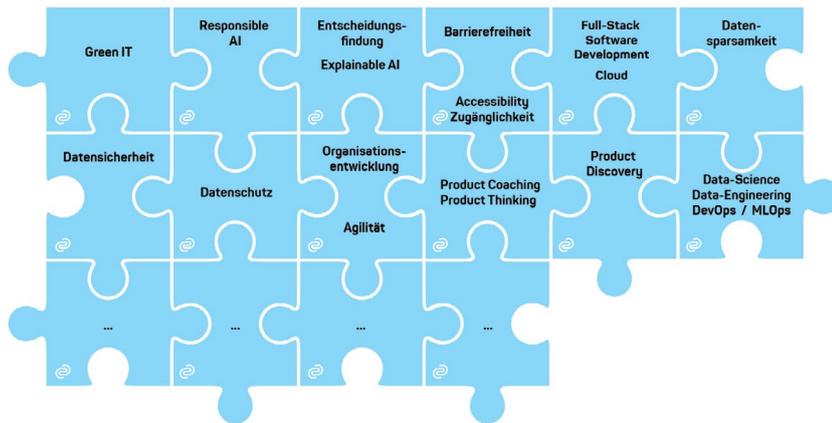


Abb. 2: Modulbaukasten der codecentric AG für Responsible Digital Solutions

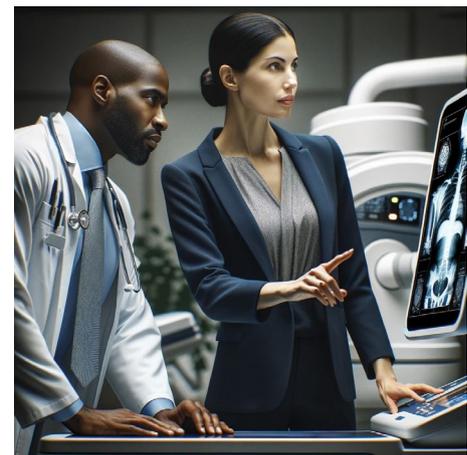


Abb. 3: Use-Case für Responsible Digital Solutions mit Responsible und Explainable AI [11]

Kund*innen & Nutzer*innen: Um wirksame Lösungen zu entwickeln, ist es essenziell, den Nutzungskontext (kontinuierlich) zu verstehen: Das Ziel der Radiolog*innen ist es, maligne Veränderungen auf MRT-Bildern frühzeitig zu erkennen, um sie rechtzeitig behandeln zu können. Dabei helfen Fachkräften ihre Erfahrung und Ausbildung. Jeden Tag viele hundert Bilder manuell zu analysieren ist anstrengend, zeitaufwendig, repetitiv und fehleranfällig. Einen Fehler zu machen und etwas zu übersehen, kann umfassende Auswirkungen für die Patient*innen haben. Gleichzeitig gibt es nur wenige Fachkräfte, die diese hochspezialisierte Aufgabe übernehmen können. Die Analyse nimmt viel Zeit ein und es bleibt weniger Zeit für Gespräche mit den Patient*innen – eigentlich ein essenzieller Teil des verantwortungsvollen Diagnoseverfahrens, den eine Maschine nicht ersetzen kann.

Lösungsentwicklung: Mithilfe von KI-gestützten Modellen kann die Bildauswertung zumindest teilautomatisiert werden. Allerdings besteht ein hoher Bedarf an Vertrauen in die Technologie bei Ärzt*innen, da Fehlentscheidungen weitreichende Folgen haben. Nur so wird die Entscheidungsunterstützung auch tatsächlich genutzt. Hier setzt *Explainable AI* an. Indem Entscheidungskriterien des Modells besser nachvollziehbar sind, kann sich die Ärzt*in sicher sein, dass das Modell die richtigen Kriterien zur Entscheidungsfindung heranzieht und damit dem Ergebnis stärker vertrauen. Die geeignetste Lösungsoption finden wir gemeinsam mit der Fachärzt*in. Zum Beispiel können der Ärzt*in nur die kritischsten Fälle zur manuellen Auswertung gezeigt werden oder die, bei denen sich der Algo-

rithmus unsicher ist. Eine andere Möglichkeit ist es, die Auswertung an die Konzentrationsfähigkeit der Ärzt*innen im Tagesverlauf anzupassen. Die auszuwertenden Bilder werden demnach durch einen Algorithmus sortiert. Zu Beginn des Tages priorisiert dieser die schwierigen, kritischen Fälle, welche die höchste Aufmerksamkeit benötigen. Die eindeutigen Bilder folgen später. So ein Vorgehen erleichtert die Arbeit, indem sie als weniger anstrengend wahrgenommen wird. Dies führt zu einer geringeren Fehlerquote und zu höherer Effizienz durch die schnellere Auswertung. Damit bleibt mehr Zeit für Interaktion mit Patient*innen. Dies führt schließlich zu einer Qualitätssteigerung der Patient*innenversorgung. Die Erweiterung des Services, z. B. um Therapievorschlüsse, sind anknüpfende Entwicklungsschritte.

Mitarbeitende: Die Entwicklung mit der Medizin GmbH erfolgt in einem interdisziplinären und vielfältig besetzten Team mit Expertise in Data Science, Engineering, Product Management, klinischer Diagnostik, Design und Rechtswesen. Dabei wird iterativ und mit kontinuierlicher Einbindung der Nutzer*innen gearbeitet.

Umwelt: Bei der Wahl der geeigneten KI-Modelle und dem Aufbau der Infrastruktur wird der ökologische Fußabdruck berücksichtigt und möglichst minimal ausgelegt. Dies führt auch zu einer Kostenreduktion durch geringere(n) Datennutzung und Energieverbrauch.

Gesellschaft: Die KI-Modelle greifen auf eine bestehende Datenbasis vieler Bilder zurück, mit der sie trainiert werden. Verzerrungen (Bias) in den Daten können zu unerwünschten, fehlerhaften Auswertungsergebnissen für

manche Patient*innen-Gruppen führen. Zum Beispiel kann es geschlechtsspezifische Unterschiede in der Diagnostik geben, die der vorliegende Datensatz nicht widerspiegelt. Dieses Risiko wird durch Responsible AI reduziert, indem die Datensätze strukturiert, analysiert, ggf. angepasst und fortlaufend überwacht werden. Zudem ist es wichtig, die Lösungen mit einer vielfältigen Zielgruppe zu testen, um die Zugänglichkeit und Inklusion zu gewährleisten. Diese Maßnahmen führen zu einer Qualitätssteigerung der Diagnostik und zahlen direkt auf den Business-Erfolg ein.

Technologie: Basierend auf dem Nutzungskontext werden geeignete Modelle und Technologien für Responsible und Explainable AI genutzt. In der medizinischen Diagnostik steht die Datensicherheit an oberster Stelle. Dementsprechend werden Anforderungen an Gesetzeskonformität berücksichtigt.

Die Zukunft gestalten – nachhaltig und mit Erfolg

Bei codecentric glauben wir, dass exzellente Software mehr ist als nur Code. Software hat das Potenzial, eine bessere Zukunft zu ermöglichen. Daher unterstützen wir Unternehmen ganzheitlich und entwickeln Lösungen, die auch morgen noch wertvoll sind. Als zertifizierte B-Corporation™ haben wir diese Werte im Unternehmen fest verankert. Wir übernehmen aktiv soziale und ökologische Verantwortung und streben in unserer täglichen Arbeit nachhaltige Lösungen an. Von diesem Wissen lassen wir unsere Kund*innen profitieren, sodass sich Geschäftserfolg mit gesellschaftlichem Mehrwert verbindet.

Wir sind fest davon überzeugt, dass die Investition in verantwortungsvolle digitale Lösungen und Produkte nicht nur eine ethische und ökologische Frage ist. Sie ist auch eine strategische Entscheidung, die langfristig zum Erfolg jedes Unternehmens beitragen wird. Lasst uns diesen Weg gemeinsam gehen! Wir freuen uns auf eure Gedanken und Diskussionen, wie wir dieses wichtige Thema gemeinsam vorantreiben können.

Elena Rüdener ist Product Consultant bei codecentric. Ihre Begeisterung für menschenzentrierte, nachhaltige Produktentwicklung und modernes Product Management ist ihr Antrieb. Sie nutzt kreative Ansätze, um digitale Produkte und Lösungen strategisch sowie nachhaltig wirksam zu gestalten. Mit Erfahrungen in verschiedenen Branchen, einem Studium des Wirtschaftsingenieurwesens und Design Thinking bringt sie vielseitige Perspektiven bei Kund*innen ein. elena.ruedener@codecentric.de

Dr. Shirin Elsinghorst hat in der Bioinformatik promoviert. Sie wendet Analyse- und Visualisierungsmethoden verschiedener Bereiche an – z. B. Machine Learning, klassische Statistik, Textanalyse etc. –, um Informationen aus Daten nutzbar zu machen. Besonders die Themen Responsible AI, Explainable AI und Data Storytelling liegen ihr am Herzen, um Daten, Analysen und KI auch Nicht-Experten zugänglich zu machen. shirin.elsinghorst@codecentric.de

Referenzen

- [1] <https://www.cio.de/a/it-effizienz-rueckt-wieder-in-den-fokus,3714164>
- [2] Gillespie, N., Lockey, S., Curtis, C., Pool, J., & Akbari, A. (2023). Trust in Artificial Intelligence: A Global Study. The University of Queensland and KPMG Australia. doi: 10.14264/00d3c94
- [3] <https://www.dihk.de/de/themen-und-positionen/fachkraefte/beschaeftigung/fachkraefteengpaesse-gefahren-erfolg-in-wichtigen-schlüsseltechnologien-107880>
- [4] <https://www.eib.org/de/surveys/climate-survey/5th-climate-survey/climate-impact-important-factor-for-jobseekers.htm>
- [5] [https://www.tuev-verband.de/?tx_epxelo_file\[id\]=953554&cHash=b68004fd6f9093d95ccb89cd7c13c11a](https://www.tuev-verband.de/?tx_epxelo_file[id]=953554&cHash=b68004fd6f9093d95ccb89cd7c13c11a)
- [6] <https://www.bitkom.org/sites/main/files/2023-04/230413PPDigitaleGesellschaft.pdf>
- [7] <https://professionalprograms.mit.edu/blog/design/why-95-of-new-products-miss-the-mark-and-how-yours-can-avoid-the-same-fate/>
- [8] Buolamwini, J., & Gebru, T. (2018). Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification. Proceedings of Machine Learning Research, 81, 77-91. <https://proceedings.mlr.press/v81/buolamwini18a/buolamwini18a.pdf>
- [9] Garcia, A.C.B., Garcia, M.G.P. & Rigobon, R. Algorithmic discrimination in the credit domain: what do we know about it?. AI & Soc (2023). <https://link.springer.com/article/10.1007/s00146-023-01676-3>
- [10] Sherman, L. E., Payton, A. A., Hernandez, L. M., Greenfield, P. M., & Dapretto, M. (2016). The Power of the Like in Adolescence: Effects of Peer Influence on Neural and Behavioral Responses to Social Media. Psychological Science, 27(7), 1027-1035. <https://journals.sagepub.com/doi/10.1177/0956797616645673>
- [11] GPT-4, <https://openai.com/chatgpt/>

Who's afraid of AI?

Wie Unternehmen Ängste und Vorbehalte ihrer Mitarbeitenden gegenüber GenAI überwinden – und Vertrauen aufbauen können

von **Janina Lermer**

Krill

KOMMT WELTWEIT VOR, STELLT EIN WESENTLICHES GLIED IN DER MARINEN NAHRUNGSKETTE DAR UND TRÄGT ZUR REGULIERUNG DES GLOBALEN KLIMAS BEI

FRISST PHYTOPLANKTON UND ALGEN, EINIGE ARTEN JAGEN ABER AUCH ZOOPLANKTON UND FISCHLARVEN

GRÖSSE
< 6 CM

VERSTECKT SICH TAGSÜBER IM KÜHLEN, DUNKLEN WASSER DER DÄMMERUNGZONE DES OZEANS UND SCHWIMMT NACHTS AN DIE OBERFLÄCHE

Durch den Einsatz von künstlicher Intelligenz (Artificial Intelligence, AI) bzw. generativer künstlicher Intelligenz (GenAI) verändert sich die Arbeitswelt aktuell rasant und wir stehen vor einem der größten Umbrüche seit der Industrialisierung. Insbesondere die neuen Fähigkeiten von GenAI im Umgang mit natürlicher Sprache schaffen neue Möglichkeiten für die Übernahme vielerlei Tätigkeiten. Neben Routinearbeiten kann AI nun auch Tätigkeiten von Wissensarbeiter*innen bis hin zu Manager*innen-Aufgaben übernehmen [1]. Gelegenheiten für den Einsatz reichen von AI-gestützter Entscheidungsfindung über Dienstleistungen mit Blick auf Kund*innenkontakt bis hin zu Programmier-Aufgaben. Gute Aussichten, könnten man sagen – doch sieht das jede*r so?

„Immer dann, wenn Veränderung kommt, gibt es Instabilität. Instabilität sorgt manchmal für Angst. Und genau das müssen wir in den nächsten Jahren wirklich angehen (...).“

*Ranga Yogeshwar,
Wissenschaftsjournalist und Autor*

Die Haltungen gegenüber GenAI sind ambivalent. Für die einen ist sie ein Hilfsmittel, das unterstützt und entlastet, für andere ist sie schon heute Treiber hin zu einer Verdichtung von Arbeitsabläufen, erhöhtem Stress und Auslöser verschiedener Ängste [2]. Eine Studie von Zhao et al. (2024)

[3] kommt beispielsweise zu dem Ergebnis, dass der dämpfende Effekt auf das individuelle Wohlbefinden, der sich aus dem Einsatz künstlicher Intelligenz ergibt, bei jungen Menschen, Männern, einkommensstarken Gruppen, hochqualifizierten Gruppen und Arbeitern im verarbeitenden Gewerbe stärker ausgeprägt ist als bei anderen.

Eines ist klar: GenAI ist gekommen, um zu bleiben und wird sich so selbstverständlich in unseren Arbeitsalltag integrieren wie damals das Smartphone in unser tägliches Leben. Doch wie können Unternehmen diese neue Technologie erfolgreich in ihre Geschäftsabläufe integrieren und gleichzeitig das Vertrauen der Belegschaft in ihre Anwendung stärken? Die Einführung und Nutzung von GenAI kann nur dann gelingen, wenn diejenigen mitgenommen werden, die auch täglich damit arbeiten sollen. Zunächst ist es daher wichtig, Ängste und Vorbehalte von Mitarbeitenden gegenüber GenAI ernst zu nehmen und ihre Bedenken zu verstehen.

Verborgene Ängste und Vorbehalte von Mitarbeitenden an die Oberfläche bringen

Vorbehalte von Arbeitnehmer*innen gegenüber GenAI können vielfältig sein und hängen oft von den spezifischen Anwendungsfällen und Kontexten ab. Häufige Bedenken sind folgende:

Oosthuizen (2022) [4] ermittelt, dass in der „vierten industriellen Revolution“ (4IR) STARA (*Smart Technology*,

Artificial Intelligence, Robotics and Algorithms) voraussichtlich ein Drittel der heute bestehenden Arbeitsplätze ersetzt wird. „Die größten gesellschaftlichen Auswirkungen werde Arbeitsplatzverlust sein“, liest es sich im Artikel „The blended future of automation and AI“ [5]. Bei der Häufigkeit solcher Aussagen und Prognosen ist es nicht verwunderlich, dass viele Arbeitnehmer*innen verunsichert sind oder gar einen Jobverlust fürchten. Des Weiteren sind sie besorgt, dass im Fokus der Unternehmen vor allem Kostensenkung und Personalabbau stehen. Es geht sogar so weit, dass Mitarbeitende den Einsatz von AI so sehr als Bedrohung ihres Jobs wahrnehmen, dass sie Wissen zurückhalten und nicht an die AI weitergeben [6].

Eine weitere Sorge von Mitarbeitenden ist der Verlust von Kontrolle. Dabei ist wahrgenommene Kontrolle, also „die Überzeugung von Personen, gewünschte Ergebnisse herbeiführen und unerwünschte Ereignisse vermeiden zu können“ [7] ein zentrales Motiv von Menschen. Wenn es um GenAI-Tools geht, ist die Angst vor dem Unbekannten zweifelsohne ein wichtiger Faktor, weshalb Arbeitnehmer*innen der Nutzung von GenAI kritisch gegenüberstehen. Allein der Gedanke daran, dass Algorithmen und maschinelles Lernen beispielsweise unsere DNA analysieren und Entscheidungen über unsere Gesundheit treffen können, ist für viele beunruhigend. Auch ist es aktuell nur bedingt nachvollziehbar, zu welchen Ergebnissen und Entscheidungsempfehlungen GenAI kommt, denn „den auf maschinellem Lernen basierenden KI-Algorithmen fehlt es an Transparenz“ [8]. Und das führt eben-

falls zu Unsicherheit. Die Angst vor dem Verlust der Kontrolle über die eigenen Daten und Entscheidungen ist ebenso verständlich wie die Angst vor Überwachung und davor, „gläsern“ zu werden. Darüber hinaus befürchten Arbeitnehmer*innen, im Zuge der Einführung von AI benachteiligt oder ungerecht behandelt zu werden (z. B. bei Auswahl- oder Bewertungsverfahren), denn „viele der gleichen Vorurteile, die in der breiten Öffentlichkeit existieren, finden ihren Weg in KI-Systeme, die Mitarbeiter verwalten“ [9]. Auch ist derzeit noch ungeklärt, wer die Haftung für algorithmisches Fehlverhalten übernimmt (digitale Verantwortungslücke) [10].

Zudem haben Arbeitnehmende vermehrt das Gefühl, mit GenAI zu konkurrieren. Die AI trifft logischere Entscheidungen, unterliegt kognitiven Verzerrungen (Biases) nicht in dem Ausmaß wie Menschen, ist objektiv und kann eine Vielzahl an Daten bei ihren Berechnungen miteinbeziehen. Darüber hinaus verzichtet der Kommunikationsstil von GenAI auf jegliche (Be-)Wertung oder emotionale Färbung, sodass das Risiko, im Dialog gekränkt oder verletzt zu werden, wegfällt. Da wirkt die Frage, wozu es den Menschen in Bezug auf Arbeit überhaupt noch braucht, nur allzu berechtigt. Menschen fangen an, sich in ihrem Menschsein in Frage zu stellen: Was bin ich als Mensch noch wert, wenn die AI sowieso alles besser kann? Werde ich in Zukunft vielleicht nicht mehr gebraucht und obsolet? Zwar birgt GenAI großes Potenzial, die Arbeit von Arbeitnehmer*innen anspruchsvoller und sinnvoller zu gestalten, indem sie weniger sinnvolle Aufgaben übernimmt und die Fähig-

keiten von Menschen sogar noch erweitert. Es kann aber auch genau das Gegenteil der Fall sein. Und zwar, dass die Arbeit weniger sinnvoll wird, indem GenAI die Autonomie von Arbeitnehmer*innen einschränkt, neue langweilige Aufgaben schafft und der Mensch – beispielsweise als Qualitätsprüfer*in – zur*m Sekretär*in wird. Arbeitnehmer*innen fühlen sich bedroht, die AI würde das „Denken übernehmen“ und sie wären ihr schutzlos ausgeliefert. Und obwohl bei der Entscheidungsfindung algorithmische Entscheidungen immer effizienter werden und die Qualität der Entscheidungen von Menschen häufig übertreffen, weisen Menschen bewusst oder unbewusst eine Abneigung gegen Algorithmen auf; ein Phänomen, das *Algorithmus-Aversion* genannt wird. [11]

Was Beschäftigte über (Gen)AI in ihrem direkten Arbeitsumfeld wirklich denken, ist bislang nur rudimentär erforscht und die bisherigen Ergebnisse sind widersprüchlich. Während eine Studie von Pfeifer (2023) [12] herausstellte, dass sich Arbeitnehmer*innen eigentlich recht gut über AI informiert fühlen und keine Ängste hatten, kam eine andere Studie von Brougham & Haar (2018) [13] zu dem Ergebnis, dass ein höheres STARA-Bewusstsein in einem negativen Zusammenhang mit organisatorischem Engagement und Karrierezufriedenheit steht und in einem positiven Zusammenhang mit Fluktuationsabsichten, Zynismus und Depression.

Wie Organisationen Mitarbeitende unterstützen können, ihre Ängste und Vorbehalte gegenüber GenAI zu überwinden

Mitunter am wichtigsten ist es, dass sich Mitarbeitende mit all ihren Ängsten ernst genommen fühlen. Damit sie ihre Sorgen und Vorbehalte gegenüber GenAI-Tools überwinden, ist es entscheidend, dass Organisationen Vertrauen aufbauen. Unternehmen müssen transparent kommunizieren und sicherstellen, dass ihre Mitarbeiter*innen verstehen, wie die Technologie funktioniert, welche Vorteile sie bietet und wie sie genutzt werden kann.

Laut der Studie von Pfeifer (2023) fühlten sich die Beschäftigten zwar gut informiert, was das Thema künstliche Intelligenz betrifft, jedoch nicht durch ihre Arbeitgeber [12]. Daher ist es wichtig, dass Unternehmen ihre Mitarbeitenden selbst aufklären und sie über aktuelle Entwicklungen und Trends im Bereich AI auf dem Laufenden halten, um ihnen Klarheit und Sicherheit zu vermitteln. Sie sollten transparent kommunizieren, welche Veränderungen innerhalb der Organisation in naher und ferner Zukunft geplant sind, und die Beschäftigten frühzeitig einbeziehen.

Unterstützung seitens des Top-Managements ist einer der Schlüsselfaktoren für eine erfolgreiche Einführung von AI [14], denn dies fördert das Vertrauen von Mitarbeitenden enorm. Entscheidend ist auch, ob Führungskräfte ein solides Verständnis von AI haben, sich kontinuierlich weiterbilden und mit der Technologie Schritt halten. Dies wiederum ermöglicht es, eine AI-Einführung gut zu unterstützen und zu fördern. [15]

Bewusste Wahl von Kommunikationsstil und Narrativ

Dafür sollten sie sich die Frage stellen, welches Narrativ sie wählen, wenn sie über (Gen)AI sprechen. Ist es die unbestechliche, stets ansprechbare „Intelligenz“, die nie um eine Antwort verlegen ist und den Menschen als Arbeitskraft den Rang ablauft? Oder sprechen sie von ihr als Partnerin, Kollaboratorin, Mitarbeitende, Assistentin, Kompagnon oder digitale Gefährtin?

Der Begriff „Artificial Intelligence“ ist in gewisser Weise auch irreleitend, da sie weder künstlich noch intelligent ist und auch nicht mit der menschlichen Intelligenz verglichen werden kann. Akata et al. (2020) führen daher den Begriff „hybride Intelligenz“ (HI) ein. Denn sie sehen die AI als Ergänzung der menschlichen Intelligenz durch kollaborative, adaptive, verantwortungsvolle und erklärbare künstliche Intelligenz. Sie definieren HI als eine Kombination aus menschlicher und maschineller Intelligenz, die den menschlichen Intellekt und die menschlichen Fähigkeiten ergänzt – anstatt sie zu ersetzen – und Ziele erreicht, die bisher weder von Menschen noch von Maschinen erreicht werden konnten. [16]

Ehrliche und transparente Kommunikation im Dialog

Insbesondere durch ehrliche Kommunikation können Unternehmen Ängste ihrer Belegschaft ab- und Vertrauen aufbauen. Es ist wichtig, offen und ehrlich über alle Belange von GenAI zu sprechen und dabei zu vermeiden, ein Schwarz-Weiß-Bild zu zeichnen, sondern die Technologie

in ihrer Komplexität anzuerkennen und Chancen sowie Risiken offenzulegen. Die Belegschaft möchte schließlich wissen, an welchen Stellen GenAI eingesetzt werden kann, um sie bei ihrer Arbeit zu unterstützen oder Kunden Mehrwert zu liefern. Es ist wichtig, die Mitarbeitenden auch zu informieren, welche neuen Aufgaben sich dadurch konkret für sie ergeben, und gemeinsam zu eruieren, ob sich im Zuge dessen vielleicht spannendere, anspruchsvollere Aufgaben auf tun.

Bezüglich interner Kommunikation sind laut Rudko et al. (2021, s. [17]) kritische Denker*innen und Hinterfrager*innen gefragt, die gleichzeitig neuen Technologien optimistisch gegenüberstehen. Ihre Studie kommt zu dem Ergebnis, dass die zweifelnden Optimist*innen („doubtful optimist“) die wichtigste Gruppe ist, um organisatorische Trends zu setzen und die Skeptiker*innen positiv zu beeinflussen.

Um ein vertrauensvolles Miteinander zu fördern, ist es zudem ratsam, nicht nur zu informieren, sondern sich mit der Belegschaft auszutauschen, beispielsweise über Trends oder mögliche Zukunftsszenarien. (Zeit-)Räume für Dialog sind entscheidend, um offene Diskussionen anzuregen und zu ermöglichen, dass Mitarbeitende u. a. auch ihre Bedenken äußern können. Anonyme Umfragen können helfen, ihre Einschätzung zu AI abzufragen, um darauf eingehen zu können.

Bezüglich der Chancen und Risiken sollten Unternehmen Risiken ehrlich benennen und gleichzeitig insbesondere die Vorteile und Möglichkeiten von GenAI herausstellen. Eine Studie von Xu et al. (2023) ergab, dass die Wahrneh-

mung von AI-Möglichkeiten signifikant positiv mit dem Wohlbefinden der Mitarbeiter am Arbeitsplatz (WWB) korrelierte [18]. Denn für Arbeitnehmer*innen bietet die Nutzung von GenAI viele Vorteile. GenAI kann zur Assistentin werden, die bei der Ideenfindung hilft, Entwürfe erstellt, E-Mails verfasst und Texte vorbereitet. Zusätzlich kann sie große Datenmengen schnell verarbeiten und Telefonate sowie Meetings verschriftlichen und dokumentieren. Dadurch können Mitarbeitende ihre Produktivität steigern und ihre Arbeitslast verringern. GenAI kann zu mehr Flexibilität und Autonomie, Kreativität und Innovation am Arbeitsplatz führen und letztendlich dazu beitragen, Stress und Arbeitsbelastung zu reduzieren und das Wohlbefinden zu steigern.

Job-Sicherheit vermitteln

Die Anspannung, eines Tages den Job zu verlieren und von AI-Anwendungen abgelöst zu werden, ist groß. Daher ist es wichtig, transparent darzulegen, welche Aufgaben genau von GenAI übernommen werden könnten und welche Tätigkeiten voraussichtlich neu entstehen. Organisationen sollten darüber hinaus nicht müde werden zu betonen, dass GenAI dazu eingesetzt werden soll, Menschen zu unterstützen und nicht zu ersetzen (vorausgesetzt, dass das der Wahrheit entspricht).

Auch evidenzbasierte Vorhersagen können hier Sicherheit geben. Im Artikel „The fourth industrial revolution“ (Oosthuizen, 2022) [4] beispielsweise wird prognostiziert, dass bis 2025 85 Millionen Arbeitsplätze durch eine Verschiebung

der Arbeitsteilung zwischen Mensch und Maschine verdrängt werden könnten, während 97 Millionen neue Rollen entstehen könnten, die besser an die neue Arbeitsteilung zwischen Mensch, Maschine und Algorithmus angepasst sind. Eine Deloitte-Studie [19] stellt heraus, dass AI in Unternehmen mit unterschiedlichen Zielsetzungen eingesetzt wird: um Prozesse effizienter zu gestalten (28 %), um bestehende Produkte und Dienstleistungen zu verbessern (25 %), um neue Produkte und Dienstleistungen zu schaffen (23 %), um die Entscheidungsfindung zu verbessern (21 %), um Kosten zu senken (20 %). Die Reduzierung des Personalbestands wurde am seltensten genannt (11 %).

Gefühl von Kontrolle und Sicherheit herstellen

Die subjektive Wahrnehmung von Kontrolle spielt eine entscheidende Rolle für das Wohlbefinden von Menschen, ihr Selbstwertgefühl und die emotionale Stabilität. Selbst dann, wenn objektiv betrachtet keine Kontrolle vorhanden ist. [20] Diese sogenannte „Illusion von Kontrolle“ [21] kann sogar wichtiger sein als die tatsächliche Kontrolle. In Organisationen besteht die Möglichkeit, das Kontrollerleben von Menschen positiv zu beeinflussen, indem Dinge vorhersehbar und erklärbar gemacht werden [22]. Daher ist eine offene und transparente Kommunikation so wichtig.

Dinge, die Arbeitnehmer*innen bei der anstehenden Einführung von AI in ihrem Unternehmen helfen können, um Kontrollverlust und ethische Bedenken zu mildern,

sind zum einen eine transparente Unternehmens-Kommunikation, gefühlte Unterstützung durch das Management, Schulungen, Ethikbüro(s) sowie Berichtsmechanismen [23]. Zudem braucht es eine Rechtsgrundlage und strenge Anforderung an die Erklärbarkeit von Ergebnissen von GenAI.

Die Unverzichtbarkeit des Menschen in Zusammenarbeit mit GenAI herausstellen

Die meisten Unternehmen möchten Produkte und Kundenerlebnisse durch GenAI verbessern, nicht menschliche Interaktion ersetzen. Menschliche Mitarbeitende werden weiterhin benötigt. Sie sind die unabdingbare Kontrollinstanz, die prüft, ob Anwendungsfälle als auch Ergebnisse von GenAI angemessen sind. Denn GenAI-Applikationen wissen nicht, was sie „sagen“ und warum. AI-Modelle kennen keine „Wahrheit“ und sind nicht in der Lage, zwischen richtig und falsch zu unterscheiden. Kritische Denker werden daher dringend gebraucht!

„Habe den Mut, dich deines eigenen Verstandes zu bedienen!“

Immanuel Kant

In diesem Sinne: Wage es, GenAI zu widersprechen!

Statt ein Konkurrenz-Denken zu schüren, sollten Unternehmen nach einem harmonischen Miteinander zwischen

Mensch und GenAI streben – im Sinne eines Narrativs von Unterstützung und Kooperation. Der Begriff „Human-AI Symbiosis“ beispielsweise betont, dass Interaktionen zwischen Menschen und AI dazu führen können, dass beide Parteien im Laufe der Zeit intelligenter werden [24]. In seinem Artikel betont Jarrahi (2018) die Komplementarität von Menschen und AI und untersucht, wie beide in ambiguen Situationen ihre individuellen Stärken in organisatorische Entscheidungsprozesse einbringen können. Er kommt zu dem Ergebnis, dass sich beide Parteien mit ihren Fähigkeiten ergänzen. Solch ein Narrativ legt nahe, dass AI-Systeme entwickelt werden sollten, um die menschlichen Beiträge zu ergänzen, anstatt sie zu ersetzen.

Schulen und Empowern von Mitarbeitenden

Um Beschäftigte im Umgang mit GenAI rechtzeitig so zu befähigen, dass sie selbst zu Mitgestalter*innen werden können, sollten Unternehmen auf Weiterbildung und Qualifizierung setzen und unternehmensweit AI-Kompetenzen ausbilden. Dazu braucht es zunächst eine Bereitschaft seitens der Arbeitnehmer*innen, fortlaufend zu lernen und sich auf neue Entwicklungen einzulassen.

Durch Schulungsprogramme können Organisationen nun sicherstellen, dass ihre Mitarbeiter*innen die Potenziale und Grenzen von GenAI verstehen und in der Lage sind, sie effektiv einzusetzen.

Eine quantitative Studie in multinationalen Unternehmen (MNCs) [25] ergab, dass kritische Fähigkeiten, die Mit-

arbeitende in Zusammenhang mit AI benötigen, folgende sind: Datenanalyse; digitale und komplexe kognitive Fähigkeiten; Entscheidungsfindung und kontinuierliches Lernen. Insbesondere auch soziale Kompetenzen sollten unbedingt weiter ausgebildet werden, sowie Führungsverhalten, Kommunikationskompetenz und im Allgemeinen zwischenmenschliche Fähigkeiten.

Eine Schlüsselkompetenz ist zweifelsohne der Umgang mit Sprache und klare Kommunikation. Es geht darum, effektiv mit GenAI-Anwendungen zu interagieren, um möglichst zielgerichtet zu bestmöglichen Ergebnissen zu kommen.

Raum für aktiven Austausch und Mitgestaltung

Angesichts der rasanten Entwicklung von GenAI-Technologien ist es für Unternehmen wichtig, sowohl ihre Stärken als auch Grenzen im Blick zu behalten. Dazu sollten sie ihre Arbeitnehmer*innen ermutigen, aktiv mitzudenken, zu hinterfragen und mitzugestalten. Die Anwendung von GenAI ist von Natur aus interdisziplinär. Daher ist es wichtig, sich über Abteilungs- und sogar Unternehmensgrenzen hinweg kontinuierlich auszutauschen, Erfahrungen und Ideen zu teilen und voneinander zu lernen.

Indem Organisationen explizit (Zeit-)Räume für Austausch zur Verfügung stellen, können sie dazu beitragen, dass Mitarbeitende sich ermutigt fühlen, zu experimentieren, sich einzubringen und Ideen zur Wertschöpfung durch GenAI beizutragen. Dies fördert nicht nur das Engagement von Mitarbeitenden, sondern kann dabei helfen, Ohn-

machtsgefühlen und Angst vor dem Unbekannten entgegenzuwirken.

Janina Lerner ist Organisationsentwicklerin mit dem Schwerpunkt Agilität. Sie hat einen Background in Kommunikationsdesign (Dipl.) und Wirtschaftspsychologie (M. Sc.) und ist freiberufliche Dozentin an verschiedenen Hochschulen. Menschengerechtes Arbeiten ist ihr ein großes Anliegen. Daher möchte sie gerne dazu beitragen, dass die „Wissenschaft vom Erleben und Verhalten des Menschen“ und die aus der Forschung resultierenden Handlungsempfehlungen in die Praxis und Arbeitswelt Einzug halten.

janina.lerner@codecentric.de

Referenzen

- [1] Keusch (2024), Künstliche Intelligenz könnte bald auch Manager abschaffen, Neue Züricher Zeitung
- [2] Gergs, Schatilow, Langes, Kämpf (2023), Human Friendly Automation – Arbeit und Künstliche Intelligenz neu denken, Frankfurter Allgemeine Buch
- [3] Zhao, Y., Yin, D., Wang, L., & Yu, Y. (2024). The rise of artificial intelligence, the fall of human well-being?. International Journal of Social Welfare
- [4] Oosthuizen, R. M. (2022). The fourth industrial revolution—smart technology, artificial intelligence, robotics and algorithms: Industrial psychologists in future workplaces. Frontiers in artificial intelligence
- [5] Khogali, H. O., & Mekid, S. (2023). The blended future of automation and AI: Examining some long-term societal and ethical impact features. Technology in

Society

- [6] Arias-Pérez, J., & Vélez-Jaramillo, J. (2022). Understanding knowledge hiding under technological turbulence caused by artificial intelligence and robotics. *Journal of Knowledge Management*.
- [7] Fritsche, I., Jonas, E., & Frey, D. (2006). Kontrollwahrnehmungen und Kontrollmotivation. In H.-W. Bierhoff & D. Frey (Hrsg.), *Handbuch der Sozialpsychologie und Kommunikationspsychologie*. Göttingen: Hogrefe
- [8] Maclure, J. (2021). AI, explainability and public reason: The argument from the limitations of the human mind. *Minds and Machines*
- [9] Liang, T. P., Robert, L., Sarker, S., Cheung, C. M., Matt, C., Trenz, M., & Turel, O. (2021). Artificial intelligence and robots in individuals' lives: how to align technological possibilities and ethical issues. *Internet Research*
- [10] Königs, P. (2022). Artificial intelligence and responsibility gaps: what is the problem?. *Ethics and Information Technology*
- [11] Mahmud, H., Islam, A. N., Ahmed, S. I., & Smolander, K. (2022). What influences algorithmic decision-making? A systematic literature review on algorithm aversion. *Technological Forecasting and Social Change*.
- [12] Pfeiffer, S. (2024). KI als Kollegin (KIK) – Repräsentative Beschäftigtenbefragung zu Künstlicher Intelligenz am Arbeitsplatz. In *Künstliche Intelligenz, Mensch und Gesellschaft: Soziale Dynamiken und gesellschaftliche Folgen einer technologischen Innovation* (pp. 15-40). Wiesbaden: Springer Fachmedien Wiesbaden
- [13] Brougham, D., & Haar, J. (2018). Smart technology, artificial intelligence, robotics, and algorithms (STARA): Employees' perceptions of our future workplace. *Journal of Management & Organization*
- [14] Hamm, P., & Klesel, M. (2021). Success Factors for the Adoption of Artificial Intelligence in Organizations: A Literature Review. *AMCIS 2021 Proceedings*, Article 10
- [15] Bhargava, A., Bester, M., & Bolton, L. (2021). Employees' perceptions of the implementation of robotics, artificial intelligence, and automation (RAIA) on job satisfaction, job security, and employability. *Journal of Technology in Behavioral Science*
- [16] Akata, Z., Balliet, D., De Rijke, M., Dignum, F., Dignum, V., Eiben, G., ... & Welling, M. (2020). A research agenda for hybrid intelligence: augmenting human intellect with collaborative, adaptive, responsible, and explainable artificial intelligence. *Computer*
- [17] Rudko, I., Bashirpour Bonab, A., & Bellini, F. (2021). Organizational structure and artificial intelligence. Modeling the intraorganizational response to the ai contingency. *Journal of Theoretical and Applied Electronic Commerce Research*
- [18] Xu, G., Xue, M., & Zhao, J. (2023). The Relationship of Artificial Intelligence Opportunity Perception and Employee Workplace Well-Being: A Moderated Mediation Model. *International Journal of Environmental Research and Public Health*
- [19] Deloitte (2020) „Thriving in the era of pervasive AI: Deloitte's state of AI in the enterprise, 3rd edition,“ *Deloitte Insights*

- [20] Lermer, E., & Hudecek, M. (2022). Unsicherheit: Globale Herausforderungen psychologisch verstehen und bewältigen. Ernst Reinhardt Verlag.
- [21] Johnson, J. E. (1973). Effects of accurate expectations about sensations on the sensory and distress components of pain. *Journal of Personality and Social Psychology*
- [22] Frey, D., & Jonas, E. (2002). Die Theorie der kognitiven Kontrolle. In D. Frey, & M. Irle (Hrsg.), *Theorien der Sozialpsychologie: Band II: Motivations, Selbst- und Informationsverarbeitungstheorien* (2., vollst überarb. u. erw. Aufl., Band 3, Bern: Hans Huber.
- [23] Kelley, S. (2022). Employee perceptions of the effective adoption of AI principles. *Journal of Business Ethics*
- [24] Jarrahi, M. H. (2018). Artificial intelligence and the future of work: Human-AI symbiosis in organizational decision making. *Business horizons*
- [25] Jaiswal, A., Arun, C. J., & Varma, A. (2022). Rebooting employees: Upskilling for artificial intelligence in multinational corporations. *The International Journal of Human Resource Management*

Mehr als nur Ergebnisse

Warum Reproduzierbarkeit in Machine
Learning der Goldstandard ist

von Denis Stalz-John

Seeanemonen

LEBEN WELTWEIT IN TIEFEN OZEANEN UND
FLACHEN KÜSTENGEWÄSSERN

RAUBTIERE. FRESSEN
KRABBen, WEICHTIERE UND
SOGAR KLEINE FISCHÉ

GRÖSSE
< 10 CM

KRALLEN SICH MIT IHRER FUSSSCHEIBE AN
HARTEN OBERFLÄCHEN FEST

Wie Reproduzierbarkeit in Machine-Learning(ML)-Projekten die Zuverlässigkeit steigert und Fehler vermeidet – ein Schlüssel zum Erfolg von KI in der Praxis.

Die Implementierung künstlicher Intelligenz in Produktionsabläufen verspricht immense Vorteile bei der Effizienzsteigerung und Kostensenkung. Das erkannte auch ein mittelständisches Unternehmen, das sich entschied, KI-Modelle zur Qualitätskontrolle an seinen Produktionsanlagen einzusetzen. Die Vision war klar: Kameras sollten mithilfe maschineller Lernmethoden fehlerhafte Produkte erkennen und aussortieren. Die praktische Umsetzung stellte sich allerdings als anspruchsvoller heraus als zuerst angenommen. Ein Pilotmodell wurde für eine Produktionslinie eingeführt und zeigte binnen Wochen beeindruckende Ergebnisse. Dieser initiale Erfolg ebnete den Weg dafür, das Modell auf eine zweite Produktionslinie zu übertragen. Diese brachte allerdings eine Reihe von Variablen mit sich – die Anordnung der Kameras, die Lichtverhältnisse und der Aufbau der Linie selbst –, die unterschiedlich zum originalen Setup waren. Dadurch bedurfte das zweite KI-Modell feinabgestimmter Anpassungen, die trotz der Erweiterung des Datensatzes und des intensiven Hyperparameter-Tunings nicht zum gewünschten Erfolg führten.

Als sich dann die Serie der gefertigten Produkte an der ersten Linie änderte, stand das Unternehmen vor der nächsten Herausforderung. Die anfänglichen Trainingskonfigurationen und verwendeten Datensätze waren nie protokolliert worden, was zu erneuten zeitintensiven und kostspieligen Anpassungsversuchen führte. All diese Bemü-

hungen konnten jedoch nicht die gewünschte Qualität liefern. Schließlich musste das Unterfangen als gescheitert angesehen werden, und die Geschäftsleitung sah sich gezwungen, das KI-Projekt stillzulegen.

Dieser Fall verdeutlicht, wie essenziell die Reproduzierbarkeit in ML-Experimenten ist. Ohne sie gibt es keine Möglichkeit, Modelle kontrolliert zu skalieren, zu übertragen oder anzupassen. Reproduzierbarkeit ist somit kein Luxus, sondern eine Notwendigkeit für die erfolgreiche Umsetzung von KI in der Praxis.

Die Umsetzung

Die Realisierung reproduzierbarer ML-Trainings ist eine Aufgabe, die eine durchdachte und methodische Vorgehensweise erfordert. Den Anfang macht eine umfassende Analyse der Entwicklungsumgebung. Hierbei ist es unerlässlich, sämtliche Elemente der Software, inklusive aller Versionen der genutzten Bibliotheken und Programme, zu erfassen. Bei der Verwendung von Python, der meistgenutzten Programmiersprache im KI-Umfeld, müssen alle verwendeten Paketversionen detailliert aufgezeichnet werden. Selbst eine geringe Abweichung in einer Version kann zu unterschiedlichen Ergebnissen führen. Paketmanager wie Poetry oder Pip haben zwar bereits Mechanismen zur Versionierung eingebaut; allerdings werden diese oft nicht beim Starten eines Experiments aktiviert, um eine Liste der Versionen zu persistieren.

Des Weiteren muss der zum Training genutzte Quellcode abgelegt werden. Der Gebrauch eines Versionierungstools

wie Git ermöglicht es, nicht nur den aktuellen Code zu sichern, sondern auch alle Änderungen, die über die Zeit hinweg vorgenommen wurden, nachzuvollziehen. Der Schlüssel hierbei ist der Commit-Hash, der ein Abbild des Codes zum Zeitpunkt des Experimentes darstellt.

Bei der Konfiguration des Trainings und Modells gibt es eine Vielzahl von Parametern. Von der Learning Rate bis hin zu den Einstellungen des Optimizers – all diese Parameter haben Einfluss auf das Trainingsergebnis. Deshalb gilt auch in diesem Fall eine Dokumentationsnotwendigkeit. Ein Aspekt, der oft übergangen wird, ist die Sicherung und Versionierung der Daten, da diese die Grundlage von ML-Trainings bilden. Hierbei hängt die Umsetzung stark von der Art und Menge der vorliegenden Daten ab und braucht somit ein individuelles Vorgehen. Einen weiteren Baustein für die Reproduzierbarkeit stellt die eindeutige Identifikation jedes einzelnen Experiments dar. Eine solche Identifikation ermöglicht es, schnell und unmissverständlich auf vergangene Experimente Bezug zu nehmen und sie zu reproduzieren. Hierbei können bereits simple Mechanismen wie der Einsatz zeitbasierter Hashes oder eindeutige alphanumerische Strings die gewünschte Eindeutigkeit schaffen. All diese Elemente müssen in ein nahtloses System integriert werden, das die vollständige Automatisierung dieser Prozesse erlaubt. Dadurch minimiert man die Wahrscheinlichkeit von menschlichen Fehlern und schafft eine zuverlässige Grundlage für wiederholbare und nachvollziehbare ML-Experimente. Plattformen wie MLflow [1] oder auch spezialisierte Experiment-Tracking-Tools

spielen eine unterstützende Rolle dabei, den Überblick zu behalten und die Reproduzierbarkeit in der praktischen Umsetzung zu gewährleisten.

Ausblick

Die Wichtigkeit der Reproduzierbarkeit in ML-Experimenten wird in der Zukunft weiter zunehmen. Sie bildet das Fundament, auf dem vertrauenswürdige und ethische KI-Technologien aufbauen müssen. Gerade in der Forschung sind reproduzierbare Ergebnisse unerlässlich, um die Wissenschaft voranzubringen und echten Fortschritt zu ermöglichen.

Derzeit befindet sich die KI-Entwicklung noch in einem Stadium, in dem die Reproduzierbarkeit oft als zweitrangig betrachtet wird. Doch mit zunehmender Reife der Technologien und dem wachsenden Bewusstsein für ihre Bedeutung wird sich dieser Ansatz ändern müssen. Große Sprachmodelle (LLMs) stellen dabei eine besondere Herausforderung dar, aber auch eine Chance, die Wichtigkeit von Nachvollziehbarkeit und Reproduzierbarkeit zu etablieren. Denn auch beim Benchmarking bereits trainierter Open-Source-Modelle gilt die dargelegte Vorgehensweise. Ein systematischer Ansatz zur Sicherstellung der Reproduzierbarkeit ist somit keine Option, sondern eine dringende Notwendigkeit. Erst wenn es Standard wird, jedes Experiment mit all seinen Facetten – von der Codeversion über Parameter, Daten und Konfigurationen bis zur Systemumgebung – akribisch zu dokumentieren und zu sichern, kann von einer vertrauensvollen Arbeit mit KI gesprochen werden. Die Reproduzierbarkeit ist der Schlüssel dazu,

KI-Modelle zuverlässig in die Praxis zu integrieren und so ihren vollen Nutzen zu entfalten.

*Denis Stalz-John spezialisiert sich auf die Bereiche Computer Vision, Semantische Segmentierung, Object Detection und Deep Learning. Sein beruflicher Einstieg erfolgte beim Corporate Research der Robert Bosch GmbH im Bereich Fahrer-Assistenzsysteme und autonomes Fahren. Seit 2018 arbeitet er als Data Scientist bei der codecentric AG.
denis.stalz-john@codecentric.de*

Referenzen

[1] <https://mlflow.org/>

Gut integriert

Wie GPT und iPaaS
Geschäftsmodelle transformieren

von **René Rother** und **Kamil Braschke**



Flamingozunge

LEBT IN DEN TROPISCHEN GEWÄSSERN DES
WESTLICHEN ATLANTIKS

ERNÄHRT SICH VOM
LEBENDEN GEWEBE DER
WEICHKORALLEN, AUF DENEN
SIE LEBT

GRÖSSE
1,8-3,5 CM

DIE MINIMALE AUFGEZEICHNETE TIEFE
BETRÄGT 0,3 M UND DIE MAXIMALE 90 M

Wir zeigen, wie die Kombination aus großen Sprachmodellen und Integrationsplattformen Geschäftsprozesse effizienter gestaltet und zu einer modernen Unternehmensautomatisierung führen kann.

Im heutigen schnelllebigen digitalen Zeitalter ist die Integration fortschrittlicher Technologien in Geschäftsprozesse unerlässlich für den Erfolg und die Wettbewerbsfähigkeit von Unternehmen. Durch die Wahl der richtigen Technologien können interne Prozesse entschlackt, beschleunigt und zugänglicher gemacht werden. Dadurch ermöglicht eine Verbindung modernster Ansätze oftmals eine nachträgliche Ausrichtung der Prozesse mit starkem Automatisierungspotential. Die Autoren haben kürzlich auf dem Interior-KI-Summit 2024 einen Vortrag für die Möbelbranche gehalten, der darstellte, wie sich Integrations-Technologien (z. B. Anypoint von MuleSoft) mit großen Sprachmodellen (auch bekannt als „LLMs“ bzw. „Large Language Models“, wie z. B. GPT-4 von OpenAI) verbinden lassen. Dies ermöglicht nicht nur, dass Menschen in natürlicher Sprache mit komplexen Systemen interagieren können, sondern ermächtigt die Sprachmodelle auch dazu, Systeme und Prozesse auf Wunsch autark zu steuern. In diesem Artikel geben die Autoren zunächst eine grobe Übersicht zu Integrationstechnologien und ihren Vorteilen. Danach werden die Anbindungsmöglichkeiten von LLMs an Bestandssoftware anhand der Möglichkeiten von GPT-4 erklärt, um zum Schluss die Symbiose aus Integrationsplattform und Sprachmodell anhand eines Use Case der Möbelbranche zu veranschaulichen.

Ein kleiner Einblick in Integrationsplattformen

In der modernen Geschäftswelt steht die digitale Integration – das heißt die Verbindung verschiedener IT-Systeme und Anwendungen innerhalb eines Unternehmens und mit externen Partnern – im Mittelpunkt der digitalen Transformation. Während diese Integration zahlreiche Vorteile bietet, bringt sie auch signifikante Herausforderungen mit sich, die Unternehmen bewältigen müssen. Dabei kommt es nicht auf die Größe des Unternehmens an, denn Legacy-Systeme, intern selbst geschriebene Anwendungen oder externe Schnittstellen, die Komplexität und die Zusammenarbeit mit den einzelnen Abteilungen bedeuten, erfordern sehr viel Kommunikation und Austausch.

Mit der wachsenden Bedeutung von iPaaS (Integration Platform as a Service) als Cloud- oder On-Premises-basierter Integrationslösung lassen sich viele dieser Herausforderungen effektiver angehen. iPaaS erleichtert es Unternehmen, durch vorgefertigte Konnektoren und Integrationstemplates rasch verschiedene Anwendungen und Systeme miteinander zu verknüpfen. Dies führt nicht nur zu einer erheblichen Zeitersparnis bei der Implementierung neuer Systeme (z. B. für die Integration großer Sprachmodelle), sondern auch zu einer Reduzierung der Kosten, da teure Investitionen in eigene Hardware und Software entfallen. Darüber hinaus lassen sich bereits vorhandene Systeme vereinheitlichen und können für die Folgeprozesse standardisiert weiterverwendet werden. **Abb. 1** veranschaulicht eindrucksvoll, wie Integration, Vereinfachung und Wiederverwendbarkeit

hervorragend miteinander harmonisieren. Während auf der linken Seite eine herkömmliche Integration dargestellt ist, die für jede Verbindung (hier: über Mobile API oder über Web App API) eine separate Entwicklung erfordert, zeigt die rechte Seite die Effizienz vereinfachter, modularer Applikationen. Diese können mühelos an jedes System angeschlossen werden, das die entsprechenden Informationen benötigt.

Zudem sollte man nicht außer Acht lassen, dass mit zentralisierten Datenverwaltungen und automatisierten Datenflüssen die Komplexität reduziert wird. Dadurch nimmt

die Datenqualität und -konsistenz zu und die Skalierbarkeit wird gefördert. Denn einheitliche und standardisierte Formate lassen sich deutlich leichter verwalten und wiederverwenden.

Trotz dieser Vorteile müssen sich Unternehmen mit der Komplexität der verschiedenen IT-Systeme und ihrer Schnittstellen auseinandersetzen und stetig den Austausch zwischen den Abteilungen/Projekten aufrechterhalten. Die fortwährend schnelle Entwicklung der Technologie verlangt kontinuierliche Aktualisierungen und Anpassungen der Integrationsysteme. Zudem ist die Datensicherheit und

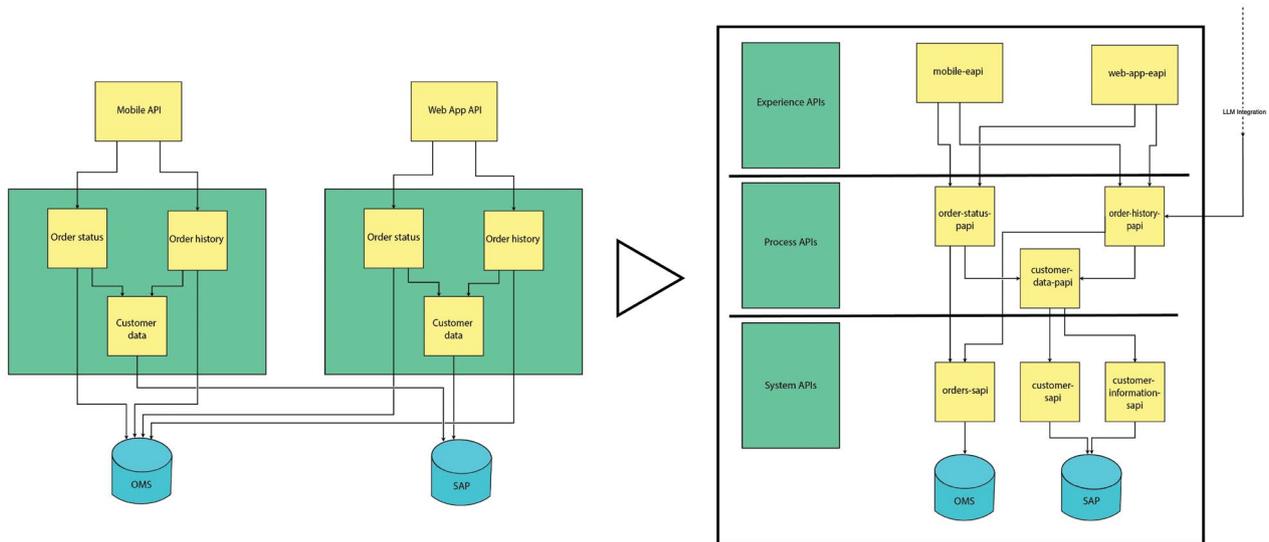


Abb. 1: Vergleich einer Systemarchitektur ohne (links) und mit (rechts) iPaas. Zusätzlich wird rechts bereits die Erweiterung um eine LLM-Integration abgebildet (gestrichelte Linie).

-qualität eine beständige Sorge, da sensible Daten geschützt und standardisiert werden müssen, um die Unternehmen vor Datenverlust und Cyberattacken zu schützen.

Die digitale Integration stellt eine bedeutende und wegweisende Entwicklung dar, die es in Zukunft ermöglichen wird, eigene Systeme und deren vielfältige Datenbestände effizienter zu nutzen. Vor dem Hintergrund der Flexibilität bildet eine solche Landschaft ein ideales Fundament, um in der Interaktion mit großen Sprachmodellen zu stehen. Die Spezifizierung von Schnittstellen gestaltet sich dabei unkompliziert und kann leichter wiederverwendet werden, während die Skalierungsmöglichkeiten beträchtlich sind.

Funktionsaufrufe mit großen Sprachmodellen

In der Theorie ist es jedem Sprachmodell möglich, durch genügend programmatische Erweiterung bzw. Verarbeitung mit externen Systemen zu interagieren. Dazu werden einem Sprachmodell entweder die Funktionen bzw. Methoden, die es aufrufen kann, im System-Prompt übergeben oder es findet eine Form von Verschlagwortung der gewünschten Methoden und Parameter im System Prompt statt, die später wieder vom Programm aufgeschlüsselt wird. Unabhängig vom gewählten Ansatz muss der vom Sprachmodell generierte Output programmatisch verarbeitet und die Strings, die die Methodenaufrufe repräsentieren, aus dem Output extrahiert werden, um dann über das Programm bzw. Skript die eigentlichen Methodenaufruf durchzu-

führen. Dieser Ansatz krankt jedoch an einer starken Halluzinationsanfälligkeit (sprich: nicht wahrheitsgetreuem Output), da selbst kleinere Änderungen des Outputs im Bezug zu Methoden- oder Parameternamen zu Fehlern in der späteren Verarbeitung führen oder die Methodenaufrufe erst gar nicht als solche erkannt werden.

Aus diesen Gründen ist es wesentlich zuverlässiger und daher auch sicherer, wenn das verwendete Sprachmodell eine sogenannte *Function-Calling*-Fähigkeit besitzt. Diese beschreibt das Vermögen der Sprachmodelle, spezifische Aufgaben durch den Aufruf interner oder externer Funktionen bzw. Methoden und Tools auszuführen. Grundsätzlich handelt es sich auch hierbei zunächst nur um einen generierten String, der weiterverarbeitet wird, da Sprachmodelle letztlich nur Text erzeugen können. Der wesentliche Unterschied jedoch ist, dass Modelle mit *Function-Calling*-Fähigkeit i. d. R. einen gesonderten Datensatz zur Feinjustierung verwenden, der auf die Darstellung von Methodenaufrufen ausgelegt ist (z. B. indem der Datensatz reich an Funktionsaufrufen und Programmcode ist). Spezielle und teure Methoden wie *Supervised Learning* können die Zuverlässigkeit der Funktionsaufrufe zusätzlich verbessern.

Mittlerweile erfolgt die Ausführung von Funktionen durch Sprachmodelle meist durch eine integrierte Schnittstelle, die es dem Modell erlaubt, mit anderen Softwaresystemen oder Diensten zu interagieren. Bei OpenAI wird dies durch zusätzliche Tools wie das „Code-Interpreter“-Tool ermög-

licht, das es dem Modell erlaubt, in einer sicheren Umgebung Programmcode auszuführen. Diese Funktionen werden in Echtzeit ausgeführt und die Ergebnisse direkt in die Interaktion mit dem Benutzer integriert. Alle drei großen proprietären Modelle unterstützen derartiges Function Calling:

- GPT-4 (OpenAI)
- Gemini (Google)
- Claude-2 (Anthropic)

Im Kontext der Open-Source-Modelle lässt sich sagen, dass diese Fähigkeit wesentlich seltener und in schlechterer Qualität zu finden ist. Anfang des Jahres (2024) hat NexusRaven-V2 durch seine 0-Shot-Fähigkeit im Function-Calling-Umfeld Aufmerksamkeit auf sich gezogen, aber selbst neue Open-Source-Modelle wie das erst kürzlich veröffentlichte Llama-3 hinken in diesem Bereich hinterher.

Wir wollen uns in diesem Artikel daher auf das Function Calling von ChatGPT beschränken, das sich über OpenAIs „GPTs“ (die konfigurierbare GPT-Variante im Premium-Abo) einfach und schnell für den gewünschten Anwendungsfall verproben lässt. Da die Konfiguration der GPTs mit der Maus erfolgen kann, eignen sie sich besonders für die ersten Gehversuche – solange man seine anzubindenden Schnittstellendefinitionen kennt.

In OpenAIs GPTs können Aufrufe von verschiedenen Methoden (Prozesssteuerung, Datenbankaufrufe, Auslösen

von Berechnungen usw.) über REST-APIs ermöglicht werden. Sobald das API über eine sichere Verbindung und Authentifizierung erreicht werden kann, ist GPT instand, zielsicher mit ihm zu interagieren. Dafür werden einfach im JSON-Format die möglichen Interaktionen der Schnittstelle definiert. GPT kann hier bereits Informationen aus der Beschreibung der Payload beziehen, wenn Parameterbezeichnungen oder Pfade solche Hinweise enthalten. Trotzdem empfehlen wir, über den *description*-Parameter weitere Hinweise zum Gebrauch und zur Funktion des API-Aufrufs in die JSON-Definition mit zu übergeben, um die Wahrscheinlichkeit für ein Fehlverhalten oder eine Arbeitsverweigerung zu minimieren.

Einen exemplarischen Ausschnitt, der ein API zum Abrufen von Bestellungsinformationen über eine Bestellnummer definiert, sieht man in **Listing 1**.

In diesem Fall wird über die Beschreibung (*description*) GPT mitgeteilt, dass über das angegebene API Informationen zu einer Bestellung besorgt werden können. Gleichzeitig erfährt das GPT hier, dass zur Verarbeitung ein weiterer Parameter, nämlich die Bestellnummer, benötigt wird. Der obige Ausschnitt reicht aus, damit GPT versteht, dass die Anfrage „Was weißt du zu meiner Bestellung?“ unvollständig ist und dass noch die Bestellnummer von den Usern erfragt werden muss. Eine reproduzierte Interaktion hierzu mit MoebelGPT, einem versatilen Assistenten eines Möbelkaufhauses, kann so aussehen wie im Kasten dargestellt.

Listing 1

```

"/api/purchase-order/{bestellnummer}": {
  "get": {
    "description": "Besorge Information zu einer
Bestellung",
    "operationId": "BestellungAPI",
    "deprecated": false,
    "parameters": [
      {
        "name": "bestellnummer",
        "in": "path",
        "required": true,
        "description": "Die Bestellnummer der
Bestellung",
        "schema": {
          "type": "integer"
        }
      }
    ]
  }
}

```

Die Informationen zur Bestellung bezieht GPT über die zuvor beschriebene Stelle, über die es wieder Zugriff auf eine Datenbank erhalten hat. Die Lese- und Schreibrechte GPTs in Bezug auf die Datenbank werden hierbei über die Schnittstelle definiert.

Ein anderes und vielseitig einsetzbares Beispiel wäre, dass ein Unternehmen im Kundensupport-Bereich ein Sprach-

User:

Habe ich ein Bett in meiner letzten Bestellung gekauft?

MoebelGPT:

Um Ihnen Informationen über Ihre letzte Bestellung und die darin enthaltenen Produkte zu geben, benötige ich Ihre Bestellnummer oder weitere Details, die mir helfen, Ihre Bestellung zu identifizieren. Können Sie mir bitte diese Informationen geben?

User:

Die Bestellnummer ist 7.

MoebelGPT:

In Ihrer letzten Bestellung (Bestellnummer 7) haben Sie tatsächlich ein Bett gekauft. Es handelt sich um ein „King-Size-Bett“ mit viel Platz, zum Preis von 499,99 €. Wenn Sie weitere Informationen zu diesem oder anderen Produkten aus Ihrer Bestellung benötigen, lassen Sie es mich wissen!

modell verwendet, um eingehende E-Mails automatisch zu analysieren und geeignete Aktionen auszulösen. Das Modell könnte anhand des Inhalts der E-Mail erkennen, ob eine Eskalation erforderlich ist, und dann den entsprechenden API-Aufruf generieren, um das Ticket zu eskalieren.

Es sollte schnell klar geworden sein, dass Unternehmen, die Integrationssoftware verwenden und ihre Schnittstellen gewissenhaft pflegen, vorbereitet sind für eine Zukunft, in der zumindest Teile der Prozesssteuerung an ein großes Sprachmodell ausgelagert werden. Eine hohe Anzahl an Schnittstellen bringt viele Einsatzmöglichkeiten für GPT-

Technologien, die de facto jetzt schon ohne einen erheblichen Mehraufwand umgesetzt werden können. Die Rolle von Integrationssoftware ist dabei essenziell, da sie nicht nur den Umgang mit einer Vielzahl von Schnittstellen erleichtert, sondern auch bei der Vereinheitlichung der Datenformate unterstützt, was besonders hilfreich für die effiziente Ansteuerung eines Sprachmodells ist. Dieser Aspekt war auch bereits in **Abb. 1** ersichtlich. Dort zeigt die rechte Seite, wie ein Sprachmodell („LLM“, gestrichelte Linie) nahtlos an das bestehende API zur Order-Historie angebunden werden kann, ohne dass dafür ein gesondertes API bereitgestellt werden muss.

Fazit

Die Integration von Sprachmodellen über API-Aufrufe ermöglicht es Unternehmen, neue Wege zur Automatisierung und Verbesserung ihrer Geschäftsprozesse zu beschreiten. Besonders fortgeschritten ist die Integration, wenn Technologien wie Camunda zur Prozesssteuerung verwendet werden, was nicht nur die Automatisierung bestehender Vorgänge verbessert, sondern auch strategische Entscheidungen durch datengesteuerte Analysen und Prognosen unterstützt. Dies eröffnet Perspektiven für eine tiefgreifende Transformation der Geschäftsmodelle und eine Steigerung der operativen Effizienz. Diese Wege lassen sich am einfachsten beschreiten, wenn der Pfad mit professionellen Integrationslösungen gepflegt wurde.

Zukünftig könnten diese Technologien weiter mit fortschrittlichen Systemen wie IoT und Big Data integriert

werden, um eine noch umfassendere Automatisierung und intelligentere Geschäftsprozesse zu ermöglichen.

***René Rother** ist IT Consultant bei der codecentric AG und Spezialist zu Themen rund um Integrationsplattformen und Applikationsentwicklung. Mit seiner tiefgehenden Expertise berät er Unternehmen bei der Optimierung ihrer technologischen Infrastruktur, um nahtlose Integrationen und effiziente digitale Lösungen zu schaffen.*

rene.rother@codecentric.de

***Kamil Braschke** ist IT Consultant bei der codecentric AG mit Fokus auf den Einsatzmöglichkeiten modernster LLM-Technologien in bestehende Unternehmensstrukturen. Mit Spannung verfolgt er die neuesten Entwicklungen der LLM-Landschaft, auf der Suche nach weiteren Möglichkeiten zur Prozessautomatisierung.*

kamil.braschke@codecentric.de

SHERLOCK

Automatisiere dein
Input-Management
mit künstlicher Intelligenz

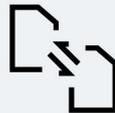


// Digitale & analoge Eingangsdokumente erkennen und verarbeiten

Nahtlose technische Integration in deinen Geschäftsprozess. In der Cloud, On-Premises, als REST-Services oder mit Workflow Management



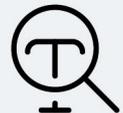
Vorverarbeitung,
Text- und Bilderkennung



Trennung von
Dokumentenströmen



Klassifikation
der Dokumente



Extraktion
der Daten

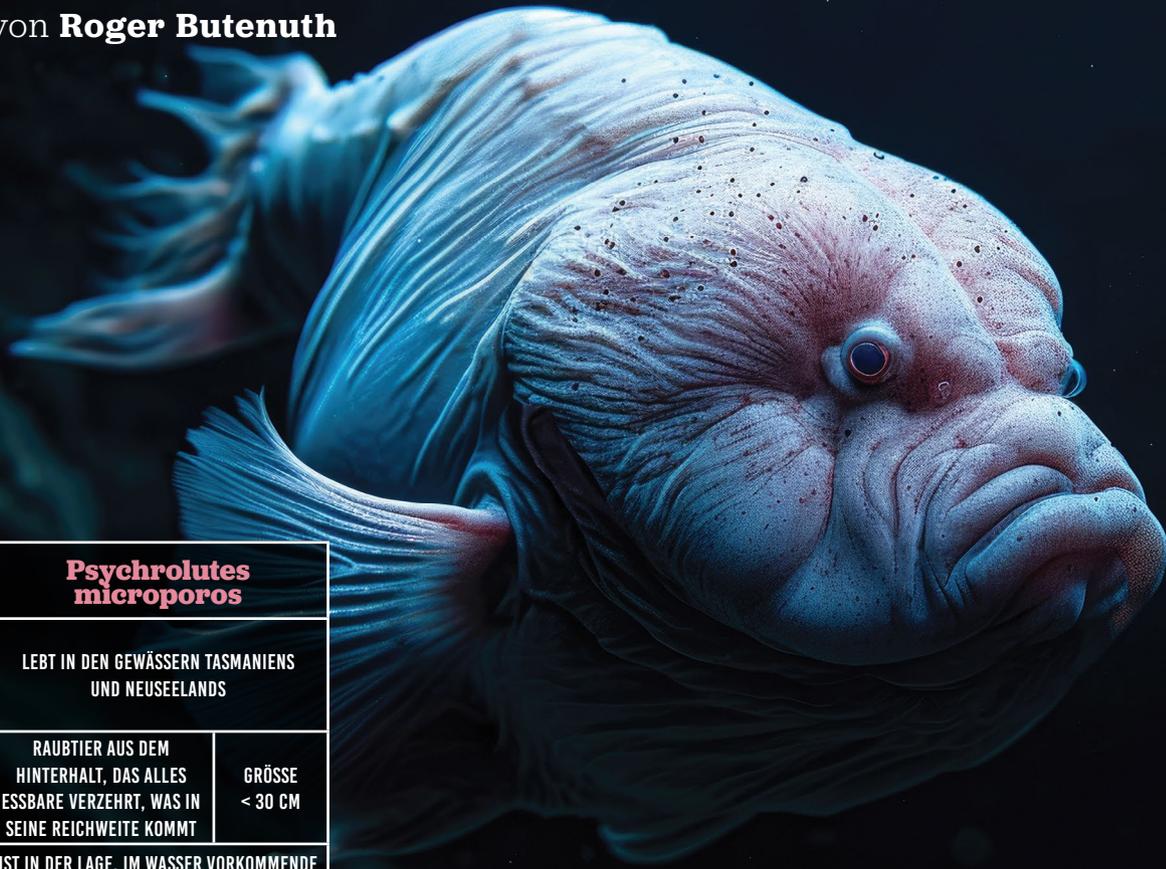


a solution by @codecentric

Vom Daten-Silo zum Daten-Zoo

Datenaustausch in Anwendungslandschaften
in den Griff bekommen

von Roger Butenuth



Psychrolutes microporos

LEBT IN DEN GEWÄSSERN TASMANIENS
UND NEUSEELANDS

RAUBTIER AUS DEM
HINTERHALT, DAS ALLES
ESSBARE VERZEHRT, WAS IN
SEINE REICHWEITE KOMMT

GRÖSSE
< 30 CM

IST IN DER LAGE, IM WASSER VORKOMMENDE
ARSENGIFTE AUF NANOSKOPISCHER EBENE ZU
ABSORBIEREN

Anwendungslandschaften sind heute meistens hybrid, oft sogar in mehreren Dimensionen: In einer Dimension existiert ein Mix aus on premises und Cloud, in der anderen zwischen Standard- und Individualsoftware (Abb. 1). Die einzelnen Anwendungen für sich sind dabei für das Business nur sehr begrenzt nützlich. Erst wenn ihre Daten verknüpft werden, kommt ihr Nutzen zur vollen Entfaltung. Aber wie macht man das? Wie bekommt man den Zoo in den Griff? Darum – und wie wir im Laufe der Zeit zu diesem Zustand gekommen sind – geht es in diesem Artikel.

Die gute alte Zeit

Früher™ war alles einfacher: ein Host, eine Datenbank, dumme Terminals, tagsüber interaktiv, nachts Batches und Backup. Im Vergleich zu heute war die Software einfach, alle Teile nutzten dieselbe Datenbank. Datenaustausch zwischen verschiedenen Teilen der Software fand durch Zugriff auf gemeinsame Tabellen statt. Die Folge war aber auch, dass die Software schwer zu ändern war: Wer zum Beispiel griff alles auf die Tabelle „Kunde“ zu? Neue Releases erforderten daher aufwendige Tests, meistens auch noch manuell. Releases waren selten, oft gab es nur ein bis zwei pro Jahr.

Mit dem Einzug der PCs in die Büros änderte sich das Szenario komplett: Plötzlich hatte jeder Rechenleistung und Speicher unter dem Schreibtisch stehen. Dazu stand für einige Aufgaben Standardsoftware bereit: Textverarbeitung, Tabellenkalkulation, E-Mail, Kalender. Manches

davon läuft komplett lokal, manches benötigt einen Server. Das war dann meistens nicht mehr der Host, sondern eine Unix- oder Windows-Maschine. Willkommen in der Welt der verteilten Systeme!

Um es noch etwas komplizierter zu machen, kam Schatten-IT hinzu: Excel-Sheets mit komplizierten Formeln und Makros. Oder als Schmeißerl Access-Datenbanken mit Visual Basic. In vielen Fällen ohne Dokumentation, Tests, Versionierung oder Trennung von Entwicklung und Produktion. Der Vorteil: kurze Release-Zyklen, jede Änderung wird sofort aktiv.

„Echte Anwendungen“ (von der zentralen IT erstellt) waren im ersten Ansatz oft sogenannte Fat Clients, die alle Logik auf dem PC ausführten und nur eine zentrale Datenbank nutzten. Zentrale Datenbank mit einem schwachen Netzwerk – insbesondere die Latenz ist kritisch – führt jedoch oft zu schlechter Performance. Mit der nächsten Evolutionsstufe, einer Drei-Schichten-Architektur, lässt sich das

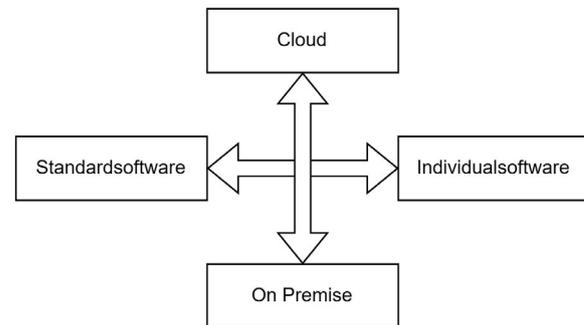


Abb. 1: Anwendungen können in zwei Dimensionen unterschieden werden.

umgehen. Tauscht man dabei noch den Fat Client durch eine Webanwendung aus, erledigt sich auch das Problem der Softwareverteilung auf PCs.

Dieser Zoo aus Host, Fat Clients, Schatten-IT und dreischichtigen Architekturen wird noch komplexer, wenn Datenaustausch oder Zusammenarbeit zwischen den Anwendungen erforderlich ist.

SOA

Irgendwann kam dann die Lösung für alle genannten Probleme: SOA (Service-orientierte Architektur, **Abb. 2**). Die Idee dahinter setzt – mal wieder – auf mehrere Schichten und hört sich auf den ersten Blick gut an: Über den Datenbanken „ganz unten“ befindet sich eine Schicht mit Datenservices, z. B. pro Domäne ein Datenservice mit einer eigenen Datenbank. In ihnen werden fachliche Constraints und Konsistenz überwacht. In der nächsten Schicht findet

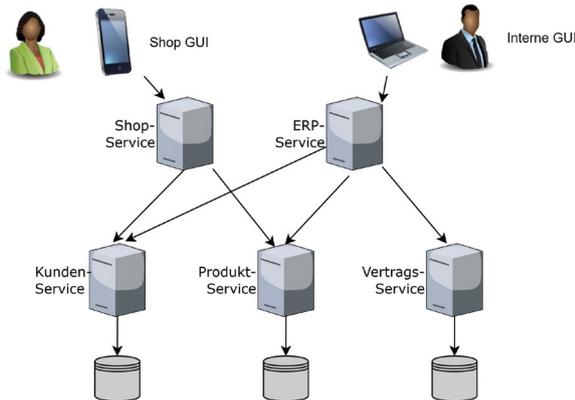


Abb. 2: SOA (Service-orientierte Architektur) sollte alle Probleme lösen.

man Services für die Fachlichkeit, die komplexere Dinge wie z. B. „Vertrag anlegen“ ausführen. „Ganz oben“ sind schließlich Services für die Oberflächen. Je nach Architekturvariante kann sich zwischen den Schichten noch ein ESB (Enterprise Service Bus) oder eine Orchestrierungsschicht befinden. Für die Kommunikation wurde dann meistens SOAP genutzt, was – theoretisch – Hersteller- und sprachunabhängig ist. Praktisch scheitert SOAP gerne an Varianten oder kleinen Inkompatibilitäten.

Der theoretische Vorteil dieser Architektur: Alle Daten und Konsistenzprüfungen haben genau eine Heimat, sie sind nur einmalig zu implementieren und können wiederverwendet werden. „Höhere“ Services müssen sich auch nicht mit den Niederungen der Datenbank herumschlagen, das erledigen ja die Datenservices für sie. Außerdem kann man einzelne Services einkaufen und verschiedene Services leicht austauschen. In der Praxis hat das nie wirklich funktioniert, wofür sich auch leicht Gründe finden lassen:

1. Es hat nie standardisierte Schnittstellen gegeben. Services lassen sich daher nicht einfach einkaufen, alles muss selbst gebaut werden.
2. Wiederverwendbare Services zu schreiben ist deutlich aufwendiger als individuelle Services zu schreiben. Die versprochene Aufwandsreduktion stellt sich daher nicht ein. Siehe auch [1].
3. Einmalig vorhandene Daten erfordern, dass sich alle Abteilungen mit Interesse an diesen Daten auf ein gemeinsames Datenmodell einigen. Allein mit dem dazu

notwendigen Abstimmungsprozess lässt sich jede Projekt-Deadline reißen.

4. Um eine Aktion an der Oberfläche durchzuführen, benötigt man immer Zugriffe durch alle Schichten bis zur Datenbank, oft nach unten aufgefächert auf mehrere Services oder mit mehrfachen Aufrufen pro Service. Beides ist sowohl für Performance – Addition von Latencies – als auch für Zuverlässigkeit – Multiplikation von Verfügbarkeiten < 1 – abträglich.

Der erste oder dritte Punkt allein reichen schon aus, die Idee von SOA zu begraben: Es ist unsinnig, sämtliche Software im Unternehmen selbst zu schreiben. Custom Software lohnt sich nur für Bereiche, in denen man sich mit Alleinstellungsmerkmalen am Markt positionieren möchte. Dinge wie Buchhaltung gehören sicher nicht dazu. Ich kenne kein Unternehmen, in dem eine SOA erfolgreich implementiert worden ist.

Gegenwart

Wenn ein Unternehmen groß genug ist, hat es von den beschriebenen Techniken vermutlich alle im Einsatz. Vom Host im Keller bis zu modernen Microservices oder SaaS (Software as a Service) in der Cloud. Dazu hat dann sicher noch jemand die Strategie vorgegeben, dass in den nächsten Jahren alles in die Cloud muss. Oder es soll die in Jahrzehnten geschriebene Software durch ein oder mehrere Produkte ersetzt werden. „Produkt“ bedeutet in diesem Zusammenhang oft auch SaaS. So werden Implementierung und Betrieb zusammen ausgelagert. Haben diese beiden

Stoßrichtungen irgendwelche Gemeinsamkeiten? Definitiv: Man gibt Kontrolle ab und kann somit einige Integrationshacks aus der Vergangenheit nicht mehr oder zumindest nicht mehr so leicht anwenden. Auf dem Host lag alles in einer Datenbank. Auch bei mehreren Datenbanken im eigenen Rechenzentrum ist der Zugriff immer noch möglich.

Integration über Zugriffe in fremde Datenbanken war noch nie eine gute Idee: Sie führt zu einer ungewollt starken Kopplung von Systemen. Bei SaaS in der Cloud ist dieser Weg technisch blockiert.

Was bleibt übrig? Mit der neuen Realität leben und Integration so durchführen, wie es die verwendeten Produkte unterstützen. Das heißt: Entweder bieten die Produkte fertige Schnittstellen (vornehmer Name: APIs) oder zumindest einen Plug-in-Mechanismus, über den man sie um eigene Software – und damit um Schnittstellen – erweitern kann.

Ist beides nicht gegeben, hat man das falsche Produkt gekauft. Integration ist dann nur noch über die Oberfläche möglich: Menschen, die lesen und tippen oder RPA ([2], Robotic Process Automation). Beide Varianten sind nicht effizient und erstrebenswert, also aufgepasst bei der Softwarebeschaffung!

APIs von Standardsoftware sind meistens passive APIs: Die Software bietet eine Schnittstelle an, die passiv darauf wartet, dass jemand aktiv darauf zugreift. Theoretisch denkbar, in der Praxis aber selten ist es, dass die Standardsoftware ein API definiert, auf das sie aktiv zugreifen

möchte, das aber von einer „anderen Seite“ implementiert werden muss.

Wenn beide Seiten nur passive Schnittstellen anbieten, benötigt man auf jeden Fall eine aktive Komponente dazwischen, von der die Initiative ausgeht. Selbst wenn eine Seite die Initiative ergreifen kann, ist es notwendig, dass beide Seiten eine gemeinsame Sprache sprechen (kompatibles API). Ist dies nicht gegeben (eher der Standardfall), muss der Adapter entweder in einer Komponente in der Mitte liegen oder als Erweiterung in einer Software auf einer der beiden Seiten implementiert werden.

Beispiel: Kundendaten

Nach den abstrakten Darstellungen ist es Zeit für ein konkretes Beispiel: Stellen wir uns ein Unternehmen vor, das in mehreren Systemen Kundendaten gespeichert hat: einem Online-Shop (in der Cloud), einem ERP-System (On Premise) und einem CRM-System (in der Cloud). Alle drei benötigen offensichtlich Kundendaten: der Shop für die Lieferung, das ERP-System zum Abwickeln von Bestellungen, das CRM-System für Marketingaktivitäten.

Beginnen wir mit einem einfachen (unrealistischen) Szenario: Nur ein System hat die Hoheit, die Daten zu ändern: der Shop, in dem der Kunde selbst seine Daten aktualisieren darf. Wenn wir Glück haben, kann der Shop für jede Änderung ein Event erzeugen, Beispiel: „Kunde 42 hat die neue Adresse...“ Das Event müssen wir nur noch an die beiden anderen Systeme verteilen und dort Kundendaten aktualisieren. Klingt auf den ersten Blick einfach, solange wir die Herausforderungen verteilter Systeme ignorieren:

1. Zuverlässigkeit (verlorene Events)
2. Reihenfolge von Events
3. Einmaligkeit von Events

In unserem Beispiel haben wir nur mit dem dritten Punkt kein Problem: Kundendaten mehrfach auf den gleichen Stand zu bringen schadet nicht. Die ersten beiden können offensichtlich zu Inkonsistenzen führen: Geht ein Event verloren, werden im Zielsystem die Kundendaten nicht aktualisiert. Wird die Reihenfolge von Events vertauscht, überschreiben alte Daten neuere Daten. Daraus folgen Inkonsistenzen zwischen den Systemen.

Wodurch können die Probleme überhaupt erst entstehen? Diverse Ausfälle (Hard- und Software): Paketverluste, defekte Rechner oder – vermutlich häufiger – Abstürze von Servern. Die in der Software eingebauten Gegenmaßnahmen führen dann zu weiteren Problemen: Stellt man Events zu, bis man eine Bestätigung erhalten hat, führen Paketverluste auf dem Rückweg dazu, dass Events doppelt zugestellt werden. Es lässt sich beweisen, dass im allgemeinen Fall eine „Exactly-once“-Zustellung nicht möglich ist. Entweder erhalten die Systeme Events doppelt oder es gehen Events verloren. In den meisten Fällen ist doppelte Zustellung das kleinere Übel gegenüber dem Verlust. Baut man seine Systeme nicht sequenziell, sondern mit Parallelität (zur Leistungssteigerung oft unvermeidlich), lässt sich Reihenfolgetreue nicht mehr garantieren.

Wie geht man damit um? Enthält ein Event den gesamten Objektzustand, gibt es gegen Vertauschungen ein einfaches

Gegenmittel: Im Event muss eine Versionsnummer oder ein Zeitstempel ergänzt werden. Diesen muss man in allen Systemen speichern. So lassen sich veraltete Events erkennen und ignorieren. Leider ist es nicht mehr so einfach, wenn nicht alle, sondern nur geänderte Attribute übertragen werden.

Ein anderer Lösungsansatz sind Events, die nur noch eine ID des geänderten Objekts enthalten. Das Zielsystem holt sich dann beim Eintreffen des Events die in jedem Fall aktuellen Daten im Quellsystem ab. Der Nachteil dieser Lösung ist der erhöhte Kommunikationsaufwand und der damit verbundene geringere Durchsatz.

Wie man sieht, steckt bei einer vermeintlich einfachen Datensynchronisation der Teufel im Detail. Dabei habe ich ein Teilproblem bis hierher ignoriert: Ich habe angenommen, das Quellsystem generiert ein Event, wenn sich Daten geändert haben. Entweder für eine interne Erweiterungsschnittstelle oder als externes Event (Kafka Event, Webhook etc.). Was ist, wenn das nicht gilt? Wie kann man sich selbst Events erzeugen? Das hängt wieder etwas vom System ab: Mit etwas Glück enthalten alle Objekte einen Zeitstempel der letzten Änderung. Dann kann man regelmäßig nach Daten suchen, die einen jüngeren Zeitstempel als die letzte Abfrage enthalten. Mit gelöschten Daten funktioniert das jedoch nur, wenn die Daten nicht sofort physisch gelöscht werden, sondern nur als gelöscht markiert werden.

Ohne Zeitstempel – oder für die Entdeckung gelöschter Objekte – hilft nur, eine Schattenkopie zu halten und die Daten im Quellsystem mit der Kopie zu vergleichen. Neben

dem Aufwand hat dieser Ansatz den Nachteil, dass er nicht so aktuell wie vom Quellsystem generierte Events ist.

Mehrweg

Wie sieht es aus, wenn wir die Bedingung fallen lassen, dass nur der Shop die Kundendaten ändern darf? Wenn die Daten überall aktuell bleiben sollen, müssen wir sie in alle Richtungen synchronisieren.

Der erste Teil der Aufgabe bleibt gleich: Aus den Änderungen müssen Events generiert werden. Die Verarbeitung muss jedoch anders ausgeführt werden: Die Daten dürfen nicht mehr „einfach so“ in die anderen beiden Systeme kopiert werden. Warum? Folgende Probleme können auftreten:

1. Daten werden in zwei (oder mehr) Systemen parallel geändert, bevor die Synchronisation „angekommen“ ist. In der Folge können Änderungen verloren gehen.
2. Wenn die Änderung durch ein Event von einem System in die anderen Systeme übertragen wird, löst sie dort Events aus. Im schlimmsten Fall handelt man sich damit eine Endlosschleife zwischen den Systemen ein.

Welche Maßnahmen können die Probleme beseitigen? In der Theorie könnte man einfach parallele Änderungen verhindern: Bevor ein Kundendatensatz in einem der Systeme geändert wird, sperrt man ihn einfach. In der Praxis wird dies kaum möglich sein: Wie bringt man dies einem gekauften System bei? Außerdem sind solche Sperren äußerst unpraktisch: Man muss sie nicht nur seinen Anwendern erklären (auch den Kunden vom Shop!), sondern auch dafür sorgen, dass Sperren nicht versehentlich gesetzt bleiben und damit Datensätze ewig blockiert werden. Wie schon bei der

unidirektionalen Synchronisierung sind auch hier Zeitstempel in den Daten hilfreich, noch besser wären über alle Systeme hinweg einheitliche Versionsnummern (wieder so ein Wunschtraum ...).

Da wir zusätzlichen persistenten Speicher benötigen, ist die Synchronisation am besten in einem zentralen Service aufgehoben, der die Events von allen entgegennimmt. In seiner Datenbank speichert er nicht nur die Daten selbst, sondern auch die ID (Primärschlüssel) sowie Versionsnummern (oder Zeitstempel) aller verbundenen Systeme. Empfängt der Service nun ein Event mit geänderten Daten, kann er diese mit den bei ihm gespeicherten Daten vergleichen. Sind sie identisch, müssen die anderen Systeme nicht mehr benachrichtigt werden. Bei Unterschieden müssen die anderen Systeme (und die eigenen Daten)

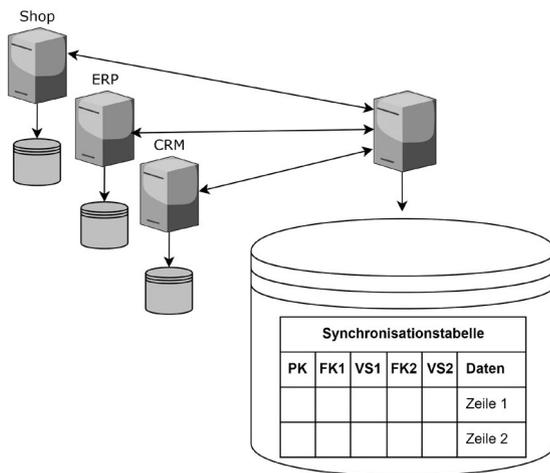


Abb. 3: Zentraler Synchronisationsservice

aktualisiert werden. Dabei muss die letzte bekannte Versionsnummer (bzw. der Zeitstempel) der Zielsysteme berücksichtigt werden: Änderungen dürfen nur eingespielt werden, wenn Versionsnummer (bzw. Zeitstempel) übereinstimmen. So wird sichergestellt, dass man keine im Zielsystem parallel aktualisierten Daten überschreibt. Der erkannte Konflikt lässt sich nun entweder durch einen Algorithmus auflösen (dürfte fachlich schwierig sein), oder man leitet ihn aus, sodass er von einem Menschen manuell aufgelöst wird. In der Praxis sollten solche Konflikte selten sein. Wann ändert schon ein Kunde seine Daten im Webshop und parallel jemand im ERP oder CRM die Daten desselben Kunden?

Entwicklern unter den Lesern dürfte das beschriebene Verfahren bekannt vorkommen: Genau die gleiche Idee steckt hinter dem optimistischen Locking in Versionsverwaltungen wie SVN oder Git: Eigene Änderungen lassen sich nur in das zentrale Repository einspielen, wenn sie auf einer aktuellen Kopie der Daten vorgenommen worden sind. Ansonsten meldet die Versionsverwaltung einen Konflikt, den der Entwickler manuell auflösen muss.

Die zentrale Datenhaltung (Abb. 3) ist auch eine gute Gelegenheit, sich über das Datenmodell fachlich Gedanken zu machen (DDD lässt grüßen, siehe Artikel von Annegret Junker auf S. 38). Übernimmt man das Modell eines der gekauften oder vorhandenen Systeme oder führt man ein Modell ein, das auf das eigene Unternehmen am besten passt? Das muss dann aber – über Adapter – irgendwie doch noch mit den Modellen in den anderen Systemen unter einen Hut gebracht werden.

Event oder Kommando?

Im Text verwende ich den Begriff „Event“. Events werden manchmal auch als „unwiderlegbar geschehene Fakten“ bezeichnet. Beispiel: „Daten des Kunden xyz wurden geändert.“ Das Event ist daher erst mal passiv. Oft steckt hinter unschuldig vorgetragenen Fakten jedoch ein impliziter Auftrag: Hinter „Ich stehe hier schon seit Stunden herum.“ steckt die implizite Aufforderung „Mach dich endlich auf den Weg!“. Fowler nennt das auch „passive-aggressive events“ [3].

Warum ist der Unterschied hier relevant? Events lassen sich einfach sequenziell speichern, zum Beispiel in Kafka Streams. Möchte man sie möglichst *genau einmal* als Kommando ausführen („*Paket an Kunde schicken*“), treten die im Text beschriebenen Probleme auf. Besonders kritisch sind nicht idempotente Aktionen, wie z. B. das Verschicken einer Bestellung.

Ausblick/Fazit

Ich habe die Herausforderungen in verteilten – fertig gekauften oder in der Cloud gemieteten – Systemen hier nur angerissen, der Stoff reicht sicher für mindestens ein Buch. Ist es deswegen schlecht, fertige Software zu benutzen und nicht alles selbst zu schreiben? Nein! Für Standardaufgaben lohnt es sich nicht, das Rad neu zu erfinden. Die Energie und Entwicklungspower sind besser in Software investiert, die neue, individuelle Märkte bedient.

Wer sich viele Systeme ins Haus holt, darf jedoch eines nicht vergessen: Um Nutzen zu stiften, müssen ihre Daten

verknüpft werden. Darin liegt die Herausforderung, aber auch die Chance für eine nutzbringende IT.

Dr. Roger Butenuth arbeitet seit 2012 bei der codecentric AG mit dem Schwerpunkt Integration. Im sonstigen Java-Umfeld interessieren ihn Themen rund um Performance, verteilte Systeme und Parallelität. Privat darf es dann auch schon mal ein Microcontroller sein, der in C++ nah an der Hardware programmiert wird.
roger.butenuth@codecentric.de

Referenzen

- [1] Frederick P. Brooks: *The Mythical Man Month*, Kapitel 1: „The Tar Pit“
- [2] RPA (Robotic Process Automation): https://de.wikipedia.org/wiki/Robotic_Process_Automation
- [3] Martin Fowler: „What do you mean by ‚Event-Driven?‘“ <https://martinfowler.com/articles/201701-event-driven.html>

LIVE
UND ON DEMAND

Unsere Webinare

Aktuelles Wissen, für dich
zusammengefasst und von
unseren Expert*innen
in deinem Wohnzimmer
präsentiert.

Du möchtest dich fortbilden, aber dafür nicht reisen?
Besuche einfach eines unserer Webinare. Sie haben alle
Vorteile eines Seminars, sind interaktiv, du siehst die
Speaker*innen und die Slides – aber musst dafür nicht
den Komfort deiner Wohnung verlassen.



On demand u. a.:

Cloudkosten-Optimierung mit FinOps

Im Webinar mit unseren FinOps-Experten Niko Blättermann und Lukas Miliunas zeigen wir dir die größten Einsparpotenziale bei deinen Cloudkosten. Du bekommst einen Überblick darüber, wie du deine laufenden Kosten signifikant reduzieren kannst und wie wir typischerweise in Projekten vorgehen, um Prozesse zu optimieren.

CompanyGPT mit und ohne OpenAI™

Die LLM-Experten Niklas Haas und Oliver Moser zeigen dir, wie du ein firmeneigenes GPT in der Praxis umsetzen kannst. Du bekommst einen Überblick darüber, welche Herausforderungen und Fallstricke du dabei beachten musst und welche Umsetzungsmöglichkeiten du hast, um das Potenzial von GPT-Modellen bestmöglich nutzen zu können.

Ist Platform Engineering für dein Unternehmen sinnvoll?

Das Webinar mit unseren Platform-Engineering-Experten Marc Schnitzius und Victor Volle hilft dir zu bewerten, ob sich der Aufbau einer internen Entwicklerplattform für dich lohnt. Du bekommst einen Überblick über mögliche Vorteile und lernst, wie eine Umsetzung konkret funktionieren kann.

Du profitierst von unserer Erfahrung aus echten Projekten, lernst Best Practices und mögliche Stolperfallen kennen und erhältst Antworten auf deine Fragen.

Alle Webinare findest du hier:

<https://www.codecentric.de/wissens-hub/webinare>



Damit die IT-Forensik nicht weint

Wie logge ich für den Ernstfall?

von **Antonia Schmalstieg**



Nacktkiemer

KOMMT IN ALLEN WELTMEEREN VOR

ERNÄHRT SICH VON SCHWÄMMEN, ANDEREN MEERES-SCHNECKEN/DEREN EIERN

GRÖSSE
< 60 MM

LEBT VON DER GEZEITENZONE BIS
IN TIEFEN WEIT ÜBER 700 M

Irgend etwas mit Logging hat vermutlich jede Person zu tun, die Software entwickelt. Doch hilft dieses Logging auch, um mit den Folgen eines Cyberangriffs umzugehen?

Logging – kennen wir alle, wissen wir alle, oder?

Logs helfen uns, nachzuvollziehen, was unsere Software tut, Fehler zu finden. Sie helfen uns aber auch, zu sehen, was mit unserer Software und unseren IT-Systemen passiert, wie mit ihnen interagiert wird.

Diesen Aspekt des Loggings ebenso auf dem Schirm zu haben, wie beispielsweise gutes Debugging zur Fehlerfindung zu ermöglichen, hilft, die Sicherheit unserer IT-Systeme zu verbessern. Logs sind der erste Schritt, einen Überblick – auch über laufende Systeme – zu bekommen und auf Anomalien reagieren zu können. Aber auch, wenn ein Angriff bereits erfolgreich durchgeführt wurde, helfen gute Logs.

Der Ernstfall – wenn die IT-Forensik hinzukommt

Aber wie sieht das aus, so ein erfolgreich durchgeführter Angriff? Seltsame Aktivitäten auf Firmenservern, vertrauliche Daten tauchen im Darknet auf oder plötzlich sind alle Systeme verschlüsselt, die Produktion steht still. All das sind Szenarien, in denen IT-Forensik-Expert*innen gefragt sind. Die Folgen eines Cyberangriffs können so vielseitig sein wie die Angriffe selbst.

Werden Folgen eines Cyberangriffs erkannt, wird meistens die IT-Forensik eingeschaltet. Forensik im Allgemeinen

befasst sich mit dem Identifizieren, Analysieren und Rekonstruieren von kriminellen Handlungen. Bekannt sind Vorgänge aus der Rechtsmedizin, wie die DNA-Analyse oder auch die Auswertung von Schriftstücken oder anderen Beweismitteln durch entsprechende Gutachter. Die IT-Forensik leistet eben diese Analyse und Rekonstruktion, allerdings in Bezug auf Cybercrime.

Wie läuft ein Forensikfall?

Völlig anders als die anderen Bereiche der Forensik arbeitet auch die IT-Forensik nicht: Ein Angriff wird festgestellt, IT-Forensiker*innen treten auf den Plan, oft sind wir unter den Ersten vor Ort.

Dort geht es vor allem erst einmal darum, Daten zu sichern und auszuwerten. Zwei Aufgaben stehen im Fokus: Zum einen muss der sogenannte Eintrittsvektor ermittelt werden, also der Weg, über den der Angriff durchgeführt werden konnte. Zum anderen geht es darum, herauszufinden, auf welchen Systemen sich die Angreifenden bewegen konnten und welche Daten im Zugriff waren. Hierfür muss eine überprüfbare Beweiskette erstellt werden.

Doch besonders im Rahmen großer Angriffsszenarien, bei denen ganze Systemlandschaften lahmgelegt werden, ist der Wiederaufbau eine wichtige Aufgabe von IT-Forensik-Expert*innen und Incident-Respondern. Und je mehr wir darüber wissen, was mit einem System passiert ist, wer die Angreifenden sind, welche Tools sie verwendet haben usw., desto besser und sicherer können wir diesen Wiederaufbau leisten. Schließlich wollen wir nicht, dass die Angreifenden

nach dem mühsamen Wiederaufbau die errichtete Systemlandschaft gleich wieder befallen!

Gutes Logging macht den Unterschied!

Natürlich sind Logs nicht der einzige Faktor, der über Erfolg und Misserfolg einer Forensik entscheidet. Doch ein Großteil der Daten, die wir analysieren, sind in irgendeiner Form Logdaten. Schnell zugängliche und vollständige Logs erlauben uns, uns möglichst zügig ein Bild über einen Vorfall zu machen. So können die Geschädigten schnell reagieren und Schäden durch Produktionsausfall, Vertragsstrafen, Umsatzausfälle usw. möglichst gering halten.

Aber mich trifft es doch nicht?

„I am not a target“ – also „Ich bin kein Ziel“ – ist leider nach wie vor einer der IT-Sicherheitsmythen, die sich mit am hartnäckigsten halten. Die Realität zeigt leider, dass potenziell jeder ein Ziel ist. Der Lagebericht des Bundesamtes für Informationstechnik (BSI) [1] zeigt es recht deutlich: Allein 2023 fanden 68 erfolgreiche Ransomware-Angriffe auf Unternehmen statt. Im Juni 2023 wurden täglich im Schnitt 332 Schadprogramm-Varianten in Umlauf gebracht, täglich wurden in Deutschland rund 21.000 infizierte Systeme erkannt. Und auch in unserer täglichen Arbeit sehen wir, dass Cyberangriffe wirklich jeden treffen können: Privatpersonen, Behörden, Kommunen, Firmen jeder Größe.

Natürlich gibt es Wege und Möglichkeiten, die Gefahr eines Angriffs zu minimieren – etwas, was jedes Entwicklungs-

team tun sollte. Doch auch mit den besten Praktiken kann es zu erfolgreichen Angriffen kommen.

Deswegen sollte die Vorbereitung auf einen Ernstfall Teil jedes IT-Projekts sein. Dazu gehören Themen wie die Etablierung der Chain of Communication (wer eskaliert im Ernstfall an wen?) und einer Chain of Command (wer trifft Entscheidungen?) oder erprobte Backup-Strategien. Dazu gehört aber auch ein Loggingkonzept, das die nötigen Daten zugänglich macht und auch davon ausgeht, dass Logs selbst Ziele von Angriffen werden.

Welche Logs braucht die IT-Forensik?

Aber welche Daten genau werden denn jetzt eigentlich gebraucht für so eine IT-Forensik? Natürlich können alle Informationen wichtig sein. Jede Information kann den entscheidenden Hinweis auf den Eintrittsvektor der Angreifenden liefern, oder darauf, was genau eigentlich passiert ist. Nicht selten geben auch Mitarbeitende wichtige Hinweise, zum Beispiel auf Phishingmails. Doch oft sind es ähnliche Logs, die uns als Forensiker*innen helfen, herauszufinden, was passiert ist.

Besonders im Fokus sind alle Logs, die Aufschluss darüber geben, wann und über welchen Weg Akteure Zugriff auf das System bekamen. Dazu gehören unter anderem:

- **Firewall-Logs:** Diese Logs geben uns vor allem über die eingehenden Verbindungen Aufschluss. Ausgehende Verbindungen werden oft nicht oder nur sehr spar-

sam geloggt. Sonst kann nachvollzogen werden, welche individuellen Nutzer*innen wann welche Seiten ansurfen, Chefs oder Kolleg*innen können persönliche Informationen zu Arbeitszeiten oder -weisen oder Surfverhalten bekommen. Das ist datenschutzrechtlich schwierig und oft sogar illegal.

- **VPN-Logs:** Besonders, wenn sich sensible Teile des Systems in einem Firmen-VPN befinden, können VPN-Logs Aufschluss darüber geben, wann und wo die Angreifenden Zugriff bekommen haben.
- **Netzwerk-Logs:** Wenn Angreifende nicht von innen kommen, müssen wir davon ausgehen, dass dies über das Netzwerk geschah. Die Spuren, die sie in den Logs hinterlassen, können uns Aufschlüsse über die Angreifenden selbst geben und uns auch helfen, eine Zeitleiste der Angriffe zu erstellen. Auch eine Extraktion von Daten kann oft über diese Logs nachvollzogen werden.

Auch auf der Systemebene gibt es eine ganze Menge Logarten, die uns Informationen über den Angriff geben können:

- **Eventlogs,** insbesondere von aktiven Domain-Controllern: In diesen Logs finden sich Änderungen von Nutzern, wie sie zum Beispiel bei einer Rechtausweitung („Privilege Escalation“) stattfinden, oder auch das Anlegen oder Löschen neuer Nutzer. Im Schnitt sehen wir diese Logs etwa einen Tag lang, bei größeren Firmen mit vielen Clients gehen Logs teilweise nur wenige Minuten zurück. Auch Systemstarts, Programminstallationen und -ausführungen oder Remote-Desktop-

Verbindungen zeigen sich oft in Eventlogs. Sie können Hinweise auf das Tooling der Angreifenden geben oder einen Kontextwechsel auf ein anderes System zeigen.

- **Logs von Login-Aktivitäten:** In diesen Logs können wir erkennen, wann Angreifende erfolgreich Zugriff auf Accounts erlangen konnten, aber auch, ob sie beispielsweise über einen Brute-Force-Angriff, also das Probieren möglichst vieler Passwörter, Zugriff erlangten oder das Passwort bereits kannten – dann ist im Log nur ein einzelner erfolgreicher Loginversuch zu erkennen. Außerdem können wir sehen, wie oft die Angreifenden im System waren und seit wann.
- **Changelogs,** vor allem für Berechtigungen: Auch mit einem lokalen Account können Angreifende oft großen Schaden anrichten. Änderungen hierfür sind nicht in den Logs der Domänen-Controller zu sehen.

Die Liste lässt sich noch weiterführen: Aus Datenbanken-Logs können Zugriffe auf sensible Daten ermittelt werden, Server-Logs können anzeigen, welches Tooling die Angreifenden installiert haben, Application-Logs können unerwartetes Verhalten der Anwendungen zeigen und damit Hinweise auf die Angreifenden liefern usw. Alles, was die Angreifenden berührt haben könnte, ist ein potenzieller Ort, an dem wir ihre digitalen Spuren finden können.

Sonderfall SIEM

Einige Firmen haben ein *Security Information and Event Management* (SIEM) im Einsatz. Das ist eine Software, die verschiedene Logs und Eventinformationen zentralisiert

sammelt und auch auswertet. So können Angriffe auf ein System oft schon, während dieser passiert, erkannt und eingedämmt werden.

Auch für Analysen vergangener Zeiträume werden SIEMs oft eingesetzt. Doch für die Forensik sind SIEMs oft gar nicht so hilfreich, wie es auf den ersten Blick scheint. SIEMs arbeiten oft mit gewaltigen Datenmengen. Deswegen werden nur Daten gesichert, die von dem SIEM zum Zeitpunkt der Sicherung als relevant eingestuft werden. Durch diese Vorfilterung können aus einem SIEM meist nicht die für eine forensische Untersuchung notwendigen möglichst vollständigen Logs zur Verfügung gestellt werden.

Ein weiteres Problem ist, dass SIEMs reaktiv funktionieren. Das heißt, sie können nur auf Angriffe reagieren, die dem Tool bekannt sind. Auch bei kompetitiven SIEM-Tools dauert es oft zwei Wochen oder länger, bis ein neuer Angriff vom Tool erkannt wird. Ein SIEM bietet also zusätzlichen Schutz, ersetzt jedoch weder gute Sicherheitsmaßnahmen darüber hinaus, noch stellt es Forensic Readiness sicher.

Was schief laufen kann

Schief laufen kann bekanntlich alles. An dieser Stelle wollen wir vor allem die größten Probleme zeigen, in die wir als IT-Forensiker*innen in Bezug auf Logs laufen:

Es gibt keine Logs.

Über dieses Problem fallen wir leider viel zu oft. Zum Glück betrifft es oft nicht komplette Systeme, aber nicht selten

sind für zentrale Komponenten des Systems keine Logs vorhanden. Unsere Arbeit ist an dieser Stelle dann sehr wenig zufriedenstellend beendet: Eine Forensik kann nur so gut sein wie ihre Beweismittel. Liegen diese nicht vor, können wir keine Aussage über den Angriff treffen.

Logs sind nicht schnell verfügbar.

Oft existieren Logs. Aber vielleicht liegen sie irgendwo tief vergraben in einer Software und müssen erst noch exportiert werden. Oder die Backuplösung der Logs erlaubt keinen schnellen Zugriff. Oder die Logs werden in proprietären Formaten vorgehalten, die niemand versteht. Oder oder oder. Wenn der Export der Logs der letzten sieben Tage 14 Tage dauert, ist das für eine wirkungsvolle IT-Forensik nicht schnell genug!

Deswegen ist es sinnvoll, sich schon frühzeitig Gedanken darüber zu machen, wie auf Logs zugegriffen werden kann. Das sollte sowohl aktuelle Logs als auch Logs für weiter zurückliegende Zeiträume umfassen.

Logs wurden zu früh gelöscht.

Der kleinere Bruder nicht vorhandener Logs sind Logs, die zwar zu einem Zeitpunkt vorhanden waren, es aber nicht mehr *sind*. Domain-Controller-Logs gehen oft nur wenige Minuten zurück, doch auch andere Logarten werden oft schneller gelöscht, als uns Forensiker*innen lieb wäre – besonders, wenn das Logging-Konzept vor allem Debugging und Performance-Analyse berücksichtigt. Ein klassischer Cyberangriff läuft oft über mehrere Tage, oft auch Wochen oder teilweise sogar Jahre. Oft können

Angreifende mit ihrem ersten Zugang zu dem System noch nicht viel anfangen, müssen Rechte ausweiten oder sich auf andere Komponenten des Systems ausbreiten. Oft brauchen sie Zeit, um die Durchführung ihres eigentlichen Angriffs vorzubereiten oder Daten zu extrahieren. Auch können die Angreifenden sogenannte „Access Broker“ sein, sich also Zugang zu Systemen verschaffen und diesen Zugang dann verkaufen, ohne selbst einen Angriff auf das System durchzuführen. Je nachdem, wie „beliebt“ das Opfer ist, kann dieser Verkauf durchaus ein paar Tage oder sogar Wochen dauern.

In den letzten Jahren lässt sich beobachten, dass der Zeitraum zwischen einer erfolgreichen Kompromittierung von Systemen und den Folgen eines Angriffs, also beispielsweise der Verschlüsselung von Systemen, tendenziell immer kürzer wird. Schließlich ist jeder Tag des Wartens ein Risiko für die Angreifenden, entdeckt zu werden. Es ist jedoch nach wie vor nicht ungewöhnlich, dass sich Angreifende teils schon Wochen oder Monate im System befinden, bevor der tatsächliche Angriff ausgeführt wird. Oft nutzen Angreifende diese Zeit, um sich im System „einzunisten“: Backdoors oder anderes Tooling zu installieren, Daten abziehen usw.

Besonders in Fällen von Industriespionage sehen wir oft Angreifende, die sich jahrelang im System befanden, denn oft ist das Ziel hier weniger offensichtlich destruktiv. Sichtbar wird diese Angriffsform oft erst, wenn Konkurrenten auf dem Markt über Abläufe in der Firma oder auch Forschungsergebnisse Bescheid zu wissen scheinen, und bei-

spielsweise ähnliche Produkte auf den Markt bringen. Unser Rekord für den Zeitraum zwischen dem ersten Eintritt der Angreifenden und seiner Entdeckung liegt bei ganzen 14 Jahren.

Um einen Angriff nachvollziehen und auch nachweisen zu können, ist es also meistens nötig, mehrere Wochen oder sogar Monate in die Vergangenheit blicken zu können. Wurden Logs für den Zeitraum eines Angriffs bereits gelöscht, fehlen uns an dieser Stelle Beweismittel, und es wird schwer oder teilweise sogar unmöglich, Aussagen über den Ablauf des Angriffs zu treffen.

Das Vorhalten von Logs stellt uns vor ähnliche Probleme wie das Sichern von Backups. Aus forensischer Sicht ist mehr natürlich besser. Aber schnell arbeiten wir mit großen Datenmengen, auch Logdaten können sich gerne mal in die Terrabytebereiche bewegen, wenn wirklich alles vorgehalten werden soll. Deswegen sollte sich für alle IT-Systeme neben einer Backupstrategie auch eine Strategie zur Speicherung von Logs überlegt werden. Welche Logs sind besonders wichtig? In welchem Abstand werden sie gesichert, und wo? Welche Logs sollten vielleicht auch über mehrere Jahre vorgehalten werden?

Eine allgemeingültige Antwort kann hier nicht gegeben werden; zumindest aus forensischer Sicht bietet es sich aber an, sich an den Logarten aus den vorherigen Kapiteln zu orientieren, und diese besonders für exponierte oder für die Geschäftsabläufe kritischer Systeme vorzuhalten. Als grobe Richtlinie freuen wir uns über möglichst voll-

ständige Logs für mindestens die letzten zwei Wochen. Ausgewählte Logs sollten mindestens die letzten 180 Tage vorgehalten werden.

Und auch, wenn dann doch ziemliche Datenmengen zusammenkommen können: Speicherplatz kostet nicht die Welt, und im Zweifelsfall können die Logs, bzw. die aus ihnen gewonnenen Erkenntnisse, das Unternehmen retten.

Logs sind im angegriffenen System.

Selbst wenn die Logs vorhanden sind, lange genug vorgehalten werden und schnell und einfach darauf zugegriffen werden kann, laufen wir immer wieder in Probleme. Befinden sich die Logs auf dem angegriffenen System, können Angreifende diese verändern, um der IT-Forensik das Leben schwer zu machen. Logs können gelöscht oder manipuliert oder auch als Teil einer Ransomware-Attacke mit verschlüsselt werden.

Der Umgang mit veränderten Logs ist Teil der IT-Forensik und oft können wir zumindest die Veränderungen nachvollziehen. Doch einfacher macht es die Arbeit nicht, und am Ende gilt, was immer gilt: Gelingt es uns nicht, die ursprünglichen Logs wiederherzustellen, verlieren wir zentrale Beweismittel. Gerade wenn Logs durch Handlungen der Angreifenden verändert wurden, müssen wir auch davon ausgehen, dass sich in eben diesen Logs wichtige Informationen über den Angriff befunden haben.

Entsprechend relevante Logs sollten deswegen getrennt gesichert werden. Im Wesentlichen bietet es sich an, diese ähnlich wie Backups zu behandeln.

Was können Entwickler*innen schon heute tun?

Viele der angesprochenen Logarten liegen eher bei Systemadmins, Netzwerkadmins oder Betriebsteams. Doch die Welt der Softwareentwicklung hat sich geändert, mit DevOps und cross-funktionalen Teams sind auch „klassische“ Entwickler*innen näher an Themen wie Infrastruktur und Betrieb gekommen.

Und auch in der reinen Applikationsentwicklung lohnt es sich, ein paar Gedanken mehr für das Logging zu verwenden und auch gezielt über einen Angriffsfall nachzudenken. Welche Änderungen könnten einen Befall der Applikation anzeigen? Wie könnten Angreifende die Software benutzen und welche Spuren würden sie dabei hinterlassen? Was sind größere Änderungen, die protokolliert werden sollten? Und wo und wie lange werden die Application-Logs vorgehalten?

Diese Informationen in einem Angriffsfall schnell zur Hand zu haben, kann wichtige Bausteine für die Rekonstruktion des Angriffes liefern.

Fazit

Angreifende nutzen, was sie in die Finger bekommen können. IT-Sicherheit sollte deswegen an jeder Stelle mitgedacht werden. Über den Ernstfall nachzudenken ist nicht immer schön, aber es verbessert unsere Situation deutlich, wenn er eintritt. Themen wie das Vorhandensein von Backups oder etablierte Abläufe für den Krisenfall spielen hier

eine große Rolle. Doch auch über die Loggingstrategien lässt sich einiges verbessern, denn sie ermöglichen der IT-Forensik eine bessere Arbeit und damit einen sichereren und schnelleren Wiederaufbau der Systeme.

*Antonia Schmalstieg unterstützt Entwicklungsteams dabei, sichere Software zu entwickeln und ist als IT-Forensikerin zur Stelle, wenn doch etwas schiefgeht. Wenn sie nicht in der Technik gräbt, motiviert sie andere, sich mit dem Thema zu befassen – denn IT-Sicherheit geht jede*n etwas an!*
antonia.schmalstieg@codecentric.de

Referenzen

[1] <https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Lageberichte/Lagebericht2023.pdf>

Die Tiefsee

„TIEFSEE-ÖKOSYSTEME SIND TEIL EINES GIGANTISCHEN OZEANISCH-PLANETAREN REGULIERUNGSSYSTEMS. DURCH DEEP OCEAN MIXING (VERMISCHUNG VON WASSER UNTERSCHIEDLICHER TEMPERATUR UND DICHTEN) TRÄGT DIE TIEFSEE DAZU BEI, MEERESSTRÖMUNGEN ANZUTREIBEN. DIE TIEFSEE ABSORBIERT AUCH WÄRME UND CO₂ AUS DER ATMOSPHÄRE. AUF DIESE WEISE FUNGIERT DIE TIEFSEE ALS PUFFER FÜR DIE AUSWIRKUNGEN DES KLIMAWANDELS. ALLERDINGS KANN DER OZEAN NUR EINE BEGRENZTE MENGE WÄRME UND NUR EINE BEGRENZTE MENGE CO₂ SPEICHERN. SOBALD IHRE AUFNAHMEFÄHIGKEIT ÜBERSCHRITTEN IST, KANN DIE TIEFSEE DIESE LEISTUNGEN NICHT MEHR ERBRINGEN. WENN DIESE SYSTEME BESCHÄDIGT WERDEN, KÖNNTEN SIE DEN KLIMAWANDEL VERSCHLIMMERN [...]“

Katherine Beem, Columbia Climate School

DER SOFTWERKER - DAS MAGAZIN

REDAKTION:

Dr. Diana Kupfer

ART DIRECTION & DESIGN:

Valentina Arambasic

HERAUSGEBER:

Uwe Friedrichsen, CTO

LESERSERVICE:

softwerker@codecentric.de

ISSN 2366-7729

Erscheinungsdatum: 06/2024

codecentric AG, Hochstraße 11, 42697 Solingen

E-Mail: info@codecentric.de

Copyright codecentric AG 2024

www.dersoftwerker.de

 codecentric

Hinweis: Die Entscheidung, geschlechtergerechte Sprache in ihren Artikeln zu verwenden, obliegt den jeweiligen Autorinnen und Autoren.

@codecentric