Agile Transformation mit DevOps, Cloud und Microservices

Die Verwandlung eines Monolithen in agile Bauelemente

Alle größeren Unternehmen und Behörden pflegen einen lebenden Dinosaurier in Form einer großen, komplexen und erfolgskritischen Software. Groß steht für viele Millionen Zeilen Quellcode. Inhärent komplexe Prozesse sollen optimal durch IT-Systeme unterstützt werden. Erfolgskritisch heißt nicht weniger, als dass ein signifikanter Ausfall der Software nicht nur schnell auf die Titelseite renommierter Tageszeitungen, sondern das Unternehmen oder die Behörde in eine echte Krise führen kann. Auch Unternehmen, die bereits agil arbeiten, stehen für solche gewachsenen und komplexen IT-Systeme noch am Anfang einer Transformation, um schneller auf neue Anforderungen reagieren zu können. Wir fassen unsere Erfahrung zusammen, wie sich in drei Schritten durch Softwareentwicklung 4.0 diese Dinosaurier in einen agilen intelligenten Bienenschwarm transformieren lassen.

Softwareentwicklung 4.0 – Der Standard in der Neuentwicklung

Unter Softwareentwicklung 4.0 verstehen wir das perfekte Zusammenspiel von Agilität, DevOps, Cloud und Microservices. Agilissimo (=maximal agil) kann nur das Unternehmen werden, welches alle vier Bausteine beherrscht, sodass sie perfekt ineinandergreifen (**siehe Abbildung 1**). Das wurde im Artikel „Maximale Agilität mit Softwareentwicklung 4.0“ [PRO2019] in dieser Zeitschrift detailliert vorgestellt.

Der Baustein *Agilität* umfasst alle Aspekte einer agilen Softwareentwicklung. Dazu gehören:

- Teams, die agile Werte leben und agile Methoden wie SCRUM, Kanban oder SAFe anwenden,
- Cross-funktionale Teams mit Fachbereich, Entwicklung inklusive Test und Betrieb (BizDevOps),
- Produktorientierung und -verantwortung statt klassischer Projekte,
- Product Owner mit der Fähigkeit und dem Mandat (schnell) zu entscheiden.

DevOps umfasst zwei Hauptaspekte. Durch eine verbesserte Zusammenarbeit werden Brüche zwischen Entwicklung und Betrieb entfernt. „You build it, you run it!“ kann eine Ausprägung dieser Zusammenarbeit und des EIN-Team-Gedankens werden. Der zweite Hauptaspekt besteht in der Automatisierung aller Aktivitäten rund um das Bauen, Testen und Installieren von Software in CI/CD-Pipelines.

Der Baustein *Cloud* bedeutet nicht(!), dass die Software in einer öffentlichen Cloud betrieben werden muss, sondern zielt auf

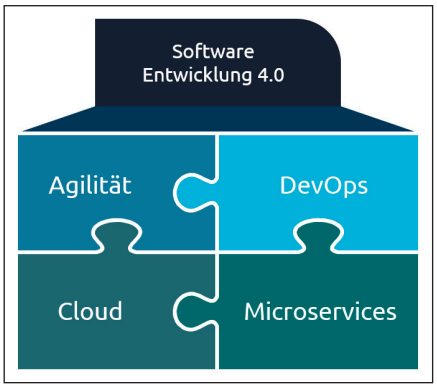

Abb. 1: Die Bausteine von Softwareentwicklung 4.0

die Nutzung von Fähigkeiten und Technologien der Cloud wie zum Beispiel Containerisierung, Self Services, Infrastructure as Code, IT-Leistung wie Strom aus der Steckdose usw. Dazu gehört auch, dass Software Cloud-ready oder Cloud-native [TH2018] entwickelt werden muss und dass dies sowohl Cloud-agnostisch [PRO2021] als auch spezifisch für einen bestimmten Cloud-Anbieter geschehen kann.
Microservices stehen in der Softwareentwicklung 4.0 synonym für eine produktorientierte Entwicklung. Ein Produkt beziehungsweise Microservice liefert einen eindeutigen Geschäftsnutzen und wird durch *ein* Team verantwortlich umgesetzt. Die Größe eines Microservice spielt dabei nicht die entscheidende Rolle. Ein IT-Produkt kann nicht nur ein sehr kleiner Service, sondern sehr wohl auch ein größerer unabhängig installierbarer Service sein, der oft auch eine grafische Benutzeroberfläche inkludiert und auch vielfach als „Self-Contained System" bezeichnet wird. Unabhängig installierbar impliziert außerdem möglichst lose gekoppelt und gut eigenständig wartbar und testbar.

Neue und insbesondere kleinere Softwareprodukte werden mittlerweile fast immer über Softwareentwicklung 4.0 erstellt. Dabei müssen sicherlich nicht für jedes Softwareprodukt täglich neue Features in den Produktivbetrieb gebracht werden. Die „Studie IT-Trends 2022" von Capgemini [CAP2022] zeigt aber, dass für fast jeden zweiten befragten Teilnehmer die Verkürzung der Entwicklungszeit neuer Produkte und Services eine der derzeit wichtigsten Anforderungen der Geschäftsleitung ist.

Lebende Dinosaurier

Die Verkürzung der Reaktionszeit gilt umso mehr für die Pflege und Weiterentwicklung der lebenden Dinosaurier der IT-Systeme, die ihre Zeit technologisch und bezüglich ihres Entwicklungsprozesses überlebt haben, aber immer noch oft das schlagende IT-Herz unserer Kunden bilden. Wir sprechen von *dem* geschäftskritischen IT-System,

- ohne das bei einem Autobauer kein Auto bestellt, gebaut und ausgeliefert werden kann,
- ohne das beim Logistikdienstleister keine Sendung beauftragt und transportiert werden kann oder
- ohne das bei einem Versicherer keine Rente beantragt, berechnet und ausgezahlt werden kann.

Diese IT-Systeme haben den Erfolg des Unternehmens in der Vergangenheit entscheidend ermöglicht und sollen diesen auch in Zukunft sichern. Sie wurden individuell entwickelt, um sich von Wettbewerben differenzieren zu können, und haben deswegen auch eine hohe inhärente Komplexität. Sie bestehen aus vielen Millionen Zeilen Code und die Ablösung durch eine vollständige Neuentwicklung mit Big-Bang-Einführung ist schlichtweg aus Risiko- und Kostengründen nicht möglich.
Diese Software-Dinosaurier leben in einer VUCA-Welt (VUCA = Volatility, Uncertainty, Complexity und Ambiguity), in der um sie herum ein Schwarm von schnell anpassbaren Services kreist. Eine neue Idee kann im Schwarm schnell umgesetzt werden, prallt dann aber immer öfter auf zu lange Entwicklungszeiten der Dinosaurier, ohne die die Idee nicht vollständig umgesetzt werden kann. Genau auf diese Dinosaurier zielt daher häufig die Verkürzung der Entwicklungszeit aus der IT-Trends-Studie.
Wie macht man aus so einem monolithischen Dinosaurier einen agilen, intelligenten Bienenschwarm (**siehe Abbildung 2**), in dem viele kleine Arbeiterbienen (Microservices) scheinbar unabhängig ihre Arbeit verrichten und sehr schnell auf neue Anforderungen reagieren?

Metamorphose eines Dinosauriers

Ist es überhaupt möglich, Dinosaurier-Software in einen agilen Bienenschwarm von Services zu verwandeln? Womit soll man anfangen? Welche Agilität kann man mit welchen Maßnahmen erreichen? Um diese Fragen zu beantworten, haben wir Projekte befragt, die sich in dieser Transformation befinden oder sie schon weitestgehend hinter sich haben.
Als Maß für die Agilität schauen wir auf die Anzahl der Releases pro Jahr. Als Release zählt nur, wenn es aus fachlicher Sicht etwas Neues enthält. Eine erneute Installation zur Fehlerbehebung ist also kein Release. Releases pro Jahr sind natürlich nicht der einzige Maßstab für Agilität. In unserer Analyse haben wir jedoch festgestellt, dass dieser objektive und messbare Wert ein vertrauensvoller Indikator ist, zeitliche Auswirkungen von Maßnahmen der agilen Transformation zu bewerten und zu vergleichen.
Auch wenn sich detaillierte Vorgehensweisen je nach Ausgangslage und konkretem Ziel unterscheiden, können wir aus unserer Projektumfrage ein klares Muster bei den Transformationsschritten und Meilensteinen ableiten und empfehlen nun eine Transformationsstrategie in drei Schritten:

- Baue das richtige Produkt.
- Baue das Produkt richtig.
- Optimiere den „Business Value".

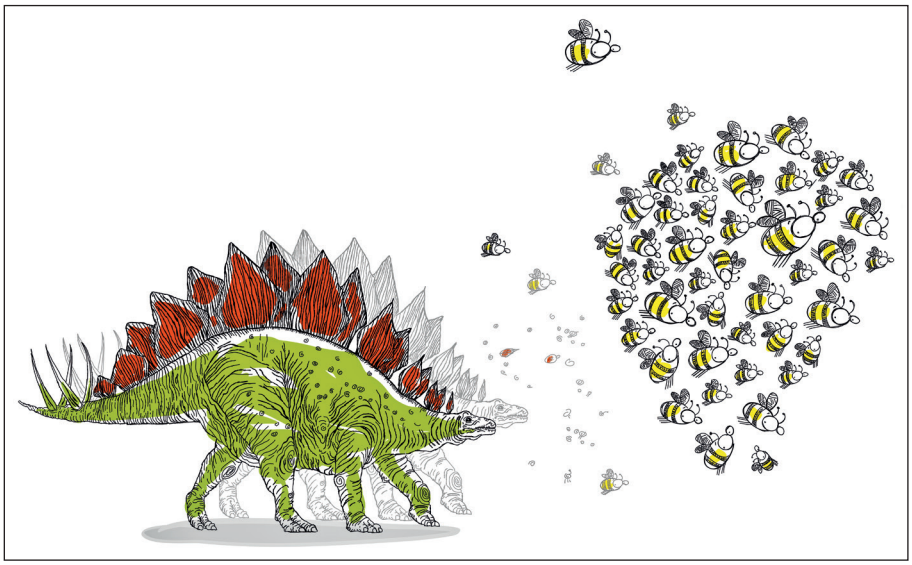

Abb. 2: Kann man einen Dinosaurier in einen intelligenten Bienenschwarm verwandeln?

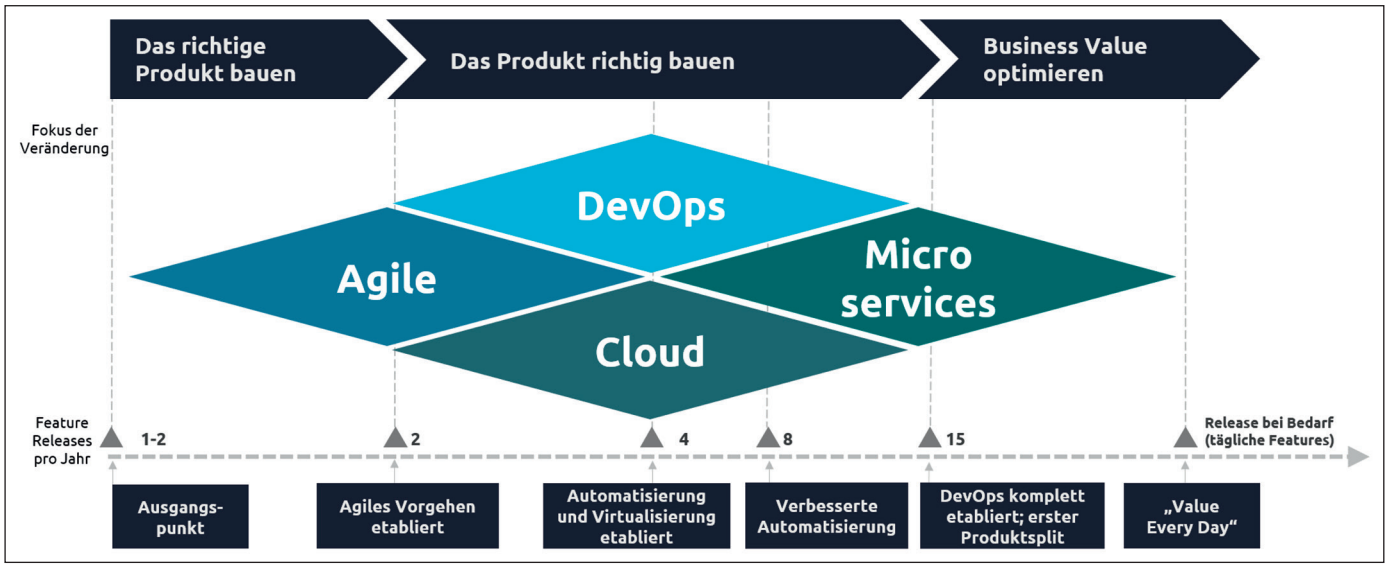

Abb. 3: Agile Transformation von Software-Dinosauriern mit Softwareentwicklung 4.0

Die Umsetzung dieser drei Schritte (**siehe Abbildung 3**) geht einher mit der vollständigen Einführung von Softwareentwicklung 4.0, wie im Folgenden detailliert beschrieben.

„Baue das richtige Produkt" durch agile Arbeitsweise

Als Erstes wird der Software-Dinosaurier auf den richtigen Weg gebracht. Wenn er sich auch langsam bewegt, soll er sicher in die richtige Richtung gehen. Der erste Schritt dazu ist die Änderung des Projektvorgehens hin zu agilen Arbeitsweisen.
Sobald die Einführung von agilen Arbeitsweisen von den Beteiligten als erfolgreich eingeschätzt wird, geben Fachbereiche das Feedback, dass sich nach einer Zeit der Anpassung Qualität und Geschwindigkeit deutlich verbessert haben. Das Geschwindigkeitsempfinden steht dabei im Widerspruch zur objektiven Anzahl der Releases, die sich häufig wenig bis gar nicht geändert hat. Die Projekte liefern weiterhin maximal zweimal im Jahr. Sie sind trotzdem schneller, weil sich die Zeit von der Idee bis zur Produktivsetzung zum Teil erheblich verkürzt. Eine direktere und vertrauensvolle Zusammenarbeit zwischen IT und Fachseite macht dies möglich. Zum Beispiel können langwierige Spezifikationsphasen durch ein gemeinsames Verständnis erheblich verschlankt werden.
In einem Projekt brauchte man zum Beispiel ursprünglich von der Idee bis zur Produktivsetzung zwei Jahre, wovon nur 2 bis 3 Monate auf die Entwicklung entfielen. Das zeigt das Potenzial einer agilen Prozessverbesserung. Das zeigt aber auch, dass sich dieser Erfolg bei der agilen Transformation nur einstellt, wenn sie nicht allein als Änderung des Entwicklungsvorgehens verstanden wird. Vielmehr spielt die enge Zusammenarbeit aller Beteiligten inklusive Fachbereich und das Annehmen eines agilen Mindsets die wesentliche Rolle. Als sehr hilfreich sehen Teams hier den Einsatz von Agile Coaches und das Hinzunehmen von Mitarbeitern mit agilen Erfahrungen, die den Rückfall in alte Denkmuster aufzeigen können.
Auch haben wir in diesem Stadium häufig eine Qualitätsverbesserung beobachtet. Das lässt sich größtenteils durch die verbesserte Zusammenarbeit zwischen IT und Fachseite erklären. Entwickler verstehen den Nutzen von Anforderungen und kurze Sprintzyklen helfen, gemeinsam zu lernen, wie das richtige Produkt aussehen sollte.

„Baue das Produkt richtig" durch Nutzung von Cloud und DevOps

Zwei Releases pro Jahr entsprechen auch bei insgesamt schnelleren Zyklen in der Regel noch nicht den zeitlichen Anforderungen zur Umsetzung neuer Geschäftsideen. Unser Dinosaurier muss schneller werden. Er benötigt mehr Füße zum schneller Laufen oder Flügel zum Fliegen. Spätestens wenn sich ein agiles Vorgehen etabliert hat, richtet sich der Fokus anschließend auf Effizienzgewinn durch Automatisierung. Dies kann auch schon früher gestartet werden, wenn man sich neben der agilen Transformation noch mehr Veränderungen gleichzeitig zutraut.
Für die anstehende Veränderung durch die Automatisierung kommen die Bausteine DevOps und Cloud ins Spiel. Automatisierte CI/ CD-Pipelines werden aufgebaut oder verbessert. Außerdem wird Virtualisierung für das flexible Bereitstellen von Build-Infrastruktur und Testumgebungen genutzt. Dadurch werden manuelle Schritte eingespart und es wird schneller, früher und flexibler integriert und getestet. Dadurch erreichen die Projekte als ersten Meilenstein vier Releases pro Jahr, ohne dass die Qualität leidet.
Für den nächsten Meilenstein von ca. acht Releases pro Jahr muss vor allem die Effizienz in der Qualitätssicherung erheblich gesteigert werden. Spätestens jetzt muss das Testen komplett in den cross-funktionalen Teams erfolgen. Der Großteil der Regressionstests muss automatisiert sein und die Tests sind kontinuierlich auszu-

Literatur & Links

[CAP2022] Capgemini, Studie IT-Trends 2022 – IT wird Kern der Wertschöpfung, siehe: **https://www.capgemini.com/de-de/service/it-trends-studie/**

[PRO2019] K. Prott, Maximale Agilität mit Softwareentwicklung 4.0, in: OBJEKTSpektrum, 01/2020, siehe: **https://www.capgemini.com/de-de/resources/softwareentwicklung40-objektspektrum/**

[PRO2021] K. Prott, Entwickeln Sie noch Cloud-Agnostic oder schon Cloud-Native?, siehe: **https://www.capgemini.com/de-de/2021/02/cloud-agnostic-cloud-native/**

[TH2018] 10 key attributes of cloud-native applications, siehe: **https://thenewstack.io/10-key-attributes-of-cloud-native-applications/**

[TW2019] St. Tilkov, E. Wolff, Microservices ersetzen den Monolithen – Ameisenprinzip, in: IX, Ausgabe April 2019

führen und zu pflegen. Je nach Ausgangssituation kann dies mit erheblichen Investitionen verbunden sein. Eine gute Praxis ist das schrittweise Ausrichten der Testautomatisierung an der Testpyramide, um die Wartbarkeit der Tests langfristig sicherzustellen. Dazu kann auch gehören, bestehende UI-Tests durch Tests auf API-Ebene gezielt zu ersetzen.
Technische Schulden können das Erreichen des Meilensteins blockieren. Wenn die Wartbarkeit des Systems durch diese bereits stark eingeschränkt ist, muss erst ein gezielter Abbau, zum Beispiel durch Refactorings, erfolgen, bevor eine Automatisierung und damit Beschleunigung der Lieferung ohne Qualitätsverlust zu erreichen ist.

„Optimiere Business Value" durch kurze Zyklen mit Microservices

Wenn auch acht Releases pro Jahr noch nicht genug sind, um neue Ideen an den Markt zu bringen, ist der nächste Meilenstein ungefähr 15 Releases pro Jahr, das heißt, man liefert fast jeden Sprint aus.
Jetzt geht es an die Zerlegung unseres Dinosauriers in kleine Arbeiterbienen. Aus einem Dinosaurier muss ein intelligenter Bienenschwarm werden. Man kann sich vorstellen, dass diese Metamorphose für die meisten Software-Dinosaurier sehr aufwendig und teuer werden wird. Deshalb ist genau zu prüfen, was als Arbeiterbiene sprich Microservice und was gegebenenfalls durch einen kleineren Flugsaurier sprich größerem Produkt (= SCS - Self-Contained System) herausgelöst wird (siehe auch [TW2019]).
Für häufigere Lieferungen müssen die Teams als Erstes weitestgehend unabhängig voneinander arbeiten. Dazu ist der Monolith in kleinere unabhängige Produkte zu zerteilen. Wir brauchen an dieser Stelle noch keine komplette Microservices-Architektur. Eine Unterteilung nach Domänen oder Subdomänen, wonach wenige Teams für ein Produkt zuständig sind, ist ausreichend.
Die Aufteilung in unabhängige Produkte sollte man in der Regel erst durchführen, wenn Agilität, DevOps und Cloud etabliert sind. Dies sind Voraussetzungen dafür, dass man durch die Verkleinerung des Monolithen eine Verkürzung der Innovationzyklen erreicht. Man kann die Aufteilung in Produkte beschleunigen, wenn man sie im vorherigen Transformationsschritt parallel zur Einführung von Cloud und DevOps vorbereitet. Zum Beispiel können Refactorings schon früh dafür sorgen, dass die späteren Produkte bereits innerhalb des monolithischen Software-Dinosauriers lose gekoppelt sind.
Mit der Aufteilung auf viele unabhängig aktualisierbare Produkte und den wesentlich häufigeren Lieferungen wird auch notwendig, dass mehr Verantwortung für den Betrieb in die Teams wandern muss, das heißt, man nähert sich einem „You build it – you run it". Die häufigen Deployments von Releases erfordern mehr Automatisierung für den Betrieb inklusive von Prozessen wie Monitorings. Das Aufsetzen und die kontinuierliche Verbesserung lässt sich am besten in den Teams erreichen.
Wenn die Transformation so weit gekommen ist, dass jeder Sprint geliefert wird, kann dezentral weiter optimiert werden. Große Produkte, die sich häufig ändern und noch immer zu unflexibel sind, werden in Microservices zerlegt, um sie bei Bedarf täglich liefern zu können.
Die verkürzte Time-to-Market an den fachlich relevanten Stellen versetzt vor allem in die Lage, Business Value zu optimieren. Hierfür sind kurze Zyklen nötig, um neue Ideen umzusetzen, die Wirkung automatisiert zu messen und schnell auf die Ergebnisse reagieren zu können.

Konkrete Projekterfahrung

Wir haben gelernt, dass es einen Weg gibt, um auch große geschäftskritische Software-Dinosaurier in einen agilen Bienenschwarm zu transformieren und damit kurze Innovationszyklen zu etablieren. Der Weg ist schwierig und zeitintensiv. Die mehrjährige Metamorphose ist in den uns bekannten Projekten noch nicht abgeschlossen und befindet sich je nach Dinosaurier im Stadium von 4 bis 15 Feature-Releases pro Jahr. Einer unserer Kunden hat zum Beispiel für sein unternehmenskritisches Kernsystem bereits die agile Arbeitsweise weitgehend eingeführt und befindet sich aktuell in der Phase „Baue das Produkt richtig" durch Nutzung von Cloud und DevOps. Es existiert auch schon eine Planung für eine Zerschlagung des Monolithen in vier Teile, aber eine darüber hinausgehende Aufteilung in Microservices ist noch nicht angedacht.

Fazit

Heißt das jetzt, dass alle Dinosaurier sterben müssen? Diese Frage können und müssen wir auch gar nicht beantworten. Unsere Aussage ist nicht, dass jeder heute entscheiden sollte, seine großen geschäftskritischen Systeme in Microservices zu zerlegen. Die Frage „Muss ich unbedingt täglich liefern?" ist nicht die entscheidende. Wichtig ist die Einschätzung, ob man mit seinen derzeitigen Innovationszyklen der Software-Dinosaurier mittelfristig anpassungsfähig genug ist. Zusätzlich sollte man sich ehrlich fragen, ob man nicht täglich Chancen verpasst, weil keiner sich traut, bestimmte Änderungen überhaupt zu erwägen.
Die Antworten auf diese Fragen sind die Gründe, sich auf den Weg der Transformation oder zum nächsten Meilenstein zu begeben. Wenn man die Entscheidung hinauszögert und später feststellt, dass man sich nicht mehr schnell genug anpassen kann, ist es eventuell zu spät. Ob man den Weg ganz bis zu Ende geht und dann den kompletten Bienenschwarm hat oder ob der schnellere Dinosaurier, der in die richtige Richtung läuft, vollständig ausreicht, muss nicht am ersten Tag der langen Reise entschieden werden. ||

Die Autoren

Dr. Karl Prott
(karl.prott@capgemini.com)
verantwortet seit über 20 Jahren als IT-Architekt bei Capgemini die Umsetzung unternehmenskritischer IT-Systeme. Seit geraumer Zeit ist er Fan von Softwareentwicklung 4.0 und unterstützt seine Kunden bei entsprechenden Transformationen.

Kathrin Potzahr
(kathrin.potzahr@capgemini.com)
ist seit über 20 Jahren Architektin in der Softwareentwicklung geschäftskritischer IT-Systeme. Sie berät Unternehmen, wie man Qualität bei kurzen Lieferzyklen frühzeitig sicherstellt und kontinuierlich hält.