h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Hochschule Darmstadt

– Fachbereich Informatik–

Antriebe der Veränderung - Aus CI/CD einer Bank zur Digitalen Transformation mit einem Ökologischen Veränderungsmodell für die IT

Abschlussarbeit zur Erlangung des akademischen Grades

Bachelor of Science (B.Sc.)

vorgelegt von

Muhammed Selim Sinan

Matrikelnummer: 750907

Referent	:	Prof. Dr. Peter Ebinger
Korreferent	:	Prof. Dr. Stefan Zander

ERKLÄRUNG

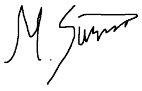
Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 01. September 2020



Muhammed Selim Sinan

ZUSAMMENFASSUNG

Als frühe Anwender von IT-Systemen etablierte sich in Banken eine veraltete IT-Architektur, die es schafft über Jahre einer ganzen Reihe von Innovationen erfolgreich Stand zu halten. Für CI/CD bedingt es eine Umgebung mit agilen Ansätzen. Die unflexible IT-Architektur wird zusätzlich mit ineffizienten Prozessen und Risikomaßnahmen beschränkt und in Folge gegenüber Veränderungen paralysiert. Handlungsunfähigkeit ist vor einer unvermeidbaren Kollision fatal. Die Umsetzung einer der effektivsten Innovationen der letzten Jahre, dem Cloud-Computing ruft in der Legacy-IT von Banken Verwerfungen auf, die eine Transformation der gesamten Organisation unvermeidbar machen.

ABSTRACT

As early users of IT systems, banks have established an outdated IT architecture that has been able to successfully withstand a whole series of innovations over the years. For CI/CD it requires an environment with agile approaches. The inflexible IT architecture is additionally limited by inefficient processes and risk measures and is consequently paralyzed against changes. The inability to act before an unavoidable collision is fatal. The implementation of one of the most effective innovations of recent years, cloud computing, is causing disruptions in the legacy IT of banks, making a transformation of the entire organization unavoidable.

INHALTSVERZEICHNIS

ABBILDUNGSVERZEICHNIS

ABKÜRZUNGSVERZEICHNIS

BaFin Bundesanstalt für Finanzdienstleistungsaufsicht

BSI Bundesamtes für Sicherheit in der Informationstechnik

KWG Kreditwesengesetz

MaRisk Mindestanforderungen an das Risikomanagement

BAIT Bankaufsichtliche Anforderungen an die IT

ISO International Organization for Standardization

IKS Internes Kontrollsystem

IDV Individuelle Datenverarbeitung

CI Continuous Integration

CD Continuous Delivery

SCM Source Code Management

SaaS Software as a Service

IaaS Infrastructure as a Service

IaC Infrastructure as Code

PaaS Platform as a Service

AWS Amazon Web Services

CEO Chief Executive Officer

VMs Virtuelle Maschinen

SOA serviceorientierte Architektur

SEU Softwareentwicklungsumgebung

JDK11 Java Development Kit 11

GCP Google Cloud Plattform

GKE Google Kubernetes Engine

SCM Source Code Management

DLT Distributed-Ledger Technologie

EBA European Banking Authority

ITM IT-Management

IKT Informations- und Kommunikationstechnik

SdB Software-defined Business

IDT Innovate-Design-Transform

Teil I

THESIS

1 EINLEITUNG

Brockhoff [4] formuliert ein Ergebnis, dass in vielen Banken heterogene, unflexible und eher zufällig gewachsene Architekturen dominieren. Das Ergebnis leitet sich aus dem historischen Kontext ab. Banken sind frühe Anwender von IT-Systemen, woraus sich über die Jahre eine komplexe IT-Architektur etablierte.

Das Ergebnis aus der veralteten und paralysierten IT-Architektur von Banken lässt sich mit der Bezeichnung Legacy-IT [9, 32] zusammenfassen. Innovationsdruck stößt auf Veränderungswiderstand und einer der effektivsten Innovationen der letzen Jahre, das Cloud-Computing hinterfragt die gesamte Architektur von Organisationen mit Legacy-IT.

Die Anpassung von bewährten Standardlösungen auf eine problematische Umgebung [4, 5, 32] lindert oftmals nur die Symptome und ist, gerade in den Systemen der Legacy-IT mit hohem Aufwand verbunden. Ein grundlegendes Re-Design nach Bussmann u. a. [5, S. 27] ist durch Cloud-Technologie sehr greifbar.

Hierfür entstehen weitere Einschränkungen, die aus den externen Rahmenbedingungen rühren. Als eine Einschränkung könnten die regulatorischen Anforderungen [13], [14], [2] angesehen werden [3]. Nach der Betrachtung von diesen kann festgestellt werden, dass aus Sicht eines Kreditinstiuts eine Wechselwirkung zwischen Veränderungsdruck und Widerstand herrscht. Eine wichtige Frage ist daher, worin die Ursachen für eine mangelnde Durchsetzungsfähigkeit für Veränderungen liegen.

In der IT-Architektur der Institute entstehen Probleme[1] und einschränkende Faktoren, die nicht nur Auswirkungen auf den Betrieb, sondern auch auf die Entwicklung der IT-Systeme haben. Für die Entwicklung und Betrieb von IT-Systemen sind in jedem Institut Regelprozesse zu definieren [13].

Zum Beispiel müssen Standardanwendungen von Entwicklern vor ihrem Einkauf nicht nur ausreichend getestet, sondern auch auf die jeweiligen internen Prozesse angepasst werden. Im Ansatz Software as a Service (SaaS) ist die Auslagerung von IT-Ressourcen eine gängige Praxis zum Betreiben von Anwendungen. Dadurch werden zusätzliche Schritte im Risikomanagement der Institute angefordert [13].

Gängige Standards der Softwareentwicklung befinden sich mit neuen Technologien, Ansätzen und Anwendungen im ständigen Wandel. Vor ihrem geplanten Einsatz müssen mögliche Auswirkungen auf die Kontrollverfahren der Institute analysiert werden [13]. Die IT-Architektur ist, bedingt durch Impulse aus neuen Technologien [5], ständigem Innovationsdruck

1 vgl. [7]

ausgesetzt und muss sich "kontinuierlich justieren" [5] können.

Die Kredit- und Finanzdienstleistungsinstitute in Deutschland können sich Veränderungen nicht mehr entziehen. Insbesondere ist das aufgrund der Disruptivität von einigen Technologien [11] und Ansätzen bedingt. Eine bevorstehende digitale Transformation von Banken und Finanzdienstleistern bietet für anpassungsfähige Institute einzigartige Möglichkeiten sich neu zu etablieren [20]. Gleichzeitig besteht auch bei Nichtbeachtung von Technologietrends die Gefahr einer Verdrängung von kleineren und effizienten Betrieben, die schnell auf die Impulse reagieren.

Lloyd Blankfein verkündete, als Vorsitzender und Chief Executive Officer (CEO) von Goldman Sachs *"Wir sind ein Technologieunternehmen. Wir sind eine Plattform."* [20]. Tatsächlich könnte aus Sicht der Digitalen Transformation eine Bank als Plattform für Finanzdienstleistungen definiert werden und neue Wertschöpfungsketten erzeugen. Bevor die Opportunitäten in Angriff genommen werden können stellt sich die Frage welche digitalen Anlagen priorisiert werden sollen.

1.1 MOTIVATION

Die nachfolgende Arbeit ist aus der Frage entstanden, wie man die CI Plattform Jenkins in einer Bank skalierbarer und flexibler gestalten könnte.

Anfänglich wurde der Fokus auf CI/CD gerichtet und anschließend Themen wie Cloud-Computing, Container-Orchestrierung und Microservices vorgenommen. Dabei wurde sich mit der Entwicklung von Docker-Images für Jenkins Agenten[2] der Continuous Integration (CI) Plattform Jenkins beschäftigt. Ziel war hierbei eine flexibel einsetzbare Umgebung zum Ausführen von verteilten Builds[3].

Ansätze wie Cloud-Computing, Infrastructure as a Service (IaaS), SaaS, DevOps und Agilität werden zum Standard für die Softwareentwicklung. Banken benötigen für die Umsetzung von neuen Technologien und Methoden umfangreiche Anpassungen ihrer Prozesse. Selbst eingekaufte Standardanwendungen unterliegen nicht selten umfangreichen Analysen und Anpassungen mit hohen Kosten.

Es sollte sich bezüglich der internen Kontrollverfahren und IT-Strategie die Frage gestellt werden, wie mit ständigen Veränderungen in der IT umzugehen ist. Ein angemessenes Risikomanagement sollte Veränderungen fördern. Eine mangelnde Anpassungsfähigkeit und Geschwindigkeit stellt möglicherweise ein höheres Risiko dar.

Agile Prinzipien und Best-Practices müssen durch diese umfangreichen Anpassungen gegenüber der Legacy-IT klein bei geben. Die Legacy-IT trotzt der Innovation. Daher hat ein Innovator der Legacy-IT zu trotzen.

2 Slave innerhalb einer Master-Slave Konfiguration in Jenkins [23, 29]
3 In diesem Zusammenhang die Erstellung von Software mit dem Build-Tool Maven

1.2 ZIEL DER ARBEIT

Die nachfolgende Arbeit soll sich mit den Hintergründen der bereits länger bekannten Einschränkungen der IT-Architektur von Banken [4, 5, 9, 20, 28, 32] beschäftigen. Diese Einschränkungen haben Auswirkungen auf den Betrieb für agile Softwareentwicklung in der IT von Banken und es stellt sich die Frage:

> "Welche Technologien und Ansätze eignen sich für eine skalierbare und flexible Softwareentwicklungsumgebung in Banken?"

Die Einführung von neuen Standards[4] für die Umsetzung von Veränderungsprozessen erfolgt im Finanzwesen jedoch eingeschränkt und verspätet. Für ihre Einführung unterliegen diese Ansätze umfangreichen Anpassungen, die ihre Vorteile negieren oder werden durch neue Technologien und Trends kurz nach ihrer verspäteten Einführung überflüssig. Die in [4] geforderte serviceorientierte Architektur (SOA) wurde durch Microservices ersetzt [9, S. 8of].

Daher wird als Voraussetzung für die Lösung dieser Forschungsfrage bedingt, Faktoren im Innovationsverlauf [1] für eine bessere Transformationsfähigkeit [25] der Legacy-IT in Banken zu identifizieren. Eine zweite Forschungsfrage:

> "Welche einschränkenden und antreibenden Faktoren gibt es für die IT von Banken, um Veränderungen durchzuführen?"

Daher hat die Aufgabe einen investigativen Charakter bezüglich der Problemursachen. Als Methodik wird sich am Design-Thinking Prozess orientiert 2.2.1.

Hierzu werden die Rahmenbedingungen für Veränderungen aus verschiedenen Perspektiven durchleuchtet:

- *Finanzwesen*, anhand einer Fallstudie [20] über Goldman Sachs' Transformation
- *Technologietrends*, die Impulse geben und später Disruptivität oder Diskontinuität [11] erzeugen könnten
- *Bankaufsicht*, die mit ihren Vorgaben [13], [14] einen Rahmen für das Finanzwesen festlegt, mit Auswirkung auf die IT

Aus dem genannten Zusammenhang werden die gegenseitigen Einflüsse identifiziert für einschränkende und antreibende Faktoren der Veränderungsprozesse analysiert.

Veränderungen sind aufgrund der Auswirkungen von Technologietrends unvermeidbar und müssen mittlerweile kontinuierlich erfolgen [1, 5, 11]. Daher stellt sich eine Frage worin die Lösung liegen könnte:

4 Cloud-Computing, Microservices, DevOps, CI/CD, Scrum, Agilität

> "Wie sollten Banken bezüglich ihrer IT mit kontinuierlichen Veränderungen umgehen?"

Die Arbeit beschäftigt sich daher mit einem Paradigmawechsel der regulierten Finanz-IT, um einen Impuls zu geben die nötigen Veränderungsprozesse effizient und kontinuierlich durchsetzen zu können.

1.3 AUFBAU DER ARBEIT

AUFBAU DER METHODIK Das Problem wird zuerst aus der Sicht der Softwareentwicklungsumgebung (SEU) definiert und daraus die allgemeinen Ursachen verfolgt. Von den Hintergründen aus werden unterschiedliche Perspektiven durchleuchtet und wesentliche Punkte deduktiv ausgearbeitet und bewertet.

Die Arbeit verwendet Schritte aus dem *Design-Thinking* Prozess (Kap. 2.2.1). Daher werden Kap. 3 schon induktiv einfache Ansätze für erste Lösungen ad hoc vorgeschlagen und beschrieben. Dies bezweckt eine frühe Datenerhebung mit Gestaltungsprozessen, sodass Ansätze und Anforderungen von der angenommenen Perspektive heraus vorgeschlagen werden können. Diese können sich somit gegenseitig im Analyseteil ausschließen, zur Filterung vor der Bewertung in Kap. 5. In diese fließen anschließend die festgehaltenen Ergebnisse.

Die Arbeit ist unter vier wesentlichen Bereichen gegliedert. Im Bereich Grundlagen (Kap. 2) werden bekannte Probleme und Ansätze aus verwandten Werken zusammengefasst und der aktuellen Stand der Technik dargestellt und daraus erste Ansätze mit gängigen Standards vorgeschlagen.

Im nächsten Bereich (Kap. 3 findet die Untersuchung zu den Rahmenbedingungen der Softwareentwicklungsumgebung in Banken statt. Hierin werden die einschränkenden Faktoren ermittelt. Zusätzlich werden ihre Auswirkungen und Risiken für das Geschäftsmodell untersucht. Aus der Literaturrecherche werden mögliche Ansätze für eine angemessene Umgebung für die Softwareentwicklung mit aktuellen Technologien und Methoden abgeleitet und erste Lösungsansätze vorgeschlagen. Hierbei wird auf die Regulatorik eingegangen.

Im Bereich (Kap. 4) werden Innovations- und Veränderungsprozesse als solche untersucht und analysiert. Dabei findet eine Synthese der Ergebnisse[5] statt, die anschließend analysiert und nach jeder Iteration kontinuierlich verbessert werden.

Im anschließenden Bereich (Kap. 5) findet die Bewertung und Zusammenfassung der Ergbenisse statt. Zuerst wird die Methodik und der Weg zur Lösung der Forschungsfragen rückblickend betrachtet und bewertet. Als nächstes werden die wesentlichen Ergebnisse aus dem vorigen Bereich bewertet, festgehalten und zusammengefasst.

5 aus dem Design-Thinking Prozess in (Kap. 2.2.1)

2 GRUNDLAGEN

2.1 VERWANDTE WERKE

Brockhoff [4] stellt aus erster Hand ein frühes Ergebnis zu den Problemen der heutigen Legacy-IT in Banken fest und beschreibt serviceorientierte Architekturen als ein möglicher Ansatz für Banken.

Bussmann u. a. [5] identifizieren weitere Probleme bezüglich der IT-Architektur und stellen Anwendungen, die interne Prozesse der "Operations und Administration" unterstützen und steuern als einer von zwei großen Bereichen für innovativer Lösungen fest. Sie beschreiben Impulse, die durch neue Technologien wahrgenommen werden und fordern eine kontinuierliche Anpassungsfähigkeit der IT-Architektur und eine strategische Neuausrichtung. Gezielt schlagen sie ein Re-Design der IT-Architektur vor. Dazu stellen sie Frage der Beherrschbarkeit und Wirtschaftlichkeit des Betriebs in diesem Zusammenhang und deuten auf große bevorstehende Veränderungen.

Ganswindt [18] erklärt die Hintergründe zu Innovation und ihre Bedingungen. Die eigene Innovationskraft wird hierbei als Spiegel der Wachstumschancen eines Unternehmens beschrieben. Ganswindt beschreibt disruptive Technologien als Pflicht für Trendsetter und fordert eine intelligente Produktfolgestrategie und nachhaltiges Portfoliomanagement. Große etablierte Unternehmen stehen dabei im Konflikt den Bestand zu schützen und Disruptionen selbst zu erzeugen. Disruptionen sind hierbei radikale Neuerungen die auch die eigenen Produktreihen angreifen. Das Vorantreiben von disruptiven Technologien setzt eine gewisse Risikobereitschaft, unternehmerischen Weitblick und sowohl technischen, als auch kreatives "Know-how" voraus. Nach der Entdeckerphase ist es wichtig in eine sorgfältige Nutzen-Risiko-Analyse überzugehen. Als Ziel fordert Ganswindt das immer-weiter Vorantreiben[1] der systematischen Suche nach neuen Lösungen und stellt die Frage an welcher Stelle zuerst optimiert werden soll. Zusammenfassend nennt Ganswindt als Mittel der ersten Wahl das Ausrichten unternehmerischen Handels auf die kontinuierliche Entwicklung und Platzierung von marktgerechter Innovation.

Fernández, Valle und Pérez-Bustamante [11] untersuchen die Wechselwirkungen von disruptiven und diskontinuierlichen Technologien gegenüber dominanten sowie entstehenden Technologien aus Sicht der Verhaltensökonomik. Hieraus wird auch das Problem von etablierten Organisationen klar, dass eine Handlungsunfähigkeit gegenüber disruptiven Technologien besteht, obwohl aus disruptiven Technologien eine geringere Gefahr ausgeht als von diskontinuierlichen Technologien. Dazu beschreiben sie, dass dis-

1 vgl. Kontinuitätskonzept an vielen Stellen

kontinuierliche Technologien gegenüber dominanteren Technologien erheblichen Widerstand aufweisen. Hierbei ist von einem Angriff die Rede mit einem zehnfachen Faktor gegenüber der angegriffenen dominanten Technologie.

Disterer [7] beschreibt eine problematische Inbetriebnahme von Anwendungen durch die mangelnde Beachtung von nicht-funktionalen Anforderungen bei der Entwicklung. Hierzu stellt er als Lösungsansatz eine Erweiterung eines ITILv3 Prozesses mit Quality Gates, die als Synchronisationspunkt für die Koordination zwischen Entwicklung und Betrieb dienen. Der Ansatz von Disterer gibt eine Referenz für eine bessere Koordination zwischen Entwicklung und Betrieb und dadurch eine bessere Integration von Komponenten.

Alt, Auth und Kögler [1] gehen auf die Relevanz von DevOps und weiteren Gestaltungsmethoden für das IT-Management (ITM) und einer innovationsorientierten IT-Strategie zur Steigerung der Innovationsfähigkeit ein, die sich aus den Kontinuitätskonzepten der Methoden ergeben. Zudem erklären sie den Innovationsverlauf und deuten auf Implikationen für das Innovationsmanagement. Bezüglich DevOps geben sie eine ausführliche Beschreibung der Hintergründe und Übersicht der Methoden.

Bornemann und Brandes [3] beschreiben in einem umfangreichen Werk den rechtlichen Rahmen der Bank-IT. Der rechtliche Rahmen leitet sich aus dem Kreditwesengesetz (KWG) ab und wird von der Aufsicht mit Verwaltungsrichtlinien und -Vorgaben den Banken auferlegt. Die nationale Aufsicht Bundesanstalt für Finanzdienstleistungsaufsicht (BaFin) veröffentlich [13], [12], [14]. Zusätzlich beaufsichtigt die European Banking Authority (EBA) bedeutende Institute direkt und verfasst die Leitlinie [2]. Die Regulatorik verweist auf gängige Sicherheitsstandards wie zum Beispiel [22] oder Standards der International Organization for Standardization (ISO).

Disterer [7] gibt einen Überblick zu den Standards ISO 27000, 27001 und 27002 und hebt ihre Bedeutung besonders hervor.

Strietzel, Steger und Bremen [32] knüpfen an die veränderten regulatorischen Rahmen der Bank-IT an und geben einen strategischen Überblick zu den Herausforderungen im Rahmen der Digitalen Transformation im Finanzwesen. Zudem gehen sie die Implikationen bezüglich der etablierten Legacy-IT ein.

Dorschel [9] bietet eine ausführliche Referenz zur Organisation und Prozesse der Bank-IT. Dabei werden als Folge von Änderungen in den regulatorischen Rahmenbedingungen der letzten Jahre die entstandene Schatten-IT beschrieben. Er beschreibt weiterhin die wesentlichen organisatorischen Paradigmen, die die Branche geprägt haben und verweist auf ein neues Paradigma, dass jedoch als Folge der Digitalisierung widersprüchliche Anforderungen gegenüber der Organisation hervorruft. Auf der einen Seite steht der Anspruch auf Beschleunigung und Flexibilisierung und auf der anderen der Störungsfreie Betrieb und Regulatorik. Die Zwänge der Legacy-IT und der Regulatorik erfordern spezifische Modifikationen der bekannten Konzepte.

Koch, Ahlemann und Urbach [25] hinterfragen das aktuelle industrialisierte Paradigma des Managements. Es steht die Frage ob die Paradigmen "Plan-Build-Run" und "Make-Source-Deliver" überhaupt den heutigen Herausforderungen gewachsen ist. Es wird bezweifelt, dass Organisationen mit einer industriellen Ausrichtung an der Digitalisierung von Geschäfts- und Wertschöpfungsmodellen treiben mitwirken können. Eine Reihe an Innovationsfragen stehen offen, die mit dem aktuellen Paradigma nicht beantwortet werden können. Für ein neues Paradigma wird die Berücksichtigung von *Design-Thinking* Ansätzen gefordert und die *Gestaltungsfähigkeit* zur Findung der richtigen Lösung für IT-Organisationen besonders hervorgehoben. Hierfür führen sie ein neues Paradigma "Innovate-Design-Transform" ein und weisen ihnen jeweils Methoden und Eigenschaften zu. Die Umwandlung zu einem Innovate-Design-Transform (IDT) Paradigma erfordert eine umfangreiche Anpassung der Organisation, die aus wissenschaftlicher Sicht als das Aufbauen von organisationalen Fähigkeiten, genannt "capabilities" verstanden wird.

2.2 STAND DER TECHNIK

Software hat eine Schlüsselrolle für digitale Innovation in allen Bereichen [1], die wichtiger denn je geworden ist. Die IT ist ständigen Veränderungen ausgesetzt. Technologietrends erzeugen immer wieder neue Impulse und Betriebe mit properietären Infrastrukturen müssen sich neu ausrichten [5].

Gupta [20] beschreiben Goldman Sachs' Weg der digitalen Transformation hin zu einem Technologieunternehmen und Plattform. Die Studie von Gupta und Simonds stellt viele Ansätze vor, mit denen Goldman Sachs erfolgreich neue Geschäftsmodelle entwickelt hat und alte Modelle Strukturen abgeschafft hat. Diese Ansätze könnten als Positivbeispiel für das deutsche Finanzwesen dienen.

So entstehen gegenüber etablierten Unternehmen, Technologieunternehmen mit effizienten Geschäftsmodellen. Sie integrieren mit unterstützenden Anwendungen IT und Geschäftsmodell [5]. Im Finanzwesen sind es die sogenannten FinTechs (Kap. 3.2)

Zuletzt hat sich die IT vor allem durch einige Schlüsseltechnologien verändert. Für die IT-Architektur hat Cloud-Computing zusammen mit Cloud-Plattformen einen entscheidenden Impuls gegeben. Sie haben IT-Ressourcen abstrahiert. Anwendungen können in kürzester Zeit und in großem Umfang skaliert werden. Software wird dadurch immer mehr als Dienstleistung bereitgestellt und findet sich immer weniger in den Festplatten von lokalen Systemen. Kurze Entwicklungszyklen durch neue und agile Methoden sorgen für eine kontinuierliche Weiterentwicklung der Anwendungen und daher eine kontinuierlichen Implementierungsbedarf in die Produktion. Software wird dadurch in immer kürzeren Abständen und ausgeliefert. Hierzu entstand der Bedarf einer erhöhten Zusammenarbeit zwischen Betrieb und Entwicklung, woraus DevOps entstand. Kontinuitätskonzepte werden daher immer relevanter, sodass Methoden wie CI und Continuous Delive-

ry (CD) zum Standard geworden sind. Gleichzeitig herrscht ein kontinuierlicher Anstieg der Anforderungen an die IT-Architektur [5], die in unflexiblen Architekturen in einem Engpass münden [4, 5]. Diese Engpässe könnten aufgrund neuer Technologien und Architekturen, wie zum Beispiel Kubernetes, Docker und Microservices gelöst werden. Grundlage hierfür bieten unter anderem gemeinsame Plattformen, wie zum Beispiel eine Cloud. Voraussetzung ist jedoch gerade in großen Betrieben eine strategische Entscheidung zu treffen, da die Lücke zwischen Innovation und Adaption in der Legacy-IT immer größer wird. Diese Entscheidung beeinflusst das ganze Unternehmen als solches.

2.2.1 *DevOps, Kontinuitätskonzepte und weitere Methoden*

DevOps ist eine Bezeichnung, die durch die Zusammensetzung der Wörter "Development" und "Operations" erzeugt wurde und entstand aus Problemen bei der Inbetriebnahme von Software [7]. Aus einer schnellen, flexiblen und nutzerzentrierten Umsetzung von funktionalen Anforderungen resultieren Benutzerakzeptanz und Kundenzufriedenheit [1, 7]. Daher ist es ein Ziel von DevOps die Zusammenarbeit zwischen Entwicklung und Betrieb zu verbessern. Die Automatisierung von Routineaufgaben und Wandel in der Zusammenarbeitskultur hat darin eine zentrale Bedeutung [1].

In der Praxis werden Entwicklung und Betrieb durch die Delivery Pipeline verbunden. Hierbei werden vor allem zwei Verfahren vereint und automatisiert, Continuous Integration und Continuous Delivery. Zusammen bilden sie einen kontinuierlichen Lebenszyklus der Entwicklung von der Idee bis hin zur Inbetriebnahme.

Für die DevOps Prinzipien gibt es keine festgelegten Beschreibungen, sodass Unternehmen sie nach ihrem eigenen Bedarf anpassen und individualisieren [1]. DevOps ist ein Zusammenschluss aus mehreren Prinzipien für eine bessere Zusammenarbeit.

Es gibt jedoch Prinzipien, die vor allem eine Kulturänderung in Organisationen fordern. Die Prinzipien von DevOps stehen für *Culture*, *Automation*, *Measurement* und *Sharing* [21].

Alt, Auth und Kögler [1, S.26f] eruieren diese Punkte weiter:

- *Kulturwandel*, zu einer gemeinsamen Verantwortung aller Beteiligten für die Auslieferung von Qualitätssoftware
- *Automatisierung* der Prozesse von Entwicklung, Test und Bereitstellung bis hin zur Produktivnahme als Schlüssel für kürzere Durchlaufzeiten, Fehlervermeidung und schnelleres Feedback
- *Kennzahlen*, die miteinander verknüpft als Beitrag zur geschäftlichen Wertschöpfung und Produktivitätssteigerung durch faktenbasierte Einstufung der aktuellen Leistungen und Definition von überprüfbaren Verbesserungszielen

- *Teilen*, zum Beispiel von Wissen, Tools, Infrastruktur und die Würdigung von Erfolgen als Prinzip um die Zusammenarbeit zwischen den Beteiligten zu prägen

CONTINUOUS INTEGRATION CI ist in der Softwareentwicklung ein Verfahren für die Erstellung von Software. Sie steht für eine kontinuierliche Integration von Softwarekomponenten in die gesamte Anwendung. Als Ergebnis liefert sie getestete *Builds*, welche eine spezifische Ausgabe der entwickelten Anwendung sind. Kontinuität wird hierbei durch Automatisierung erreicht. Hierfür werden Zwischenschritte, so genannte *Stages* in einer *Pipeline* abgearbeitet. Die Pipelines definieren den jeweiligen Ablauf für die Erstellung eines Builds.

In der Praxis ist sie ein Ansatz, um den Erstellungsprozess von Software zu automatisieren. Builds werden in größeren Organisationen nicht auf den lokalen Rechnern von Entwicklern erstellt, sondern in einer gemeinsamen Plattform. Die Plattform steuert zudem den in der Pipeline enthaltenen Ablauf, wie Testen und Auslieferung. Eine weit verbreitete Plattform für CI ist Jenkins. Für Jenkins gibt es viele Erweiterungen, wodurch die Anwendung sehr Anpassungsfähig ist. CI Plattformen werden in der Regel mit weiteren Anwendungen der Softwareentwicklung, wie zum Beispiel das Source Code Management (SCM) kombiniert.

CONTINUOUS DELIVERY CD ist ein Verfahren für die Auslieferung von Builds an die produktive Umgebung, die sich im Rahmen von DevOps von Entwicklung hin zu Test- und Produktionsumgebung erstreckt Alt, Auth und Kögler [1]. Die Produktionsumgebung ist hierbei der tatsächliche geschäftliche Betrieb von IT. Durch die Erweiterung von CI mit CD entsteht die eigentliche Pipeline, die auch "Delivery Pipeline" genannt wird. Das Ziel von Continuous Delivery ist die Auslieferung und Abnahme von Software und enthält Aufgaben für Qualitäts-, Risiko- und Change-Prozesse [1].

Neben CD existiert der hiervon schwierig unterscheidbare Begriff *Continuous Deployment*, welcher sich im Anschluss von CI, auf die automatisierte Inbetriebnahme in die Produktionsumgebung konzentriert [1]. Continuous Deployment kann als das Konzept für die kontinuierliche Inbetriebnahme von Builds eingegrenzt werden und ist durchaus Bestandteil der technischen Umsetzung von CD.

DEVOPS LEBENSZYKLUS Abb. 2.1 stellt das DevOps Lebenszyklus mit ihren jeweiligen kontinuierlichen Verfahren dar, die das Produkt und die Beteiligten umfassen. Agile Entwicklung, CI und CD sind jeweils Verfahren, die vor DevOps existierten [1]. Sie werden in diesem Zusammenhang jedoch als ein kontinuierliches Modell gesehen, die eine Schnittstelle zwischen den Beteiligten herstellt und das Produkt dadurch am Leben erhält und wachsen lässt. Im Gegensatz dazu stehen sequenzielle Modelle wie *Plan-Build-Run [25]*, in der ein Prozess am Ende als abgeschlossen gilt. Das Wesentliche an

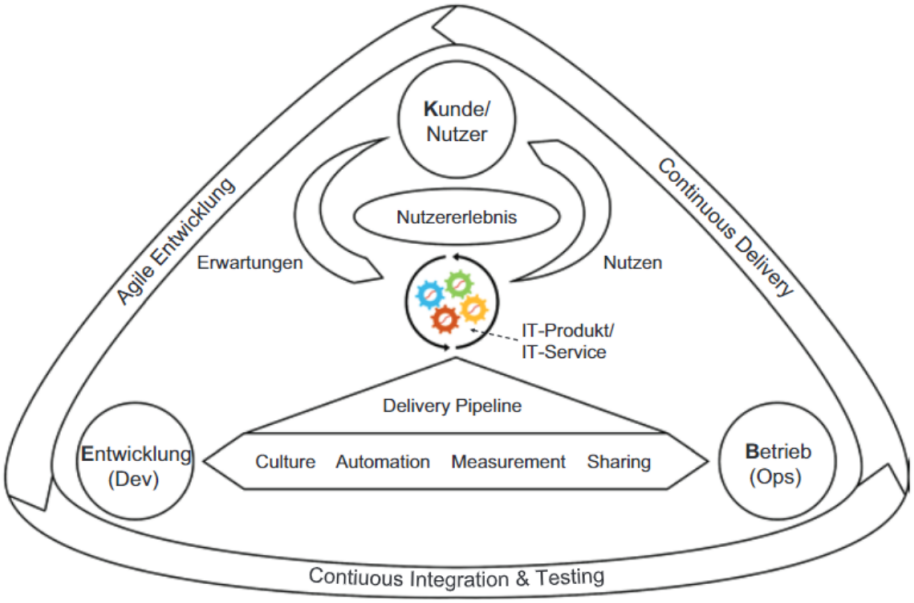


Abbildung 2.1: Alt, Auth und Kögler [1], S. 28, DevOps im Überblick

diesem Modell (Abb. 2.1) ist ein Kontinuitätsprinzip [1] der Prozesse auf allen Ebenen.

ITIL Im Rahmen des Innovationsdrucks vom IT-Management haben nach Alt, Auth und Kögler [1] Organisationen ihre Leistungen nach dem ITIL-Service-Lebenszyklus strukturiert und den Ablauf an den ITIL-Prozessen ausgerichtet. ITIL bedingt die Kombination von Zweckmäßigkeit "Utility" und Einsatzfähigkeit "Warranty" für einen Wertbeitrag eines Services auf das Geschäft. Zweckmäßigkeit wird durch die Realisierung einer geforderten Funktion oder durch die Beseitigung einer Einschränkung erreicht. Als Qualitätsmerkmal gilt die Einsatzfähigkeit. Sie bezieht sich auf eine ausreichende Verfügbarkeit, Kapazität, Kontinuität und Sicherheit. Es ist Notwendig das bisherige Verständnis von IT-Services um das Kriterium "Innovationsbeitrag" zu ergänzen. Services müssen sich in einer Innovationskultur bezüglich ihres Beitrags zur Innovationsgenerierung messen lassen.

DESIGN-THINKING PROZESS *Design-Thinking* gehört zu den kreativen und innovativen Prozessen für die Gestaltung von Lösungen. Ihr wesentlicher Merkmal ist die agile und zeitnahe Bereitstellung von Prototypen [1]. Die Schritte aus Design-Thinking sind keine fest definierten Rahmen. Sie sind ein Werkzeug, das individualisiert werden kann.

In [34] wurden die Schritte des Prozesses erklärt:

- *Analyse* besteht aus *Empathize* und *Define* Hierin gilt es die Bedürfnisse und Probleme des Kunden zu verstehen und sich dabei auf die Ursachen zu fokussieren. Der Teilnehmer soll hierbei sich in den Kunden

hinein versetzen und anfangen Probleme des Kunden mit einem selbst lösen.

- *Synthese* besteht aus *Ideate*, *Prototype* und *Testing*. Hierin werden Ideen gesammelt. Dabei spielt in der anfänglichen Iterationen zuerst die Quantität eine wesentliche Rolle.

SOFTWARE DEFINED BUSINESS Alt, Auth und Kögler [1, S. 51] definieren Software-defined Business (SdB) als das Erkennen von Software als zentrales Gestaltungselement. Software ergänzt und substituiert hierbei teilweilse die Hardware. Daher sind Kompetenzen zur Softwareentwicklung und ihr Management eine Grundlage für Innovationsfähigkeit.

Softwareentwicklungskompetenzen sind daher sowohl in frühen Phasen des Innovationsmanagements als auch bei der kontinuierlichen Weiterentwicklung wichtig [1].

INFRASTRUCTURE AS CODE Eine wichtige Voraussetzung für DevOps ist die Automatisierung von Prozessen, die vor allem im Bereich von materiellen Ressourcen, wie Hardware bisher schwierig war. Das Prinzip der *Dematerialisierung* ist auch ein wichtiger Bestandteil der digitalen Transformation. Infrastructure as Code (IaC) ist die Abstraktion von IT-Infrastrukturen und -Ressourcen auf eine Ebene analog zu einer Programmiersprache. Diese Analogie impliziert schon eine Automatisierung in der Ausführung der jeweiligen Prozesse. Über Schnittstellen kann die Infrastruktur angefordert und konfiguriert werden. Die Bereitstellung von IT-Ressourcen kann aufgrund der steigenden Anforderungen an die IT-Architektur [1, 4, 5] nicht mehr manuell erfolgen und muss automatisiert werden. Daher ist dieser Ansatz für die Umsetzung von DevOps erforderlich.

Nach Alt et al. [1] sorgt eine programmierbare Infrastruktur für Flexibilität, senkt die Fehlerrate und beschleunigt die Installation und Konfiguration von Systemen.

INFRASTRUCTURE AS A SERVICE IaC bietet eine Möglichkeit die Infrastruktur programmierbar zu gestalten und ihre Prozesse dementsprechend zu automatisieren. An ihre Schnittstellen anknüpfend entsteht die Möglichkeit eine Plattform für eine serviceorientierte IT-Architektur aufzubauen. Hieraus resultiert IaaS als Delivery-Modell [1] für die IT-Ressourcen. Während sich IaC auf das Deployment von Infrastrukturen konzentriert, ist IaaS die Delivery-Pipeline im Bereich der IT-Infrastruktur. Ihre Aufgabe ist analog zu CD das Nutzererlebnis der Entwickler, die Entwicklung zu verbessern. Das Produkt ist in diesem Fall die Infrastruktur, die als ein Service bereitgestellt wird.

CLOUD-COMPUTING Ansätzen wie IaC und IaaS entstanden aus dem Bereich Cloud-Computing [1]. Sie stellen mittlerweile eine Plattform für IT-Infrastruktur und ganze IT-Architekturen zur Verfügung und bieten eine

Möglichkeit die Kosten des IT-Betriebs durch Auslagerung erheblich zu senken. Beispiele für Cloud-Plattformen sind Google Cloud Plattform (GCP), Amazon Web Services (AWS) und Microsoft Azure.

Cloud-Plattformen sich darauf IT-Ressourcen dem Kunden nach Bedarf extern zur Verfügung zu stellen. Wenn der Bedarf sinkt, werden die Ressourcen an eine andere Stelle allokiert. Bezahlt wird hierbei die beanspruchte Zeit der Ressourcen.

2.3 AUSGANGSSITUATION: CI IN BANKEN

In diesem Abschnitt wird die Ausgangssituation beschrieben, aus der die Problemstellung definiert wurde. Sie gibt einen kleinen Einblick wie die Rahmenbedingungen für Entwickler in der IT einer Bank mit modernen Methoden wie DevOps aussehen könnten und wo die Grenzen für Veränderungen liegen. Von diesem Standpunkt aus wird ein erster Lösungsweg vorgeschlagen gängigen Standards aus dem Stand der Technik.

SOFTWAREENTWICKLUNGSUMGEBUNG Die SEU bezeichnet in dieser Arbeit ein Netzwerk für die eigene Softwareentwicklung in einer Bank. Unter anderem müssen in Banken Test-, Integrations- und Produktionsumgebungen getrennt werden [13].

Die SEU ist eine Umgebung innerhalb welcher die Integration von Quellcode aus dem SCM durchgeführt und anschließend mithilfe einer Delivery Pipeline an die CD Gruppe ausgeliefert wird. CD hat anschließend die Aufgabe die Anwendung abzunehmen und in die Produktionsumgebung bereitzustellen.

Die Integrationsumgebung ist einer CI Gruppe zugehörig. Ihre Aufgabe besteht in der Weiterentwicklung der Systeme und Anwendungen für die SEU. Sie kann als eine Plattform für die unterschiedlichen Projektgruppen für Anwendungsentwicklung gesehen werden. Die Zusammenarbeit der SEU ist nach *DevOps* Methoden ausgerichtet (Abb. 2.1), mit der Besonderheit, dass die Kunden von CI die Entwickler aus den Projektgruppen sind. Die CI Gruppe ist aus Sicht der Gesamtorganisation in Wahrheit ein kleiner Betrieb für die Entwicklung. Sie ist auf den tatsächlichen IT-Betrieb der Organisation bezüglich der Infrastruktur angewiesen. Wie bereits erwähnt sind die Umgebungen streng voneinander getrennt [13], woraus die hohen Laufzeiten für die Bereitstellung von angeforderten Virtuelle Maschinen (VMs) stammen.

Aus diesem Grund könnte die Verfügbarkeit der SEU Systeme gefährdet werden. Der IT-Betrieb ist hierbei Service- aber auch Prozessorientiert. Daher könnte die Skalierbarkeit durch externe Infrastrukturen mit effizienteren Prozessen möglicherweise erreicht werden. Die Anbindung der SEU an eine externe Cloud stellt eine wesentliche Veränderung und Auslagerung nach [13, 14] dar. Eine Entscheidung für oder gegen eine externe Cloud ist daher eine strategische Entscheidung mit tiefgreifenden Auswirkungen, die analysiert werden müssen [3, 13, 14].

Die SEU könnte jedoch innerhalb des eigenen Netzwerks über Container-

Technologie flexibel und skalierbar implementiert werden. Die gängigen Lösungen [23, 29] unterliegen jedoch aufgrund diverser Besonderheiten der Legacy-IT in einigen Finanzinstituten [4, 5] aufwändigen Anpassung, die gegebenenfalls die Effizienzsteigerung der neuen Technologie negieren könnte. Im kleinen Umfang könnten jedoch einzelne Komponenten modularisiert und mit Containern leichtgewichtig und skalierbar umgesetzt werden [5], (Par. 2 2.3.

ANWENDUNGEN FÜR DIE BUILD-PIPELINE

- **Gitlab**, als SCM
- **Jenkins**, als CI Plattform für die Build-/ Delivery-Pipeline
- **Artifactory**, als Repository für verwendete Artefakte

GRUNDLEGENDES RE-DESIGN

Bussmann u. a. [5] nennt zwei Wege für den Ersatz und grundlegenden Re-Design von Kernsystemen. Diese Wege werden verkürzt als zwei Prozesse dargestellt:

- Anwendungen schrittweise entkernen, erneuern und modularisieren und funktionale Anforderungen konsolidieren
- Kernsysteme durch völlig neue, selbst entwickelte Applikationen austauschen oder einsetzen von Standardsoftware

NACHVOLLZIEHBARKEIT DER SEU

Die SEU sollte ist aus mehreren Gründen möglichst nachvollziehbar zu gestaltet. Eine flexible Umgebung setzt auch Wartbarkeit und Anpassungsfähigkeit voraus. Dies ist durch die Verwendung von gängigen Standards möglich. Systeme sollten möglichst unabhängig von einzelnen Verantwortlichen verständlich und offen sein. Eine Individualisierung von Systemen sollte möglichst vermieden werden. Sie sollten auf einen gemeinsamen Nenner gebracht werden.

Dadurch kann eine schnelle Anpassung der Systeme an neue Technologien und Methoden erfolgen. Gleichzeitig können Systeme für neue Geschäftsmodelle in kurzer Zeit weiterentwickelt werden. Während einer Störung der Systeme oder in Krisensituationen kann auch schnell reagiert werden.

Dies hat auch einen positiven Effekt auf interne Kontrollverfahren. Die Umgebung ist nicht nur verständlicher für den Prüfer sondern auch sicherer durch die Verwendung von bereits geprüften Standards.

Nachvollziehbarkeit wird in Banken auch für die Rückverfolgung von Softwareartefakten vorausgesetzt. In der Praxis müssen Ergebnisse aus der Entwicklung rückverfolgbar sein bis auf den Quellcode, der Anforderung und dem Prozess aus der sie entstanden sind.

REPOSITORY FÜR ARTEFAKTE

Eine Standardsoftware hierfür die Anwendung Artifactory. Diese speichert in einem Snapshot die Ergebnisartefakte aus dem Build-Job und die zum erstellen verwendeten Artefakte. Das Artifactory wird ebenfalls verwendet, um Tools und Libraries für den Build zu

beziehen. Dazu wird Artifactory entweder als Proxy für den Zugriff auf ausgesuchte öffentliche Repositories verwendet oder die Ressourcen werden in der Artifactory Instanz hinterlegt.

DEVOPS MIT MIRCROSERVICES Daraus folgt, dass einzelne umfangreiche Systeme für die SEU keine Flexibilität bieten. Für automatisierte Prozesse und skalierbare, flexible Umgebungen können die Anwendungen aus der SEU in noch kleinere Module aufgeteilt werden. Insbesondere ist es wichtig, dass einzelne Komponenten der Integrationsplattform trennbar sind und unabhängig voneinander laufen. Komponenten sollten "self-contained" aufgebaut werden. In folge dessen sind sie flexibler können auch unabhängig voneinander skaliert werden. Das würde Engpässe erheblich reduzieren. Die Eliminierung von Abhängigkeiten sorgt für eine bessere Nachvollziehbarkeit, da hierbei gängige Standards angewendet werden können.

SKALIERUNG VON JENKINS MIT KUBERNETES Pathania [29] zeigt wie er Jenkins mit Kubernetes, Docker und Public-Cloud skaliert. Dabei konfiguriert er Jenkins für die Nutzung innerhalb einer Public-Cloud und Kubernetes. Er veranschaulicht einen Ansatz, um den Jenkins Master über Replikas zu skalieren. Zudem beschreibt er die Skalierung von Jenkins Agenten für die Verteilung der Builds. Für die Agenten werden Images erzeugt, woraus Jenkins mithilfe eines Docker Plugins einen Container für den Build generiert. Für die Agenten erstellt er ein Image aus Ubuntu und installiert die Build-Tools, wie in jedem anderen Ubuntu System. Dadurch erzeugt er die Images über einen Container, welches vorher konfiguriert wurde. Insbesondere beschreibt Pathania ausführlich die einzelnen Schritte, die für die Orchestrierung der ganzen Anwendung mit Kubernetes erforderlich sind.

Üblicherweise werden Anwendungen in Kubernetes mit Replikas skaliert. Das heißt in der Praxis, dass eine neue Instanz der Anwendung erzeugt wird und ein Load-Balancer die Anfragen unter den verschiedenen Instanzen verteilt. Das resultiert in der vorwiegend horizontalen Skalierung von Anwendungen.

Jenkins besteht bei einer Konfiguration mit verteilten Builds aus mehreren Komponenten. Ein Jenkins Master, welche die Anfragen bearbeitet und Build-Jobs triggert und ausführende Agenten für die Verteilung von Builds. Die Komponenten sollten getrennt voneinander skaliert werden, da sie vorwiegend für die ausführenden Komponenten erforderlich ist.

Hierfür gibt es in Kubernetes mehrere Abstraktionsebenen. Die größte Einheit, die die ganze Architektur zusammenfasst ist der Cluster. Dieser besteht aus mehreren Knoten, die jeweils mehrere Pods beinhalten. Die Knoten sind virtuelle Maschinen und enthalten die Hardware. Innerhalb dieser Knoten werden die Container ausgeführt. Diese sind noch einmal unter Pods zusammengefasst, die eine Arbeitseinheit oder Service darstellen.

Ein sinnvoller Ansatz ist die ganze Anwendung innerhalb eines Knotens aufzubauen und sie durch Replikas der Knoten zu skalieren. Dabei sollte die Anwendung modularisiert werden und in unabhängig voneinander skalier-

bare Microservices aufgeteilt werden. Anwendungen können komplex sein und ein Engpass entsteht in wenigen Bereichen, sodass nicht die ganze Anwendung repliziert werden muss. In der Praxis werden die Funktionen der Anwendung in mehreren Containern aufgeteilt, die miteinander kommunizieren. Die Replikation von Containern bewirkt, dass Anwendungen nur in den Komponenten skaliert werden, in der es erforderlich ist. Diese Praxis schafft Redundanzen ab und und sorgt für einen Wirkungsgrad in der Nutzung von IT-Ressourcen.

ANLEGEN EINES KUBERNETES CLUSTERS FÜR JENKINS MIT GKE Google beschreibt in [23] einen Ansatz für eine skalierbare Architektur von Jenkins mit der Google Kubernetes Engine (GKE).

In Abb. 2.2 werden die Komponenten in mehreren Knoten, genannt Nodes zusammengefasst. In Node 1 befindet sich der Jenkins Master. Innerhalb dieses Nodes kann der Jenkins Master repliziert werden. Im gleichen Node oder in weiteren Nodes können die Jenkins Agenten, genannt Jenkins Executor, erzeugt werden.

Der Load Balancer aus der GKE arbeitet vollautomatisch und erfordert nur wenige Parameter zur Konfiguration. Das reduziert den Aufwand in der Entwicklung dieser Architektur erheblich.

So lange in einem Node genug Ressourcen frei sind können weitere Container in so genannten Pods erzeugt werden. Zudem können beliebig viele weitere Nodes aktiviert oder erzeugt werden. Ein Node ist in GKE eine virtuelle Maschine aus der Computing Engine.

Daraus resultiert eine hohe Flexibilität für die Skalierung der Anwendungen. Diese können sowohl vertikal anhand der Ressourcen von den Nodes, als auch Horizontal über die Aktivierung von weiteren Nodes oder Pods skaliert werden. Vor allem bestehen für die horizontale Skalierung mehrere Abstraktionsebenen.

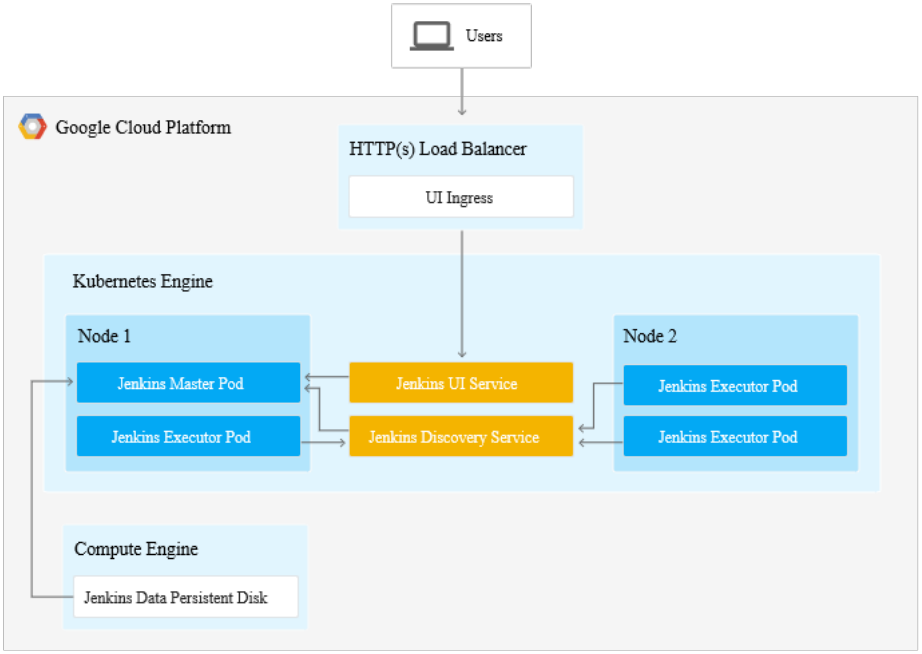


Abbildung 2.2: Deployment von Jenkins in einem Kubernetes Cluster mit mehreren Knoten [23]

3 FINANZINSTITUTE, INNOVATION UND REGULATORIK

Das nachfolgende Kapitel untersucht mithilfe einer Fallstudie die Auswirkungen von innovativen Technologien und Veränderungen auf das Finanzwesen. Sie enthält die wesentlichen Schritte aus *Design-Thinking* für eine innovative Lösungsfindung, die mit dem Schritt *emphatize* beginnt. Zudem betrachtet sie die Regulatorik und analysiert die Rahmenbedingungen.

Dazu beschäftigt sie sich mit den Rahmenbedingungen für Veränderungen in Kreditinstituten aus verschiedenen Perspektiven und betrachtet dabei die möglichen Einschränkungen und antreibenden Faktoren für die an verschiedenen Stellen bekannten Probleme in der IT-Architektur von Banken. Hierin werden auch die Auswirkungen der Regulatorik des Finanzwesens bezüglich der Maßnahmen für die IT untersucht.

3.1 TRANSFORMATION VON GOLDMAN SACHS

Goldman Sachs sieht sich nicht als Finanzdienstleister, sondern als ein Technologieunternehmen und als eine Plattform [20]. Diese Transformation könnte als Vorbild für Institute des deutschen Finanzwesens dienen und eventuell wiederverwendbare Lösungen bieten.

Auf der einen Seite stehen die FinTechs, die sich Marktanteile im Privatkundenbereich sichern und zunehmenden Druck auf die etablierten Institute ausüben (3.2). Auf der anderen Seite richten sich anpassungsfähige Institute neu aus und etablieren sich mit Technologien und Plattformen in neue Geschäftsmodelle.

Eines dieser Institute ist Goldman Sachs. Gupta und Simonds [20] haben die Transformation von Goldman Sachs hin zum selbsternannten Technologieunternehmen [20] beschrieben. Nachfolgend werden die relevanten Punkte aus dieser Studie untersucht.

AUSWIRKUNG VON TECHNOLOGIEN AUF GOLDMAN SACHS

zu betrachten wie Goldman Sachs neue Technologien wahrnimmt und

Folgende Auswirkungen ergaben sich durch neue Technologien für Goldman Sachs und waren ein Impuls für Veränderung [20]:

- Daten und Metriken wurden wertvoller als die Instinkte der Händler.
- Die Auswirkung von Technologie im Finanzwesen verstärkte sich seit der Finanzkrise 2008 umso stärker
- Cloud, Open-Source Software und Schnittstellen führten zu einer erheblichen Reduzierung von Zeit und Kosten in der eigenen Entwicklung von Technologien

- Verstärkter Wettbewerb durch FinTechs

Ein wichtiger Punkt für Goldman Sachs ist hierbei die Reduktion der Kosten für die eigene Entwicklung von Technologien. Verwendung von Open-Source oder Standardsoftware reduziert die Komplexität der IT-Architektur [5]. Sie bringt die SEU auf einen gemeinsamen Nenner und reduziert die Kosten.

Mit der Verwendung von Open-Source Software für die Entwicklung von Anwendungen können gängige Lösungen einem individuellen Bedarf angepasst und verwendet werden. Hierdurch könnte sehr viel Aufwand in der Entwicklung eines eigenen Konzepts für einen Anwendungsfall erspart bleiben.

Gängige Lösungen sind einfacher zu verstehen und würden auch die Einstiegshürden für Fachkräfte reduzieren, wohingegen properitäre Anwendungen spezialisiertes Personal benötigen. Open-Source und Standardsoftware könnten somit als ein antreibender Faktor für die Softwareentwicklung gesehen werden.

Parallel dazu sind Cloud-Plattformen ein weiterer Antreiber für die Softwareentwicklung. Sie bieten eine automatisierte Plattform für IT-Ressourcen, sodass eine Infrastruktur in kürzester Zeit umgesetzt werden kann. Es gibt eine ganze Reihe an Konzepten, die durch Cloud-Plattformen ermöglicht werden. Diese beinhalten IaaS, IaC, SaaS und Platform as a Service (PaaS). Cloud-Plattformen bieten einen flexiblen und skalierbaren Standard für die seit langem angeforderte SOA [4] in Banken und könnten auch ein Anstoß für die digitale Transformation ganzer Unternehmen sein. Für Goldman Sachs ist sie definitiv einer der Antreiber für die eigene Entwicklung von offenen und flexiblen Plattformen [20].

IT-STRATEGIE VON GOLDMAN SACHS Die Integration von neuen Technologien war für Goldman Sachs Opportunität und unvermeidbar [20] zugleich:

- Kostensenkung durch eine Effizienzsteigerung des Betriebs
- Entwicklung von internen und externen Plattformen
- Unterstützung des Kerngeschäfts
- Wertschöpfung von neuen Geschäftsmodellen

Gerade der verstärkte Wettbewerb durch FinTechs und der finanzielle Druck nach der Finanzkrise bedingten, dass sich das Unternehmen verändert. Die Unvermeidbarkeit dieser Entscheidung zeigt auch, dass Opportunitäten nicht ausreichen, um Veränderungen umzusetzen. Goldman Sachs musste sich anpassen, um in einer digitalen Welt geschäftsfähig zu bleiben. Sie musste neue Technologien integrieren, weil das alte Modell nicht mehr tragbar war.

Der Ansatz neue Technologien anzunehmen hat weitreichende Auswirkungen für die Zukunft von Goldman Sachs [20]. Eine gewisse Risikoakzeptanz von oft unvorhersehbaren Auswirkungen sollte vorhanden sein, um sich für neue Technologien zu überwinden. Handlungsunfähigkeit könnte mit der Zeit ein größeres Risiko erzeugen.

EFFIZIENZSTEIGERUNG Wie Gupta [20] erkennt, hat Goldman Sachs die Automatisierung von Geschäftsprozessen durch die Zentralisierung von Kernelementen vorangetrieben, wodurch Redundanzen entfernt wurden.

Ezra Nahum betont in [20, Zitat, S.5], dass die unterschiedlichen Geschäfte von Goldman Sachs bisher in "Silos" als unabhängige und eigenverantwortliche Einheiten ausgeführt wurden. Es stellte sich die Frage, was ihre gemeinsamen Nenner sind. Ein Ansatz war in einer gemeinsamen technologischen Plattform zu arbeiten und in kleineren Teams sich darauf zu konzentrieren, was das jeweilige Geschäft ausmacht. Zu ihrer Unterstützung unterliegt ihnen ein größeres gemeinsamen Team.

Übertragen auf die SEU von Banken können diese *Silos* [20] auch als Metapher für die monolithischen Systeme [5] aus dem Finanzwesen gesehen werden. Eine SOA alleine reicht daher nicht aus, um mit kontinuierlichen Veränderungen der IT umzugehen. Unterstützende Funktionen könnten unter gemeinsamen Plattformen zusammengefasst werden. Ein Beispiel wäre eine gemeinsame Datenbank. Goldman Sachs konnte mit einem gemeinsamen "Data Lake" und Mashine-Learning die Marktdaten effektiver bewerten [20].

Die Anwendungen selbst müssten in kleinere funktionale Komponenten aufgeteilt werden [5]. Microservice Architekturen bieten hierfür einen möglichen Ansatz. Die wesentlichen Funktionen für die Kernaufgabe einer Anwendung machen nur einen kleinen Teil der ganzen Anwendung aus.

Gupta [20] beschreibt, dass die Umsetzung dieser Strategie sich als schwierig gestaltet. Sie erfordert die Zusammenarbeit aller Beteiligten. Die verschiedenen Abteilungen müssen für die Umsetzung dieser Strategie kurzfristige Kompromisse eingehen und als finanziell eigenverantwortliche Einheiten die Kosten vorerst tragen.

Cohen betont in [20, Zitat, S.5] hierzu das wesentliche Problem:

> " Die Vision ist eine Sache, aber der wirkliche Fortschritt besteht darin, die Organisation dazu zu bringen, die Barrieren abzubauen, um in einer digitalen Welt zu agieren und die traditionell tiefen vertikalen Barrieren zu überwinden"

Innerhalb der IT sind diese Probleme bereits bekannt und werden von Disterer [8] beschrieben. Ansätze aus DevOps [1] können hierbei erfolgreich die Zusammenarbeit innerhalb der IT verstärken. Im Finanzwesen wird für unvereinbare Verantwortungsbereiche eine strenge Funktionstrennng gefordert [13]. Dies wirkt sich auch auf eine strikte Trennung zwischen Test- und Produktivumgebung aus Finanzdienstleistungsaufsicht [13] und könnte die

Zusammenarbeit erschweren. Die Umsetzung der Funktionstrennung auch auf personeller Ebene für die IT begünstigt möglicherweise die Vertiefung von Barrieren und erschwert die Umsetzung von DevOps. Möglicherweise ist dies ein einschränkender Faktor für die Softwareentwicklung.

GEMEINSAME FUNKTIONEN ZENTRALISIEREN? Verallgemeinert könnte dieser Ansatz der Effizienzsteigerung (Kap. 3.1) auch für die Effizienzsteigerung in anderen Bereichen genutzt werden.

Zuerst werden Gemeinsamkeiten identifiziert. Als nächstes werden effektive Plattformen als ein Antrieb für Effizienz errichtet. Die Einheiten oder Komponenten können sich so auf ihre eigentliche Kernaufgabe konzentrieren.

Es könnte hierbei nach der konstruktiven Frage "Welche Gemeinsamkeiten haben wir?" statt nach der Frage "Was unterscheidet uns?" aufgebaut werden. Ersteres impliziert schon die Identifizierung von Gemeinsamkeiten, wodurch Redundanzen nicht unentdeckt bleiben und Synergie erzeugt wird. Die zweite Frage setzt den Fokus auf die Abgrenzung der Einheiten voneinander, wodurch die kulturellen "Silos" immer höher werden. Unterschiedliche Bedürfnisse müssen nicht zu unterschiedlichen Technologien führen.

TECHNOLOGIEPLATTFORMEN Goldman Sachs sah eine Opportunität darin, die internen Plattformen auch für ihre Kunden zugänglich zu machen. Darunter zählt die Plattform Marquee, die für interne Zwecke der Marktanalyse entwickelt wurde [20]. Die Anwendungen darin wurden selfcontained entwickelt, sodass die Integration in interne und externe Systeme mit geringem Aufwand realisierbar war [20]. Dadurch konnten die internen Plattformen schnell angepasst werden für neue Geschäftsmodelle. Die in der Plattform enthaltenen Anwendungen für die Marktanalyse konnten dadurch Geschäftskunden zugänglich gemacht werden, um den Kundendialog zu verbessern und Goldman Sachs zum bevorzugten Partner für Handel zu machen [20].

Was als interne Plattform angefangen hat konnte schnell umfunktioniert werden. Dies war durch ein offenes Design und geringen Abhängigkeiten der Plattform von Goldman Sachs möglich. Sie ist dabei den Kompromiss eingegangen eine wertvolle Anwendung unentgeldlich verfügbar zu machen, was kontrovers war. Im Gegenzug dafür war Goldman Sachs auf den lokalen Rechnern der Händler ihrer Geschäftskunden allgegenwärtig [20].

TECHNOLOGIEN UND PLATTFORMEN VON GOLDMAN SACHS

- "Data Lake" für Mashine-Learning durch eine *hybride Cloud*[1]
- "Marquee", eine Bewertungsplattform für Marktanalysen
- "SIMON", eine Plattform und Onlinemarktplatz für "Structured Notes"

1 Gemeinsamer Einsatz einer extern und einer intern betriebenen Cloud-Plattform

NEUE GESCHÄFTSMODELLE Mit der Entwicklung von SIMON, einem Onlinemarktplatz für "Structured Notes", erhöhte Goldman Sachs ihre Reichweite zu einer neuen Nutzergruppe. Das Wachstumspotenzial von einer Plattform mit einem einzigen Anbieter wurde erreicht, sodass die eigene Plattform an weitere konkurrierenden Emittenten geöffnet wurde. Dies erhöhte die Kundenzufriedenheit, da eine größere Auswahl und Vielfalt an Angeboten hierdurch entstand [20].

3.1.1 *Zusammenfassung zu Goldman Sachs*

Gupta [20] fasst zusammen, dass die Realität nach 2008 Goldman Sachs und ihre Konkurrenz gezwungen hat sich neu auszurichten und wirft einige Fragen auf: [2]

> "Einblicke in die Strategie von Goldman Sachs wurden durch Initiativen wie Marquee und Marcus deutlich, aber wie passten diese und andere Initiativen der letzten Zeit in das allgemeine Geschäftsmodell des Unternehmens? Hat sich das Kerngeschäft von Goldman Sachs verändert, oder waren Produkte wie SIMON und Marcus an der Peripherie angesiedelt? Was trieb das Unternehmen dazu, den Zugang zu internen Tools zu öffnen, die lange Zeit als proprietärer Wettbewerbsvorteil angesehen worden waren, und wie konnte es rechtfertigen, Wettbewerber zum Verkauf an seine Kunden einzuladen? "

Die Realität nach 2008 könnte diese Fragen von sich aus beantworten. Hierzu sollten die veränderten Rahmenbedingungen der IT im Finanzwesen verstanden werden. Zudem ist die Realität zu erkennen, dass neue Technologien immer schneller entstehen und ihre Auswirkungen immer stärker werden.

Eismann [10] beschreibt, dass aus den Plattformansätzen von Apple und Google einiges gelernt werden kann. Beide Unternehmen bieten Plattformen für Apps an und verdienen an ihrem Vertrieb deutlich mit. Dabei müssen sie die Apps nicht selbst entwickeln. Durch Plattformen findet eine *Auslagerung von Innovation [10]* statt.

Daher ergibt es Sinn, dass Goldman Sachs sich die Plattformansätze von führenden Technologieunternehmen aneignet und selbst ein Ökosystem mit Plattformen aufbauen will.

Vielen Banken scheint diese Idee jedoch fremd [10]. Eismann [10] begründet, dass Kernbankensysteme niemals dafür ausgelegt wurden externe Software zu integrieren, was aus Sicherheitsgründen vertretbar ist, jedoch eine starke Limitation für Innovation aufgrund kostspieligen Anpassungen darstellt.

Alt, Auth und Kögler [1, S. 14] definieren, dass der aktuelle Innovationsverlauf auf *Diskontinuitäten* beruht und disruptiv stattfindet und haben hierzu einen entscheidenden Vorschlag:

2 Gupta [20], S. 10, aus dem Englischen übersetzt

> "Für die Vorwegnahme künftiger Innovationen ist die Gestaltung künftiger Zukunftszustände bzw. Verwendungszusammenhänge durch die zeitnahe Realisierung von Prototypen von besonderer Bedeutung."

Besonders sollte hierbei die Gestaltung *künftiger Zukunftszustände [1]* hervorgehoben werden. Das IT-Management muss in der Lage sein Technologietrends zu verstehen und ihre Auswirkungen auf ihr Geschäft zu transferieren. Dabei gibt es Unterschiede zwischen Diskontinuität und Disruption von Technologien [11]. Die *künftigen Verwendungszusammenhänge* sind hierbei für die Wertschöpfung von neuen Opportunitäten wichtig. Opportunitäten können jedoch verpasst werden, wenn sie nur eine Erweiterung von Geschäftsmodellen darstellen. Um die Kontinuität des Kerngeschäfts aufrecht zu erhalten sind die *künftigen Zukunftszustände [1]* essenziell. Für die zeitnahe Realisierung von Prototypen erfordert es kreative [1] und schnelle Ansätze, wie zum Beispiel der *Design Thinking* Prozess.

3.2 DISKONTINUITÄT UND DISRUPTION

Die unflexible IT-Architektur in Banken ist keine neue Erkenntnis und wird seit langer Zeit gefordert [4], [5]. Mittlerweile sollte herbei nicht mehr von Bedarf oder Forderung gesprochen werden. Die Opportunität für eine flexible SOA oder ist bereits durch Cloud-Plattformen vergangen. Die geforderte SOA wurde zuletzt durch Mircroservice-Architekturen ersetzt. Diese Beobachtung ist Beispiel für eine *Diskontinuität* durch die Cloud-Computing Technologie.

Veränderungen werden nicht nur durch ein Wertschöpfungspotenzial hervorgerufen. Viel mehr wird die IT in Betrieben mit alten Strukturen durch äußere Kräfte gezwungen sich anzupassen [1], [20]. Grundlage hierfür sind neue Technologietrends [5], die mittlerweile in ihrer Auswirkung jedoch gravierender sind. Eine entsprechende Verdrängung im Wettbewerb kann viel schneller erfolgen als bisher. Neben Impulse für Innovation mit gefolgter Diskontinuität [1] kann zunehmend eine Disruption durch neue Technologietrends beobachtet werden. Fernández, Valle und Pérez-Bustamante [11] definieren die voneinander schwierig zu unterscheidenden Konzepte der "diskontinuierlichen" und "disruptiven" Technologien.

Verdrängungen im Wettbewerb finden nicht mehr durch eine Konkurrenz auf gleicher Augenhöhe statt. Eine Verdrängung kann auch unerwartet von kleinen Betrieben mit effizienten Abläufen kommen, die durch Schnelligkeit, Verfügbarkeit, Skalierbarkeit und Qualität ihrer Leistungen überzeugen kann. Gerade im Bankwesen könnten Verdrängungen stattfinden, die jenseits der bisherigen Rahmenbedingungen agieren. Der hierfürige Technologietrend könnte im Finanzwesen beispielsweise die Blockchain-Technologie sein. Analog zum genossenschaftlichen Prinzip der Selbsthilfe, Selbstverantwortung und Selbstverwaltung könnten unabhängige Organisationen entstehen, nach dem historischen Muster, aus denen die Genossenschaftsbanken

entstanden sind.

Die Vorwegnahme künftiger Innovation durch die Gestaltung von Zukunftszuständen [1] ist ein Ansatz, um insbesondere Diskontinuitäten entgegen zu wirken. Für Disruptivität kann hierzu ein zusätzlicher Fokus ergänzt werden. Während der Gestaltung von Zukunftszuständen könnten Risiken für das Kerngeschäft mit kreativen Ausblicken auf disruptive Technologien antizipiert werden. Eine realistische Bedrohung auf den künftigen Zustand des Unternehmens könnte den nötigen Katalysator für einen Paradigmenwechsel liefern oder zumindest den Fokus der Risikomaßnahmen auf Innovation setzen.

Nach einem erfolgreichen Paradigmenwechsel, durch die Erkennung der Bedrohungslage sollten disruptive Technologien mit den aktuellen Rahmenbedingungen abgeglichen werden. Dadurch könnten die Einschränkungen für Innovation identifiziert werden und es könnte definiert werden in welchen Bereichen ein Paradigmenwechsel erforderlich ist.

FINTECHS Aus den FinTechs entstehen mittlerweile volllizensierte Banken mit direktem Vertrieb über das Internet, wie zum Beispiel N26. Dazu entstehen Banken, die Schnittstellen für Unternehmen anbieten und somit eine reine Plattform für Finanzdienstleistungen sind, beispielsweise die Solarisbank. Auch etablierte Banken haben veränderte Rahmenbedingungen und Impulse erkannt und passen sich dem aktuellen Stand der Technik an [20], [10].

Darren Cohen[3] sieht im Bereich Wertpapierhandel von FinTechs keine Gefahr für Goldman Sachs [20]. Hierzu wird in [20] aufgeführt, dass laut Cohen die Anforderungen für neue Marktteilnehmer schwierig sind und für junge FinTechs unüberwindlich. In [20] ist nach Cohen die Abwicklung im Bereich Wertpapierhandel hochreguliert und erfordert eine hohe Bilanzsumme, hohe Infrastrukturinvestitionen und einen erheblichen, globalen Einfluss.

Cohens Begründung pauschalisiert jedoch die Rahmenbedingungen der FinTechs. Abwicklungen mit hohen Summen finden bereits über Blockchain Plattformen statt. Diese Unternehmen agieren jenseits der Rahmenbedingungen und Regulierungen von konventionellen Instituten (Kap. 3.2).

BLOCKCHAIN Im Gegenzug zu Cohens Anforderungen für neue Marktteilnehmer (Kap. 3.1) gelten einige Blockchains als hochgesichert, enthalten hohe Bilanzsummen, besitzen einer der vermutlich leistungsstärksten Infrastrukturnetzwerke weltweit und sind dadurch global in jeder Hinsicht.

An der Distributed-Ledger Technologie (DLT) können sich unter anderem FinTechs beteiligen und die Eintrittsbedingungen mit Leichtigkeit überwinden. Zu den bekanntesten Handelsplattformen, die aus der Blockchain-Technologie entstanden sind zählen, Kraken, Poloniex, Bitfinex, Shapeshift, Coinbase, um einige zu nennen.

3 Vorsitzender von Goldman Sachs' PSI Group, vgl. [20]

Abwicklungen in diesen Handelsplätzen finden unter anderem dezentral mithilfe von Smart-Contracts und Reserven statt [17].

Die DLT wird auch von etablierten Instituten im Finanzwesen genau verfolgt. Man könnte argumentieren, dass dies aus der Befürchtung vor dem nächsten Sprung einer Innovation und somit Disruption rührt [1, 18]. Diese Disruption könnte weitaus stärker sein als der letzte Innovationssprung mit Cloud-Computing.

Für die Abwicklung von Schuldscheindarlehen existiert jüngst das deutsche Pilotprojekt finledger [35]. Basierend auf der DLT erschafft ein Konsortium aus deutschen Kreditinstituten die Plattform *finledger*, die zu einem Standard werden und nach der Pilotierung allen Instituten zugänglich werden soll [35]. Dieses Pilotprojekt ist ein wesentlich wichtiger Schritt in die richtige Richtung. Innovationen erfordern Gestaltungsmethoden, die auf eine schnelle Generierung von Prototypen und somit Datenerhebung arbeiten. Selbst wenn die Plattform nicht den wirtschaftlichen Effekt erzielt, erzielt es den weitaus wichtigeren Innovationseffekt als eine Anlage, die Innovation antreibt[18]. Koch, Ahlemann und Urbach [25] bezeichnet solche antreibenden Anlagen als "capabilities".

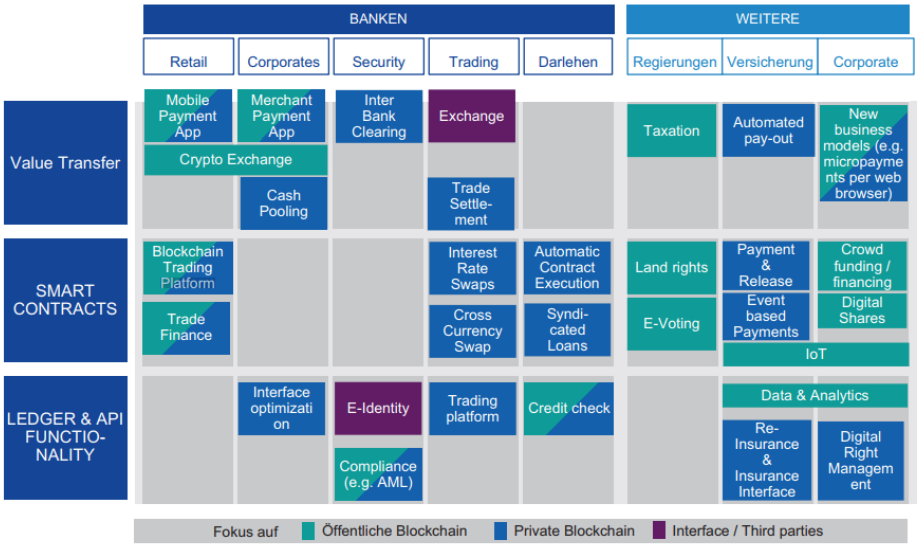


Abbildung 3.1: Korschinowski et al. [26], S. 283, "Anwendungsfelder der Blockchain-Technologie. (Quelle: KPMG)"

In Abb. 3.1 wird ein Überblick der Anwendungsgebiete der DLT gezeigt. In einigen Anwendungsbereichen könnten hieraus Ideen für zukünftige Plattformen nach der Vision von Goldman Sachs 3.1 entstehen.

IDEE: SOFTWARE DEVELOPMENT "AS CODE" Eine künftige Disruption würde möglicherweise als logische Folge auf die Auslagerung und Dematerialisierung fast aller Komponenten in der IT entstehen. Es stellt sich die Frage ob der Entwickler oder die Entwicklungsarbeit selbst als eine vollautomatisierte Dienstleistung bereitgestellt werden kann. Dieses zu-

künftige Konzept könnte als Trend "Software Development as Code"[4] genannt werden, als Folge auf die neuen Rahmenbedingungen durch Cloud-Architekturen. Hierbei wird auch eine mögliche Vollautomatisierung von IT-Projektmanagement impliziert durch Orchestrierung der Einheiten. Der Begriff "Software Development as a Service" existiert bereits im Rahmen einer Serviceorientierung der Entwicklung durch agile Methoden [27].

Freelancing und *Consulting* ist verbreitet in der IT und kann insbesondere aufgrund der zunehmenden Möglichkeiten und Akzeptanz für Remote-Arbeit sich immer weiter entwickeln.

Dieses Konzept wäre aus Sicht der Innovationsförderung nicht negativ behaftet. Denn dadurch würden Zusammenarbeitsmodelle gefördert, die auf Offenheit, Agilität, Modularisierung und Kontinuität basieren und sich effektiv in den Ergebnissen auch zeigen und die Softwareentwicklung als solche optimieren.

Möglicherweise könnte nach einer Transformation der gesamten IT-Architektur hin zu einer Cloud-Architektur eine Disruption oder Diskontinuität entstehen mit unvorsehbaren Auswirkungen auf die IT. Cloud-Plattformen bieten eine Architektur, die allgegenwärtig und fortschrittlicher sein würde. Entwicklungs- und Produktivumgebungen werden weiterhin streng getrennt sein müssen [13]. Das ist jedoch kein Hindernis für neue Geschäftsmodelle in der Softwareentwicklung, so lange eine kontinuierliche Zusammenarbeit zwischen diesen Bereichen besteht.

Die Voraussetzungen aus regulatorischer Sicht wären die Rückverfolgbarkeit von Ergebnisartefakten im Softwareentwicklungsprozss, mithilfe von Automatisierung und Nachvollziehbarkeit der Delivery Pipeline und Risikomaßnahmen. Durch den Zugang einer Instanz der Entwicklungsumgebung gegenüber diesen Plattformen könnten Dienstleistungen für Entwicklungsarbeit analog zum Cloud-Computing gebucht werden.

Je nach Auslegung und Perspektive wäre diese beispielhafte Vorstellung eine Innovation oder Gefahr. Um ihr zu entgegnen sollte das Szenario durchgegangen werden. Für Entwickler entstehen möglicherweise Opportunitäten oder auch Risiken.

Solche Entwicklungen werden unvermeidbar sein und zeigen sich zuerst in Form von Impulsen [5] und nehmen mit zunehmender Reife der Technologie eine disruptive Eigenschaft an [1, 18]. Die genannten Beispiele für disruptive und diskontinuierlichen Technologien zeigen, dass die Entwicklung keinen Halt macht und ständig gestalterisch fortgeführt werden kann und sich früher oder später materialisieren wird.

3.3 REGULATORISCHER RAHMEN FÜR DIE IT IM FINANZWESEN

Die *Finanzkrise 2008* war mit einem enormen Vertrauensverlust gegenüber Banken verbunden und die Empörung der Gesellschaft ist durch das Internet und der daraus folgenden Transparenz viel größer als zuvor gewesen

4 Im Rahmen der vollautomatisierten Bereitstellung von Entwicklerteams und Beratern

[10]. Eine Welle von regulatorischen Veränderungen als Folge auf die Auswirkungen dieser Krise herbeigeführt, um zukünftige systemische Krisen zu vermeiden [20]. Das Ergebnis daraus ist ein Anstieg der regulatorischen Anforderungen mit gleichzeitiger Zunahme der technischen Anforderungen[5] in der IT.

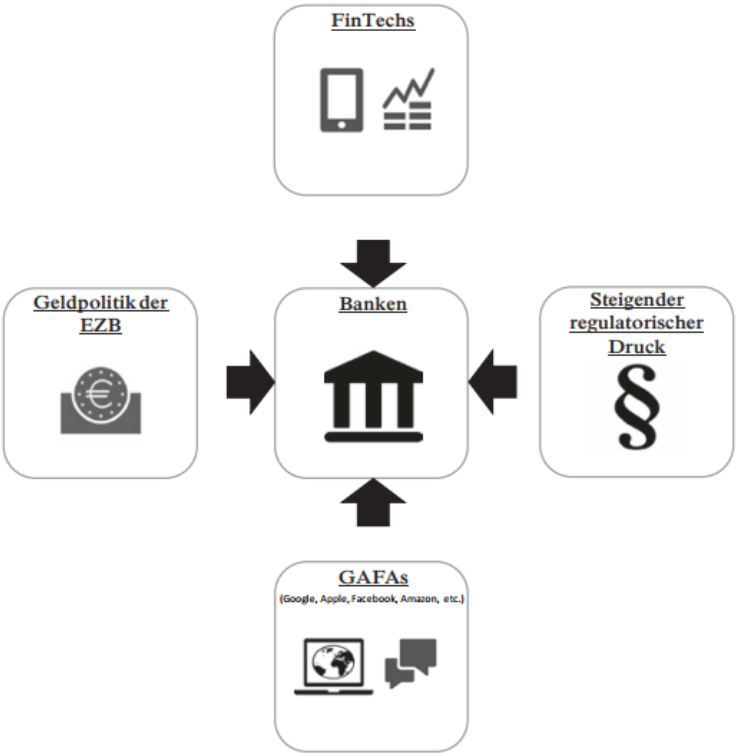


Abbildung 3.2: Finanzbranche unter Druck (Quelle:Smolinski und Gerdes [31, S. 44])

Daraus resultiert ein zunehmender Druck auf das Finanzwesen (Abb. 3.2), der sich in wirtschaftlichem Druck und zusätzlichen Kosten zeigt [31].

BAFIN ANFORDERUNGEN Aus dem KWG der Bundesrepublik Deutschland entstehen Anforderungen [13] an Kredit- und Finanzdienstleistungsinstitute, die sich auch auf die IT der Banken auswirken [3]. Im KWG wird die BaFin für die Aufsicht der Institute nominiert [3]. Sie veröffentlicht Vorgaben an die von ihr beaufsichtigten Institute, um eine einheitliche Verwaltungspraxis sicherzustellen, mit der sich die Institute auf die Prüfungen der BaFin einstellen [15].

Die BaFin präzisiert in [13] die Anforderungen des KWG im Bereich Risikomanagement und Auslagerung. In [14] werden diese Anforderungen für den Bereich der IT nochmals konkretisiert. Hierdurch sollen operationelle Risiken auch in der IT angemessen gesteuert werden. Knittl [24] beschreibt die Umsetzung dieser Anforderungen anhand einer Fallstudie zu einem internationalen Finanzkonzern.

5 insbesondere der "kontinuierliche Anstieg der Anforderungen an die IT-Architektur" [8]

FINANZWESEN UNTER DRUCK Abb. 3.2 verbildlicht, wie Banken von allen Seiten unter Druck stehen. Die Darstellung könnte fortgeführt werden mit der Annahme, dass die äußeren Kräfte sich auch gegenseitig beeinflussen. Die Technologien und Trends aus den "GAFAs" können mit Sicherheit als ein Antreiber für die FinTechs gesehen werden. Die Antreiber von Veränderung auf der vertikalen Achse der Abbildung werden von Einschränkenden Instituten auf der horizontalen Achse durchkreuzt. Die Aufsichten evaluieren ebenfalls neue Technologien und berücksichtigen sie jeweils bei der Novellierung ihrer Rahmenwerke [16]. Bis diese angepasst werden vergeht Zeit und bis sich die etablierten Institute hierauf auch anpassen vergeht ebenfalls wertvolle Zeit. In FinTechs ist dieses Problem aufgrund ihrer Agilität und Effizienz geringer. Etablierte Institute müssen daher einen Weg finden das Zusammenspiel zwischen neuen Technologien und regulatorischen Anforderungen zu antizipieren und eigene Initiativen als Pilotprojekte schon zu starten. Hierzu passt ebenfalls der Ansatz der Vorwegnahme durch die Gestaltung von Zukunftszuständen [1].

ERMESSEN DER INSTITUTE In [13, 14] fällt auf, dass viele Vorgaben durch entsprechende Wörter in ihrem Gewicht relativiert werden. Die Maßnahmen zu ihrer Umsetzung liegen hierbei im Ermessen der jeweiligen Institute. Im Interesse der Institute ist in erster Linie das Bestehen von Prüfungen der Aufsicht. Die Ermessensfreiheit könnte in einigen Bereichen für Unklarheit sorgen, da die Mindestanforderungen an das Risikomanagement (MaRisk) sehr abstrakt formuliert ist [19]. Eine Kultur innerhalb der internen Kontrollverfahren die sich nur auf Umsetzung beschränkt zum bestehen von Prüfungen sollte als Folge vermieden werden. Eine Mitgestaltung ist hierbei wichtig um eine gemeinsame Wissensplattform aufzubauen. Giroverband [19] hat einen gemeinsamen Leitfaden gestaltet, dass selbst in Papierform eine Plattform darstellt. Selbst gemeinsame Wissensplattformen könnten einen ersten Beitrag zur Innovationskraft sein und somit als Anlage gesehen werden. Der antreibende Faktor dahiner wäre die Kommunikation und Schnittstellen. Die Voraussetzung hierfür wäre Offenheit.

Die Flexibilität, die die BaFin in ihren Rahmenwerken vorsieht [13] sollte als stärke wahrgenommen werden und auf die interne Kultur übertragen werden. Die internen Kontrollverfahren und das Risikomanagement müssen in erster Linie praxisnah und flexibel bezüglich der IT vorgehen und ihren Ermessensspielraum zu Nutze machen, um mit neuen Technologien umgehen zu können. Die Ermessensfreiheit, die die BaFin beschreibt erzeugt nur Druck und Unsicherheit in Instituten, denen es an Gestaltungsfähigkeit fehlt. Die BaFin verlangt mit der IT-Strategie [14] ein aktives Mitgestalten seitens Management. Ebenso sieht sie genug Gestaltungsfreiheit für das Risikomanagement vor. Daher sollte das Risikomanagement und das ITM diesen Ansatz sich aneignen. Banken sind zugegebenermaßen auf Profitmaximierung angewiesen.

Für die MaRisk gibt es auch zusätzliche Erläuterungen der BaFin [12]. Speziell für die IT hat die BaFin in [14] die MaRisk noch einmal konkretisiert. Daneben gibt es institutspezifische Interpretationen und Werke [19].

RISIKOMANAGEMENT Für das Risikomanagement fordert die BaFin die Festlegung von Strategien und die Einrichtung von internen Kontrollverfahren, bestehend aus einem Internes Kontrollsystem (IKS) und einer interne Revision. Ein Risikomanagement für die IT soll insbesondere den Betrieb schützen [13], da hierin die operationellen Risiken liegen.

Ein Ausfall der IT-Systeme in Banken hätte mit Sicherheit Auswirkungen auf kritische Geschäftsprozesse, die bis hin zu einer systemischen Krise führen könnten.

Daher fordert die BaFin [13] vom IT-Risikomanagement:

1. Sie soll Überwachungs- und Steuerungsprozesse für IT-Risiken einrichten.
2. Ihre Prozesse umfassen: Risikokriterien, Risiken, Schutzbedarf, Schutzmaßnahmen, Risikobehandlung und -minderung
3. Sie soll Risiken beim Einkauf von Software *angemessen* bewerten.

Diese Forderungen [13] zur Bewertung von IT-Risiken gelten auch beim Einsatz von selbst entwickelten Anwendungen, die von ihr als Individuelle Datenverarbeitung (IDV) bezeichnet wird. Entsprechende Maßnahmen sollen jedoch nach dem Schutzbedarf der unterstützten Prozesse und verarbeiteten Daten festgelegt werden [13].

Die Überwachungs- und Steuerungsprozesse für IT-Risiken sollten daher klare Grenzen für die Verarbeitung von Daten definieren. Eine klare Unterscheidung zwischen IDV Anwendungen und Anwendungen die keine IDV darstellen ist nötig, um angemessene Maßnahmen festzulegen. Da IDV Anwendungen jedoch hierbei pauschalisiert gesehen werden und aufgrund der darauf folgenden aufwendigen Maßnahmen negativ behaftet sind wird ihre Katalogisierung entmutig. Somit entsteht oft eine “Schatten-IT” in Banken [9].

Die Ermessensfreiheit zu den Maßnahmen des Risikomanagements begrenzt sich in diesem Fall auf den Schutzbedarf der Geschäftsprozesse und Daten. Daher muss die Katalogisierung von Anwendungen bezüglich ihres Schutzbedarfs in einem Spektrum ausgeweitet. Es kommt darauf an welche Daten von der Anwendung verarbeitet werden. Dieser Punkt sollte nachvollziehbar zu den Anwendungen dokumentiert werden.

IT-SYSTEME UND PROZESSE Die BaFin [13] fordert zu IT-Systemen und Prozessen:

1. IT-Systeme und Prozesse sollen Integrität, Verfügbarkeit, Authentizität und Vertraulichkeit der Daten mit gängigen Standards sicherstellen.
2. IT-Systeme und ihre zugehörigen Prozesse sollen hinsichtlich ihrer Eignung regelmäßig überprüft werden.

3. IT-Systeme sollen nach *wesentlichen* Veränderungen und vor erstmaligem Einsatz getestet werden.

4. Ein Regelprozesse der Entwicklung, des Testens, der Freigabe und der Implementierung in die Produktionsprozesse soll etabliert werden.

5. Produktions- und Testumgebungen sollen grundsätzlich voneinander getrennt werden.

Besonders hervorzuheben ist die Verantwortung der IT-Systeme und Prozesse für die Sicherstellung der Datensicherheit. Im Fokus stehen die verarbeiteten und gespeicherten Daten in den IT-Systemen und Prozessen.

Hierzu fordert die BaFin [13] eine Orientierung am Schutzbedarf der verarbeiteten Daten. Sie verweist mit gängigen Standards auf den IT-Grundschutzkatalog[6] des Bundesamtes für Sicherheit in der Informationstechnik (BSI) und auf den internationalen Sicherheitsstandard ISO/IEC 2700X[7] der ISO.

Die Auslagerung von IT-Ressourcen wäre beispielsweise bei der Entwicklung von Modellen für Mashinelles-Lernen nötig und dürfte aufgrund der Datensicherheit nicht ohne weiteres auf einer externe Cloud-Plattform durchgeführt werden.

Auch im Bereich der Automatisierung von Prozessen ist es wichtig zu erkennen, welche unterstützten Prozesse schutzbedürftig sind und welche nicht. IT-Prozesse können grob zwischen *Change* und *Run* unterteilt werden [9]. Die Run-Prozesse regeln den Betrieb und die Change-Prozesse erzeugen üblicherweise als Projektform Veränderungen im Aufbau oder Ablauf der Organisation.

Die Softwareentwicklung selbst enthält innerhalb einer getrennten Testumgebung auf dem ersten Blick keine operationellen Risiken für den Geschäftsbetrieb der Bank. Für die Integrierung von Ergebnissen aus diesen Change-Prozessen in den Betrieb ist eine vorherige Freigabe vorgesehen, beispielsweise in Form von "Quality-Gates" [7].

Für ihre Unterstützung existiert innerhalb der IT ein eigener Betrieb für Plattformen und Anwendungen der Softwareentwicklung. Ein hoher Schutzbedarf der Entwicklung entsteht in Folge von Schnittstellen bedingt durch Prozesse für die Inbetriebnahme von Software. Besonders ist in der Anwendung von DevOps die Delivery-Pipeline, womit die Komponenten in die Produktionsumgebung integriert werden ein kritischer Prozess. Nach DevOps sind die Abläufe zwischen CI und CD für einen automatisierten und hochfrequenten Ablauf eng miteinander verbunden. In CD liegt auch die Verantwortung über Risiken und Qualitätssicherung [1]. Es stellt sich die Frage wie Anwendungen zur Ausführung der Delivery Pipeline bezüglich ihres Schutzbedarfs einzuordnen sind.

KONSEQUENZEN FÜR DIE SEU Grenzen zeigen sich zum Beispiel in Form von unterschiedlichen Zonen innerhalb der IT-Architektur. Diese Grenzen

6 vgl. IT-Grundschutzkompendium [22]
7 vgl. Disterer [8]

sind nötig, um die Risiken für den Betrieb kontrollieren zu können. Hierfür wird eine klare Nachvollziehbarkeit der Ergebnisse aus den Change-Prozessen bedingt.

Innerhalb der SEU sind die Rollen schwierig voneinander zu unterscheiden. Konfiguration, Administration, Wartung und Weiterentwicklung von Anwendungen für die SEU sind Tätigkeiten in denen Kunde, Entwickler und Betrieb (Abb. 2.1) teilweise ineinander verschmelzen. Auch die technische Umsetzung der Überwachungsprozesse erfolgt meist von den gleichen Entwicklern der SEU. Die enge Kopplung zwischen Entwicklung und Betrieb ist in DevOps vorgesehen [] und setzt agile und kontinuierliche Prinzipien voraus, die aus Sicht des Risikomanagements schwierig zu kontrollieren sein könnten.

Die Auslagerung der SEU auf externe Infrastrukturen ist aufgrund dieser engen Kopplung der Abläufe in DevOps beschränkt, weil hierdurch sensible Daten ausgetauscht werden könnten. Zudem könnte eine zu hohe Abhängigkeit zu einer Cloud-Plattform für den Betrieb der SEU entstehen. Realistischer wäre es bei Bedarf nur einzelne rechenintensive Services auszulagern mit Ansätzen für eine hybride Cloud [20].

Selbst wenn diese Modularisierung mit neuen Technologien wie Container-Virtualisierung realisierbar ist, bestehen für die Verwendung von externen Cloud-Plattformen Probleme bezüglich der Datensicherheit.

WESENTLICHE VERÄNDERUNGEN Bei wesentlichen Veränderungen (*Change-Prozesse*) bedingt die BaFin [13] vom Risikomanagement die Auswirkung der Veränderung auf die Kontrollverfahren zu analysieren. Hierfür sind die später in die Arbeitsabläufe eingebundenen Organisationseinheiten, Risikocontrolling, Compliance und die Interne Revision zu beteiligen.

Daraus könnte ein Hindernis für solche Veränderungen entstehen. Wesentliche Veränderungen in den IT-Systemen könnten einen Anpassungsbedarf des Kontrollverfahrens bedeuten. Vielmehr sollte das interne Kontrollverfahren von einer ständigen Veränderung in der IT ausgehen und sich auf Auswirkungen vorbereiten.

BANKAUFSICHTLICHE ANFORDERUNGEN AN DIE IT Die Bankaufsichtliche Anforderungen an die IT (BAIT) gibt seit 2018 neben der MaRisk auf der gleichen Grundlage des KWG einen Rahmen, insbesondere für das Management der IT-Ressourcen und das IT-Risikomanagement. Die Anforderungen aus MaRisk werden lediglich konkretisiert. Zudem Präzisiert es die Anforderungen des § 25b des KWG zu Auslagerung von Aktivitäten und Prozessen [14].
Die BAIT ist daher besonders für die Auslagerung von IT-Ressourcen zu berücksichtigen und insbesondere aufgrund der zunehmenden Praxis von SaaS und Cloud-Computing relevant.

IT-STRATEGIE Die Geschäftsleitung soll eine nachhaltige IT-Strategie festlegen, die konsistent mit der Geschäftsstrategie ist [14].

Aus den Mindestinhalten der geforderten IT-Strategie wird deutlich, dass Geschäftsmodell und IT in einem engen Verhältnis gesehen werden. Die IT steht aus regulatorischer Sicht nicht mehr größtenteils als Risikofaktor im Fokus. Vielmehr kann sie aufgrund einer konkreten Strategie das Geschäftsmodell ergänzen. Diese Anforderung ist eine klare Förderung der Rolle der IT.

IT-GOVERNANCE Die IT-Governance wird als Struktur zur Steuerung und Überwachung des IT-Betriebs und der Entwicklung der IT-Systeme und Prozesse auf Basis der IT-Strategie [14] gesehen. Für das Risikomanagement, Entwicklung und Betrieb innerhalb der IT wird eine qualitativ und quantitativ angemessene Ausstattung mit Personal gefordert.

EBA GUIDELINES Bedeutende Institute werden zusätzlich von der EBA beaufsichtigt [3]. Die EBA veröffentlicht für sie eine Richtlinie [2], die sich auf die MaRisk bezieht und für die systemrelevanteren Institute wichtig sind.

IDEE FÜR EIN INNOVATIVES RISIKOMANAGEMENT Die von Disterer [7] beschriebenen Probleme im Betrieb von Anwendungen durch die mangelnde Beachtung von nichtfunktionalen Anforderungen und die Einführung von Quality Gates könnten hierbei auf Probleme für die internen Kontrollverfahren ausgeweitet werden.

Eine Koordination zwischen Entwicklung und Betrieb sollte das IT-Risikomanagement miteinbeziehen. Dabei sollte die Geschwindigkeit des Entwicklungs- oder Integrationsprozesses nicht beeinträchtigt werden. Hierfür sollte das IT-Risikomanagement agile Methoden verwenden. Die Schnelligkeit hat bei der Beantwortung von Compliancefragen dabei eine wichtige Rolle. Entwickler könnten zuerst die Prozesse des IT-Risikomanagements optimieren und automatisieren, um sich Handlungsfreiheit für Innovation und wesentlichen Veränderungen innerhalb der Organisation zu schaffen. Sie sollten in das Risikomanagement integriert werden. Ein Entwickler aus einem anderen Projekt könnte als Compliance- und Risikoverantwortlicher parallel wichtige Compliancefragen auf technischer Ebene schnell überprüfen und vorab beantworten. Dadurch könnte unter Vorbehalt schon mit neuen Technologien und Methoden an Lösungen gearbeitet werden oder Veränderungen in Gang gesetzt werden.

Um einer Disruption zu begegnen ist das Risiko von verschwendetem Aufwand in Kauf zu nehmen, um sich eine Anpassungs- und Reaktionsfähigkeit zu schaffen, die geschäftskritisch ist [5].

3.3.1 *Paradigmawechsel der Regulatorik*

RELEVANZ DER REGULATORIK FÜR ENTWICKLER Die BaFin Anforderungen [13, 14], Gängige Sicherheitsstandards [8, 22] und weitere Rahmen sind für Entwickler wichtig, um ihre Verantwortung über die IT-Risiken in Banken und anderen systemrelevanten Betrieben zu verstehen. Insbesonde-

re sind sie für die nachvollziehbarer Entwicklung von Anwendungen und Systemen und für ihren nachhaltigen Betrieb, Überwachung und Überprüfung zu berücksichtigen. Um verwachsene und unflexible Architekturen zu vermeiden sollten Entwickler, im Bereich der Anwendungen, die interne Prozesse unterstützen [5], sich mit diesem Thema auseinandersetzen.

Neben den Leistungsansprüchen sollten die Sicherheitsansprüche für Veränderungen im Vordergrund stehen. Die Implementierung von neuen Technologien wirft bei wesentlichen Veränderungen viele Compliancefragen auf [13]. Diese hängen mit den Sicherheitsansprüchen zusammen und sind unvermeidbar, können aber durch eine Effizienzsteigerung der Steuerungs- und Überwachungsprozesse zu einer Stärke werden und die Fähigkeit wesentliche Veränderungen durchzuführen erhöhen

Aufgrund befürchteten aufwendigen Maßnahmen sollte die Durchführung von wesentlichen Veränderungen nicht entmutigt werden. Die Maßnahmen fokussieren sich den Schutzbedarf von kritischen Prozessen und Daten (Kap. 3.3). Es wäre daher eine Selbstbeschränkung die Maßnahmen aufgrund eines mangelnden Verständnisses der Regulatorik zu pauschalisieren, somit die Ermessensfreiheit aufzugeben und präzisere Vorgaben von der Aufsicht zu fordern.

Ein "Schwarz-Weiß" Denken als Folge zum Bestehen von Prüfungen oder die Umsetzung des Trugschlusses "Security through Obscurity" sind zu vermeiden. Das Pauschalisieren der geforderten Maßnahmen des Risikomanagement könnte eine "Schatten-IT" [9, S. 104] erzeugen, aufgrund ihrer Befürchtungen.

Ein Beispiel wäre die Katalogisierung von IDV Anwendungen [14], [9, Kap. 6.8]. Ein mangelndes Verständnis über die Hintergründe der Forderungen würde sich in einer mangelnden Selbsteinschätzung der Maßnahmen zeigen. Dies hätte zu Folge, dass Entwickler aufgrund ihrer Befürchtungen gegenüber den Maßnahmen die Kommunikation und somit Offenheit gegenüber des Risikomanagements und ITM einschränken.

3.3.2 *Aufsicht und IT-Strategie*

Bezüglich des Finanzwesens befindet sich möglicherweise auch die deutsche Verwaltungspraxis im Wandel. Insbesondere hat die Cloud-Technologie neue Impulse für das Thema *Auslagerung von IT-Ressourcen [13, 14]* erzeugt. Die MaRisk wird voraussichtlich 2021 erneuert und darüber hinaus sind Erleichterungen aufgrund der Umsetzung der EBA-Richtlinien [2][8] zu erwarten [16].

Die Anpassung der Verwaltungspraxis kommt jedoch gezwungenermaßen und konnte dem Veränderungsdruck der Institute (Abb. 3.2) nicht standhalten. Die Aufsicht und ihre beaufsichtigten Institute sollten sich einen Weg überlegen, wie Veränderungen kontinuierlich stattfinden können, wie sie

8 seit 30. Juni 2020 in kraft getreten

Bussmann u. a. [5] fordern.

Im Gegensatz zur MaRisk nimmt die EBA im Anhang ihrer Richtlinie [2] Stellung zum Thema Innovation, Transformation sowie IT-Strategie.

Die EBA fasst den Vorschlag eines Korespondenten zusammen [2, S. 50]:

> " Der Text [9] sollte auch die Ausrichtung der IKT-Strategie[10] nach Innovation enthalten, zur Vermeidung von Disruption und zur Unterstützung schlanker digitaler Transformation, die auf IKT-Architektur basiert. Es[11] sollte auch das richtige Portfolio von Änderungen enthalten, zur Ausrichtung der IKT-Transformation in Einklang mit Geschäftstransformation. "[12]

Die Sorgen der Beteiligten zu disruptiven Verläufen [1, 18] von Technologiesprüngen und der damit zusammenhänge Aufwand eine Transformation der Architektur durchzuführen wird hierbei deutlich. Hierbei ist auch von richtigem Portfoliomanagement die Rede, um für die richtigen Änderungen zu priorisieren.

Die Antwort der EBA hierauf verweist auf die IT-Strategie, die Teil der Gesamtstrategie ist [2, S. 14ff]. Somit stehen Banken in der Eigenverantwortung sich eine Strategie für disruptive Technologiesprünge zu erarbeiten.

9 Bezogen auf *Guidelines on ICT and security risk management* [2, S. 14ff]
10 Informations- und Kommunikationstechnik (IKT) -Strategie übersetzt von "ICT-Strategy"
11 Die Richtlinie
12 Übersetzt aus dem Englischen, Respondent unbekannt

4

INNOVATIONS- UND TRANSFORMATIONSPROZESSE

Dieses Kapitel analysiert die Einflüsse aus den Rahmenbedingungen auf Veränderungsprozesse von Organisationen und trägt kontinuierlich zur Findung der Ursachen der genannten Probleme und Ansätze für ihre Lösung bei. Der *Design-Thinking* Prozess wird hierbei als iterative Gestaltungsmethode für Ideen, Prototypen und kontinuierlicher Evaluierung verwendet. Daher bauen die nachfolgenden Abschnitte aufeinander auf und sollen somit zu einer kontinuierlichen Verbesserung beitragen und in ihrer letzten Ebene ausgeliefert werden.

4.1 EINFLÜSSE UND KRÄFTE DER INNOVATION

Ganswindt [18] und Koch, Ahlemann und Urbach [25] identifizieren bereits, dass das ITM nicht in der Lage ist die Maßnahmen für eine Transformation zu priorisieren. Die gegenseitigen Einflüsse sind ineinander verwachsen und es werden häufig zu erst die Symptome bekämpft (Anh.A, Kap. 2.3).

KOMPONENTEN DER LEGACY-IT Aufgrund dieser Verwachsungen wird in der folgenden Auflistung eine grobe Lokalisierung und Zusammenfassung vorgenommen:

- **Organisation:** tiefe Silos [20], Schatten-IT [3], Abhängigkeiten, Handlungsunfähigkeit, Redundanzen, mangelnde Priorisierung, langsame Prozesse
- **IT-Architektur:** Komplexität [4], steigende Anforderungen [4], Kosten für Eigenentwicklung [20], Proprietäre Anwendungen [5], veraltete Systeme, mangelnde Skalierbarkeit, Legacy-IT
- **Regulatorik:** IDV Katalogisierung[3], fehlende Präzisierung, steigende Anforderungen

Es fällt auf, dass es erheblich schwierig ist die Einschränkungen für eine Transformation zu faktorisieren. Es ist weitaus einfacher direkte Gegenmaßnahmen auf die beobachtete Einschränkung vorzuschlagen oder antreibende Eigenschaften wie Agilität zu fordern. Daher werden die antreibenden Faktoren nachfolgend betrachtet.

4.1.1 *Innovationsverlauf als disruptive S-Kurven*

Alt, Auth und Kögler [1, Kap. 2.2] erklären die Abläufe einer Innovation. Die Abbildung [1, Abb. 2.1] beschreibt einen Innovationsverlauf als eine *S-Kurve* [18], die in Folge mehrerer aneinander gereihte Innovationen gesamtheitlich

betrachtet einen disruptiven Verlauf mit Sprüngen zeigen. Darunter liegt die Adaptionskurve, die als die Ableitung der Innovationskurve erscheint.

Einschränkenden Faktoren, die in Banken beobachtet werden, könnten auf dem ersten Blick als Faktor der Adaptionskurve in [1, Abb. 2.1] zugeordnet werden und die Adaption beschränken. Demgegenüber stehen die zu erarbeitenden Gegenmaßnahmen zur Lösung des Problems. Instinktiv können einige Einschränkungen, die eine Eigenschaft darstellen negiert werden. Komplexität wird beispielsweise zu Nachvollziehbarkeit. Die Meisten benannten Einschränkungen sind jedoch keine Effekte, die die Innovationskraft [18] beeinflussen. Sie beeinflussen aber die bevorstehende Transformation.

Innovationen unterliegen kreativen Prozessen [1, S.14], weswegen eine kreative Methode sich für ihr Verständnis eignen könnte. Gerade der *Design-Thinking* Prozess sieht die iterative Gestaltung und Evaluierung von Prototypen vor. Dadurch können Zusammenhönge untersucht werden und der Lösungsansatz kann ständig angepasst werden.

In folge wurde Abbildung 4.1 in mehreren Iterationen der Syntheseschritte "ideate-prototype-testing" aus dem Design-Thinking Prozess (Kap. 2.2.1) mithilfe von den Punkten aus [1, S. 14] und dem erarbeiteten Rahmen in Kap. 3 generiert.

BESCHREIBUNG INNOVATIONSVERLAUF Innovationen und Innovationsverläufe sind voneinander zu unterscheiden. Die Innovation ist das Neue, während ein Innovationsverläuf als ihr Einschlag in die Organisation oder Gesellschaft gesehen werden kann. Ein Innovationsprozess wäre eine gezielte Regelung oder Steuerung nach dem Einschlag.

Ein Innovationsprozess setzt daher einen Auslöser voraus und beginnt mit *Sharing* (Abb. 2.1).

Dieser Einschlag kann kontrolliert anhand einer Strategie in die Umgebung gesetzt werden oder durch äußere Einflüsse unkontrolliert in Form einer Disruption oder Diskontinuität [11] des aktuellen Zustands eindringen.

Das Eindringen der Disruption wird in [2, 11] als Angriff gesehen, was für ein IT-Risikomanagement berechtigt wäre. Diskontinuität könnten dagegen als Ausstoß gesehen werden.

Sie Zeigen sich als Impuls, die Bewegungen anstoßen. Weitergeführt werden in (Abb. 4.1) die Antriebe der Innovation beschrieben. Diese steigern die Innovationskraft und wirken gegen die Widerstände, genannt Einschränkungen. Die Antriebe sind hierbei in Form von Anforderungen nur ein Soll-Zustand der Organisation. Sie ermitteln sich aus der Last der Legacy-IT.

URSPRUNG DER IDEE Der Begriff der antreibenden und einschränkenden Faktoren generierte in Zusammenhang mit dem Transformationsziel der heutigen IT-Architekturen hin zu einer Cloud-Architektur die Idee von einem Heißluftballon als Metapher für die sich transformierende Organisation.

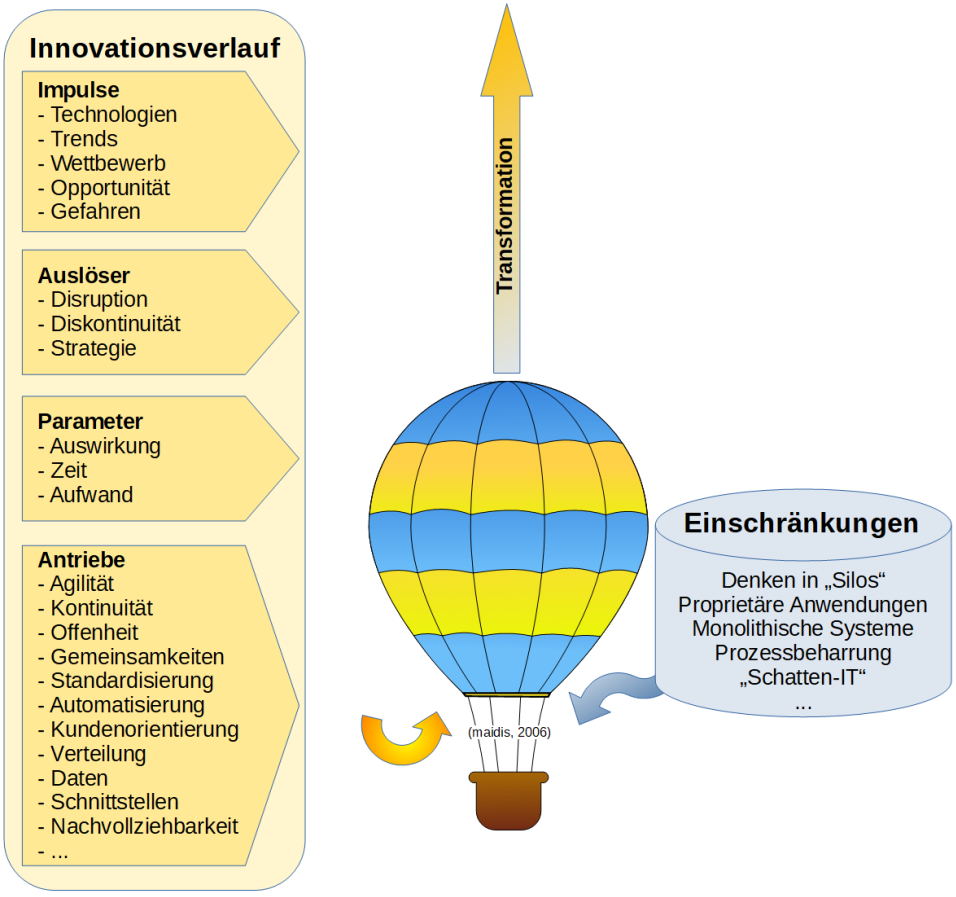


Abbildung 4.1: Innovationsverlauf und Regelung einer Transformation (Quelle: eigene Darstellung, Heißluftballon von maidis [36])

AUSWIRKUNG AUF DIE TRANSFORMATION Wird das ganze mit einem entsprechenden physikalischen Zusammenhang betrachtet scheint es sofort ersichtlich, dass geplante Transformationen mit Gegengewichte belastet werden. Der Arbeitsaufwand für die Transformation steigt exponentiell, je schwergewichtiger die Organisation ist. Veränderungsprozesse oder Transformationen können nicht mit linearen Metriken geplant werden. Daher entsteht auch das Problem von exponierenden Budgets in Anpassungsprojekten.

Um eine Transformation zu planen reichen Aufwände und Zeitrahmen nicht aus. Die Transformation bedingt eine Innovationskraft, die sich aus der Wechselwirkung zwischen vorhandenem Antrieb und Einschränkungen ergibt. Sind genügend Antriebe vorhanden und wenig Last kann auch Traktion entstehen.

Die Maßnahmen der Legacy-IT in Banken richten sich nach den falschen Metriken und Zielen. Ziel ist es nicht eine Arbeit hinter sich zu bringen. Technologiesprünge werden frequentierter und Strecke immer länger. Die aktuellen Maßnahmen richten sich nur nach den Einschränkungen. Diese sind erstmal nicht relevant, da sie sich nicht selbst überwinden können. Für die Innovationskraft spielt der materielle Zusammenhang keine Rolle.

BILDLICH WEITERGEFÜHRT Die Analogie eines Heißluftballon für die Regelung der Transformation kann folgendermaßen fortgeführt werden:

Die Kräfte im Ballon sind dynamisch und bewegen mit genug Antrieb ein statisches Abbild der Organisation, welcher im Korb des Heißluftballon liegt und einen Zustand der Organisation beschreibt. Der Korb ist in sich geschlossen und beinhaltet den Ablauf und Aufbau des Kerngeschäfts. Dieser trägt zu ihrer Unterstützung viele weitere Funktionen mit, die jeweils die Bewegung der Organisation durch ihre Last exponentiell einschränken. Die Bewegung kann auf zwei Arten beschleunigt werden. Der Antrieb wird erhöht oder ein Teil der Einschränkungen abgeworfen. Sie kann nicht gesteuert werden, wodurch externe Einflüsse sie stark beeinflussen können. Das Objekt gleitet hierbei im Strom einfach mit.

LEHREN AUS DOCKER Ein statisches Abbild mit Prozessbeschreibungen und Ausführung als eine in sich geschlossene Einheit ist aus der Virtualisierung von Systemen bekannt. Mit zunehmender Automatisierung der Geschäftsprozesse finden sich auch zunehmende Gemeinsamkeiten zwischen Organisationen und IT-Systemen.

Best-Practices zur Beschreibung von Images mit Docker könnten als Inspiration für Organisationen dienen. Besonders das Minimalisieren von Images durch schlanke und funktionale Ebenen sowie Kubernetes zur flexiblen Skalierung und Orchestrierung von ganzen System könnten als Idee für eine flexible und skalierbare Orchestrierung von Veränderungen für die Organisation dienen, um ihre Transformationsfähigkeit zu erhöhen. Der schmerzvolle

Rollout bei wesentlichen Veränderungen in großen Organisationen könnte in eine vollautomatisierte Orchestrierung umgewandelt werden. Agile Prinzipien in der Kultur wären das Docker und ein neues effektives Change-Management das Kubernetes des Managements.

Nach einer solchen zukünftigen Entwicklung wäre die Idee von *Development as a Service* 3.2 mit entsprechenden eingespielten Abläufen durchaus realistisch. Andere disruptive Entwicklungen sind auch nicht ausgeschlossen. Umso wichtiger ist es einen Weg zu finden die Digitale Transformation zu beherrschen.

4.2 FRAGE AN DAS MANAGEMENT: CHANGE ODER RUN?

TRANSFORMATIONSFÄHIGKEIT ALS PRINZIP Abb. 4.1 stellt für den Umgang mit wesentlichen Veränderungen ein erstes Prototyp dar. Die Abbildung beschreibt möglicherweise einen Zustand vor einer bevorstehenden Transformation.

Daher dient sie als Grundlage, um Anforderungen für einen innovativen Soll-Zustand zu simulieren. Ein innovativer Soll-Zustand der Organisation würde eine höchstmögliche Transformationsfähigkeit bedeuten.

Gleichzeitig zeigt sie, dass mit Anforderungen und Kenntnissen über Einschränkungen eine bevorstehende Transformation nur geregelt werden kann und die Richtung nicht absehbar ist. Gas geben ohne zu lenken ist für FinTechs und Start-ups kein großes Problem, da aufgrund ihrer kleinen Größe die Auswirkungen von Veränderungen einfacher analysiert werden können.

Für die größeren, etablierten Institute gilt ein erhöhter Schutzbedarf [3] und dadurch zusätzliche Maßnahmen für die Kontrolle von operationellen Risiken [13, 14]. Diese müssen gezwungenermaßen sich neu ausrichten Bussmann u. a. [5] und Gupta [20], indem sie ihre hohe Masse in Bewegung versetzen. Analog impliziert es eine Unkontrollierbarkeit von möglichen Problemen und Risiken.

Koch, Ahlemann und Urbach [25, S. 11.4.1] fordern für ein Management der Transformation:

> "Erstens ist zu akzeptieren, dass der Transformationsprozess nicht im Kontext eines einmaligen begrenzten Projektes vollzogen werden kann. Vielmehr bedarf es einer langfristigen Initiative, damit die notwendigen Fähigkeiten entwickelt werden können. Zweitens darf die Transformation nicht als rein technologische Initiative interpretiert werden."

Die Entwicklung von skalierbaren und flexiblen Systemen für die SEU von Banken setzt wesentliche Veränderungen in der IT-Architektur, Prozesse und Organisation eines Instituts voraus. Erfolgen diese nicht werden umfangreiche Anpassungen der verwendeter Standardsoftware und Abänderungen von gängigen Lösungen bedingt 2.3.

In folge dessen sind die Veränderungsprozesse selbst zu optimieren und die

Einschränkungen hierfür zu eliminieren. Kontinuierliche Veränderungen setzen kontinuierliche Prozesse voraus (Abb. 2.1). Standards in der Softwareentwicklung sind agil und verändern sich ständig. Eine flexible IT-Architektur ist erforderlich, um sich diesen Veränderungen anzupassen [5].

ERRICHTUNG VON ANLAGEN Aus Sicht der Entwicklung und Beteiligten an Change-Prozessen könnte bezüglich der Einschränkungen auf die Regulatorik oder Betrieb verwiesen werden.

Dagegen spricht aus Sicht des Betriebs, dass Change-Prozesse zuerst sich selbst zu optimieren haben und dadurch die richtigen Prioritäten setzen, bevor sie den ohnehin konservativen Betrieb optimieren. Die Skalierbarkeit und Flexibilität der IT-Architektur sollte nicht aufgrund ihrer Symptome gefordert werden, sondern als eine Anlage[1], um weitere Innovationen zu fördern.

Möglich wäre eine Strategie, in der Entwickler und Geschäftsleitung eng zusammenarbeiten. Beim Durchleuchten der Einschränkungen aus verschiedenen Perspektiven fällt auf, dass die konservative Haltung des IT-Betriebs und der Regulatorik eine valide Berechtigung, gerade im Finanzwesen hat. Die Anforderungen selbst sind trotzdem flexibel gestaltet und bieten einen Ermessensspielraum.

GESTALTUNGSMETHODEN FÜR IT-MANAGEMENT Neben externe Auslöser von Innovation, die Unternehmen aufgrund von Disruption und Diskontinuität zu Veränderungen zwingen [11, 20] existiert die IT-Strategie als interner Auslöser [1, 14], dessen Verantwortung in erster Linie der Geschäftsleitung unterliegt [14]. Daher sollten zu erst die Veränderungsprozesse optimiert werden. Eine klare IT-Strategie optimiert sich selbst bevor Forderungen gestellt werden. Dies wird durch Gestaltungsmethoden wie *Design-Thinking* hervorgehoben, indem sie die Quellen zu fremden Problemen zuerst auf sich selbst richten [34]. Gestaltungsmethoden wie Design-Thinking werden gerade für Innovationsprozesse und für das ITM empfohlen [1, 25]. Zudem sollte die in Finanzdienstleistungsaufsicht [13, AT 8.2] geforderte Analyse und Beteiligung bei wesentlichen Veränderungen schon in den Gestaltungsprozessen stattfinden [9]. Eine Mitgestaltung nach agilen Methoden (Abb. 2.1) ist auch im IT- und Risikomanagement erforderlich, um effiziente Abläufe und Maßnahmen zu schaffen.

VERMEIDEN VON TRENDFORDERUNGEN Unklare Forderungen wie Agilität sollten vermieden werden. Agile Methoden wiederum sind eine klare Forderung. Agilität ist auch keine Kulturförderung, da sie eine Leistungserwartung für künftige impulsive Bewegungen stellt oder eine Impulserwartung in einer Umgebung voller Widerstände. Nachhaltigere kulturelle Ansätze wären in (Kap. 4.4.4)

1 vgl. “capabilities” in [25]

4.3 FORDERUNG EINES EFFEKTIVEN VERÄNDERUNGSMANAGEMENTS

Für die Vorhersehbarkeit von Auswirkungen ist ein kontrollierter Umgang mit Veränderungsprozessen erfordert. Um wesentliche Veränderungen kontinuierlich zu ermöglichen muss die Analyse ihrer Auswirkungen [13] ebenfalls effektiv sein. Die Vorhersehbarkeit der Auswirkungen von Veränderung sollte nicht nur aufgrund der Regulatorik erfüllt werden. Sie ist als eine Voraussetzung für eine klare Sicht und Zielorientierung sein und die Transformation anstreben.

Ein effektives Veränderungsmanagement hat das Ziel einen Zustand der höchstmöglichen Transformationsfähigkeit zu erreichen. Hierfür ist ein grundlegend innovatives Paradigma vonnöten.

INNOVATIVER SOLL-ZUSTAND DER KULTUR Nach Bussmann u. a. [5, S.30] ist eine kontinuierliche Justierung für den Unternehmenserfolg wichtig. Als eine der Schwierigkeiten nennt er ein effektives Management der Veränderungsprozesse.

Diese Schwierigkeit hat sich in Banken bis heute noch bewährt. Daher wird zum Prototyp aus Abb. 4.1 ein erweitertes Modell bedingt. Für den Umgang mit kontinuierlichen Veränderungen erfordert es effiziente kontinuierliche Veränderungsprozesse. Damit sollen bevorstehende und unvermeidbare Transformationen der Organisation kontrollierbar stattfinden. Die Frage zu einem Umgang mit kontinuierlichen Veränderungen betrifft das Veränderungsmanagement.

Koch, Ahlemann und Urbach [25, S. 184f] ordnen das Veränderungsmanagement zu den wichtigsten Anforderungen an ein zukünftiges Paradigma der IT-Organisation. Das bisherige Modell *Plan-Build-Run* ersetzen sie mit einem neuen Paradigma, dem *Innovate-Design-Transform*.

Daher wird klar warum das Modell aus Abb. 4.1 nicht zufriedenstellend genug ist. Sie identifiziert anhand der Antreiber und Einschränkungen viele Anforderungen aus [25, Tab. 11.1] aus den Bereichen Innovations- und Gestaltungsfähigkeit. Für ein Paradigmenwechsel großer Organisationen spielt die Transformationsfähigkeit die entschiedenste Rolle.

4.4 ÖKOLOGISCHES VERÄNDERUNGSMODELL DER IT

In Anlehnung an *DevOps* [1] und das *innovate-design-transform* Paradigma von Koch, Ahlemann und Urbach [25] wurde ein kontinuierliches Modell für die gezielte Steuerung von Innovationsabläufen [18] ausgearbeitet.

Während das DevOps Lebenszyklus [1] sich auf einen kontinuierlichen Prozess für die Optimierung eines IT-Produkts oder IT-Service bezieht, soll diese Abbildung 4.1 sich auf einen kontinuierlichen Prozess für die Optimierung der IT-Organisation beziehen, indem die Veränderungsprozesse selbst in einem kontinuierlichen Prozess ihr Ergebnis ständig verbessern können. Dieser ist die Steigerung der Transformationsfähigkeit der Organisation

und die Anhäufung von antreibenden Anlagen.

4.4.1 *Von Innovationsverlaufen zum Innovationsmotor*

Der für Abb. 4.1 ausgearbeitete Innovationsverlauf stellt einen ersten Zusammenhang für die schwer zu überblickenden und zu beherrschenden und Abläufe, wodurch Innovationsprozesse in Gang gesetzt werden. Die Impulse, welche die Ausgaben der schwierig zu erkennenden fremden Abläufe sind konnten dadurch grob klassifiziert werden. Das Innovationsmanagement hat zu erkennen, dass der Innovationsprozess extern bereits gestartet werden kann und die Ausgaben der externen Abläufe die Organisation durchdringen. Für die Beherrschbarkeit der unvermeidbaren Innovationseinflüsse hat das Innovationsmanagement sich zur Aufgabe machen die externen Impulse wahrzunehmen und die Abläufe abzufangen und gegebenenfalls zu übernehmen. Dabei ist es von besonderer Bedeutung das Risikomanagement in diesen Überwachungsprozess mit gestalterischen Tätigkeiten miteinzubeziehen. Ein sich aufbrauender Veränderungsprozess macht sich mit kleinen Impulsen bemerktbar, bevor es disruptiv in die Athmosphäre der Organisation eindringt. Die Gefahr der Disruption kann auch mitigiert werden, indem die Innovation gekapert und angenommen wird.

Das Risikomanagement hat eine parallele Einheit zu errichten, die sich nicht als Ordnungsbehörde mit Strafzetteln in Form von Moniten sieht, sondern gegen Disruption als taktische Einheit aggressiv in Form von Mitgestaltung und Initiative vorgeht. Gleichzeitig aber als eine konsequente Einheit aggressiv Diskontinuitäten ausstößt, da hieraus die größere Gefahren entstehen können [11].

Die BaFin und EBA sollten die Gefahr der Diskontinuität ebenfalls erkennen und eine klare Anforderung an die IT-Strategie und Risikomanagement zur Erhaltung der Transformationsfähigkeit des Unternehmens mit innovationsorientierten Einheiten stellen. Daher hat die Gestaltung des Innovationsmanagements Implikationen auf das Risikomanagement. Die Aufsicht sieht die entsprechenden Begriffe in [13] und [14] nicht vor. Sie fordert in [14] jedoch eine IT-Strategie, dass einige Ausgangsvariablen[2] von Innovationsabläufen beinhaltet. Disruptive Technologien greifen den Markt an [11] und somit auch das Geschäftsmodell der etablierten Institute (3.2) und sind daher ein operationelles Risiko. Diese Tatsache ist seit längerem bekannt [18] und wird nicht berücksichtigt.

2 In diesem Kapitel als Faktoren für Veränderung grob eingeführt

4.4.2 *Forderungen zu MaRisk*

Aufgrund der Risikosituation in Zeiten der Digitalen Transformation wird zu den MaRisk [13] im Rahmen der Ermessensfreiheit der Institute folgende Maßnahmen gefordert:

1. Die IT-Strategie hat die Transformationsfähigkeit des Instituts sicherzustellen und Innovationsprozesse kontinuierlich zu verbessern
2. Das Risikomanagement hat Impulse aus Innovationsverläufen zu überwachen und gegen Disruptionen ein Notfallkonzept für Transformationsprozesse auszuarbeiten. Insbesondere hat sie die Aufgabe eine Diskontinuität zu erkennen und eine Transformation aktiv vorzubereiten. [11]

4.4.3 *Ökologie des IT-Betriebs*

Das *innovate-design-transform* Paradigma von Koch, Ahlemann und Urbach [25] wurde um den Punkt Accumulate erweitert, um die Errichtung von antreibenden Anlagen [18, 25] miteinzubeziehen. Gerade dieser Schritt erzeugt ein Kontinuitätsprinzip, dass eine Steigerung der Transformationsfähigkeit vorsieht. Accumulate steht hierbei für Wertschöpfung. Sie kann auch für das Sammeln von Daten und Erkenntnissen stehen. Dieser Schritt ist ein Beitrag für einen ständigen Verbesserungsanspruch der IT-Architektur. Der Betrieb darf nicht als eine statische Einheit gesehen werden. Der Betrieb muss ständig in Traktion im Rahmen einer kontinuierlichen Transformation bleiben. Die Bereiche Entwicklung und Produktion wurden hierbei umbenannt. Sprint steht für die Agile Entwicklung. Die Produktion wurde in Plant umbenannt, um eine Nachhaltigkeit der IT-Architektur vorauszusetzen. Produktion bezeichnet die industrialisierte Legacy-IT, die abgeschafft werden soll durch ein ökologisches System, dass ein Verbesserungsanspruch an sich selbst stellt und den Eintritt von Schadstoffen vermeidet. Schadstoffe sind hierbei operationelle Risiken, die besonders die Materialisierung einer der Legacy-IT durch Anhäufung von Diskontinuitäten darstellt.

4.4.4 *Prinzipien der Veränderung*

1. **Offenheit** [4] ist vermutlich das höhste Gut im Rahmen von Innovationsverläufen. Diese basieren auf *Sharing*, daher braucht es eine Empfänglichkeit, damit ein Übertrag der Innovation stattfindet und ein Innovationsprozess überhaupt entstehen kann.
2. **Zusammenarbeit** bezeichnet hierbei das aktive *Sharing* und ist eine Voraussetzung für die Initiative, um gemeinsam Innovationsprozesse auszulösen, zu steuern und eine nachhaltige IT-Strategie zu entwickeln
3. **Nachvollziehbarkeit** Bevor an Nachvollziehbarkeit der Systeme gedacht wird, sollte an die Nachvollziehbarkeit der Organisation im Rah-

men einer freien und innovationsfördernden Kultur gestrebt werden. Während *Sharing* eine Innovation anstoßt reicht womöglich *Empathize* schon für eine Transformation aus

4.4.5 *Anforderungen für Transformationsprozesse*

1. **Trennbarkeit**, "Teile und herrsche", leitet sich aus der Maßnahme der Modularisierung und Entkernung von Systemen Bussmann u. a. [5] ab und ist eine Anforderung aufgrund von Abhängigkeit als das Hauptsymptom der Legacy-IT. Ihre Effizienz kann mit dem Rahmen der Microservices begründet werden.

2. **Effizienzsteigerung** Die Effizienzsteigerung wirkt sich auf die verwendeten Ressourcen und Zeitaufwand der Prozesse, Systeme oder Anwendungen aus. Dazu zählen auch Kosten- und Personalaufwand. Es stellt sich die Frage in welchen Bereichen eine Effizienzsteigerung am sinnvollsten ist.

   Die Effizienzsteigerung wird vor allem im Bereich der Veränderungsprozesse "Change" bezüglich ihres Zeitaufwands erforderlich. Im Bereich der betrieblichen Prozesse "Run" ist sie primär für den Aufwand der Ressourcen erforderlich.

   Eine Effizienzsteigerung im Betrieb ermöglicht mehr Mittel für die "Change" Prozesse [30]. Dagegen steht das Prinzip, dass Ressourcen im Gegensatz zu Zeit finanziert und eingekauft werden können. Bevor an Opportunitäten in Form einer strategischen Weiterentwicklung der Produkte [30] gedacht wird, sollte die Weiterentwicklung der hierfür vorausgesetzten Veränderungsprozesse berücksichtigt werden. Die Fähigkeit für eine kontinuierliche Anpassung [5, 18] der Produkte wird mit aktuellen Methoden, wie in Abb. 2.1 dargestellt, bereits vorausgesetzt [1].

3. **Standardisierung** Dieser Punkt umfasst Anwendungen, Systeme und Prozesse. Die Verwendung von gängigen sowie standardisierten Lösungen bringt eine Reihe von Vorteilen. Sie erhöht allen voran die Nachvollziehbarkeit der Systeme und sorgt für eine Effizienzsteigerung von Prozessen [1, 5, 32].

   Beteiligte, insbesondere die Prüfer der Bankaufsicht und die interne Revision, können so mit einem Grundwissen über die IT die meisten Systeme auf Anhieb verstehen. Die Implementierung von *Sicherheitsstandards [8, 22]* ist hierbei besonders wichtig und bietet eine Grundlage für die Einhaltung der regulatorischen Vorgaben [13, 14].

   Der Einsatz von Standardsoftware führt zu einem effizienteren Ablauf in ihrer Wartung und Weiterentwicklung [5]. Die Komplexität von operativen Systemen schränkt die Einführung von Standardsoftware jedoch ein [5, S.27]. Das Prinzip der *Standardisierung* in Verbindung mit *Offenheit* ergibt die Verwendung von "Open-Source" Software, die eine

höhere Effizienzsteigerung aufgrund der Wiederverwendbarkeit der Komponenten bezweckt [4, 20].

Die Einführung von Standardsoftware hat ebenfalls Auswirkungen auf eingespielte Prozesse und die daraus folgenden Einzelentscheidungen beeinflussen in ihrer Summe die Entscheidung für oder gegen den Standard [28, Tab.1].

5 BEWERTUNG UND ZUSAMMENFASSUNG

5.1 METHODIK UND ERGEBNISSE

Design-Thinking (Kap. 2.2.1) hat als Methode die vorliegende Arbeit erheblich geprägt. Anfänglich wurde sie beiläufig berücksichtigt und gewann für die Erforschung der Ursachen erheblich an Bedeutung. Der Prozess ist hierbei kein fester Rahmen, sodass Elemente aus "Requirement-Engineering" und ein agiler Vorgang eine Rolle gespielt haben. Vielleicht ist das gerade die Stärke, weil der Prozess Offenheit vorsieht und individualisiert werden kann. Wichtig ist der Anfang des Prozesses mit *empathize*.

Dieses Modell ist auch von Agilität und Kontinuität geprägt, worin sich das Ergebnis ständig verbessert. Mittendrinn können auch Iterationen erfolgen. Besonders effektiv ist es, wenn dabei aus dem aktuellen Zustand heraus ein Perspektivwechsel gezielt hin zur Problemquelle stattfindet. Am Ende kann mit einer erheblich größeren Datenmenge eine Synthese aus allen Perspektiven generiert werden als Deduktion. Dabei war die Ausgangssituation von einem sehr eingeschränkten Problembild geprägt. Eine Immersion (Kap. 2.2.1) hat das Horizont innovativ und effektiv erweitert. Der Design-Thinking Prozess könnte als ein "Quick-Search" für Innovation gesehen werden. Daneben sind andere Gestaltungsmethoden ebenfalls zu betrachtet.

Sie hinterlässt auch viele beiläufige Ergebnisse, die zur Immersion beitragen[1]. Dazu Besteht die Gefahr zu divergieren, da sie mit einer erheblichen Geschwindigkeit induktiv vorgeht und auf kontinuierliche Verbesserung der Prototypen beruht. Dadurch entstehen viele Artefakte. Für die Qualitätssicherung der Lösung verlangt es einen effizienten Vorgang der Integration der Komponenten und Implementierung einer Lösung mit entsprechenden Methoden. Für eine erste Lösungsrichtung bieten ihre Syntheseschritte durch ständige Iterationen eine schnelle Deduktion. Ergebnisse können entsprechend von einer Innovationsphase in die Entwicklungsphase übergehen, in welchem die Qualitätssicherung in einer separaten Umgebung sichergestellt wird. Agilität ist daher immer mit einer Nutzen-Risiko Analyse verbunden.

5.1.1 *Aus CI/CD zu einer nachhaltigen IT*

Die Anfangssituation aus der SEU in Kap. 2.3 wurde bis zu einer unzumutbaren Hindernissschwelle bearbeitet und erste Lösungsansätze mit gängigen Standards beschrieben. Die vorgeschlagenen Ansätze lösen jedoch nicht den Ursprung des Problems, da sie umfassende Auswirkungen und Anpas-

1 Anhang: Docker Images für Jenkins Agenten

sungen nach sich ziehen, die eine Transformation der ganzen Organisation fordern.

Für die Ursache des Problems fand ein Wechsel der Perspektive in Kap. 3.1 zu einem Wettbewerber statt, der eine Transformation bereits durchgemacht hat. Diese Transformation ist als Fallbeispiel von Gupta [20] empirisch beschrieben und enthält wiederum verschiedene Stellungnahmen aus der Managementebene.

In den Schritten *empathize*, *define* und *ideate* wird die Ursache des Problems analysiert möglichst ohne vorherige Annahme zur Quelle des Problems. Wichtig ist, dass hierbei wirklich die Perspektive gewechselt wird und eine Immersion in die Umgebung des Kunden stattfindet. Daher kann diese Gruppe statt Analyse [34] auch Immersion genannt werden.

Hierbei wurde in dieser Arbeit die Erfolge der betrachteten Perspektive hervorgehoben und untersucht. Anschließend wurde eine Steigerung dieses Erfolges gesucht.

Hierfür sind Schritte in *Synthese* eine Datenerhebung [34, S. 61] durch Gestaltung von Zukunftszuständen, die nach Alt, Auth und Kögler [1, S. 14] für eine *Vorwegnahme künftiger Innovationen* von besonderer Bedeutung ist.

Die Opportunitäten durch Goldman Sachs Plattform [20] und der Hintergrund von disruptiven Technologien [11] wurde zum Anlass genommen Ideen zu formulieren. In Kap. 3.1 wurden weitere Zukunftszustände gestaltet. Wichtig ist hierbei, dass ungezwungen so viele Daten wie möglich generiert und erhoben werden. Mit dem richtigen Verständnis aus den vorigen immersiven Schritten kann eine Deduktion nach den Anforderungen des Kunden stattfinden.

Die vorgeschlagenen Ideen werden daher mit einer anderen Perspektive weiter gefiltert. Im Rahmen des Finanzwesens standen den disruptiven Technologien und FinTechs auf dem ersten Blick die Regulatorik entgegen.

Die erhobenen Daten waren anfänglich neue Technologien zur Lösung des Problems und Einschränkungen im Rahmen der IT-Architektur und Regulatorik. Die regulatorischen Forderungen an sich sind keine Einschränkung gegenüber Innovationsverläufen. Die Einschränkungen werden durch die Symptome der diskontinuierlichen Legacy-IT hervorgerufen, da Transformationen nicht durch immaterielle Zustände beschränkt werden können.

Das Problem hierbei ist, dass ab einer gewissen Größe des Unternehmens sie sich gegenseitig voraussetzen. Die Optimierung der Überwachungs- und Steuerungsprozesse erfordert eine agile Architektur, die widerum durch sich selbst eingeschränkt ist. Ganswindt [18] benennt das Ganze als Henne-Ei Problem, dass an vielen Stellen in unterschiedlichen Formen auftaucht.

Daher wurde früh erkannt, dass die Ursache in der Transformationsfähigkeit der Organisation liegen könnte. Aus einer gemeinsamen Betrachtung der Sicht der SEU 2.3 und Goldman Sachs [20] spielten hierfür neben regulatorischen Anforderungen als Einschränkung auch die Kultur als Antrieb eine wichtige Rolle.

Der Perspektivwechsel mit der *Design-Thinking* Methode 2.2.1 durch eine Iterationen in den anfänglichen Empathize Zustand ermöglichte die Sicht

der Regulatorik auf Technologien und Veränderungen zu betrachten und ist daher für die Generierung von weiteren Ergebnissen wichtig, bis die Lösung der Problemursache gefunden wurde.

Die Aufsicht wäre sicherlich nicht daran interessiert noch mehr Komplexität durch zu präzise Anforderungen zu manifestieren. Die Ermessensfreiheit der Institute setzte Gestaltungsfähigkeit voraus.

Hierfür wurden Prototypen nach *Design-Thinking* gestaltet, die unweigerlich zeigen, dass antreibende Anlagen zur Innovationsförderung wichtig sind. Für die Priorisierung fehlte dabei ein klares Steuerungsprozess, daher wurde das Modell mit den dargestellten Ergebnissen kontinuierlich verbessert.

PROTOTYP: REGELUNG DER TRANSFORMATION Für Prozesse wird im Rahmen des Innovationsdrucks vom IT-Management gerne auf Frameworks zurückgegriffen, wie zum Beispiel ITIL (Kap. 2.2.1), worin Services sich bezüglich ihres Innovationsbeitrags messen lassen sollen [1]. Im Rahmen dieser Arbeit wurden die Frameworks wie ITIL oder auch das offene Framework IT4IT wahrgenommen, jedoch bewusst nicht weiter untersucht. Im Rückblick kann festgestellt werden, dass viel mehr ein klares Bild vonnöten ist, was Frameworks wie ITIL bezwecken sollen, vor allem für das ITM.

Es kann festgestellt werden, dass die Darstellung in (Abb. 4.1) einen Beitrag für das Verständnis für Innovationskraft eines IT-Produktes leistet, um wiederum für eine IT-Strategie mit klaren Metriken zu priorisieren. Daraus wird klar, dass antreibende Technologien eine Anlage für die Generierung weiterer Innovationen sind. Die antreibenden Faktoren, die aus dem Ziel der Maximierung der Transformationsbewegung generiert wurden stellen möglicherweise eine Anforderung für Antriebe oder antreibenden Anlagen dar. Hieraus könnten weitere Ansätze generiert werden, um gezielt die Effektivität einer Innovation zu messen. Diese wäre wichtig für eine Priorisierung seitens ITM.

Das schwierige ist jedoch hierbei die Transformation zu steuern und somit die Bewegungen zu kontrollieren. Das Finanzwesen erfordert eine Organisation mit absehbaren Manövern. Starke Bewegungen in Form von disruptiven Sequenzen sind riskant. Daher wird eine Transformation in Zusammenhang mit Disruptivität als Gefahr gesehen [2].

Aus diesem Hintergrund wurde ein weiteres Modell erzeugt mit Kontinuitätskonzept und einer Anhäufung von antreibenden Anlagen, wodurch einerseits disruptive Sequenzen linearisiert werden und gleichzeitig ein kontinuierlicher Antrieb entsteht, so die Theorie.

5.2 ÖKOLOGISCHES VERÄNDERUNGSMODELL DER IT

Die Darstellung in (Abb. 5.1) ist eine Vorwegnahme eines künftigen Zukunftzustandes nach [1], die aus einer Orientierung nach *Design-Thinking* Schritten entstanden ist und ständigen Iterationen ausgesetzt war, die den Rahmen dieser Arbeit sprengen würden. Nach der Identifizierung von möglichen Antrieben der Transformation, wurde [18] genauer untersucht. Nach

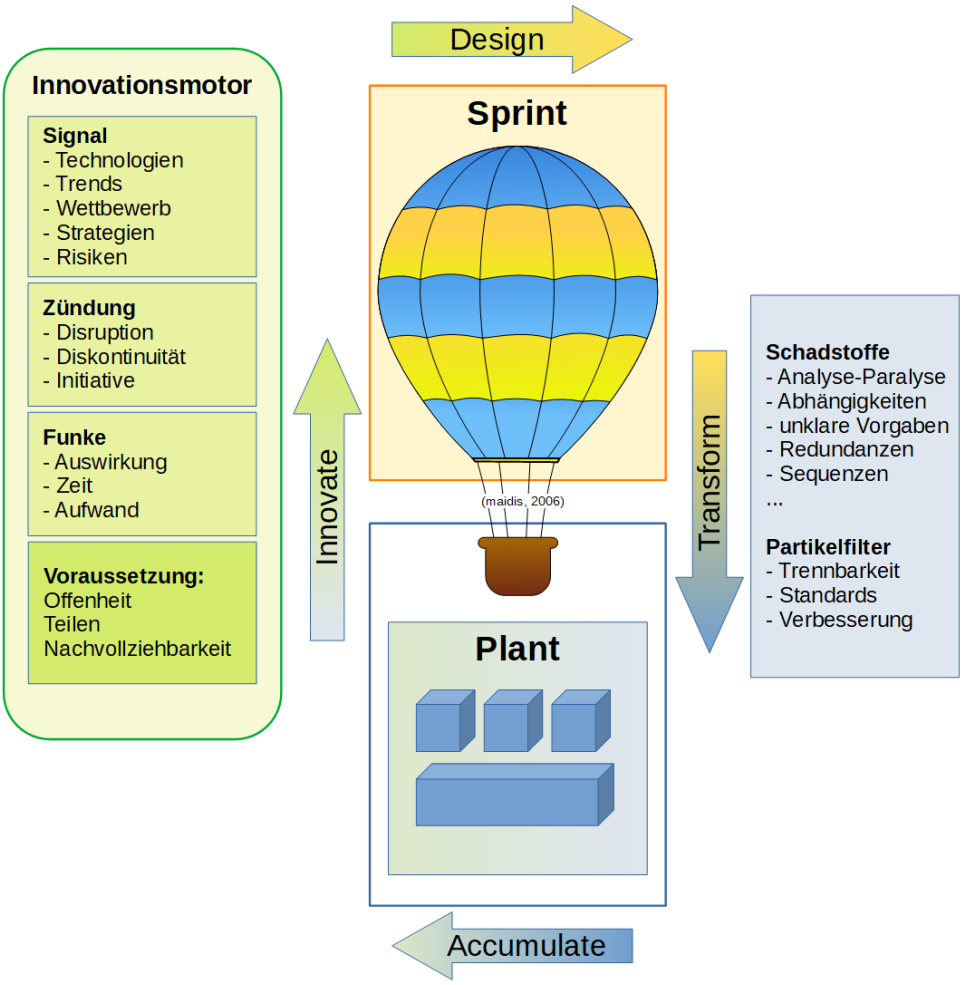


Abbildung 5.1: Ökologisches Veränderungsmodell der IT mit Anhäufung von antreibenden Anlagen (Quelle: eigene Darstellung, Heißluftballon von maidis [36])

vgl. (Kap. 4.4)

dem Fund des *Treibstoffs*[2] musste ein Motor her. Die Regelung der Transformation durch manuelle, sequenzielle Abläufe erzeugt durch ihre Ineffizienz viele Schadstoffe die sich als einschränkende Faktoren materialisieren. Die Anhäufung dieser Faktoren ist das Ergebnis in Form einer Legacy- und Schatten-IT.

Für eine nachhaltige Organisation gehört es sich ihre Schadstoffe mit dem Einpflanzen von antreibenden Anlagen zu reinigen.

5.3 ZUSAMMENFASSUNG UND AUSBLICK

Es ergibt sich ein Gesamtbild, dass die aufgeführten Faktoren der Veränderungsprozesse keine direkte Ablauf- oder Handlungsanweisungen darstellen. Daher kann das Problem der Priorisierung in der IT-Strategie stammen.

Diese leiten sich aus dem gegenwärtigen Zustand ab und beschreiben *Einflüsse* gegen Veränderungsprozesse und haben wesentliche Auswirkungen auf ihren Ablauf. Insbesondere die antreibenden Faktoren für Veränderungsprozesse können als ihre funktionalen Anforderungen gesehen werden.

Das IT-Risikomanagement sieht mit ihren Maßnahmen jedoch in Disruptionen die Gefahr. Fernández, Valle und Pérez-Bustamante [11] unterscheiden dabei Disruption und Diskontinuität. Etablierte Organisationen sind gegenüber Disruption Handlungsunfähig. Dabei geht die eigentliche Gefahr aus der Manifestierung von Diskontinuitäten aus. Diskontinuierliche Technologien weisen einen erheblich höheren Widerstand gegenüber der dominanten Technologie auf [11].

Die Diskontinuität ihrer Legacy-IT hat das Finanzwesen als Geisel genommen und das IT-Risikomanagement zu ihrem Helfer gemacht.

Kontinuierliche Ansätze wie *DevOps* und *Design-Thinking* helfen dabei die Veränderungsprozesse selbst agil und kontinuierlich zu gestalten und zu verstehen. Daher hat das IT-Risikomanagement sich nach Innovationsprozessen zu richten, um sich entgegen der Legacy-IT zu bewegen.

Darüber hinaus wird deutlich, dass die Informatik weitreichende Kompetenzen verfügt aus dem Zusammenhang heraus Probleme auch aus anderen Bereichen zu lösen. Beginnend von der Ausgangssituation eines vorwiegend technischen Problems haben innovationsorientiere Ansätze weitreichende Implikationen auf ihre nicht-technische Umgebung. Die Abkürzung IT sollte im Sinne einer Organisation künftig nicht für Informationstechnologie stehen sondern für "Innovation und Transformation", wodurch es Bezug auf die Eingangs- und Ausgangsströme des Paradigmas "Innovate-Design-Transform" nimmt. Dazwischen liegt unser Ergebnis, das Akkumulieren als Wertschöpfung.

Kreative Methoden sind durch die ad-hoc Generierung von Prototypen oder Beschreibungen wichtige Mittel für die Gestaltung von immateriellen Zuständen und daher für Innovationsprozesse geeignet. Die Synthese ihrer Ergebnisse ergibt sich aus der Kohärenz zwischen gegenwärtigem Zustand,

2 Begriff aus [18]

zukünftigen Risiken sowie Opportunitäten der verschiedenen Perspektiven.

Ganswindt [18] beschreibt:

> " Ein Unternehmen der Wissensgesellschaft muss sich deshalb so organisieren, dass die Ressource Wissen zu einer steten Quelle eines Ideenstroms wird – egal, von welchem Ort der Welt aus der Mitarbeiter tätig ist. Es ist ein Muss, bei den Mitarbeitern Begeisterung für neue Ideen zu wecken, das Vertrauen in die eigene Kreativität zu fördern und ihnen dann Raum zu lassen, das Neue auch umzusetzen und ihr Wissen zu teilen "

Sie bieten keine direkten Anweisungen zur Implementierung einer Lösung, sind aber für ihre Strategie besonders wertvoll. Die steigende Kontinuität, Geschwindigkeit und Auswirkung von Veränderungen verlangen das Treffen von schnellen Entscheidungen, wofür Agilität und Kreativität in der Strategie bedingt wird. Geschieht dies nicht kann auch kein effektiver Veränderungsprozess erzeugt werden, woraus Ziele der Strategie erfolgreich umgesetzt werden. Mit ineffizienten Innovationsprozessen entsteht für die Transformation eine Paralyse durch Analyse, mit darauf folgender Handlungsunfähigkeit.

Für eine effektive Transformation bedingt es reibungslose Veränderungsprozesse, welche die Transformationsfähigkeit der Umgebung messen können. Danach sollte die Entwicklung von Gegenmaßnahmen als antreibende Anlagen für die Transformationsfähigkeit priorisiert werden. Als einer der einflussreichsten Maßnahmen fallen hierunter Plattformen für die IT (Kap. 3.1) und Anwendungen für die Unterstützung von Steuerungs- und Überwachungsprozessen [5].

Das Kontinuitätskonzept hat bereits in vielen Bereichen der Softwareentwicklung eine Grundlage für effiziente und effektive Vorgehen geschaffen und sich dadurch als Prinzip bewährt. Beispiele wären CI für die Integration von Komponenten und CD für ein effektives und kontrolliertes Ausrollen in die Umgebung. Zusammengefasst in DevOps ergeben sie einen kontinuierlichen Lebenszyklus der Verbesserung von IT-Produkten. Das gleiche Prinzip kann auch auf Prozesse und Organisationen angewandt werden.

Die Digitale Transformation erzeugt viel Ungewissenheit mit darauf folgender Handlungsunfähigkeit. Das Problem an der Partizipation sind die Eintrittshürden für Organisationen mit Legacy-IT. Jedem Versuch einer Transformation widerstehen die gegenseitigen Abhängigkeiten der Systeme und Prozesse. Organisationen beginnen die Transformation an falscher Stelle, wodurch Projekte eingestellt werden. Daher wird auch ein kontinuierlicher Lebenszyklus der Verbesserung von Veränderungsprozessen bedingt. Ein Change-Management, dass mithilfe eines starken Innovationsmanagements Anlagen für den Antrieb der Transformationsfähigkeit identifiziert, errichtet, fördert und somit kontinuierlich den Innovationsmotor beschleunigt. Das Paradigma *innovate-design-transform* [25] könnte mit dem zusätzlichen Punkt *accumulate* ein inkrementelles Kontinuum formen und Grundla-

ge für kontinuierliche Verbesserungen der Veränderungsprozesse sein. Ein erweitertes Modell mit Metriken für Innovationsfähigkeit, Gestaltungsfähigkeit, Transformationsfähigkeit und Häufbarkeit.

AUSBLICK Die Umwandlung von sequenziellen und aufwändigen Rollouts für Changes in eine vollautomatisierte Orchestrierung mit kontinuierlich steigender Transformationsfähigkeit der Organisationen könnte womöglich neben der für die IT revolutionären Cloud-Plattformen eine viel größere Auswirkungen nach sich ziehen.

Zukünftig könnten Organisationen sich mit der Beherrschung von "Innovation und Transformation" eine unvorstellbare Schnelligkeit aufbauen. Die Weiterentwicklung des *innovate-design-transform* Paradigmas würde die Effizienz von Transformationsprozessen durch Effektivität von Innovationsprozessen mit einem entsprechend inkrementellen Antrieb steigern. Die Transformationsfähigkeit als kulturelles Prinzip könnte eine von Grund auf agile Organisation nach sich ziehen. Entsprechend würden umfassende Tätigkeiten ausgelagert oder konsolidiert werden, sodass die Kernkompetenz eines Unternehmens zukünftig aus Innovationskraft und Transformationsfähigkeit besteht. "Innovation und Transformation" verschafft der IT somit eine neue Bedeutung.

Teil II

APPENDIX

A

DOCKER IMAGES FÜR JENKINS AGENTEN

Der nachfolgende Anhang wurde während der Ausgangssituation am Anfang der Thesis vorgesehen und sollte die Ergebnisse aus der Erstellung von Docker-Images für die SEU eines Kreditinstituts technisch darstellen. Mit der Strategie das Problem von unten induktiv hochzuklettern und von oben deduktiv vorzugehen führte unvermeidlich zu einer Divergenz der ursprünglich vorgesehenen Lösungen einiger Forschungsfragen. Ursprünglich vorgesehen Lösungen implizieren schon ein "Confirmation Bias".

Die ursprüngliche Forschungsfrage [1] verlangt die Umstände der Einschränkungen:

> " Welche Einschränkungen können in einigen Banken beobachtet werden und auf welche Umstände, regulatorisch oder historisch bedingt, sind sie zurückzuführen?"

Aus der Durchleuchtung des Problems von weiteren Blickwinkeln 3 wurde klar, dass teilweise Symptome des Problems behandelt wurden. Daher macht es mehr Sinn auf Prinzipien wie die Standardisierung bekannte Lösungen zu referenzieren. Die vorgenommenen Anpassungen sind durchaus für die Migration der SEU auf eine Cloud-Plattform in einem Kreditinstitut interessant, können jedoch aufgrund der pandemiebedingten Remotearbeit nicht für den Anwendungsfall evaluiert werden und sind daher nicht zu berücksichtigen. Daher entschloss sich der Autor die Ergebnisse und vorgenommenen Themen als Anhang beizufügen, da die Beschäftigung mit diesem Thema für eine Immersion und eine daraus resultierende Idee, welcher verworfen wurde für den Lösungsweg relevant ist.

A.1 ENTWICKLUNG VON DOCKER-IMAGES FÜR JENKINS AGENTEN

Während der Entwicklung eines Kubernetes Clusters für die SEU gibt es viele Parameter, die sich ständig verändern können. Beispielsweise kann die Entscheidung für oder gegen eine Public-Cloud offen stehen oder es stehen Anpassungen offen. Daher kann nicht immer auf der größten Ebene angefangen werden.

Ein Ansatz für das Re-Design von Anwendungen ist die Entkernung und Modularisierung von Systemen [5]. Für einen gemeinsamen Nenner sollte die SEU anhand ihrer wesentlichen Funktionen aufgeteilt werden 2.3. Dadurch kann anschließend die Entwicklung ihrer Komponenten parallel durchgeführt werden.

1 vgl. Exposé für eine Bachelorarbeit zum Thema Continuous Integration in Banken - Skalierbarkeit und Flexibilität der Build-Aufträge durch Containerisierung

KLEINSTER GEMEINSAMER NENNER DER SEU Der kleinste gemeinsame Nenner der SEU sind die einzelnen Stages aus den Pipelines, die nach dem Regelprozess für die Softwareentwicklung entwickelt wurden und die einzelnen Schritte aus dem Prozess darstellen.

Der größte Skalierungsbedarf besteht im Build-Stage, da sie sehr Rechen- und Speicherintensiv ist. Diese Komponente sollte als erstes optimiert werden. Für die Build-Jobs gemäß der Pipeline ist die Plattform, in der sie sich befindet in erster Linie nicht relevant.

WORAUF ES ANKOMMT Die Build-Jobs werden in der Regel verteilt auf Agenten. Wichtig hierbei ist ihre Kommunikation zu den beteiligten Systemen. Beteiligte Systeme sind:

- Versionsverwaltung: Sie verwaltet den Quellcode, der für den Build von ihr bezogen wird
- Artefaktrepository: Sie ist die Quelle, aus der Softwareartefakte bezogen werden und sichert und verfolgt die Ergebnisartefakte
- Integrationsplattform: Sie delegiert die Build-Jobs, welche die praktische Ausführung der Pipelines definieren

Agenten sind self-contained und werden in der Regel auf hierfür konfigurierten Maschinen ausgeführt. Daher kann bereits mit der Entwicklung von Images für die Agenten begonnen werden, während wichtige Aspekte des Gesamtsystems noch geklärt werden.

VORTEILE VON CONTAINER Container bieten klare Vorteile gegenüber klassischen VMs.

USE-CASE Die Images werden zur Erzeugung von Containern innerhalb eines Kubernetes Clusters für die Skalierung einer Komponente der Integrationsplattform Jenkins verwendet. Diese werden erzeugt, für den Einsatz als Agent innerhalb einer Master-Slave Konfiguration von Jenkins für verteilte Builds. Diese Agenten sind für die Ausführung eines Java Builds mit dem Build-Management Tool Maven zuständig.

ANFORDERUNGEN

- Aufgrund von hohen Gebühren für cloud-out-traffic soll die Image möglichst klein sein
- Die Umgebung soll Java Development Kit 11 (JDK11) und Maven enthalten und entsprechend konfiguriert werden
- Beim Bauen der Image sollen alle Ressourcen intern über eine private Artifactory-Instanz bezogen werden
- Zum Erstellen eines Builds soll Maven mit Artifactory kommunizieren und Daten übertragen können

- Zur Evaluation der Umgebung soll Maven einen erfolgreichen Build zu einem Testprojekt erstellen

A.1.1 *Auswahl eines Basis-Image*

Der erste Schritt in einem Dockerfile ist das Festlegen eines Basis-Images, aus welcher die Umgebung konfiguriert werden soll. Hierzu wird am Anfang der Dockerfile `FROM image:tag` definiert. Theoretisch kann eine Image auch aus einer leeren Umgebung `FROM scratch` erzeugt werden. Das ist jedoch unüblich, selbst bei Containern für embedded Systeme.

Hierzu gibt es im Docker-Hub öffentliche Images, die umkonfiguriert werden können. Die offizielle Maven image `maven:3.6.3-jdk-11`, die Maven und JDK11 bereits installiert hat, könnte beispielsweise für eine einfache Anwendung verwendet werden. Für den Use-Case innerhalb der Continuous Integration Plattform können die potenziellen Basis-Images aus dem Dockerhub zwischen folgenden Stufen in ihrer Ausstattung unterschieden werden:

1. Betriebssystem: `ubuntu, debian, alpine`
2. SDK: `jdk11, openjdk11, adoptopenjdk11, amazoncoretto11`
3. Build-Management: `maven`

Bei einer hohen Ausstattungsstufe besteht jedoch eine hohe Abhängigkeit von einer bestimmten Image, die wiederrum auf niedrigeren Stufen basiert. Für eine möglichst hohe Nachvollziehbarkeit wäre es sinnvoll eine möglichst niedrige Stufe zu wählen und die Umgebung innerhalb der eigenen Infrastruktur zu konfigurieren. Für eine möglichst hohe Effizienz der Container sind jedoch umfangreiche Kenntnisse über Betriebssysteme erforderlich. Diese fällt je nach Umfang in den Tätigkeitsbereich eines Systementwicklers.

VORTEILE VON SELBST GEBAUTEN IMAGES Für das Gesamtbild der Skalierbarkeit ist es jedoch sinnvoll das Ganze mit einer hohen Kompetenz anzugehen. Möglichst minimale und effiziente Container-Umgebungen wirken sich kumulativ auf den Ressourcenbedarf des ganzen Clusters aus. Bei einer Public-Cloud werden üblicherweise die genutzten Ressourcen mit einem Studensatz abgerechnet. Ein weiterer Vorteil entsteht durch das Prinzip des Layerings bei Container-Umgebungen. Auf einer eigenen Image können weitere Images basieren. Hierdurch muss lediglich eine Basis-Image entwickelt werden, in welchem nur die essentiellen Anpassungen vorgenommen werden, mit dem Ziel ein intern evaluiertes Betriebssystem bereitzustellen. Basierend hierauf können unabhängig voneinander Images aufbauen mit den entsprechenden Tools für die unterschiedlichen Programmiersprachen. Dadurch entsteht eine hohe Synergie und Flexibilität für die Systementwicklung und den Betrieb der Systeme.

MINIMALE IMAGES Eine Best-Practice innerhalb der Entwicklung von Docker-Images ist ihre Größe so niedrig wie möglich zu halten.

Die Linux-Image `alpine` ist hierfür als Basis-Image besonders beliebt, da sie mit einer Größe von nur `5 MB` eine Linux-Umgebung abbildet und eine Package-Repository besitzt [6]. Alpine benutzt jedoch nicht die C Standardlibrary `glibc`, sondern `musl libc` [33]. Daher sollte die Nutzung gerade zum Kompilieren von Quellcode abgewägt werden. Auch mit den offiziellen Docker-Images von Ubuntu und Debian kommt man auch mit entsprechenden Praktiken auf eine niedrige Größe. Für den Anwendungsfall innerhalb der CI Plattform ist es wichtig eine kleinstmögliche Java Installation durchzuführen. Hierfür gibt es entsprechende `slim` Editionen der JDK.

Ein Vergleich zwischen der normalen `openjdk:11-jdk` mit der `-slim`[2]:

```
openjdk:11-jdk
SIZE 306.67 MB
sha256:9efbdac6886418e7c473ee4d59ff728d029fad773364cb671794333d-
d16e4158

openjdk:11-jdk-slim
SIZE 216.36 MB
sha256:a67455311b2982c4cb795c8a38d55cf17e840c8fdb9ca82d14cd38d8-
43e24111
```

Beide Images basieren auf `debian:buster`, wobei die zweite Image auf `debian:buster-slim` basiert.

Ein vergleich zwischen `alpine:3.12` und `debian:buster-slim`

```
alpine:3.12
SIZE 2.3 MB
sha256:c929c5ca1d3f793bfdd2c6d6d9210e2530f1184c0f488f514f1bb808-
0bb1e82b

debian:buster-slim
SIZE 26.46 MB
sha256:9f8288b62780eaa2ee6341c6f8632990eae83d9b1caac9d0e4545e6f-
c8ce5e6d
```

Auf die absolute Größe kommt es nicht an. Viel wichtiger ist es bei der Konfiguration der Images sich genau mit den Layers zu befassen. Jedes Befehl in der Dockerfile erzeugt einen neuen Layer. Das Löschen von Dateien wirkt sich nicht auf die vorigen Layer aus. Daher sollten Befehle verkettet als ein Befehl ausgeführt werden und möglichst wenige Daten erzeugt werden.

Aufgrund der geringen Auswirkung der absoluten Größe wird möglichst auf eine Standardumgebung als Basis-Image verwendet. Hierfür wurde `debian:buster-slim` ausgewählt. Die Debian Image auf Dockerhub ist eine offizielle Image, die hauptsächtlich vom Debian Maintainer tianon gepflegt und veröffentlicht wird [37, 38].

2 Vgl. Images aus index.docker.io

WEITERE SCHRITTE

- Konfiguration der Images
- Kommunikation mit Artifactory
- Bezug von Build-Tools über Artifactory
- Installation von Java
- Cacerts
- Installation von Maven
- Konfiguration von Maven
- Einfügen eines Testjobs
- Bauen von Images
- Evaluation der Images
- Bereitstellung der Images
- Integration in Jenkins
- Agent über Secure Shell
- Agent über Java Network Launch Protocol

LITERATUR

[1] Rainer Alt, Gunnar Auth und Christoph Kögler. *Innovationsorientiertes IT-Management mit DevOps*. Springer Fachmedien Wiesbaden, 2017. DOI: 10.1007/978-3-658-18704-0. URL: https://doi.org/10.1007/978-3-658-18704-0.

[2] European Banking Authority. *EBA Guidelines on ICT and security risk management*. Courbevoie, Frankreich, 2019. URL: https://eba.europa.eu/eba-publishes-guidelines-ict-and-security-risk-management.

[3] Kerstin Bornemann und Felix Brandes. "Rechtlicher Rahmen des Digital Banking". In: *Praxishandbuch Digital Banking*. Hrsg. von Volker Brühl und Joachim Dorschel. Wiesbaden: Springer Fachmedien Wiesbaden, 2018, S. 357–406. ISBN: 978-3-658-18890-0. DOI: 10.1007/978-3-658-18890-0_18. URL: https://doi.org/10.1007/978-3-658-18890-0_18.

[4] Franz-Theo Brockhoff. "Effiziente Vertriebsunterstützung auf Basis einer serviceorientierten IT-Architektur". In: *Innovationen durch IT: Erfolgsbeispiele aus der Praxis Produkte — Prozesse — Geschäftsmodelle*. Hrsg. von Lothar Dietrich und Wolfgang Schirra. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, S. 381–396. ISBN: 978-3-540-34843-6. DOI: 10.1007/3-540-34843-3_24. URL: https://doi.org/10.1007/3-540-34843-3_24.

[5] Johannes Bussmann, Michael Fritsch, Christopher Schmitz, Jens Niebuhr und André Scholz. "Impulse durch neue Technologietrends". In: *Innovationen durch IT: Erfolgsbeispiele aus der Praxis Produkte — Prozesse — Geschäftsmodelle*. Hrsg. von Lothar Dietrich und Wolfgang Schirra. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, S. 21–35. ISBN: 978-3-540-34843-6. DOI: 10.1007/3-540-34843-3_3. URL: https://doi.org/10.1007/3-540-34843-3_3.

[6] Natanael Copa. *docker-alpine*. Online im Internet. 2020, August 16. URL: https://github.com/alpinelinux/docker-alpine.

[7] Georg Disterer. "ITIL-basierte Inbetriebnahme neuer Anwendungen". In: *HMD Praxis der Wirtschaftsinformatik* 48.2 (2011), S. 48–57. DOI: 10.1007/BF03340567.

[8] Georg Disterer. "ISO/IEC 27000, 27001 and 27002 for Information Security Management". In: *Journal of Information Security* 04.02 (2013), S. 92–100. DOI: 10.4236/jis.2013.42011. URL: https://doi.org/10.4236/jis.2013.42011.

[9] Joachim Dorschel. "Organisation und Prozesse der Bank-IT in der Digitalisierung". In: *Praxishandbuch Digital Banking*. Hrsg. von Volker Brühl und Joachim Dorschel. Wiesbaden: Springer Fachmedien Wiesbaden, 2018, S. 95–139. ISBN: 978-3-658-18890-0. DOI: 10.1007/978-3-658-18890-0_6. URL: https://doi.org/10.1007/978-3-658-18890-0_6.

[10] Florian Eismann. "Web 2.0 Banking – Was Kreditinstitute von der Fidor Bank lernen können". In: *Multi- und Omnichannel-Management in Banken und Sparkassen: Wege in eine erfolgreiche Zukunft*. Hrsg. von Harald Brock und Ingo Bieberstein. Wiesbaden: Springer Fachmedien Wiesbaden, 2015, S. 115–128. ISBN: 978-3-658-06538-6. DOI: 10.1007/978-3-658-06538-6_7. URL: https://doi.org/10.1007/978-3-658-06538-6_7.

[11] E. Fernández, S. Valle und G. Pérez-Bustamante. "Disruption Versus Discontinuity: Definition and Research Perspective From Behavioral Economics". In: *IEEE Transactions on Engineering Management* 67.3 (2020), S. 963–972.

[12] Bundesanstalt für Finanzdienstleistungsaufsicht. *Anlage 1: Erläuterungen zu den MaRisk in der Fassung vom 27.10.2017*. Bonn, Deutschland, 2017. URL: https://www.bafin.de/SharedDocs/Downloads/DE/Rundschreiben/dl_rs0917_marisk_Endfassung_2017_pdf_ba.pdf?__blob=publicationFile&v=5.

[13] Bundesanstalt für Finanzdienstleistungsaufsicht. *Mindestanforderungen an das Risikomanagement (MaRisk)*. Bonn, Deutschland, 2017. URL: https://www.bafin.de/SharedDocs/Downloads/DE/Rundschreiben/dl_rs1709_marisk_pdf_ba.pdf?__blob=publicationFile&v=2.

[14] Bundesanstalt für Finanzdienstleistungsaufsicht. *Bankaufsichtliche Anforderungen an die IT (BAIT)*. Bonn, Deutschland, 2018. URL: https://www.bafin.de/SharedDocs/Downloads/DE/Rundschreiben/dl_rs_1710_ba_BAIT.pdf?__blob=publicationFile&v=9.

[15] Bundesanstalt für Finanzdienstleistungsaufsicht. *Verwaltungspraxis*. Online im Internet. 2020, April 04. URL: https://www.bafin.de/DE/RechtRegelungen/Verwaltungspraxis/verwaltungspraxis_node.html;jsessionid=BBF2E4BF1B2116261748D049E90FACFB.1_cid392.

[16] Bundesanstalt für Finanzdienstleistungsaufsicht. *MaRisk-Novelle: Ist eine Verschiebung der MaRisk-Novelle zu erwarten?* Online im Internet (Stand 18.08.2020). 2020, April 16. URL: https://www.bafin.de/SharedDocs/FAQs/DE/Corona/Bankenaufsicht/Allgemeine_Aufsichtsthemen/2300_faq_corona_verschiebung_marisk_novelle.html?id=13962418.

[17] Legal Counsel at Interstellar Foundation Lindsay X. Lin und Stellar Development. "Deconstructing Decentralized Exchanges". In: *Stanford Journal of Blockchain Law Policy* (5. Jan. 2019). https://stanford-jblp.pubpub.org/pub/deconstructing-dex. URL: https://stanford-jblp.pubpub.org/pub/deconstructing-dex.

[18] Thomas Ganswindt. "Treibstoff der Wirtschaft". In: *Innovationen durch IT: Erfolgsbeispiele aus der Praxis Produkte — Prozesse — Geschäftsmodelle*. Hrsg. von Lothar Dietrich und Wolfgang Schirra. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, S. 57–64. ISBN: 978-3-540-34843-6. DOI: 10.1007/3-540-34843-3_5. URL: https://doi.org/10.1007/3-540-34843-3_5.

[19] Deutscher Sparkassen und Giroverband. *Mindestanforderungen an das Risikomanagement Interpretationsleitfaden Version 6.1*. 2019. URL: https://www.stiftung-wissenschaft.de/kunden/ebusti/xpage/s-wissenschaft.nsf/xsp/.ibmmodres/domino/OpenAttachment/kunden/ebusti/xpage/s-wissenschaft.nsf/2D49B25D9D45CDBCC12581F0004AC370/Anhang/DSGV_MaRisk-ILF_Version_6-1_Internet.pdf.

[20] Sara Gupta Sunil und Simonds. "Goldman Sachs' Digital Journey". In: *HBS CASE COLLECTION*. Mai 2019 überarbeitet. 2017, S. 518–039.

[21] Joanne Humble Jez und Molesky. "Why Enterprises Must Adopt Devops to Enable Continuous Delivery". In: *Cutter IT journal : the journal of information technology management* 24.8 (2011), S. 6–13. ISSN: 1048-5600. URL: http://twinkfed.homedns.org/TechDocs/itj1108.pdf#page=6.

[22] *IT-Grundschutz-Kompendium: Stand Februar 2020*. Bonn, Deutschland, 2020. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Kompendium/IT_Grundschutz_Kompendium_Edition2020.pdf?__blob=publicationFile&v=6.

[23] *Jenkins on Kubernetes Engine | Solutions | Google Cloud*. URL: https://cloud.google.com/solutions/jenkins-on-kubernetes-engine.

[24] Michael Knittl Silvia und Kunzewitsch. "Herausforderungen und Lösungsansätze bei der MaRisk-Umsetzung in einem internationalen Finanzkonzern". In: *INFORMATIK 2013 – Informatik angepasst an Mensch, Organisation und Umwelt*. Hrsg. von Matthias Horbach. Bonn: Gesellschaft für Informatik e.V., 2013, S. 1202–1215.

[25] Petra Koch, Frederik Ahlemann und Nils Urbach. "Die innovative IT-Organisation in der digitalen Transformation". In: *Managementorientiertes IT-Controlling und IT-Governance*. Hrsg. von Stefan Helmke und Matthias Uebel. Wiesbaden: Springer Fachmedien Wiesbaden, 2016, S. 177–196. ISBN: 978-3-658-07990-1. DOI: 10.1007/978-3-658-07990-1_11. URL: https://doi.org/10.1007/978-3-658-07990-1_11.

[26] Sven Korschinowski, Maximilian Forster und Luca Reulecke. "Blockchain – wie Banken die Technologie aus Prozess- und Produkt-Sicht nutzen können". In: *Praxishandbuch Digital Banking*. Hrsg. von Volker Brühl und Joachim Dorschel. Wiesbaden: Springer Fachmedien Wiesbaden, 2018, S. 277–290. ISBN: 978-3-658-18890-0. DOI: 10.1007/978-3-658-18890-0_13. URL: https://doi.org/10.1007/978-3-658-18890-0_13.

[27] T. J. Lehman und A. Sharma. "Software Development as a Service: Agile Experiences". In: *2011 Annual SRII Global Conference*. 2011, S. 749–758.

[28] Stephan Manz. "Digitale Transformation im Banking – lessons learned". In: *Praxishandbuch Digital Banking*. Hrsg. von Volker Brühl und Joachim Dorschel. Wiesbaden: Springer Fachmedien Wiesbaden, 2018, S. 161–187. ISBN: 978-3-658-18890-0. DOI: 10.1007/978-3-658-18890-0_8. URL: https://doi.org/10.1007/978-3-658-18890-0_8.

[29] Nikhil Pathania. *Pro Continuous Delivery*. Apress, 2017. DOI: 10.1007/978-1-4842-2913-2. URL: https://doi.org/10.1007/978-1-4842-2913-2.

[30] Klaus Rausch und Andreas Rothe. "Innovation erfordert eine geänderte IT-Governance". In: *Innovationen durch IT: Erfolgsbeispiele aus der Praxis Produkte — Prozesse — Geschäftsmodelle*. Hrsg. von Lothar Dietrich und Wolfgang Schirra. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, S. 451–471. ISBN: 978-3-540-34843-6. DOI: 10.1007/3-540-34843-3_29. URL: https://doi.org/10.1007/3-540-34843-3_29.

[31] Remigiusz Smolinski und Moritz Gerdes. "Mit ganzheitlichem Innovationsmanagement zur Finanzbranche der Zukunft". In: *Innovationen und Innovationsmanagement in der Finanzbranche*. Hrsg. von Remigiusz Smolinski, Moritz Gerdes, Martin Siejka und Mariusz C Bodek. Wiesbaden: Springer Fachmedien Wiesbaden, 2017, S. 37–57. ISBN: 978-3-658-15648-0. DOI: 10.1007/978-3-658-15648-0_3. URL: https://doi.org/10.1007/978-3-658-15648-0_3.

[32] Markus Strietzel, Sebastian Steger und Till Bremen. "Digitale Transformation im Banking – ein Überblick". In: *Praxishandbuch Digital Banking*. Hrsg. von Volker Brühl und Joachim Dorschel. Wiesbaden: Springer Fachmedien Wiesbaden, 2018, S. 13–29. ISBN: 978-3-658-18890-0. DOI: 10.1007/978-3-658-18890-0_2. URL: https://doi.org/10.1007/978-3-658-18890-0_2.

[33] Alpine Linux Development Team. *ABOUT*. Online im Internet (Stand: 2020, August 17). URL: https://www.alpinelinux.org/about/.

[34] Hüseyin Yüksel. *Digitale Transformation- Themenblock 3*. In: Digitale Transformation, Wahlpflichtmodul Sommersemester 2019. Darmstadt, Deutschland, 2019. URL: https://lernen.h-da.de/course/view.php?id=9634.

[35] *aus fintech wird finledger*. Online im Internet (Stand 24.08.2020). URL: https://www.finledger.de/.

[36] maidis. *Horizontal Striped Hot Air Balloons 1*. 2006. URL: https://openclipart.org/detail/18939.

[37] tianon. *tianon/debuerreotype/amd64*. Online im Internet (Stand 17.08.2020. URL: https://doi-janky.infosiftr.net/job/tianon/job/debuerreotype/job/amd64/.

[38] tianon. *debuerreotype/docker-debian-artifacts @ dist-amd64*. Online im Internet. 2020, August 03. URL: https://github.com/debuerreotype/docker-debian-artifacts/tree/dist-amd64#readme.