MASTERARBEIT

DESKTOP- UND WEB-ANWENDUNGEN

Unterscheidung, Einsatzgebiete und technische Möglichkeiten

ausgeführt am

Studiengang

Informationstechnologien und Wirtschaftsinformatik

Von: Philipp Pirker

Personenkennzeichen: 2010320008

Graz, am 25. November 2021 .....

Unterschrift

EHRENWÖRTLICHE ERKLÄRUNG

Ich erkläre ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die benutzten Quellen wörtlich zitiert sowie inhaltlich entnommene Stellen als solche kenntlich gemacht habe.

.....

Unterschrift

DANKSAGUNG

An dieser Stelle möchte ich mich bei einigen Personen bedanken, die mich speziell bei der Umsetzung dieser Masterarbeit unterstützt haben:

- Ein großes Dankeschön gilt meinem Betreuer DI Christoph Pangerl für das stets schnelle und konstruktive Feedback.
- Ein weiterer Dank gilt an die fünf Experten, die sich für meine Interviews die Zeit genommen haben und mir damit sehr wertvolle Informationen und Wissen zu den Themenpunkten geliefert haben. Die Erkenntnisse aus diesen Interviews waren zielführend für die umfassende Beantwortung der Forschungsfrage.
- Danke auch an mein näheres Umfeld, das mich während der Zeit der Umsetzung immer wieder unterstütz und ermuntert hat.

KURZFASSUNG

Web-Anwendungen sind heute kaum mehr aus dem Alltag wegzudenken. Oft bemerken Benutzerinnen und Benutzer nicht einmal, dass es sich bei der Anwendung um eine Web-Anwendung handelt. Web-Technologien sind ein Schlüsselelement für viele Funktionen und Services, die heutige Applikationen ausmachen. Sie ermöglichen die gleichzeitige Benutzung einer Anwendung von vielen Nutzerinnen und Nutzern, ohne dass dabei eine Installation seitens der nutzenden Personen notwendig ist.

Doch auch wenn Web-Anwendungen immer öfter zum Einsatz kommen, stellt sich die Frage, ob sie Desktop-Anwendungen auf allen Gebieten ersetzen können. Desktop-Anwendungen bieten viele Vorteile und Einsatzmöglichkeiten. Deshalb werden in dieser Arbeit die Eigenschaften und Merkmale sowie die Einsatzgebiete von Desktop- und Web-Anwendungen genauer betrachtet und analysiert. Dabei wird auf die Vor- und Nachteile der beiden Anwendungstypen eingegangen, um genauer abgrenzen zu können, wo deren Stärken und Schwächen liegen.

Die vorliegende Arbeit teilt sich in eine Literatur- und eine empirische Forschungsarbeit. In der Literaturarbeit werden die theoretischen Grundlagen dargelegt und die entsprechenden Erkenntnisse zu den oben angeführten Themenpunkten angeführt. Im empirischen Teil der Arbeit sollen die aus der Literatur gewonnenen Erkenntnisse durch Expertenwissen ergänzt werden. Dabei werden zwei wissenschaftliche Methoden, die Experteninterviews und die qualitative Inhaltsanalyse nach Mayring, angewendet.

Mit Hilfe der Erkenntnisse aus der Literaturarbeit und der empirischen Forschungsarbeit soll die Forschungsfrage beantwortet werden. Im Fazit wird die Forschungsfrage nochmals erörtert und schließlich beantwortet.

ABSTRACT

It is hard to imagine today's world without web applications, although users often do not even realize that they are using a web application. Web technologies are a key element for many functions and services that make up our applications today. They enable large numbers of users to use an application simultaneously, without requiring any installation on their part.

However, even though Web applications are being used more and more, the question is whether they can replace all desktop applications. Desktop applications also offer many advantages and have proven uses. Therefore, this thesis analyses the characteristics and features, as well as the areas of use of desktop and web applications, in greater detail. The advantages and disadvantages of the two application types are discussed, in order to delineate more precisely where their strengths and weaknesses lie.

This thesis is divided into a literature search and an empirical research part. The literature portion establishes the theoretical cornerstones and explains the corresponding findings on the abovementioned topics. Expert interviews were then conducted and the results analysed using the Mayring approach, in order to validate and supplement the findings of the literature search.

Combining the knowledge gained from the literature search and the expert interviews yielded sufficient knowledge about the topic to answer the research question in detail. Thus, the concluding chapter clarifies the different use cases for which the two different types of application are suitable and provides an answer to the research question.

INHALTSVERZEICHNIS

1 EINLEITUNG

"Alles, was wir hören, ist eine Meinung, keine Tatsache. Alles, was wir sehen, ist eine Perspektive, keine Wahrheit." Marcus Aurelius

Wir leben in einer vollständig vernetzten Welt und fast täglich erscheinen neue Technologien, Lösungen und Tools, die uns die Arbeit erleichtern oder sogar ganz abnehmen. Durch diese Technologien und das Internet ist es Menschen möglich, ihre Tätigkeiten an jedem Ort zu erledigen. Ob To-Do-Liste, Schrittzähler oder E-Mail-Programme − Apps (Applikationen) unterstützen im täglichen Leben. Dazu wird nur ein Endgerät und abhängig von der Applikation möglicherweise auch eine konstante Internetverbindung benötigt.

Doch auch wenn die neuesten Technologien zur Verfügung stehen, muss oft aus spezifischen Gründen ein Kompromiss zwischen Alt und Neu eingegangen werden. So muss in der Softwareentwicklung zu Beginn eines Projektes entschieden werden, ob die Software als Desktop- oder als Web-Anwendung entwickelt werden soll. Dahinter stehen verschiedene Überlegungen, die sich in hohem Maße auf die Entwicklung und den Funktionsumfang der Anwendung auswirken. Ausschlaggebend für die Entscheidung zwischen diesen beiden Anwendungstypen sind vor allem die Anforderungen an die Anwendung bzw. an die Funktionen, die diese bereitstellen soll.

Desktop-Anwendungen oder native Anwendungen sind durch ihre Anpassung an das entsprechende Betriebssystem optimiert, wenn es um die Stabilität, die Nutzung von Ressourcen und Effizienz geht. Sie bieten außerdem mehr Sicherheit, da die meisten Funktionen auch offline funktionieren und somit keine Verbindung mit dem Internet benötigt wird. Durch die lokale Speicherung von Daten sind sie vor Diebstahl geschützt. Doch Desktop-Anwendungen haben auch Nachteile. Durch ihre Anpassung an ein bestimmtes Betriebssystem sind sie nicht portabel und sie müssen für jedes gewünschte Betriebssystem eigens entwickelt bzw. daran angepasst werden. Dies führt zu einem erheblichen Aufwand und dadurch zu hohen Entwicklungskosten. Dennoch gibt es Anwendungsgebiete, bei denen Desktop-Anwendungen sinnvoll sind.

Web- oder Online-Anwendungen basieren auf dem Client-Server-Prinzip. Sie werden nicht auf dem lokalen Endgerät installiert, sondern laufen auf einem externen Server, der meist vom Software-Hersteller betrieben wird. Der Zugriff auf diese Web-Anwendung erfolgt in der Regel mit einem Web-Browser. Dies ermöglicht es, die Web-Anwendung mit jedem beliebigen Betriebssystem zu benutzen. Auch mobile Endgeräte sind davon nicht ausgeschlossen. Die Web-Anwendungen werden in Programmiersprachen geschrieben, die von allen Web-Browsern unterstützt werden. Dadurch müssen Web-Anwendungen nur einmal entwickelt werden und können dann auf allen Betriebssystemen und Geräten betrieben werden. Dafür ist jedoch eine aktive Internetverbindung erforderlich. Ist diese Verbindung nicht gegeben, können diese Applikationen nur eingeschränkt bzw. gar nicht benutzt werden. Speziell in den letzten Jahren wurde der Fokus bei der Entwicklung von Anwendungen immer stärker auf Web-Anwendungen gerichtet. Diese haben aber auch Einschränkungen. Der Zugriff auf die Ressourcen des

Endgeräts ist nicht immer vollständig möglich, was dazu führt, dass gewisse Funktionen, z. B. der Zugriff auf Funktionen des Betriebssystems, nur umständlich oder gar nicht zur Verfügung gestellt werden können. Doch durch die schnelle Weiterentwicklung von Web-Technologien steigen auch die Möglichkeiten bei der Umsetzung von Web-Anwendungen.

1.1 Ausgangssituation

Mit Hilfe des Internets und der modernen Web-Technologien ist es technisch möglich, gewisse Funktionen von Desktop-Anwendungen als Web-Anwendung zur Verfügung zu stellen. Da die Installation entfällt, sind die Web-Applikationen auch auf ressourcenschwachen Endgeräten verfügbar. Es genügt ein Web-Browser, der meist standardmäßig auf jedem internetfähigen Endgerät installiert ist und mit dem die Web-Anwendung aufgerufen und verwendet werden kann. Dadurch haben Web-Anwendungen bereits viele Vorteile gegenüber klassischen Desktop-Anwendungen.

Jedoch bieten auch Desktop-Anwendungen spezielle Funktionalitäten und Vorteile. Sie sind im Gegensatz zu Web-Anwendungen nicht direkt vom Internet abhängig und können somit relativ unabhängig auf jedem dafür geeigneten Endgerät betrieben werden. Sie bieten dadurch oft auch mehr Zuverlässigkeit und Stabilität − Eigenschaften, die in bestimmten Branchen essenziell für den Betrieb sind. Auch der direkte Zugriff auf Ressourcen oder Funktionen des Betriebssystems ist für gewisse Anwendungsfälle zwingend notwendig.

1.2 Problemstellung

Obwohl Web-Technologien und Web-Applikationen bereits zum Alltag für Software-Entwicklerinnen und Software-Entwickler gehören, gibt es noch Punkte, die ungeklärt sind. So bleibt die Frage offen, ob Web-Anwendungen aktuell in der Lage sind, ganze Desktop-Anwendungen zu ersetzen. Dafür müssen die beiden Anwendungstypen detailliert betrachtet werden.

Desktop-Applikationen bereiten zwar erhebliche Entwicklungskosten bzw. einen hohen Aufwand, da sie für verschiedene Betriebssysteme entwickelt oder an diese angepasst werden müssen, sie haben jedoch auch ihre Berechtigung. Hierbei müssen für eine Gegenüberstellung mit Web-Anwendungen die ausschlaggebenden Vor- und Nachteile betrachtet sowie die Einsatzgebiete ermittelt werden. Es bedarf einer Aufstellung aller Faktoren, um diesen Anwendungstyp weiter beurteilen zu können.

Web-Applikationen benötigen meist eine aktive Internetverbindung, da das Endgerät das Programm in der Regel nicht selbst ausführt. Stattdessen wird das Programm auf einem externen Server betrieben und das Endgerät dient lediglich als Anzeige- und Eingabegerät. Dies ist ein Beispielszenario, bei dem unklar ist, ob eine Desktop-Applikation nicht die bessere Variante wäre. Auch wenn Desktop-Applikationen nicht mehr die modernste Art und Weise der Software-Entwicklung darstellen, haben sie dennoch Vorteile. Deshalb muss auch auf die verschiedenen

Formen von Web-Anwendungen eingegangen werden, um genauer beurteilen zu können, inwieweit diese die Funktionen von Desktop-Anwendungen übernehmen können. Aufgrund der ständigen Weiterentwicklung verschiedenster Web-Technologien steigen die möglichen Funktionen immer weiter. Auf Basis dieser aktuell möglichen Funktionen kann eine qualitative Gegenüberstellung der beiden Anwendungstypen durchgeführt werden.

1.3 Forschungsfrage

Vor diesem Hintergrund soll der Frage nachgegangen werden, bei welchen Einsatzsatzgebieten Desktop- und Web-Anwendungen sinnvoll erscheinen. Mit Hilfe einer Gegenüberstellung der beiden Varianten sind die jeweiligen Unterschiede aufzuzeigen. Zudem soll analysiert werden, inwieweit es möglich ist, Desktop- durch Web-Anwendungen zu ersetzen.

Aus den genannten Zielen leitet sich für diese Masterarbeit die folgende Forschungsfrage ab:

Über welche Einsatzgebiete erstrecken sich Desktop- und Web-Anwendungen, was unterscheidet diese und inwieweit können Web-Anwendungen Desktop-Anwendungen ersetzen?

1.4 Abgrenzung und Begriffsdefinitionen

Speziell bei den Web-Anwendungen gibt es unterschiedliche Begrifflichkeiten. Um einen Überblick zu schaffen, werden diese in Kapitel 3 aufgeführt und detailliert beschrieben. Diese Definitionen dienen nicht direkt der Beantwortung der Forschungsfrage, sie werden aber der Vollständigkeit halber angeführt.

Die Arbeit gliedert sich in einen Literaturteil und in einen empirischen Forschungsteil. Die Literaturarbeit wurde anhand von Fachliteratur erstellt. Die entsprechenden Quellen sind im Quellenverzeichnis zu finden. Die Daten für die empirische Forschungsarbeit wurden durch Experteninterviews gewonnen. Anschließend wurden diese mit Hilfe der qualitativen Inhaltsanalyse nach Mayring aufbereitet und zusammengefasst.

1.5 Aufbau der Arbeit

Das erste Kapitel beginnt mit der Literaturarbeit zur Forschungsfrage. Hierbei soll speziell auf die in der Forschungsfrage aufgegriffenen Thematiken eingegangen werden und die Kernaspekte sollen näher erläutert werden. Damit soll ein Basiswissen geschaffen werden, das dazu dient, weitere Erkenntnisse zu gewinnen.

In Kapitel 2 der Literaturarbeit wird das Thema der Desktop-Anwendungen beschrieben und untersucht. Nach der Einleitung wird zunächst auf die spezifischen Eigenschaften und Merkmale von Desktop-Anwendungen eingegangen. Dies soll insbesondere dazu dienen, Desktop-Anwendungen von Web-Anwendungen zu unterscheiden. Danach werden die Einsatzgebiete von Desktop-Anwendungen aufgezählt und näher beschrieben. Dadurch soll eine Abgrenzung von

Desktop- und Web-Anwendungen mit Blick auf deren jeweilige Stärken ermöglicht werden. Darauffolgend werden die Vor- und Nachteile von Desktop-Anwendungen dargestellt. Dies soll ebenfalls dabei helfen, die Anwendungsart besser zu verstehen und zu erkennen, in welchen Fällen ein Einsatz ratsam und in welchen er nicht sinnvoll ist. Danach folgt ein Überblick über die Arten der Lizensierung von Desktop-Anwendungen, in dem alle gängigen Lizenzmodelle angeführt werden. Eine Zusammenfassung der Themenpunkte bildet den Schluss des Kapitels.

Kapitel 3 der Literaturarbeit behandelt das Thema der Web-Anwendungen. In der Einleitung wird ein Einblick in die Geschichte der Entwicklung von Web-Anwendungen gegeben. Darauffolgend werden deren spezifischen Merkmale und Eigenschaften beschrieben. Danach wird auf das Client-Server-Prinzip, das die Basis für Web-Anwendungen bildet, eingegangen. Anschließend werden die verschiedenen Arten von Web-Anwendungen dargestellt. Des Weiteren werden die typischen Cloud-Service-Modelle erklärt. Auch in diesem Kapitel werden die Einsatzgebiete für Web-Anwendungen genannt, um diese mit Desktop-Anwendungen vergleichen zu können. Danach werden die Vor- und Nachteile von Web-Anwendungen erläutert, gefolgt von einer Darstellung der Lizensierungsmodelle, die typisch für Web-Anwendungen sind. Den Schluss bildet auch in diesem Kapitel eine Zusammenfassung.

Kapitel 4, das letzte Kapitel der Literaturarbeit, beinhaltet das Thema der Migration bzw. der Migrationsstrategien. Nach einer Einleitung werden die typischen Anwendungsmuster bei einer Migration beschrieben und erläutert. Danach werden spezifische Herangehensweisen näher betrachtet.

In Kapitel 5 werden die verwendeten Methoden sowie die Vorgehensweise bei der empirischen Forschungsarbeit beschrieben. Einerseits werden die Vorbereitung und die Durchführung der Experteninterviews erklärt. Andererseits wird auf die Durchführung der qualitativen Inhaltsanalyse nach Mayring sowie die Auswertung der gewonnenen Daten eingegangen.

Die Ergebnisse der Arbeit werden in Kapitel 6 dargestellt. In Kapitel 7 folgt eine Diskussion der Ergebnisse, die auch mit den Erkenntnissen aus der Literaturarbeit abgeglichen werden.

Am Ende der Arbeit steht das Fazit, in dem die Ergebnisse der Arbeit zusammengefasst werden und die Forschungsfrage beantwortet wird. Abschließend wird ein Ausblick auf weiterführende Forschung gegeben.

2 DESKTOP-ANWENDUNGEN

Eine Desktop-Anwendung ist laut Definition jede Art von Software, die auf einem Computer, also auf einem Laptop oder einem Desktop-PC, installiert werden kann und die durch die Ausführung bestimmte Funktionalitäten oder Aufgaben erfüllt. Bis zum Aufkommen des Internets waren Desktop-Anwendungen die typische Form von Software-Anwendungen. Auch wenn Web-Applikationen mittlerweile viele Funktionen übernehmen können, sind Desktop-Anwendungen nicht veraltet. Desktop-Anwendungen umfassen noch immer eine Menge an Einsatzgebiete, in denen sie Vorteile gegenüber modernen Technologien bieten (Ford, 2020).

In diesem Kapitel werden zuerst die spezifischen Eigenschaften und Merkmale von Desktop-Anwendungen beschrieben. Durch das Aufzeigen soll eine Gegenüberstellung zu Web-Anwendungen, die im nächsten Kapitel erläutert werden, ermöglicht werden. Die Beschreibung von Eigenschaften und Merkmale soll auch dazu dienen, Desktop-Anwendungen genauer definieren zu können.

Danach wird auf die für Desktop-Anwendungen spezifischen Einsatzgebiete eingegangen. Hier werden aktuelle und typische Einsatzgebiete von Desktop-Anwendungen aufgezeigt und näher beschrieben. Dieser Abschnitt gibt insbesondere Auskunft darüber, wo Desktop-Anwendungen aus welchen Gründen eingesetzt werden.

Darauffolgend werden die Vor- und Nachteile aufgelistet. Daraus sollen die Stärken und Schwächen von Desktop-Anwendungen ersichtlich werden. In diesem Abschnitt wird auch thematisiert, dass bestimmte Funktionen sowohl Vor- als auch Nachteile haben können.

Zuletzt wird auf das Thema der Lizensierung eingegangen. Dabei wird zwischen kommerzieller und nichtkommerzieller Software unterschieden. Zudem wird auf die verschiedenen Arten von nichtkommerzieller Software eingegangen.

2.1 Eigenschaften/Merkmale

Bei einer Desktop-Anwendung handelt es sich meist um eine Art von Software, die fest auf einem Endgerät installiert wird. Dabei dient diese Anwendungssoftware dazu, bei allgemeinen oder betrieblichen Problemstellungen gewisse Tätigkeiten zu übernehmen (Borges, Schumacher, & Redeker, 2002). Dazu werden die benötigten Dateien zunächst von einer externen Quelle bezogen. Quellen können dabei Datenträger wie USB-Sticks, CDs oder auch das Internet sein. Heute werden Desktop-Anwendungen meist von den entsprechenden Herstellern per Internet zur Verfügung gestellt. Die benötigten Dateien bzw. Software-Pakete werden auf dem Endgerät installiert und das Programm ist dann ausführbar. Für die Lagerung bzw. die Speicherung der benötigten Daten wird während der Installation ein entsprechender Speicherort im Dateisystem des Computers bzw. des Betriebssystems erstellt. Dieses beinhaltet die Daten, bis die Desktop-Anwendung vom Endgerät entfernt wird.

Desktop-Anwendungen werden auch als ‚native Applikationen' bezeichnet. Unter einer nativen Applikation wird ein Programm bzw. eine Software verstanden, die für einen bestimmten Zweck und für ein bestimmtes Betriebssystem entwickelt wurde. Diese Software kann nicht auf anderen Betriebssystemen ausgeführt werden. Da die native Applikation speziell für ein vorgesehenes Betriebssystem und unter Umständen auch für ein bestimmtes Gerät entwickelt wurde, hat sie Zugriff auf viele Funktionen von Hardware und Software und kann damit Funktionen bieten, die für andere Applikationen nicht möglich wären. Durch die spezielle Anpassung an das Betriebssystem bzw. an die Hardware können die nativen Applikationen zusätzlich schneller und effizienter arbeiten als Applikationen, die nicht angepasst werden. (Eisenman, 2018).

Desktop-Applikationen bzw. native Applikationen werden in Programmiersprachen entwickelt, die vom definierten Betriebssystem unterstützt werden. Für die Entwicklung werden eigene Entwicklungsumgebungen bereitgestellt, die spezielle Werkzeuge für die Entwicklerinnen und Entwickler beinhalten. Diese Werkzeuge bzw. Tools übernehmen dabei Standardaufgaben und bieten bereits fertige Funktionen und Anzeigeelemente für die Benutzeroberfläche. Durch die Bereitstellung von Ressourcen und Funktionen erleichtern moderne Entwicklungsumgebungen die Entwicklung von Applikationen deutlich.

Ist eine Desktop-Anwendung auf einem Endgerät installiert, so kann diese in der Regel jederzeit, unabhängig von anderer Software, gestartet werden. Es ist auch keine aktive Internetverbindung erforderlich, da sich alle benötigten Daten und Dateien bereits auf dem Endgerät befinden. Speziell die Unabhängigkeit vom Internet ist ein zentraler Faktor bei Desktop-Anwendungen. In gewissen Situationen ist es z. B. aus Sicherheitsgründen nicht möglich, eine Anbindung an das Internet zu gewähren. Andererseits ist es auch nicht immer möglich, eine Internetverbindung zur Verfügung zu stellen. Durch die Unabhängigkeit vom Internet können Desktop-Anwendungen trotzdem verwendet werden.

2.2 Einsatzgebiete

Desktop- bzw. native Anwendungen kommen grundsätzlich immer dann zum Einsatz, wenn der Nutzerkreis eingeschränkt ist. Wenn eine Anwendung nur für ein bestimmtes Betriebssystem vorgesehen ist, ist es kaum sinnvoll, eine Web-Anwendung zu entwickeln. Mit der nativen Anwendung können alle Vorteile des Betriebssystems genutzt werden und die Anwendung kann im Hinblick auf Stabilität und Sicherheit optimiert werden.

Der Einsatz von Desktop- bzw. nativen Anwendungen ist vor allem sinnvoll, wenn die Anwendung intensive und komplexe Rechenaufgaben übernehmen soll (Jensen, 2017). Durch die Anpassung an ein Betriebssystem und den daraus folgenden Zugriff auf Hardware-Ressourcen und Funktionen des Betriebssystems kann eine hohe Performance und Effizienz erreicht werden. Die Anpassung an das Dateisystem ermöglicht die schnelle Bearbeitung und Verschiebung von Daten innerhalb des Gerätes. Berechnungen können an die vorliegende Hardware angepasst werden und dadurch kann das volle Potential genutzt werden. Andererseits kann durch die Optimierung auf ein System auch die Batterielaufzeit erhöht werden, indem Prozesse mit hohem

Energieverbrauch minimiert werden. Durch den direkten Zugriff auf Hardware-Ressourcen wird z. B. der Zugriff auf die Kamera oder auf biometrische Sensoren erleichtert.

Auch bei der Arbeit bzw. der Bearbeitung großer Datenmengen ist der Einsatz von Desktop-Anwendungen sinnvoll. Ein Beispiel sind hierfür Messdaten. Werden große Mengen an Messdaten erhoben und lokal gespeichert, so ist eine Desktop-Anwendung deutlich effizienter als eine Web-Anwendung. Die Daten werden von der Anwendung lokal geladen werden und die Änderung muss auch wieder lokal gespeichert. Müssten die Datenmengen für jeden Zugriff von einem externen Datenserver bezogen werden, würde dies die Produktivität massiv beeinträchtigen.

2.3 Vorteile

Desktop- bzw. native Anwendungen verursachen einen höheren Aufwand und höhere Entwicklungskosten als Web-Anwendungen. Dennoch bieten sie eine Vielzahl von Vorteilen. In diesem Abschnitt wird auf die spezifischen Vorteile von Desktop- bzw. nativen Anwendungen eingegangen.

Performance

Desktop-Anwendungen sind native Applikationen und somit an das entsprechende Betriebssystem angepasst bzw. dahingehend optimiert. Die Desktop-Anwendung hat somit vollen Zugriff auf das Dateisystem und alle verfügbaren Funktionen. Dadurch verfügt sie über einen klaren Vorteil, wenn es um Performance geht. Wenn die Hardware des Systems leistungsstark ist, kann die Anwendung diese Leistung auch nutzen. Speziell bei der Verarbeitung bzw. Berechnung von großen Datenmengen, die zentral am System liegen, gibt es deutliche Performance-Vorteile.

Nativen Applikationen stehen meist auch alle Hardware-Ressourcen zur Verfügung. Dieser direkte Zugriff bringt weitere Performance-Vorteile, denn die vorhandenen Ressourcen können optimal genutzt werden. Die Applikationen werden dazu bereits bei der Implementierung an die Hardwarefunktionen angepasst. Es gibt auch Anwendungsgebiete, bei denen spezielle Hardware für eine Software verwendet wird, damit die Zusammenarbeit zwischen beiden optimiert werden kann.

Offline-Einsatz

Ein Beispiel hierfür ist Microsoft Office als Desktop-Version. Sobald die Software auf einem Endgerät installiert ist, kann sie jederzeit gestartet bzw. aufgerufen werden. Hierzu ist keine Datenverbindung erforderlich. Allerdings gibt es einige Funktionen, die eine Internetverbindung benötigen. Die Hauptfunktionen der Software können aber uneingeschränkt genutzt werden.

Ein weiterer Vorteil beim Offline-Einsatz von Desktop-Anwendungen ergibt sich bei einer schlechten Internetverbindung. Bei der Verwendung von Web-Anwendungen muss je nach Anwendung mindestens eine konstante, meist auch eine schnelle Internetverbindung vorhanden sein. Eine langsame Internetverbindung kann sich bei Web-Anwendungen negativ auf die

Produktivität des Benutzers auswirken. Bei der Desktop-Anwendung befinden sich alle benötigten Daten am Gerät und somit kommt es nicht zu Einschränkungen der Produktivität.

Kostenfaktor

Langfristig gesehen sind die meisten Desktop-Anwendungen günstiger als Web-Anwendungen. Die Anschaffungskosten von Desktop-Anwendungen können bei bestimmten Software-Produkten wie Computer-Aided-Design(CAD)-Programmen abschrecken. Rechnet man diese Anschaffungskosten über die Nutzungsdauer, so sind die Kosten in den meisten Fällen dennoch günstiger als bei Tarifmodellen. Wenn eine langfristige Benutzung geplant ist und die Software nicht immer auf dem aktuellen Stand sein muss, ist eine Desktop-Anwendung zu empfehlen.

Sicherheit

Durch die Möglichkeit, Desktop-Anwendungen auch ohne aktive Internetverbindung zu betreiben, kann die Sicherheit der Anwendung und auch der bearbeiteten Daten gewährleistet werden. Werden Daten auf einem externen Server, z. B. in einer Cloud gespeichert, müssen sie vom Endgerät zum externen Speicherplatz transportiert werden. Dies bietet eine Angriffsmöglichkeit, da Hacker die sensiblen Daten auf diesem Weg abgreifen können. Hierbei gewährt die lokale Speicherung der Daten deutlich mehr Sicherheit.

Keine automatischen Upgrades

Automatische Updates sind bei Web-Anwendungen ein Standard. Kritischer sind dabei die Software-Upgrades, da sie die Version einer Software ändern, was wiederum zu Kompatibilitätsproblemen mit Dateien führen kann. Updates und Upgrades werden von den Anbietern meist zwar angekündigt, können aber nicht verhindert oder verschoben werden. Kommt es durch ein Update oder ein Upgrade zu Kompatibilitätsproblemen, so ist die Software bis zur Behebung des Fehlers nicht mehr richtig nutzbar. Dies kann speziell in großen Unternehmen massive Ausfallkosten verursachen.

Desktop-Anwendungen zwingen in der Regel keine Updates auf. Upgrades müssen meist manuell durchgeführt werden, da auch eine passende Lizenz erworben werden muss. Moderne Desktop-Anwendungen benachrichtigen zwar über mögliche Updates, diese können jedoch verschoben werden. Viele Desktop-Anwendungen bieten auch die Möglichkeit, automatische Updates vollständig zu deaktivieren. Hat das Gerät eine aktive Internetverbindung, ist dies nicht ratsam, da Updates oft Sicherheitslücken schließen. Dennoch kann es aus kompatibilitätstechnischen Gründen notwendig sein.

Keine Urheberrechtsverletzungen

Bei Web-Anwendungen und Cloud-Services werden die Daten zum größten Teil auf externen Servern gespeichert und sie können von dort aus verwaltet und freigegeben werden. Dies hat den Vorteil, dass Dateien schnell weitergereicht werden können und dass auch mehrere Parteien auf dieselbe Datei Zugriff haben und diese ändern können. Handelt es sich jedoch um urheberrechtlich geschützte Dateien, so ist dies ein kritischer Punkt.

Urheberrechtlich geschützte Dateien dürfen grundsätzlich nicht in Clouds gespeichert werden, da dies ein Verstoß gegen das Urheberrecht ist. Daten und Dateien dürfen nur vom Urheber oder

von der Urheberin selbst in einer Cloud gespeichert werden. Werden urheberrechtlich geschützte Daten bzw. Dateien in einer Cloud oder auf einem externen Datenserver gespeichert und für weitere Personen freigegeben, so kann dies rechtliche Folgen haben. Da bei Desktop-Anwendungen alle Daten lokal auf dem Endgerät gespeichert werden, können hier keine unbeabsichtigten Verletzungen des Urheberechts auftreten.

Eigentumsrecht

Bei genauer Betrachtung der allgemeinen Geschäftsbedingungen von Cloud-Providern fällt auf, dass sich diese eine Vielzahl an Rechten sichern. Dateien, die in der Cloud gespeichert werden, können je nach Vertrag vom Cloud-Betreiber selbst verwendet, genutzt, gespeichert, vervielfältigt oder veröffentlicht werden. Bei Web-Anwendungen werden Daten und Dateien meist direkt online gespeichert. Bei manchen Web-Anwendungen ist eine Offline-Speicherung nicht möglich.

Es gibt auch spezielle Cloud-Betreiber, die erweiterte Privatsphäre und Sicherheit der Daten gewährleisten. Dies führt aber meist zu höheren Tarifkosten. Somit bieten auch hier Desktop-Anwendungen den Vorteil, dass die eigenen Dateien auf dem lokalen Endgerät gespeichert bleiben und nicht für andere ersichtlich sind. Speziell bei sensiblen Daten empfiehlt sich eine lokale Speicherung.

2.4 Nachteile

Native Anwendungen bieten viele Vorteile gegenüber Web-Anwendungen. Jedoch gibt es auch einige Punkte, bei denen Desktop-Anwendungen im Vergleich zu Web-Anwendungen zurückbleiben. In diesem Abschnitt werden die Nachteile von Desktop- bzw. nativen Anwendungen aufgezeigt.

Entwicklungskosten

Desktop-Anwendungen sind native Applikationen und müssen deshalb für jedes Betriebssystem angepasst bzw. separat entwickelt werden. Abhängig von der Anzahl an Betriebssystemen kann dies zu hohen Entwicklungskosten führen. Die am meisten verbreiteten Betriebssysteme bei Laptops und Desktop-PCs sind aktuell Windows, MacOS und Linux. Bei Smartphones bzw. mobilen Endgeräten sind es Android und iOS. Sollte eine moderne Anwendung alle gängigen Betriebssysteme abdecken, so wäre eine Entwicklung für mindestens fünf Betriebssysteme notwendig. Dies erfordert eine große Zahl von Fachleuten, die wiederum Kenntnisse in den verschiedenen Betriebssystemen und den zugehörigen Programmiersprachen besitzen müssen. Aufgrund dessen ergeben sich in solchen Situationen erhebliche Entwicklungskosten, was dazu führt, dass sich nur große Konzerne eine solche Entwicklung leisten können.

Fehlende Portabilität

Für jedes Betriebssystem gibt es spezielle Entwicklungsumgebungen, die den Entwicklerinnen und Entwicklern die Entwicklung und Anpassung der Anwendung an ein bestimmtes Betriebssystem erleichtern. Doch diese Anpassung stellt wiederum einen großen Nachteil dar, wenn es um die Portabilität von Software geht. Die an ein Betriebssystem angepasste Software

kann dann nicht auf anderen Betriebssystemen verwendet werden und muss angepasst, im schlimmsten Fall sogar vollständig neu entwickelt werden.

Benötigter Speicherplatz

Im Gegensatz zu Web-Anwendungen werden Desktop-Anwendungen auf dem lokalen Endgerät installiert und sie speichern meist auch alle Daten darauf. Bei Textverarbeitungsprogrammen stellt dies kein Problem dar, da die Dateien nur wenig Speicherplatz benötigen. Anders ist es z. B. bei Videoschnittprogrammen. Hier müssen zuerst die Videos lokal abgespeichert werden, damit sie ins Videoschnittprogramm geladen werden können. Je nach Länge und Qualität der Aufnahmen kann es dabei zu einem hohen Speicherverbrauch kommen. Verfügt das Gerät nicht über genug Speicherplatz, kann das geschnittene Video im schlimmsten Fall nicht abgespeichert werden.

Die Installation der Anwendung selbst kann auch ein Problem hinsichtlich des Speicherplatzes verursachen. Anwendungen verfügen manchmal über große Bibliotheken die hunderte von Dateien enthalten, die je nach Tätigkeit von der Anwendung benötigt werden. Ohne den entsprechenden Speicherplatz kann die Anwendung nicht vollständig installiert werden. Dies führt meist zu Einschränkungen in der Funktionalität der Software.

Neue Updates von Programmen benötigen manchmal ebenfalls viel Speicherplatz, wenn auch nur vorübergehend. Programme laden die neuen Daten aus dem Internet herunter, installieren sie entsprechend und löschen erst dann die alten nicht mehr benötigten Daten von der Festplatte. Ist nicht genügend Speicherplatz dafür verfügbar, können Updates nicht durchgeführt werden oder sie werden während der Durchführung abgebrochen.

Manuelle Updates

Nicht alle modernen Desktop-Applikationen verfügen über automatische Updates. Manche Benutzerinnen und Benutzer deaktivieren diese automatischen Updates bei Anwendungen. Abgesehen davon, dass dadurch keine neuen Funktionen erhältlich sind, stellt dies kein großes Problem dar, wenn das Endgerät keine Verbindung mit dem Internet hat. Ist jedoch eine Internetverbindung gegeben, treten durch veraltete Software-Sicherheitslücken auf. Updates liefern nicht immer nur neue Funktionen, sondern beseitigen auch Fehler und Lücken im System. Werden Programme nicht regelmäßig aktualisiert, kann dies zu einem relevanten Sicherheitsproblem führen. Hacker nutzen gezielt solche Schwachpunkte in der Software aus, um an sensible Daten zu kommen. Deshalb sollten Anwendungen immer auf dem aktuellen Stand gehalten werden, wenn das Endgerät eine aktive Internetverbindung hat. Im Gegensatz zu Web-Anwendungen müssen Benutzerinnen und Benutzer von Desktop-Anwendungen dies oft selbst übernehmen.

2.5 Lizensierung

Beinahe jedes Unternehmen benötigt heute Software bzw. Programme für die Kommunikation, die Logistik, die Verwaltung, Finanzen und für weitere Anwendungen. Die dafür benötigten Softwarelösungen sind meist nicht zur freien Verfügung bestimmt und es werden Lizenzen

benötigt. Dabei gibt es eine Vielfalt von Softwarelizenzen. Diese unterscheiden sich z. B. in Bezug auf die Kosten, gewisse Beschränkungen und Bedingungen sowie Vereinbarungen hinsichtlich Services und Support.

Per definitionem ist eine Softwarelizenz ein Vertag zwischen dem Unternehmen, das die Software vertreibt, und der Endbenutzerin bzw. dem Endbenutzer. Anhand einer Lizenz schützt ein Unternehmen bzw. der Verfasser der Software sein geistiges Eigentum. Des Weiteren wird eine gewisse Abgrenzung über die Ansprüche definiert. Auch rechtsverbindliche Angaben, z. B. die Weitergabe von Lizenzen, Nutzungsrechte, Ansprüche auf Garantie und Rechte der Kundin bzw. des Kunden werden im Lizenzvertrag beschrieben.

2.5.1 Kommerzielle Software

Unter kommerzieller Software ist ein Programm bzw. eine Software zu verstehen, die speziell für den Verkauf an Endbenutzerinnen und Endbenutzer konzipiert und entwickelt wurde. Sie dient in der Regel einem kommerziellen Zweck und hat den Sinn, Einnahmen zu generieren (Saleck, 2005). In den meisten Fällen wird kommerzielle Software nicht direkt verkauft, sondern lizensiert. Zu Beginn wurde kommerzielle Software von Unternehmen als Lösung für bestimmte Probleme entwickelt und an Organisationen oder Privatpersonen verkauft, die diese Lösung benötigten.

Mittlerweile gibt es auch Open-Source-Anwendungen, die die Lösungen kommerziell an Organisationen oder private Anwenderinnen und Anwender verkaufen bzw. lizensieren. Hierbei kommt es oft zu Missverständnissen, denn Open Source bedeutet nicht, dass eine Software kostenlos zur Verfügung steht. Es gibt sehr wohl Open-Source-Software, bei der die Distributionslizenz kostenpflichtig ist, es wird jedoch immer der Quellcode mitgeliefert, damit die Kundin bzw. der Kunde gewisse Anpassungen an das eigene System vornehmen kann.

Bestimmte Unternehmen erlauben Privatpersonen, die von ihnen bereitgestellte kommerzielle Softwarelösung für nichtkommerzielle Zwecke zu verwenden. Ein Beispiel dafür ist die Software TeamViewer. Diese Softwarelösung ermöglicht den Remote-Desktop-Zugriff auf andere Endgeräte. Für den gewerblichen Gebrauch ist eine Lizenz erforderlich, private Anwenderinnen und Anwender können dieses Programm jedoch kostenlos nutzen.

2.5.2 Nichtkommerzielle Software

Das Thema der freien Software kann zur Verwirrung führen. Hier haben sich bestimmte Fachbegriffe etabliert, die nicht immer leicht zu unterscheiden sind, speziell wenn es um Begriffe wie ‚Freeware' und ‚Free Software' geht. Im Folgenden werden die meistverbreiteten Begriffe genauer beschrieben.

Freie Software

Freie Software unterliegt keinen Urheberrechten, sie ist veränderbar bzw. frei modifizierbar und kann von der Benutzerin bzw. dem Benutzer vollständig geändert oder umgestaltet werden. Freie Software darf auch weiterverteilt und gewinnbringend eingesetzt werden. Eine häufige Erscheinungsform von freier Software ist Open-Source-Software, auf die nachfolgend

eingegangen wird. Freie Software zeichnet sich nicht nur durch den Nutzen und die geringen Kosten aus, sondern sie zeigt auch neue Wege für die Wissensvermittlung in der Bildung und die Weitergabe von Geisteseigentum (Grassmuck, 2004).

Freeware

Unter Freeware wird kostenfreie Software verstanden. Sie unterliegt einem Urheberrecht, ist aber kostenlos erhältlich. Der Quellcode ist dabei nicht immer frei verfügbar und eine Anpassung bzw. Weiterentwicklung nicht immer möglich. Freeware wird meist für die private Nutzung kostenlos mit Grundfunktionen zur Verfügung gestellt. Ist sie für Unternehmen bestimmt oder werden erweiterte Funktionen gewünscht, dann muss dafür bezahlt werden.

Open-Source-Software

Open-Source-Software wird erfolgreich in Wirtschaft und Technik eingesetzt (Brügge, et al., 2004). Dabei handelt es sich um einen Programmcode, der öffentlich bereitgestellt wird. Jede Person die entsprechenden Kompetenzen dafür hat, kann den Quellcode ansehen, duplizieren und auch verändern. Open-Source-Programme werden meist von Gemeinschaften aus Entwicklerinnen und Entwicklern geschaffen und weiterentwickelt, wobei diese oft sogar kostenlos bzw. aus eigenem Interesse an diesem Konstrukt arbeiten. Dadurch steht heute eine Vielzahl an Applikationen zur Verfügung, die kostenfrei genutzt werden können. Wie in Kapitel 2.5.1 beschrieben, ist Open-Source-Software jedoch nicht immer kostenfrei. Der Bezug der Software ist meist mit keinen Kosten verbunden, jedoch sind gewisse Dienstleistungen der bereitstellenden Unternehmen kostenpflichtig. Dennoch bieten Open-Source-Lösungen Vorteile wie etwa die vollständige Einsicht in den Code und dadurch Transparenz, Flexibilität bei der Anpassung an ein gegebenes System, niedrigere Kosten und keine Vertragsverbindlichkeiten.

2.6 Zusammenfassung

Trotz des ständigen Anstieges an Web-Anwendungen haben Desktop-Anwendungen noch eine klare Daseinsberechtigung. Desktop- bzw. native Anwendungen haben ihre eigenen Einsatzgebiete und haben Vorteile, die Web-Anwendungen nicht bzw. nur mit zusätzlichem Aufwand bieten. Desktop-Anwendungen sind auch nicht veraltet, sie beruhen nur auf einem anderen Prinzip als Web-Anwendungen. Auch wenn Web-Anwendungen die grundlegenden Funktionen der Desktop-Anwendung ersetzen können, behalten sie ihre Einsatzgebiete bei.

Desktop-Anwendungen haben die Eigenschaft, dass sie schnell am Endgerät installiert werden. Dabei werden die Anwendungen von einer externen Quelle (CD, USB-Stick, Internet) bezogen und auf das Datenlaufwerk des Endgeräts übertragen. Dies bietet wiederum den Vorteil, dass die Anwendung auch ohne eine aktive Internetverbindung betrieben werden kann, denn die Daten für die Anwendung sind am Endgerät gespeichert.

Eine weitere Eigenschaft von Desktop-Anwendungen ist, dass sie von Entwicklerinnen und Entwicklern an das jeweilige Betriebssystem angepasst werden. Da die Desktop-Anwendung direkt auf einem System installiert wird, muss sie an das entsprechende Betriebssystem angepasst werden. Dies bietet den Vorteil, dass alle Funktionen des Betriebssystems genutzt

werden können und dadurch die Performance der Anwendung gesteigert werden kann. Ein Nachteil ist dabei, dass die Desktop-Anwendung von den Entwicklerinnen und Entwicklern für alle gewünschten Betriebssysteme extra entwickelt werden muss. Dies bringt einen großen zusätzlichen Entwicklungsaufwand und in der Folge höhere Entwicklungskosten mit sich.

Zum Einsatz kommen Desktop-Anwendungen meist, wenn der Nutzerkreis eingeschränkt ist, bzw. wenn die Anwendung auf ein System bzw. auf ein Betriebssystem optimiert werden soll. Die Anwendung kann dabei z. B. auf Sicherheit oder Performance optimiert werden. Soll eine Anwendung nur auf einem bestimmten Betriebssystem laufen, ist eine Web-Anwendung nur selten sinnvoll. Durch die Anpassung an ein System bzw. Betriebssystem kann die Desktop-Anwendung optimal betrieben werden und sie bietet deutliche Vorteile gegenüber Web-Anwendungen. Heute werden aber auch schon Kombinationen aus Desktop- und Web-Anwendungen angeboten. Ein Beispiel ist Office 365, bei dem z. B. Word als Desktop- und Web-Anwendung betrieben werden kann.

Durch die Anpassung an das Betriebssystem und die Hardware haben Desktop-Anwendungen im Hinblick auf die bereits genannten Punkte ‚Performance' und ‚Sicherheit' deutliche Vorteile. Speziell die lokale Datenverarbeitung ist dadurch schnell und sicher durchzuführen. Rechenintensive Programme, bei denen ständig viele Daten generiert und gespeichert werden, laufen als Desktop-Anwendung deutlich schneller und stabiler.

Der Offline-Einsatz der Desktop-Anwendung bietet einen weiteren Vorteil gegenüber Web-Anwendungen. Auch wenn Web-Anwendungen bereits gewisse Funktionen offline zur Verfügung stellen können, stehen bei Desktop-Anwendungen immer alle Funktionen zur Verfügung. Des Weiteren bietet der Offline-Einsatz einen deutlichen Vorteil beim Sicherheitsaspekt. Sensible Daten können vom Internet getrennt verwaltet und gespeichert werden.

Ein großer Nachteil von Desktop-Anwendungen ist die fehlende Portabilität der Anwendungen. Die Entwicklerinnen und Entwickler müssen die Anwendung für alle gewünschten Betriebssysteme neu entwickeln bzw. modifizieren. Dies führt zu dem weiteren Nachteil, dass für Desktop-Anwendungen hohe Entwicklungskosten entstehen können. Auch die manuellen Updates sind ein Nachteil, denn sie gefährden die Sicherheit des Systems und der Daten.

3 WEB-ANWENDUNGEN

Alles begann mit einer statischen Webseite, erstellt von Tim Berners-Lee und Robert Cailliau vom Europäischen Kernforschungszentrum (CERN). Die damalige Webseite war eine Ansammlung von Dokumenten, bestehend aus statischen Elementen. Die dafür benutzte Skriptsprache war bereits Hypertext Markup Language (HTML), die auch heute noch zum Einsatz kommt, wenn auch in einer moderneren Form. Das Internet hat sich in der Zeit bis heute erheblich weiterentwickelt. Durch immer weiter entwickelte Web-Technologien wurde es möglich, nicht nur Informationen, sondern vollständige Anwendungen online zur Verfügung zu stellen. Komplexe Web-Anwendungen sind heute überall zu finden, ganz gleich, ob es sich dabei um soziale Netzwerke, Onlineshops oder Speicherdienste handelt (Casteleyn, Daniel, Dolog, & Matera, 2009).

Der Begriff ‚Web-Anwendung' bezeichnet eine Software-Anwendung, die von der Benutzerin bzw. dem Benutzer über einen Web-Browser ausgeführt wird. Ein Web-Browser hat dabei die Aufgabe, Informationen von einem Server zu empfangen, diese zu visualisieren und Interaktionen, z. B. Eingaben der Benutzerin bzw. des Benutzers, zurück an den Server zu senden. Der große Vorteil an diesem Prinzip entsteht dadurch, dass die Benutzerin bzw. der Benutzer nicht von einem Betriebssystem abhängig ist, da für die Benutzung nur ein Web-Browser benötigt wird. Nachdem auf jedem gängigen Betriebssystem ein Web-Browser vorinstalliert ist, können Web-Anwendungen plattformübergreifend eingesetzt werden. Aufgrund dessen wurden Web-Anwendungen seit den 1990er Jahren immer beliebter bei Entwicklerinnen und Entwicklern sowie bei Benutzerinnen und Benutzern.

Entwicklerinnen und Entwickler müssen die Anwendung nicht für mehrere Betriebssysteme entwickeln. Eine Software wird nur einmal entwickelt und kann dann auf allen Betriebssystemen mit aktuellen Web-Browsern eingesetzt werden. Jedoch gibt es auch hier Einschränkungen. Obwohl es bereits viele Standards bzw. Normen gibt, verarbeiten Web-Browser den Quellcode oft auf verschiedene Art, was zu Komplikationen führen kann. Hierbei sind Entwicklerinnen und Entwickler in hohem Maße gefordert, denn sie müssen auf alle gängigen Web-Browser Rücksicht nehmen bzw. sich dafür entscheiden, dass gewisse Web-Browser nicht unterstützt werden.

Doch Web-Anwendungen waren nicht immer so beliebt wie heute. Zu Beginn mussten Entwicklerinnen und Entwickler oft radikale Lösungen für die aufkommenden Probleme finden, da es an Standards bzw. Normen fehlte. Unter anderem aufgrund fehlender Standards bei Web-Browsern war es nur begrenzt möglich, Web-Anwendungen flüssig auf den damaligen Betriebssystemen bzw. Computermodellen zu betreiben. Abhilfe wurde mit eigens entwickelten Client-Programmen geschaffen, die erst recht wieder auf dem Rechner der Benutzerin bzw. des Benutzers installiert werden mussten. Zudem waren die Client- und Serverkomponenten an das Betriebssystem bzw. an die Architektur des Rechners gebunden. Webseiten wurden als statische Dokumente übermittelt, was die Interaktivität bei der Arbeit mit der Webseite erschwerte. Die Durchführung von Änderungen benötigte zudem Zeit, da die Webseite aktualisiert werden musste.

Der Durchbruch gelang im Jahr 1995, als JavaScript vorgestellt wurde. Dabei handelt es sich um eine Skriptsprache, die Entwicklerinnen und Entwicklern die Möglichkeit bietet, Elemente der Benutzeroberfläche dynamisch zu modifizieren. JavaScript brachte auch den Vorteil, dass Webseiten deutlich an Schnelligkeit gewannen, da die Informationen nicht immer an den Server retour geliefert werden mussten. Dadurch muss der Server nicht die Informationen für die Generierung der Webseite senden, denn dies übernahmen fortan die Skripte. Die meisten modernen Web-Anwendungen benutzen heute JavaScript (Flanagan, 2012).

Ein weiterer Durchbruch folgte im Jahr 2005, als Ajax vorgestellt wurde. Entwicklerinnen und Entwicklern wird mit Hilfe von Ajax die Entwicklung von asynchronen Web-Anwendungen ermöglicht (Carl, 2006). Web-Browser bzw. Web-Anwendungen können mit Ajax Daten vom Server beziehen bzw. Daten an diesen senden, ohne dass die Webseite neu geladen werden muss. Es werden nur die benötigten Daten gesendet oder geliefert, was dazu führt, dass die Effizienz von Web-Anwendungen deutlich gesteigert wird, da nicht die gesamte Webseite heruntergeladen werden muss.

Web-Anwendungen bestehen meist zumindest aus HTML, Cascading Style Sheets (CSS) und JavaScript-Code. HTML wurde entwickelt, um eine logische Struktur zu schaffen, in der verschiedene Inhalte angezeigt werden können, und stellt somit eine Art Grundgerüst der Webseite dar (Musciano & Kennedy, 2002). CSS dient zur grafischen Gestaltung der Webseite. Mit der Skriptsprache können die einzelnen Elemente aufgegriffen und verändert werden (Magazine, 2012).

In der Entwicklungsgeschichte von Web-Anwendungen gab es aber noch deutlich mehr Technologien, die das Ziel hatten, die Entwicklung von Web-Anwendungen zu vereinfachen. Beispiele dafür sind Java, Silverlight und Adobe Flash. In früheren Zeiten wurden diese Web-Technologien eingesetzt, um z. B. Videos oder Musik abspielen zu können. Heute übernimmt diese Funktionen HTML5, das bereits integrierte Tags dafür bereitstellt.

Inzwischen sind Web-Technologien so weit entwickelt, dass Web-Anwendungen oft gar nicht mehr von ihren Desktop-Anwendungen zu unterscheiden sind. Als Beispiel dient hier Microsoft Office, das die meisten Anwendungen, z. B. Word, Excel und PowerPoint, bereits als vollständige Web-Anwendungen bereitstellt. Somit müssen die Programme nicht mehr am Rechner der Benutzerin bzw. des Benutzers installiert werden, um Dokumente bearbeiten zu können. Dazu kommt, dass das Internet heute allgegenwärtig ist. Aber es wird nicht nur für bzw. bei der Arbeit genutzt, sondern auch zu Hause für private Zwecke, z. B. Unterhaltung (Gaming), Kommunikation (Soziale Medien) oder geschäftliche Interaktionen (Online-Banking).

In diesem Kapitel werden zunächst die spezifischen Eigenschaften und Merkmale von Web-Anwendungen beschrieben. Danach folgt eine Beschreibung des Client-Server-Prinzips. Darauffolgend werden die verschiedenen Arten, die Cloud-Service-Modelle sowie die Vor- und Nachteile von Web-Anwendungen aufgelistet. Zum Schluss werden die Arten der Lizenzierung beschrieben und das Kapitel wird zusammengefasst.

3.1 Eigenschaften/Merkmale

Web-Anwendungen sind Anwendungen, die bestimmte Funktionen anbieten und über einen Web-Browser aufgerufen werden können. Genauer handelt es sich bei einer Web-Anwendung um ein Client-Server-Programm, bei dem Informationen zwischen Client und Server ausgetauscht werden. Der Client stellt die Benutzerin bzw. den Benutzer dar, die bzw. der das Programm benutzt. Der Server nimmt Aufrufe bzw. Eingaben des Clients entgegen und bearbeitet bzw. speichert diese. Ein vereinfachtes Beispiel einer Web-Anwendung ist ein Kontaktformular auf einer Webseite, bei dem die Daten eingegeben und an den Server gesendet werden können. Die Web-Anwendung sendet die Daten vom Client an den Server und dieser speichert sie in einer Datenbank ab. Es gibt Merkmale, die Web-Anwendungen spezifizieren und eine Unterscheidung von typischer Software erleichtern. Diese Merkmale werden folgend aufgezählt.

Responsive-Design

Die Anforderungen und Erwartungen der Benutzer an Web-Anwendungen sind höher denn je. Die Anwendungen müssen ständig zur Verfügung stehen. Außerdem sollen sie auf allen Endgeräten einsetzbar sein, was speziell bei mobilen Endgeräten eine Herausforderung darstellen kann. Der Fokus liegt hierbei speziell auf dem Responsive-Design, damit die Web-Anwendung immer perfekt an die Bildschirmgröße des Endgerätes angepasst wird. Werden wichtige Funktionselemente wie Buttons nicht richtig angezeigt, so schränkt dies das Nutzererlebnis ein und die Web-Anwendung verliert an Attraktivität.

Skalierbarkeit

Web-Anwendungen werden meist in einer Cloud-Umgebung betrieben. Dies bietet den Vorteil, dass bei ansteigenden Lasten die Infrastruktur einfach angepasst werden kann. Zum Beispiel kann bei steigenden Zugriffen, etwa zu Weihnachten, die Anzahl der Server mit wenigen Mausklicks erweitert werden und somit die Last gut verteilt werden. Wird weniger Leistung benötigt, so können mit wenigen Mausklicks Server abbestellt bzw. andere Ressourcen wie RAM und Speicher reduziert werden, auch um Kosten zu sparen. Wird eine Anwendung nicht in einer Cloud-Umgebung betrieben, kann auf solche Vorteile nicht zugegriffen werden. Ohne Cloud-Umgebung müssten die Hardware-Komponenten immer zur Verfügung stehen und entweder im Leerlauf betrieben oder heruntergefahren werden, wenn sie nicht benötigt werden.

Plattformübergreifend

Ein weiteres Merkmal von Web-Anwendungen ist die plattformübergreifende Kompatibilität der Anwendung. Nachdem die Web-Anwendung primär auf einem Server bzw. in einer Cloud-Umgebung betrieben wird, kann die Anwendung von der Benutzerin bzw. dem Benutzer mit dem Browser genutzt werden. Somit gibt es keine Einschränkungen für Betriebssysteme mehr, denn jedes Betriebssystem hat einen Browser bzw. für jedes gängige Betriebssystem (Windows, MacOS, Linux) ist ein Browser verfügbar (Krämer, 2021). Dadurch muss die Web-Anwendung nur einmal entwickelt werden, was Entwicklungskosten spart.

Modularer Aufbau

Desktop-Anwendungen sind meist Monolithen. Das bedeutet, dass das ganze Programm ein einziges komplexes Konstrukt ist. Müssen Änderungen an einer Funktion durchgeführt werden, so müssen meist große Teile angepasst werden. Dies ist aufwändig und verursacht speziell bei komplexen Anwendungen hohe Kosten. Es gibt auch Web-Anwendungen, die als Monolith aufgebaut wurden, aber die Anzahl sinkt (Newman, 2020).

Moderne Web-Anwendungen werden anhand von Microservices aufgebaut. Jede Funktion der Anwendung besitzt einen eigenen Service. Die Web-Anwendung ist die Summe aller Services. Diese sind miteinander verknüpft und erfüllen somit ihren Zweck. Der Vorteil von Microservices ist dabei, dass eine Änderung einer Funktion nur den zuständigen Service betrifft. Somit muss nur dieser Service anpasst werden. Die Kompatibilität bleibt erhalten und der Entwicklungsaufwand ist deutlich geringer (Nadareishvili, Mitra, McLarty, & Amundsen, 2016).

Einfaches Testen

Durch den modularen Aufbau der Microservices wird auch das Testen der Anwendung während der Entwicklung vereinfacht. Die Entwicklerinnen und Entwickler können in einer Testumgebung neue Features bzw. modifizierte Services testen, ohne das Produktivsystem zu gefährden. Anhand von Unit-Tests können die einzelnen Bausteine getestet werden. Durch automatisierte Tests können Fehler und die Entwicklungskosten weiter reduziert werden.

3.2 Client-Server-Prinzip

Web-Anwendungen basieren auf dem Client-Server-Modell, bei dem die Ressourcen und Services den Clients durch den Server bereitgestellt werden. Klassische Beispiele sind hierfür Mail-, Daten- und Webserver. Auf den Servern lagern bestimmte Ressourcen, die von den Clients (z. B. Smartphones, Tablet, Laptops) abgerufen werden können (Abts, 2019).

Das Client-Server-Modell folgt einem bestimmten Ablauf. Die initiale Verbindung geht vom Client aus. Dieser schickt eine Anfrage an den Server. Der Server kann dabei entscheiden, ob er die angeforderte Verbindungsanfrage annimmt ober ablehnt. Nimmt der Server die Anfrage des Clients an, so wird folgend eine Verbindung aufgebaut. Diese wird durch ein bestimmtes Protokoll ermöglicht, das je nach Anwendungsfall unterschiedlich sein kann. Bei einem Aufruf eines Webservers bzw. einer Webseite kommt meist das HTTP-Protokoll zum Einsatz. Bei einer Verbindung mit einem Mailserver kann z. B. das SMTP-Protokoll verwendet werden. Die verschiedenen Protokolle haben bestimmte Eigenschaften und Funktionalitäten wie die Authentifizierung. Dabei wird geprüft, ob die benötigten Informationen für die Anmeldung mit den Daten am Server übereinstimmen.

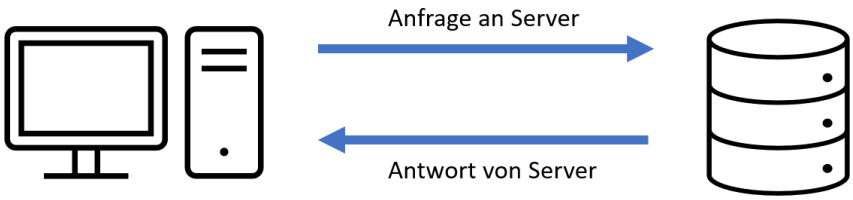

Abbildung 3-1: Einfache Abbildung Client-Server-Modell

3.3 Arten von Web-Anwendungen

Web-Anwendungen ersetzen die alten Desktop-Anwendungen. Sie sind bequemer zu bedienen, einfach zu aktualisieren und nicht an ein Gerät gebunden. Was Benutzerinnen und Benutzer meistens nicht auffällt ist, dass eine Web-Anwendungen verschiedenen Formen haben kann. In der Entwicklung von Web-Anwendungen haben sich unterschiedliche Muster für die Umsetzung dieser etabliert. Dabei ist zu erwähnen, dass diese Formen bzw. Typen von Anwendungen auch miteinander kombiniert werden können. Die gängigsten Formen werden nachfolgend aufgelistet und beschrieben (Hoffman, 2020).

3.3.1 Statische Web-Anwendungen

Eine Web-Anwendung wird als statisch bezeichnet, wenn sie direkt an den Browser der Benutzerin bzw. des Benutzers übermittelt werden kann, ohne dass serverseitig Änderungen am Inhalt durchgeführt werden müssen. Das heißt, der HTML-, CSS- und JavaScript-Code bleibt unverändert. Statische Web-Anwendungen nutzen in der Regel Technologien, die im Browser inkludiert sind, anstatt ständig auf den Server zuzugreifen. Dies bedeutet, dass bei einer statischen Web-Anwendung der Browser allein für das Aufrufen von Daten und das Kompilieren in HTML verantwortlich ist. Bei typischen Web-Anwendungen übernimmt dies grundsätzlich der Server.

3.3.2 Dynamische Web-Anwendungen

Eine dynamische Web-Anwendung zeigt bei jedem Aufruf einen unterschiedlichen Inhalt an. Dies wird durch bestimmte Faktoren definiert, z. B. einer angemeldeten Benutzerin bzw. einem angemeldeten Benutzer, bei der bzw. dem die persönlichen Daten oder Bestelldetails angezeigt werden, oder durch die Tageszeit, anhand der die Webseite in unterschiedlichen Farben angezeigt wird. Grundsätzlich wird bei dynamischen Web-Anwendungen zwischen zwei Arten, nämlich einer clientseitigen und einer serverseitigen, unterschieden.

Von clientseitigem Skripting spricht man, wenn die Webseite auf eine Aktion reagiert und sich dadurch verändert. Beispiele dafür sind Mausbewegung auf einem Feld, Mausklicks oder Eingaben mit der Tastatur. Durch die Aktion wird eine Veränderung angeregt, die einen

clientseitigen Inhalt generiert. Ein clientseitiger Inhalt ist ein Inhalt, der vom Gerät der Benutzerin bzw. des Benutzers generiert wird. Meist wird dabei ein Inhalt vom Server heruntergeladen und dann vom Web-Browser in die Webseite integriert. Solche Änderungen werden in der Regel mit JavaScript abgewickelt. Von serverseitigem Skripting spricht man, wenn sich eine Webseite beim bzw. durch das Laden ändern. Beispiele hierfür sind Anmeldeseiten und Warenkörbe. Serverseitige Inhalte werden generiert, wenn eine Webseite geladen wird.

3.3.3 Single-Page-Applikationen

Eine Single-Page-Applikation ist eine Web-Anwendung, die nur eine HTML-Webseite als Shell für alle Webseiten der Anwendung verwendet und deren Interaktionen mit der Endbenutzerin bzw. dem Endbenutzer durch die Verwendung von JavaScript, HTML und CSS realisiert werden. Der größte Teil der Entwicklung einer Single-Page-Applikation findet im Frontend statt, im Gegensatz zu traditionellen Web-Anwendungen, die in hohem Maße auf Webserver-Interaktionen angewiesen sind und bei jeder Navigation neue Webseiten laden. Single-Page-Applikationen ähneln in ihrem Verhalten und ihrer Entwicklung nativen Anwendungen, aber sie laufen innerhalb eines Browser-Prozesses, im Gegensatz zu nativen Anwendungen, die in einem eigenen Prozess laufen. Eine Single-Page-Applikation ist eine vollständige Web-Anwendung, die nur eine Seite hat, die als Shell für alle anderen Anwendungswebseiten verwendet wird und JavaScript, HTML5 und CSS für alle Frontend-Interaktionen nutzt. In Single-Page-Applikationen gibt es keine vollständigen Rückmeldungen an den Server, keine vollständigen Aktualisierungen einer einzelnen Webseite und keine eingebetteten Objekte. Stattdessen wird ein HTML-Element, dessen Inhalt von einer Ansicht zur anderen durch einen Frontend-Routing- und Templating-Mechanismus ersetzt wird, genutzt (Fink & Flatow, 2014).

Single-Page-Applikation bieten einige Vorteile. Sie sind sehr schnell, da die meisten Inhalte nur einmalig geladen werden müssen. Dadurch kommt es auch zu einer Vereinfachung bei der Entwicklung (Mikowski & Powell, 2013). Es besteht keine Notwendigkeit, Codes zu schreiben, um Seiten auf dem Server zu rendern. Single-Page-Applikationen sind leicht zu debuggen, da moderne Web-Browser bereits Tools dafür bieten und die Applikationen somit einfach untersucht werden können. Die Erstellung von mobilen Versionen wird vereinfacht, da der Code für die native mobile Anwendung zum Großteil übernommen werden kann. Single-Page-Applikationen können den lokalen Cache nutzen. Die Anwendung benötigt nur zum Senden und Empfangen von Daten eine aktive Internetverbindung. Danach kann die Anwendung theoretisch auch offline weiter genutzt werden, da die Daten zwischengespeichert werden.

Auf der anderen Seite gibt es auch einige Nachteile, die Single-Page-Applikationen mit sich bringen. Die SEO-Optimierung einer Single-Page-Applikation kann sich kompliziert gestalten. Oftmals sind die Anwendungen beim ersten Laden langsamer, da die Frameworks auf das Endgerät der Benutzerin bzw. des Benutzers geladen werden müssen. Auch erfordern die Applikationen, dass JavaScript im Browser aktiviert ist. Ist dies nicht der Fall, so kann es zu Problemen bei der Ausführung der Anwendung kommen. Hierfür gibt es zwar Lösungen, diese sind aber mit weiterem Aufwand verbunden. Single-Page-Applikationen sind anfälliger für Cross-Site-Scripting-Angriffe und dadurch weniger sicher als andere Anwendungen.

3.3.4 Multi-Page-Applikationen

Multi-Page-Applikationen funktionieren auf traditionelle Art und Weise. Jede Änderung, z. B. das Anzeigen und Senden der Daten an den Server, erfordert das Rendern einer neuen Seite vom Server im Browser. Durch die Mengen an Inhalt sind diese Anwendungen in der Regel größer als Single-Page-Applikationen und sie haben mehrere Ebenen von Benutzeroberflächen. Durch AJAX muss die Anwendung nicht ständig neu geladen werden, sondern es werden nur die benötigten Daten gesendet oder empfangen (III, 2008). Mit der Größe von Multi-Page-Applikationen steigt gleichzeitig auch die Komplexität der Entwicklung dieser Anwendungen.

Ein Vorteil von Multi-Page-Applikationen ist, dass sie sehr gut für die Suchmaschinenoptimierung geeignet sind. Durch die unterschiedlichen Seiten besteht die Möglichkeit, dass jede Seite für ein Keyword optimiert wird. Dadurch ergeben sich bessere Chancen für das Ranking. Außerdem kann die Anwendung durch eine mehrstufige Menüführung übersichtlicher gestaltet werden. Multi-Page-Applikationen haben auch gewisse Nachteile. Meist ist durch die Größe der Anwendung die Entwicklung deutlich komplexer. Entwicklerinnen und Entwickler sind auf Frameworks angewiesen.

3.3.5 Rich-Internet-Applikationen

Rich-Internet-Anwendungen (RIA) sind webbasierte Anwendungen, die einige Eigenschaften von grafischen Desktop-Anwendungen aufweisen. Mit leistungsstarken Entwicklungswerkzeugen erstellt, können Rich-Internet-Anwendungen schneller laufen und ansprechender sein. Sie können der Benutzerin bzw. dem Benutzer ein besseres visuelles Erlebnis und mehr Interaktivität bieten als herkömmliche Browser-Anwendungen, die nur HTML und HTTP verwenden. Entwicklerinnen bzw. Entwickler können fast jede beliebige Funktionalität in eine webbasierte grafische Oberfläche einbetten, so dass diese so aussieht und sich so verhält, als wäre sie eine herkömmliche Desktop-Anwendung. Mit modernen Werkzeugen können Entwicklerinnen und Entwickler komplexe Anwendungsbildschirme mit einer Vielzahl von gemischten Medien wie mehreren Schriftarten, Bitmap- und Vektorgrafikdateien, Animationen, Online-Konferenzen, Audio und Video erstellen. Diese Anwendungen bieten Funktionalitäten, die weit über das bloße Lesen und Browsen hinausgeht, und sie können über das Web bereitgestellt werden. Diese Web-Anwendungen werden Rich-Internet-Anwendungen genannt. Rich-Internet-Anwendungen bieten direkte Interaktion. In einer traditionellen seitenbasierten Web-Anwendung ist die Interaktion auf eine kleine Gruppe von Standard-Steuerelementen beschränkt: Kontrollkästchen, Optionsfelder und Formularfelder. Dies behindert die Erstellung von benutzerfreundlichen und ansprechenden Anwendungen erheblich. Eine RIA kann eine breitere Palette von Steuerelementen verwenden, die eine größere Effizienz ermöglichen und das Benutzererlebnis verbessern. In RIAs können Benutzerinnen bzw. Benutzer zum Beispiel direkt mit Seitenelementen interagieren, indem sie diese bearbeiten oder per ‚drag and drop' verschieben.

HTML-basierte Standard-Webseiten werden einmal geladen. Wenn etwas auf einer Seite aktualisiert wird, muss die Änderung an den Server zurückgeschickt werden, der die Änderungen vornimmt und dann die gesamte Seite erneut sendet (Müller, 2020). Mit HTTP und HTML ist das

nicht anders möglich. Bei herkömmlichen webbasierten Anwendungen müssen Benutzerinnen bzw. Benutzer aufgrund von Netzwerkverbindungsproblemen, Verarbeitungsbeschränkungen und anderen Problemen warten, während die gesamte Seite neu geladen wird. Selbst bei Breitbandverbindungen können die Wartezeiten lang und störend sein. Rich-Internet-Applikationen enthalten jedoch zusätzliche Technologien wie Echtzeit-Streaming, leistungsstarke clientseitige virtuelle Maschinen und lokale Caching-Mechanismen, die die Latenz (Wartezeiten) reduzieren und die Reaktionsfähigkeit erhöhen. Eine Reihe kommerzieller Entwicklungstools ermöglichen diese Teil-Seiten-Aktualisierung.

Aufgrund ihrer Fähigkeit, Teile von Seiten zu verändern, ohne diese neu zu laden, können RIAs der Benutzerin bzw. dem Benutzer ein schnelles und genaues Feedback, eine Echtzeit-Bestätigung von Aktionen und Entscheidungen sowie informative und detaillierte Fehlermeldungen geben. Mit RIA-Tools können die Benutzeroberfläche und die Erfahrung mit verschiedenen Browsern und Betriebssystemen sorgfältiger kontrolliert und konsistent gemacht werden (David, 2013). Wenn keine Internet-Verbindung verfügbar ist, kann eine RIA immer noch verwendet werden, wenn die App so konzipiert ist, dass sie ihren Status lokal auf dem Client-Rechner behält. Abhängig von der Anwendung und den Netzwerkeigenschaften können RIAs oft besser abschneiden als herkömmliche Anwendungen. Insbesondere Anwendungen, die Roundtrips zum Server vermeiden, indem sie lokal auf dem Client verarbeitet werden, sind wesentlich schneller. Das Auslagern solcher Verarbeitungen auf die Client-Rechner kann auch die Serverleistung verbessern. Der Nachteil ist, dass kleine, eingebettete und mobile Geräte, die immer häufiger anzutreffen sind, möglicherweise nicht über die notwendigen Ressourcen verfügen, um solche Anwendungen zu nutzen.

3.3.6 Progressive Web-App (PWA)

Progressive-Web-Apps werden nicht mit einer einzelnen, spezifischen Technologie entwickelt. Es handelt sich weder um ein neues Framework noch um eine neue Sprache. Stattdessen handelt es sich bei Progressive-Web-Apps um eine Reihe von Strategien, Techniken und Application-Programming-Interfaces (APIs), die es Entwicklern ermöglichen, den Nutzerinnen und Nutzern ein natives mobiles Erlebnis zu bieten. Eine progressive Web-Anwendung nutzt die neuesten Technologien, um das Beste aus Web- und mobilen Apps zu kombinieren (Alter, 2017). Die Fortschritte bei modernen Browsern und hinsichtlich der Verfügbarkeit von Services sowie der Cache- und Push-APIs haben es Web-Entwicklerinnen und -Entwicklern ermöglicht, dass Benutzerinnen und Benutzer Web-Apps auf ihrem Startbildschirm installieren, Push-Benachrichtigungen erhalten und sogar offline arbeiten können.

Progressive-Web-Apps sind schnell, oft wird etwas auf dem Gerät der Benutzerin bzw. des Benutzers in weniger als ein paar Sekunden gerendert (Liebel, 2018). Sie sind zuverlässig, auch ohne stabile Datenverbindung und auch auf alten Geräten. Durch die Aktivierung von Benachrichtigungen können Benutzerinnen und Benutzer auf alles aufmerksam gemacht werden, was in ihrer App passiert, auch wenn der Browser nicht geöffnet ist. Eine PWA funktioniert überall. Im Gegensatz zu normalen Web-Apps in Web-Browsern verbessert sich das Nutzererlebnis durch eine Sammlung von Funktionen.

Progressive-Web-Apps bieten auch eine Reihe an Vorteilen gegenüber anderen Web-Apps. Ihre Hauptseite wird auch ohne aktive Internetverbindung geladen. Progressive-Web-Apps sind performanter, da sie z. B. rechenintensive Arbeiten an sogenannte Web-Worker auslagern können. Diese Web-Worker verhindern, dass die Benutzeroberfläche bzw. die auf der Benutzeroberfläche ablaufenden Ereignisse nicht verlangsamt werden. Außerdem können PWAs zum Homescreen hinzugefügt werden. Somit muss die Web-App nicht jedes Mal per URL aufgerufen werden. Ein weiterer Vorteil von PWAs sind die Push-Benachrichtigungen. Eine normale Web-App in einem Browser kann nicht per Push-Benachrichtigung alarmieren (Sheppard, 2017).

3.3.7 Cross-Plattform-Applikation

Bei der Cross-Plattform-Applikationen geht es darum, eine einzige Anwendung zu erstellen, die auf verschiedenen Betriebssystemen ausgeführt werden kann, anstatt verschiedene App-Versionen für jede Plattform zu entwickeln. Die treibende Kraft für die plattformübergreifende Anwendungsentwicklung ist es, Software zu produzieren, die in mehr als einer spezifischen digitalen Umgebung gut funktioniert, mit dem Hauptziel, sie an eine breitere Kundenbasis zu verkaufen (Scott, 2020). Es erfordert den Einsatz eines einzigen Skripts, anstatt separate Skripte für jede Plattform zu schreiben. Dies beschleunigt die Entwicklungszeit erheblich und verkürzt die Zeit bis zur Markteinführung, wovon alle profitieren, vom Entwicklerteam bis hin zum Marketing.

Die plattformübergreifende Anwendungsentwicklung bietet außerdem die Möglichkeit, ein größeres Publikum zu erreichen. Da plattformübergreifende Anwendungen internetbasiert sind, sind Aktualisierungen einfach und bequem. Die Benutzerinnen und Benutzer müssen keine separaten Updates herunterladen, was die Wartung und den Support mehrerer App-Versionen erfordern würde. Die App wird automatisch für alle Kundinnen und Kunden aktualisiert, um sicherzustellen, dass sie immer die aktuelle Version der App haben, was sich positiv auf die Leistung auswirkt. Des Weiteren führen alle oben genannten Vorteile zu erheblichen Kosteneinsparungen. Mit der Entwicklungsgeschwindigkeit, die die plattformübergreifende Entwicklung mit sich bringt, ist die Time-to-Market für jede Plattform kürzer, als wenn jede App von Grund auf neu erstellt werden muss. Das bedeutet, dass mit der Software viel schneller Umsätze generiert werden können. Es ist nicht notwendig, für jede Plattform, auf der die App laufen soll, ein eigenes Software-Entwicklungsteam anzusetzen, was im Laufe der Zeit zu erheblichen Einsparungen führt. Die plattformübergreifende App-Entwicklung ist zu einer wichtigen Alternative zur traditionellen, nativen Entwicklung geworden (Peppers, 2015).

3.4 Cloud-Service-Modelle

Unternehmen stehen vor noch nie dagewesenen Belastungen ihrer IT-Infrastruktur, da sie versuchen, die wachsenden Kundenerwartungen an schnelle, zuverlässige und sichere Dienste zu erfüllen. Während sie daran arbeiten, die Verarbeitungsleistung und die Speicherkapazitäten ihrer IT-Systeme zu erhöhen, stellen diese Unternehmen oft fest, dass die Entwicklung und Wartung einer robusten, skalierbaren und sicheren IT-Infrastruktur teuer ist. Jedoch ist es für

Unternehmen nicht immer zwingend notwendig, eine eigene IT-Infrastruktur zu besitzen und zu betreiben (Kavis M. J., 2014).

Anstatt zusätzliche Hardware zu erwerben, kann ein Unternehmen auf Cloud-Computing zurückgreifen. Cloud-Computing ist ein schnell wachsender Industriezweig, der es Unternehmen ermöglicht, sich von der IT-Infrastruktur vor Ort zu lösen und stattdessen auf internetbasierte Dienste zu setzen. Cloudbasierte Anbieter bieten oft Dienste wie Software, Speicher und Verarbeitung zu erschwinglichen Preisen an. Manchmal kann ein Unternehmen durch die Implementierung einer cloudbasierten Lösung sogar Kosten einsparen.

Cloud-Computing wird in drei verschiedenen Servicemodellen angeboten, die jeweils eine einzigartige Reihe von Geschäftsanforderungen erfüllen. Diese drei Modelle sind bekannt als ‚Software as a Service' (SaaS), ‚Platform as a Service' (PaaS) und ‚Infrastructure as a Service' (IaaS). Die Preismodelle für Cloud-Dienste werden in nutzungsabhängige, abonnementbasierte und hybride Modelle unterteilt, die eine Kombination aus nutzungs- und abonnementabhängigen Preismodellen darstellen (Kavis, 2014).

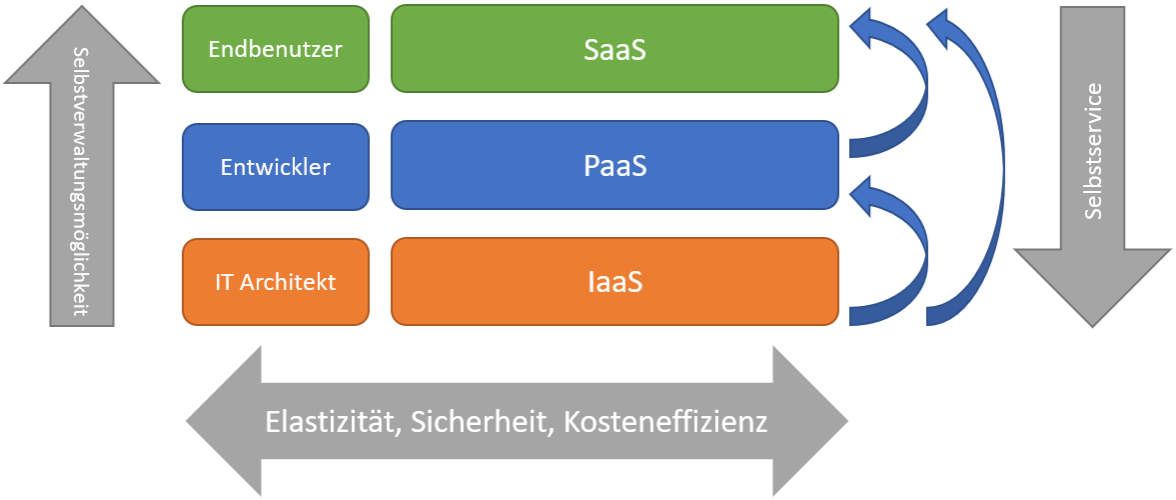

Abbildung 3-2: Überblick von Cloud-Service-Modellen (in Anlehnung an De, 2016)

3.4.1 Infrastructure as a Service

Infrastructure as a Service bietet Anwenderinnen und Anwendern Infrastrukturunterstützung (d. h. Rechenleistung, Speicher, Betriebssystem, Netzwerk) als Service. Das IaaS-Modell ermöglicht es Kundinnen und Kunden, ein neues Projekt schnell zu starten, indem sie Rechenressourcen mieten. Die wichtigsten Merkmale einer IaaS-Cloud sind Skalierbarkeit und Elastizität, so dass die Rechenressourcen nach oben und unten angepasst werden können. Die meisten IaaS-Cloud-Provider bieten Skalierbarkeit unter der Kontrolle der Kundinnen und Kunden mit direkten Self-Service-Schnittstellen, über die Kundinnen und Kunden die Kontrolle und Verwaltung der Rechenleistung sowie die Skalierung der Ressourcen anfordern können (De, 2016).

3.4.2 Platform as a Service

‚Platform as a Service' ist ein Mechanismus zur Kombination von IaaS mit einem abstrahierten Satz von Middleware-Diensten, Software-Entwicklungs- und Bereitstellungs-Tools, die dem Unternehmen eine konsistente Möglichkeit zur Erstellung und Bereitstellung von Anwendungen in einer Cloud oder vor Ort bieten. Ein PaaS bietet einen konsistenten Satz von Programmier- oder Middleware-Services, die sicherstellen, dass Entwicklerinnen und Entwickler eine gut getestete und gut integrierte Methode zur Erstellung von Anwendungen in einer Cloud-Umgebung haben. Eine PaaS-Umgebung bringt Entwicklung und Bereitstellung zusammen, um eine besser verwaltbare Methode zur Erstellung, Bereitstellung und Skalierung von Anwendungen zu schaffen. Ein PaaS erfordert ein IaaS (Hurwitz, Nugent, Halper, & Kaufmann, 2013).

3.4.3 Software as a Service

‚Software as a Service' ist eine Geschäftsanwendung, die von einem Provider in einem mandantenfähigen Modell erstellt und gehostet wird. Mehrmandantenfähigkeit bezieht sich auf die Situation, in der eine einzelne Instanz einer Anwendung in einer Cloud-Umgebung läuft, aber mehrere Kundenorganisationen bedient, die alle ihre Daten getrennt halten. Die Kundinnen bzw. Kunden zahlen für den Service pro Benutzerin bzw. Benutzer entweder über ein monatliches oder ein jährliches Vertragsmodell. Das SaaS-Modell setzt sowohl auf PaaS als auch auf IaaS auf (Hurwitz, Nugent, Halper, & Kaufmann, 2013).

3.4.4 Data as a Service

‚Data as a Service' ist eng mit SaaS verwandt. Es handelt sich dabei um einen plattformunabhängigen Service, der eine Verbindung mit der Cloud herstellt, um Daten zu speichern und abzurufen. Darüber hinaus gibt es eine Reihe von spezialisierten Datendiensten, die in einer Big-Data-Umgebung von großem Nutzen sind. Große Anbieter bieten Dienste, bei denen umfangreiche Datenmengen in kürzester Zeit abgefragt werden können. Abfragen bei sehr großen Datenmengen würden in einem typischen Rechenzentrum zehnmal so lange dauern. Hunderte von spezialisierten Analysediensten wurden von Unternehmen wie IBM, Google und anderen entwickelt (Hurwitz, Nugent, Halper, & Kaufmann, 2013).

3.5 Einsatzgebiete

Bei Web-Anwendungen wird immer versucht, einen möglichst großen Nutzerkreis zu schaffen. Dabei müssen die Entwicklerinnen und Entwickler keine Rücksicht auf die jeweiligen Betriebssysteme der Endgeräte nehmen, da diese in der Regel keine Rolle spielen, weil die Web-Applikation mit dem Browser zugänglich ist. Es gilt, die Anwendung für so viele Benutzerinnen und Benutzer wie nur möglich erreichbar zu machen, unabhängig von Endgerät und Betriebssystem (Grigorik, 2013).

Web-Anwendungen ermöglichen auch schwachen Endgeräten, eine Anwendung zu benutzen, die ihre Hardwareleistung übersteigt. Nachdem die Web-Anwendung auf dem Server bzw. in einer Cloud-Umgebung läuft, dient das Endgerät der Benutzerin bzw. des Benutzers nur mehr als Eingabe- und Ausgabegerät. Die Benutzerin bzw. der Benutzer tätigt am Endgerät Eingaben und bekommt bestimmte Rückmeldungen auf das Endgerät zurück.

Durch moderne Frameworks und leistungsstarke Hardware in fortschrittlichen Cloud-Umgebungen können Web-Anwendungen bereits viele Funktionalitäten von Desktop-Anwendungen übernehmen. Auch hinsichtlich der Performance und der Sicherheit können moderne Web-Anwendungen mit Desktop-Anwendungen konkurrieren. Dabei stellen sie auch viele neue Funktionen zur Verfügung, die Desktop-Anwendungen in dieser Form nicht bieten können.

Entwicklerinnen und Entwickler, die eine neue Anwendung entwickeln möchten, entscheiden sich mittlerweile meist für Web-Anwendungen. Ein Grund dafür ist, dass es weniger Kompatibilitätsprobleme gibt, da die Anwendung nur einmal entwickelt werden muss und nicht auf unterschiedlichen Betriebssystemen läuft. Dadurch wird auch die Wartbarkeit der Anwendung erleichtert, da sonst Änderungen für jedes Betriebssystem einzeln übernommen und getestet werden müssen (Ackermann, 2021).

3.6 Vorteile

Im Vergleich zu Desktop-Anwendungen bieten webbasierte Anwendungen eine ganze Reihe von Vorteilen. Auf diese Anwendungen kann von jedem Computer aus über das Internet zugegriffen werden, statt dass sie auf jedem Computer, von dem aus zugegriffen werden soll, einzeln installiert werden müssen. Die Verwendung webbasierter Software wird allgemein als ‚Software as a Service' bezeichnet, bei der Anwendungen in einer virtuellen, cloudbasierten Umgebung ausgeführt werden.

Geräteübergreifende Zugänglichkeit

SaaS-Anwendungen können auf den meisten Browsern verwendet werden und funktionieren einheitlich auf jedem Betriebssystem, unabhängig von der verwendeten Aktualisierung oder Version. Dies hilft auch bei der Problemlösung, da die Konsistenz für jede Benutzerin bzw. jeden Benutzer gegeben ist. Heute ist es bereits möglich, die meiste Desktop-Software tatsächlich als eine webbasierte Anwendung zu entwickeln, was bedeutet, dass sie von jedem mobilen Gerät genutzt werden kann, ganz gleich wo man sich befindet. Dies gibt der Benutzerin bzw. dem Benutzer die Möglichkeit, zu wählen, wann und wo sie bzw. er die Anwendung nutzt, und fördert das flexible Arbeiten in Unternehmen mit dem Ziel, die Gesamtproduktivität der Mitarbeiterinnen und Mitarbeiter zu erhöhen (Thatcher, et al., 2006).

Einfache Wartbarkeit

Die Bereitstellung von SaaS-Anwendungen ist ein einfacher Prozess, der im gesamten Unternehmen ausgerollt werden kann. Der Zugriff wird gewährt, sobald die Software auf dem Host-Server installiert ist. Jeder Aktualisierung kann über den Host-Server durchgeführt werden,

ohne dass jeder einzelne Rechner aktualisiert werden muss. Dies bedeutet, dass neue Software oder Upgrades einfacher implementiert werden können und die Wartung über einen zentralen Punkt erfolgen kann. Der Zeitaufwand für Änderungen wird reduziert, und das System ist konsistent. Auch kleine Unternehmen verschwenden viel Zeit mit der Aktualisierung von Software. Wenn ein Desktop-Software-Update ausgerollt wird, muss es auf jedem einzelnen Rechner, auf dem sich das Programm befindet, installiert werden. Oft muss dies geschehen, bevor eine weitere Zusammenarbeit möglich ist. Das ist ein entscheidender Grund, warum Entwicklerinnen und Entwickler auf Software für die Entwicklung von Web-Anwendungen umsteigen − sie müssen nur ein einziges Update für die App ausrollen, anstatt es einzeln auf mehreren Geräten zu installieren (Misra & Egoeze, 2014).

Sicherheit

Webbasierte Anwendungen bieten eine gute Möglichkeit, sicher auf zentralisierte Daten zuzugreifen. Auf die Server kann nur die Person oder das Team, die bzw. das sie verwaltet, direkt zugreifen. Durch den Einsatz von Cloud-Computing können Server vollständig redundant und repliziert werden, um Ausfallzeiten infolge einer Katastrophe zu vermeiden. Dadurch entfällt die Notwendigkeit, die Sicherheit jedes Geräts, auf dem die Anwendung genutzt wird, zu gewährleisten. Insgesamt wird das Risiko eines unbefugten Zugriffs reduziert und die Umsetzung von Sicherheitsmaßnahmen ist einfacher, da sie zentral erfolgt.

Mitarbeiterinnen und Mitarbeiter aktualisieren ihre Software oft aus Zeitgründen nicht. Viele Unternehmen sind nicht immer auf dem neuesten Stand, wenn es um ihre Software geht. Dies führt zu Gefahren rund um die Sicherheit der Systeme. Oft werden Updates für Programme veröffentlicht, nur um Sicherheitslücken zu flicken, und nicht nur, um Fehler zu beheben. Das Problem ist: Wenn die Updates nie stattfinden, werden die Lücken nie gepatcht und die Computer sind weiterhin angreifbar (Hoffmann, 2020).

Flexibilität und Skalierbarkeit

Einen der größten Vorteile bei Web-Anwendungen stellen die Flexibilität und die Skalierbarkeit dar. Diese ermöglichen den Unternehmen, auf äußere Einflüsse bzw. Anforderungen zu reagieren und diese schnell umzusetzen. Wenn es z. B. notwendig ist, dass mehr Prozesse gleichzeitig ablaufen, kann die webbasierte Software dies unterstützen. Bei Problemen können die Server vollständig ausgetauscht werden, ohne dass dies Auswirkungen auf das gesamte Betriebssystem hat. Dadurch werden Ausfallzeiten, die sonst auftreten können, verringert. (Henderson, 2006).

Integration von anderen Systemen

Webbasierte Anwendungen bieten eine weitaus größere Möglichkeit zur Integration mit anderen Systemen als Desktop-Anwendungen. Desktop-Software ist isoliert, im Vergleich zu webbasierten Anwendungen, die wesentlich interoperabler sind. Dies liegt daran, dass Web-Anwendungen leichter miteinander verknüpft werden können als zwei völlig getrennte Systeme (O'Brien, Linthicum, & Fuller, 2003).

Anpassung an verschiedene Geräte

Inhalte innerhalb webbasierter Anwendungen können leicht für die Verwendung auf verschiedenen Geräten, z. B. Mobilgeräten, angepasst werden. Dies macht die Anwendung benutzerfreundlich und angenehm zu bedienen, da die Darstellung der Informationen entsprechend angepasst werden kann. Tatsächlich ist es sogar einfacher, webbasierte Anwendungen anzupassen als Desktop-Anwendungen. Webbasierte Software kann auch auf Änderungen je nach Browser reagieren. Dies unterstützt das mobile Arbeiten und stellt sicher, dass die Mitarbeiterinnen und Mitarbeiter die Software immer zur Hand haben, wenn sie sie benötigen.

Wenn es um die Entwicklung von desktopbasierten kundenspezifischen Softwareanwendungen geht, muss auch die Leistung der Maschinen berücksichtigt werden, auf denen sie laufen sollen. Manchmal kann die Einführung eines anspruchsvollen neuen Softwareprogramms auch eine vollständige Überholung der Hardware erfordern. Das ist ein enormer Aufwand. Bei der Verwendung von Web-Apps können diese jedoch von fast jedem Gerät mit einem Browser genutzt werden. Die Rechenleistung kommt vom Server, was die notwendigen Hardwareanforderungen vor Ort reduziert.

3.7 Nachteile

Einschränkungen durch Browser-Technologien

Bei der Entwicklung von Web-Anwendungen stehen einer Entwicklerin bzw. einem Entwickler meist die drei Standard-Programmiersprachen HTML, CSS und JavaScript zur Verfügung. Bei HTML und CSS handelt es sich nicht wirklich um Programmiersprachen, sondern um sogenannte Skriptsprachen. Anhand dieser können Entwicklerinnen und Entwickler ihre Web-Anwendungen aufbauen. Auch wenn JavaScript sich mit der Zeit zu einer umfangreichen und ausgereiften Programmiersprache entwickelt hat, bringt sie Nachteile mit sich. In gewissen Situationen bieten andere Programmiersprachen wie C++ Vorteile wie eine bessere Wartbartbarkeit des Codes.

HTML und CSS sind meist unzureichend, wenn es um reichhaltige Interaktivität geht. Dadurch geraten die Nutzerinnen bzw. Nutzer meist in Abhängigkeit von externen Komponenten wie Frameworks und Plug-ins. So erhöhen sich wiederum die Komplexität und der Supportaufwand für die Web-Anwendung. Dennoch sind Entwicklerinnen und Entwickler auf diese Frameworks angewiesen, da oft nur mit diesen die Einschränkungen von Browser-Technologien umgangen werden können. Je komplexer die Web-Anwendung, umso eher werden Frameworks benötigt.

Abhängigkeit vom Internet

Moderne Web-Anwendungen sind manchmal in der Lage, Operationen ohne aktive Internetverbindung auf dem Client-Gerät auszuführen. Voraussetzung dafür ist, dass die Web-Anwendung zumindest einmal vollständig im Browser geladen sein muss. Danach stehen je nach Web-Anwendung spezielle Funktionen zur Verfügung, die auch ohne eine aktive Internetverbindung ausführbar sind.

Aber auch wenn Web-Anwendungen zum Teil bereits gewisse Tasks ohne Internetverbindung ausführen können, ist früher oder später immer zwingend eine Internetverbindung notwendig. Da die Web-Anwendung zum Großteil auf einem Webserver ausgeführt wird, müssen benötigte Daten bzw. Informationen zwischen dem Server und dem Client ausgetauscht werden. Dies führt zu einer Abhängigkeit von einer konstanten Anbindung an das Internet, um die Web-Anwendung sinnvoll nutzen zu können.

Diese Abhängigkeit gilt auch in Hinsicht zum Webserver bzw. der Webseite. Um die Web-Anwendung nutzen zu können, muss der Webserver bzw. die Webseite erreichbar sein, da die Funktionen dort ausgeführt werden. Fällt der benötigte Webserver aus, so stehen die Services der Web-Anwendung für die Zeit des Ausfalles nicht zur Verfügung.

Leistung

Kleine Anwendungen sind durchaus performant und so gut wie kaum von Desktop-Anwendungen zu unterscheiden. Durch die Übertragungsgeschwindigkeiten des Internets können kleine Mengen an Daten bzw. Informationen schnell an entfernte Webserver übertragen werden. Diese werden dort bearbeitet und die Ergebnisse wieder an den Client zurückgesendet. Da die Webserver auch leistungsstark sein können, gibt es bei den Bearbeitungszeiten zwischen Web-Anwendungen und Desktop-Anwendungen kaum Unterschiede.

Je größer und komplexer Anwendungen werden, umso mehr Rechenleistung wird benötigt. Dies stellt in der Regel kein Problem dar, da Webserver gleich viel und sogar noch viel mehr Leistung haben können als ein Client-Gerät, jedoch wird die Übertragung der Daten meist immer komplexer. Je mehr Daten übertragen werden müssen, umso langsamer wird auch die Verarbeitungszeit von Web-Anwendungen (Matam & Jain, 2017).

Sicherheit

Anwendungen werden immer häufiger von den Entwicklerinnen und Entwicklern über App-Marktplätze zur Verfügung gestellt. Beispiele hierfür sind der Apple App-Store, der Play Store von Google oder der Microsoft Store von Microsoft. Diese Stores helfen nicht nur dabei, den Zugang zu den Applikationen für Benutzerinnen und Benutzer zu erleichtern, sondern sie überwachen auch die Applikationen. Hierbei werden die Applikationen auch auf Sicherheitsrisiken überprüft.

Web-Applikationen werden meist nicht über App-Stores, sondern direkt über die Webserver bzw. Webseiten zur Verfügung gestellt. Dadurch ist eine Überprüfung der Sicherheit der Anwendung aus der Position der Anwenderinnen und Anwender nicht direkt möglich. Web-Anwendungen können dabei Sicherheitslücken aufweisen und diese sind für die Benutzerinnen und Benutzer nicht ersichtlich. Dies ist speziell dann gefährlich, wenn sensible Daten an solche Web-Anwendungen gesendet werden. Benutzerinnen bzw. Benutzer haben in diesen Fällen meist nur die Möglichkeit, der Web-Anwendung zu vertrauen (Stuttard & Pinto, 2011).

3.8 Lizensierung

Durch den Vormarsch von Web-Anwendung haben sich unter anderem vollkommen neue Ertragsmodelle entwickelt. Software-Unternehmen bieten ihre Services in verschiedenen

Varianten für ihre Kunden an. Für die private Nutzung sind Web-Anwendung meist sogar kostenfrei und werden über ein anderes Ertragsmodell finanziert.

Abo-Services

Software-Unternehmen haben damit begonnen, Software nicht nur mehr als Lizenz zu verkaufen. Stattdessen werden Software-Pakete in Form von Abonnement-Modellen angeboten. Damit bieten Software-Unternehmen den Vorteil, dass Kunden die Modelle flexibel an die eigenen Bedürfnisse anpassen können und nur für die Nutzung dieser Funktionen bezahlen müssen. Oftmals handelt es sich bei den Abo-Modellen um eine Kombination aus Desktop- und Web-Anwendung. Als Beispiel kann hier das Microsoft Office 365 Paket erwähnt werden. Darin sind unter anderem Programme wie Word, PowerPoint, Excel und Outlook als Desktop sowie auch als Web-Anwendung enthalten.

Freie Web-Anwendungen

Viele Web-Anwendungen stehen speziell für private Benutzerinnen und Benutzer kostenfrei zur Verfügung. Die Anbieter dieser Web-Anwendungen verfolgen dabei ein Geschäftsmodell, bei dem das Ertragsmodell nicht auf den Verkauf von Lizenzen aufgebaut ist. Meist werden die Entwicklung und der Betrieb durch Werbung oder durch das Sammeln und Verkaufen von Daten finanziert. Beispiele hierfür sind Social-Media-Plattformen wie Facebook und YouTube. Diese Plattformen bieten den Benutzerinnen und Benutzern die Services kostenlos an. Unternehmen können auf den Plattformen Werbung schalten, die den Benutzerinnen und Benutzern angezeigt wird.

3.9 Zusammenfassung

Aufgrund fehlender Standards und Normen stellten Web-Anwendungen früher für Entwicklerinnen und Entwickler eine große Herausforderung dar, die mit hohem Aufwand verbunden war. Durch die Entwicklung des Internets und die ständige Weiterentwicklung und Verbesserung der Web-Technologien und Frameworks wurde es für Entwicklerinnen und Entwickler immer einfacher und attraktiver, Web-Anwendungen zu entwickeln. Heute sind Web-Anwendungen beliebter denn je, sei es bei Benutzerinnen und Benutzern oder den Entwicklerinnen und Entwicklern.

Web-Anwendungen müssen nicht auf einem Endgerät installiert werden, sondern können über den Web-Browser aufgerufen werden. Die Web-Anwendung funktioniert dabei nach dem einfachen Client-Server-Prinzip. Das heißt, das Gerät der Benutzerin bzw. des Benutzers schickt eine Anfrage an den Server, dieser verarbeitet die Anfrage und sendet eine Antwort zurück an das Endgerät. Die Web-Anwendung wird also vollständig am Server betrieben. Das Endgerät dient nur mehr als Ein- und Ausgabegerät.

Durch den Betrieb auf dem Server ist die Web-Anwendung plattformunabhängig. Somit muss sie nur ein einziges Mal entwickelt werden. Eine Anpassung an die unterschiedlichen Betriebssysteme bzw. die Hardware ist nicht notwendig. Außerdem werden moderne Web-Anwendungen im Responsive-Style entwickelt, das heißt, sie passen sich automatisch an die

Bildschirmgröße des Endgeräts an und die Benutzeroberfläche passt sich entsprechend an. Dadurch kann der reibungslose Betrieb auf unterschiedlichsten Endgeräten wie Desktop-PCs, Tablets und Smartphones gewährleistet werden.

Weitere Eigenschaften von Web-Anwendungen sind die Skalierbarkeit sowie die Modularität der Anwendung. Da die Web-Anwendung meist in einer Cloud-Umgebung betrieben wird, ist eine Skalierung der Anwendung einfach möglich. Bei Spitzenlasten können innerhalb kurzer Zeit weitere Server bzw. Ressourcen hinzugeschalten werden. Damit ist ein einwandfreier Betrieb der Anwendung gewährleistet.

Moderne Web-Anwendungen sind modular aufgebaut. Dabei handelt es sich nicht mehr um große Monolithen, sondern um viele einzelne Microservices. Einzelne Funktionen werden durch eigene Microservices realisiert. Dies hat den Vorteil, dass Anpassungen einzelner Funktionen nicht die ganze Anwendung beeinflussen. Des Weiteren vereinfacht sich dadurch das Testen der Web-Anwendung, denn die Komponenten können einzeln getestet werden, bevor sie in das Produktivsystem integriert werden. Somit können Fehler schon im Vorfeld behoben werden und sie führen nicht zu einem fehlerhaften System.

Web-Anwendungen werden oft in Cloud-Umgebungen betrieben. Dabei gibt es verschiedene Cloud-Service-Modelle (Zhang, 2022). Beim SaaS wird eine fertige Anwendung zur Verfügung gestellt. Dies ist vor allem für Benutzerinnen und Benutzer praktisch, denn sie benötigen keine Installation, sondern können die Software einfach nutzen. ‚Platform as a Service' stellt für Entwicklerinnen und Entwickler viele Funktionen für eine einfachere und praktikablere Entwicklung von Web-Anwendungen zur Verfügung. Die Entwicklerinnen und Entwickler können mit den Tools ihre Anwendung entwickeln und auch auf der Plattform testen. ‚Infrastructure as a Service' bietet eine komplette IT-Infrastruktur in Form eines Abo-Modells an. Der Provider kümmert sich dabei um die Wartung der IT-Infrastruktur.

Zum Einsatz kommen Web-Anwendungen unter anderem, wenn ein großer Nutzerkreis erreicht werden soll. Durch die Plattformunabhängigkeit können Benutzerinnen und Benutzer mit den unterschiedlichsten Betriebssystemen auf die Anwendung zugreifen. Dazu wird nur ein Browser und eine aktive Internetverbindung benötigt. Oft werden Web-Anwendungen auch als Alternative zu Desktop-Anwendungen entwickelt, zum Beispiel Office 365. Neue Anwendungen werden häufig nur mehr als Web-Anwendung veröffentlicht, da die Entwicklung für unterschiedliche Betriebssysteme einen deutlichen Mehraufwand und somit auch höhere Kosten verursacht.

Durch den Einsatz von Web-Anwendung entstehen einige Vorteile gegenüber Desktop-Anwendungen. Es kann eine geräteübergreifende Zugänglichkeit für Benutzerinnen und Benutzer geschaffen werden. Web-Anwendungen ermöglichen aufgrund ihrer Modularität eine einfachere Wartung. Im Hinblick auf Datenverlust erhöht sich die Sicherheit, da die Daten meist auf redundanten Servern gespeichert werden. Die Flexibilität und die Skalierbarkeit der Anwendung werden durch den Betrieb in Cloud-Umgebungen erhöht. Auch eine Integration von anderen Systemen per API ist leichter möglich. Jedoch gibt es auch einige Nachteile, z. B. die Einschränkungen durch die bestehenden Browser-Technologien. Zudem besteht immer eine Abhängigkeit vom Internet. Die Sicherheit von Daten kann nicht immer vollständig gewährleistet werden, weshalb eine Speicherung von sensiblen Daten nicht optimal ist.

4 MIGRATIONSSTRATEGIEN

Der Begriff ‚Migrationsstrategien' (auch ‚Migrationsszenarien' oder ‚Anwendungsmigration' genannt) beschreibt den grundsätzlichen Prozess der Verschiebung bzw. der Migration eines Software-Programms von einer Umgebung in eine andere. Ein typisches Migrationsszenario ist z. B. die Migration einer Applikation von einem lokalen Rechenserver in die Cloud. Ein weiteres Beispiel wäre die Migration von einer öffentlichen in eine private Cloud oder umgekehrt (Laszewski & Nauduri, 2012)

Bei der Migration von Anwendungen auf andere Umgebungen gibt es einige Herausforderungen, die während des Prozesses auftreten können. Bei den Anwendungen, die migriert werden sollen, handelt es meist um solche, die speziell für ihre Umgebung entwickelt wurden. Das heißt, die Anwendungen benötigen spezielle Schnittstellen, Netzwerkarchitekturen oder Laufumgebungen. Diese Umgebung muss bei der Migration so gut wie möglich hergestellt werden, damit die Anwendung in der anderen Umgebung laufen kann. Die Migration ist deutlich schwieriger, je stärker eine Anwendung an eine spezielle Umgebung angepasst bzw. dafür entwickelt wurde. Im Gegensatz dazu sind Anwendungen in virtualisierten Umgebungen deutlich leichter zu migrieren (Garverick, 2018).

4.1 Anwendungsmigrationsmuster

Bei der Anwendungsmigration wird eine Gesamtstrategie bestimmt. Dabei werden die jeweiligen Abhängigkeiten und die technischen Anforderungen der zu migrierenden Anwendung analysiert. Der Weg der Anwendung von einer Umgebung in eine andere muss nicht immer gleich verlaufen. Es gibt verschiedene Ansätze, wie Anwendungen erfolgreich migriert werden können. Diese verschiedenen Anwendungsmigrationsmuster werden nachfolgend beschrieben (Blokdyk, 2019).

Rehosting (‚lift and shift')

Rehosting, auch ‚lift and shift' genannt, ist eine gängige Migrationsstrategie. Dabei wird eine Anwendung von einem lokalen Server ohne große Anpassungen auf eine virtuelle Maschine in der Cloud verschoben. Der Vorteil von Rehosting liegt in der Schnelligkeit der Migrationsstrategie. Das reine Verschieben in eine andere Umgebung mit nur geringen Änderungen geht schnell und verursacht nur geringe Kosten. Der Nachteil dabei entsteht dadurch, dass die Anwendung nicht richtig an die Cloud-Umgebung angepasst wird. Eine Anwendung muss an eine Cloud-Umgebung angepasst werden, um von allen Funktionen der Cloud profitieren zu können. Wird diese Anpassung nicht vorgenommen, so ist der Betrieb nicht effizient und es können höhere Betriebskosten anfallen. Rehosting kann zum Großteil mit modernen Tools automatisiert werden. Dennoch sollten Anpassungen an die neue Umgebung manuell vorgenommen werden.

Refactor (Re-Architect)

Oftmals können spezialisierte Anwendungen nicht einfach in eine Cloud-Umgebung geschoben und dort betrieben werden. Bei manchen Anwendungen kann es notwendig sein, größere Änderungen vorzunehmen. Diese zielen meist auf die Funktionalität und die Skalierbarkeit in der Cloud-Umgebung ab, z. B. eine Kodierung von größeren Teilen der Applikation. Damit wird die Anbindung an die Funktionen der Cloud geschaffen. Eine weitere Vorgehensweise ist die Umwandlung von Monolithen in Microservices. Dabei werden Anwendungen in Funktionsbereiche aufgespalten. Bei zukünftigen Änderungen muss nicht die gesamte Anwendung bearbeitet werden, sondern es werden nur die entsprechenden Microservices weiterentwickelt.

Replatform

Um die Anwendung besser an die Architektur der Cloud-Umgebung anpassen zu können, werden beim Replatform-Prozess Änderungen an der Applikation durchgeführt. Es handelt sich also um einen Mittelweg zwischen Rehosting und Refactor. Dabei werden z. B. alte durch moderne cloudbasierte Datenbanksysteme ersetzt. Dies kann die Performance der Anwendung steigern.

Retire/Replace

Wenn die Anwendung bereits alt bzw. der Wert der Anwendung nicht hoch ist, ist es oft nicht sinnvoll, die Anwendung zu migrieren. Dabei übersteigen die Kosten für die Migration oft die Kosten einer Neuanschaffung. Meist gibt es bereits fertige SaaS-Dienste, die dieselben Funktionen wie die alte Software bieten. Zudem sind diese modernen Dienste in der Regel auch schneller und einfacher als die veraltete Anwendung.

Chicken-Little-Strategie

Die Chicken-Little-Strategie zielt darauf ab, ein System vollständig zu verwerfen und ein neues zu entwickeln. Dabei kann eine Anwendung mit den Funktionalitäten des alten Systems ausgestattet werden. Dennoch ist es damit möglich, die Anwendung vollständig an eine neue Umgebung anzupassen. Bei alten Anwendungen kann es vorkommen, dass eine Kompatibilität mit einer neuen Umgebung nicht überall vollständig gewährleistet werden kann.

Database First

Die Datenbank ist meist eine der wertvollsten Komponenten einer Anwendung. In ihr sind alle wesentlichen Informationen gespeichert. Um sicherzustellen, dass keine Daten verloren gehen, wird bei der Database-First-Migration die Datenbank in das neue System migriert. Hierfür gibt es verschiedene Ansätze. Sobald die Kompatibilität der Datenbank mit der neuen Umgebung gegeben ist, werden die weiteren Komponenten der Anwendung migriert.

Database Last

Bei ‚Database Last' erfolgt die Migration der Datenbank zum Schluss. Erst wenn alle Komponenten fertig migriert wurden, kommt die Datenbank dazu. Dies kann z. B. aus Gründen der Datensicherheit erforderlich sein. Wird ein System entwickelt, so gibt es Angriffspunkte, die noch nicht geschlossen wurden. Um das System zu testen, werden Testdaten verwendet. Sobald

das System einsatzbereit ist, werden die Daten aus der alten Datenbank in das neue System migriert. Somit kann die Sicherheit der Daten gewährleistet werden.

Composite-Database-Approach

Hierbei werden das alte und das neue System miteinander verbunden. Die Funktionen werden auf die jeweiligen Systeme verteilt und gewisse Schnittstellen für die Zusammenarbeit geschaffen. Die kommt vor allem bei sehr großen bzw. komplexen Systemen in Frage. Teile des alten Systems können dann Schritt für Schritt am neuen System migriert und getestet werden. Bis zur vollständigen Migration bleibt das alte System produktiv.

4.2 Herangehensweise

Die Frage bei der Migration von Applikationen ist, wann sie erfolgreich verlaufen ist. Sie ist oft nicht einfach zu beantworten. Um eine Migration sinnvoll durchzuführen, sollte diese richtig geplant und konzipiert werden. Ist dies nicht der Fall, kann es während der Umsetzung zu Mehraufwand und somit zu Mehrkosten kommen. Grundsätzlich sollte eine gut geplante Migration den Fokus auf die Anpassung der Anwendung sowie all deren Funktionen und Zugehörigkeiten an das neue System richten und dabei auch alle Anforderungen des neuen Systems abdecken.

Damit Probleme und damit einhergehender Mehraufwand vermieden werden, gibt es spezielle Herangehensweisen, um die Wahrscheinlichkeit einer erfolgreichen Migration zu erhöhen. Ein Punkt für eine erfolgreiche und strukturierte Migration ist die Definition von strategischen Meilensteinen. Für die Migration von Anwendungen sollten die folgenden strategischen Meilensteine angestrebt werden:

- Definition des Anwendungsportfolios
- Sicherheit- und Compliance-Themen definieren
- Bereits bestehende Cloud-Ressourcen auflisten
- Aktuellen Bestand und Zustand der Rechenressourcen auflisten
- Risikoanalyse und Strategie für Risikominimierung entwickeln

Die Migration der Anwendung kann dabei wie ein Projekt geplant werden. Dabei sollten Projektziele, Arbeitspakete sowie die Deadline bestimmt werden. In der Planung sollte speziell das Alter der alten Anwendung berücksichtigt werden. Je nach Alter können die Kosten für eine Migration immer weiter ansteigen, da immer mehr Schnittstellen und Anpassungen benötigt werden. Die Anwendung und alle ihre Komponenten sollten analysiert und dokumentiert werden. Daraus kann eine detaillierte Beschreibung der Anwendung und der benötigten Funktionen generiert werden. Wird die Anwendung nicht detailliert beschrieben, so können während des Migrationsprozesses wichtige Funktionen der Anwendung verloren gehen und die Nachbesserung dieser verlorenen Funktionen führt wiederum zu nachträglichen Kosten.

Des Weiteren sollten die Betriebskosten des gesamten Systems analysiert werden. Dies dient unter anderem dazu, die Kosten für das neue System vergleichbar zu machen. Durch eine genau

Kostenaufstellung können Alternativen leichter verglichen werden. Oft ist es günstiger, eine Anwendung auslaufen zu lassen und auf eine neue modernere Anwendung umzusteigen. Speziell im Hinblick auf die Lebensdauer von Software ist es oft sinnvoller, eine alte Anwendung nicht mehr in ein neues System zu migrieren.

Durch die detailliere Aufzeichnung aller benötigten Punkt kann ein Umsetzungsplan für die Migration erstellt werden. Dieser umfasst alle benötigten Funktionen der alten Anwendung sowie alle Kosten der Migration und des neuen Systems. Der Plan für die Umsetzung sollte auch eine Risikoanalyse sowie einen detaillierten Zeitplan für die geplanten Schritte beinhalten.

5 METHODEN

Die vorliegende Masterarbeit setzt sich aus einem theoretischen und einem empirischen Teil zusammen. Der theoretische Teil besteht aus einer Literaturarbeit. Dabei wurden die Themenpunkte, die die Forschungsfrage umfasst, anhand von Fachliteratur aufgearbeitet und detailliert beschrieben. Die Literaturarbeit ist jedoch nicht ausreichend, um die Forschungsfrage zu beantworten.

Die empirische Arbeit umfasst Experteninterviews, die mit Personen geführt wurden, die auf dem jeweiligen Fachgebiet arbeiten bzw. Erfahrung haben, die qualitative Inhaltsanalyse nach Mayring sowie die abschließende Aufarbeitung und Gegenüberstellung der Erkenntnisse. Anhand der aus den Interviews zusätzlich gewonnenen Daten und Informationen kann detaillierter auf das Thema eingegangen werden. Die Methode der Experteninterviews wurde gewählt, da sich die Forschungsfrage allein auf Grundlage der Fachliteratur nicht umfassend beantworten lässt.

Mit der empirischen Untersuchung wird das Ziel verfolgt, anhand der in den Interviews erhobenen Informationen die aus der Forschungsfrage resultierende Forschungslücke zu schließen. Mit Hilfe der Informationen aus den Experteninterviews konnten neue Einblicke gewonnen und Erhebungen aus der Literaturarbeit untermauert werden. In diesem Kapitel wird das Vorgehen bei den Experteninterviews und bei der Auswertung anhand der qualitativen Inhaltsanalyse nach Mayring erläutert.

5.1 Experteninterviews

Die Entscheidung für die Durchführung von Experteninterviews wurde getroffen, um qualitatives Expertenwissen zu der Thematik sammeln zu können. Die Vorbereitungen für die Experteninterviews und die weiteren Schritte werden nachfolgend beschrieben.

5.1.1 Erstellung des Interviewleitfadens

Für die Durchführung der Interviews ist es notwendig, eine fixierte Struktur festzulegen. Diese Struktur erleichtert anschließend die Aufbereitung und Auswertung der gesammelten Informationen. Anhand der Struktur des Interviewleitfadens kann sichergestellt werden, dass die definierten Fragen bzw. Fragestellungen immer in der vorhergesehenen Reihenfolge gestellt und beantwortet werden. Durch die Beibehaltung der definierten Reihenfolge kann die Kodierung im Anschluss effizienter durchgeführt werden.

Die definierten Fragen bzw. Fragestellungen wurden aus der Forschungsfrage bzw. aus der daraus resultierenden Forschungslücke und aus offenen Themenpunkten der Literaturarbeit abgeleitet. Die Beantwortung dieser Fragen durch Experten stellt sicher, dass die Forschungsfrage umfassend beantwortet werden kann. Zu Beginn des Interviews wurde kurz über die persönliche Erfahrung des Experten gesprochen, um einen Eindruck von der Person zu erhalten.

5.1.2 Erhebung der Daten – Interviewdurchführung

Im Zeitraum von Juli bis August 2021 wurden fünf Experteninterviews geführt und die entsprechenden Informationen gesammelt. Aufgrund der andauernden Corona-Pandemie wurden alle Experteninterviews online mit Hilfe von Microsoft Teams geführt. Die Gespräche wurden mit einem Smartphone aufgenommen. Die jeweiligen Aufnahmen wurden abgespeichert und für die Transkription vorbereitet. Auf Wunsch der Interviewteilnehmer wurden die Namen anonymisiert, gewisse Details wie die Position bzw. die aktuelle Stelle in einem Unternehmen werden nachfolgend zu den Personen beschrieben. Die geführten Interviews dauerten zwischen 15 und 25 Minuten.

Zu Beginn der Interviews gab es eine kurze Vorstellungsrunde und es wurden nochmals die Forschungsarbeit, die Forschungsfrage und das daraus resultierende Ziel der Beantwortung erläutert. Danach wurde auf die Erfahrungen der Person im Hinblick auf das Thema eingegangen, um einen Überblick darüber zu erhalten. Die weiteren Fragen werden nachfolgend aufgelistet.

Im Laufe des Interviews wurden den Experten folgende Fragen in dieser Reihenfolge gestellt:

Einstieg bzw. persönliche Erfahrungen

- Welche Erfahrungen haben Sie mit Desktop-Anwendungen?
- Welche Erfahrungen haben Sie mit Web-Anwendungen?

Desktop-Anwendungen

- Wie würden Sie Desktop-Anwendungen beschreiben?
- Wann würden Sie primär Desktop-Anwendungen einsetzen?
- Welche Vor- und Nachteile sehen Sie bei Desktop-Anwendungen?

Web-Anwendungen

- Wie würden Sie Web-Anwendungen beschreiben?
- Wann würden Sie primär Web-Anwendungen einsetzen?
- Welche Vor- und Nachteile sehen Sie bei Web-Anwendungen?

Migration bzw. Migrationsstrategien

- Welche Erfahrungen haben Sie mit Migrationsstrategien bzw. -szenarien?
- Wann ist eine Migrationsstrategie Ihrer Meinung nach erfolgreich?

Persönliche Meinung zur Forschungsfrage

- Sind Sie der Meinung, dass eine Web-Anwendung eine Desktop-Anwendung vollständig ersetzen kann?
    - Wenn ja, wie kann es funktionieren?
    - Wenn nein, warum nicht?

Die angeführten Fragen bilden die wesentlichen Kernaspekte der Forschungsfrage ab und dienen der Beantwortung der Forschungsfrage. Zusätzlich wurde im Laufe des Interviews noch weiter auf die Antwort des jeweiligen Experten eingegangen bzw. nachgefragt. Dadurch konnten weitere Informationen gewonnen werden.

5.1.3 Interviewpersonen

Für die Durchführung der Interviews wurden Experten kontaktiert, die langjährige Erfahrung auf dem Gebiet der Software-Entwicklung haben. Sie wurden über Kontakte aus dem Studiengang ausfindig gemacht. Die Personen B2 und B5 hatten bereits bei anderen wissenschaftlichen Arbeiten Interviews gegeben und wurden aufgrund der Qualität des Inputs weiterempfohlen. Um den Datenschutz zu gewährleisten und Rückschlüsse auf die Identität zu verhindern, wurden die jeweiligen Personen und die Unternehmen anonymisiert. Die nachfolgende Tabelle bietet eine Liste der interviewten Personen und gibt Aufschluss über ihre Funktion, ihr Unternehmen und die Branche, in der sie arbeiten. Anschließend wird kurz auf die einzelnen Interviewpersonen eingegangen.

ID	Funktion	Branche	Unternehmen
B1	Senior-Software-Entwickler	Sicherheitssoftware	U1
B2	Full-Stack-Software-Engineer	Steuerungs- und Verkehrsleit-Software	U2
B3	Senior-Software-Entwickler	Logistiksoftware	U3
B4	Software-Entwickler	Online-Shops	U4
B5	Lead-Software-Engineer	Online-Patentmanagement-Software	U5

Abbildung 5-1: Übersicht Interviewpersonen

Die Interviewperson B1 ist Senior-Software-Entwickler bei einer Softwarefirma mit Spezialisierung auf Sicherheitssoftware. Sie hat zuvor in der Softwareentwicklung bei einer Firma für Buchhaltungssoftware gearbeitet. Privat beschäftigt sich die Person intensiv mit neuen Cloud- und Web-Technologien.

Die Interviewperson B2 ist Full-Stack-Software-Entwickler bei einer Firma, die Steuerungs- und Verkehrsleit-Software produziert. Sie hat über zwanzig Jahre Erfahrung in der Softwareentwicklung. Die Person ist außerdem selbstständig und berät Unternehmen zu diversen IT-Themen.

Die Interviewperson B3 ist Senior-Software-Entwickler bei einer Logistikfirma. Sie hat zuvor bei einer anderen Logistikfirma gearbeitet, dort primär im Desktop-Bereich. Bei der aktuellen Firma

wird nur im Web-Bereich gearbeitet. Die Person beschäftigt sich des Weiteren im privaten Umfeld mit den neuesten Web-Technologien.

Die Interviewperson B4 ist Software-Entwickler, war zuerst selbstständig im Bereich Web-Anwendungen und danach bei einer Softwarefirma für Online-Shops. Aktuell ist sie wieder im selbstständig und berät und betreut Unternehmen in Hinblick auf Web-Technologien.

Die Interviewperson B5 ist Lead-Software-Engineer bei einer Softwarefirma, die sich auf Online-Patent-Management spezialisiert hat. Sie hat langjährige Erfahrung mit der Entwicklung von Web-Anwendungen.

5.2 Qualitative Inhaltsanalyse nach Mayring

Für die Auswertung der in den Experteninterviews gesammelten Informationen wurde die Methode der qualitativen Inhaltsanalyse nach Mayring gewählt. Die transkribierten Interviews liegen als Basis vor. Mit Hilfe der qualitativen Inhaltsanalyse nach Mayring können diese Texte Schritt für Schritt systematisch bearbeitet werden. Dadurch können wiederum neue Erkenntnisse zu der Thematik gewonnen werden. Ziel der qualitativen Inhaltsanalyse war es, weitere Details, die durch die Literaturarbeit nicht gewonnen werden konnten, zu finden.

5.2.1 Deduktive Kategorienanwendung

Die Analyse wurde anhand der deduktiven Kategorienbildung, die eines von zwei Verfahren darstellt, durchgeführt. Mayring bezeichnete dieses Analyseverfahren unter anderem auch als Strukturierung. Primäres Ziel der Strukturierung ist es, im bereits vorhandenen Datenmaterial eine gewisse Struktur zu erkennen. Dies geschieht, indem anfangs Kategorien gebildet werden, um anschließend das vorliegende Material in Bezug auf die gebildeten Kategorien zu durchsuchen (Mayring, 2015).

Wird ein standardisiertes Erhebungsinstrument wie ein Interviewleitfaden verwendet, dann ist die deduktive Kategorienanwendung bzw. die Strukturierung eine geeignete Methode, denn dadurch ist es möglich, die Kategorien aus den Fragen des Interviewleitfadens abzuleiten. Nachdem für die Experteninterviews ein Interviewleitfaden verwendet wurde, wurden die Kategorien entsprechend der Fragestellungen definiert und für die weitere Durchführung der Inhaltsanalyse herangezogen.

5.2.2 Aufbereitung und Auswertung der Daten

Um die Informationen aus den Interviews aufzubereiten und auszuwerten, mussten die Interviews transkribiert, also in eine schriftliche Form gebracht werden. Alle Interviews mit den Expertenpersonen wurden per Tonaufnahme aufgezeichnet. Diese Tonaufnahmen wurden in die Software MAXQDA, die speziell für qualitative Analysen geeignet ist, importiert und entsprechend den Regeln transkribiert. Dieser Arbeitsschritt war besonders zeitintensiv, dennoch ist er

bedeutsam. Dabei werden Inhalte der Interviews nochmals durchgegangen und verinnerlicht, was für die weitere Auswertung von großem Vorteil ist.

Mit Hilfe des Wissens der Expertenpersonen bzw. der gesammelten Informationen aus den Interviews konnten die Aspekte der Literaturarbeit ergänzt bzw. erweitert werden. Außerdem konnten durch die Informationen auch neue Erkenntnisse gewonnen werden. Des Weiteren wird die Aussagekraft durch die von den Expertenpersonen genannten Anwendungsbeispiele gestärkt.

Im folgenden Abschnitt werden die jeweiligen Schritte bei der Auswertung der Daten näher erläutert. Wie bereits beschrieben wurde als Auswertungsmethode die qualitative Inhaltsanalyse nach Mayring gewählt. Nach dem Ansatz der Strukturierung wird das Material mit Hilfe der Kategorien durchgearbeitet und die Kategorien werden hervorgehoben. Diese markierten Kategorien werden in einem weiteren Schritt paraphrasiert und generalisiert. Durch die Generalisierung können die Informationen anschließend kompakt zusammengefasst werden.

Mayring (2015) definiert in seinem Buch für die Strukturierung ein eigenes Ablaufmodell, nach dem auch in dieser Auswertung vorgegangen wurde. Die Schritte bei diesem Vorgehen werden nachfolgend aufgeführt und genauer erläutert.

1. **Festlegung des Datenmaterials**

   Das Datenmaterial besteht aus den Tonaufnahmen, die während der Interviews erstellt wurden. Daraus resultieren nachfolgend die Transkripte, die zu den einzelnen Interviews angefertigt wurden. Diese Transkripte bilden die Basis für die weitere Untersuchung und stellen somit die Ausgangsdaten für die empirische Arbeit dar.

2. **Analyse der Entstehungsbedingung**

   Das Datenmaterial entstand durch das Führen von Experteninterviews mit den Interviewpersonen, die in Tabelle 5-1 aufgeführt wurden. Diese Interviews wurden mit einem Smartphone aufgenommen. Die Sprachaufnahme wurde anschließend exportiert und am Arbeitsgerät in das Programm MAXQDA importiert. Dort wurde die Audiodatei eigenhändig transkribiert.

3. **Formale Charakteristika des Materials**

   Beim Material handelt es sich wie bereits beschrieben um eine Verschriftlichung der aufgenommenen Interviews mit den Expertenpersonen. Im Zuge der Verschriftlichung wurden keine Änderungen an den Aussagen vorgenommen, allerdings wurden Aussprachefehler und dialektale Ausdrücke korrigiert. Versprecher und Sätze, die begonnen, dann aber neu formuliert wurden, wurden nicht übernommen, da sie keine Aussagekraft haben.

4. **Fragestellung der Analyse**

   Die Definition einer leitenden Fragestellung ist für eine zielgerichtete Inhaltsanalyse essenziell. Die Experteninterviews wurden geführt, um weitere und vor allem themenspezifische Informationen zur Beantwortung der Forschungsfrage zu erhalten. Das primäre Ziel der empirischen Forschungsarbeit ist es, die Forschungslücke mit Hilfe dieser

Informationen zu schließen. Aus diesem Grund orientiert sich die leitende Fragestellung an der Forschungsfrage selbst. Daraus leiten sich die folgenden Untersuchungsobjekte ab:

- Definition und Merkmale von Desktop- und Web-Anwendungen
- Einsatzgebiete von Desktop- und Web-Anwendungen
- Vor- und Nachteile von Desktop- und Web-Anwendungen
- Migrationsthematik
- Expertenmeinung zur Forschungsfrage

Mit Hilfe der Definition und der Nennung von Merkmalen soll es einfacher werden, die beiden Arten von Anwendungen zu unterscheiden und eine klare Trennlinie zu definieren. Durch die Bestimmung der Einsatzgebiete und Beispiele soll dies verdeutlicht werden. Die Aufzählung der Vor- und Nachteile beider Arten soll zur weiteren Klärung beitragen. Sie erlaubt eine Einschätzung darüber, ob ein vollständiger Ersatz durch Web-Anwendungen möglich ist. Durch die Behandlung der Migrationsthematik sollen zentrale Punkte offengelegt und aktuelle Probleme aufgezeigt werden. Schließlich wird die persönliche Meinung jeder Expertenperson einbezogen und analysiert. Anhand dieser Punkte soll die Forschungsfrage umfassend beantwortet werden.

5. Bestimmung der Analysetechnik und Festlegung des Kategoriensystems

Wie bereits erwähnt wurde die deduktive Kategorienanwendung bzw. Strukturierung gewählt, da auf den bereits vorhandenen Interviewleitfaden zurückgegriffen werden konnte, aus dem die Kategorien generiert wurden. Als Analyseverfahren für die Daten wurde die klassische Zusammenfassung eingesetzt.

6. Bestimmung der Analyse-Einheit

Die Bestimmung der Analyse-Einheit dient dazu, die Häufigkeit und die Zeitpunkte für die Einschätzung des Materials zu definieren. Diese Analyse-Einheit besteht aus einer Kodier-Einheit, einer Kontext-Einheit und einer Auswertungs-Einheit. Voll- und Teilaussagen der Expertenpersonen werden als Kodier-Einheit bezeichnet. Die vollständige Verschriftlichung der Expertenperson dient als Kontext-Einheit. Die Analyse-Einheit beschreibt das komplette Material der Auswertung, also alle Verschriftlichungen der geführten Interviews (Mayring, 2015).

Anhand der zuvor definierten Punkte wurde die Analyse des Datenmaterials durchgeführt. Die Dokumentation dazu ist im Anhang zu finden. Des Weiteren werden folgend die Kategorienbildung und die Kodierung beschrieben.

5.2.3 Kategorienbildung

Durch die Anwendung des Verfahrens der deduktiven Kategorienanwendung können die Kategorien wie bereits beschrieben aus dem Interviewleitfaden generiert werden. Im vorliegenden Fall wurden die Kategorien zu den jeweiligen Fragestellungen gebildet. Die aus dem Interviewleitfaden, für die Kodierung abgeleiteten Kategorien werden folgend aufgelistet:

- K1: Thematische Erfahrung Desktop-Anwendungen
- K2: Thematische Erfahrung Web-Anwendungen
- K3: Beschreibung und Merkmale Desktop-Anwendungen
- K4: Einsatzmöglichkeiten Desktop-Anwendungen
- K5: Vorteile Desktop-Anwendungen
- K6: Nachteile Desktop-Anwendungen
- K7: Beschreibung und Merkmale Web-Anwendungen
- K8: Einsatzmöglichkeiten Web-Anwendungen
- K9: Vorteile Web-Anwendungen
- K10: Nachteile Web-Anwendungen
- K11: Thematische Erfahrung Migration bzw. Migrationsstrategien
- K12: Erfolgreiche Migration

5.2.4 Kodierung

Wie bereits für die Transkription wurde auch für die Kodierung die Software MAXQDA verwendet, da diese die Kodierung mittels implementierter Funktionen erleichtert. In der Software werden dazu Textstellen markiert und einer Kategorie zugewiesen. Durch die Markierung der jeweiligen Stellen können relevante Stellen schneller gefunden und weiterverarbeitet werden.

Die Kodierung erfolgte in Anlehnung an die Literatur nach Mayring (2015) bzw. Online-Quellen, die sich auf diese Literatur beziehen. Dazu war es zunächst notwendig, einen Leitfaden, den sogenannten Kodierleitfaden, zu erstellen. Dieser diente als Wegweiser für die grobe Kategorisierung der Textstellen und enthielt in der ersten Fassung die definierten Kategorien, die zuvor aus dem Interviewleitfaden abgeleitet wurden, sowie Definitionen dazu und auch vorläufige Kodierregeln. Die Definitionen dienen dazu, klarzustellen, was eine Kategorie ausmacht. Die Kodierregeln hingegen sollen eine klare Abgrenzung ermöglichen und verhindern, dass unrelevante Textstellen, die nichts mit der Kategorie zu tun haben, einbezogen werden.

Nachdem der erste Entwurf des Kodierleitfadens erstellt war, wurde das Transkript des ersten Interviews anhand dieses Leitfadens durchgegangen und die entsprechenden Textstellen wurden kategorisiert. Im Anschluss wurde mit dem Transkript des zweiten Interviews ebenso verfahren. Nach der Kodierung wurden beide Transkripte nochmals mit dem Kodierleitfaden durchgegangen.

Anschließend wurden die kategorisierten Felder in Hinblick auf den Kodierleitfaden nochmals detailliert betrachtet. Dabei wurden die Kodierregeln einer Prüfung unterzogen, bei der untersucht wurde, ob sie eindeutig definiert wurden, um eine präzise Kodierung zu ermöglichen. Die Kodierregeln wurden während dieses Vorganges entsprechend angepasst bzw. erweitert. Folgend wurden die beiden Transkripte nochmals mit dem angepassten Kodierleitfaden

durchgesehen und entsprechende Stellen angepasst. Zuletzt wurden auch die weiteren Verschriftlichungen der Interviews anhand des angepassten Kodierleitfadens durchgearbeitet und kodiert.

Sobald alle Verschriftlichungen der Interviews kodiert waren, wurde nochmals eine Anpassung am Kodierleitfaden vorgenommen. Um den Kodierleitfaden zu verbessern, bekam jede Kategorie im Leitfaden zwei Ankerbeispiele aus dem zuvor kodierten Transkript zugewiesen. Diese Ergänzung des Kodierleitfadens bringt mehrere Vorteile. Auf der einen Seite kann der Kodierleitfaden besser nachvollzogen bzw. verstanden werden, was auch die Übersichtlichkeit und Schlüssigkeit steigert. Auf der anderen Seite wird es damit auch anderen Personen ermöglicht, den Kodierleitfaden zu verwenden. Zum Schluss wurden die verschriftlichten Interviews noch ein letztes Mal anhand des adaptierten Kodierleitfadens durchgegangen. Durch die Anpassungen konnten nochmals Details bei den kodierten Textstellen angepasst werden und somit konnte die Qualität der Kodierung gesteigert werden.

Auf die Kodierung der Transkripte und die endgültige Erstellung des Kodierleitfadens folgte die Paraphrasierung der kodierten Textstellen. Dies dient unter anderem dazu, die Aussagen und Meinungen der Experten mit eigenen Worten zu formulieren. Die kodierten Textstellen wurden dazu der Reihe nach mit dem Tool der Software MAXQDA paraphrasiert. Die Software zeigt danach die erstellte Paraphrase neben der kodierten Textstelle in Notizform an. Anschließend wurden die Paraphrasen aller Transkripte in eine Excel-Tabelle übertragen.

Als letzter Schritt folgte die Generalisierung. Dabei wurden die einzelnen Paraphrasen, die zuvor gebildet wurden, nochmals durchgegangen und es wurde versucht, nur die wesentlichen Informationen zu behalten. Die Informationen, die keine Relevanz hatten, wurden durch dieses Verfahren herausgefiltert. Somit blieb zum Schluss eine Liste übrig, die die Kernpunkte zu den Kategorien abbildet. Diese Liste ist in Form einer Tabelle im Anhang zu finden.

6 ERGEBNISSE

In diesem Kapitel wird auf die durch die qualitative Inhaltsanalyse nach Mayring gewonnenen Erkenntnisse eingegangen. Die Kategorien wurden anfangs aus den Fragen des Interviewleitfadens abgeleitet. Das Hauptaugenmerk lag auf Merkmalen, die für die Beantwortung der Forschungsfrage ausschlaggebend sind. Dabei handelt es sich um die jeweiligen Beschreibungen von Desktop- sowie Web-Anwendungen und die persönlichen Meinungen der befragten Experten zum Forschungsthema. Anhand der übrigen Merkmale konnte die Forschungsfrage nicht direkt beantwortet werden. Dennoch wurden sie ausgewählt, denn sie helfen dabei, die Forschungslücke umfassender und detaillierter zu betrachten und sie zu schließen. Eine qualitative Ausarbeitung wäre somit ohne diese Merkmale nicht machbar, da diese oft die Basis für den Aufbau der Hauptpunkte bilden. Als Beispiel kann hier die Betrachtung der unterschiedlichen Einsatzgebiete gewählt werden, die dazu dient, die Unterschiede zwischen den beiden Technologien herauszuarbeiten.

Dennoch gab es auch Kategorien, die im Zuge der qualitativen Inhaltsanalyse herausgefiltert wurden. Dabei handelt es sich um die Fragen zu den Erfahrungen der befragten Personen mit den Themen der Desktop-Anwendungen, der Web-Anwendungen und der Migration bzw. Migrationsstrategien. Diese Fragestellungen wurden anfangs gewählt, um ein Bild der jeweiligen Expertenperson bzw. einen Eindruck zu erhalten, inwieweit diese Person mit den Themen vertraut ist. Auch boten diese Fragestellungen einen guten Einstieg in die jeweiligen Interviews bzw. in die Themenblöcke der Interviews. Die Erkenntnisse aus diesen Fragestellungen haben jedoch keine Aussagekraft bzw. Relevanz für die untersuchte Fragestellung und wurden somit aus der weiteren Betrachtung ausgeschlossen.

Die folgende Abbildung zeigt die ausgewählten Kategorien aus dem Interviewleitfaden und die Generalisierungen der Expertenaussagen. Wie bereits beschrieben wurden die Aussagen der Experten paraphrasiert und danach generalisiert. Die Abbildung zeigt die Erkenntnisse zu den jeweiligen Themenpunkten in Form einer Übersicht.

Kategorie	Generalisierung
Beschreibung und Merkmale Desktop-Anwendungen	Native Anwendung mit Hardwarezugriff, die am Desktop läuft; werden lokal installiert und benötigen lokale Hardware-Ressourcen; unabhängig von Internetverbindung; sie wird installiert und man benötigt keinen Browser; direkter Hardwarezugriff für mehr Leistung; Installation notwendig

Einsatzmöglichkeiten Desktop-Anwendungen	Einsatz für Performance bzw. direkten Hardwarezugriff; wenn Performance oder Offline-Funktionalität benötigt; bei eingeschränkter Internetanbindung; hardwareintensive Programme wie CAD; Integration von Desktop-Software
Vorteile Desktop-Anwendungen	Schnelligkeit; Hardwarezugriff; Spezialisierung auf System; perfektes Zusammenspiel von Hardware und Software; alle Features bzw. Funktionen des Geräts verfügbar; Offline-Funktionalität; unabhängige Installation; Performance und keine Ressourcen-Limitierung durch Layer; Anbindung an andere Desktop-Software
Nachteile Desktop-Anwendungen	Meist nur ein aktiver Benutzer; keine Cross-Plattform; Installationen und manuelle Updates; Entwicklungskosten; Berechtigungen für Installation; eingeschränkte Portabilität; Ausrollen mühsam
Beschreibung und Merkmale Web-Anwendungen	Client-Server-Prinzip; Zugriff über Browser; Anwendung in Browser verpackt; über Web-Browser bedient; werden in Browser aufgerufen, müssen aber nicht im Internet verfügbar sein; wird über Netzwerk aufgerufen und lädt beim ersten Aufruf Ressourcen; benötigt Internetverbindung; meist kein direkter Zugriff auf Hardware
Einsatzmöglichkeiten Web-Anwendungen	Zugriff für viele Clients gleichzeitig bereitstellen; große Menge an Usern erreichen; für flexible Anwendungen; einmalige Entwicklung; keine Installation; Cross-Plattform; schnell die breite Masse erreichen; Skalierung

Vorteile Web-Anwendungen	Kollaboration; einfachere Entwicklung; Unterstützung durch Frameworks; keine Installation; großes Publikum; Cross-Plattform; automatische Updates; weniger Ressourcen; einfacheres Deployment; einfachere Skalierung; geringere Entwicklungskosten; Plattform-Unabhängigkeit; sofortige Verwendbarkeit; Portabilität; keine Installation
Nachteile Web-Anwendungen	Performance; Konflikte bei Offline-Änderungen; Internetabhängigkeit; Sicherheit; Einschränkung durch Netzwerk; Akzeptanzproblem bei Web-Browsern; Limitierung durch Browser; Interaktion mit Desktop-Software schwer; Offline-Funktion
Erfolgreiche Migration	Funktionsumfang und Datenformate müssen erhalten bleiben; Benutzer muss glücklich sein; keine Fehler; nach Migration sollte die Anwendung gleich oder besser funktionieren und das Userinterface gleich bleiben; Benutzerkomfort gleich oder größer, wenn altes System abgeschaltet werden kann und keine Abhängigkeiten
Eigene Meinung Forschungsfrage	Ersatz in vielen Fällen möglich, es gibt aber noch Spezialfälle, wo es nicht möglich ist; Technisch möglich, wirtschaftlich nicht immer sinnvoll; Ersatz möglich, alle Technologien vorhanden, Frage nach Technologie; vollständiger Ersatz möglich, spezielle Bereiche bilden Ausnahme; können ersetzen;

Abbildung 6-1: Kategorien und Generalisierungen

Folgend wird detailliert auf die aus der qualitativen Inhaltsanalyse gewonnenen Erkenntnisse eingegangen. Als Basis dienen hierbei die Punkte aus der Generalisierung, die Abbildung 6-1 entnommen werden können. Bei den Generalisierungen handelt es sich um ursprüngliche Expertenaussagen, die in einem weiteren Schritt kategorisiert, danach paraphrasiert und schließlich generalisiert wurden. Die Aussagen aus den Experteninterviews wurden somit auf das Wesentliche reduziert. In diesem Punkt werden die kompakten Aussagen nochmals aufgearbeitet und interpretiert.

Beschreibung und Merkmale Desktop-Anwendungen

Bei einer Desktop-Anwendung handelt es sich um eine Anwendung die nativ, also spezialisiert auf das Betriebssystem, auf einem Desktop-Gerät betrieben wird. Die Anwendung wird entweder eigens für das jeweilige Betriebssystem entwickelt oder nach Möglichkeit an das Betriebssystem angepasst. Bei unterschiedlichen Betriebssystemen wie Windows und MacOS ist eine Anpassung meist nicht möglich, sondern die Anwendung muss eigens für das jeweilige System entwickelt werden, um eine Kompatibilität mit den Systemkomponenten zu gewährleisten. Hauptmerkmal einer Desktop-Anwendung ist, dass sie installiert werden muss.

Desktop-Anwendungen werden auf dem entsprechenden System fest installiert und danach lokal ausgeführt. Sie werden dabei über ein Installationsmedium installiert. Dabei kann es sich um eine CD, einen USB-Stick oder weitere Speichermedien handeln, die die Installationsdateien beinhalten. Heute werden Installationsmedien meist in komprimierter Form über das Internet heruntergeladen.

Für die Ausführung einer Desktop-Anwendung ist in weiterer Folge in der Regel keine Internetverbindung erforderlich. Desktop-Anwendungen sind meist so konzipiert, dass sie ohne Abhängigkeit vom Internet alle Funktionen zur Verfügung stellen können. Jedoch gibt es hierfür Ausnahmen, bei denen Desktop-Anwendungen gewisse Informationen aus dem Internet abfragen. Oft handelt es sich auch um eine Web-Anwendung, die über einen Container betrieben wird und nur den Anschein erweckt, eine Desktop-Anwendung zu sein.

Eine Desktop-Anwendung zeichnet sich des Weiteren durch ihren direkten Hardware-Zugriff auf das System aus. Hierbei ist sie in der Lage, über die Schnittstellen des Betriebssystems auf die Hardwarekomponenten zuzugreifen und diese für die eigenen Funktionalitäten zu verwenden. Ein Beispiel sind Simulationsprogramme, die die Rechenleistungen der Prozessoren und der Grafikkarte nutzen. Durch die Spezialisierung auf ein System und dessen Komponenten kann auch die Effektivität der Anwendung gesteigert werden.

Einsatzmöglichkeiten Desktop-Anwendungen

Die typischen Einsatzgebiete für Desktop-Anwendungen sind jene, bei denen Leistung und Geschwindigkeit, also Performance gefragt ist. Der Vorteil von Leistung und Geschwindigkeit ergibt sich bei Desktop-Anwendungen aufgrund des direkten Hardware-Zugriffs. Eine Anwendung wie ein CAD-Zeichenprogramm, das viel Rechenleistung benötigt, hat somit mehr Ressourcen zur Verfügung und kann Funktionen schneller ausführen. Dies hängt aber auch von der Konfiguration der Hardwarekomponenten selbst ab.

Ein weiteres Einsatzgebiet für Desktop-Anwendungen sind Programme oder Systeme, bei denen keine Internetverbindung vorhanden ist oder sein darf. Desktop-Anwendungen werden so konzipiert, dass sie unabhängig vom Internet arbeiten können. Hierzu zählen z. B. interne Systeme, bei denen aufgrund von Sicherheitsthemen eine Internetanbindung nicht gegeben sein kann.

Ein weiteres Einsatzgebiet von Desktop-Anwendungen ist die Anforderung der Integration bzw. Operationalität mit anderen Desktop-Anwendungen. Durch Schnittstellen des Systems können Desktop-Anwendungen Funktionen anderer Desktop-Anwendungen nutzen. Dies ist für Web-Anwendungen nur mit großem Aufwand erreichbar.

Vorteile von Desktop-Anwendungen

Desktop-Anwendungen bieten unterschiedliche Vorteile. Der direkte Hardwarezugriff, der laut Expertenmeinungen auch einen Vorteil darstellt, generiert weitere Vorteile. So erhöht sich z. B. durch den direkten Hardwarezugriff die Schnelligkeit der Anwendung. Auch können alle Funktionen der Hardwarekomponenten und des Systems verwendet werden. Dies führt zu einer großen Auswahl an Möglichkeiten und dazu, dass Desktop-Anwendungen meist viele, unter anderem auch leistungsstarke Funktionen bieten.

Ein weiterer Vorteil von Desktop-Anwendungen ist die Spezialisierung auf das System bzw. das Betriebssystem. Einerseits wird so ein perfektes Zusammenspiel von Hardware und Software erlangt. Andererseits können alle vom Betriebssystem zur Verfügung gestellten Funktionen genutzt werden. Meist bieten die jeweiligen Betriebssysteme auch eigene Entwicklungsplattformen an, die die Entwicklung der Anwendung für das System zusätzlich erleichtern.

Weitere Vorteile sind die unabhängige Installation und in weiterer Folge die Unabhängigkeit vom Internet bzw. die Offline-Funktionalität. Die Installation erfolgt über ein Installationsmedium, das die benötigten Dateien zur Verfügung stellt. Ist die Desktop-Anwendung installiert, so kann sie in der Regel ohne weitere Abhängigkeiten verwendet werden. Durch die feste Installation und dadurch, dass keine Abhängigkeit vom Internet besteht, kann die Anwendung jederzeit verwendet werden. Außerdem bietet die Offline-Funktionalität weitere Vorteile, z. B. wenn es um die Sicherheit von Daten geht.

Ein weiterer Vorteil ist, dass keine direkte Ressourcenlimitierung durch das System vorhanden ist. Die Desktop-Anwendung kann die Ressourcen des Endgerätes bzw. des Betriebssystems in der Regel uneingeschränkt nutzen. Web-Anwendungen sind hierbei deutlich eingeschränkter. Durch die verschiedenen Layer verlieren sie an Leistung.

Nachteile von Desktop-Anwendungen

Durch die fixe Installation auf einem Desktop-Gerät ist meist nur die gleichzeitige Benutzung durch eine Benutzerin bzw. einen Benutzer möglich. Müssen mehrere Personen dieselbe Anwendung verwenden, so benötigt jede ein Endgerät, auf dem diese Anwendung installiert ist. Handelt es sich dabei um eine leistungsintensive Anwendung, so benötigen alle Personen ein leistungsstarkes Endgerät.

Ein großer Nachteil von Desktop-Anwendungen ist, dass sie nur eine Plattform unterstützen. Somit muss eine Desktop-Anwendung für jede Plattform einzeln entwickelt werden. Dies erfordert wiederum Know-how auf den unterschiedlichen Systemen, wodurch die Entwicklungskosten steigen. Soll eine Anwendung für die gängigen Betriebssysteme bereitgestellt werden, so muss sie für Windows, MacOS und Linux entwickelt werden. Dadurch entsteht ein erheblicher Mehraufwand.

Ein weiterer Nachteil von Desktop-Anwendungen sind die Installation und die manuellen Updates. Für viele Benutzerinnen und Benutzer ist eine Installation mühsam. Auch Updates erfordern manchmal zusätzliche Schritte, die die Benutzerinnen und Benutzer überfordern können.

Nachteilig ist auch die Ausrollung der Software bei Großkunden. Große Unternehmen haben meist Unternehmensrichtlinien für Software und die Geräte werden so weit eingeschränkt, dass nur durch die IT eine Installation von neuer Software oder von Updates möglich ist. Dies ist für Provider von Software oft besonders schwierig, da auf den Endgeräten spezielle Berechtigungen notwendig sind, damit die neue Software oder ein Update ausgerollt werden kann.

Beschreibung und Merkmale Web-Anwendungen

Web-Anwendungen basieren auf dem Client-Server-Prinzip. Dieses ermöglicht es erst, durch die Verbindung von Endgeräten mit Servern verschiedenste Aufgaben in einem Netzwerk zu verteilen. So sendet in der Regel der Client eine Anfrage an den Server. Dieser wiederum bearbeitet die Anfrage oder führt anhand dieser etwas aus und schickt dann eine Rückmeldung an den Client. Dies ist das grundlegende Funktionsprinzip von Web-Anwendungen.

Die Kommunikation findet über das Client-Server-Prinzip statt. Als Netzwerk dient meist das Internet bzw. − wenn keine Internetverbindung vorhanden ist − ein internes Netzwerk, das den Client bzw. die Clients mit einem oder mehreren Servern verbindet. Die Web-Anwendung wird hierfür in den meisten Fällen über einen Web-Browser bedient. Dieser dient als Werkzeug für die Anzeige und die Navigierung innerhalb der Applikation. Für moderne Web-Anwendungen gibt es oft auch Desktop-Clients. Hierbei handelt es sich um Container-Frameworks, die einen Web-Browser imitieren und darin die Web-Anwendung aufrufen. Dies bietet den Vorteil, dass die Web-Anwendung um gewisse Funktionen erweitert werden kann, die in einem Web-Browser nicht unterstützt werden.

Für die Benutzung einer Web-Anwendung ist in der Regel eine fixe Internetverbindung erforderlich. Beim ersten Aufruf der Webseite werden die benötigten Ressourcen vom Server geladen und im Zwischenspeicher des Endgerätes abgespeichert. Danach stehen manchmal gewisse Funktionen zur Verfügung, ohne dass eine Internetverbindung benötigt wird. Sobald aber Eingaben von Benutzerinnen bzw. Benutzern verarbeitet werden sollen, müssen die Befehle bzw. Eingaben an den Server übermittelt werden, der diese dann weiterverarbeitet.

Web-Anwendungen haben aufgrund der Limitierung von Web-Browsern oft keinen direkten Zugriff auf Hardware-Komponenten. Dieser Zugriff wird meist erst durch aufwändige Schnittstellen zu Verfügung gestellt. Jedoch benötigen moderne Web-Anwendungen meist auch keine Hardware-Komponenten des Clients, da sie auf leistungsstarken Servern betrieben

werden. Hauptsächlich werden nur noch Mediageräte eingebunden, z. B. Webcams und Mikrofone.

Einsatzmöglichkeiten Web-Anwendungen

Web-Anwendungen kommen primär zum Einsatz, wenn viele Benutzerinnen und Benutzer gleichzeitig erreicht bzw. die Anwendung von ihnen gleichzeitig verwendet werden soll. Es gilt also, große Mengen an Benutzerinnen und Benutzer zu erreichen. Meist haben diese Anwendungen den Zweck, Inhalte zu verteilen oder die Benutzerinnen und Benutzer zu vernetzen. Beispiele sind Videoplattformen wie YouTube oder Managementsysteme wie SAP. Auf diese Plattformen können beliebig viele Benutzerinnen und Benutzer gleichzeitig zugreifen, vorausgesetzt, dass die Infrastruktur dafür geschaffen ist.

Dies führt zu einem weiteren Einsatzgebiet für Web-Anwendungen. Durch das Hosting in Server- oder Cloudfarmen kann die Infrastruktur, in der eine Web-Anwendung betrieben wird, einfach skaliert werden. Das bedeutet, dass mit wenigen Mausklicks eine Web-Anwendungen zigtausende Userinnen und User mehr bewältigen kann. Die Web-Anwendung selbst ändert sich dabei nicht. Es werden bloß weitere Server zu der vorhandenen Infrastruktur beigeschalten, um die Lasten abdecken zu können.

Ein weiteres Einsatzgebiet für Web-Anwendungen ist, wenn die zu entwickelnde Anwendung nur einmalig erstellt wird und dennoch alle gängigen Geräte und Systeme damit abgedeckt werden sollen. Web-Anwendungen funktionieren über Cross-Plattform, da sie über einen Browser aufgerufen werden. Somit ist keine direkte Installation auf einem spezifischen System notwendig und so gut wie jedes moderne Endgerät bzw. Betriebssystem besitzt zumindest einen vorinstallierten Web-Browser. Daher muss keine weitere Software auf dem Endgerät installiert werden und die Web-Anwendung ist sofort benutzbar.

Vorteile Web-Anwendungen

Web-Anwendungen fördern durch die Vernetzung von Benutzerinnen und Benutzern die Zusammenarbeit in den verschiedensten Bereichen. Die Personen können unterschiedlichste Web-Anwendungen nutzen und dabei mit anderen Leuten zusammen an Projekten arbeiten. Als Beispiel kann hier Office 365 genannt werden, bei dem die Benutzerinnen und Benutzer gemeinsam und auch gleichzeitig an Dokumenten arbeiten können. Dies erleichtert die Zusammenarbeit und ermöglicht sie auch über weitere Entfernungen.

Die Entwicklung von Web-Anwendungen gestaltet sich mittlerweile wesentlich einfacher als diejenige von Desktop-Anwendungen. Für die Entwicklerinnen und Entwickler gibt es verschiedene Entwicklungstools und Frameworks, die ihnen bestimmte Funktionen bereitstellen und somit den Aufwand und die Komplexität der Entwicklung reduzieren. Des Weiteren muss die Web-Anwendung nur einmalig entwickelt werden und ist dann für alle gängigen Betriebssysteme einsatzbereit. Dies reduziert die Kosten der Entwicklung führt zu dem weiteren Vorteil, dass Web-Anwendungen kostengünstiger entwickelt werden können als Desktop-Anwendungen, wenn die Anwendung auf mehreren Systemen laufen soll. Durch die Portabilität kann die Anwendung auf den unterschiedlichsten Geräten und Plattformen genutzt werden.

Ein weiterer Vorteil von Web-Anwendungen ist das einfache Deployment der Anwendung. Im Gegensatz zu Desktop-Anwendungen, die auf jedem Endgerät extra installiert werden müssen, wird die Web-Anwendung nur einmalig auf den Webserver geladen und ist ab diesem Moment auf jedem Endgerät einsatzbereit. Dieser Vorteil gilt auch für die weiteren Updates der Web-Anwendung. Diese werden einfach auf den Server gespielt und stehen somit automatisch für alle zur Verfügung. Somit müssen sich Benutzerinnen und Benutzer keine Gedanken um die Updates machen. Des Weiteren können dadurch auch Sicherheitslücken schnell geschlossen werden und die Datensicherheit der Personen ist gewährleistet.

Von großem Vorteil ist auch die sofortige Verfügbarkeit von Web-Anwendungen für die Benutzerinnen und Benutzer. Hierfür ist keine Installation der Anwendung notwendig, denn diese kann einfach über den Browser aufgerufen werden. Desktop-Anwendungen haben oft große Installationspakete, die zuerst bezogen werden müssen, z. B. per Download über das Internet. Bis die Software einsatzbereit ist, kann einige Zeit vergehen. Dies ist speziell für Unternehmen mit vielen Clients nachteilig, weshalb der Einsatz von Web-Anwendungen einen deutlichen Vorteil bietet.

Die Skalierbarkeit ist ein weiterer großer Vorteil von Web-Anwendungen. Heutige Provider von Servern bzw. Clouds bieten meist ein Web-Interface, bei dem mit wenigen Mausklicks bzw. Eingaben die Rechen- bzw. Speicherressourcen angepasst werden können. Somit können Betreiber von Web-Anwendungen auf spezielle Ereignisse, z. B. die Auslastungsspitzen zu Weihnachten, reagieren. Werden danach wiederum weniger Ressourcen benötigt, können diese auch wieder reduziert werden. Dies bietet große Kostenvorteile durch eine effiziente Nutzung von Ressourcen.

Nachteile Web-Anwendungen

Ein Nachteil von Web-Anwendungen ist die Performance der Anwendung. Dies gilt aber eher für ältere Versionen, denn durch moderne Frameworks und ausgeklügelte Workarounds erreichen Web-Anwendungen heute die Leistung von Desktop-Anwendungen. Auch die Server, auf denen sie betrieben werden, verfügen mittlerweile über enorme Ressourcen. Durch die ständige Weiterentwicklung und Optimierung der Web-Technologien wird sich dieser Nachteil in Zukunft erübrigen.

Ein weiterer Nachteil von Web-Anwendungen sind die einhergehenden Konflikte, die bei der gemeinsamen Benutzung entstehen können. Arbeitet eine Person online an einer Datei und eine andere Person bearbeitet dieselbe Datei offline und lässt diese zu einem späteren Zeitpunkt hochladen, so entsteht ein Versionskonflikt. Dabei blockieren diese Konflikte oft die weitere Synchronisierung von Dateien und stören somit den Arbeitsfluss.

Ein Problem von Web-Anwendungen ist die Abhängigkeit vom Internet bzw. von einem Netzwerk, das Client und Server verbindet. Der Client muss eine Anfrage an den Server schicken, um eine Antwort zu bekommen. Ohne Netzwerk bzw. Internet ist diese Anfrage nicht möglich. Gewisse Web-Anwendungen ermöglichen in eingeschränkter Funktionalität einen Offline-Betrieb, jedoch muss die Anwendung zuvor mindestens einmal initial geladen werden und somit müssen Ressourcen vom Server bezogen werden. Ist keine Internetverbindung vorhanden, so ist die Nutzung der Web-Anwendung nicht möglich. Ein weiteres Problem kann aus einer Einschränkung

des Netzwerkes resultieren. Ist die Netzwerkverbindung zwischen Client und Server eingeschränkt bzw. langsam, leidet auch die Benutzbarkeit der Web-Anwendung darunter. Des Weiteren gibt es spezielle Anwendungsfälle von Web-Anwendungen wie Cloud-Gaming, die ohne eine schnelle Netzwerkverbindung nicht benutzbar sind.

Ein weiterer Nachteil von Web-Anwendungen ist die Datensicherheit. Bei der Benutzung von Web-Anwendungen werden ständig Daten, oft auch sensible, zwischen dem Client und dem Server ausgetauscht und auch am Server gespeichert. Diese Daten können unter anderem beim Transport abgefangen oder vom Server selbst gestohlen werden. Handelt es sich dabei um sensible bzw. geheime Daten, dann kann durch einen Datendiebstahl ein erheblicher Schaden entstehen. Große Firmen bzw. Betreiber von Datencentern sind bemüht, die Sicherheit der Daten zu gewährleisten. Dennoch können nicht immer alle Angriffspunkte vollkommen geschlossen werden und somit bleibt ein Restrisiko für die Benutzerinnen und Benutzer.

Auch Web-Browser können einen Nachteil für Web-Anwendungen darstellen. Es gibt viele Benutzerinnen und Benutzer von veralteten Betriebssystemen und somit auch veralteten Browser-Versionen. Diese unterstützen oftmals nicht die neuen Web-Technologien und schränken die Nutzbarkeit einer modernen Web-Anwendung ein. Dieses Akzeptanzproblem führt auch zu erhöhtem Aufwand bei den Entwicklerinnen und Entwicklern von Web-Anwendungen, die dazu gezwungen sind, die Web-Anwendungen mühsam an die veralteten Browser anzupassen, damit diese dort unterstützt werden. Auch wenn die Browser nicht veraltet sind, haben sie oft dennoch gewisse Limitierungen, wodurch die Benutzbarkeit der Web-Anwendungen beschränkt wird. Nicht alle Web-Browser bieten dieselben Funktionen an. Firefox, Chrome, Safari, Opera und Edge, also die gängigsten modernen Web-Browser, unterscheiden sich hinsichtlich der Vielfalt der Funktionen. Dadurch kann es vorkommen, dass eine Benutzerin bzw. ein Benutzer von Chrome eine andere Nutzererfahrung erlebt als eine Nutzerin bzw. ein Nutzer von Firefox. Dies stellt ein großes Problem für Entwicklerinnen und Entwickler von Web-Anwendungen dar.

Ein letzter Nachteil von Web-Anwendungen ist die eingeschränkte Kompatibilität mit Desktop-Applikationen, was meist auf Layer und die Browser zurückzuführen ist. Um eine Web-Anwendung mit einer Desktop-Anwendung zu verbinden, sind komplexe Schnittstellen notwendig. Dadurch wird in der Regel von einer Fusionierung von Desktop- und Web-Anwendung abgesehen.

Erfolgreiche Migration

Für eine erfolgreiche Migration darf der Funktionsumfang einer Web-Anwendung anders als bei einer Desktop-Anwendung nicht schwinden und auch die typischen Datenformate sollten beibehalten werden. Werden bei der Migration Funktionen gestrichen, so kann es vorkommen, dass die Benutzerin bzw. der Benutzer nicht mehr in der Lage ist, dieselben Schritte mit der Web-Applikation auszuführen, wie es mit der Desktop-Anwendung zuvor möglich war. Dies würde zur Folge haben, dass Personen nicht auf die Web-Anwendung umsteigen, da sie von den Funktionen abhängig sind bzw. diese benötigen.

Die Zufriedenheit der Benutzerin bzw. des Benutzers stellt eine wesentliche Erfolgsvariable für eine Migration dar. Diese sollte immer aus der Sicht der Benutzerin bzw. des Benutzers erfolgen

bzw. deren Lage einbeziehen. Eine Anwendung sollte einen Nutzen für die Userinnen und User stiften und entsprechend an deren Bedürfnisse angepasst werden. Sind diese unzufrieden mit einer neuen Variante, so wird sich auch die Nutzung dieser Anwendung nur schwer durchsetzen.

Die migrierte Anwendung sollte dabei keine Fehler aufweisen. Auftretende Fehler beim Arbeiten mit Anwendungen können für Benutzerinnen und Benutzer frustrierend sein, da meist der Work-Flow darunter leidet und die Person nicht effektiv arbeiten kann. Anwendungen sollten vor der Freigabe speziell auch bei einer Migration in allen möglichen Fällen getestet werden. Sind Funktionen eingeschränkt bzw. treten dabei Fehler auf, so ist die Migration nicht erfolgreich abgeschlossen. Eine Anwendung sollte nach einer erfolgreichen Migration zumindest gleich bzw. besser funktionieren als davor. Dabei sollte sich auch am bekannten User-Interface nicht zu viel ändern, damit die neue Variante schneller angenommen wird. Änderungen hierbei sollten in kleinen Schritten vorgenommen werden. Dies beinhaltet auch die Möglichkeit, ungewünschte Änderungen, die bei den Userinnen und Usern nicht gut angekommen sind, wieder rückgängig zu machen. Auch der Userkomfort sollte nach einer erfolgreichen Migration gleichbleiben bzw. besser werden. Eingabefelder z. B. sollten nicht verschoben oder entfernt werden, sondern logisch, am besten dem Arbeitsprozess entsprechend angepasst werden.

Eine erfolgreiche Migration bedeutet außerdem, dass ein altes System nicht mehr benötigt wird und abgeschaltet werden kann. Solange ein neues System vom alten abhängig ist, ist die Migration nicht erfolgreich abgeschlossen. Systeme, die von alten Systemen abhängig sind, sind der Worst-Case einer Migration.

Eigene Meinung Forschungsfrage

B1 meinte, dass Desktop-Anwendungen in vielen Fällen durch Web-Anwendungen ersetzt werden können. Die Technologien sind vorhanden und es wird immer weiter in diese Richtung gegangen. Es gibt jedoch Spezialfälle, bei denen eine Desktop-Anwendung klare Vorteile bietet, bzw. Einsatzgebiete, bei denen Web-Anwendungen nicht umsetzbar wären.

B2 meinte, dass der Ersatz von Desktop-Anwendungen durch Web-Anwendungen technisch gesehen realisierbar wäre. Wie auch B1 sagte die Person, dass die Technologien bereits vorhanden sind und neue Projekte bereits stark in Richtung Web tendieren. Dennoch ist die Migration von Desktop- auf Web-Anwendungen nicht immer wirtschaftlich sinnvoll. Wenn eine Desktop-Applikation betrieben wird und es keine großen Probleme oder Nachteile gibt, ist eine vorläufige Migration kaum sinnvoll.

B3 meinte, dass Desktop-Anwendungen auf jeden Fall durch Web-Anwendungen ersetzt werden können. Es seien dafür alle notwendigen Technologien vorhanden und die Zukunft der Entwicklerinnen und Entwickler weise immer stärker in Richtung Web-Technologien bzw. Web-Anwendungen. In den meisten Fällen lautet die Frage nicht mehr, ob eine Web-Anwendung entwickelt werden soll, sondern nur, welche Web-Technologie verwendet wird.

B4 war der Meinung, dass ein vollständiger Ersatz noch nicht möglich sei. Als Beispiel wurden typische Desktop-Anwendungen wie CAD-Programme genannt, die aufgrund der Ressourcen am Desktop-PC deutliche Vorteile haben. Dennoch gibt es bereits auch in diesen Bereichen Web-Anwendungen, die zum Großteil dieselben Funktionalitäten bieten.

B5 war der festen Überzeugung, dass ein Ersatz von Desktop-Anwendungen durch Web-Anwendungen auf jeden Fall möglich ist. Die Person begründete ihre Meinung mit den im beruflichen Umfeld gewonnenen Erkenntnissen.

7 DISKUSSION

Im folgenden Kapitel wird noch einmal auf die Erkenntnisse eingegangen, die im vorherigen Kapitel beschrieben wurden. Dabei sollen die zentralen Kategorien einander gegenübergestellt und somit verglichen und interpretiert werden. Es gilt, die Vor- und Nachteile abzuwägen, um die Forschungsfrage beantworten zu können. Um im nächsten und letzten Kapitel ein abschließendes Fazit formulieren zu können, soll anhand der erhobenen Erkenntnisse aus der Literaturarbeit und der empirischen Forschung die Forschungsfrage beantwortet und somit die Forschungslücke geschlossen werden.

Zu Beginn werden in der folgenden Tabelle 7-1 die ausgewählten Kategorien nochmals aufgelistet und es wird angegeben, wie oft diese Punkte angesprochen wurden. Dabei zählt jede Interviewperson nur einmal. Je öfter der Punkt angesprochen wurde, umso relevanter ist dieser auch für die Betrachtung dieses Forschungsthemas.

Kategorie	Aussage
Beschreibung und Merkmale Desktop-Anwendungen	• Native Anwendung, die am Desktop läuft (4/5) • Werden lokal installiert (3/5) • Unabhängig von Internetverbindung (2/5) • Direkter Hardwarezugriff (3/5)
Einsatzmöglichkeiten Desktop-Anwendungen	• Performance und Hardwarezugriff (3/5) • Offline-Funktionalität (2/5) • Integration von Desktop-Software (1/5)
Vorteile Desktop-Anwendungen	• Schnelligkeit/Performance (2/5) • Direkter Hardwarezugriff (1/5) • Spezialisierung auf System (3/5) • Offline-Funktionalität (2/5) • Anbindung an Desktop-Software (1/5)
Nachteile Desktop-Anwendungen	• Gleichzeitige Nutzung (1/5) • Keine Cross-Plattform (3/5) • Installation und manuelle Updates (2/5) • Entwicklungskosten (1/5) • Ausrollen der Software (3/5)

Beschreibung und Merkmale Web-Anwendungen	• Client-Server-Prinzip (1/5) • Zugriff über Browser (5/5) • Aufruf über Netzwerk (3/5) • Kein direkter Hardwarezugriff (1/5)
Einsatzmöglichkeiten Web-Anwendungen	• Gleichzeitige Nutzung (4/5) • Einmalige Entwicklung (1/5) • Keine Installation (1/5) • Cross-Plattform (3/5) • Skalierung (1/5)
Vorteile Web-Anwendungen	• Einfachere Entwicklung (3/5) • Kollaboration (1/5) • Keine Installation (4/5) • Viele User möglich (2/5) • Portabilität (5/5)
Nachteile Web-Anwendungen	• Performance (1/5) • Konflikte (1/5) • Internetabhängigkeit (3/5) • Limitierung durch Browser (3/5) • Sicherheit (1/5)
Erfolgreiche Migration	• Funktionsumfang, Datenformate gleich (2/5) • Glückliche User (1/5) • Keine Fehler (2/5) • Anwendung gleich oder besser (3/5) • Keine Abhängigkeiten vom alten System (1/5)
Eigene Meinung Forschungsfrage	• Möglich (2/5) • Möglich, mit Ausnahmen (3/5) • Nicht möglich (0/5)

Abbildung 7-1: Häufigkeitstabelle

Beschreibung und Merkmale Desktop-Anwendungen

Bei der Umfrage bezeichneten vier der fünf Befragten Desktop-Anwendungen als eine native Anwendung, die am Desktop läuft. Dies stimmt mit den aus der Literaturarbeit gewonnenen Erkenntnissen überein. Eine Desktop-Anwendung läuft somit immer lokal auf einem dafür vorgesehenen Betriebssystem. Wiederum gaben drei von fünf befragten Experten an, dass die Desktop-Anwendungen dafür lokal installiert werden und einen direkten Hardwarezugriff haben. Nur zwei von fünf Experten gingen auf das Thema der Unabhängigkeit vom Internet ein. Der Grund dafür mag sein, dass moderne Desktop-Anwendungen zum Teil auch nicht mehr vollkommen unabhängig vom Internet sind. Sie nutzen auch immer wieder Dienste aus dem Internet oder laden bei der Benutzung weitere Inhalte herunter.

Einsatzmöglichkeiten Desktop-Anwendungen

Bei den Einsatzgebieten für Desktop-Anwendungen sahen drei von fünf Experten Anwendungen, die Leistung und meist damit einhergehend auch direkten Hardwarezugriff benötigen. Die Applikationen binden oftmals auch externe Hardware ein und bieten dafür spezielle Schnittstellen zu den Hardwarekomponenten. Dennoch ist auch hier anzumerken, dass das Einsatzgebiet nicht nur Desktop-Anwendungen umfasst, denn auch Web-Anwendungen steigern die Performance immer weiter. Durch Erweiterungen bei Web-Browsern können auch immer mehr externe Geräte angebunden werden.

Auch hier wurde von zwei Experten das Thema der Offline-Funktionalität angesprochen. Wie bereits unter dem vorherigen Punkt beschrieben sind auch Desktop-Anwendungen nicht immer völlig unabhängig von einer Internet- bzw. Netzwerkverbindung. So kann es vorkommen, dass Desktop-Anwendungen ohne Internetverbindung gewisse Funktionen nicht unterstützen. Ein Beispiel sind hierfür Computerspiele. Diese werden lokal am Desktop-PC installiert und ausgeführt. Sie können ohne aktive Internetverbindung benutzt bzw. gespielt werden, jedoch steht dabei die Mehrspieler-Option nicht zur Verfügung.

Vorteile von Desktop-Anwendungen

Als Vorteile nannten nur zwei von fünf Experten die Schnelligkeit bzw. Performance und nur einer von fünf erwähnte den direkten Hardwarezugriff. Dies steht im Einklang mit der oben aufgestellten Behauptung, dass mittlerweile auch Web-Anwendungen sehr performant sind. Des Weiteren haben auch sie zum Teil Zugriff auf die Hardware. Somit ist dies nicht mehr immer ein direkter Vorteil.

Im Gegensatz dazu sahen drei von fünf Experten die Spezialisierung auf das System als einen Vorteil. Dies stellt im Vergleich zu Web-Anwendungen tatsächlich einen erheblichen Vorteil dar, sofern diese Spezialisierung notwendig ist. Wenn eine Desktop-Anwendung einen ganz speziellen Nutzen erbringen soll und dadurch an bestimmte Parameter angepasst wird, dann ist dies ein großer Vorteil gegenüber Web-Anwendungen, denn diese werden immer so ausgelegt, dass sie möglichst portabel sind und dadurch müssen immer gewisse Abstriche gemacht werden. Handelt es sich um ein einzigartiges System mit speziellen Anforderungen, so kann eine Desktop-Anwendung, die darauf ausgelegt ist, von großem Vorteil sein.

Nachteile Desktop-Anwendungen

Bei den Nachteilen erwähnten drei von fünf Experten, dass Desktop-Anwendungen nicht portabel, also nicht crossplattformfähig sind. Dies ist ein wesentlicher Nachteil, speziell für die Personen bzw. Unternehmen, die diese Anwendung entwickeln. Soll die Anwendung für mehrere Systeme verfügbar sein, so muss sie von den Entwicklerinnen und Entwicklern mehrfach, also für jedes System einzeln, entwickelt bzw. zumindest angepasst werden. Dies ist mit großem Aufwand verbunden und stellt die Personen bzw. Unternehmen vor hohe Entwicklungskosten. Im Gegensatz dazu sind Web-Anwendungen einfacher auf die Masse ausrollbar.

Ein weiterer Punkt, der von drei der fünf Experten genannt wurde, ist das Problem des Ausrollens der Software selbst, z. B. bei der Kundin bzw. beim Kunden. Für die Installation von Desktop-Anwendungen bzw. deren Updates sind meist bestimmte Berechtigungen am Desktop-Gerät notwendig. Speziell große Unternehmen verwalten ihre Geräte in einer Domaine und blockieren die Installation von einzelner Software. Dies macht das Ausrollen von Software in solchen Unternehmen schwierig. Hier ist gute Zusammenarbeit mit der IT-Abteilung der Kundin bzw. des Kunden gefragt.

Zudem wurde angeführt, dass eine gleichzeitige Nutzung von Desktop-Anwendungen durch verschiedene Personen nicht möglich ist. Hierbei muss jedoch auf den Kontext geachtet werden. Eine Web-Anwendung ist auf einem Endgerät auch nicht gleichzeitig für verschiedene Personen nutzbar. Die gleichzeitige Nutzung ergibt sich nur, wenn die Web-Anwendung auf einem Server betrieben wird und sich beliebig viele Nutzerinnen und Nutzer mit ihren Endgeräten verbinden.

Beschreibung und Merkmale Web-Anwendungen

Bei der Beschreibung von Web-Anwendungen wurde von allen fünf interviewten Experten angesprochen, dass der Zugriff über einen Browser verläuft. Ein Experte ging dabei noch auf das Client-Server-Prinzip ein, das bereits in der Literaturarbeit beschrieben wurde. Der Browser dient als Kommunikationstool für den Client mit dem Server, der sich in einem internen Netzwerk bzw. im Internet befindet. Mit Hilfe des Browsers wird eine erste Anfrage an den Server gesendet und dieser gibt eine entsprechende Rückmeldung. Auch dieser Punkt, also der Aufruf über das Netzwerk, wurde von drei Experten angesprochen.

Zudem merkte ein Experte an, dass Web-Anwendungen keinen direkten Hardware-Zugriff haben, wobei dies nicht immer stimmt. Moderne Web-Anwendungen können über Browserschnittstellen auf eine große Anzahl von Hardware bzw. Hardwarekomponenten zugreifen. Durch den Wandel in Richtung Web-Anwendungen bzw. die ständige Weiterentwicklung von Web-Technologien steigt auch die Kompatibilität bzw. die Anzahl der Schnittstellen von Browsern. Somit ist es nur eine Frage der Zeit, bis Web-Anwendungen vollen Zugriff auf Systemkomponenten haben werden.

Einsatzmöglichkeiten Web-Anwendungen

Web-Anwendungen werden vor allem in Bereichen eingesetzt, in denen viele User gleichzeitig dieselbe Anwendung nutzen können sollen. Dies wurde auch von vier Experten angemerkt. Web-Anwendungen werden einmalig am Webserver installiert und konfiguriert und alle Nutzerinnen und Nutzer können die Anwendung ohne Installation am Endgerät nutzen. Dies ist auch für die

Entwicklerinnen und Entwickler der Software von Vorteil, denn so müssen keine Installationsmedien erstellt und verteilt werden. Die Anwendung ist einfach per Internetlink erreichbar und sofort benutzbar. Auch der Aspekt, dass die Web-Anwendung nicht von der Benutzerin bzw. vom Benutzer installiert werden muss, wurde von einem Experten angesprochen.

Ein weiterer Themenpunkt, der von drei Experten genannt wurde, ist, dass Web-Anwendungen über Systeme hinweg kompatibel sind und überall genutzt werden können. Hierbei ist zu beachten, dass manche Systeme Browser zur Verfügung stellen, die teilweise aus Sicherheitsgründen nur wenige Funktionen zur Verfügung stellen. In solchen Fällen kann auch bei Web-Anwendungen eine Nutzung nicht möglich sein. Als Beispiel hierfür dient JavaScript. Manche Browser haben JavaScript deaktiviert, um das automatische Ausführungen von Skripten zu verhindern. Da jedoch die meisten Web-Anwendungen bzw. deren Frameworks auf JavaScript aufbauen, ist es heute fast verpflichtend, dass JavaScript aktiviert ist, damit die Web-Anwendung benutzt werden kann.

Ein Einsatzgebiet, das zwar nur einmal angesprochen wurde, aber dennoch essenziell ist, ist die Skalierung von Anwendungen. Durch den Betrieb von Web-Anwendungen in Clouds oder Serverfarmen kann die IT-Infrastruktur flexibel angepasst und erweitert werden. Steigt die Anzahl der Benutzerinnen und Benutzer stark an, so stellt das für die Betreiberinnen und Betreiber von Web-Anwendungen kein Problem dar. Meist kann mit Hilfe des Web-Interfaces des Hosters schnell eine Erweiterung der Ressourcen durchgeführt werden.

Vorteile Web-Anwendungen

Vier der Experten sahen einen großen Vorteil von Web-Anwendungen darin, dass keine Installation notwendig ist. Dies ist speziell für Benutzerinnen und Benutzer, die nicht computeraffin sind, von großem Vorteil. Zudem stehen die Anwendungen sofort zur Verfügung, ein Download und eine Installation sind nicht notwendig. Das ist praktisch, wenn eine Person sich nur einmal einen Überblick über eine Anwendung verschaffen möchte. Zudem können viele Benutzerinnen und Benutzer gleichzeitig dieselbe Anwendung nutzen. Dieser Punkt wurde von zwei Experten betont.

Einen weiteren Vorteil sahen drei Experten in der einfacheren Entwicklung. Wie bereits beschrieben gibt es zahlreiche Frameworks und Tools, die Entwicklerinnen und Entwickler die Entwicklung von Web-Anwendungen erleichtern. Außerdem müssen die Entwicklerinnen und Entwickler die Web-Anwendung nur einmal erstellen und nicht für jedes System individuell anpassen. Hier kommt wieder das Thema der Portabilität ins Spiel. Auch dieser Aspekt wurde von fünf Experten angesprochen, da er einen großen Vorteil gegenüber Desktop-Anwendungen bietet.

Nachteile Web-Anwendungen

Drei Experten sahen einen Nachteil bei der Limitierung durch den Browser, da dieser die Web-Anwendung einschränken kann. Doch wie bereits beschrieben werden auch die Web-Browser ständig weiterentwickelt und durch neue Web-Technologien gibt es immer mehr Möglichkeiten, auch Web-Anwendungen schneller und leistungsstärker zu machen. Dies steigert auch die

Performance. Zum Thema der Performance meinte nur ein Experte, dass sie ein Nachteil sei, alle anderen Experten sahen darin nicht mehr ein Problem. Der Vergleich von Desktop- und Web-Anwendungen, die in beiden Formen vorhanden sind, etwa bei Office Produkten, zeigt, dass es hier kaum mehr Leistungsunterschiede gibt.

Ein wesentlicher Nachteil, der von drei Experten angesprochen wurde, ist die Abhängigkeit vom Internet. Dies ist eine Schwachstelle von Web-Anwendungen, jedoch ist heute fast immer eine Internetanbindung vorhanden, wodurch auch dieser Nachteil an Bedeutung verliert. Durch den stetigen Ausbau des Glasfaser- und des Funknetzes ist in der Regel eine Internetverbindung gegeben. Außerdem sind die Stellen, an denen es keine Internetverbindung gibt, meist Orte, an denen ohnehin keine Web-Anwendungen genutzt werden.

Erfolgreiche Migration

Beim Thema der erfolgreichen Migration waren alle Experten derselben Meinung. Geht es nach ihnen, müssen auf jeden Fall der Funktionsumfang der Anwendung wie auch das Datenformat bestehen bleiben. Es dürfen keine Funktionen gestrichen werden, da sonst ein Umstieg auf das System erschwert würde. Die Benutzerinnen und Benutzer stehen nach Ansicht der Experten im Vordergrund und sie müssen zufrieden sein, sonst ist eine Migration nicht erfolgreich verlaufen. Die Anwendung sollte somit gleich oder auch besser funktionieren als zuvor und keine Fehler aufweisen. Es sollten auch keine Abhängigkeiten von einem alten System bestehen.

Eigene Meinung Forschungsfrage

Bei der letzten Frage des Interviews wurden die Experten nach ihrer eigenen Meinung gefragt. Sie konnten ihre Einschätzungen zur Forschungsfrage äußern, mussten diese aber auch mit Argumenten begründen. Die Antworten zeigen eine klare Tendenz. So waren alle Experten der Meinung, dass Web-Anwendungen Desktop-Anwendungen ersetzen können.

Zwei der fünf Experten waren der Ansicht, dass Web-Anwendungen mittlerweile alle Desktop-Anwendungen ersetzen können. Sie argumentierten unter anderem, dass moderne Web-Technologien alle Funktionen bieten, die notwendig wären, und dass der Trend immer mehr in Richtung Web geht und es nur eine Frage der Zeit ist, bis alle Anwendungen im Web laufen.

Drei Experten waren der Meinung, dass Web-Anwendungen alle Desktop-Anwendungen ersetzen könnten, es aber Ausnahmen gibt, bei denen dies nicht sinnvoll wäre. Dafür nannten sie z. B. finanzielle Gründe, da eine Anwendung nicht ersetzt werden sollte, wenn sie alle Funktionen erfüllt. Auch bieten Desktop-Anwendungen immer noch gewisse Vorteile und oft ist es sinnvoller, eine Desktop-Anwendung, statt einer Web-Anwendung zu verwenden.

8 FAZIT

Desktop- und Web-Anwendungen haben spezielle Vorteile und Einsatzgebiete. Auch weisen beide Anwendungsarten Nachteile in bestimmten Anwendungsfällen auf. Dennoch ist der Trend in Richtung Web-Anwendungen nicht mehr aufzuhalten, was sich daran zeigt, dass inzwischen beinahe alle neuen Anwendungen als Web-Anwendung entwickelt werden.

Forschungsfrage – Erkenntnisse

Am Beginn der Arbeit wurde folgende Forschungsfrage gestellt:

Über welche Einsatzgebiete erstrecken sich Desktop- und Web-Anwendungen, was unterscheidet sie und inwieweit können Web-Anwendungen Desktop-Anwendungen ersetzen?

Durch die Literaturarbeit und die empirische Forschungsarbeit konnten folgende Erkenntnisse gewonnen werden:

Einsatzgebiete Desktop-Anwendungen

- Für direkten Hardwarezugriff
- Wenn kein Netzwerk vorhanden ist

Einsatzgebiete Web-Anwendungen

- Keine geplante Installation
- Wenn Skalierung notwendig ist
- Für Cross-Plattform
- Wenn viele Nutzerinnen und Nutzer erreicht werden sollen

Unterscheidung von Desktop- und Web-Anwendungen

Desktop-Anwendungen:

- Laufen nativ auf einem Desktop-System
- Müssen installiert werden
- Benötigen in der Regel keine Internetverbindung
- Vollständiger Zugriff auf lokale Ressourcen

Web-Anwendungen:

- Laufen auf einem Webserver oder in der Cloud
- Können sofort benutzt werden
- Benötigen eine Netzwerkverbindung für vollständige Funktionalität
- Bedienung über Web-Browser
- Einmalige Entwicklung deckt alle Systeme ab

Ersatz von Desktop-Anwendungen durch Web-Anwendungen

Es ist möglich, wenn auch nicht immer sinnvoll, Desktop-Anwendungen durch Web-Anwendungen zu ersetzen. Auch wenn Desktop-Anwendungen veraltet sind, haben sie noch ihre Daseinsberechtigung. Es gibt immer wieder Bereiche, bei denen der Einsatz von Desktop-Anwendungen nicht falsch ist, auch wenn eine Web-Anwendung möglich wäre.

Heutige Technologien ermöglichen es Entwicklerinnen und Entwicklern, Web-Anwendungen für alle Einsatzgebiete zu entwickeln. Modernste Frameworks und Browser helfen dabei, die Nachteile von Web-Technologien immer weiter zu reduzieren. Dennoch ist es wirtschaftlich bzw. finanziell oft nicht sinnvoll, eine Web-Anwendung zu entwickeln.

Damit eine Desktop-Anwendung durch eine Web-Anwendung ersetzt werden kann, muss ein Netzwerk vorhanden sein. Darin befinden sich zumindest ein Server und ein Client. Auf dem Server wird die Web-Anwendung betrieben und der Client greift über einen Web-Browser darauf zu. Dabei muss beachtet werden, dass es Programme gibt, die einen Server lokal auf einem Desktop-Gerät betreiben können, auf dem wiederum eine Web-Anwendung ausgeführt werden kann.

Abgesehen von gewissen Offline-Funktionalitäten benötigen Web-Anwendungen fast immer eine Internetverbindung, um alle Funktionen bieten zu können. Hinsichtlich Desktop-Anwendungen ist zu beachten, dass diese in den meisten Fällen auch auf eine Netzwerk- bzw. eine Internetverbindung angewiesen sind, um alle Funktionen zur Verfügung stellen zu können. Eine aktive Internetverbindung ist somit das Hauptkriterium für vollständig einsatzfähige Desktop- und Web-Anwendungen.

Ausblick

Wie zuvor angemerkt sind Web-Anwendungen und teilweise auch Desktop-Anwendungen vom Internet bzw. von einer Netzwerkverbindung abhängig. In einer weiterführenden Arbeit könnte untersucht werden, inwieweit Web-Anwendungen Offline-Funktionen ohne Internet bereitstellen können und welche Einschränkungen sich für moderne Desktop-Anwendungen ergeben, wenn diese ebenfalls keine Internetverbindung zur Verfügung haben.

ANHANG A - Qualitative Inhaltsanalyse

NR	IP	Paraphrase	Generalisierung	Kategorie Nr.	Kategorie	Reduktion
1	B1	Erfahrungen mit Desktop-Anwendungen eher als Benutzer anstatt als Entwickler.	Erfahrung als Benutzer	K1	Thematische Erfahrung Desktop-Anwendungen	K1, K2, K11 nicht relevant.
2	B1	Früher Cloud-Anwendungen für kleine Unternehmen für buchhalterische Zwecke. Aktuell Dienstplanungssoftware für Krankenhäuser, Berufsfeuerwehren und ähnliche Sachen.	Cloud-Anwendungen, Dienstplanungssoftware	K2	Thematische Erfahrung Web-Anwendungen	
3	B1	Die Desktop-Anwendung ist eine Anwendung, die am Desktop läuft. Es handelt sich um eine native App die direkten Zugriff auf Hardware hat. Sie muss nicht auf einem Server laufen.	Native Anwendung mit Hardwarezugriff die am Desktop läuft.	K3	Beschreibung und Merkmale Desktop-Anwendungen	
4	B1	Ein Einsatz ist sinnvoll, wenn Performance benötigt wird oder wenn die Hardware direkt angesprochen werden muss. Beispiel ist eine grafische Anwendung die Zugriff auf die Grafikkarte benötigt.	Einsatz für Performance bzw. direkten Hardwarezugriff	K4	Einsatzmöglichkeiten Desktop-Anwendungen	
5	B1	Desktop-Anwendungen sind in der Regel schneller. Sie haben die Möglichkeit, direkt auf die Hardware zuzugreifen. Durch die Spezialisierung auf ein System kann die Entwicklung einfacher sein.	Schnelligkeit, Hardwarezugriff, Spezialisierung auf System	K5	Vorteile Desktop-Anwendungen	
6	B1	Desktop-Anwendungen sind von Nachteil, wenn man viele Benutzer hat, die gleichzeitig etwas ausführen wollen.	Meist nur ein aktiver Benutzer.	K6	Nachteile Desktop-Anwendungen	

7	B1	Eine Web-Anwendung basiert auf dem Client-Server-Prinzip, das heißt, dass ein Client Zugriff auf einen Server hat, bzw. von einem Server Daten abfragt. Der Client steigt üblicherweise über einen Web-Browser ein. Eine Web-Anwendung sieht aus wie eine native Anwendung, ist aber in einem Web-Browser verpackt.	Client-Server-Prinzip, Zugriff über Browser, Anwendung in Browser verpackt	K7	Beschreibung Web-Anwendungen	
8	B1	Web-Anwendungen kommen zum Einsatz, wenn viele Benutzer gleichzeitig auf die Funktionen zugreifen können sollen.	Zugriff für viele Clients gleichzeitig bereitstellen.	K8	Einsatzmöglichkeiten Web-Anwendungen	
9	B1	Mehrere Menschen können mit Web-Anwendungen zusammenarbeiten. Die Entwicklung von Web-Anwendungen ist für Entwickler einfacher, da sie mit Frameworks arbeiten können und die Anwendung nur für eine Plattform entwickeln müssen.	Kollaboration, einfachere Entwicklung, Unterstützung durch Frameworks	K9	Vorteile Web-Anwendungen	
10	B1	Ein Nachteil bei Web-Anwendungen ist eigentlich nur die Performance, die aber auch immer besser wird. Ein weiteres Problem sind die Konflikte, wenn offline Änderungen vorgenommen und diese dann hochgeladen werden.	Performance, Konflikte bei offline Änderungen	K10	Nachteile Web-Anwendungen	
11	B1	Selbst keine Migration durchgeführt, aber als Benutzer an Beispielen wie Office miterlebt.	keine Migration durchgeführt	K11	Thematische Erfahrung Migration und Migrationsstrategien	
12	B1	Für eine Erfolgreiche Migration muss der Funktionsumfang erhalten bleiben, wenn nicht sogar wachsen. Datenformate sollten beibehalten werden.	Funktionsumfang und Datenformate müssen erhalten bleiben.	K12	Erfolgreiche Migration	

13	B1	In vielen Szenarien können Web-Anwendungen Desktop-Anwendungen ersetzen, aber nicht überall. Desktop-Anwendungen können in speziellen Fällen nicht ersetzt werden, bzw. ist es finanziell gesehen nicht sinnvoll.	Ersatz in vielen Fällen möglich, es gibt aber noch Spezialfälle, wo es nicht möglich ist.	K13	Eigene Meinung Forschungsfrage	
14	B2	Über 25 Jahre Erfahrung mit Desktop-Anwendungen und diversen zugehörigen Technologien.	25 Jahre Erfahrung	K1	Thematische Erfahrung Desktop-Anwendungen	K1, K2, K11 nicht relevant.
15	B2	Ungefähr gleich viel Erfahrung wie mit Desktop-Anwendungen.	sehr viel Erfahrung	K2	Thematische Erfahrung Web-Anwendungen	
16	B2	Desktop-Anwendungen werden lokal installiert und benötigen lokale Hardwareressourcen. Desktop-Anwendungen können unabhängig von der Internetverbindung gestartet werden.	Werden lokal installiert und benötigen lokale Hardware-Ressourcen. Unabhängig von Internetverbindung	K3	Beschreibung und Merkmale Desktop-Anwendungen	
17	B2	Einsatz von Desktop-Anwendungen, wenn Performance oder eine Offlinefunktionalität benötigt wird bzw. bei Interop Szenarien.	Wenn Performance oder Offline-Funktionalität benötigt.	K4	Einsatzmöglichkeiten Desktop-Anwendungen	
18	B2	Vorteile von Desktop-Anwendungen sind das perfekte Zusammenspiel von Hardware und Software, dass alle Features bzw. Funktionen vom Gerät bzw. vom Betriebssystem genutzt werden können und die Offline Funktionalität	perfektes Zusammenspiel von Hardware und Software, alle Features bzw. Funktionen des Geräts verfügbar, Offline-Funktionalität	K5	Vorteile Desktop-Anwendungen	
19	B2	Ein Nachteil von Desktop-Anwendungen ist, dass sie nicht Cross-Plattform funktionieren. Ein weiterer Nachteil sind die Installation und die manuellen Updates. Auch die Entwicklungskosten sind ein Nachteil.	kein Cross-Plattform, Installationen und manuelle Updates, Entwicklungskosten	K6	Nachteile Desktop-Anwendungen	

20	B2	Eine Web-Anwendung ist eine Anwendung, die über einen Browser bedient wird. Dabei werden Dateien in der Regel nicht am lokalen Endgerät gespeichert. Eine Web-Anwendung kann typischerweise nur mit einer Internetverbindung bedient werden.	über Web-Browser bedient, benötigt Internetverbindung	K7	Beschreibung Web-Anwendungen	
21	B2	Web-Anwendungen werden eingesetzt, wenn eine große Menge von Userinnen und Usern erreicht werden soll.	große Menge an User erreichen	K8	Einsatzmöglichkeiten Web-Anwendungen	
22	B2	Vorteile von Web-Anwendungen sind, dass keine Installation notwendig ist, dass ein großes Publikum angesprochen werden kann, dass die Anwendung auf den meisten Plattformen funktioniert und dass Updates automatisch passieren. Weitere Vorteile sind, dass weniger Ressourcen benötigt werden, das Deployment ist einfacher, die Skalierung ist einfacher, die Entwicklungskosten sind geringer und Frameworks unterstützen bei der Entwicklung.	keine Installation, großes Publikum, Cross-Plattform, automatische Updates, weniger Ressourcen, einfacheres Deployment, einfachere Skalierung, geringere Entwicklungskosten, Frameworks	K9	Vorteile Web-Anwendungen	
23	B2	Ein Nachteil von Web-Anwendungen ist, dass man online sein muss. Auch das Thema Security stellt einen Nachteil dar, da es nicht kontrollierbar ist, wo die Daten liegen.	Internet-Abhängigkeit, Sicherheit	K10	Nachteile Web-Anwendungen	
24	B2	Die meiste Erfahrung mit Migration von Desktop- auf Web-Anwendungen, aber auch von Web- auf Desktop-Anwendungen.	~~Erfahrung Desktop auf Web~~	K11	Thematische Erfahrung Migration und Migrationsstrategien	

25	B2	Der wichtigste Punkt für eine erfolgreiche Migration ist die Benutzerin bzw. der Benutzer, welcher vor dem Bildschirm sitzt. Wenn sich diese ärgern, dann ist die Migration nicht gut verlaufen. Es gibt sehr wohl Migrationen, bei denen es schlechter wurde. Für eine erfolgreiche Migration sollten sich keine Fehler einschleichen. Migrationsszenarien können über viele Jahre laufen.	Benutzer muss glücklich sein. Keine Fehler.	K12	Erfolgreiche Migration	
26	B2	Durch die Menge an Legacy Code, welcher noch im Umlauf ist, ist es eher unwahrscheinlich, dass Web-Anwendungen alle Desktop-Anwendungen ersetzen, da der Migrationsaufwand einengen würde. Es hängt eher nicht von technischen Themen ab, sondern von finanziellen Aufwänden. Speziell ältere Projekte sind nur sehr schwer zu migrieren, da sie Algorithmen enthalten, die keiner kennt, was zu enormen Aufwänden führen würde.	Technisch möglich, wirtschaftlich nicht immer sinnvoll.	K13	Eigene Meinung Forschungsfrage	
27	B3	Im ersten Unternehmen hauptsächlich Desktop-Anwendungen, für Lagerlogistik entwickelt.	Erfahrung mit Desktop durch Arbeit	K1	Thematische Erfahrung Desktop-Anwendungen	K1, K2, K11 nicht relevant.
28	B3	Erfahrung mit Web-Anwendungen aus privatem Interesse und später auch primär im Beruf eingesetzt.	Erfahrung durch privates Interesse und Beruf	K2	Thematische Erfahrung Web-Anwendungen	
29	B3	Eine Desktop-Anwendungen läuft auf einem Desktop-System und somit nativ. Man benötigt keinen Browser dafür. Es handelt sich meist eindeutig um eine Desktop-Anwendung, wenn man die Anwendung am Client installieren muss.	Läuft nativ auf einem Desktop-System. Sie wird installiert und man benötigt keinen Browser.	K3	Beschreibung und Merkmale Desktop-Anwendungen	

30	B3	Desktop-Anwendungen werden hauptsächlich bei Stationen mit eingeschränkter Internetanbindung eingesetzt.	bei eingeschränkter Internetanbindung	K4	Einsatzmöglichkeiten Desktop-Anwendungen	
31	B3	Ein Vorteil von Desktop-Anwendungen ist, dass man diese allein auf dem Endgerät installieren kann. Die Anwendung ist dadurch unabhängig von einem Browser. Das Installationsmedium kann einfach weitergegeben werden.	Unabhängige Installation	K5	Vorteile Desktop-Anwendungen	
32	B3	Ein Nachteil von Desktop-Anwendungen ist, dass sie nicht Plattform unabhängig sind, was bedeutet, dass die Anwendung für jede Plattform einzeln entwickelt bzw. angepasst werden muss. Auch die Installation stellt einen Nachteil dar. Ein weiterer Nachteil sind die benötigten Berechtigungen für Installationen und Updates.	kein Cross-Plattform, Berechtigungen für Installation	K6	Nachteile Desktop-Anwendungen	
33	B3	Web-Anwendungen sind Anwendungen, welche im Browser aufgerufen werden, weshalb ein Browser benötigt wird. Nur weil Web-Anwendungen über einen Browser aufgerufen werden, bedeutet das nicht, dass sie im Internet verfügbar sind. Die Anwendungen können auch in einem internen Netz zur Verfügung gestellt werden. Web-Anwendungen haben meist keinen direkten Zugriff auf die Hardware der Endgeräte, was für Probleme sorgt.	Werden in Browser aufgerufen, müssen aber nicht im Internet verfügbar sein. Meist kein direkter Zugriff auf Hardware.	K7	Beschreibung Web-Anwendungen	
34	B3	Web-Anwendungen kommen zum Einsatz, wenn flexible Anwendungen zur Verfügung gestellt werden sollen.	Für flexible Anwendungen.	K8	Einsatzmöglichkeiten Web-Anwendungen	

35	B3	Vorteile von Web-Anwendungen sind die Plattform Unabhängigkeit und dass die Web-Anwendungen sofort von Endgeräten verwendet werden können.	Plattform Unabhängigkeit, sofortige Verwendbarkeit	K9	Vorteile Web-Anwendungen	
36	B3	Netzwerkprobleme erschweren es, Web-Anwendungen bzw. Cloud-Lösungen zur Verfügung zu stellen. Ein weiterer Nachteil von Web-Anwendungen ist das Akzeptanzproblem rund um Web-Browser. Viele Leute halten noch an alten Browsern fest, welche die neuen Web-Technologien nicht unterstützen.	Einschränkung durch Netzwerk, Akzeptanzproblem bei Web-Browsern.	K10	Nachteile Web-Anwendungen	
37	B3	Erfahrung mit verschiedenen Migrationsstrategien. Einmal von neu angefangen, um Altlasten loszuwerden. In anderem Szenario wurde versucht, eine Desktop-Applikation Schritt für Schritt ins Web zu migrieren. Erfahrungen mit Big-Bang, also dem Ansatz alles neu zu machen, da alte Technologien komplett ersetzt wurden. Weitere Erfahrung mit schrittweiser Migration von Desktop-Anwendungen ins Web.	Erfahrung mit unterschiedlichen Strategien.	K11	Thematische Erfahrung Migration und Migrationsstrategien	
38	B3	Bei einer erfolgreichen Migration sollte die Anwendung, so wie sie vorher als Desktop-Anwendung funktioniert und existiert hat, danach als Web-Anwendung gleich oder noch besser funktioniert. Das Userinterface und der Funktionsumfang sollten möglichst gleich sein.	Nach Migration sollte die Anwendung gleich oder besser funktionieren, sowie das Userinterface gleichbleiben.	K12	Erfolgreiche Migration	

39	B3	Es ist möglich Desktop-Anwendungen vollständig durch Web-Anwendungen zu ersetzen, da mittlerweile alles vorhanden ist, was benötigt wird. Web-Anwendungen müssen nicht zwingend im WWW liegen. Mit Web-Anwendungen kann man heute so gut wie alles umsetzten und es gibt immer weniger Probleme. Der Trend geht immer weiter in die Richtung. Bei der Entwicklung von neuen Applikationen ist die Frage nicht mehr ob, Web- oder Desktop-Anwendung, sondern welche Web-Technologie man verwendet.	Ersatz möglich, alle Technologien vorhanden, Frage nach Technologie	K13	Eigene Meinung Forschungsfrage	
40	B4	Erfahrung mit Desktop-Anwendungen als Entwickler sowie auch als Anwender.	als Entwickler und Anwender	K1	Thematische Erfahrung Desktop-Anwendungen	K1, K2, K11 nicht relevant.
41	B4	Erfahrung mit Web-Anwendungen durch die Arbeit in der Online-Shop-Entwicklung und als selbstständiger Entwickler.	Durch Arbeit uns Selbstständigkeit	K2	Thematische Erfahrung Web-Anwendungen	
42	B4	Desktop-Anwendungen sind Programme, die den Zweck verfolgen, einen direkten Hardware-Zugriff zu haben. Wenn man mehr Leistung benötigt, dann sind Desktop-Anwendungen von Vorteil.	Direkter Hardwarezugriff für mehr Leistung.	K3	Beschreibung und Merkmale Desktop-Anwendungen	
43	B4	Desktop-Anwendungen haben ihre Einsatzgebiete in Bereichen, wo direkte Hardwarezugriffe gefragt sind, wie z.B. bei CAD-Zeichenprogrammen. Diese Programme sind speziell optimiert und laufen am Desktop besser als im Web.	hardwareintensive Programme wie CAD	K4	Einsatzmöglichkeiten Desktop-Anwendungen	
44	B4	Vorteile von Desktop-Anwendungen sind die Performance und dass es keine Ressourcenlimitierung durch einen Layer gibt.	Performance und keine Ressourcen-Limitierung durch Layer	K5	Vorteile Desktop-Anwendungen	

45	B4	Nachteile von Desktop-Anwendungen sind die eingeschränkte Portabilität, wodurch man die Software auf unterschiedliche Systeme hin anpassen muss.	eingeschränkte Portabilität	K6	Nachteile Desktop-Anwendungen	
46	B4	Eine Web-Anwendung ist eine Anwendung, die über das Internet aufgerufen wird und welche beim ersten Aufruf Ressourcen über das Netzwerk lädt. Diese Anwendung muss nicht installiert werden, sie kann einfach aufgerufen werden.	wird über Netzwerk aufgerufen und lädt beim ersten Aufruf Ressourcen	K7	Beschreibung Web-Anwendungen	
47	B4	Eine Web-Anwendung wird eingesetzt, wenn die Anwendung nur einmal entwickelt werden soll und wenn eine Installation keinen Vorteil bringt. Dadurch kann man mit einer Anwendung mehrerer Plattformen abdecken.	einmalige Entwicklung, keine Installation, Cross Plattform	K8	Einsatzmöglichkeiten Web-Anwendungen	
48	B4	Vorteil von einer Web-Anwendung ist die Portabilität, also dass mit einem Entwicklungszweig viele Endgeräte abgedeckt werden können. Ein weiterer Vorteil ist, dass keine Installation notwendig ist.	Portabilität, keine Installation	K9	Vorteile Web-Anwendungen	
49	B4	Web-Anwendungen sind durch den Browser limitiert.	Limitierung durch Browser	K10	Nachteile Web-Anwendungen	
50	B4	Keine Erfahrung mit Migration, da Entscheidung immer zu Beginn getroffen wurde.	Entscheidung immer vor Beginn	K11	Thematische Erfahrung Migration und Migrationsstrategien	
51	B4	Eine Migration ist erfolgreich, wenn der Benutzerkomfort der Anwendung gleich bleibt bzw. erhöht wird.	Benutzerkomfort gleich oder größer	K12	Erfolgreiche Migration	

52	B4	Web-Anwendungen können Desktop-Anwendungen in gewissen Bereichen vollständig ersetzen. In speziellen Bereichen, wo direkter Hardwarezugriff notwendig ist, ist es noch nicht möglich.	vollständiger Ersatz möglich, spezielle Bereiche bilden Ausnahme	K13	Eigene Meinung Forschungsfrage	
53	B5	Vor vielen Jahren eine Desktop-Anwendung entwickelt.	Desktop-Anwendung entwickelt	K1	Thematische Erfahrung Desktop-Anwendungen	K1, K2, K11 nicht relevant.
54	B5	Erfahrung in der Web-Entwicklung durch Beruf.	Beruf	K2	Thematische Erfahrung Web-Anwendungen	
55	B5	Eine Desktop-Anwendung wird auf einem Computer installiert, also eine Installation ist notwendig, um die Anwendung lauffähig zu bekommen.	Installation notwendig	K3	Beschreibung und Merkmale Desktop-Anwendungen	
56	B5	Bei Integration mit Software wie z.B. Office-Paket, ist der Einsatz von Desktop-Software sinnvoll.	Integration von Desktop-Software	K4	Einsatzmöglichkeiten Desktop-Anwendungen	
57	B5	Desktop-Anwendungen bieten einen Vorteil bei Integration bzw. Interaktion mit anderer Desktop-Software.	Anbindung an andere Desktop-Software	K5	Vorteile Desktop-Anwendungen	
58	B5	Das Ausrollen von Desktop-Anwendungen bei großen Kunden kann aufgrund von Beschränkungen sehr mühsam sein.	Ausrollen mühsam	K6	Nachteile Desktop-Anwendungen	
59	B5	Eine Web-Anwendung läuft in einem Browser und es macht keinen Unterschied, welches Endgerät verwendet wird.	läuft in Browser	K7	Beschreibung Web-Anwendungen	
60	B5	Web-Anwendungen kommen zum Einsatz, wenn Leute schnell und breit erreicht werden sollen, z.B. Content verteilen. Skalierung ist auch ein großes Thema.	schnell die breite Masse erreichen, Skalierung	K8	Einsatzmöglichkeiten Web-Anwendungen	

61	B5	Die Vorteile von Web-Anwendungen sind, dass keine Installation notwendig ist und dass sie Plattform übergreifend funktionieren.	keine Installation, Cross Plattform	K9	Vorteile Web-Anwendungen	
62	B5	Nachteile von Web-Anwendungen sind die Interaktion mit Desktop-Software und die Offline-Fähigkeit.	Interaktion mit Desktop-Software schwer, Offline-Funktion	K10	Nachteile Web-Anwendungen	
63	B5	Erfahrung mit Migration von Desktop auf Web.	Desktop auf Web	K11	Thematische Erfahrung Migration und Migrationsstrategien	
64	B5	Eine Migration ist erfolgreich, wenn das alte System abgeschalten werden kann und es keine Abhängigkeiten mehr gibt.	wenn altes System abgeschalten werden kann und keine Abhängigkeiten	K12	Erfolgreiche Migration	
65	B5	Web-Anwendungen können Desktop-Anwendungen ersetzen und der Weg geht immer weiter in Richtung Web.	können ersetzen	K13	Eigene Meinung Forschungsfrage	

ANHANG B - Kodierleitfaden

Kategorie	Definition	Ankerbeispiel	Kodierregeln
K1: Thematische Erfahrung Desktop-Anwendungen	Alle Textstellen, die auf eine Deutung der persönlichen Erfahrung mit dem Thema Desktop-Anwendungen hinweisen.	B2: Also ich habe extrem, bzw. sehr viele Jahre Erfahrung mit Desktop-Applikationen. Es sind mittlerweile über 25 Jahre. Ich habe da auch diverse Technologien verwendet. B3: Ich habe im Unternehmen in dem ich angefangen habe zu arbeiten vor 6 Jahren, hauptsächlich Desktop-Applikationen entwickelt.	Nur die Erfahrungen, keine Erklärungen vom Begriff Desktop-Anwendungen.
K2: Thematische Erfahrung Web-Anwendungen	Alle Textstellen, die auf eine Deutung der persönlichen Erfahrung mit dem Thema Web-Anwendungen hinweisen.	B1: In der alten Firma habe cih Cloud-Anwendungen gemacht fürs Finanzmanagement der kleinen Unternehmen. B4: Ich habe davor, also bevor ich im Bereich Online-Shops gearbeitet habe, war ich Selbstständig und habe so eine Plattform aufgezogen für Restaurant-Bewertungen und dann halt auch verkauft.	Nur die Erfahrungen, keine Erklärungen vom Begriff Web-Anwendungen.
K3: Beschreibung und Merkmale Desktop-Anwendungen	Alle Textstellen, die auf eine Beschreibung oder Merkmale von dem Begriff Desktop-Anwendungen hinweisen.	B5: Ja ich glaube das entscheidenste Merkmal ist glaube ich, dass die Anwendung auf deinem Computer installiert wird. Also das heißt eine Installation ist notwendig um die Applikation lauffähig zu bekommen. Das ist also das entscheidende Merkmal, würde ich jetzt meinen. B2: Also ich würde etwa einmal sagen, Desktop-Anwendungen werden lokal installiert, brauchen lokal Hardware Ressourcen, Desktop-Anwendungen können immer gestartet werden, unabhängig davon, ob ich jetzt online bin oder offline.	Nur die Beschreibungen und spezifische Merkmale des Begriffes Desktop-Anwendungen.
K4: Einsatzmöglichkeiten Desktop-Anwendungen	Alle Textstellen, die auf die Einsatzmöglichkeiten von Desktop-Anwendungen hinweisen.	B2: Mittlerweile bzw. früher war die Entscheidung eher einfacher, weil die Web-Applikationen diese Features nicht geboten haben die sie jetzt bieten, wie Web-Assemblies und diese Dinge. Aber ich würde sie eher einsetzen wenn es um Performance geht und um Interop Szenarien. B3: Es ist eigentlich hauptsächlich bei Stationen wo man wenig Internetanbindung hat. Also es gibt im Lagerumfeld, es gibt aber auch bei Einkaufzentren, Kassen z.B. da hat man sehr häufig Desktop-Anwendungen im Einsatz. Also primär dort, wo man nicht unbedingt einen Browser haben möchte.	Nur die Einsatzmöglichkeiten, keine Begriffsbeschreibungen.
K5: Vorteile Desktop-Anwendungen	Alle Textstellen, die auf die Vorteile von Desktop-Anwendungen hinweisen.	B4: Vorteile sind halt Performance z.B., das ist ein eindeutiger Vorteil. Ich habe keine Ressourcenlimitierung durch einen Layer. Ich kann mit der Desktop-Anwendung auf alles zugreifen. B1: Also die sind in der Regel schneller. Das wäre eine Sache. Wir haben direkten Hardwarezugriff, möglicherweise, ist aber nicht notwenig. Hängt davon ab wie es gemacht wird. Das definitiv. Vielleicht könnte man sagen, es ist einfacher zu entwickeln sein könnte, wenn man eine spezifische Sache gut kennt und für die entwickelt.	Alle Aspekte, welche Vorteile von Desktop-Anwendungen aufzeigen. Keine Nachteile.
K6: Nachteile Desktop-Anwendungen	Alle Textstellen, die auf die Nachteile von Desktop-Anwendungen hinweisen.	B5: Also das Ausrollen der Software ist einfach immer ein riesen Problem. B3: Wenn man Plattform unabhängig sein möchte ist es ein erheblicher Nachteil. Also man muss im schlimmsten Fall die selbe Applikation drei mal schreiben oder häufiger, wenn noch eine Windows Version rauskommt.	Nur Nachteile.
K7: Beschreibung und Merkmale Web-Anwendungen	Alle Textstellen, die auf eine Beschreibung oder Merkmale von dem Begriff Web-Anwendungen hinweisen.	B1: Ich würde sagen dass es in der Regel eine Anwendung ist, wo du Client Zugriff auf einen Server hast. B4: Eine Web-Anwendung ist für mich eine Anwendung, die ich über das Internet aufrufe. Bei der es beim ersten Mal sein kann, dass sie Ressourcen herunterladet, aber ich mich primär nicht um die Installation oder um das Dahinter kümmern muss.	Nur die Beschreibungen und spezifische Merkmale des Begriffes Web-Anwendungen.
K8: Einsatzmöglichkeiten Web-Anwendungen	Alle Textstellen, die auf die Einsatzmöglichkeiten von Web-Anwendungen hinweisen.	B5: Bei jeder Applikation wo du schnell und breit Leute erreichen willst. B2: Also vorallem dann, wenn ich ein riesengroßes Publikum erreichen möchte. Also wenn ich sage ich habe, k.A. 3 Millionen User, ich habe irgendeinen Ticketstore, oder so für Firmen, die E-Business Seiten haben, die irgendwelche Produkte kaufen kann.	Nur die Einsatzmöglichkeiten, keine Begriffsbeschreibungen.
K9: Vorteile Web-Anwendungen	Alle Textstellen, die auf die Vorteile von Web-Anwendungen hinweisen.	B4: Die Vorteile sind definitiv die Portabilität, das ich mit einem Entwicklungszweig viele Geräte abdecken kann, ohne dass ich viele Variationen davon brauche. B1: Du hast mehrere Menschen die miteinander kolaborieren können. Das ist schon eine Sache die ein Vorteil ist. Die andere ist, für mich als Entwickler finde ich ist es relativ einfach, wenn du ein paar Frameworks kennst die du verwenden kannst und es ist auch einfacher andere Entwickler zu finden, die das machen, als jene die spezialisiert sind auf native Sachen.	Alle Aspekte, welche Vorteile von Web-Anwendungen aufzeigen. Keine Nachteile.
K10: Nachteile Web-Anwendungen	Alle Textstellen, die auf die Nachteile von Web-Anwendungen hinweisen.	B3: Es ist vorallem hier in Österreich und in Deutschland ein ziemlich großes Problem. Also man kann nicht so einfach Cloud-Lösungen zur Verfügung stellen, weil meistens gibt es da Netzwerkprobleme also da muss ich das genau so freigeben für das Netzwerk. B5: Das Thema Offline-Fähigkeit ist natürlich ein Problem, aber ich glaube, dass man das heute fast vernachlässigen kann.	Nur Nachteile.
K11: Thematische Erfahrung Migration bzw. Migrationsstrategien	Alle Textstellen, die auf eine Deutung der persönlichen Erfahrung mit den Themen Migration und Migrationsstrategien hinweisen.	B2: Auch schon ewig viel. Also ungefähr gleich viel an Zeit wie bei Desktop. B3: Ja also ich habe einerseits Big-Bang gehabt, also wir machen einfach alles neu. Ist dem geschuldet wenn man halt alte Technologie verwendet, dass das teilweise gar nicht geht.	Nur die Erfahrungen, keine Erklärungen vom Begriff Migration bzw. Migrationsstrategien.
K12: Erfolgreiche Migration	Alle Textstellen, die auf eine Deutung der erfolgreichen Migration hinweisen.	B5: Man muss das alte System abdrehen können, sonst ist es nicht erfolgreich. Wenn du danach beides betreuen musst, dann hast du den Worst-Case erreicht. B4: Also vom Funktionsumfang ist es das eine und der Wunsch natürlich, dass der Benutzerkomfort von der Anwendung gleich bleibt bzw. erhöht wird.	Nur die Meinung zum Thema erfolgreiche Migration. Keine Definitionen von Migration oder Strategien.
K13: Eigene Meinung Forschungsfrage	Alle Textstellen, die auf eine Deutung der persönlichen Meinung der befragten Person hinweisen im Hinblick zur Forschungsfrage.	B2: Also ich dentiere eher zu nein muss ich ehrlich sagen. Weil ich eben der Meinung bin, dass wir auf der Welt noch zu viel Legacy Code haben und dieser Legacy Code nur durch Desktop-Applikationen irgendwie weiter verwendet werden kann. Und der Migrationsaufwand einfach einengt. B5: Ja also ich glaube schon, bzw. bin überzeugt davon dass eine Web-Anwendung das kann.	Nur die eigene Meinung, keine Erklärungen von Begriffen.

ANHANG C - Interviewleitfaden

Interviewleitfaden

Vorbemerkung:

Das Ziel der Masterarbeit ist es, herauszufinden bei welchen Einsatzgebieten Desktop- bzw. Web-Anwendungen sinnvoll sind, wie sich diese beiden Anwendungsarten voneinander unterscheiden und ob Web-Anwendungen in der Lage sind, Desktop-Anwendungen zu ersetzen. Im Rahmen des Interviews sollen die folgend aufgeführten Fragen durch die Expertise der befragten Personen geklärt werden und damit zur Beantwortung der Forschungsfrage beitragen.

Thema Einstieg

1. Welche Erfahrung haben Sie mit Desktop-Anwendungen?
2. Welche Erfahrung haben Sie mit Web-Anwendungen?

Thema Desktop-Anwendungen

3. Wie würden Sie Desktop-Anwendungen beschreiben?
4. Wann würden Sie primär Desktop-Anwendungen einsetzen?
5. Welche Vor- und Nachteile sehen Sie bei Desktop-Anwendungen?

Thema Web-Anwendungen

6. Wie würden Sie Web-Anwendungen beschreiben?
7. Wann würden Sie primär Web-Anwendungen einsetzen?
8. Welche Vor- und Nachteile sehen Sie bei Web-Anwendungen?

Thema Migrationsstrategien

9. Welche Erfahrung haben Sie mit Migrationsstrategien bzw. Szenarien?
10. Wann ist eine Migrationsstrategie Ihrer Meinung nach erfolgreich?

Thema Web- vs. Desktop-Anwendung

11. Sind Sie der Meinung, dass eine Web-Anwendung eine Desktop-Anwendung vollständig ersetzen kann?
    a. Wenn ja, wie kann es funktionieren?
    b. Wenn nein, warum nicht?

ABKÜRZUNGSVERZEICHNIS

API	Application Programming Interface
CAD	Computer-aided design
CD	Compact Disc
CSS	Cascading Style Sheets
HTML	Hypertext Markup Language
IaaS	Infrastructure as a Service
PaaS	Platform as a Service
PWA	Progressive Web App
RAM	Random Access Memory
RIA	Rich Internet Application
SaaS	Software as a Service
SEO	Search Engine Optimization
SMTP	Simple Mail Transfer Protocol
USB	Universal Serial Bus

ABBILDUNGSVERZEICHNIS

LITERATURVERZEICHNIS

Abts, D. (2019). *Masterkurs Client/Server-Programmierung mit Java: Anwendungen entwickeln mit Standard-Technologien.* Springer Vieweg.

Ackermann, P. (2021). *Webentwicklung - Das Handbuch für Fullstack-Entwickler.* Bonn: Rheinwerk Computing.

Alter, T. (2017). *Building Progressive Web Apps: Bringing the Power of Native to the Browser.* O'Reilly.

Blokdyk, G. (2019). *Application Migration a Complete Guide.* Brisbane: Emereo.

Borges, M., Schumacher, J., & Redeker, T. (2002). *PC-Wissen. Die Welt der Hardware und Software.* München: Markt+Technik.

Brügge, B., Harhoff, D., Picot, A., Creighton, O., Fiedler, M., & Henkel, J. (2004). *Open-Source-Software. Eine ökonomische und technische Analyse.* Berlin: Springer.

Carl, D. (2006). *Praxiswissen Ajax.* Köln: O'Reilly.

Casteleyn, S., Daniel, F., Dolog, P., & Matera, M. (2009). *Engineering Web Applications.* Heidelberg: Springer.

David, M. (2013). *HTML5: Designing Rich Internet Applications.* Oxon: Taylor & Francis.

De, D. (2016). *Mobile Cloud Computing Architectures, Algorithms and Applications.* Boca Raton: CRC Press.

Eisenman, B. (2018). *Learning React Native.* Sebastopol: O'Reilly.

Fink, G., & Flatow, I. (2014). *Pro Single Page Application Development: Using Backbone.js and ASP.NET.* New York: Apress.

Flanagan, D. (2012). *JavaScript kurz & gut.* Köln: O'Reilly.

Ford, N. (2020). *Fundamentals of Software Architecture: An Engineering Approach. A Comprehensive Guide to Patterns, Characteristics, and Best Practices.* O'Reilly.

Garverick, J. (2018). *Migrating to Azure - Transforming Legacy Applications into Scalable Cloud-First Solutions.* New York: Apress.

Grassmuck, V. (2004). *Freie Software. Zwischen Privat- und Gemeineigentum.* Bonn: Bundeszentrale für politische Bildung.

Grigorik, I. (2013). *High Performance Browser Networking.* Sebastopol: O'Reilley.

Henderson, C. (2006). *Building Scalable Web Sites: Building, Scaling, and Optimizing the Next Generation of Web Applications.* O'Reilly.

Hoffman, A. (2020). *Web Application Security: Exploitation and Countermeasures for Modern Web Applications.* O'Reilly.

Hoffmann, A. (2020). *Web Application Security.* O'Reilly.

Hurwitz, J., Nugent, A., Halper, F., & Kaufmann, M. (2013). *Big Data for Dummies.* Hoboken: John Wiley & Sons.

III, A. T. (2008). *Ajax: The Definitive Guide: Interactive Applications for the Web.* O'Reilly.

Jensen, P. (2017). *Cross-Platform Desktop Applications: Using Node, Electron, and NW.js.* Manning.

Kavis, M. (2014). *Architecting The Cloud - Design Decisions For Cloud Computing Service Models (SaaS, PaaS, and IaaS).* Hoboken: Wiley.

Kavis, M. J. (2014). *Architecting the Cloud: Design Decisions for Cloud Computing Service Models (SaaS, PaaS, and IaaS).* Wiley.

Krämer, A. (2021). *Cross-Plattform-Apps mit Xamarin.Forms entwickeln: Mit C# für Android und iOS programmieren.* Hanser Verlag.

Laszewski, T., & Nauduri, P. (2012). *Migration To The Cloud.* Waltham: Syngress.

Liebel, C. (2018). *Progressive Web Apps: Das Praxisbuch. Plattformübergreifende App-Entwicklung mit Angular und Workbox. Für Browser, Windows, macOS, iOS und Android.* Rheinwerk Computing.

Magazine, S. (2012). *CSS Essentials.* Freiburg: Smashing Magazine.

Matam, S., & Jain, J. (2017). *Pro Apache JMeter: Web Application Performance Testing.* Apress.

Mayring, P. (2015). *Qualitative Inhaltsanalyse - Grundlagen und Techniken.* Weinheim: Beltz.

Mikowski, M., & Powell, J. (2013). *Single Page Web Applications: JavaScript end-to-end.* Manning.

Misra, S., & Egoeze, F. (2014). *Framework for Maintainability Measurement of Web Application for Efficient Knowledge-Sharing on Campus Intranet.* Springer.

Müller, P. (2020). *Einstieg in HTML und CSS: Webseiten programmieren und gestalten mit HTML und CSS.* Rheinwerk.

Musciano, C., & Kennedy, B. (2002). *HTML and XHTML The Definitive Guide.* Sebastopol: O'Reilly.

Nadareishvili, I., Mitra, R., McLarty, M., & Amundsen, M. (2016). *Microservice Architecture.* Sebastopol: O'Reilly.

Newman, S. (2020). *Monolith to Microservices - Evolutionary Patterns to Transform Your Monolith.* Sebastopol: O'Reilly.

O'Brien, M., Linthicum, D., & Fuller, J. (2003). *Next Generation Application Integration: From Simple Information to Web Services.* Addison-Wesley.

Peppers, J. (2015). *Xamarin Cross-platform Application Development.* Birmingham: Packt Publishing.

Saleck, T. (2005). *Chefsache Open Source. Kostenvorteile und Unabhängigkeit durch Open Source.* Wiesbaden: Vieweg.

Scott, A. D. (2020). *JavaScript Everywhere: Building Cross-Platform.* O'Reilly.

Sheppard, D. (2017). *Beginning Progressive Web App Development - Creating a Native App Experience on the Web.* Illinois: Apress.

Stuttard, D., & Pinto, M. (2011). *The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws.* Wiley.

Thatcher, J., Urban, M., Lawson, B., Burks, M. R., Rutter, R., Kirkpatrick, A., . . . Waddel, C. D. (2006). *Web Accessibility: Web Standards and Regulatory Compliance.* Apress.

Zhang, Y. (2022). *Mobile Edge Computing.* Springer.