WHITEPAPER

# Ausfallsicherheit der IT-Anwendung

Wie „On-Premise“-Umgebungen im Rahmen eines Paradigmenwechsels zu fortschrittlichen Lösungen für die Geschäftsautomatisierung führen können.

**Stefan Appel** ISR | **Helge Schneider** CENIT

ISR Information Products AG

## EINFÜHRUNG

Die Idee und Motivation für dieses Dokument sind eine langjährige Beschäftigung mit IBM FileNet-Lösungen für große Unternehmen mit hohen und komplexen Anforderungen an Verfügbarkeit und Skalierbarkeit. Dies war und ist immer noch eine Herausforderung, wenn man heute über „On-Premise“-Umgebungen spricht. Aber die Technologie hat sich weiterentwickelt und heute können wir diese FileNet-Umgebungen auch auf Kubernetes- oder Cloud-Plattformen betreiben. Mit der erwähnten Entwicklung haben sich die Dinge ein wenig verändert und einige der früheren Anforderungen sind heute einfacher zu handhaben. Aber es haben sich auch neue Herausforderungen und Fragestellungen ergeben.
An dieser Stelle kann die ISR Information Products AG als IT-Dienstleister für Enterprise Content Management (ECM) Produkte mit langjähriger Erfahrung im Aufbau von ECM-Plattformen, deren Wartung und schließlich der Entwicklung und Bereitstellung von Lösungen auf diesen Plattformen unterstützen.

# 1 | ON-PREMISE GOES CLOUD

Die Idee und Motivation für dieses Dokument sind eine langjährige Beschäftigung mit IBM FileNet-Lösungen für große Unternehmen mit hohen und komplexen Anforderungen an Verfügbarkeit und Skalierbarkeit. Dies war und ist immer noch eine Herausforderung, wenn man heute über „On-Premise“-Umgebungen spricht. Aber die Technologie hat sich weiterentwickelt und heute können wir diese FileNet-Umgebungen auch auf Kubernetes- oder Cloud-Plattformen betreiben. Mit der erwähnten Entwicklung haben sich die Dinge ein wenig verändert und einige der früheren Anforderungen sind heute einfacher zu handhaben. Aber es haben sich auch neue Herausforderungen und Fragestellungen ergeben. An dieser Stelle kann die ISR Information Products AG als IT-Dienstleister für Enterprise Content Management (ECM) Produkte mit langjähriger Erfahrung im Aufbau von ECM-Plattformen, deren Wartung und schließlich der Entwicklung und Bereitstellung von Lösungen auf diesen Plattformen unterstützen.

## 1.1 | Herausfordernde „On-Premise“-Architekturen für ECM-bezogene Infrastrukturen

Die Herausforderungen beim Betrieb einer „traditionellen“ On-Premise und verteilten Anwendung mit hoher Verfügbarkeit können recht komplex sein und sind oft nicht einfach zu bewältigen. Dies gilt insbesondere, wenn man an die folgenden Anforderungen oder Erwartungen an Erweiterungen der Anwendung denkt:

- Ihre Endnutzer erwarten heutzutage eher eine permanente Verfügbarkeit mit 24/7, sodass Ausfälle des Systems für die Systemwartung, neue Releases oder auch bei schwerwiegenden Problemen auf ein kleines Minimum an erwarteten Ausfällen reduziert werden müssen.
- Das Unternehmen und die Stakeholder erwarten in der Regel nur moderate Kosten für den Betrieb der Anwendung. Wenn die Lösung skaliert wurde, sind oft nur die Kosten für die zugrundeliegende Hardware einkalkuliert, nicht aber den Aufwand, der entsteht, wenn zusätzliche Konfigurationen an der bestehenden Hardware und Software vorgenommen werden müssen.
- Im Allgemeinen wird vom Betriebspersonal erwartet, dass es über „High-End“-Kenntnisse für nahezu jede beteiligte Komponente in der Lösung verfügt. Denken Sie also an eine moderne ECM-Lösung auf der Grundlage von IBM FileNet: Dabei handelt es sich i.d.R. nicht nur über um die grundlegenden IBM-Produkte, sondern auch über Anwendungsserver, Datenbanken, LDAP- oder Identitätsmanagementsysteme, Lastenverteilungen über Hard- oder Software Load Balance, Netzwerke, Webserver und vieles mehr.
- Die Betriebs- und Entwicklungsteams haben bei diesen Lösungen in der Regel getrennte Aufgaben (der Betrieb ist für einen geräuschlosen Dienst verantwortlich und die Entwicklung „infiltriert“ fremde Angelegenheiten), die Zusammenarbeit zwischen beiden ist minimal. DevOps ist ein Konzept, das beide Welten miteinander verbinden soll, aber in den meisten Unternehmen wird es nicht oder noch nicht vollständig umgesetzt.
- Projektteams arbeiten oft mit agilen oder Scrum-Methoden und wollen ihre Neuentwicklungen schnell auf den Testsystemen oder sogar in der Produktion sehen. Traditionelle komplexe Lösungen erfüllen diese Voraussetzung nur, wenn eine vollständige Automatisierung des Deployments vorhanden ist (bekanntes Problem bei IBM FileNet-Lösungen im Allgemeinen durch das Fehlen entsprechender Tools).

Aber es gibt noch ein weiteres großes Problem bei traditionellen Architekturen, z.B. wenn Sie darüber nachdenken, Ihre bestehende Lösung mit neuen Komponenten zu erweitern, sei es wegen Leistungsoptimierung oder wegen der Notwendigkeit, neue Anwendungen oder weitere Benutzer in Ihre bestehende Umgebung zu integrieren.

Das bedeutet im Allgemeinen, dass Sie nicht nur Ihre neuen Komponenten wie einen neuen Content-Server in ECM-Architekturen integrieren müssen, sondern auch bedenken müssen, dass das Farming oder Clustering in Ihren Anwendungsservern oder vielleicht auch in Ihrer Lastausgleichs- oder Webservice-Infrastruktur angepasst werden muss. Die Lastverteilung muss neu ausbalanciert und das Risiko von Ausfällen im Normalfall dazu neu erfasst werden.

In Anbetracht dieser Herausforderungen müssen wir uns mit Aspekten der Hochverfügbarkeit sowie mit deren Grundsätzen für die letzten „Verteidigungslinien" der Anwendbarkeit zur Hochverfügbarkeit befassen.

Der effektivste Weg, diese Ziele zu erreichen, ist die Vermeidung eines einzelnen Fehlerpunkts (eines sog. SPoF = Single Point of Failure) in einer komplexen Softwarearchitektur, deren Ausfall zur Belastung in der Gesamtlösung werden kann. Dies kann man effektiv durch die folgenden Mechanismen verhindern, indem man:

- Verwendung redundanter Komponenten - d. h. 2 Backend-Server, 2 Frontend-Server, 2 ThirdParty Komponenten von Drittanbietern für die relevanten Anwendungen
- Mindestens ein zusätzliches oder Backup-Rechenzentrum (Cold, Standby oder Hot)
- Nutzung von Serverfarmen oder Clustern, wann immer möglich
- Verwendung hochverfügbarer Datenfreigaben (Shares) für Dateisysteme
- Einsatz von vorgelagerten Webservern oder sonstigen Lastverteilern[1]

## 1.2 | Eine IBM ECM-Lösung mit Best Practices

Unter Abbildung 1 sehen Sie ein Beispiel für eine geclusterte IBM FileNet-Umgebung mit doppelter Redundanz. Die FileNet Content Platform Engine sowie IBM Content Navigator und IBM Content Collector sind mindestens zwei Komponenten in einem Rechenzentrum (RZ oder DC für Data Center).

Die zugrundeliegenden Datenbanken werden synchron oder in Zeitabschnitten synchronisiert - je nachdem, welche Mechanismen der jeweiligen Datenbankanwendung zugrunde liegen (aktiv/aktiv oder aktiv/passiv).

Die Anwendungsserver werden nach dem One-Cell-Konzept konfiguriert und die darauf ausgeführten Anwendungen geclustert. Vorgeschaltete Hardware-Load-Balancer (dasselbe gilt für Software-Lösungen mit Lastausgleich) garantieren die korrekte Verteilung der Arbeitslast und mindern das Risiko von Datenverlusten durch intelligente Mechanismen.

[1] Sie bietet auch die Möglichkeit einer echten Kontrolle der Datenlast und der Leistung.

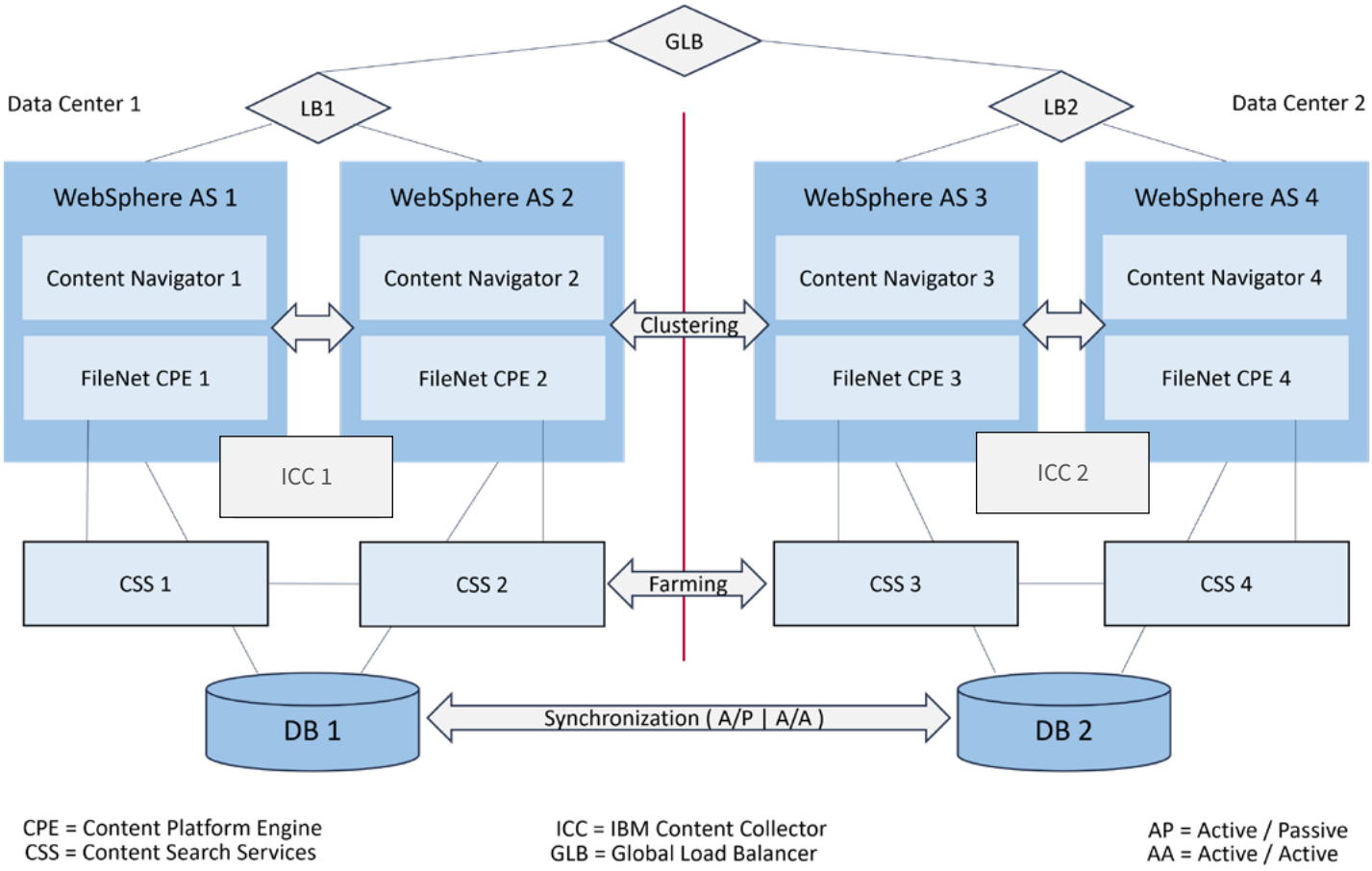


*Abbildung 1: Eine typische ECM-Architektur*

In dieser ECM-Lösung sind zwei Rechenzentren vorhanden und die Komponenten sind größtenteils redundant in einem Rechenzentrum ausgelegt. Alle Komponenten führen eine Synchronisation untereinander durch. Dieses Beispielszenario zeigt, wie ein Up-Front-Load-Balancer zwischen den beiden Rechenzentren (RZ1/RZ2) die Anfragen über das Frontend verteilen können, wenn der Modus durch einen Aktiv/Aktiv-Ansatz gesteuert wird. In anderen Szenarien mit einem Aktiv/Standby-Ansatz kann er dies auch als seinen Einstiegspunkt verwenden, anstatt zu vorab zu entscheiden, an welchem Standort das aktive Rechenzentrum gerade gemeinsam genutzt wird.

In allen diesen Szenarien - auch wenn Sie Ihre Komponenten durch andere Softwareprodukte als die von IBM ersetzen oder wenn Sie mehr als zwei Rechenzentren oder als weiteren Schutz eine demilitarisierte Zone (DMZ) aufsetzen oder in eigene Zonen trennen - ist doch eins allen gemeinsam: Der Aufwand, um solche eine Architektur zu erreichen, ist oft sehr aufwendig in der Planung und oft nicht einfach zu realisieren. Denken Sie dabei an gleiche Konfigurationsquellen - wie werden diese gemeinsam genutzt? Ist es einfacher, diese Art von Lösung mit zusätzlichen Komponenten zu skalieren, wenn Sie diese benötigen, und sie wieder zu entfernen, wenn sie nicht mehr benötigt werden?

Zusammenfassend lässt sich sagen, dass man alle vorhandenen Skillsets im Hintergrund haben und sehr gut planen muss, wenn man eine geplante Änderung hieran vornimmt, und das ist nur der einfachere Teil der Geschichte. Denken Sie an einen ungeplanten oder nicht gewünschten Ausfall, der durch nicht vorhersehbare Situationen verursacht wird. Ein weiterer Aspekt sind die finanziellen Kosten für die Aufrechterhaltung einer solchen Architektur. In den meisten Fällen müssen Systemadministratoren und ECM-Administratoren Hand in Hand mit dem Entwicklungsteam arbeiten, wenn Fehler behoben oder Vorfälle ausgewertet werden müssen.

## 1.3 | Anleitung zur Hochverfügbarkeit

Ein wichtiges Merkmal einer nachhaltigen Lösung ist sicherlich die Verfügbarkeit der Lösung selbst. Die Anwender wollen nicht unter ständigen Systemausfällen leiden und im Idealfall bemerken sie den Ausfall einer Komponente nicht einmal dann, wenn diese bereits beschädigt ist und eine Zeit lang nicht zur Verfügung steht. Die Hochverfügbarkeit einer Anwendung wird anhand der folgenden Merkmale definiert:

1. Die allgemeine Systemverfügbarkeit[2] ist immer nur so gut wie die am wenigsten verfügbare Systemkomponente.
2. Es wird davon ausgegangen, dass jeder „Server“ durch Redundanz hochverfügbar eingerichtet ist, d.h. es sollten mindestens 2 Knoten auf jedem Servertyp vorhanden sein.
3. Alle abhängigen Dienste in der Infrastrukturebene (Dateispeicher, Datenbank, Netzwerk) müssen mit der entsprechenden Verfügbarkeit und Leistungsqualität eingerichtet werden.
4. Ein einziger Datenbankserver kann zur Unterstützung aller Server verwendet werden, aber für die verschiedenen Server werden unterschiedliche Datenbanken benötigt. Je nach Nutzungsmuster und Verfügbarkeitsanforderungen sollte man jedoch die Verwendung verschiedener Datenbankserverinstanzen in Betracht ziehen - wenn z. B. eine Content Serverinstanz eine viel höhere Verfügbarkeit erfordert als z.B. eine Workflow Instanz, dann sollte man eher eine eigene Datenbankserverinstanz zur Unterstützung dieser Verfügbarkeitsstufe vorsehen.

Das folgende Kapitel zeigt einige weitere Beispiele aus der Praxis auf.

[2] Details auf https://en.wikipedia.org/wiki/High_availability (Zugriff am 16.02.2023)

## 1.4 | Allgemeiner Failover-Ansatz

Ein Failover stellt sicher, dass bei Ausfall von Komponenten ausreichend weitere vorhanden sind, die die Last im System übernehmen können. Zudem spricht man auch von Failover bei übergreifenden Rechenzentren sowie bei Datenbanksystemen und auch bei Speichermedien wie Netzwerkshares.

- Der allgemeine Ansatz zur Erreichung einer hohen Verfügbarkeit basiert auf einer Aktiv/Aktiv-Konfiguration, die sich über zwei Rechenzentren erstreckt.
- Der Zugriff auf die beiden Rechenzentren erfolgt über zentrale Netzwerk-Lastverteiler, um die Last zu verteilen, etwa 50/50 für jedes Rechenzentrum. Jedes Rechenzentrum verfügt über genügend Kapazität, um 100 % der regulären Produktionsauslastung zu unterstützen. Mit dieser Aktiv/Aktiv-Konfiguration können unter normalen Betriebsbedingungen Lastspitzen von bis zu 200 % bewältigt werden.
- Die FileNet Content Engine wird in einer Farmkonfiguration geclustert (mehrere Knoten über mehrere Rechenzentren).
- Es werden zwei Software-Cluster (einer pro Rechenzentrum) konfiguriert, die unabhängig voneinander arbeiten können. Daher wird der Ausfall eines Clusterknotens dazu führen, dass seine Arbeitslast auf andere Knoten innerhalb der Farm verteilt wird, wobei der Load Balancer den nächstgelegenen Nachbarn innerhalb des lokalen Netzsegments bevorzugt. Der rechenzentrumsübergreifende Verkehr wird auf ein Minimum beschränkt. Die Aufrechterhaltung von Sitzungen wird, soweit möglich, innerhalb eines bestimmten Rechenzentrums beibehalten, um so Querkommunikation zu vermeiden.

### 1.4.1 | Szenario 1

Die folgenden Fälle beschreiben typische Fehlerszenarien und die damit verbundenen Auswirkungen.
Ein Server fällt aus:

Die Last wird weiterhin zu 50 % auf die Rechenzentren verteilt.

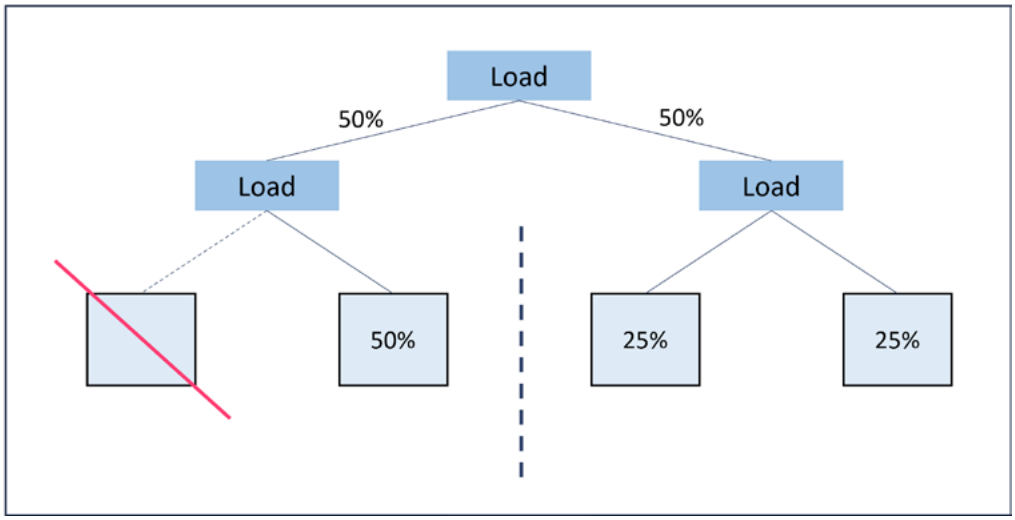


*Abbildung 2: Failover Beispiel 1*

Die Gesamtkapazität beträgt weiterhin 150 % der regulären Last.

Auswirkungen:
Der verbleibende Server in Data Center 1 muss 50 % der Last unterstützen - das Sizing des Ansatzes bietet dazu genügend Kapazität. Alle Benutzer mit Sitzungen in Data Center 2 werden keine Leistungsänderungen feststellen.

Der Load Balancer in Data Center 1 erkennt den Verlust der Keepalive-Heartbeat-Kommunikation und stellt keine Sitzungen mehr für den verlorenen Server bereit. Nach dem Auftreten dieses Fehlers ermöglicht eine manuelle „Out of Service“- oder „single user mode“-Konfiguration eine unabhängige administrative Arbeit auf dem Server, um das Problem zu ermitteln und eventuelle Fehler zu beheben. Nach der Behebung des Problems wird die „Out of Service“-Konfiguration zurückgesetzt und nach einer kurzen Zeitspanne erkennen die Load Balancer die Keepalive-Kommunikation erneut und die maximale Lastverteilung wird automatisch erreicht.

### 1.4.2 | Szenario 2

Zwei Server fallen in einem einzigen Rechenzentrum aus:
Alle Lasten werden von dem verbleibenden Rechenzentrum getragen.

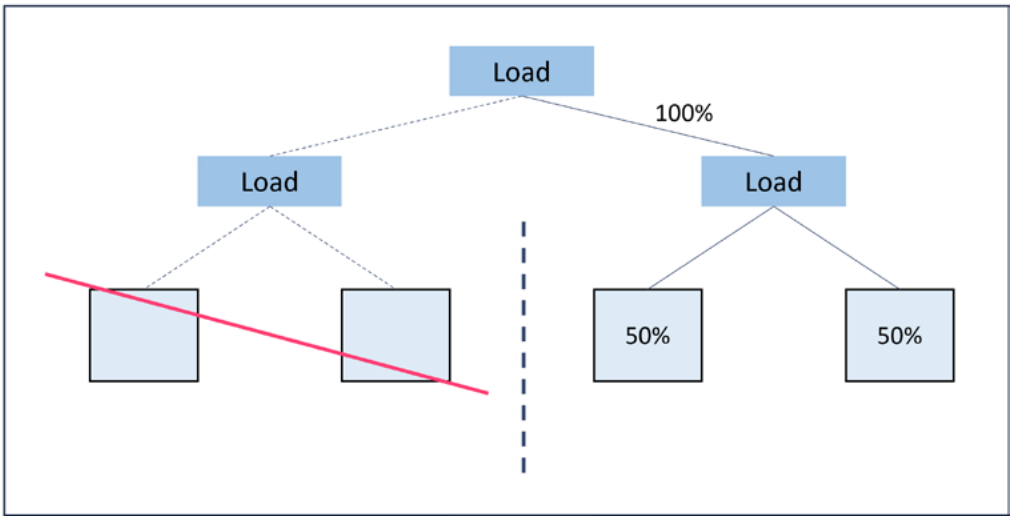


*Abbildung 3: Failover Beispiel 2*

Die Gesamtkapazität beträgt 100 % der regulären Last.

Auswirkungen:
Nur Data Center 2 wird die Last allein unterstützen. Der Top Load-Balancer wird automatisch die Bereitstellung von Sitzungen für das verlorene Rechenzentrum einstellen. Nach dem Auftreten dieses Fehlers ermöglicht eine manuelle „Out of Service“-Konfiguration eine unabhängige administrative Arbeit an dem fehlerhaften Data Center. Nach Behebung des Problems wird die „Out of Service“-Konfiguration zurückgesetzt und nach einiger Zeit werden Keepalive-Kommunikationen erkannt und die maximale Lastverteilung wird automatisch erreicht.

Dieses Verhalten kann z.B. auch für Wartungsarbeiten in einem einzelnen Rechenzentrum genutzt werden.

### 1.4.3 | Szenario 3

Ein physischer Server in jedem Rechenzentrum fällt aus:
Die Last wird weiterhin zu 50 % auf die Rechenzentren verteilt.

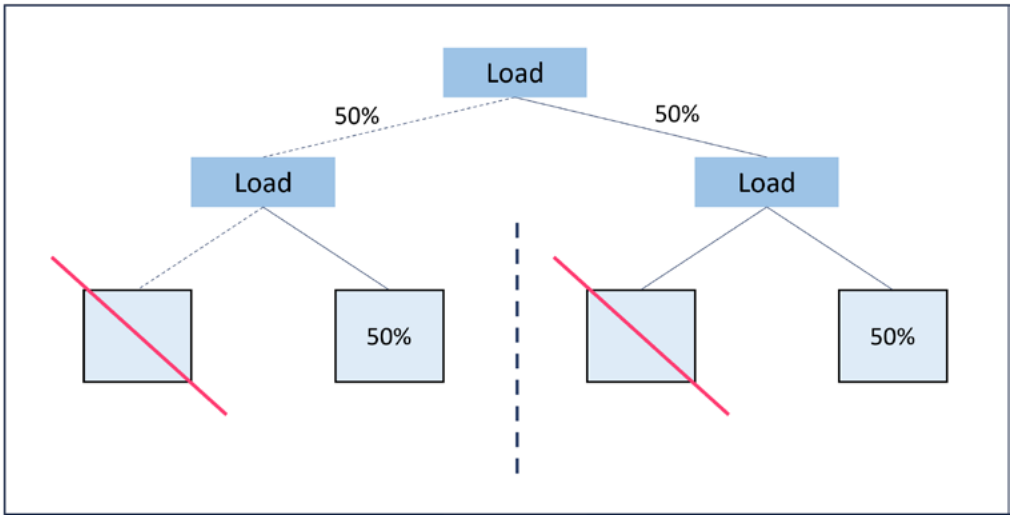


*Abbildung 4: Failover Beispiel 3*

Die Gesamtkapazität beträgt 100 % der regulären Last.

Auswirkungen:
Die verbleibenden Server in Data Center 1 und Data Center 2 müssen jeweils 50 % der Last tragen. Durch diese Dimensionierung wird sichergestellt, dass innerhalb der Infrastruktur genügend Kapazität auch für dieses Szenario vorhanden ist. Der Load Balancer in den einzelnen Rechenzentren stellt automatisch keine Sitzungen mehr für den ausgefallenen Server bereit. Nach dem Auftreten dieses Fehlers ermöglicht eine manuelle „Out of Service“-Konfiguration eine unabhängige administrative Arbeit auf dem fehlerhaften Server. Nach Behebung des Problems wird die „Out of Service“-Konfiguration zurückgesetzt und nach einiger Zeit werden Keepalive-Kommunikationen erkannt und die Lastverteilung wird automatisch erreicht.

# 2 | EIN PARADIGMENWECHSEL

## 2.1 | Die etablierte Art Software zu entwickeln und zu betreiben

Bei herkömmlichen On-Premise Systemen wird die IT-Infrastruktur und die auf dieser Infrastruktur betriebenen Dienste mit Blick auf einen langen Lebenszyklus aufgebaut. In dieser Art von Umgebungen ist es nicht ungewöhnlich, dass virtuelle Server und Anwendungen eine Lebensdauer von manchmal bis zu einem Jahrzehnt haben. Wer aber heute länger als nur ein paar Jahre in der Branche tätig ist, kennt die Anwendungsdienste, die Anfang der 2010er Jahre eingeführt wurden und heutzutage immer noch auf denselben Servern laufen, während einige dieser Systeme seither vielleicht mit nur ein paar Sicherheits-Patches aktuell gehalten wurden.

Der wichtigste Paradigmenwechsel, der diese Art der Bereitstellung und des Betriebs von Anwendungen in Frage stellt, lässt sich mit der Analogie „Rinder gegen Haustiere“ (im Original “Cattle vs. Pets”, vgl. auch Kapitel 2.3) beschreiben, die der ehemalige Microsoft-Ingenieur Bill Baker[3] geprägt hat. In gewisser Weise werden die zugrundeliegenden Server und auch der Software-Stack wie eine Gruppe von Haustieren behandelt, die man hegt und pflegt. Haustiere sind schwer zu ersetzen, man schenkt ihnen viel Liebe und Aufmerksamkeit und jedes ist auf seine Weise einzigartig.

Was bedeutet das technisch gesehen?

- **Feste Identitäten:** Anwendungen verlangen von Servern, dass sie eine feste Identität (wie einen Hostnamen oder sogar eine IP-Adresse) über Monate, manchmal sogar mehrere Jahre, behalten.
- **Veränderbare Konfiguration:** Administratoren nehmen dauerhafte Änderungen an einzelnen Servern vor, um betriebliche Probleme zu beheben. Diese Änderungen werden (meist) sorgfältig in einem Change Management System dokumentiert und dann (manchmal) manuell in anderen Phasen oder beim Wiederaufbau eines Systems nach dessen Totalverlust repliziert.
- **Manuelle Bereitstellung:** Die Skalierung eines Anwendungsclusters zur Anpassung an die Leistungsanforderungen des Kunden erfordert eine mehrwöchige Vorankündigung, damit neue Server vorbereitet und in das Netz- und Gerätemanagement integriert werden können.
- **Virtualisierungs-Overhead:** Einige Anwendungen können aufgrund widersprüchlicher Software-Abhängigkeiten nicht effizient denselben virtuellen Server nutzen. Viele Ressourcen werden reserviert und zugewiesen, aber selten vollständig genutzt.
- **Monolithische Anwendungen:** Anwendungen bestehen aus monolithischen Einheiten, die einen großen Teil der Anwendungsfunktionen in sich vereinen und daher auf einmal bedient und aktualisiert werden müssen. Selbst eine kleine Änderung einer Funktion in einem solchen System führt zu einer vollständigen Aktualisierung des gesamten Systems einschließlich der entsprechenden Prozesse (z. B. Qualitätssicherung). Das Ausrollen neuer Funktionen oder kleinerer Verbesserungen in die Produktion birgt daher ein hohes Risiko von Ausfallzeiten und Nebeneffekten, die sich auf den gesamten Anwendungsdienst auswirken und ist mit hohem Aufwand verbunden.

[3] Details auf http://cloudscaling.com/blog/cloud-computing/the-history-of-pets-vs-cattle (Zugriff am 16.02.2023)

## 2.2 | Herausforderungen in einer Cloud-Umgebung

War alles, was wir in den letzten 20 Jahren getan haben, ist dumm und einfältig? Ganz und gar nicht: In einer traditionellen lokalen Umgebung gibt es keine anderen Optionen, die einen stabilen und sicheren Betrieb garantieren könnten. Ihr On-Premise-Anwendungsdienst ist wie ein Einfamilienhaus. Einmal aufgebaut, ist es schwierig, es abzureißen oder neue Einrichtungen hinzuzufügen. Die Instandhaltung des Hauses erfordert einen hohen manuellen Aufwand, und manche Änderungen sind zu kostspielig und invasiv, da sie das Haus für mehrere Wochen unbewohnbar machen würden.

Unter diesen Umständen können wir mit unserem alten Ansatz, Anwendungen zu erstellen und zu betreiben, die Vorteile einer typischen Cloud-Plattform nicht nutzen. Hier sind nur einige Beispiele für Nachteile, mit denen wir konfrontiert werden, wenn wir versuchen, die etablierten Verfahren in eine Cloud-Umgebung zu übertragen:

- **Skalierbarkeit:** Neue Server können in Sekundenschnelle bereitgestellt und je nach Anzahl der Endnutzer automatisch skaliert werden. Aber wie soll das funktionieren, wenn das Hinzufügen eines neuen Servers zum Anwendungscluster eine ganze Woche manueller Installations- und Konfigurationsaufgaben erfordert?
- **Verteilung:** Cloud-Anbieter bieten Rechenzentren in allen Teilen der Welt an. Aber wie können Sie Anwendungen schnell einrichten oder migrieren, wenn die Installation eines Stacks ein mehrmonatiges Projekt erfordert?
- **Ausfallsicherheit:** Typische Cloud-Anbieter ermöglichen die flexible Verteilung Ihrer Anwendung auf mehrere verschiedene Verfügbarkeitszonen. Wenn ein Rechenzentrum ausfällt, können zwei andere den Betrieb übernehmen. Aber wie würden Sie dies effizient tun, wenn ein Failover-Verfahren die Interaktion vieler verschiedener Teams und die Durchführung manueller Aufgaben erfordert?

> ***„Everything fails all the time."***
> *WERNER VOGELS*

## 2.3 | Den Elefanten in seine „Einzelteile zerlegen"

Um auf die ursprüngliche Analogie zurückzukommen: Um schneller vorankommen zu können, müssen wir unsere Infrastruktur wie eine Viehherde behandeln: Eine namenlose Kuh ist leicht zu ersetzen, man schenkt ihr nicht viel Liebe und Aufmerksamkeit und sie ist ein gewöhnliches Tier. Wenn in einer großen Rinderherde mit Tausenden von Tieren eines von ihnen stirbt, kann der Viehzüchter am Ende des Tages immer noch Milch und Fleisch liefern. Moderne, Cloud-native Anwendungen sind so konzipiert, dass sie vom Ausfall eines Subsystems oder eines Teils der Infrastruktur nicht beeinträchtigt werden.

Ein notwendiger Schritt zur Überwindung der zuvor beschriebenen Einschränkungen besteht darin, eine monolithische Anwendung in mehrere, kleinere Dienste aufzuteilen. Bei diesem Architekturmuster, das allgemein als „Microservices" bekannt ist, konzentriert sich jede kleine Komponente auf die Bereitstellung einer bestimmten Funktionalität im Backend oder Frontend einer Anwendung. Andere Microservices oder Anwendungen können diese Dienste über eine gut dokumentierte und gepflegte API nutzen. Was sind nun die Vorteile?

- Die Nebeneffekte einer Codeänderung sind begrenzt und leichter zu beheben. Anwendungen können häufiger und mit geringerem Risiko aktualisiert werden, da die Trennung der Bereiche den Explosionsradius von möglicherweise neu eingeführten Problemen begrenzt.
- Spezifische Engpässe einer Anwendung können durch die Skalierung nur eines Microservices anstelle aller beteiligten Komponenten entschärft werden.
- Es ist viel einfacher, einzelne Mikrodienste mit wenigen Abhängigkeiten und Einschränkungen auf mehrere Server zu verteilen, als eine monolithische Anwendung im verteilten Modus zu betreiben.

Das bringt uns zu der Frage, wie eine solche Idee umgesetzt werden kann. Wir werden feststellen, dass dies einen Ansatz erfordert, bei dem ein Ziel im Mittelpunkt steht: **Automatisierung**.

# 3 | AUTOMATISIERUNG

## 3.1 | Container

Eine der wichtigsten Entwicklungen, die ein ganzes Ökosystem hervorgebracht hat, war die Einführung von Docker[4] in den frühen 2010er Jahren. Docker nahm ein bestehendes, mehrere Jahrzehnte altes Prinzip und schuf eine benutzerfreundliche Lösung, um es zu nutzen: Linux-Container. Es handelt sich dabei um eine standardisierte Möglichkeit, Anwendungen in unveränderliche Vorlagen zu verpacken, die dann auf jedem anderen Rechner mit kompatibler CPU-Architektur ausgeführt werden können.

Wie der Name schon sagt, sind die Anwendungen in einem Container „in sich geschlossen“. Der Container ist nicht nur mit den ausführbaren Dateien einer Anwendung ausgestattet, sondern auch mit allen Abhängigkeiten, die sie möglicherweise benötigt: spezifische Versionen der erforderlichen Bibliotheken, Standardkonfigurationsdaten, Installationsverfahren und vielleicht einige Stammdaten, die für das Funktionieren der Anwendung wichtig sind. Außerdem sind Container unveränderlich und zustandslos. Von außen betrachtet sehen sie alle gleich aus - wie Kühe in einer Rinderherde. Jeder laufende Container einer Gruppe von Containern kann einen anderen Container ersetzen. Da es sich bei einem Container nur um einen isolierten Satz von Programmen, Software-Abhängigkeiten und Konfigurationsvorlagen handelt, ist für die Ausführung der containerisierten Anwendung keine vollständige virtuelle Maschine erforderlich. Auf diese Weise können mehrere Container auf einer einzigen virtuellen Maschine ausgeführt werden, selbst wenn sie widersprüchliche Voraussetzungen oder Software-Abhängigkeiten haben.

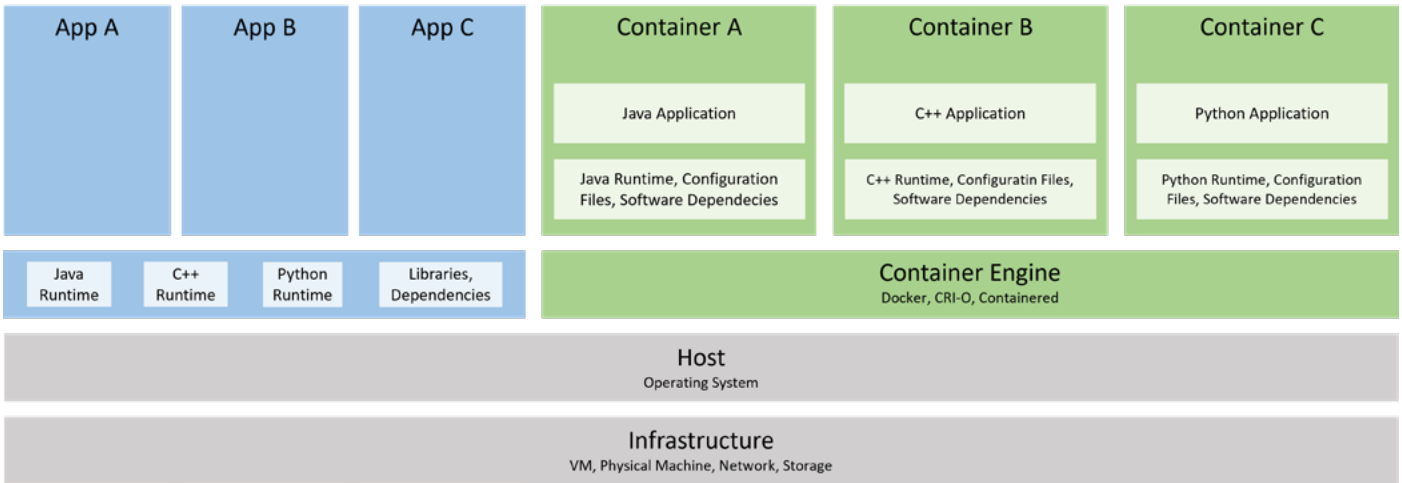


*Abbildung 5: Native und containerbasierte Anwendungen*

[4] Details auf https://www.docker.com (Zugriff am 16.02.2023)

Was bedeutet dies in Bezug auf Ausfallsicherheit und Verfügbarkeit? Wenn ein einzelner Prozess ausfällt, ist nur sein Container betroffen. Andere Container desselben Typs können weiterhin eingehende Anfragen bearbeiten. Außerdem gehen keine Daten verloren, wenn ein Container den relevanten Zustand (d. h. die vom Benutzer in einem Formular bereitgestellten Daten) auf einem externen Speicher aufbewahrt, den er mit den anderen Containern teilt. Wenn eine Anwendung während bestimmter Stunden des Tages stark ausgelastet ist, kann die Anwendung skaliert werden, indem zusätzliche Container ohne manuellen Aufwand gestartet werden, um die Nachfragespitzen abzudecken. Nach Erreichen der Spitze können die nicht mehr benötigten Container automatisch deaktiviert werden.

## 3.2 | Container-Orchestrierung

Bisher haben wir erörtert, wie Container eine Lösung für die Paketierung, Verteilung und Ausführung von Arbeitslasten sind, aber es fehlen noch einige Teile. In einem realen Szenario wollen wir diese Container verteilt ausführen, um einen hochverfügbaren und skalierbaren Anwendungsdienst aufzubauen. Daher benötigen wir neben dem Prinzip der Container eine weitere Schicht, die alle diese Container verwaltet.

Stellen Sie sich eine Rinderherde vor, die auf einer Wiese ohne Zäune, ohne Tore, ohne Viehzüchter und ohne Ställe herumtobt. An diesem Punkt brauchen Sie einen Cowboy, der eine Farm einrichtet und die Kontrolle über die Verwaltung unserer großen Containerherde übernimmt. Hier wird das Prinzip der Container-Orchestrierung eingeführt.

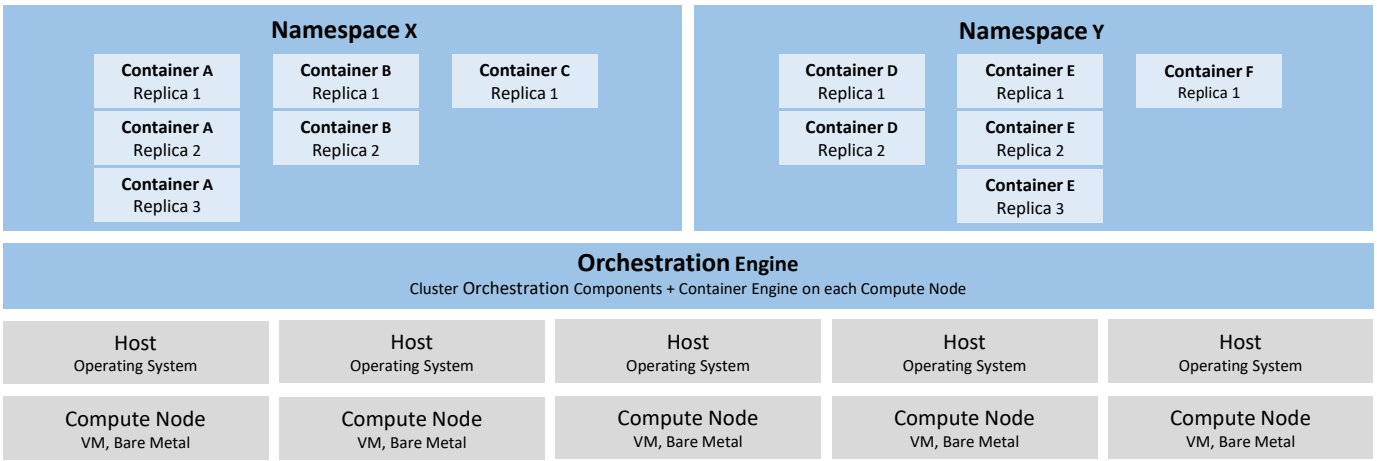


*Abbildung 6: Container Orchestrierung in einem Cluster*

Hier die wichtigsten Aufgaben im Rahmen Container-Orchestrierung und damit verbundenen Automatisierung:

- **Infrastruktur:** Wo können wir einen Rechenknoten finden, der noch Ressourcen für den Betrieb des Containers hat? Wohin kann ein Container migriert werden, wenn ein Knoten nicht mehr funktioniert?
- **Sicherheit:** Welche Anmeldeinformationen benötigt eine Anwendung, um sich mit einem entfernten Dienst zu verbinden? Wer ist in der Lage, Änderungen an der Konfiguration von Containern vorzunehmen?
- **Vernetzung:** Welche Container dürfen mit welchen anderen Containern kommunizieren? Wie entdecken Container andere Dienste?
- **Speicherung:** Wie werden Container-Workloads mit Speichersystemen (wie File Storage oder Object Storage) verbunden? Was passiert mit persistenten Daten, wenn ein Container auf einen anderen Knoten verschoben wird?
- **Vorgänge:** Welche Schritte sind zu unternehmen, wenn ein Container nicht rechtzeitig einsatzbereit ist oder nicht mehr reagiert? Wie kann eine Reihe von Containern aktualisiert werden, ohne dass eine Ausfallzeit der Anwendung für die Nutzer und Clients spürbar wird?
- **Verkehr:** Wohin sollen die Anfragen der Endnutzer geleitet werden? Welcher Container ist bereit, Benutzeranfragen entgegenzunehmen?

# 4 | IBM CLOUD PAK FÜR DIE GESCHÄFTSAUTOMATISIERUNG

Das IBM Cloud Pak for Business Automation[5] besteht aus einer Reihe von Komponenten, die es Kunden weltweit ermöglichen, ihre Geschäftsprozesse zu automatisieren. Cloud Paks nutzen die Vorteile von KI-Technologie, anpassbare "Low-Code" oder sogar "No-Code"-Funktionen und Datenerfassung zur Gewinnung von Erkenntnissen aus den tatsächlich ablaufenden Prozessen. Dadurch können sich die Mitarbeitenden auf wertschöpfende Aufgaben konzentrieren und nicht auf die sich wiederholenden oder sogar langweiligen Aufgaben.

Im Folgenden finden Sie eine Liste ausgewählter Komponenten des IBM Cloud Pak for Business Automation (CP4BA) und deren Verwendung:

- **Business Automation Navigator:** Front-End für Cloud Pak für Business Automation Komponenten
- **FileNet Content Manager:** Digitalisierung von Inhalten und Verwaltung der Lebenszyklen von Inhalten
- **Business Automation Workflow:** Orchestrierung und Visualisierung von Geschäftsprozessen
- **Automation Workstream Services:** Codefreie Lösung zur Rationalisierung, Automatisierung und Beschleunigung von Aktivitäten
- **Automation Decision Services:** Erstellen, Verwalten und Ausführen von Entscheidungsdiensten
- **Automation Document Processing:** Extraktion von Geschäftswissen aus Dokumenten mit Hilfe von Techniken des maschinellen Lernens
- **Business Automation Insights:** Ereignisverarbeitung, Dashboarding und Data Lake-Funktionen

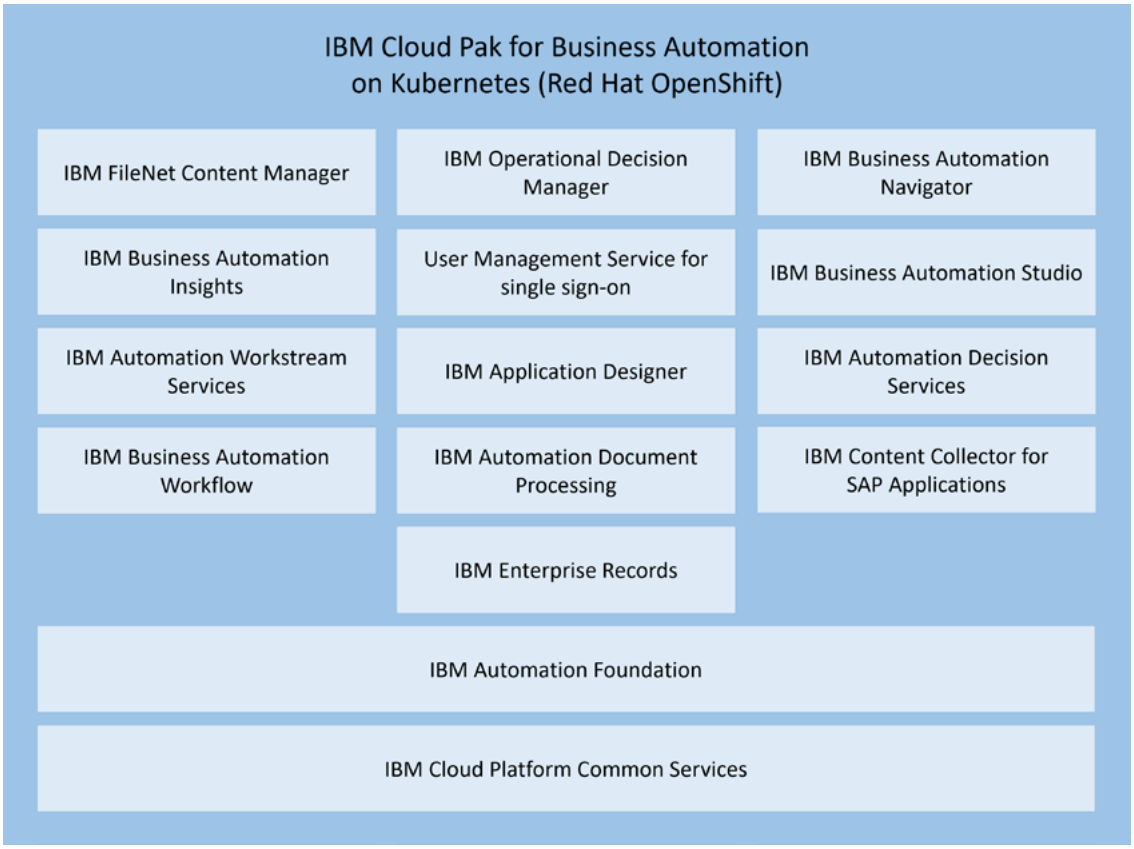


*Abbildung 7: IBM CP4BA Komponenten auf Basis von Red Hat OpenShift[6]*

[5] Details auf https://www.ibm.com/docs/en/cloud-paks/cp-biz-automation (Zugriff am 16.02.2023)
[6] Details auf https://www.redhat.com/en/technologies/cloud-computing/openshift (Zugriff am 16.02.2023)

## 4.1 | Cluster- und Pod-Skalierbarkeit

Das Cloud Pak for Business Automation basiert auf Red Hat OpenShift und nutzt daher mehrere Funktionen einer Kubernetes[7] Umgebung. Die Bereitstellung erfolgt über einen Operator, der auch die Skalierung der Komponenten entsprechend den Betriebsanforderungen ermöglicht.
Der Installationsprozess des Cloud Pak for Business Automation gliedert sich in die folgenden Schritte:

- **Planung:** Einsatz und Clustertyp auswählen
- **Bediener-Installation**
  - Option a:
    Installation über IBM-Katalog / Formularansicht in OpenShift
  - Option b:
    Ausführen von Skripten zur Einrichtung und Bereitstellung von Clustern
- **Bearbeiten und Anwenden** einer benutzerdefinierten Ressource

Die Haupteinrichtung und die Auswahl der zu verwendenden Cloud Pak for Business Automation-Komponenten erfolgt innerhalb Ihrer benutzerdefinierten Ressourcenkonfiguration. In dieser einfachen YAML-Datei werden die Komponenten und ihre detaillierte Konfiguration aufgeführt. Darüber hinaus werden die Anzahl der Instanzen, ihre erforderliche Speicher- und CPU-Auslastung sowie entsprechende Grenzwerte angegeben. Wenn die benutzerdefinierte Ressource ihrem Cluster zugewiesen wird, führt der Operator die Installation aus.

Wenn Sie möchten, dass Ihr Deployment in Bezug auf die Skalierung seiner Komponenten dynamisch ist, können Sie Kubernetes-Funktionen wie die Cluster-Autoskalierung[8] oder die horizontale Pod-Autoskalierung[9] nutzen.

Cluster-Autoskalierung bedeutet, dass Kubernetes Ihren Cluster (die Anzahl der Arbeitsknoten[10] ) automatisch vergrößert, wenn die Anforderungen steigen. Kubernetes skaliert ihn aber auch automatisch herunter, wenn die zusätzlichen Knoten nicht mehr benötigt werden. Dasselbe ist auf der Ebene eines Pods[11] möglich: Wenn eine Komponente viele Anfragen erhält, kann der horizontale Pod-Autoscaling-Mechanismus automatisch zusätzliche Pods (sie werden Replikate genannt) erstellen, um alle Anfragen zu erfüllen. All dies wird mit Hilfe von Load-Balancing[12] gehandhabt.

Ein weiterer wichtiger Aspekt einer Installation des Cloud Pak for Business Automation in einem Cluster ist die Speicherung, insbesondere im Hinblick auf eine hohe Verfügbarkeit. Als Beispiel sei hier der Amazon Simple Storage Service[13] (S3) genannt. Dies ist ein Cloud-basierter Objektspeicher, der in Kombination mit dem Cloud Pak for Business Automation verwendet werden kann. Ohne auf die Details von S3 einzugehen, können Sie die S3-Replikation[14] abonnieren, die eine automatische Replikation von S3-Objekten über verschiedene AWS-Regionen hinweg gewährleistet. Im Falle eines plötzlichen Ausfalls eines Buckets wird ein replizierter Bucket verwendet. Dies führt zu einem kontinuierlichen Hintergrundspeicherdienst für eine ”Cloud Pak for Business Automation“.

[7] Details auf https://kubernetes.io (Zugriff am 16.02.2023)
[8] Details auf https://kubernetes.io/blog/2016/07/autoscaling-in-kubernetes (Zugriff am 16.02.2023)
[9] Details auf https://kubernetes.io/docs/tasks/run-application/horizontal-pod-autoscale (Zugriff am 16.02.2023)
[10] Details auf https://kubernetes.io/docs/concepts/architecture/nodes (Zugriff am 16.02.2023)
[11] Details auf https://kubernetes.io/docs/concepts/workloads/pods (Zugriff am 16.02.2023)
[12] Details auf https://kubernetes.io/docs/concepts/services-networking (Zugriff am 16.02.2023)
[13] Details auf https://aws.amazon.com/s3 (Zugriff am 16.02.2023)
[14] Details auf https://aws.amazon.com/s3/features/replication (Zugriff am 16.02.2023)

# 5 | ZUSAMMENFASSUNG

In dem vorliegenden Whitepaper wurden die Herausforderungen für ECM-bezogene Infrastrukturen anhand bewährter Verfahren beschrieben (Kapitel 1). Doch Technologien entwickeln sich weiter und damit auch die Lösungen für ECM-Infrastrukturen. Unter Bezugnahme auf den beschriebenen Paradigmenwechsel (Kapitel 2) ist heute eine Grundlage für eine fortgeschrittene Automatisierung von Unternehmen verfügbar (Kapitel 3). Früher oder später wird auch Ihr Unternehmen vor ähnlichen Automatisierungsherausforderungen stehen, so dass es immer eine gute Idee ist, die Potenziale von Technologien und deren möglichen Nutzen zu kennen. Wir hoffen, dass dieser Beitrag einige tiefere Einblicke in diese Themen vermitteln konnte.

# ÜBER UNS

ISR gestaltet die nachhaltige Digitalisierung. Innovative Technologien aus den Bereichen Product Lifecycle Management, Digitale Fabrik und Enterprise Information Management schaffen dafür die Basis.

Um in den Kernthemen Dokumentenlogistik und Analytics die #1 im Markt zu werden, hat die CENIT-Gruppe 2022 eine Mehrheitsbeteiligung an der ISR Information Products AG erworben, unter deren Branding diese Themen noch stärker profiliert werden.

Mit CENIT und ISR an Ihrer Seite bringen Sie die Optimierung Ihrer horizontalen und vertikalen Geschäftsprozesse voran, digitalisieren Ihre Dokumente und nutzen das volle Potenzial Ihrer Unternehmensinformationen.

Die Kompetenz der Berater beider Unternehmen entsteht aus der Kombination von fachübergreifendem Prozessverständnis und tiefer Fachexpertise. Der durchgängige Beratungsansatz gibt Ihnen Sicherheit, dass Lösungen entstehen, die optimal zu Ihrer spezifischen Wertschöpfungskette passen.

Als ganzheitlich aufgestellter Partner übernehmen wir die Verantwortung von der individuellen Beratung über die Implementierung innovativer IT-Lösungen bis zum wirtschaftlichen Betrieb und begleitenden Trainings. Mit rund 250 Expert*innen von CENIT und ISR stellen wir uns auf die spezifische Situation Ihres Unternehmens ein und gewährleisten damit die notwendige Praxisnähe für messbare, operative Optimierungen. Als Team erschließen wir gemeinsam für Ihr Unternehmen das Erfolgspotential der Digitalisierung. Die Erfahrungen sprechen für sich: CENIT ist 35 Jahre und ISR bereits 30 Jahre am Markt. Namhafte Kunden in Schlüsselindustrien der Wirtschaft vertrauen auf die Expertise der CENIT und ISR als Trusted Advisor. Als Team erschließen wir gemeinsam für Ihr Unternehmen das Erfolgspotential der Digitalisierung und sichern so ihren Vorsprung am Markt.

ISR
A CENIT COMPANY

# KONTAKTIEREN SIE UNS!

## ISR INFORMATION PRODUCTS AG

Hintern Brüdern 23
38100 Braunschweig
Tel. +49 (0) 531 12 08-0
hello@isr.de



Braunschweig | Münster | Hamburg | Köln | München | Frankfurt a. M.