TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Fakultät Informatik

Automatisierte Provisionierungsmechanismen für Laufzeitumgebungen von Legacy z/OS Anwendungen mit „IBM Cloud Provisioning and Management for z/OS“ am Beispiel der „Rechnungsschreibung“ bei DATEV e.G.

Bachelorarbeit im Studiengang Informatik

vorgelegt von

David Krug

Matrikelnummer 3036355

Erstgutachter:	Prof. Dr. Korbinian Riedhammer
Zweitgutachter:	Prof. Dr. Friedhelm Stappert

Hinweis: Diese Erklärung ist in alle Exemplare der Abschlussarbeit fest einzubinden. (Keine Spiralbindung)

TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Prüfungsrechtliche Erklärung der/des Studierenden

Angaben des bzw. der Studierenden:

Name: Vorname: Matrikel-Nr.:

Fakultät: Studiengang:

Semester:

Titel der Abschlussarbeit:

Ich versichere, dass ich die Arbeit selbständig verfasst, nicht anderweitig für Prüfungszwecke vorgelegt, alle benutzten Quellen und Hilfsmittel angegeben sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Ort, Datum, Unterschrift Studierende/Studierender

Erklärung zur Veröffentlichung der vorstehend bezeichneten Abschlussarbeit

Die Entscheidung über die vollständige oder auszugsweise Veröffentlichung der Abschlussarbeit liegt grundsätzlich erst einmal allein in der Zuständigkeit der/des studentischen Verfasserin/Verfassers. Nach dem Urheberrechtsgesetz (UrhG) erwirbt die Verfasserin/der Verfasser einer Abschlussarbeit mit Anfertigung ihrer/seiner Arbeit das alleinige Urheberrecht und grundsätzlich auch die hieraus resultierenden Nutzungsrechte wie z.B. Erstveröffentlichung (§ 12 UrhG), Verbreitung (§ 17 UrhG), Vervielfältigung (§ 16 UrhG), Online-Nutzung usw., also alle Rechte, die die nicht-kommerzielle oder kommerzielle Verwertung betreffen.

Die Hochschule und deren Beschäftigte werden Abschlussarbeiten oder Teile davon nicht ohne Zustimmung der/des studentischen Verfasserin/Verfassers veröffentlichen, insbesondere nicht öffentlich zugänglich in die Bibliothek der Hochschule einstellen.

Hiermit ☐ genehmige ich, wenn und soweit keine entgegenstehenden Vereinbarungen mit Dritten getroffen worden sind,

☐ genehmige ich nicht,

dass die oben genannte Abschlussarbeit durch die Technische Hochschule Nürnberg Georg Simon Ohm, ggf. nach Ablauf einer mittels eines auf der Abschlussarbeit aufgebrachten Sperrvermerks kenntlich gemachten Sperrfrist

von Jahren (0 - 5 Jahren ab Datum der Abgabe der Arbeit),

der Öffentlichkeit zugänglich gemacht wird. Im Falle der Genehmigung erfolgt diese unwiderruflich; hierzu wird der Abschlussarbeit ein Exemplar im digitalisierten PDF-Format auf einem Datenträger beigefügt. Bestimmungen der jeweils geltenden Studien- und Prüfungsordnung über Art und Umfang der im Rahmen der Arbeit abzugebenden Exemplare und Materialien werden hierdurch nicht berührt.

Ort, Datum, Unterschrift Studierende/Studierender

Kurzdarstellung

Innerhalb der DATEV eG gewinnen PaaS (Plattform as a Service) Ansätze immer mehr an Bedeutung. Ein Vorteil dabei ist die unkomplizierte, automatisierte Provisionierung von Laufzeitumgebungen. Im Vergleich dazu ist der Bereitstellungsprozess für legacy z/OS Anwendungen mit vielen manuellen Schritten und vielen Absprachen, auch abteilungsübergreifend, verbunden.

Das Ziel dieser Forschung ist es zu bestimmen, ob die automatische Bereitstellung von Laufzeitumgebungen für legacy z/OS Anwendungen möglich ist. Dazu werden folgende Forschungsfragen gestellt:

1. Ist es technisch möglich mit dem „IBM Cloud Provisioning and Management for z/OS“-Toolkit eine Laufzeitumgebung für legacy z/OS Anwendungen automatisiert bereitzustellen?
2. Wird dadurch der aktuelle Bereitstellungsprozess schneller und auch sicherer?
3. Erzeugt die Nutzung einen Mehrwert bei den Stakeholdern, also den Entwicklerteam und den Administratorenteams?

Um diese Forschungsfragen zu beantworten, wurde zunächst anhand einer Beispielanwendung, der DATEV Rechnungsschreibung, das „IBM Cloud Provisioning and Management for z/OS“-Toolkit untersucht. Das Toolkit bietet zwei Möglichkeiten für die automatisierte Bereitstellung von Laufzeitumgebungen. Es wurde sich für eine dieser Möglichkeiten entschieden, diese wurde implementiert.
Anschließend wurde der dadurch ermöglichte Prozess aufgezeigt und mit dem etablierter Prozess verglichen. Dabei stellte sich heraus, dass der ermöglichte Bereitstellungsprozess schneller und durch weniger Absprachen auch weniger fehleranfällig ist. Dennoch ist die Lösung nicht optimal und das Toolkit bietet weitere Möglichkeiten zur Verbesserung.
Schließlich sind Interviews mit den Stakeholdern bezüglich eines Mehrwertes des neuen Prozesses durchgeführt worden. Sowohl die befragten Entwickler als auch die befragten Administratoren sehen in dem Toolkit eine Chance auf Verbesserung des aktuell etablierter Bereitstellungsprozesses. Jedoch ist die momentane Lösung zwar funktionsfähig, aber noch bezüglich firmenweiten und einfacheren Einsatz zu optimieren.

Auf dieser Grundlage lässt sich sagen, dass das „IBM Cloud Provisioning and Management for z/OS“-Toolkit die automatisierte Bereitstellung von Laufzeitumgebungen für legacy z/OS Anwendungen ermöglicht. Die in dieser Arbeit implementierte Möglichkeit ist nicht optimal, bietet aber bereits einen Mehrwert für die Stakeholder. Weiterführende Forschung könnte sich mit der Implementierung der zweiten Möglichkeit und mit den damit verbundenen weiteren Verbesserungen des Bereitstellungsprozesses beschäftigen.

Inhaltsverzeichnis

Kapitel 1.

Einleitung

„I recently predicted the last mainframe will be unplugged on March 15, 1996“[1] - ein in der Großrechner-Welt bekannt gewordenes Zitat. Es handelt sich um eine 1993 getroffene Vorhersage, nämlich dass der letzte Mainframe, auch Großrechner genannt, am 15 März 1996 abgeschaltet werden wird. Warum war diese Vorhersage falsch? Wieso wird sich im Jahre 2020 dennoch mit dieser Technologie beschäftigt? Und was genau ist ein Großrechner?

In einem Satz ist ein Großrechner[2] ein leistungsstarkes, zentralisiertes Serversystem. In dieser Arbeit wird nur auf Mainframes aus dem Hause IBM eingegangen. Damit ist auch der Technologiestack festgelegt. Das verwendete Betriebssystem ist z/OS, darauf werden Middleware Produkte wie CICS[3], das Datenbanksystem Db2[4] sowie die Messaging Lösung „IBM MQ“[5] betrieben. Als Programmiersprachen werden z.B. COBOL, IBM Assembler, C und C++ verwendet. Seit ca. 1997 ist es theoretisch möglich Java auf dem Mainframe zu verwenden.[6]. (Habe hier versucht es irgendwie zu schaffen, dass klar wird dass es seit 1997 zwar möglich ist aber noch nicht wirklich genutzt wird)

Der IBM Mainframe hat eine lange Geschichte. Vor mehr als fünfzig Jahren wurde der allererste Großrechner, das sog. SSystem/360"vorgestellt. Bis in die 90er Jahre spielte der IBM Mainframe eine Hauptrolle auf dem Computermarkt, dann gewannen zunehmend verteilte Client-Server-Systeme an Bedeutung.[7] Seitdem gilt der Mainframe bereits als „legacy“ und damit als Ältlast"[8].

Wieso also wird sich mit der Mainframe Technologie noch beschäftigt? Eine Antwort: Auf dem Mainframe werden geschäftskritische Anwendungen in der ganzen Welt gehostet. So

[1] [Also 93]
[2] Beschreibung im Absatz 2.1 zu finden
[3] Anwendungsserver, CICS Beschreibung Absatz 2.3.1
[4] Beschreibung Absatz 2.3.2
[5] Beschreibung im Absatz 2.3.3 zu finden
[6] [Stee 03]
[7] [Ceru 03]
[8] [lega 20]

verarbeiten Großrechner auch heutzutage weltweit circa 1,2 Millionen CICS Transaktionen pro Sekunde.[9] Im Vergleich hierzu werden 63.000 Google Suchanfragen pro Sekunde abgesetzt. [10]

Aus der Kombination von hohem Workload, der Abhängigkeit von einem Hersteller (IBM) und dem als veraltet geltenden Technologiestack entstehen jedoch zunehmend Risiken. Es wird immer schwieriger, Nachwuchs in diesem Bereich zu finden. Zum einem, da Mainframe-Know How kaum noch an Universitäten gelehrt wird. Die Seite des Hochschulkomasses[11] liefert z.B. weder für „Mainframe“ noch für „Großrechner“ einen Treffer. Zum anderen ist der demographische Faktor bei den Wissensträgern nicht zu vernachlässigen. Diese sind - wie die Technologien auf dem Mainframe - in die Jahre gekommen und erreichen das Rentenalter. Ein weiteres Problem ist, dass eine Firma, die einen IBM Großrechner mit z/OS betreibt, von dem oben genannten proprietären Technologiestack abhängig ist, dass heißt es existiert eine starke Hersteller- und Plattformabhängigkeit, z.B. in Bezug auf CICS, Db2, IBM-COBOL-Compiler, Assembler.

Offensichtlich betreiben dennoch einige Firmen einen IBM Großrechner. Darunter zählen hauptsächlich Banken, Versicherungen, Fluggesellschaften usw.. Der gemeinsame Nenner dieser Unternehmen ist, dass sich über die Jahre und Jahrzehnte enorme Investitionen auf dem Mainframe angesammelt haben. Die entstandenen Kernsysteme haben hohe Anforderungen an Massendatenverarbeitung, Sicherheitsstandards und Hochverfügbarkeit. All diese Punkte sprechen nach wie vor für die Nutzung eines Großrechners, z.B. auch bei der DATEV e.G.. [IBM 14]

Die DATEV e.G. wurde am 14.02.1966 von 65 Steuerbevollmächtigen gegründet. Sie verfolgten mit der Gründung das Ziel, Buchführungsaufgaben für ihre Mandanten mit Hilfe der neu aufkommenden EDV zu bewältigen. Aufgrund hohen Mitgliederwachstums wurde hierfür bereits 1969 in einen firmeneigenen IBM-Großrechner investiert.[DATE 17] Heute umfasst das Leistungsspektrum der DATEV e.G. unter anderem das Rechnungswesen, Personalwirtschaft, Consulting, IT-Sicherheit, Weiterbildung für ihre Kunden, in erster Linie Steuerberater, Wirtschaftsprüfer und Rechtsanwälte, und deren Mandanten. Ein nicht unbeträchtlicher Teil dieser betriebswirtschaftlichen Anwendungen läuft bis heute auf einem IBM Großrechner im DATEV Rechenzentrum. So werden pro Tag circa 150.000 Batch Jobs[12] und circa 90 Millionen CICS-Transaktionen verarbeitet. Diese Last wird von circa 14.000 aktiven Modulen erzeugt. Wie in der Abbildung 1.1 zu sehen ist, ist COBOL mit circa 46% Prozent die am häufigsten verwendete Programmiersprache am Großrechner bei der DATEV e.G.. Durch diese Module werden unter anderem im Monat circa 11 Millionen Lohnabrechnungen erstellt und circa eine Millionen Umsatzsteuer-Voranmeldungen durchgeführt.

[9] [IBM 19b]
[10] [Sull 16]
[11] [inte]
[12] Beschreibung in Absatz **??**

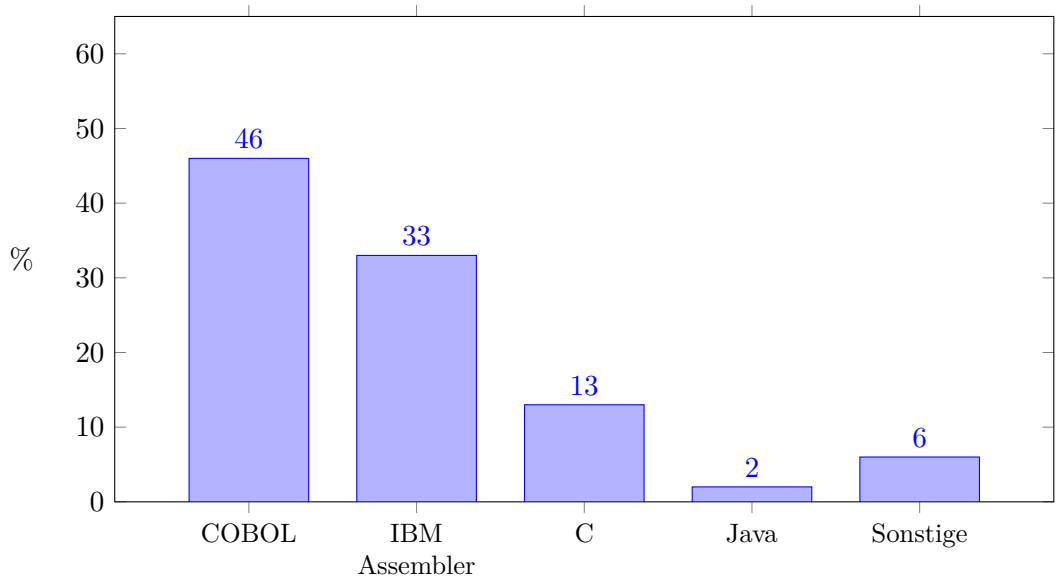


Abbildung 1.1.: Anteil der verwendeten Programmiersprachen auf dem Mainframe bei DATEV eG in Prozent

1.1. Problemstellung

Im Jahre 2020 ist der größte Konkurrent für den Mainframe die Cloud. Laut einer Vorhersage aus dem Jahr 2018[13] soll im Jahre 2020 circa 79 Prozent des weltweiten Workloads in einer Cloud verarbeitet werden. Für die Entwicklung von neuen Online-Anwendungen im cloud-native Stil wurde bei der DATEV e.G. eine Platform-as-a-Service (PaaS)-Lösung geschaffen und neue DevOps Prozesse aufgebaut. Damit können Entwicklerteams unter anderem Datenbanken und Messaginglösungen entweder manuell über einen Marktplatz (siehe Abbildung **??**) oder automatisiert per CI/CD-Pipeline[14] der Laufzeitumgebung ihrer Anwendung hinzufügen. Dadurch ergibt sich für jede einzelne Anwendung der in Abbildung **??** dargestellte Entwicklungsprozess. Wie zu sehen ist liegt der Fokus dabei auf der Entwicklereffizienz und der Verringerung der Zeit bis Änderungen beim Kunden ankommen. Diese CI/CD-Pipeline wurde mittels Jenkins realisiert. Den Entwicklern stehen moderne Entwicklungsumgebungen und git als Sourceverwaltung standardmäßig zur Verfügung.

cloud entwicklungsprozess einfügen!!!

Der Entwicklungsprozess für z/OS Anwendungen bei DATEV e.G. erscheint im Vergleich zu dieser PaaS-Lösung veraltet. So wurde 2010 eine auf Eclipse basierende Entwicklungsumgebung für COBOL und IBM Assembler in der DATEV e.G. flächendeckend bereitgestellt. Zuvor wurde mit Hilfe der in Abbildung 1.2 gezeigten Oberfläche, dem sog. ISPF gearbeitet. Diese stellte z.B. nur ein Syntaxhighlighting zur Verfügung. 2018 wurde git auch für COBOL

[13] Statistik im Anhang A.1
[14] Glossar **??**

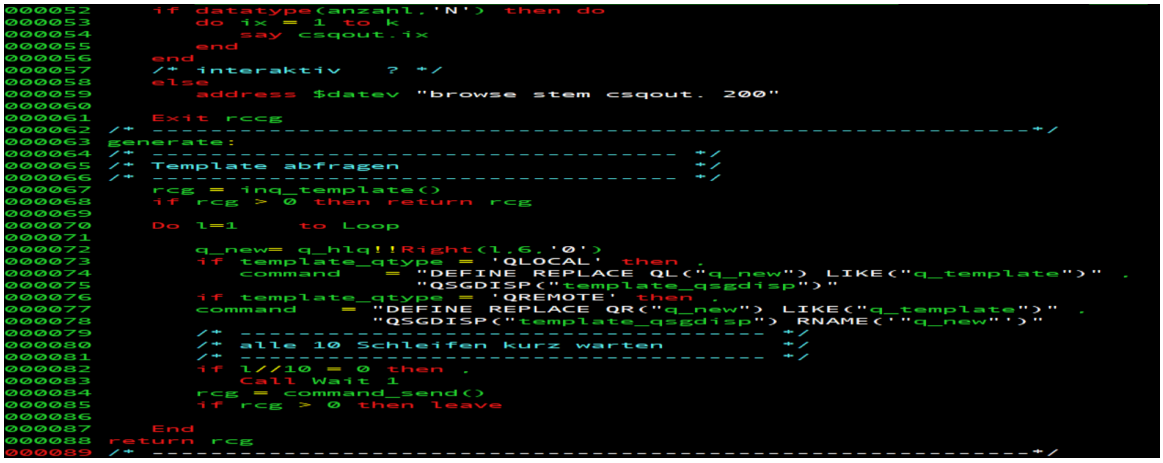


Abbildung 1.2.: Auszug aus einem REXX Skript in der ISPF Oberfläche

und IBM Assembler Sourcen eingeführt. Dieses weit verbreitete Standard-Tool ist im z/OS Umfeld tatsächlich eine entscheidende Neuerung. Dadurch wurde ein bis dato verwendetes eigenentwickeltes Tool für die Sourceverwaltung der z/OS Sourcen abgelöst. Das Tooling wurde somit modernisiert, jedoch nicht der Entwicklungsprozess selbst. Es teilen sich sehr viele Anwendungen die gleichen Test-CICS/Db2/MQ Ressourcen. Das heißt auch, dass eine Parallelentwicklung an unterschiedlichen Features nur mit viel Abstimmungsaufwand und Absprachen innerhalb eines Entwicklungsteams, teilweise auch abteilungsübergreifend, möglich ist. Werden Änderungen an bestehenden Ressourcen durchgeführt oder werden neue Systemumgebungen benötigt, entsteht weiterer Abstimmungsaufwand und weitere Absprachen. Dadurch wird der Prozess fehleranfällig und langsam.

Es bleibt die Frage, wie wird mit den vielen Mainframebestandsanwendungen bei der DATEV e.G. in Zukunft umgegangen? Die komplette Ablösung dieser Anwendungen durch cloud-native Lösungen ist eine Option, deren zeitlicher Rahmen und Machbarkeit aktuell nicht absehbar ist.[15] Für die Funktionsfähigkeit dieses Bestandsgeschäfts, das die Core-Business-Funktionalitäten der DATEV e.G. darstellt, muss also effiziente Weiterentwicklung und Wartung gewährleistet werden. Auch im Falle einer geplanten Ablöse von Anwendungen muss je nach Strategie (z.B. „Rewrite“/ „Rearchitect“)[16] das Alt-System parallel dazu über Jahre oder Jahrzehnte gepflegt und funktional aktuell gehalten werden. Daraus folgt, dass aus Sicht der DATEV e.G. weiter in die IBM Mainframe Plattform investiert werden muss. Dies bedeutet Investitionen in die bereitgestellte Infrastruktur (Hardware, Betriebssysteme, Lizenzen), insbesondere auch Investitionen, die die oben genannten Anforderungen an Weiterentwicklung, Wartung und Entwicklungseffizienz sowie Effizienz im Betrieb adressieren.

[15] **??**

[16] (Kommentar, Strategiepatterns lt. Gartner, ich schick DIr enien Link)

1.2. Ziel der Arbeit

In diesem Zusammenhang läuft aktuell bei DATEV e.G. ein Proof of Concept bezüglich automatisierter Builds von z/OS Anwendungen auf Basis von Jenkins basierten Pipelines. Dies ist auch die Voraussetzung für automatisierte Tests von z/OS Programmen im Rahmen des „Continuous Integration, Continuous Deployment“[17] Ansatzes. Ist dieser Rahmen gegeben ist eine Integration von z/OS Ressourcen in einen Marktplatz möglich. Die dafür notwendige automatisierte Provisionierung einer z/OS Anwendungsumgebung, d.h Laufzeit, Middleware etc., ist aktuell noch weitgehend unerforscht. Hier sind die Prozesse bei DATEV und anderen Kunden oft noch proprietär, hoch spezialisiert, manuell und nicht modernisiert. Gerade bei Mitarbeitern im Betrieb, die als Administratoren für die Middleware arbeiten, sind die Bedenken groß, ob man diese Cloud-Prozesse auf hochspezialisierte individuelle Komponenten wie CICS, DB2, IBM MQ anwenden kann. Die von IBM hier angebotenen Lösungen, die durch das „IBM Cloud Provisioning and Management for z/OS“-Toolkit ermöglicht werden, haben sich noch nicht flächendeckend durchgesetzt, aber es herrscht Interesse an Erfahrungen und Einschätzungen. Hier setzt diese Arbeit an und klärt folgende Fragen:

- Ist es generell technisch möglich mit dem „IBM Cloud Provisioning and Management for z/OS“-Toolkit Laufzeitumgebungen automatisiert bereitzustellen?
- Wird dadurch der aktuelle Bereitstellungsprozess schneller und auch sicherer?
- Erzeugt die Nutzung einen Mehrwert bei den Stakeholdern, also den Entwicklerteams und den Administratorenteams?

Um diese Fragen zu beantworten wird die Provisionierung einer z/OS Laufzeitumgebung für eine spezielle Anwendung untersucht. Die Anwendung sollte ein CICS als Anwendungsserver, eine Db2 Datenbank und IBM MQ als Messaginglösung benötigen, um für diese 3 Haupt-Technologien (Middleware-Komponenten) eine Aussage treffen zu können. Die genaue Vorgehensweise wird im Kapitel 3 beschrieben.

[17] Beschreibung in Absatz **??**

Kapitel 2.

Grundlagen

In diesem Kapitel werden für diese Arbeit wichtige Begriffe erläutert.

2.1. Mainframe / Großrechner

Im modernen Sprachgebrauch kann ein Großrechner oder auch Mainframe als größte zur Verfügung stehende Serverart betrachtet werden. Er wird von Unternehmen verwendet, um kommerzielle Datenbanken, Transaktionsserver und Anwendungen, die einen hohen Grad an Sicherheit und Verfügbarkeit benötigen, zu hosten. Im Gegensatz zu verteilten Serversystemen, bei denen die Funktionalitäten auf einzelne Server, wie zum Beispiel einen E-Mail-Server, einen Datenbank-Server, einen Web-Server usw. aufgeteilt sind, handelt es sich bei einem Mainframe um ein zentralisiertes System. Die einzelnen Funktionalitäten werden von sogenannte „Subsysteme“, auch „Middleware“ genannt, zur Verfügung gestellt. Darunter zählen unter anderem Datenbanksysteme und Anwendungsserver. [Ebbe 11]

2.2. Mainframe Anwendungen bei DATEV e.G.

Das Betriebssystem des IBM Mainframes ist z/OS. Darauf aufbauend benötigen klassische z/OS Anwendungen bestimmte Middleware. Bei der DATEV e.G. handelt es sich unter anderem um folgende Middlewarekomponenten:

- Laufzeitumgebung: CICS oder Batch
- Datenhaltung: VSAM oder Db2
- Message Queuing: IBM MQ

Diese Subsysteme stehen in jeder Stage zur Verfügung. Eine Stage beschreibt eine isolierte Systemumgebung mit eigenen Subsystemen und Ressourcenverwaltung. Die DATEV e.G. unterscheidet vier Stages:

- Testplex:
  Labor für Änderungen am System, beispielsweise einer neuen Betriebssystemsversion
- Entwicklung:
  Implementierung neuer Features und Durchführung kleiner Tests
- Qualitätssicherung:
  Durchführung von Integrationstests
- Produktion:
  Software, die für den Kunden bereitsteht

2.3. Subsysteme / Middleware

Für die Beantwortung der Forschungsfragen liegt der Fokus auf dem Erstellen („Provisionieren “) einer anwendungsspezifischen Laufzeitumgebung mit einer Datenhaltung und Message Queuing innerhalb des Testplexes und der Entwicklung. Als Laufzeitumgebung wird „CICS“, als Datenhaltung „Db2“ und für das Message Queuing „IBM MQ“ verwendet. Diese werden im Folgenden erläutert, hierzu dient Abbildung 2.1 als Überblick.

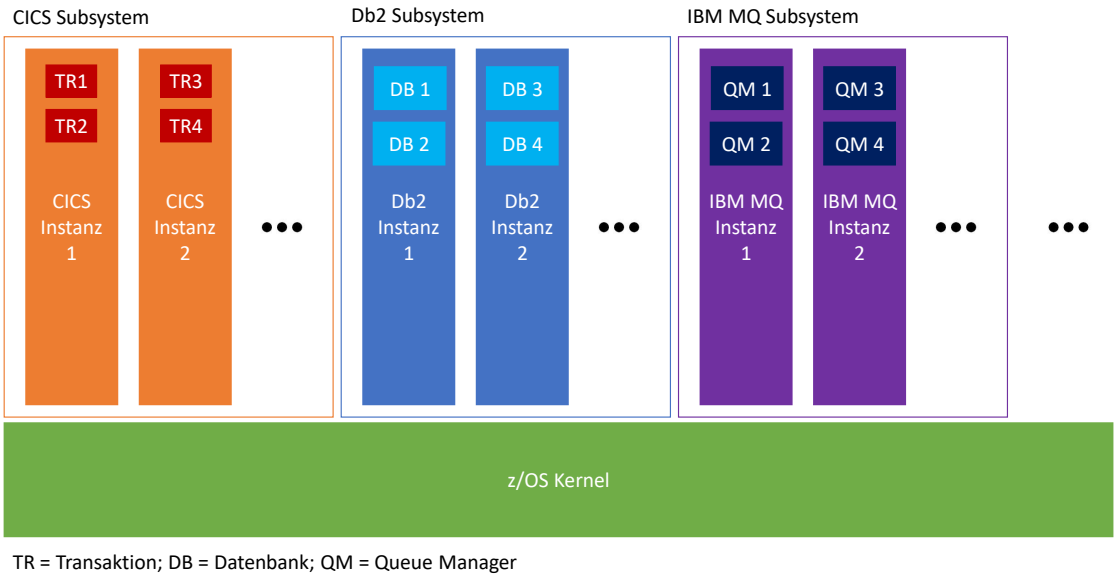


Abbildung 2.1.: Architekturübersicht über die Subsysteme bei DATEV eG

2.3.1. Customer Information Control System

Das Customer Information Control System, kurz CICS, ist ein Applikationsserver für einen IBM-Großrechner mit Betriebssystem z/OS und damit eine IBM Middleware. Ein Applikationsserver stellt eine Umgebung zur Verfügung, in der Anwendungen gehostet werden können. Dabei kümmert sich dieser unter anderem um Transaktionalität, Webkommunikation und Sicherheit. Hierfür stellen Applikationsserver eine API zur Verfügung. CICS hat einen Vorteil gegenüber anderen Anwendungsservern, es unterstützt verschiedene Programmiersprachen. CICS ist ein Multi-Language Application Server und unterstützt z.B. COBOL, Assembler, Java und PLI. So können Programme innerhalb einer Anwendung in der für ihren Use-Case am besten geeigneten Sprache implementiert werden. [Rayn 11]

Das CICS Subsystem einer Stage umfasst mehrere CICS Instanzen.

2.3.1.1. CICS Instanz

Unter einer CICS Instanz ist ein einzelner Bereich, der auf dem z/OS Kernel aufsetzt, zu verstehen. Dieser Bereich ist mittels einer eindeutigen CICS ApplicationID gekennzeichnet und kann darüber explizit angesprochen werden. Eine CICS Instanz verwaltet mehrere CICS Transaktionen.

Wenn in dieser Arbeit von dem CICS gesprochen wird, ist die CICS-Instanz damit gemeint.

2.3.1.2. CICS Transaktion

Ein Businessablauf wird im CICS in einer Transaktion gekapselt. Eine Transaktion kann mehrere Programme unterschiedlicher Programmiersprachen umfassen und wird über eine eindeutige „TransaktionsID“ identifiziert..

Über die TransaktionsID wird der Ablauf gestartet. Dies kann sowohl per Webanfrage oder per Messaging Queue als auch aus einem anderen Programm heraus oder manuell geschehen. In der Transaktion werden alle Änderungen, die Programme an Ressourcen, wie zum Beispiel einer Datenbank oder Dateien tätigen, protokolliert. So wird im Falle eines Fehlers die Möglichkeit eines Rollbacks sichergestellt. [Rayn 11]

2.3.2. Db2

Db2 ist ein relationales Datenbanksystem, welches unter anderem als Subsystem eines z/OS Betriebssystems läuft. Einer Stage können mehrere Datenbanksysteme, auch Instanzen genannt, zugeordnet werden. In einer Instanz befinden sich die Datenbanken und Tabellen.

2.3.3. IBM MQ

IBM MQ ist eine Messaging-Lösung der IBM. Diese ermöglicht den asynchronen Datenaustausch zwischen Anwendungen mittels sogenannter Queues. Alle IBM MQ Begrifflichkeiten, die in dieser Arbeit verwendet werden, werden im Folgenden erläutert. [Aran 13]

Das IBM MQ Subsystem einer Stage setzt sich aus einem oder mehreren Queue Managern zusammen. Ein Queue Manager kann daher als IBM MQ Instanz gesehen werden.

2.3.3.1. Queue Manager

Bei einem Queue Manager handelt es sich um die zentrale Ressource eines IBM MQ Systems. Er verwaltet alle anderen IBM MQ Ressourcen. Dazu gehören unter anderem die Speichersteuerung der Daten und die Wiederherstellung dieser im Falle eines Fehlers. Desweiteren koordiniert er den Zugriff aller Anwendungen auf die Nachrichten in den von ihm verwalteten Queues. Um hierbei die Konsistenz sicherzustellen, sorgt er für Locking und die notwendige Isolation der Queues. [Aran 13]

2.3.3.2. Queues

In Queues werden die Nachrichten, die von Programmen gesendet und gelesen werden gespeichert. Es gibt verschiedene Arten von Queues, die im Kontext dieser Arbeit relevanten Queues sind folgende:

Die Local Queue.
Dabei handelt es sich um die einzige Queue Art, bei der die Nachrichten physikalisch gespeichert werden. Die anderen Queue Arten nutzen als Basis immer eine Local Queue.

Initiation Queue
Die sogenannte „Initiation Queue“ ist eine spezielle Art der Local Queue. Diese dient dem Queue Manager dazu, unter bestimmten Bedingungen eine Trigger-Nachricht darauf zu schreiben. Daher kann eine andere Local Queue so definiert sein, dass sobald eine Nachricht auf sie geschrieben wird eine solche Trigger-Nachricht erzeugt wird. Dies ermöglicht, dass Anwendungen nur starten, wenn wirklich Daten zum Verarbeiten vorhanden sind. [Aran 13]

2.3.3.3. Process

Für das Auslösen von Anwendungen wird nicht nur die Initiation Queue benötigt, sondern auch sogenannte „Processes“. So muss der Local Queue, die den Start einer Anwendung auslösen soll, bei der Definition nicht nur die Initiation Queue bekannt gemacht werden, sondern auch ein Process. Ein Process legt den „Type“ und den Namen der zu startenden Anwendung fest. Als „Type“ können beispielhaft CICS oder auch WINDOWSNT für Windows unterstütze Plattformen genannt werden. Ist der „Type“ CICS, muss der Name der Transaktion angegeben werden, für Windows Plattformen der Dateipfad der auszuführenden exe. [Aran 13]

2.4. „IBM Cloud Provisioning and Management for z/OS“

Die Verwaltung von Subsystemen ist mit vielen manuellen Schritten verbunden.[1] Betreut werden die einzelnen Subsysteme über alle Stages hinweg von eigens dafür entstandenen Administratorenteams. Die „CICS Administration“ kümmert sich um alles rund um das CICS Subsystem. Die „Db2 Administration“ stellt Datenbanken und Tabellen auf Anfrage der Entwickler bereit. Die „IBM MQ Administration“ verwaltet die IBM MQ Ressourcen. Um diese Aufgabe zu bewältigen sind aktuell viele proprietäre und veraltete Tools im Einsatz.

Für die Automation dieser Prozesse bietet die IBM „IBM Cloud Provisioning and Management for z/OS“ an. Dies umfasst zwei Lösungen, „z/OS Provisioning Toolkit“[2] und „z/OS Management Facility“[3]. Bei beiden Lösungen stehen die sogenannten „Templates“ im Mittelpunkt. Mit Hilfe eines Templates können Instanzen erzeugt werden. Diese Instanz kann eine oder mehrere verschiedene Subsystem-Instanzen enthalten.
Ein Template besteht aus drei Dateien:

„Aktiondefinitionfile“
Hier werden die Aktionen, die ein Anwender mit einer Instanz eines Templates durchführen kann, festgelegt. Einer Aktion wird ein sogenannter „Workflow“ zugewiesen.

Ein Workflow ist über eine XML Datei, die sogenannte „Workflowdefinitionfile“, definiert. Diese lässt sich grob in zwei Teile untergliedern:

- Variablendefinition
- Steps

[1] Analyse des aktuellen Bereitstellungsprozesses, siehe in Absatz 4.1
[2] siehe Absatz 2.4.1
[3] siehe Absatz 2.4.2

In der Variablendefinition werden, wie der Name schon sagt, alle Variablen, die für diesen Workflow notwendig sind definiert.

Ein Step beschreibt einen Teilablauf eines Workflows. Ein Workflow kann aus mehreren Steps bestehen. Die Steps werden in Definitionsreihenfolge ausgeführt. Allerdings können Bedingungen für die Durchführung eines Steps definiert werden. So ist es beispielsweise möglich, einen Step nur durchzuführen, wenn eine bestimmte Variable einen bestimmten Wert besitzt. Innerhalb eines Steps können sowohl interne und externe Scripte als auch JCLs und somit Programme ausgeführt werden. Darüber hinaus besteht die Möglichkeit REST-Calls auszuführen. Durch ein XML Schema wird sichergestellt, dass die Workflowdefinitionfile keine syntaktischen Fehler beinhaltet. Sowohl die Variablendefinition als auch die Steps können in externe XML Dateien ausgelagert werden. Dadurch können Variablen an einer Stelle im Template definiert und in alle Workflowdefinitionfiles aufgenommen werden. [Rott 18]

Neben den Workflowdefinitionfiles muss in einer Aktion auch der Pfad der sogenannte „Variableinputfile“ angegeben sein.

„Variableinputfile“
In dieser Datei werden den in der Workflowdefinitionfile definierten Variablen Werte zugewiesen. Somit kann das Template konfiguriert werden.

„Manifest-File“
Hier wird dem Template mitgeteilt, an welchem Speicherort sich die oben genannten Dateien befinden. Da ein Template immer provisioniert werden kann, wird hier auch der Speicherort des Provisionierungsworkflows angegeben. Zusätzlich kann noch eine Beschreibung des Templates hinzugefügt werden. [IBM 19b]

2.4.1. z/OS Provisioning Toolkit

z/OSPT bietet ein Kommandozeileninterface für die Bereitstellung und das Verwalten von Laufzeitumgebungen. In Abbildung 2.2 werden die möglichen Kommandozeilenbefehle mittels des Befehls „zospt -h“ in einem Kommandofenster angezeigt. Mit z/OSPT werden zwei weitere Begriffe eingeführt.

```
$ zospt -h
IBM z/OS Provisioning Toolkit V1.1.5

Usage: zospt [OPTIONS] COMMAND [arg...]

Options:
 --version   : Displays the command line version.
 -h (--help) : Displays the command line help.

Commands:
    build       PATH [-h (--help)] -t (--tag) <imageName>              Build an image
    images      [-h (--help)]                                          List all images
    inspect     <imageName> | <containerName> | <containerId>          Inspect an image or a container
                [-h (--help)]
    rm          <containerName> | <containerId> ... [-f (--force)]     Remove one or more containers
                [-h (--help)]
    rmi         <imageName> ... [-h (--help)]                          Remove one or more images
    run         <imageName> [--draft]                                  Run an image in a new container
                [--link <containerName> | <containerId>:<alias>]
                [--name <containerName>] [-h (--help)] [-q (--quiet)]
    start       <containerName> | <containerId> ... [-h (--help)]      Start one or more containers
    stop        <containerName> | <containerId> ... [-h (--help)]      Stop one or more containers
    ps          [-a (--all)] [-f (--filter) <filter>] [-h (--help)]    List containers

Run 'zospt COMMAND --help' for more information on a command.
```

Abbildung 2.2.: z/OSPT mögliche Kommandozeilenbefehle

„Images“
Dabei handelt es sich grundsätzlich um ein Template, jedoch kann dieses Template über eine weitere Inputdatei verändert werden. Dadurch kann ein Template mit spezifischen Änderungen provisioniert werden, ohne dass ein neues Template erzeugt werden muss. Dies erhöht die Flexibilität der Templates weiter.

„Container“
Dabei handelt es sich um eine Template-Instanz [4]. [IBM 19a]

2.4.2. z/OS Management Facility

Der Hauptaugenmerk dieser Arbeit liegt bei z/OSMF, da damit die Verwaltung von Workflows und Templates über eine browserbasierende Schnittstelle möglich ist. Durch diese Oberfläche, in Abbildung 2.3 dargestellt, ist es einfacher zu bedienen als das Kommandozeileninterface von z/OSPT und somit wird der Einstieg in die Provisionierung erleichtert.

Die linke Seite der Abbildung 2.3 zeigt den Umfang der z/OSMF Funktionen. Für diese Arbeit besitzt nur der Menüpunkt „Cloud Provisioning“ Relevanz. Unter diesem Punkt sind die Funktionalitäten für die automatisierte Bereitstellung von Templates zu finden. [Rott 18]

Zuerst ist das „Resource Management“ zu nennen.. Darunter werden sogenannte „Domains“ und die dazugehörigen „Tenants“ verwaltet. Unter einer „Domain“ ist ein System zu verstehen, dass Systemressourcen in Ressourcenpools gliedert. „Tenants“ sind die dazugehöri-

[4] Beschreibung in Absatz **??**

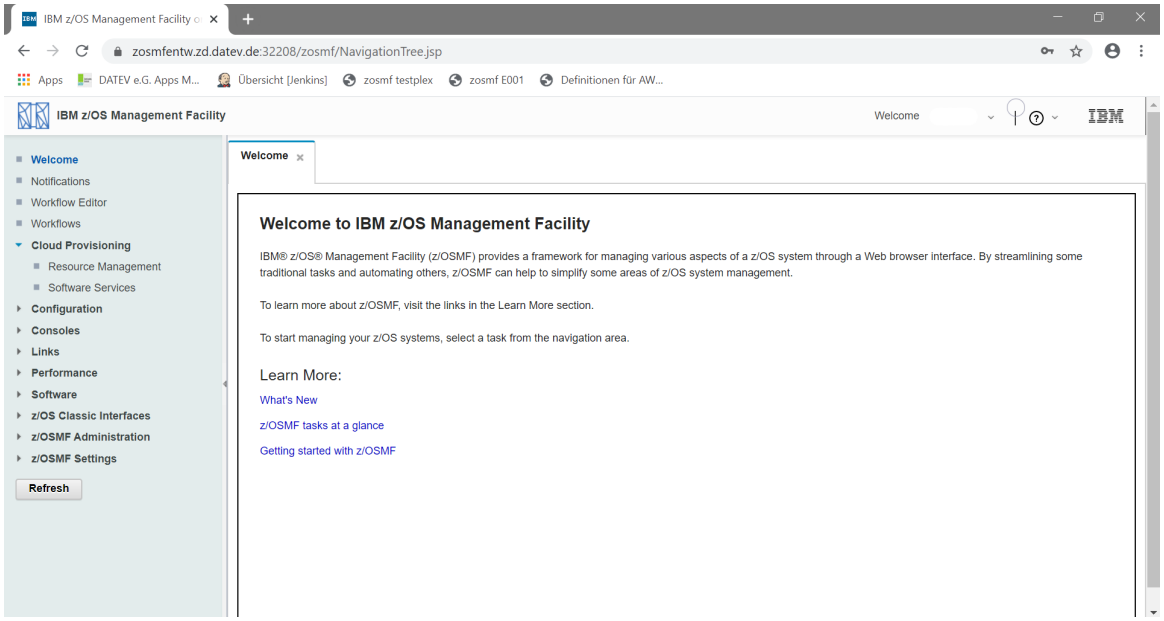


Abbildung 2.3.: z/OSMF Willkomens Ansicht

gen Rechtegruppen, die dem Anwender den Zugriff auf und die Nutzung von zugeordneten Templates ermöglicht. Einem Template muss sowohl eine „Domain“ als auch ein „Tenant“ zugewiesen werden. [Rott 18]

Zur Verwaltung der Templates und Template-Instanzen kommen die „Software Services“ zum Einsatz. Dort können neue Templates über die Manifest-File hinzugefügt werden. Dann muss, wie oben beschrieben, eine „Domain“ und ein „Tenant“ zugwiesen werden. Anschließend kann das Template, falls es keine Fehler beinhaltet, veröffentlicht werden. Es ist zu empfehlen vorher einen „Test Run“ durchzuführen. Dabei wird eine Instance testweise provisioniert. Diese Template-Instanz verhält sich genauso wie eine Instanz, die aus einem veröffentlichten Template erzeugt wurde. Somit können damit das Template und die in der Aktiondefintionfile definierten Aktionen getestet werden. [Rott 18]

Kapitel 3.

Vorgehensweise

Um einen Überblick über die momentan in der DATEV e.G. etablierten Bereitstellungsprozesse für die Middlewarekomponenten CICS, Db2 und IBM MQ zu bekommen, wurden diese analysiert.

Wie in Absatz **??** beschrieben wird für die Beantwortung der Forschungsfragen eine Beispielanwendung, die als Laufzeitumgebung CICS, als Datenbank Db2 und als Message Lösung IBM MQ verwendet, benötigt. Hier bietet sich die „DATEV Rechnungsschreibung“ an. Dabei handelt es sich um eine legacy z/OS Anwendung, ein Teil dieser Anwendung erfüllt die oben genannten Kriterien. Um die genauen Anforderungen an die Middleware ausfindig zu machen, wurde dieser Teil analysiert. Anhand dieser Anforderungen wurde mittels z/OSPT und z/OSMF ein Template bereitgestellt.

Im Folgenden wird speziell auf die Vorgehensweise bei der Implementierung dieses Templates eingegangen. Begonnen wurde auf dem Testplex. Dieser ist komplett von anderen Systemumgebungen abgekapselt, um Auswirkungen von Fehlern in den Tests auf die Entwicklung und Produktion zu vermeiden. In dieser Umgebung wird zunächst untersucht, wie es möglich ist eine CICS-Instanz zu provisionieren. Hierfür wird vorerst ein von der IBM bei der Installation von z/OSMF mitgeliefertes minimales CICS Template verwendet. Anschließend wird ein umfangreicheres mitgeliefertes Template an die Anforderungen der Anwendung angepasst. Für die Provisionierung von Db2 Datenbanken existiert innerhalb der DATEV e.G. bereits eine REST-API. Es wird versucht diese innerhalb des Templates zu nutzen. IBM MQ Queues werden mittels Standard IBM Jobs provisioniert. Da es sich beim Testplex, wie im Absatz 2.2 beschrieben, nur um eine Testumgebung für Systemtests handelt, werden vorerst nur die benötigten Middleware-Systeme provisioniert.

Nachdem eine CICS-Instanz, eine Db2 Datenbank und IBM MQ Queues auf dem Testplex sowohl provisioniert als auch deprovisioniert werden können, folgt der nächste Schritt. Dabei handelt es sich um den Wechsel vom Testplex in die Entwicklungsstage. In der Entwicklungsstage sind alle Anwendungsdaten, die für diese Tests notwendig sind, vorhanden. Somit kann hier die Integration der Beispielanwendung in die provisionierte CICS-Instanz stattfinden.

Es folgten Interviews mit Vertretern aus dem Administratorenteam, Entwicklerteam und dem Team für die Technologiestrategie. Dadurch wird eine Bewertung bezüglich des möglichen Nutzwertes, sowie der Akzeptanz der Technologie durch die Stakeholdern bei DATEV e.G., erfasst.

Kapitel 4.

Analyse

Im Folgendem wird der aktuelle Bereitstellungsprozess für die Laufzeitumgebung, dem dazugehörigen Datenbanksystem und einer Messaging Lösung getrennt voneinander dargestellt. Anschließend erfolgt eine Beschreibung der Beispielanwendung „DATEV-Rechnungsschreibung“. Die dafür benötigten Informationen stammen aus Gesprächen mit Mitarbeiter 1 aus der Abteilung, die für die DATEV-Rechnungsschreibung zuständig ist. Es wird vor allem der technische Aspekt beleuchtet.

4.1. Aktueller Bereitstellungsprozess

Die in diesem Absatz genannten Informationen stammen aus Gesprächen mit Mitarbeitern aus den jeweiligen Administratorenteams. Wie in Absatz 2.2 beschrieben benötigt eine z/OS Anwendung zunächst eine Laufzeitumgebung, im Fall dieser Arbeit handelt es sich um CICS.

4.1.1. Bereitstellung einer CICS-Instanz

Um eine lauffähige CICS-Instanz einzurichten, sind mehrere Schritte notwendig. Der komplette Prozess wird in Abbildung 4.1 dargestellt. Wie zu sehen ist, ist der Initiator des Prozesses das Entwicklerteam. Zunächst wird dort während der Entwicklungsphase festgestellt, dass ein neue CICS-Instanz benötigt wird. Hier hilft das CICS Administratorenteam mittels Beratung aus. Während einer Beratungsphase, die via Telefon, Emails oder Terminen stattfindet, wird sichergestellt, ob wirklich eine neue CICS-Instanz notwendig ist oder ob nicht eine bereits bestehende Instanz genutzt werden kann. Falls eine neue CICS-Instanz benötigt wird, wird ein RACF Eintrag für diese Instanz beantragt. Dieser Eintrag wird dann vom RACF Team erzeugt. Um sicherzustellen, dass die CICS-Instanz in den täglichen Sicherungen enthalten ist, muss das System Automations-Team benachrichtigt werden.

Nun kann mit dem eigentlichen Anlegen der CICS-Instanz begonnen werdem. Dabei müssen folgende Schritte manuell durchgeführt werden. Es werden nur die Schritte, die im Laufe

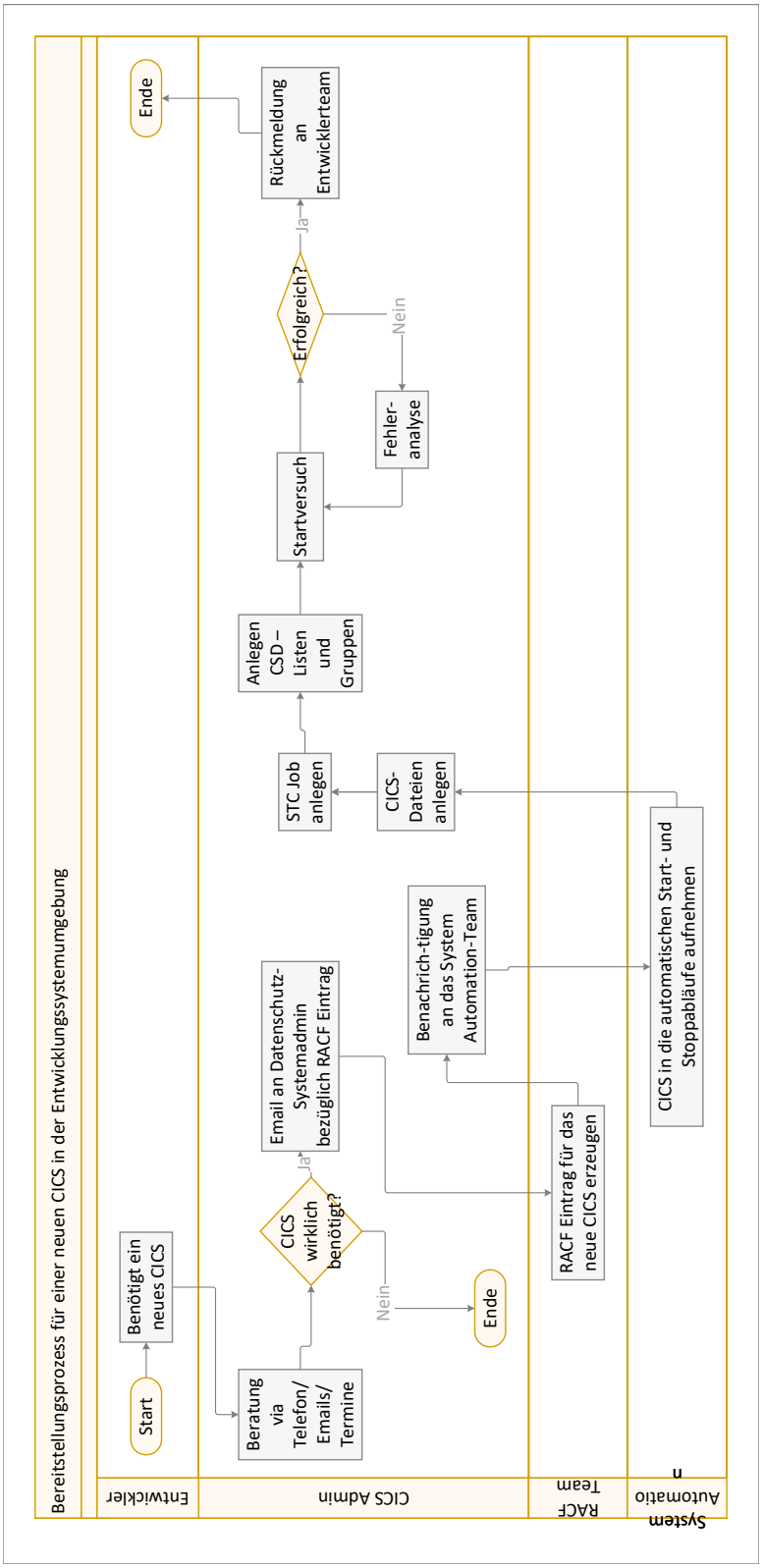


Abbildung 4.1.: Bereitstellungsprozess einer CICS-Instanz

dieser Arbeit durch das „IBM Cloud Provisioning and Management for z/OS“-Toolkit automatisiert werden, dargestellt. Es handelt sich um das Anlegen von:

- CICS spezifische Dateien
- „CICS System Definition“
- Started Task Control-Job

CICS spezifische Dateien

Zunächst müssen CICS spezifische Dateien im z/OS angelegt werden. Im Fall des dieser Arbeit zugrunde liegenden Beispiels handelt es sich um siebzehn verschiedene VSAM[1] Dateien. Diese Dateien benötigt die CICS-Instanz um zum Beispiel Systemfehler zu protokollieren oder den Debugger aktivieren zu können.

„CICS System Definition“

In der Datei „CICS System Definition“, kurz CSD, muss jede Ressource, die dem System zur Verfügung stehen soll, definiert werden. Eine CSD Datei kann für mehrere CICS-Instanzen verwendet werden und besteht aus mehreren Einträgen. Ein Eintrag besteht aus einer Gruppe und einer Liste. Die Gruppe ist hierbei die Definition einer Systemressource und muss manuell angelegt werden. Bei der Liste handelt es sich um das System, welches diese Ressource benötigt. Dort ist unter anderem für jede CICS-Instanz hinterlegt, zu welchem Db2 Datenbanksystem und welchem IBM MQ Messagingsystem sich diese Instanz verbinden soll.

Started Task Control-Job

Bei einem Started Task Control-Job, kurz STC Job, handelt es sich um einen Batch Job, der mit Hilfe des „START“-Konsolenkommandos innerhalb von z/OS gestartet werden kann. Dieser Batch Job wird deshalb auch als Started Task bezeichnet.[Cass 07] Bei der DATEV e.G. existiert für jede Instanz eines Subsystems ein solcher Job, so also auch für CICS. In diesem werden zunächst einige zur Laufzeit benötigten Bibliotheken und Dateien eingebunden, unter anderem die CICS spezifischen Dateien[2]. Außerdem werden hier die SIT [3] Parameter definiert. Zunächst wird festgelegt welche Standard SIT verwendet werden soll. Anschließend können diese Standardwerte überschrieben werden. Zu diesen Parametern zählen unter anderem der eindeutige Name der CICS-Instanz, der Speicherort der dazugehörenden CSD und die Information, ob eine Verbindung zu einem Db2 Datenbanksystem hergestellt werden soll.

[1] Virtual Storage Access Method, spezielle Dateiart, die schnelle I/O-Zugriffe ermöglicht.[Love 13]
[2] Beschreibung in Absatz 4.1.1
[3] CICS system initialization table

Nach der Durchführung dieser Schritte und einem erfolgreichen Startversuch, steht dem Entwickler eine neue CICS-Instanz zur Verfügung. Der komplette Ablauf dauert, unter der Annahme, dass alle Beteiligten verfügbar sind, nur diese Anforderung umsetzen müssen und für die Beratung ein Arbeitstag veranschlagt wird, circa zwei Arbeitstage.

4.1.2. Bereitstellungsprozess einer Db2 Datenbank

Wie in Abbildung 4.2 zu erkennen ist, ist der Bereitstellungsprozess einer neuen Db2 Datenbank mit vielen Aufgaben im Entwicklerteam verbunden. Zunächst müssen sogenannte Projektinformationen, unter anderem Daten der Voruntersuchung, vom Entwicklerteam bereitgestellt werden. Das Projektkürzel, der Datenbank- und Projektname und die Projektbezeichnung müssen mit den involvierten Abteilungen besprochen werden. Über den sogenannten „Datenbankänderungsantrag“ wird ein Genehmigungsprozess angestoßen. Wenn alle Genehmigungen erteilt wurden, kann ein Dateneigentümer festgelegt werden. Anschließend muss die Datenbank mittels eines Datenbankmodells vom Entwicklerteam beschrieben werden und eine Usergruppe, die im späteren Verlauf die Datenbankzugriffsrechte benötigt, beantragt und angelegt werden. Die eigentliche manuelle Erstellung der Datenbank wird mittels des Datenbankmodells und den Projektinformationen im Anschluss dazu durchgeführt.

Die Zugriffsrechte für die zuvor beantrage Usergruppe auf die neue Datenbank werden beantragt. Schließlich steht dem Entwicklerteam die neue Db2 Datenbank zur Verfügung. Wird die Db2 Datenbank in Verbindung mit einem CICS verwendet, so sind weitere manuelle Schritte vom CICS Administratorenteam notwendig. Der komplette Ablauf dauert, unter der Annahme, dass alle Beteiligten verfügbar sind, nur diese Anforderung umsetzen müssen und für die Beratung ein Arbeitstag veranschlagt wird, circa zwei Arbeitstage.

4.1.3. Bereitstellungsprozess einer IBM MQ Queue

Auch bei dem Bereitstellungsprozess, siehe Abbildung 4.3, einer IBM MQ Queue ist das Entwicklerteam der Initiator.

Die Grundlage diese Prozesses ist ein Antrag auf Erstellung einer neuen IBM MQ Queue. Zuvor findet eine Beratung via Telefon, Email oder Terminen statt. Die Queues werden anschließend manuell eingerichtet und stehen dem Entwicklerteam zur Verfügung. Wird die Queue in Verbindung mit einem CICS verwendet, so sind weitere manuelle Schritte vom CICS Administratorenteam notwendig. Trotz des scheinbar schmalen Prozesses dauert der Ablauf unter der Annahme, dass alle Beteiligten verfügbar sind, nur diese Anforderung umsetzen müssen und für die Beratung ein Arbeitstag veranschlagt wird, circa zwei Arbeitstage.

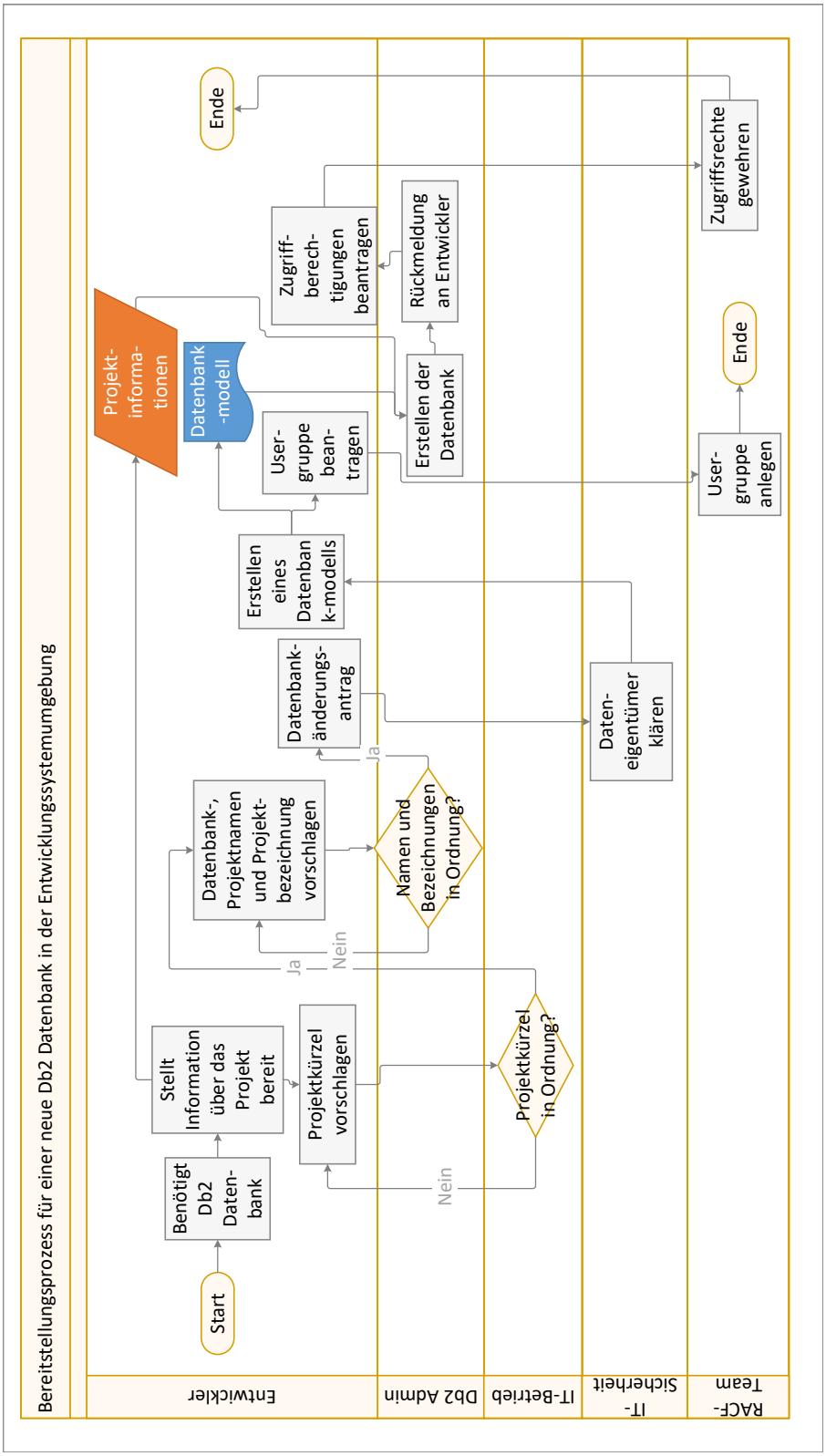


Abbildung 4.2.: Bereistellungsprozess einer Db2 Datenbank

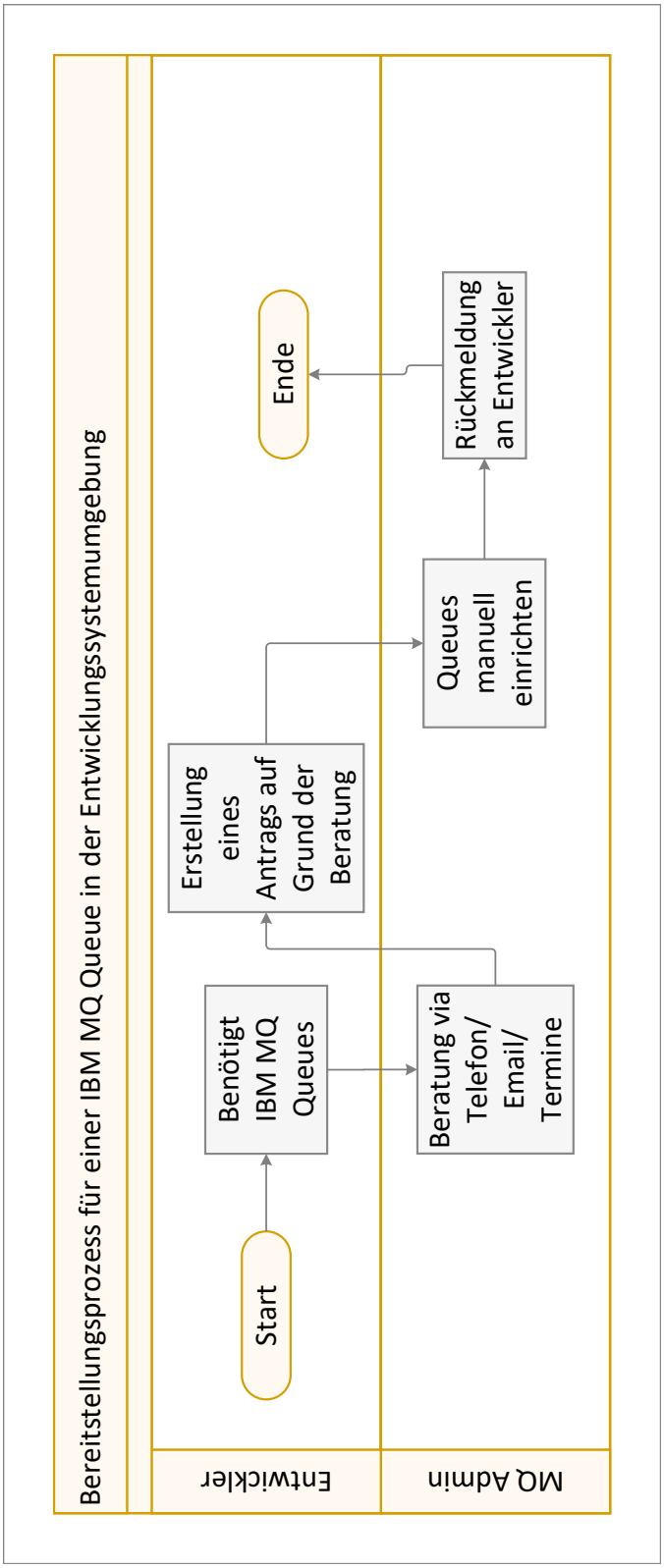


Abbildung 4.3.: Bereistellungsprozess einer IBM MQ Queue

4.1.4. Zusammenfassung aktueller Bereitstellungsprozess

Wie in den drei Diagrammen, Abbildungen 4.1, 4.2 und 4.3, zu erkennen ist, ist der aktuelle Bereitstellungsprozess noch mit vielen manuellen Schritten verbunden. Außerdem ist der Hauptaufwand in den Administratorenteams angesiedelt. Das Entwicklerteam ist der Initiator des Ablaufs. Folglich kümmert es sich um Formulare und die erste Kontaktaufnahme zum Administratorenteam.

Zusätzlich zu den vielen manuellen Schritten sind die vielen Absprachen zwischen mehreren Abteilungen zu nennen. Steht ein beteiligtes Team nicht zu Verfügung, kommt es zu Verzögerungen, das Team muss warten, der komplette Zeitplan kann sich dadurch nach hinten verschieben. Der Prozess für die Bereitstellung einer CICS-Instanz, mit einer Db2 Datenbank und IBM MQ Queues dauert in der Summe circa sechs Arbeitstage. Es setzt sich aus der Dauer der Einzelprozesse zusammen, für jedes Subsystem wird mit circa zwei Arbeitstagen gerechnet. Natürlich ist ein parallelisierter Ablauf der einzelnen Teilprozesse möglich, so kann die Gesamtdauer im besten Fall auf circa zwei bis drei Arbeitstage verkürzt werden.

Ein weiterer Punkt ist, dass die Kommunikation beziehungsweise der Initiator für den Start des gesamten Prozesses meist per Zuruf stattfindet. So existiert für die erste Kontaktaufnahme kein Formular, keine Automation oder ähnliches. Zur Kommunikation wird auf E-Mail, Telefon oder mittels Terminen zurückgegriffen.

4.2. DATEV-Rechnungsschreibung

Für diese Arbeit wurde die DATEV-Rechnungsschreibung als Beispielanwendung herangezogen, weil sie folgenden Anforderungen entspricht. Es handelt sich zum einem um eine in sich abgeschlossene Anwendung, die nur zu Beginn des Prozesses von anderen Anwendungen abhängig ist. Zum anderen benötigt die DATEV-Rechnungsschreibung ein CICS als Laufzeitumgebung, eine Db2-Datenbank und IBM MQ als Messaginglösung. Somit kann ein umfangreicher Bereitstellungsmechanismus untersucht werden.

Bei dem Gesamtablauf handelt es sich um einen Batch-Ablauf auf dem Großrechner der DATEV e.G. Dieser setzt sich aus folgenden Teilen zusammen:

- Sammeln von Berechnungssätze
- Tägliche Bewertung
- Rechnungsaufbereitung

Für die Beantwortung der Forschungsfragen ist nur ein Teil der „Tägliche Bewertung“relevant, die Preisermittlung.

4.2.1. Tägliche Bewertung

Dieser Ablauf läuft einmal täglich von Montag bis Freitag und ist für die Preisermittlung und Kundenzuordnung zuständig. Zur Realisierung wurden die Programmiersprachen Assembler, COBOL und Java genutzt. Am Ende dieses Ablaufes steht die ARUBA[4]-Db2-Datenbank. Dort werden die Berechnungsdaten der letzten 36 Monate aufbewahrt. Dabei handelt es ich um insgesamt circa 3,8 Milliarden Datensätze von einer Gesamtgröße von circa 400 GB mit Indizes. Diese Datensätze beinhalten alle Informationen für die endgültige Erzeugung der Rechnungen.

4.2.2. Preisermittlung

Die Preisermittlung ist für die Berechnung der Preise mit den dazugehörigen kundenindividuellen Abhängigkeiten, beispielsweise Rabatte, zuständig. Die Eingabe beläuft sich an Lasttagen auf bis zu 180.000 Geschäftspartner. Im DATEV e.G. Umfeld ist ein Geschäftspartner entweder eine Kanzlei oder ein einzelner Mandant. Aufgrund dieser Last wurde die Berechnung zum einen in CICS-Instanzen ausgelagert und zum anderen wurde der Ablauf in zwei Teile zerlegt:

- Bereitstellen der Preisinformationen
- Berechnung der Preise

Bereitstellen der Preisinformationen
Bevor die eigentliche Ermittlung der Preise stattfindet, werden zunächst die Preisinformationen und die kundenindividuellen Preisabhängigkeiten, wie zum Beispiel Rabatte, ermittelt. Für die Verarbeitung werden zwei Queues verwendet. Eine startet eine Transaktion im CICS, die andere wartet auf deren Antwort. Innerhalb der Transaktion werden alle benötigten Preisinformationen und -abhängigkeiten mit Hilfe einer Db2 Datenbank ermittelt. Diese Informationen werden dann in einem sogenannten „SHARED GETMAIN“-Bereich gespeichert. Dabei handelt es sich im Prinzip um einen Hauptspeicherbereich, der dem CICS Subsystem zur Verfügung steht. Die Adresse dieses Bereiches wird den Transaktionen zur Verfügung gestellt. Somit greifen die einzelnen Transaktionen nicht mehr direkt auf die Datenbank zu, sondern stattdessen auf den schnelleren Hauptspeicher. Diese Vorarbeit ist notwendig, da es aufgrund von bis zu 60 Millionen Datenbankzugriffen zu massiven Einbußen bezüglich der Performance führen würde.

[4] Abrechnungs- und Umsatz-Basis

Berechnung der Preise

Um die Berechnungsdaten der 180.000 Geschäftspartner an CICS-Instanzen zu übertragen, stehen dem System weitere Queues zur Verfügung. Darunter ist eine allgemeine Queue in der alle Aufträge, die für die Weiterverarbeitung zur Verfügung stehen, geschrieben werden. Pro Geschäftspartner wird ein Auftrag angelegt. In diesem Auftrag befinden sich die Namen vier weiterer Queues. Eine dieser Queues beinhaltet alle Informationen, die für die Preisermittlung des dazugehörigen Geschäftspartners notwendig sind. Hierzu zählt unter anderem die Adresse des vorher beschriebenen Hauptspeicherbereichs. In den restlichen drei Queues sind die Ergebnisse der Preisermittlung gespeichert. Die Ergebnisse stehen somit dem Batch-Ablauf zur Weiterverarbeitung zur Verfügung. Für jede der vier Queues existieren jeweils 100 vorgefertigte Namen. Somit können auch maximal nur 100 Aufträge gleichzeitig auf Weiterverarbeitung warten. Falls dieses Limit erreicht ist, wartet der Batch-Ablauf so lange, bis einer der Aufträge fertig gestellt wird. Sobald ein Auftrag in die allgemeinen Auftragsqueue geschrieben wird, wird eine CICS-Transaktion gestartet. Diese führt die Preisermittlung durch und schreibt das Ergebnis auf die dazugehörigen Queues. Ist dies geschehen, stehen die Queues wieder für einen neuen Auftrag zur Verfügung. Es können maximal 30 Transaktionen zeitgleich arbeiten.

Kapitel 5.

Realisierung

In diesem Kapitel wird beschrieben, wie die Aufgabe dieser Arbeit gelöst wurde. Dazu wird nach der im Kapitel 3 beschriebenen Reihenfolge der Arbeitsschritte vorgegangen. Es ist nochmal zu erwähnen, dass zunächst die Provisionierung einer CICS-Instanz untersucht wird. Danach wird in weiteren Schritten zuerst eine Db2 Datenbank und schließlich IBM MQ Queues dem Bereitstellungsprozess hinzugefügt. Die Funktionsfähigkeit der so generierten Laufzeitumgebung wird mittels eines Testablaufes am Beispiel der DATEV-Rechnungsschreibung sichergestellt. Es folgt eine Bewertung der implementierten Provisionierungslösung durch die Stakeholder bei DATEV e.G. (Entwickler, Administration, Technologiestrategie). Zuletzt wird noch ein Fazit zur Realisierung gezogen und ein Ausblick im Bezug auf die technischen Aspekte gegeben.

5.1. Testplex

Vor Beginn der eigentlichen Untersuchung mussten zunächst alle benötigten Rechte beantragt werden. Hierzu zählen unter anderem die Rechte für die Nutzung des Test-Plexes, die Nutzung von z/OSMF und z/OSPT und die Rechte für die Templateverwaltung innerhalb von z/OSMF. Außerdem war es auf dem Testplex möglich, die Rechte für das Erstellen der CICS Dateien, das Recht, um eine CICS-Instanz starten zu dürfen und die Rechte für die Administration von Db2 und IBM MQ einer persönlichen UserID zu geben. Dies stellt kein Problem dar, weil es sich bei dem Testplex um eine reine Systemtestumgebung handelt. Das „IBM Cloud and Management for z/OS“-Toolkit benötigt lesenden Zugriff auf den Speicherpfad der Template Dateien. Schließlich konnte mit dem ersten Versuch, ein bei der Installation von z/OSMF mitgeliefertes minimales CICS Template zu provisionieren, begonnen werden.

5.1.1. IBM Standard CICS Template

Trotz der Vorteile durch die Weboberfläche von z/OSMF wurde zunächst auf z/OSPT gesetzt. Diese Entscheidung fiel auf Grund der höheren Flexibilität von z/OSPT, durch

das Konzept der Images[1]. Da es sich um ein mitgeliefertes Template handelt, sind alle benötigten Workflow Definitionsdateien und Template Dateien vorhanden. Somit konnte mittels des Konsolenbefehesl „zospt build“ ein Image erzeugt werden. Jedoch zeigte sich ein weiterer Nachteil des Kommandozeileninterfaces, es ist nicht möglich Templates eine Domain und einen Tenant zuzuweisen. Dies hatte zur Folge, dass der Befehl „zospt build“ fehlschlug. Für alle folgenden Aufgaben wurde deshalb z/OSMF genutzt.

Um das Template in die Software Services[2] von z/OSMF aufzunehmen, sind die Template- und Workflow-Dateien in einem Unix Dateisystem auf dem Großrechner abgelegt. Dabei wurden ihm eine Domain und ein Tenant zugewiesen. Bevor das Template provisioniert werden kann, müssen Änderungen in der Variableinputfile vorgenommen werden. Dazu mussten die Werte, der in der Tabelle 5.1 genannten Variablen angepasst werden. Die Kurzbeschreibungen und die Beschreibungen aller Variablen, die im Standard Template vorhanden sind, ist in [IBM 19b] zu finden. Mit Hilfe der z/OSMF Oberfläche konnte das Template akutalisiert und somit die Änderungen übernommen werden.

Als nächster Schritt wurde ein Testlauf und somit ein erster Versuch das Template zu provisionieren durchgeführt. Dabei kam es anfangs zu Rechteproblemen, da die Anforderungen und Rahmenbedingungen der DATEV e.G. in dem standardisierten IBM Template natürlich nicht berücksichtigt waren, beispielsweise ist die Berechtigung für das Starten von Jobs von DATEV Vorgaben abhängig und CICS-Start-Mechanismen haben spezifische Anforderungen an die Eingabeparameter. Nach den notwendigen Anpassungen funktionierte das Provisonieren und die definierten Aktionen des IBM Standard CICS Templates im DATEV Umfeld.

5.1.1.1. DATEV e.G. spezifischen CICS Template

Nachdem das minimale IBM Standard CICS Template funktionsfähig war und erste Erfahrungen mit z/OSMF gesammelt worden waren, wurde ein allgemeines, mitgeliefertes Template untersucht. Ziel dieses nächsten Schritts war die Provisionierung einer funktionsfähigen DATEV e.G. spezifische CICS-Instanz. Um dieses Template an die DATEV Umgebung an-

[1] Beschreibung siehe Absatz 2.4.1
[2] Beschreibung in Absatz 2.4.2

Variablenname	Kurzbeschreibung
DFH_REGION_SEC	Legt fest, ob für das CICS Sicherheit im Allgemeinen aktiviert ist.
DFH_REGION_SECPRFX	Wenn DFH_REGION_SEC gesetzt ist, legt den Namen Prefix bei Authentificationanfragen für Ressourcen fest.
DFH_REGION_APPLID	Applikations ID der zu provisionierenden CICS-Instance.
DFH_LE_HLQ	High-level qualifier[3] für die Sprachumgebung[4]
DFH_REGION_HLQ	High-level qualifier für die CICS Dateien.
DFH_REGION_LOGSTREAM	Legt fest, wie die Log Dateien für die provisionierte CICS-Instanz erstellt werden sollen.
DFH_STC_ID	User ID mit dem die CICS-Instanz startet.
DFH_REGION_DFLTUSER	Default User ID für die CICS-Instanz.
DFH_REGION_VTAMNODE	Name des VTAM Knotens, wenn die CICS-Instanz hochfährt.
DFH_REGION_MEMLIMIT	Dem CICS maximal zur Verfügung stehender Speicherplatz.
DFH_ZOS_PROCLIB	Datei auf dem Großrechner, die den Job enthält, der für das Erzeugen der CICS-Instanz zuständig ist.
DFH_ZOS_VSAM_VOLUME	Speichersystem auf welchem die Dateien gespeichert werden sollen. Entscheidung kann auch an das System abgeben werden.
DFH_CICS_USSHOME	Homeverzeichnes des Unix System Services
DFH_CICS_HLQ	High-level qualifier von dem CICS Installationsort.

Tabelle 5.1.: Zu verändernde Variablen im minimalen CICS Template

zupassen und letztendlich eine „DATEV CICS-Instanz “ zu provisionieren, wurden folgende Schritte durchgeführt.

- Analyse des bestehenden Templates und der darauffolgenden Minimalisierung
- Umgang und Anpassung der CSD Datei
- Anpassung der Jobs und Skripte mit Schwerpunkt auf der „createCICS.jcl“-Datei.

Die Analyse ergab, dass das mitgelieferte Template mit insgesamt 76 verwendeten Dateien sehr umfangreich ist. Es zählen alle Dateien, die direkt mit dem Template in Verbindung stehen. Im Zentrum des Templates steht die Workflow Definitionsdatei „provision.xml “ mit circa 583 Zeilen Code. In dieser sind alle Steps, die bei einer Provisionierung durchgeführt werden, definiert. Das Template beinhaltet nicht nur die Möglichkeit, CICS-Instanzen mit unterschiedlichen Konfigurationen zu provisionieren, sondern auch, ob dies mit Skripten

oder mit der REST-Api geschieht. Zu diesen unterschiedliche Konfigurationen zählt unter anderem die Unterscheidung, ob die CICS-Instanz einem Sysplex hinzugefügt werden soll oder nicht. Die Übersichtlichkeit des allgeminen Templates ist dadurch geringer. So wurden neben allen für ein DATEV e.G. spezifischen CICS nicht notwendigen Dateien auch nicht benötigte Variablen und Steps entfernt. Wie in der Tabelle 5.2 zu sehen ist, konnten dadurch circa die Hälfte der Dateien gelöscht werden und bei der provision.xml wurde ein Drittel an Quellcode eingespart werden. Diese Version dient dem weiteren Vorgehen als Grundlage.

	IBM Standard CICS Template	DATEV e.G. spezifisches Template
Verwendete Dateien	76	36
provision.xml	circa 583 Codezeilen	circa 199 Codezeilen.

Tabelle 5.2.: Verlgeich des beiden Templates im Bezug auf deren Umfang

Als nächster Schritt wurde in Zusammenarbeit mit den CICS Administratorenteam festgelegt, wie im Umfeld der automatisierten Bereitstellung die CSD Dateien verwaltet werden soll. Es wurde festgelegt, dass jede provisionierte CICS-Instanz ihre eigene CSD Datei zur Verfügung gestellt bekommen soll. Hierfür soll die bestehende, von den Kollegen gepflegte Datei kopiert und mit bestimmten Namenkonventionen gespeichert werden. Somit ist sichergestellt, dass durch die neu provisionierten Instanzen die existierenden und bei DATEV im Einsatz befindlichen Instanzen nicht beeinflusst werden. Das ermöglicht auch jedem Nutzer der Provisionierung, die CSD seiner eigenen CICS-Instanz ohne Seiteneffekte zu bearbeiten. Ein weiterer Vorteil ist, dass bei der Deprovisionierung der CICS-Instanz diese Kopie der Standard Datei ohne Nebenwirkungen gelöscht werden kann. Dadurch, dass die Datei, die von den Kollegen gepflegt wird, als Grundlage verwendet wird, sind alle provisionierten Instanzen immer auf dem aktuellsten Stand. Um dies umzusetzen, musste ein JCL Job geschrieben werden, der den Kopiervorgang implementiert. Dieser Job wurde mittels eines neuen Steps in den z/OSMF Workflow eingebunden. Außerdem mussten bestimmte Gruppen zu der CSD Liste der CICS-Instanz hinzugefügt werden. Die JCL ist in Abbildung 5.1 dargestellt. Die Reihenfolge ist relevant, da es der Initialisierungsreihenfolge entspricht.

```
1  //INIT   EXEC PGM=DFHCSDUP
2  //STEPLIB  DD DSN=CICS.TS54.SDFHLOAD,DISP=SHR
3  //DFHCSD   DD DSN=CICS.DFHCSD.XPROV.TCICS42,DISP=SHR
4  //SYSPRINT DD SYSOUT=V
5  //*Reihenfolge ist WICHTIG!!
6  //SYSIN    DD *
7  ADD LIST(TCICS42) GROUP(TESTPCT)
8  ADD LIST(TCICS42) GROUP(DB0C)
9  ADD LIST(TCICS42) GROUP(RCTTEST)
10 ADD LIST(TCICS42) GROUP(FCTT1)
11 ADD LIST(TCICS42) GROUP(MQPROV01)
12 //
```

Listing 5.1: Hinzufügen weiterer CSD Gruppen zur Liste der provisionierten CICS-Instanz mittels eines Jobs

Im nächsten Schritt wurden die Jobs und Skripte angepasst. Zunächst wurden die Namen der CICS Dateien[5] an die DATEV e.G. internen Namenskonventionen angepasst. Ein spezielles Augenmerk lag auf der Anpassung der „createCICS.jcl“-Datei. In dieser befindet sich die Definition des STC Jobs für das provisionierte CICS. Im Standard IBM Template beinhaltet diese zunächst ein Makro für die Validierung der SIT Parameter. Zusätzlich werden noch bevor die Jobdefinition beginnt alle aus der Datei für die Eingabevariablen benötigten Variablenwerte in temporäre Zwischenvariablen eingefügt. Danach folgt die Definition des Jobs, diese setzt sich aus folgenden Hauptbestandteilen zusammen:

- Einbindung der benötigten Bibliotheken
- Einbindung der zuvor angelegten CICS spezifischen Dateien
- Definition der SIT Parameter

Sowohl bei die Einbindung der benötigten Bibliotheken als auch das Einbinden der zuvor angelegten CICS spezifischen Dateien beschränkte sich auf das Hinzufügen weiterer DD-Statements.

In Abbildung 5.2 ist zu sehen, dass es vor allem bei der Definition der SIT Parameter zu tief verschachtelten if-Bedingungen kommen kann. Es handelt sich um den Code, der für das Einlesen der Variable „DFH_REGION_SITPARAMS“ aus der Eingabedatei zuständig ist. In dieser Variable werden die SIT Parameter als Komma separierter String angegeben. Für die Erzeugung eines DATEV e.G. spezifischen CICS wurde das Makro für die Validierung von SIT Parametern beibehalten. Alles danach wurde zunächst durch eine zur Verfügung gestellten DATEV e.G. Standard JCL, für die Erzeugung eines CICS, ersetzt. Nach und

[5] Beschreibung in Absatz 4.1.1

nach wurde damit die notwendige Logik, wie die aus Abbildung **??**, hinzugefügt . Damit wurde die vorher statische DATEV e.G. Standard JCL durch Verwendung von Template internen Variablen flexibilisiert.

```
1  #set ($value5 = $!{instance-DFH_REGION_SITPARMS})
2  #set ($multipart = "NO")
3  #set ($tempStr = "")
4  #if($value5 != "")
5  #foreach( $sit in $value5.split(","))
6  #if($multipart == "YES")
7  #if( $sit.indexOf(')') > 0 )
8  ## Validate SIT
9  #validateSit($tempStr.concat($sit.trim()))
10 #set ($multipart = "NO")
11 #else
12 #set ($tempStr = $tempStr + $sit.trim() + ",")
13 #end
14 #else
15 #if( $sit.indexOf('(') > 0 && $sit.indexOf(')') == -1 )
16 #set ($multipart = "YES")
17 #set ($tempStr = $sit.trim() + ",")
18 #else
19 #validateSit($sit.trim())
20 #end
21 #end
22 #end
23 #end
```

Listing 5.2: Setzen der SIT Parameter durch Auslesen der „DFH_REGION_SITPARAMS“ Variablen

Es wurden nur die wirklich benötigten SIT Parameter aufgenommen. Die anzunehmenden Werte wurden einzeln mit dem CICS Administratorenteam besprochen und festgelegt. Es ist zu beachten, dass es im IBM Standard Template zwei Möglichkeiten gibt, diese Parameter zu setzen. Für bestimmte SIT Parameter besteht eine Variable innerhalb des Templates. Für alle anderen ist die Variable „DFH_REGION_SITPARAMS“ vorgesehen. In dieser Arbeit wurde hauptsächlich mit letzterer Variante gearbeitet. Dadurch sind die SIT Parameter nur an einer Stelle im Template zu verwalten, beziehungsweise wird die Verwaltung nicht auf zwei Arbeitsweisen verteilt.

Durch die beschriebene Vorgehensweise wurde erfolgreich die Provisionierung eines DATEV e.G. spezifischen CICS-Instanz umgesetzt. Sichergestellt wurde dies mit einem Anmelde-

vorgang an dieses CICS, wie in Abbildung 5.1 zu sehen ist. Außerdem sind alle Standard DATEV eG Transaktionen in dieser Instanz funktionsfähig. Die Deprovisionierung verlief nach Plan.

Abbildung 5.1.: Login Bildschirm der provisionierten DATEV spezifischen CICS-Instanz

5.1.1.2. Bereitstellung Db2

In diesem Absatz wird die Provisionierung einer Db2 Datenbank beschrieben. In der Systemumgebung Testplex bedeutet dies die Provisionierung der Datenbank ohne Tabelle und Daten.

Für die Erstellung einer Db2 Datenbank existiert innerhalb der DATEV e.G. eine REST-API. Wie im Absatz **??** beschrieben, ist es möglich, innerhalb eines Workflow Steps einen REST-Request abzusenden. Der Code ist in Abbildung A.4 im Anhang zu finden. So muss im Body des Requests unter anderem der Datenbankname und eine UserID übergeben werden. Der Code für das Löschen der Datenbank sieht ähnlich aus, nur handelt es sich in diesem Fall um einen DELETE-Request. Die zwei notwendigen Steps wurden erzeugt und in den Workflow eingebunden.

Die API ist nur dazu fähig, Datenbanken auf einem bestimmten Datenbanksystem zu erzeugen. Um die Datenbank aus der CICS-Instanz heraus nutzen zu können, muss dem CICS dieses Datenbanksystem mitgeteilt werden. Hierfür ist, wie in Abbildung 5.1 in Zeile acht bereits zu sehen ist, das Hinzufügen einer weiteren CSD Gruppe notwendig ist, sowie die Aufnahme weitere Bibliotheken in der „createCICS.jcl“. Dieser Aufruf wurde mittels neuer Variablen im Template möglichst dynamisch gestaltet und mussten in der Variableinputfile gesetzt werden.

5.1.1.3. Bereitstellung IBM MQ

In diesem Absatz wird die Provisionierung einer IBM MQ Queue im Testplex beschrieben. Es ist auch möglich einen IBM MQ Queue Manager zu provisionieren, der Fokus dieser Arbeit liegt aber auf der Bereitstellung von Queues. Für die Bereitstellung eines IBM MQ Queue Mangers bei DATEV e.G. ist laut IBM MQ-Administration vorerst keine automatische Bereitstellung vorgesehen, gegebenenfalls kann dies in einem zukünftigen Szenario umgesetzt werden. Ebenfalls in Abstimmung mit MQ- und CICS-Administration wurde entschieden, die Funktion eines Starts einer CICS Transaktionen über eine Queue vorerst nicht umzusetzen. Der Fokus lag auf der Prüfung, wie es möglich ist, eine einzelne Queue zu provisionieren, nicht die voll umfängliche Umsetzung der Anforderung der Anwendung DATEV-Rechnungsschreibung.

Die IBM stellt Programme für die Verwaltung und das Nutzen von Queues zur Verfügung. Diese können mittels eines Jobs und bestimmten Parametern gestartet werden. In Abbildung 5.2 ist die JCL des Jobs für das Erstellen einer Queue zu sehen. Das auszuführende Programm ist „CSQUTIL“ und als Parameter wird der Queuemanager übergeben. Unter dem DD Namen „MQSCIN“ ist der IBM MQ Befehl für das Erzeugen einer Queue zu sehen. Um zu Prüfen, ob die Queue auch funktionsfähig ist, wurde nach dem Erstellen auch mit Hilfe eines Jobs, eine Message auf die Queue geschrieben und wieder abgeholt. Der Job für das Löschen der Queues ist analog aufgebaut.

Ähnlich wie in Absatz 5.1.1.2 für die Datenbank-Provisionierung beschrieben, muss der CSD Datei eine weitere Gruppe für den Queuemanager angegeben werden. Zu sehen in Abbildung 5.1 in Zeile 11. Dadurch hat die CICS-Instanz Zugriff auf alle Queues, die sich innerhalb dieses Managers befinden. Des Weiteren ist die Aufnahme weiterer Bibliotheken in der „createCICS.jcl“ notwendig.

5.2. Entwicklungsstage

Innerhalb der Entwicklungsstage sind die Sicherheits- und Rechtevorschriften schärfer als auf dem Test-Plex. So wäre es zwar möglich, alle für die administrativen Aufgaben notwendigen Rechte einer persönlichen UserID zu geben. Dies würde bedeuten, dass alle Anwender dieses Templates diese Rechte auch benötigen. Damit bestünde eine potentielle Gefahr für das System, da sie damit auch außerhalb des Templates diese Rechte besitzen würden. Somit wurde in Absprache mit den Administratorenteams für CICS und IBM MQ festgelegt, hierfür jeweils einen technischen User[6] zu beantragen. Diesem werden nur die für das Template benötigten Rechte übergeben und er ist somit use-case-spezifisch. Um als Anwender das Template nutzen zu können, werden nur die Rechte benötigt, Jobs mit diesen technischen

[6] User ID mit zunächst keinen Berechtigungen

```
****** ***************************** Top of Data ******************************
000100 //P$PRVMQ1 JOB (0000,00000,00000,00000),KRUG,
000200 //             CLASS=V,MSGCLASS=V,TIME=(4,3),
000300 //             MSGLEVEL=(1,1),REGION=0M,NOTIFY=XXXXXXX,USER=YYYYYYY
000400 //*-------------------------------------------------------------------*
000500 //*                                                                   *
000600 //* DISPLAY QUEUES                                                    *
000700 //*                                                                   *
000800 //*-------------------------------------------------------------------*
000900 //STEPDISQ  EXEC  PGM=CSQUTIL,PARM='M00I',REGION=0M
001000 /*
001100 //STEPLIB    DD DSN=MQS.TEST.LLT.SCSQAUTH,DISP=SHR
001200 //            DD DSN=MQS.TEST.LLT.SCSQANLE,DISP=SHR
001300 //SYSPRINT DD SYSOUT=*
001400 //SYSIN      DD    *
001500   COMMAND DDNAME(MQSCIN) FAILURE(STOP)
001600 /*
001700 //MQSCIN     DD    *
001800   DEFINE QLOCAL(AWTB.PABHGMSHARE) -
001900                 REPLACE -
002000                 MAXDEPTH(6) -
002100                 PROCESS(AWTP.AWTP) -
002200                 TRIGGER -
002300                 MAXMSGL(1300) -
002400                 INITQ(SERVICE.TCICS42.INITQ) -
002500                 TRIGTYPE(EVERY) -
002600                 QDEPTHHI(80) -
002700                 QDEPTHLO(40)
002720 /*
002800 //PUT       EXEC PGM=CSQ4BCS2,
002900 //             PARM=('AWTB.PABHGMSHARE M00I')
003100 //STEPLIB    DD DSN=MQS.TEST.LLT.SCSQAUTH,DISP=SHR
003200 //            DD DSN=MQS.TEST.LLT.SCSQANLE,DISP=SHR
003300 //            DD DSN=MQS.TEST.LLT.SCSQLOAD,DISP=SHR
003400 //STDOUT    DD SYSOUT=*
003500 //STDERR    DD SYSOUT=*
003600 //SYSPRINT DD SYSOUT=*
003700 //SYSIN     DD *
003800 TEST
003900 /*
004000 //GET       EXEC PGM=CSQ4BCJ1,
004100 //             PARM=('M00I AWTB.PABHGMSHARE 1 D N')
004300 //STEPLIB    DD DSN=MQS.TEST.LLT.SCSQAUTH,DISP=SHR
004400 //            DD DSN=MQS.TEST.LLT.SCSQANLE,DISP=SHR
004500 //            DD DSN=MQS.TEST.LLT.SCSQLOAD,DISP=SHR
004600 //SYSDBOUT DD   SYSOUT=*
004700 //SYSABOUT DD   SYSOUT=*
004800 //SYSPRINT DD   SYSOUT=*
004900 //SYSOUT   DD   SYSOUT=*
005000 //
****** ***************************** Bottom of Data ***************************
```

Abbildung 5.2.: Define IBM Queue, am Beispiel einer Trigger Queue

Usern ausführen zu dürfen. Für Db2 ist ein solcher User nicht notwendig, da das Datenbanksystem hinter der REST-API für alle zugänglich ist und jeder darauf Datenbanken erstellen darf.

Bei der Übertragung des Templates vom Test-Plex in die Entwicklungsstage waren Anpassungen in allen drei Bereichen des Templates notwendig.

5.2.1. CICS Anpassung

Dass der CICS spezifische technische User zum Einsatz kommt, musste der „Job“ Baustein jeder JCL in jedem Step modifiziert werden. Dafür bietet z/OSMF die Möglichkeit beim Zuweisen des „Tenants“ eine Standard Jobkarte, die vor jeden Job des Templates eingefügt wird, zu hinterlegen. Die CICS spezifischen Dateien können von der täglichen Datensicherung der Entwicklungsstage ausgeschlossen werden. Da diese bei der Deprovisionierung gelöscht werden. Um dies zu gewährleisten musste der Messageclass Parameter mit dem Wert „NONE“ angegeben werden.

Außerdem wird die CSD Datei, die als Vorlage gilt, durch die Standard Entwicklungsstage CSD Datei ersetzt. In der Entwicklungsstage kommen im Vergleich zum Testplex andere

Db2 und IBM MQ Bibliotheken zum Einsatz. Dahingehend wurde die „createCICS.jcl“-Datei angepasst. Zusätzlich musste ein SIT Parameter angepasst werden, so dass die Log Dateien funktionisfähig sind. Eine weitere CSD Gruppe musste hinzugefügt werden. Siehe Zeile 16 im Codeabschnitt 5.3. Diese sorgt dafür, dass die Bibliotheken, die die kompilierten Programme der kompletten Entwicklungsstage beinhalten, zur Verfügung stehen. Außerdem kam noch eine neue Bibliothek hinzu. Diese dient später als Ablageort der kompilierten Programme, die explizit nur in diese CICS-Instanz vorhanden sind. Dies ist ein Standardvorgehen innerhalb der DATEV e.G. um neue Programmversionen zu testen.

5.2.2. Db2 Anpassung

Eine genauere technische Analyse der DATEV-Rechnungsschreibungsdatenbank kam zu dem Ergebnis, dass es zwar möglich wäre diese Datenbank zu provisionieren, dies aber den zeitlichen Rahmen dieser Arbeit übersteigen würde. Der Grund hierfür ist die Komplexität der benötigten Tabellen. So wird auf drei Tabellen für die Ermittlung der Produktstammdaten lesend zugegriffen, auf neun weitere bei der Bestimmung der Preisabhängigkeiten. Auf die Tabellen wird nicht direkt zugegriffen, sondern über Views[7]. Bei den meisten werden innerhalb der View noch weitere Tabellen, teilweise aus anderen Datenbanken, gejoint. Insgesamt besteht das System aus 14 Tabellen, die auf vier Datenbanken aufgeteilt sind, und 12 Views für den Zugriff auf diese Tabellen.

Die Db2 Administration muss dafür Vorarbeit leisten. Mit dieser wurde begonnen, jedoch stellte sich heraus, dass die Komplexität (circa 600 Zeilen Code[8] für einen kleinen Teil an Tabellen) der Anwendung DATEV Rechnungsschreibung im Rahmen dieser Arbeit als zu umfangreich angesehen wurde. Sollte sich die Provisionierung generell als zielführend erweisen wird dieser Einmalaufwand erbracht werden.

Für die weitere Arbeit werden Datenbanken, die in einem anderen Datenbanksystem bereits vorhanden sind, genutzt. Hierfür mussten die dafür vorgesehenen Variablen in der Eingabedatei des Templates angepasst werden. Dadurch ändert sich die Gruppe in Zeile acht im Codeabschnitt 5.1 von „DB0C “ auf „DB0T “. Außerdem wurden sowohl in der Provisionierungs- als auch in der Deprovisionierungsdatei die Datenbanksteps auskommentiert und somit kommen diese nicht mehr zum Einsatz.

5.2.3. IBM MQ Anpassung

Da für die DATEV Rechnungsschreibung, wie im Absatz **??** beschrieben, sehr viele gleichartige Queues benötigt werden, wurde für die Erstellung dieser von den IBM MQ Admi-

[7] Alias eines Datenbankabfrage, auf die wie auf eine normale Tabelle zugegriffen werden kann
[8] Data Definition Language im Anhang A.2 zufinden

nistratorenteam ein REXX Skript angefertigt. Dies geschah unabhängig dieser Arbeit zum Zeitpunkt der Einführung des aktuellen DATEV Rechnungsschreibungsprozesses. Dieses Skript steht dieser Arbeit zur Verfügung. Für die Provisionierung IBM MQ Queues waren folgende Arbeitsschritte notwendig.

- Anpassung des zur Verfügung stehenden Skriptes
- Implementierung von Jobs für restliche Queues
- Anpassung der CICS CSD Datei

Hierfür wurden zunächst die Eingabeparameter durch vorher angelegte Templatevariablen ersetzt. Diese steuern, wie viele Queues jeweils angelegt werden, auf welchen Queue Manager die Queues angelegt werden und den ersten Qualifier des Queuenamens. Für den restlichen Queuenamen existiert auch eine Variable, in dieser werden die Namen als Komma separierte Liste angeben und ausgelesen. Anhand dieser Namen wird dann die maximale Queuetiefe und die maximale Länge einer einzelnen Nachricht festgelegt. Im alten Skript wurden die Queues mit Hilfe einer Queue, die als Vorlage dient, angelegt. Im Fall einer Provisionierung kann nicht davon ausgegangen werden, dass diese Vorlagen zur Verfügung stehen. Deshalb wurden die benötigten Parameter explizit manuell angegeben. Um die damit erstellten Queues zu testen, wurde eine Routine entwicklelt, die eine Nachricht auf die Queue schreibt und diese wieder abholt. Anschließend wurde das Skript in den Provisionierungsworkflow mit Hilfe eines neuen Steps aufgenommen.

Für die Deprovisionierung der Queues besteht noch kein Skript. Als Grundlage kann das vorher angepasste Provisionierungsskript dienen. Hierfür musste der „Define“-Befehl für die Erstellung von Queues durch den „Delete“-Befehl ausgetauscht werden. Die Logik für die Ermittlung der maximalen Queuetiefe und der maximalen Nachrichtenlänge wird dafür nicht mehr benötigt und konnte entfernt werden.

Die durch die beiden Skripte erstellten Queues sind nur für den Datenaustausch zwischen der CICS Transaktion für die Preisermittlung und dem Batch Ablauf zuständig. Wie in Absatz **??** beschrieben, benötigt der Ablauf noch weitere Queues. Da es sich hierbei um spezielle Queues handelt, wurde auf die im Absatz 5.1.1.3 gezeigte Technik zurückgegriffen. Bei der Antwort-Queue für die Ermittlung der Listenpreise handelt es sich um eine Queue ohne besondere Parameter. Es werden noch zwei Trigger-Queues benötigt, die über Prozesse eine Transaktion im CICS starten. Als letzter Baustein für das Triggering der Transaktion wird noch eine Initiation Queue benötigt. Diese muss im CICS hinterlegt sein.

Jeder CICS-Instanz kann nur eine Initiation Queue zugewiesen sein. Dadurch benötigt jedes CICS eine eigene Initiation Queue. Die Zuweisung geschieht in der IBM MQ CSD Gruppe. Somit müsste für jede provisionierte CICS-Instanz im Voraus eine solche CSD Gruppe angelegt werden. In Absprache mit der IBM MQ-Administrations wurde entschieden, die

Verwaltung der IBM MQ CSD Gruppe komplett dem Template zu übergeben. Diese Entscheidung hatte eine Änderung des in Abbildung 5.1 gezeigten Codes zur Folge. So wird, wie in Abbildung 5.3 dargestellt, zunächst eine Gruppe angelegt und erst anschließend dem CSD hinzugefügt.

```
1  //INIT   EXEC PGM=DFHCSDUP
2  //STEPLIB  DD DSN=CICS.TS54.SDFHLOAD,DISP=SHR
3  //DFHCSD   DD DSN=CICS.DFHCSD.XPROV.TCICS42,DISP=SHR
4  //SYSPRINT DD SYSOUT=V
5  //*Reihenfolge ist WICHTIG!!
6  //SYSIN    DD *
7  DEFINE MQCONN(M00I)
8         G(MQPROV01)
9         MQNAME(M00I)
10        INITQ(SERVICE.TCICS42.INITQ)
11 ADD LIST(TCICS42) GROUP(TESTPCT)
12 ADD LIST(TCICS42) GROUP(DB0T)
13 ADD LIST(TCICS42) GROUP(RCTTEST)
14 ADD LIST(TCICS42) GROUP(FCTT1)
15 ADD LIST(TCICS42) GROUP(MQPROV01)
16 ADD LIST(TCICS42) GROUP(RPL)
17 //
```

Listing 5.3: Erstellung einer neuen CSD Gruppe

Für jeden IBM MQ bezogenen Job wurde zuallerletzt die Jobkarte angepasst und der technische User der CICS-Administration durch den technischen User der IBM MQ-Admninistration, der für administrative Aufgaben berechtigt ist, ausgetauscht.

5.2.4. Testablauf

Für die Prüfung der Funktionsfähigkeit der so generierten Laufzeitumgebung steht dieser Arbeit ein Testablauf zur Verfügung. Dieser wurde von den Mitarbeitern der DATEV Rechnungsschreibung beigesteuert. Dabei handelt es sich um einen Teilablauf des gesamten DATEV Rechnungsschreibungsprozesses. In diesem Ablauf wird nur die Preisermittlung, die die Laufzeitumgebung CICS benötigt, getestet. Als Eingabe dienen vordefinierte Dateien und die Ergebnisse werden ebenfalls in Dateien geschrieben. Der Ablauf liegt in Form von zwei Jobs vor. Beide sind in der gleichen JCL Datei definiert, somit starten beide zeitgleich. Dies ist notwendig, da der erste Job die Verarbeitung im CICS über die Queues startet und der zweite auf die Ergebnisqueues lauscht.

Um den Ablauf auch auf der vorher provisionierten Laufzeitumgebung zu starten, musste lediglich der verwendete Queue Manager angepasst werden. Über die Queues und das verwendete Triggering wird die Transaktion im richtigen CICS gestartet. Um die Ausgabe zu prüfen wurde der gleiche Testablauf mit den gleichen Eingabedateien auf der für Testzwecke üblichen Laufzeitumgebung durchgeführt. Anschließend wurden die Ausgabedateien beider Läufe verglichen.

5.3. Bereitstellungsprozess aktuelles Template

Bei dem Bereistellungsprozess, der durch das aktuelle Template möglich gemacht wird, sind drei Fälle zu unterscheiden:

1. Fall:
   Dem Entwicklerteam steht das Template in z/OSMF zur Verfügung und es wurde noch keine Instanz dieses Templates provisioniert. Es wird eine neue Instanz benötigt.

2. Fall:
   Dem Entwicklerteam steht das Template in z/OSMF zur Verfügung und es steht bereits eine Instanz dieses Templates zur Verfügung. Es wird eine weitere Instanz benötigt.

3. Fall:
   Das Administratorenteam führt Änderungen an einer Workflow Definitionsdatei durch. Hier ist zwischen zwei weiteren Fällen zu unterscheiden:

   a) Das Template wurde noch nicht veröffentlicht.

   b) Das Template wurde veröffentlicht.

5.3.1. 1. Fall

Der Mitarbeiter meldet sich an der zOSMF Oberfläche an und klickt auf den Menüleistenpunkt „Cloud Provisioning“. Anschließend öffnet er die „Software Services“ und wählt dort das oben genannte Template aus. Er kann es ohne Änderungen provisionieren und damit seine Programmabläufe testen.

5.3.2. 2. Fall

Mit dem aktuellen Stand muss der Mitarbeiter wissen, an welchem Speicherort das Template abgelegt ist, da er die Template - nicht die Workflowdateien - kopieren muss. Es sind Änderungen der Variableinputfile notwendig. Unter anderem ist eine andere CICS Application ID zu wählen. Um die Queues und IBM MQ Prozesse aus Fall eins nicht zu überschreiben, muss ein anderer Queue Manager gesetzt werden. Dieser Queue Manager muss von den zuständigen Administratorenteam manuell bereitgestellt werden. Die Erzeugung einer von Fall eins unabhängigen Instanz setzt die Aufnahme eines neuen Templates, welches die veränderten Dateien beinhalten, in z/OSMF voraus.

5.3.3. 3. Fall

Ein Template ist dann veröffentlicht, wenn es den berechtigten Teams zur Verfügung steht. Zunächst muss der Speicherort der zu bearbeiteten Dateien bekannt sein. Anschließend kann die Änderung mit einem Editor nach Wahl durchgeführt werden.

5.3.3.1. 3. Fall a)

Hier kann das Template in der z/OSMF Oberfläche per Mausklick aktualisiert werden.

5.3.3.2. 3. Fall b)

Um die Funktionsfähigkeit der veralteten Instanzen weiterhin sicherzustellen, muss eine neue Version des Templates erzeugt werden. Dies ist auch per Mausklick zu lösen.

5.4. Fazit Realisierung

Am Ende der Realisierung steht ein funktionsfähiges Template. Dieses Template provisioniert ein CICS und die benötigten IBM MQ Queues. Wie in Absatz 5.2.2 beschrieben, wurde eine Db2 Datenbank wegen hoher Komplexität außen vorgelassen. Auf dem Testplex wurde bewiesen, dass die Provisionierung einer Datenbank möglich ist. Des Weiteren wäre die Provisionierung von Tabellen mit hohem einmaligen Arbeitsaufwand ebenfalls möglich. Ein Testablauf der Beispielanwendung DATEV Rechnungsschreibung in einer provisionierten, isolierten CICS-Laufzeitumgebung konnte korrekt durchgeführt werden.

Folgende Probleme wurden im Rahmen der Implementierung erkannt:

- Nicht sprechende Fehlermeldungen von z/OSMF
- Nicht identifizierbare Programmiersprache
- Nicht optimales Zugriffsrechtekonzept

Als erstes Problem sind nicht sprechenden Fehlermeldungen von z/OSMF, Abbildung 5.3, zu nennen. z.B. wird bei dem Hinzufügen und Aktualisieren eines Templates in z/OSMF

Abbildung 5.3.: Beispiel einer Fehlermeldung von zOSMF

das Template und damit alle davon benötigten Dateien auf Syntaxfehler geprüft. Die in Abbildung 5.3 gezeigte Meldung tritt dann ein, wenn ein solcher Syntaxfehler vorhanden ist. Es ist aber nicht zu erkennen, welcher Fehler genau vorliegt, noch nicht einmal in welcher Datei dieser auftritt. Zudem auch keine genaue Anzahl an auftretenden Fehlern. Dieser Umstand, kombiniert mit 36 bestehenden Dateien, erschwert die Fehlersuche. Im Gegensatz dazu wird im Fehlerfall beim Ausführen eines Steps immer der Fehlercode und der genaue Ort des Fehlers ausgegeben. Beispielsweise wird bei einem Step, in dem ein REST Aufruf durchgeführt wird, und ein Fehler auftritt, der Requestcode und die hinterlegte Fehlermeldung an der z/OSMF Oberfläche angezeigt.

Ein weiteres Problem ist eine nicht genau identifizierbare Programmiersprache, die für die dynamische Generierung von Skripten genutzt wird. So ermöglicht diese die dynamische Wertzuweisung von zum Beispiel REXX-Variablen durch Variablen des Templates. Außerdem besteht eine Art von String Verarbeitung. Zu beachten ist, dass wenn am Zeilenanfang ein „#“ steht, kann diese Programmiersprache verwendet werden. In Abbildung 5.4 ist ein Beispiel zu sehen. Dort werden die Queuenamen, die als kommaseparierte Liste in der Templatevariable „DFH_MQ_QUEUENAMES“ angegeben sind, ausgelesen und in eigenen REXX Variablen gespeichert. Zu sehen ist zunächst eine „set“ Anweisung, mit der Variablen zugewiesen werden können, If-Bedingungen und eine foreach-Schleife stehen außerdem zur Verfügung.

```
1  i=0
2  #set ($names = $!{instance-DFH_MQ_QUEUENAMES})
3  #set ($multipart = "NO")
4  #set ($tempStr = "")
5  #if($names != "")
6  #foreach( $queue in $names.split(","))
7  i=i+1
8  names.i="$queue"
9  #end
10 #end
11 names.0=i
```

Listing 5.4: Auslesen der „DFH_MQ_QUEUENAMES“ Variablen und schreiben in REXX Variablen

In Abbildung 5.5 wird das Ergebnis, welches zur Laufzeit ausgeführt wird, dargestellt. Es ist zu erkennen, dass nur noch die für das REXX Skript notwendigen Codeabschnitte vorhanden sind. Dadurch können sehr dynamische Templates erstellt werden. Jedoch wurde weder eine Dokumentation zu dieser Sprache, noch um welche Sprache es sich genau handelt gefunden. Somit liegt dem Wissen über diese Sprache nur der Code aus Beispielen der IBM zu Grunde.

```
1  i=0
2  i=i+1
3  names.i="L1.GPNRBERINFO"
4  i=i+1
5  names.i="L1.KLAMMERINFOLIST"
6  i=i+1
7  names.i="L1.KUNDENPREISINFOLIST"
8  i=i+1
9  names.i="L1.PABHREFERENZLIST"
10 names.0=i
```

Listing 5.5: Zur Laufzeit erzeugtes Skript, der Grundlage aus Codeabschnitt 5.4

Ein weiterer Problempunkt ist das mit z/OSMF und dem Template einhergehende Zugriffsrechtekonzept. Die z/OSMF Berechtigungsgruppen sind nicht an die DATEV e.G. internen Richtlinien angepasst. Die Aufnahme in eine solche Gruppe, um zum Beispiel die z/OSMF Oberfläche nutzen zu dürfen, geschieht auf Zuruf und manuelles Hinzufügen einer User ID durch einen Mitarbeiter. Außerdem ist der Einsatz einer für das ganze Template gültigen Standard Jobkarte, um technische User verwenden zu können, nicht optimal. z/OSMF bietet hier eigentlich eine Möglichkeit in der Stepdefinition einen „runAsUser“ anzugeben. Unter

diesem User würde der Step dann ausgeführt werden. Folglich ist das die Stelle an der zum Beispiel für CICS Steps der technische User für administrative CICS Aufgaben angegeben werden müsste. So würde das Gewähren der expliziten Rechte zum Starten eines Jobs mit der technischen User Id entfallen und damit die manuelle Arbeit des „Gewährens“, was mittels eines Formulars beantragt wird. Jedoch um einen „runAsUser“ in der Stepdefinition angeben zu können, muss in der dem Template zugewiesen „Domain“ ein sogenannter „Cloud Security Admin“ hinterlegt sein. Dieser würde sicherstellen, dass nur die für ein Template zugelassenen User dieses Template auch provisionieren dürfen. In dieser Arbeit wird die mitgelieferte „Default Domain“ genutzt, in dieser ist kein „Cloud Security Admin“ angegeben. Da es sich um die Standard „Domain“ handelt, darf diese nicht geändert werden. Somit müsste eine eigene „Domain“ angelegt werden um einen „Cloud Security Admin“ hinterlegen zu können. Dadurch, dass sich z/OSMF bei der DATEV e.G. noch in einem Teststadium befindet, wird von der Erstellung einer eigenen „Domain“ abgesehen. Dies ist der Grund für den nicht optimalen Einsatz der oben genannten Jobkarten. An diesen beiden Fällen ist zu erkennen, dass das Rechtekonzept noch nicht für einen firmenweiten Einsatz ausgelegt ist und noch überarbeitet und angepasst werden muss. Dies ist jedoch explizit nicht Bestandteil dieser Arbeit.

5.5. Interviews

Bevor die eigentlichen Interviews durchgeführt wurden, wurden den einzelnen Stakeholdern, also CICS Administration, Db2 Administration, IBM MQ Administration, Entwicklern und einer Mitarbeiterin des Technologiestrategieteams, die Ergebnisse dieser Arbeit vorgestellt. Es wurde sowohl die z/OSMF Lösung (siehe Absatz 5.3) als auch die durch z/OSPT ermöglichte Lösung (siehe Absatz 6) erläutert. Der Schwerpunkt der Vorstellung wurde an das jeweilige Team angepasst. So wurde bei den Administratorenteams vor allem auf die Erstellung der Templates, welche für ihr Arbeitsgebiet relevant ist, eingegangen.

Sowohl der Fragenkatalog, als auch die ausgefüllten und digitalisierten Fragebögen sind im Anhang A.3 zu finden. Für das Entwicklerteam und der Mitarbeiterin des Technologiestrategieteams waren nur die Fragen 1., 2. und 6. bis 10. des Fragebogens von Relevanz. Die Fragebögen werden im Folgenden zunächst nach Gruppen ausgewertet. Schließlich wird daraus ein allgemeines Stimmungsbild abgeleitet.

5.5.1. CICS Administratoren

Der momentan etablierte Bereitstellungsprozess wird von der CICS Administration mit hohen manuellen Aufwand verbunden. Dies kombiniert mit viel Abstimmungsbedarf zwischen den Administratoren- und Entwicklerteams führt dazu, dass der Prozess als langsam und verbesserungswürdig angesehen wird. In der Umsetzung mit z/OSMF sieht die CICS Administration trotz des vermuteten hohen Einarbeitungsaufwandes bereits einen Mehrwert. Der Hauptvorteil des vorgestellten z/OSPT Lösungsansatzes sei dessen Flexibilität. Jedoch schreckt die dadurch benötigte Komplexität des zu erstellenden dynamischen Templates ab. Dieser Effekt wird durch fehlende Toolunterstützung und dem dadurch fehlenden Syntaxhighlighting beim Editieren der Template Dateien bzw. der Workflowdefinitionfiles verstärkt. Nach der Hürde des Einarbeitungsaufwandes und Eingewöhnung in das Editieren von Template Dateien und der Workflowdefinitionsdateien stehe einer aufwandssparenden Provisionierung mittels des „IBM Cloud Provisioing and Management for z/OS“-Toolkits nichts im Wege.

5.5.2. Db2 Administratoren

Das ganze „IBM Cloud Provisioing and Management for z/OS“-Toolkit wird als sehr mächtig, aber komplex beschrieben. Im Vergleich dazu funktioniere der momentan etablierte Bereitstellungsprozess sehr gut, da dieser bereits lange eingesetzt wird. Jedoch könnten lange Wartezeiten, die durch die vielen Abhängigkeiten zwischen Personen und Abteilungen zu Stande kommen, durch einen automatisierten Ablauf mittels des Toolkits eliminiert werden. Der durch z/OSMF ermöglichte Prozess zeige zwar das eine Automatisierung in diesem

Bereich möglich ist, aber auch das noch viel Forschungsaufwand und Weiterentwicklung in diesem Bereich notwendig ist, um die Provisionierung wirklich nutzen zu können. z/OSPT diene dabei als Hilfsmittel den Bereitstellungsprozess in eine CI/CD-Pipeline aufzunehmen und so weiter zu automatisieren. Das durch das Toolkit ermöglichte automatisiertes Deployment von z/OS Middleware wird als notwendiger Schritt, um den Mainframe weiterhin erfolgreich zu betreiben, betrachtet.

5.5.2.1. IBM MQ Administratoren

Die IBM MQ Administratoren stimmen überein, dass der momentan etablierte Bereitstellungprozess mit einem hohen manuellen Arbeitsaufwand verbunden ist. Durch Arbeiten auf Zuruf und Kommunikation über Telefon, Email oder Terminen entstehen häufig Rückfragen. Die Meinungen über die Lösungen mit z/OSMF und z/OSPT sind jedoch unterschiedlich. So biete der z/OSMF Ablauf zwar einem Mehrwert durch Abbau von manuellen Eingriffen, allerdings sei dieser bezogen auf die Queues noch sehr spezifisch. Um einen größeren Mehrwert zu generieren, ist das automatische Provisionieren eines Queuemanagers mit in das Template aufzunehmen. Hier sei der zusätzliche Arbeitsaufwand nicht zu vernachlässigen. Beim z/OSPT Lösungsansatz wird kritisiert, dass es sich nur um „pseudo“Docker Container handle. Die hier von der IBM gewählte Namensgebung führt zur Verwirrung, da ein z/OSPT Container zwar ein Container im Sinne von einem Behältnis für Middleware ist, jedoch nicht im Sinne eines Docker Containers, der in unterschiedlichen Systemumgebungen lauffähig ist. Ein weiterer Kritikpunkt ist, dass das gesamte Toolkit im Vergleich zu Jenkins nicht einfach genug zu verwenden sei. Wenn diese Probleme behoben werden können, könnten sich die IBM MQ Administratoren vorstellen IBM MQ Ressourcen mittels des Toolkits zu verwalten. Dabei sei zu beachten, dass erst noch eigene Erfahrungen mit dem Toolkit gesammelt werden sollten, bevor eine endgültige Bewertung möglich ist.

5.5.2.2. Entwicklerteam der DATEV Rechnungsschreibung

Der Entwicklerfragebogen wurde zusammen mit zwei Entwicklern ausgefüllt.

Aus Sicht des Entwicklers wird vor allem für den in Absatz 5.3.2 beschriebenen Fall viel Wissen über die z/OSMF Oberfläche und das Template selbst benötigt. Dieses Wissen müsse auch bei nicht häufiger Nutzung über einen längeren Zeitraum erhalten werden. Deshalb sei der z/OSPT Lösungsansatz, mit dem auf die z/OSMF Oberfläche durch den Einsatz mittels Jenkins oder dem DATEV „Marktplatz“ verzichtet werden kann, besser geeignet. Ist diese Integration möglich wird ein Hauptvorteil darin gesehen, dass der Bereitstellungsprozess mehr in den Händen des eigenen Teams liegt. So sei eine Steigerung der Effizienz möglich. Es stehe dem Sammeln von Erfahrungen mit dem Prozess und dem kompletten Toolkit nichts im Wege. Für die Zukunft könne sich die Nutzung auch für die Qualitätsicherungs- und

Produktionsstage, um dort beispielsweise CICS-Instanzen horizontal zu skalieren, vorgestellt werden.

5.5.2.3. Fachberaterin im Bereich Technologiestrategie

Laut der Fachberaterin im Bereich Technologiestrategie ist der gezeigte Ablauf beziehungsweise die z/OSMF Oberfläche für die Aufgabe des Provisionierens von z/OS Middleware geeignet. Jedoch sei es besser wenn z/OSMF in den bereits existierenden „Marktplatz“ für DATEV Cloud Lösungen integriert wäre. Der Prozess, der mit Hilfe von z/OSPT ermöglicht wird, wird als gut angesehen, da durch ihn die Entwicklung von z/OS Anwendungen an die Vorgehensweise der Cloud Native Entwicklung angenähert wird. Hier kommt die Rolle des Build Engineers auch für solche Anwendungen ins Spiel. Dieser kümmert sich um die Erstellung und Pflege der Build-Pipeline. Große Nachteil im momentan etablierten Bereitstellungsprozess sei vor allem, dass eine Anzahl von Entwicklern, die parallel an einem Produkt arbeiten, sich die gleiche Entwicklungssystemumgebung teilen. So arbeiten alle mit der gleichen CICS-Instanz, der gleichen Test-Datenbank und mit den gleichen IBM MQ Queues. Dadurch beeinflussen Änderungen des einen Entwicklers die Tests der anderen Kollegen, es entsteht Koordinationsaufwand. Falls Änderungen an der Umgebung notwendig sind, kann während dieser Zeit kein Entwickler weiterarbeiten. Hier liege der Vorteil des „IBM Cloud Provisioing and Management for z/OS“-Toolkits. Es ermögliche aus Entwicklersicht eine sehr einfache, schnelle Möglichkeit eine isolierte Umgebung bereitzustellen, unabhängig von den Administratorenteams. Zusätzlich diene die Konfigurationsdateien auch als Dokumentation, welche Ressourcen für ein erneutes Erstellen der Umgebung notwendig sind.

Abschließend lässt sich sagen, dass aus Sicht einer Fachberaterin im Bereich Technologiestrategie dieses Toolkit die Entwicklung beziehungsweise den Bereitstellungsprozess deutlich verbessern könne. So sei für den Entwickler ein an die Cloud Native Welt angenäherter Entwicklungsprozess möglich. Dadurch wird der Wechsel zwischen beiden Umgebungen immer fließender.

5.5.3. Meinungsbild

Über alle Gruppen hinweg lassen sich folgende Punkte zusammenfassen:

- neuer Prozess notwendig
- z/OSPT Lösung bevorzugt
- erste Erfahrungen sammlen

Es stimmen alle Gruppen überein, dass der momentan etablierte Bereitstellungsprozess für Mainframesubsysteme durch viele Absprachen und Abstimmungsaufwand zeitaufwändig ist. Sie würden einen automatisierten und dadurch schnelleren und weniger fehleranfälligen Prozess begrüßen.

Jedoch muss dieser Prozess aus Entwicklersicht mit minimalem Konfigurationsaufwand verbunden sein. Dies könnte durch eine Provisionierung mittels z/OSPT und einer Integration in eine Jenkins Build Pipeline oder durch die Einbindung in den „DATEV Marktplatz“ mittels eines entwickelten „Service Brokers“. gewährleistet werden. Aus Sicht der Administratoren sind mit dieser Umsetzung nur wenige allgemeine Templates zu verwalten, da die Entwickler mit z/OSPT Images arbeiten und keine weiteren Templates erzeugen. Um diese Punkte zu ermöglichen, muss das Template umgestaltet werden. Der dadurch in den Administratorenteams entstehende Aufwand und die damit verbundene steile Lernkurve hat eine abschreckende Wirkung.

Trotz dieser abschreckenden Wirkung sind auch die Administratorenteams bereit, falls die Kapazitäten vorhanden sind, den Bereitstellungsprozess mit Hilfe des „IBM Cloud Provisioning and Management for z/OS“ zu verbessern. Aus Sicht der Technologiestrategie ist dies ein wichtiger und notwendiger Schritt hin zu einem Cloud Native ähnlichen Prozess.

Kapitel 6.

Ausblick

Je weiter sich das Projekt der vorliegenden Arbeit dem Abschluss näherte desto mehr kristallisierte sich ein Hauptproblem heraus. Das erstellte Template ist sehr auf die DATEV Rechnungsschreibung spezialisiert, das heißt, es ist funktionsfähig, kann aber nicht ohne zeitaufwändige Eingriffe in das Template, in die Workflowdefinitionsdateien und die eigentlichen REXX Skripte und Jobs, an eine andere Anwendung angepasst werden. Folglich müsste das Template dynamischer implementiert sein. Um dies zu verdeutlichen, wird als Beispiel die Provisionierung von IBM MQ Queues herangezogen. Momentan werden die Prozesse und die Trigger Queues statisch angelegt. Das heißt, dass sowohl Namen als auch die damit verknüpften Queueparameter fest hinterlegt sind, um nur ein Beispiel zu nennen. Besser wäre es, alle Parameter in der Eingabedatei des Templates anzugeben. Aus IBM MQ Sicht ist hinzuzufügen, dass die fehlende automatisierte Bereitstellung eines Queue Managers den gewünschten Effekt, einer weitgehende Automatisierung und Unabhängigkeit von der Administration, abschwächt. Während der Realisierung stellte sich ebenfalls heraus, dass ein Template, das mehrere Subsysteme beinhaltet und dadurch sehr anwendungsspezifisch ist, nicht für einen firmenweiten Einsatz geeignet ist. So ist zu empfehlen, dass für jedes Subsystem, also CICS, Db2 und IBM MQ, ein separates Template realisiert wird.

Angenommen es besteht für jedes Subsystem ein Template und das IBM MQ Template beinhaltet die Provisionierung eines Queue Managers, so könnte jeder Entwickler seine eigene Instanzen der Templates besitzen und beispielsweise für eigene Tests nutzen. Dennoch wäre der ermöglichte Bereitstellungsprozess nicht optimal. So müsste für jede kleine Änderung an der Konfigurationsdatei ein neues Template erzeugt werden, siehe zweiter Fall im Abschnitt 5.3.2. Das dort genannte Beispiel einer CICS-Instanz und eindeutigen Application IDs wird hier aufgegriffen. Eine Möglichkeit diese Problem zu lösen, wäre einen Pool mit verfügbaren Application IDs bereitzustellen und dann mittels eines Programms eine ungenutzte Application ID zu bestimmen. Dieses Programm kann dann als Step in das Template aufgenommen werden. Jedoch müsste immer noch für jede kleine Änderung an der Konfigurationsdatei ein neues Template erzeugt werden.

Hier schafft z/OSPT Abhilfe. Damit kann, wie in Absatz 2.4.1 beschrieben, mit Hilfe einer Konfigurationsdatei das Template von außen gesteuert werden. Dadurch fällt das Kopieren

des Template für den Mitarbeiter weg, dieser muss mittels des Kommandozeileninterfaces ein Image bauen und daraus einen Container erzeugen. Das Kommandozeileninterface hat einen weiteren Vorteil. Mit dessen Hilfe können Arbeitsschritte für die Provisionierung der Middleware in einen Jenkins-Ablauf aufgenommen werden. Somit läuft der Prozess automatisiert ab und nähert sich modernen Entwicklungsabläufen wie denen aus der Cloud Native Entwicklung an.

Angenommen es existieren jeweils ein CICS, ein Db2 und ein IBM MQ Template und diese sind so realisiert, dass sie firmenweit eingesetzt werden können. Dann wäre der nächste Schritt, die Aufnahme in den „DATEV Marktplatz“, möglich. Der „DATEV Marktplatz“ ist eine Weboberfläche mit der sich Entwicklerteams ihre benötigte PaaS-Umgebung konfigurieren können. Heute stehen ihnen dort Dienste wie MongoDB, PostgreSQL, Kafka und viele weitere zur Verfügung. In weiter Zukunft könnten hier auch Dienste wie CICS, Db2 und IBM MQ zur Auswahl stehen. Dabei ist in Betracht zu ziehen, ob für den User nur bestimmte vorgefertigte Profile, wie „klein“, „mittel“ und „groß“, auswählbar sind. Die im Hintergrund verbundenen Templates und Images müssten dahingehend angepasst werden. Um einen solchen „Service Broker“ zu verwirklichen könnte die von z/OSMF zur Verfügung gestellte REST-API verwendet werden. Diese ermöglicht den Zugriff auf fast alle z/OSMF Funktionalitäten mittels Http-Requests. Für die „Tenant“ Zuweisung zu einem Template wird weiterhin die z/OSMF Oberfläche benötigt. Daran ist zu erkennen, dass von Seiten von z/OSMF beziehungsweise von IBM ebenfalls noch Verbesserungsmöglichkeiten bestehen.

Diese technische Umsetzung ermöglicht in Zukunft den in Diagramm 6.1 dargestellten Bereitstellungsprozess. Es ist zu erkennen, dass Verantwortung von den Administratorenteams an die Entwicklerteams übertragen wird. Dadurch wird Kommunikationsaufwand eingespart und einem Entwickler steht binnen weniger Minuten eine funktionsfähige Laufzeitumgebung für seine legacy z/OS Anwendung zur Verfügung. Bei Problemen oder Beratungswunsch unterstützen die Administratorenteams weiterhin. Für die Realisierung dieser Lösung ist viel Zeitaufwand vor allem auf Seiten der Administration einzuplanen.

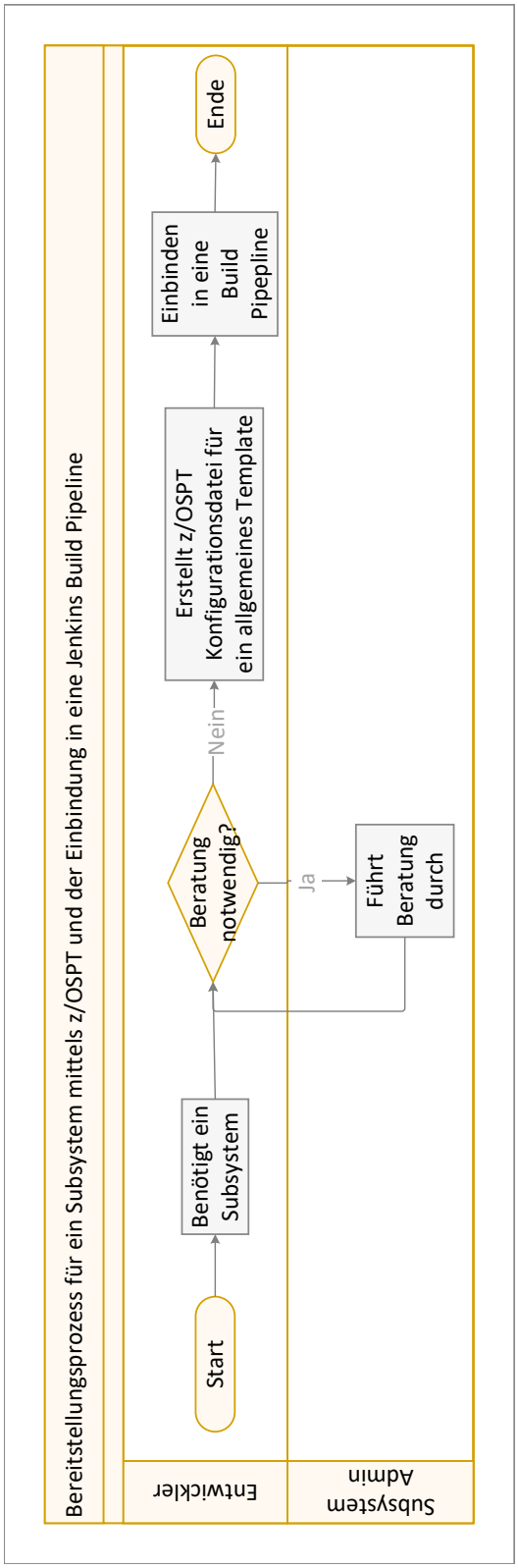


Abbildung 6.1.: Bereistellungsprozess eines Subsystems mittels einer z/OSPT Konfigurationsdatei

Kapitel 7.

Zusammenfassung

Zusammenfassend lässt sich sagen, dass es generell möglich ist mit dem „IBM Cloud Provisioning and Management for z/OS“-Toolkit Laufzeitumgebungen für legacy z/OS Anwendungen automatisiert bereitzustellen. Das funktionsfähige Template verkürzt den Bereitstellungsprozess deutlich. Durch den Abbau der Kommunikation zwischen den Abteilungen und nur einmaligem Erstellen der Skripte ist es zudem weniger fehleranfällig. Die Stakeholder sehen in diesem Template auch einen Mehrwert. Jedoch ist es noch nicht perfekt. Der Bereitstellungsprozess ist noch immer mit einigen manuellen Schritten verbunden. So muss das Template manuell kopiert werden und Änderungen an der Konfiguration müssen innerhalb des Templates stattfinden.

Hierfür wurde in der Arbeit eine Lösung mit Hilfe von z/OSPT beleuchtet. Diese sieht in einer Endausbaustufe eine einfache Einbindung des Templates in einen automatisierten Build-Prozess, zum Beipsiel mit Jenkins, vor. Außerdem würde der Einsatz von z/OSPT das Einbinden in den DATEV eG internen „Marktplatz“ für Cloud Lösungen ermöglichen. Um diese Ziele zu erreichen muss noch viel Aufwand in die Gestaltung des Templates gesteckt werden. Zusätzlich müsste ein sogenannter „Service Broker“ für die Einbindung der einzelnen Subsysteme in den „Marktplatz“ implementiert werden. Diese beiden Lösungsansätze stoßen sowohl bei den Administratorenteams als auch beim involvierten Entwicklerteam auf fruchtbaren Boden. Dadurch wird eine Ähnlichkeit zum Cloud Native-Bereitstellungsprozesses hergestellt. Dies ist ein weiterer Schritt um dem Image eines veralteten Systems mit veralteten langsamen Prozesses zu entkommen.

Anhang A.

Anhang

A.1. IT workload distribution worldwide in 2018 and 2020, by cloud type

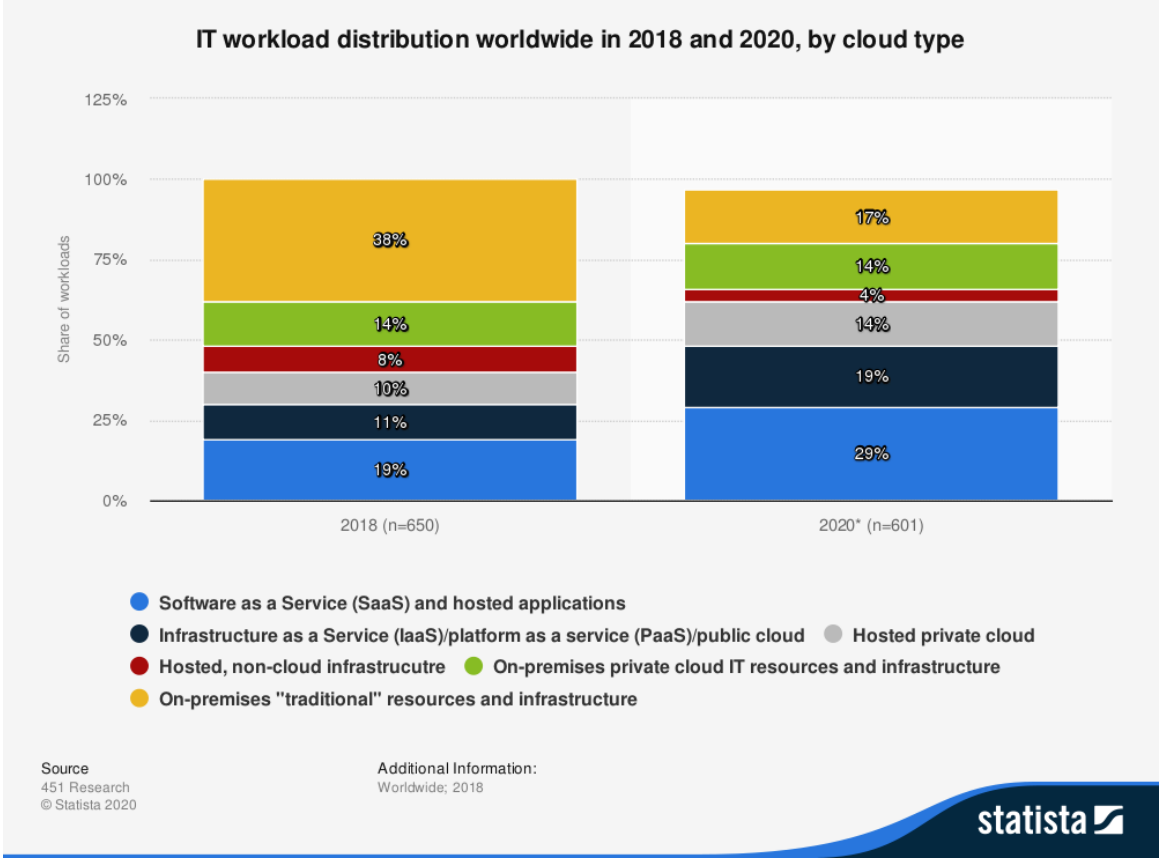


Abbildung A.1.: Weltweiter It Workload im Jahr 2018 und als Vorhersage im Jahr 2020 bei Cloudtyp

A.2. Produktstammdaten Tabellen Data Definition Language

,,

```
1  CREATE TABLE PSSSCHEMA.TPAUSPSSBASELBART
2          (PARTITIONID INTEGER
3                                                          NOT NULL
4  WITH DEFAULT 1
5          ,PID INTEGER
6                                                          NOT NULL
7          ,AUSPRAEGUNG SMALLINT
8                                                          NOT NULL
9          ,GUELTIGAB DATE
10                                                         NOT NULL
11         ,LFDNR INTEGER
12                                                         NOT NULL
13 WITH DEFAULT
14         ,GUELTIGBIS DATE
15                                                         NOT NULL
16 WITH DEFAULT "9999-12-31"
17         ,PSSID INTEGER
18 WITH DEFAULT NULL
19         ,ZUSATZID CHARACTER(4) FOR SBCS DATA
20 WITH DEFAULT NULL
21         ,BEZEICHNUNG VARCHAR(100) FOR SBCS DATA
22 WITH DEFAULT NULL
23         ,MWSTANTEILFREI DECIMAL(5, 2)
24 WITH DEFAULT NULL
25         ,MWSTANTEILREDUZIERT DECIMAL(5, 2)
26 WITH DEFAULT NULL
27         ,MWSTANTEILVOLL DECIMAL(5, 2)
28 WITH DEFAULT NULL
29         ,CONSTRAINT PID PRIMARY KEY
30         (PARTITIONID
31         ,PID
32         ,AUSPRAEGUNG
33         ,GUELTIGAB
34         ,LFDNR
35               )
36       )
37       IN DATABASE PSSBAPRV
```

```
38 APPEND NO
39 NOT VOLATILE CARDINALITY
40 DATA CAPTURE NONE
41 AUDIT NONE
42 CCSID EBCDIC
43 PARTITION BY RANGE
44        (PARTITIONID NULLS LAST ASC
45        )
46        ( PARTITION 1
47            ENDING ( 1
48        ) INCLUSIVE
49        , PARTITION 2
50            ENDING ( 2
51        ) INCLUSIVE
52        );
53
54 CREATE UNIQUE INDEX PSSSCHEMA.PPAUSPSSBASELBART
55          ON PSSSCHEMA.TPAUSPSSBASELBART
56         (PARTITIONID ASC
57         ,PID ASC
58         ,AUSPRAEGUNG ASC
59         ,GUELTIGAB ASC
60         ,LFDNR ASC
61         )
62          INCLUDE NULL KEYS
63          CLUSTER
64          PARTITIONED
65          DEFINE YES
66          COMPRESS NO
67          BUFFERPOOL BP2
68          CLOSE YES
69          DEFER NO
70          COPY NO
71          PARTITION BY RANGE
72         (PARTITION 1
73                    USING STOGROUP STAPSA01
74                                   PRIQTY -1
75                                   SECQTY -1
76                                   ERASE NO
77                    FREEPAGE 0
```

```
78                      PCTFREE 10
79                      GBPCACHE CHANGED
80          ,PARTITION 2
81                      USING STOGROUP STAPSA01
82                                  PRIQTY -1
83                                  SECQTY -1
84                                  ERASE NO
85                      FREEPAGE 0
86                      PCTFREE 10
87                      GBPCACHE CHANGED);
88
89    CREATE TABLE PSSSCHEMA.TMAXNUM
90          (MAXNUMID INTEGER
91                                                          NOT NULL
92          ,MAXNUM INTEGER
93                                                          NOT NULL
94          ,MAXNUBEZ CHARACTER(42) FOR SBCS DATA
95                                                          NOT NULL
96  WITH DEFAULT "X"
97          ,BEZEICHNUNG VARCHAR(100) FOR SBCS DATA
98                                                          NOT NULL
99  WITH DEFAULT "X"
100         ,CONSTRAINT MAXNUMID PRIMARY KEY
101         (MAXNUMID
102         )
103         )
104         IN DATABASE PSSBAPRV
105   APPEND NO
106   NOT VOLATILE CARDINALITY
107   DATA CAPTURE NONE
108   AUDIT NONE
109   CCSID EBCDIC;
110
111       COMMENT ON TABLE PSSSCHEMA.TMAXNUM
112             IS "maximale Nummer";
113
114
115 SET CURRENT SQLID = "DB2SADM";
116
117
```

```
118     COMMENT ON COLUMN PSSSCHEMA.TMAXNUM.MAXNUMID
119             IS "ID fuer maximalen Nummer";
120
121
122 SET CURRENT SQLID = "DB2SADM";
123
124
125     COMMENT ON COLUMN PSSSCHEMA.TMAXNUM.MAXNUM
126             IS "maximale Nummer";
127
128
129 SET CURRENT SQLID = "DB2SADM";
130
131
132     COMMENT ON COLUMN PSSSCHEMA.TMAXNUM.MAXNUBEZ
133             IS "Bezeichnung fuer maximale Nummer";
134
135
136 SET CURRENT SQLID = "DB2SADM";
137
138
139     COMMENT ON COLUMN PSSSCHEMA.TMAXNUM.BEZEICHNUNG
140             IS "Bezeichnung fuer maximale Nummer";
141
142 CREATE UNIQUE INDEX PSSSCHEMA.PMAXNUM
143           ON PSSSCHEMA.TMAXNUM
144          (MAXNUMID ASC
145          )
146           INCLUDE NULL KEYS
147           CLUSTER
148           DEFINE YES
149           COMPRESS NO
150           BUFFERPOOL BP2
151           CLOSE YES
152           DEFER NO
153           COPY NO
154           USING STOGROUP STALDL01
155                 PRIQTY -1
156                 SECQTY -1
157                 ERASE NO
```

```
158           FREEPAGE 0
159           PCTFREE 99
160           GBPCACHE CHANGED
161           PIECESIZE 2097152K;
162
163 CREATE FUNCTION PSS.WHICH_PARTITIONID
164     (
165      MAXID INTEGER )
166     RETURNS INTEGER
167     VERSION V1
168     DISALLOW DEBUG MODE
169     ASUTIME NO LIMIT
170     INHERIT SPECIAL REGISTERS
171     WLM ENVIRONMENT FOR DEBUG MODE DB0ITWLM
172     APPLICATION ENCODING SCHEME EBCDIC
173     QUALIFIER UGPSENT
174     DYNAMICRULES RUN
175     WITH EXPLAIN
176     WITHOUT IMMEDIATE WRITE
177     ISOLATION LEVEL UR
178     OPTHINT ""
179     REOPT NONE
180     VALIDATE RUN
181     ROUNDING DEC_ROUND_HALF_EVEN
182     DATE FORMAT ISO
183     DECIMAL( 31 )
184     FOR UPDATE CLAUSE REQUIRED
185     TIME FORMAT ISO
186     CURRENT DATA NO
187     DEGREE 1
188     PACKAGE OWNER UGPSENT
189     BUSINESS_TIME SENSITIVE NO
190     SYSTEM_TIME SENSITIVE NO
191     ARCHIVE SENSITIVE NO
192     APPLCOMPAT V10R1
193     LANGUAGE SQL
194     NO EXTERNAL ACTION
195     PARAMETER CCSID EBCDIC
196     DETERMINISTIC
197             NOT SECURED
```

```
198         CALLED ON NULL INPUT
199         READS SQL DATA
200         SPECIFIC WHICH_PARTITIONID
201 BEGIN
202          DECLARE MAXNUM INTEGER;
203          SELECT MAXNUM
204                 INTO MAXNUM
205            FROM AVADMIN.AMAXNUM
206           WHERE MAXNUMID = MAXID;
207          RETURN MAXNUM;
208 END;
209
210 SET PATH = "PSS","SYSIBM","SYSFUN","SYSPROC","SYSIBMADM","PSSSCHEMA";
211
212 CREATE VIEW PSSSCHEMA.VPAUSPSS_BASELBART
213           ( PARTITIONID
214           , PID
215           , AUSPRAEGUNG
216           , GUELTIGAB
217           , LFDNR
218           , GUELTIGBIS
219           , PSSID
220           , ZUSATZID
221           , BEZEICHNUNG
222           , MWSTANTEILFREI
223           , MWSTANTEILREDUZIERT
224           , MWSTANTEILVOLL
225           ) AS
226 SELECT B.* FROM TPAUSPSSBASELBART B WHERE B.PARTITIONID =
227   PSS.WHICH_PARTITIONID ( 3011 )
228 ;
229
230   CREATE TABLE PSSSCHEMA.TPAUSPSSPREISE
231         (PARTITIONID INTEGER
232                                                     NOT NULL
233 WITH DEFAULT 1
234         ,ARTNR INTEGER
235                                                     NOT NULL
236         ,PREISTYPID SMALLINT
237                                                     NOT NULL
```

```
238         ,GUELTIGAB DATE
239                                                                 NOT NULL
240         ,STAFFELNR INTEGER
241                                                                 NOT NULL
242         ,PID INTEGER
243                                                                 NOT NULL
244 WITH DEFAULT
245         ,PREISREGEL CHARACTER(2) FOR SBCS DATA
246                                                                 NOT NULL
247 WITH DEFAULT "X"
248         ,GUELTIGBIS DATE
249                                                                 NOT NULL
250 WITH DEFAULT "9999-12-31"
251         ,PRODUKTPREIS DECIMAL(11, 3)
252 WITH DEFAULT NULL
253         ,EINZELPREIS DECIMAL(8, 3)
254 WITH DEFAULT NULL
255         ,PREISINTERVALL DECIMAL(11, 3)
256 WITH DEFAULT NULL
257         ,PREISEINHEIT DECIMAL(8, 3)
258 WITH DEFAULT NULL
259         ,STAFFELVERTEILUNG CHARACTER(1) FOR SBCS DATA
260 WITH DEFAULT NULL
261         ,INTERVALLVON INTEGER
262 WITH DEFAULT NULL
263         ,INTERVALLBIS INTEGER
264 WITH DEFAULT NULL
265         ,PREISTYPBEZ VARCHAR(100) FOR SBCS DATA
266                                                                 NOT NULL
267 WITH DEFAULT "X"
268         ,PREISAB DECIMAL(8, 3)
269 WITH DEFAULT NULL
270         ,PREISABRELEVANZ CHARACTER(1) FOR SBCS DATA
271                                                                 NOT NULL
272 WITH DEFAULT "K"
273         ,EINHEIT INTEGER
274 WITH DEFAULT NULL
275         ,CONSTRAINT PPAUSPSSPREISE PRIMARY KEY
276         (PARTITIONID
277         ,ARTNR
```

```
278          ,PREISTYPID
279          ,GUELTIGAB
280          ,STAFFELNR
281          )
282          )
283         IN DATABASE PSSBAPRV
284  APPEND NO
285  NOT VOLATILE CARDINALITY
286  DATA CAPTURE NONE
287  AUDIT NONE
288  CCSID EBCDIC;
289
290  CREATE INDEX PSSSCHEMA.IPAUSPSSPREISE
291          ON PSSSCHEMA.TPAUSPSSPREISE
292         (PARTITIONID ASC
293         ,PID ASC
294         )
295          INCLUDE NULL KEYS
296          NOT CLUSTER
297          DEFINE YES
298          COMPRESS NO
299          BUFFERPOOL BP2
300          CLOSE YES
301          DEFER NO
302          COPY NO
303          USING STOGROUP STAPSA01
304               PRIQTY -1
305               SECQTY -1
306               ERASE NO
307          FREEPAGE 0
308          PCTFREE 10
309          GBPCACHE CHANGED
310          PIECESIZE 2097152K;
311
312 CREATE INDEX PSSSCHEMA.IPAUSPSSPREISE2
313          ON PSSSCHEMA.TPAUSPSSPREISE
314         (PARTITIONID ASC
315         ,ARTNR ASC
316         ,PREISTYPID ASC
317         ,GUELTIGAB ASC
```

```
318         ,INTERVALLVON ASC
319         )
320          INCLUDE NULL KEYS
321          NOT CLUSTER
322          DEFINE YES
323          COMPRESS NO
324          BUFFERPOOL BP2
325          CLOSE YES
326          DEFER NO
327          COPY NO
328          USING STOGROUP STAPSA01
329               PRIQTY -1
330               SECQTY -1
331               ERASE NO
332          FREEPAGE 0
333          PCTFREE 10
334          GBPCACHE CHANGED
335          PIECESIZE 2097152K;
336
337 CREATE UNIQUE INDEX PSSSCHEMA.PPAUSPSSPREISE
338          ON PSSSCHEMA.TPAUSPSSPREISE
339         (PARTITIONID ASC
340         ,ARTNR ASC
341         ,PREISTYPID ASC
342         ,GUELTIGAB ASC
343         ,STAFFELNR ASC
344         )
345          INCLUDE NULL KEYS
346          CLUSTER
347          DEFINE YES
348          COMPRESS NO
349          BUFFERPOOL BP2
350          CLOSE YES
351          DEFER NO
352          COPY NO
353          USING STOGROUP STAPSA01
354               PRIQTY -1
355               SECQTY -1
356               ERASE NO
357          FREEPAGE 0
```

```
358           PCTFREE 10
359           GBPCACHE CHANGED
360           PIECESIZE 2097152K;
361
362     CREATE TABLE PSSSCHEMA.TPAUSPSSBART
363           (PARTITIONID INTEGER
364                                                          NOT NULL
365   WITH DEFAULT 1
366           ,PID INTEGER
367                                                          NOT NULL
368           ,ARTNR INTEGER
369                                                          NOT NULL
370   WITH DEFAULT
371           ,ANDAT DATE
372                                                          NOT NULL
373   WITH DEFAULT "1966-02-14"
374           ,OPDATBEN CHARACTER(1) FOR SBCS DATA
375                                                          NOT NULL
376   WITH DEFAULT "J"
377           ,AUSDIENSTREL CHARACTER(1) FOR SBCS DATA
378                                                          NOT NULL
379   WITH DEFAULT "N"
380           ,VERTRIEBSREL CHARACTER(1) FOR SBCS DATA
381                                                          NOT NULL
382   WITH DEFAULT "N"
383           ,VERTRELDAT DATE
384   WITH DEFAULT NULL
385           ,KOMMASTELLEN INTEGER
386   WITH DEFAULT NULL
387           ,ERTRNR INTEGER
388                                                          NOT NULL
389   WITH DEFAULT
390           ,GFEDNR INTEGER
391                                                          NOT NULL
392   WITH DEFAULT
393           ,UPLONR INTEGER
394                                                          NOT NULL
395   WITH DEFAULT
396           ,MWSTEUERSATZ INTEGER
397   WITH DEFAULT NULL
```

```
398         ,POLINR INTEGER
399                                                         NOT NULL
400 WITH DEFAULT
401         ,EXPGNR INTEGER
402                                                         NOT NULL
403 WITH DEFAULT
404         ,EXPONR INTEGER
405                                                         NOT NULL
406 WITH DEFAULT
407         ,ARTIKELTYPID INTEGER
408                                                         NOT NULL
409 WITH DEFAULT
410         ,ARTIKELTYPALT CHARACTER(4) FOR SBCS DATA
411                                                         NOT NULL
412 WITH DEFAULT '0000'
413         ,BERBESTEINHID INTEGER
414                                                         NOT NULL
415 WITH DEFAULT
416         ,BERBESTEINHALT CHARACTER(4) FOR SBCS DATA
417                                                         NOT NULL
418 WITH DEFAULT '0000'
419         ,LEISTGRUPID INTEGER
420                                                         NOT NULL
421 WITH DEFAULT
422         ,LEISTGRUPALT CHARACTER(4) FOR SBCS DATA
423                                                         NOT NULL
424 WITH DEFAULT '0000'
425         ,BERFREQID INTEGER
426                                                         NOT NULL
427 WITH DEFAULT
428         ,NUTZERID INTEGER
429 WITH DEFAULT
430         ,LEISTARTID INTEGER
431                                                         NOT NULL
432 WITH DEFAULT
433         ,BERFREQALT CHARACTER(4) FOR SBCS DATA
434                                                         NOT NULL
435 WITH DEFAULT '0000'
436         ,LEISTARTALT CHARACTER(4) FOR SBCS DATA
437                                                         NOT NULL
```

```
438 WITH DEFAULT "99"
439         ,NUTZERALT CHARACTER(1) FOR SBCS DATA
440                                                 NOT NULL
441 WITH DEFAULT "K"
442         ,BARTBEZ_20 CHARACTER(20) FOR SBCS DATA
443                                                 NOT NULL
444 WITH DEFAULT "X"
445         ,ARTIKELTYPBEZ CHARACTER(50) FOR SBCS DATA
446                                                 NOT NULL
447 WITH DEFAULT "Keine Zuordnung"
448         ,BERBESTEINHBEZ CHARACTER(50) FOR SBCS DATA
449                                                 NOT NULL
450 WITH DEFAULT "Keine Zuordnung"
451         ,LEISTGRUPBEZ CHARACTER(50) FOR SBCS DATA
452                                                 NOT NULL
453 WITH DEFAULT "Keine Zuordnung"
454         ,BERFREQBEZ CHARACTER(50) FOR SBCS DATA
455                                                 NOT NULL
456 WITH DEFAULT "Keine Zuordnung"
457         ,NUTZERBEZ CHARACTER(50) FOR SBCS DATA
458                                                 NOT NULL
459 WITH DEFAULT "Keine Zuordnung"
460         ,LEISTARTBEZ CHARACTER(50) FOR SBCS DATA
461                                                 NOT NULL
462 WITH DEFAULT "Keine Zuordnung"
463         ,BARTBEZ_100 VARCHAR(100) FOR SBCS DATA
464                                                 NOT NULL
465 WITH DEFAULT "X"
466         ,ERTRBEZ VARCHAR(100) FOR SBCS DATA
467                                                 NOT NULL
468 WITH DEFAULT "X"
469         ,GFEDBEZ VARCHAR(100) FOR SBCS DATA
470                                                 NOT NULL
471 WITH DEFAULT "X"
472         ,UPLOBEZ VARCHAR(100) FOR SBCS DATA
473                                                 NOT NULL
474 WITH DEFAULT "X"
475         ,POLIBEZ VARCHAR(100) FOR SBCS DATA
476                                                 NOT NULL
477 WITH DEFAULT "X"
```

```
478         ,EXPGBEZ VARCHAR(100) FOR SBCS DATA
479                                                          NOT NULL
480 WITH DEFAULT "X"
481         ,EXPOBEZ VARCHAR(100) FOR SBCS DATA
482                                                          NOT NULL
483 WITH DEFAULT "X"
484         ,HAKONR INTEGER
485 WITH DEFAULT NULL
486         ,HAKOBEZ VARCHAR(100) FOR SBCS DATA
487 WITH DEFAULT NULL
488         ,INPGNR INTEGER
489                                                          NOT NULL
490 WITH DEFAULT
491         ,INPGBEZ VARCHAR(100) FOR SBCS DATA
492                                                          NOT NULL
493 WITH DEFAULT "X"
494         ,BEZ035 CHARACTER(35) FOR SBCS DATA
495                                                          NOT NULL
496 WITH DEFAULT "X"
497         ,CONSTRAINT PPAUSPSSBART PRIMARY KEY
498         (PARTITIONID
499         ,PID
500         )
501         ,CONSTRAINT UPAUSPSSBART UNIQUE
502         (PARTITIONID
503         ,ARTNR
504         )
505         )
506         IN DATABASE PSSBAPRV
507  APPEND NO
508  NOT VOLATILE CARDINALITY
509  DATA CAPTURE NONE
510  AUDIT NONE
511  CCSID EBCDIC
512  PARTITION BY RANGE
513         (PARTITIONID NULLS LAST ASC
514         )
515         ( PARTITION 1
516             ENDING ( 1
517         ) INCLUSIVE
```

```
518         , PARTITION 2
519             ENDING ( 2
520         ) INCLUSIVE
521         );
522
523 CREATE UNIQUE INDEX PSSSCHEMA.PPAUSPSSBART
524         ON PSSSCHEMA.TPAUSPSSBART
525         (PARTITIONID ASC
526         ,PID ASC
527         )
528         INCLUDE NULL KEYS
529         CLUSTER
530         PARTITIONED
531         DEFINE YES
532         COMPRESS NO
533         BUFFERPOOL BP2
534         CLOSE YES
535         DEFER NO
536         COPY NO
537         PARTITION BY RANGE
538         (PARTITION 1
539                     USING STOGROUP STAPSA01
540                                    PRIQTY -1
541                                    SECQTY -1
542                                    ERASE NO
543                     FREEPAGE 0
544                     PCTFREE 10
545                     GBPCACHE CHANGED
546         ,PARTITION 2
547                     USING STOGROUP STAPSA01
548                                    PRIQTY -1
549                                    SECQTY -1
550                                    ERASE NO
551                     FREEPAGE 0
552                     PCTFREE 10
553                     GBPCACHE CHANGED);
554
555 CREATE UNIQUE INDEX PSSSCHEMA.UPAUSPSSBART
556         ON PSSSCHEMA.TPAUSPSSBART
557         (PARTITIONID ASC
```

```
558         ,ARTNR ASC
559         )
560          INCLUDE NULL KEYS
561         NOT CLUSTER
562          DEFINE YES
563          COMPRESS NO
564          BUFFERPOOL BP2
565          CLOSE YES
566          DEFER NO
567          COPY NO
568          USING STOGROUP STAPSA01
569               PRIQTY -1
570               SECQTY -1
571               ERASE NO
572          FREEPAGE 0
573          PCTFREE 10
574          GBPCACHE CHANGED
575          PIECESIZE 2097152K;
576
577 CREATE UNIQUE INDEX PSSSCHEMA.UPAUSPSSBART
578          ON PSSSCHEMA.TPAUSPSSBART
579         (PARTITIONID ASC
580         ,ARTNR ASC
581         )
582          INCLUDE NULL KEYS
583         NOT CLUSTER
584          DEFINE YES
585          COMPRESS NO
586          BUFFERPOOL BP2
587          CLOSE YES
588          DEFER NO
589          COPY NO
590          USING STOGROUP STAPSA01
591               PRIQTY -1
592               SECQTY -1
593               ERASE NO
594          FREEPAGE 0
595          PCTFREE 10
596          GBPCACHE CHANGED
597          PIECESIZE 2097152K;
```

A.3. Interview Fragebögen

Vorlage

1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**

1 2 3 4 5
☐ ☐ ☐ ☐ ☐

2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**

1 2 3 4 5
☐ ☐ ☐ ☐ ☐

3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**

1 2 3 4 5
☐ ☐ ☐ ☐ ☐

   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**

1 2 3 4 5
☐ ☐ ☐ ☐ ☐

4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**

1 2 3 4 5
☐ ☐ ☐ ☐ ☐

5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**

6. **Wie würden Sie den aktuellen Bereitstellungsprozess beurteilen?**

7. **Können Sie sich vorstellen, mit dem Toolkit täglich zu arbeiten?**

8. **Wenn 7. Mit ja beantwortet wurde, begründen Sie ihre Meinung.**

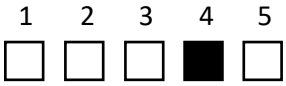
9. **Wenn 7. Mit nein beantwortet wurde, was müsste sich ändern, dass dem so wäre?**

10. **Freitext für sonstiges und Anmerkungen:**

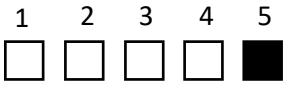
CICS Administrator 1

1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**

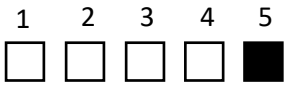
   + flexibel Versionierung und Publish
   - Startschwierigkeiten viele verschiedene Sprachen und Dokumentarten
2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**



   + APIfizierung
   + Container-Gedanke
   + konfigurierbar über PT-File von außerhalb der Templates
   -Template muss sehr dynamisch sein
3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**



   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**

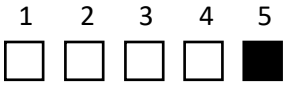


4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**

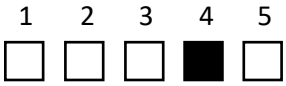
   + lediglich editieren der Files
   -Fehlendes Highlighting für „mixed JCL“
5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**
   - Hoher Ersteinrichtungsaufwand
   - Einarbeitung
   - Abschreckende Wirkung (verschiedene Sprachen usw.)

CICS Administrator 2

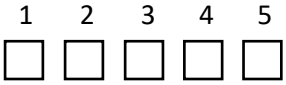
1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**



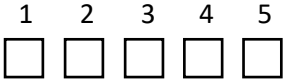
   Das Template ist aus CICS-Sicht ablauffähig und mehrfach einsetzbar.
2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**



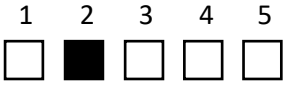
   Großer Vorteil ist die Flexibilität durch den Einsatz von Variablen. Nachteil ist die damit verbundene Komplexität des dahinterliegenden Templates.
3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   Die Oberfläche wurde vom CICS-Team bisher nicht genutzt, daher ist keine Bewertung möglich. Zudem ist wenig Erfahrung mit vergleichbaren Tools wie Cloud Foundry vorhanden.
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**



   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**



4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**



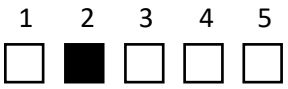
   Keine Toolunterstützung. Editieren per Notepad ohne Highlighting. Keine sprechenden Fehlermeldungen. Dokumentation der JCL-Skriptsprache nicht vorhanden.
5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**
   Prinzipiell kann mit dem Toolkit alles erreicht werden, allerdings ist sehr viel Anpassungsarbeit notwendig, um es auf die Firmengegebenheiten anzupassen. Hilfreich wären mehr Beispiele, bessere Dokumentation, Step by Step Anleitung oder ein Wizard.
6. **Wie würden Sie den aktuellen Bereitstellungsprozess beurteilen?**
   Hoher manueller Aufwand zu erbringen. Kein SelfService für den Entwickler vorhanden. Vorherige Abstimmung zwischen Sysprog und Entwicklung notwendig.
7. **Können Sie sich vorstellen, mit dem Toolkit täglich zu arbeiten?**
   Jein, man könnte es sich vorstellen, aber es gibt auch viele Hürden.

Db2 Administrator 1

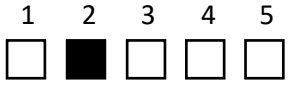
1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**

   Der Ablauf zeigt, dass eine Automatisierung möglich ist. Diese Erkenntnis ist sehr wertvoll. Um die Provisionierung aber wirklich nutzen zu können ist noch viel Weiterentwicklung notwendig.
2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**

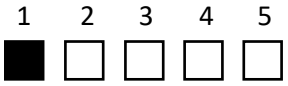
   Für den aktuellen Stand der Untersuchung halte ich z/OSPT noch nicht für relevant. In der Endausbaustufe (Automatisiertes Deployment innerhalb einer CI/DC-Pipeline z.B. mit Jenkins) wird ein CLI-Interface wie z/OSPT aber sehr wichtig und vereinfacht die Nutzung für Entwickler
3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**



   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**



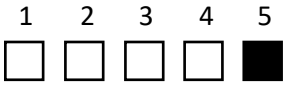
4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**



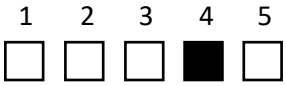
   (wenn es ohne automatisches Deployment gemacht wird)
5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**
   Mächtiges Tool aber auch sehr komplex
6. **Wie würden Sie den aktuellen Bereitstellungsprozess beurteilen?**
   Er funktioniert sehr gut aber man ist von anderen Personen abhängig und hat dadurch natürlich Wartezeiten, die man mit einem automatischen Prozess eliminieren würde.
7. **Können Sie sich vorstellen, mit dem Toolkit täglich zu arbeiten?**
   Ja
8. **Wenn 7. Mit ja beantwortet wurde, begründen Sie ihre Meinung.**
   Nur wenn man es kapselt (jenkins)

Db2 Administrator 2

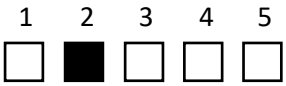
1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**



   Ablauf ist für mich nachvollziehbar. Siehe auch Anmerkungen (Frage 10)
2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**



   Die Nutzung von Z/OSPT macht Sinn, denn die Commands (z.B. build und run) sind meiner Meinung nach einfach in einen Quellcode zu integrieren und geläufig. Problematisch sehe ich jedoch die Verwendung der Begriffe Container und Image, da hier Begriffe vertauscht und synonym verwendet werden.
3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**



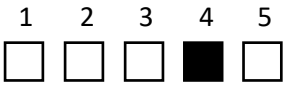
   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**

4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**

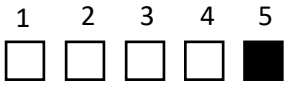
   Besser wäre es, wenn zur Bearbeitung eine IDE verwendet werden würde (Syntax)
5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**
   Zukunft des modernen Deployments auf dem Mainframe. Ähnlich der offenen Welt. Bringt die Plattform z nach vorne!
6. **Wie würden Sie den aktuellen Bereitstellungsprozess beurteilen?**
   Aktuell noch sehr komplex. Die Bereitstellung ist aktuell noch sehr anwendungsspezifisch und sehr statisch. Es wird ein sehr umfangreiches Wissen über alles beteiligten Subsysteme benötigt. Hoher Konfigurationsaufwand und Vorarbeit von Nöten (Rechtekonzept, Funktionsuser, etc.)
7. **Können Sie sich vorstellen, mit dem Toolkit täglich zu arbeiten?**
   Ja.

IBM MQ Administrator 1

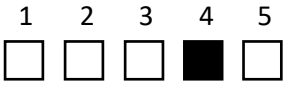
1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**



   - Mehrwert dadurch, dass mehr Verantwortung bei dem Entwickler ist
   - Weniger händische Eingriffe
2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**

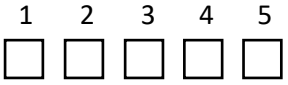


   - Wird das von IBM noch weiterentwickelt?
   - Features, die bereits vorhanden sind, sind schon gut
   - Flexibilität ist höher als mit dem Ablauf aus 1.
3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**



   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**

4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**

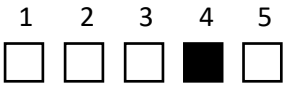


   Da noch nicht selbst damit gearbeitet wurde, kann es nicht beurteilt werden.
5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**
   - Gezeigtes ist gut, aber Zeitaufwand ist mit einzubeziehen und die zu leistenden Vorarbeiten
   - MQ Queue Manager ist komplexer → noch mehr Zeitaufwand und notwendige Vorarbeiten werden mehr
6. **Wie würden Sie den aktuellen Bereitstellungsprozess beurteilen?**
   - Deutlich manueller Arbeitsablauf
   - Viele Rückfragen und viel Arbeiten auf Zuruf, Kommunikation über Email, Telefon oder Termine

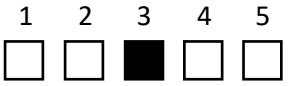
IBM MQ Administrator 2

1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**

   - Fehlender Umfang (speziell für MQ)
   - Nur spezifische Queues mit speziellen Parametern
2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**



   - Weiterhin Abstimmung mit Dritten (RACF, IP, Storage) notwendig
   - Nur „pseudo“ Docker Container
3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**



   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**

4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**

   - Automation nicht vorhanden
     - Zum Beispiel keine Analogie zur Jenkins-Replay-Funktion
5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**
   - Viele gute Ansätze
   - Nicht einfach genug zu verwenden im Vergleich zu Jenkins und einer PaaS Lösung
6. **Wie würden Sie den aktuellen Bereitstellungsprozess beurteilen?**
   - Deutlich manueller Arbeitsablauf
   - Viele Rückfragen und viel Arbeiten auf Zuruf, Kommunikation über Email, Telefon oder Termine

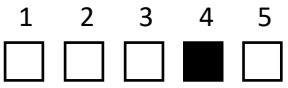
7. **Können Sie sich vorstellen, mit dem Toolkit täglich zu arbeiten?**
   Ja und Nein

Entwickler 1

1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**

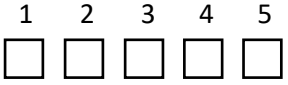
   - Es wird viel Wissen (bezüglich der Oberfläche und des Templates) benötigt und dieses muss auch bei geringer Nutzung über einen längeren Zeitraum erhalten werden.
   - Außerdem ist weiterhin viel Zuarbeit der Administration notwendig.
   - Es stellt sich die Frage, wer die DDL für Datenbanken erstellt.
   - Ansonsten schon ganz gut.

2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**

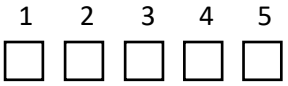


   - Vor allem ein Ausblick für eine Nutzung in der QS und Produktionsstages
   - Ablauf mehr in den Händen des eigenen Teams → höhere Effizienz, aber auch höhere Verantwortung.
   - Unkomplizierte Nutzung mittels Jenkins und den „DATEV Marktplatz“

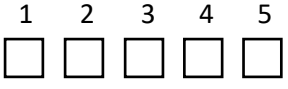
3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**



   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**



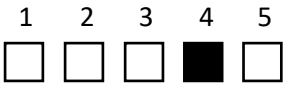
4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**



5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**
   ---
6. **Wie würden Sie den aktuellen Bereitstellungsprozess beurteilen?**
   - Zeitaufwändig durch viele Absprachen bezüglich Erstaufwand.
   - Wenns läuft, läufts.
   - Abhängigkeit von dritten (z.B. Admins)
7. **Können Sie sich vorstellen, mit dem Toolkit täglich zu arbeiten?**
   Ja

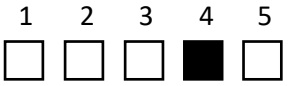
Mitarbeiter des Technologiestrategieteams

1. **Es wurde ein Template für die Rechnungsschreibung, welches ein CICS, die benötigten MQ Queues und theoretisch die benötigte Db2 Datenbanken beinhaltet, vorgestellt. Der Ablauf, der damit einhergeht, beschränkt sich zunächst auf z/OSMF. Bewerten Sie diesen, begründen Sie Ihre Bewertung.**



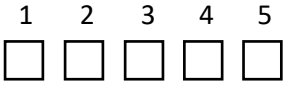
   z/OSMF stellt sich für mich als eine Oberfläche dar, die ich für eine solche „Task“ nutzen kann. Sieht einfach aus. Besser würde es mir gefallen, wenn ich den bereits existierenden „Marketplace“ unserer DATEV Cloud Lösung nutzen könnte.

2. **Es wurde eine Ergänzung mit z/OSPT, zu oben genannten Ablauf, erläutert. Bewerten Sie diese, begründen Sie Ihre Bewertung.**

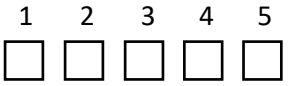


   Unkomplizierte Nutzung mittels Jenkins und den „DATEV Marktplatz“ Die Konfiguration eines Skriptes mit z/OSPT ist für die Einbindung der Provisionierung in automatische Build-Prozesse hilfreich, und damit wichtig. Es muss sich jemand um den Aufbau der Build-Pipeline kümmern, (Build Engineer) die Rolle haben wir aktuell in den z/OS Projekten noch nicht. Aber es macht Sinn und bringt die z/OS Anwendungen näher an die Vorgehensweise der Cloud Native Entwicklung.

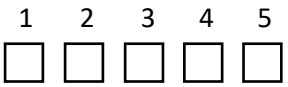
3. **Bewerten Sie folgende Punkte bezüglich der Benutzerfreundlichkeit der Oberfläche:**
   a. **Verwaltung der Templates in z/OSMF (nicht für Entwickler relevant)**



   b. **Verwaltung der Instanzen in z/OSMF (nicht für Entwickler relevant)**



4. **Bewerten Sie die gezeigte Arbeitsweise für Änderungen an den Workflow Definitionsdateien. (nicht für Entwickler relevant)**



5. **Wie ist Ihr erster persönlicher Eindruck zum Toolkit? (nicht für Entwickler relevant)**
   ---
6. **Wie würden Sie den aktuellen Bereitstellungsprozess beurteilen?**
   Aktuell teilen sich n Entwickler an einem Produkt die gleiche Entwicklungsumgebung, arbeiten im gleichen CICS und auf der gleichen Test-Datenbank. Änderungen beeinflussen auch die Tests der anderen Kollegen, und müssen koordiniert werden. Die

A.4. Workflow Step mit REST-Call

,,

```
<step name="db2_create_database">
        <title>db2_create_database.</title>
        <description>Erzeugt mit der Hilfe des DB2
      Service Brokers eine Datenbank</description>
        <instructions substitution="false">
      Erzeugt DB2 Datenbank.</instructions>
        <weight>10</weight>
        <skills>REST</skills>
      <autoEnable>true</autoEnable>
      <rest>
      <httpMethod>PUT</httpMethod>
      <schemeName substitution="false">http</schemeName>
      <hostname substitution="false">hostname</hostname>
      <uriPath>uriPath</uriPath>
      <requestBody substitution="true">
{
  "service_id": "${instance-DFH_DB2_SERVICEID}",
  "plan_id": "${instance-DFH_DB2_PLANID}",
 "organization_guid": "DUMMY",
  "space_guid": "DUMMY",
  "parameters": {
            "GROUP": "UGZTAL",
            "VUSERID": "${_step-stepOwnerUpper}",
            "DBNAME": "${instance-DFH_DB2_DATABASENAME}",
            "UserID": "${_step-stepOwnerUpper}",
            "Passwort": "DUMMY"
  }
}
      </requestBody>
      <expectedStatusCode>201</expectedStatusCode>
      <requestHeaders substitution="false">
      {"Authorization":"Basic VDMwMTkzQTpYZjN1I2RJNA=="}
      </requestHeaders>
      </rest>
   </step>
```

Abbildungsverzeichnis

Tabellenverzeichnis

Quellcodeverzeichnis

Literaturverzeichnis

[Also 93] S. Alsop. “IBM still has the brains to be a player in client/server platforms”. *InfoWorld*, Vol. 15, No. 10, p. 4, 1993.

[Aran 13] C. Aranha. *IBM WebSphere MQ V7.1 and V7.5 features and enhancements. IBM redbooks*, IMB Corp. International Technical Support Organization, Poughkeepsie, NY, 1st ed. Ed., 2013.

[Cass 07] P. Cassier. *System programmer's guide to Workload manager. IBM redbooks*, IBM International Technical Support Organization, United States?, 4th ed. Ed., 2007.

[Ceru 03] P. E. Ceruzzi. *A history of modern computing. History of computing*, MIT Press, Cambridge, Mass., 2. ed. Ed., 2003.

[DATE 17] DATEV eG. “Geschichte der Datev”. 2017.

[Ebbe 11] M. Ebbers, J. Kettner, W. O'Brien, and B. Ogden. *Introduction to the new mainframe: Z/OS basics. IBM redbooks*, IBM Corporation International Technical Support Organization, Poughkeepsie, NY, third edition , (march 2011) Ed., 2011.

[IBM 14] IBM. “Access method control block (ACB)”. 2014.

[IBM 19a] IBM. “CICS Application Server Software for IBM Z”. 2019.

[IBM 19b] IBM. “Using IBM z/OS Provisioning Toolkit”. 2019.

[inte] internetagentur Köln Frankfurt sunzinet TYPO3 Programmmierung. “Studieren in Deutschland und promovieren in Deutschland - Hochschulkompass”.

[lega 20] legacy. “Was ist ein Legacy System?”. 22.2.2020.

[Love 13] M. Lovelace. *VSAM demystified. IBM redbooks*, IBM Corp. International Technical Support Organization, Poughkeepsie, NY, 3rd ed. Ed., 2013.

[Rayn 11] C. Rayns. *CICS transaction server from start to finish. Redbooks*, IBM International Technical Support Organization, Poughkeepsie, N.Y., 2011.

[Rott 18] R. J. T. Rotthove. *IBM z/OS Management Facility V2R3. Redbooks*, IBM Redbooks, [Place of publication not identified], 2018.

[Stee 03] B. Steegmans. *DB2 for z/OS and OS/390: Ready for Java. IBM redbooks*, IBM, International Technical Support Organization, [S.l.], 1st ed. Ed., 2003.

[Sull 16] D. Sullivan. “Google now handles at least 2 trillion searches per year - Search Engine Land”. 2016.