Industrie- und Handelskammer Hannover

Abschlussprüfung Winter 2024/25 Fachinformatikerin

für Daten- und Prozessanalyse

Dokumentation zur betrieblichen Projektarbeit

Einführung einer neuen Vergütungssystematik für medizinische Leistungen mittels Datenaustauschverfahrens

Nutzung von RPA zur Vermeidung neuer manueller Aufwände durch Anpassung der bestehenden RPA-Übergangslösung an die neue Datenstruktur

Abgabedatum: Hannover, den Datum

Prüfungsbewerberin:

Name
Straße
Ort PLZ
Prüflingsnr.: Nummer

Ausbildungsbetrieb:

Kaufmännische Krankenkasse - KKH
Karl-Wiechert-Allee 61
30625 Hannover

KKH

Inhaltsverzeichnis

Abkürzungsverzeichnis

Abkürzung	Bedeutung
BPMN	Business Process Model and Notation
ERM	Entity-Relationship-Model
JSON	JavaScript Object Notation
KKH	Kaufmännische Krankenkasse - KKH
MRSA	Morbiditätsorientierter Risikostrukturausgleich
RPA	Robotic Process Automation
SQL	Structured Query Language
UML	Unified Modeling Language
vdek	Verband der Ersatzkassen

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar

Begriff	Erklärung
ADA	KKH-Webanwendung zur Bearbeitung von Abrechnungsdaten
Aktivitäten	Bausteine, die in Automatisierungsprozessen Aktionen wie Dateneingabe, Logikentscheidungen oder Dateioperationen ausführen
Fachlicher Dispatcher	Prozess, der auf neue Rechnungen prüft und die Daten dann in die passende Tabelle überträgt
Hybrid-DRG	Abrechnungssystem für diagnosebezogene Fallgruppen (DRG), das sowohl ambulante als auch stationäre Leistungen umfasst
Orchestrator	Zentrale Steuerungsplattform von UiPath, die Automatisierungsprozesse verwaltet, überwacht und steuert
Performer	Prozess, der die in der Queue vom Dispatcher bereitgestellten Queue-Items abarbeitet und die eigentlichen Schritte der Automatisierung durchführt
Queue	Warteschlange im Orchestrator, die Aufträge speichert und eine strukturierte sowie priorisierte Abarbeitung durch Performer ermöglicht
Queue-Item	Einzelne Aufträge in der Queue, die jeweils eine Aufgabe oder einen Datensatz repräsentieren
Scope	Bestimmter Bereich des Prozesses in dem Variablen oder Argumente gültig sind
Storage Bucket	Speicherbereich für Dateien und Daten im Orchestrator
Trigger-Table-Dispatcher	Prüft regelmäßig auf neue Daten in der passenden Tabelle und überträgt diese dann in die Queue des Orchestrators als Queue-Items
UBLE	KKH-Webanwendung zum Erfassen und Anzeigen übriger sonstiger Leistungen

1 Einleitung

1.1 Vorwort

Die folgende Projektdokumentation erläutert den Ablauf des IHK-Abschlussprojekts, das die Autorin im Rahmen ihrer Ausbildung zur Fachinformatikerin für Daten- und Prozessanalyse durchgeführt hat. Im Projekt werden Anpassungen am Abrechnungsprozess vorgenommen, um Hybrid-DRGs abzubilden. Diese diagnosebezogenen Fallgruppen umfassen Leistungen, die sowohl ambulant als auch stationär erbracht werden. Bisher waren die Abrechnungen nur für stationäre Leistungen verfügbar, obwohl sie auch ambulant möglich gewesen wären. Ab 2024 wurde jedoch eine einheitliche Vergütung für Vertragsärzte und Krankenhäuser für gleichwertige Leistungen eingeführt.

1.2 Projektumfeld

Der Ausbildungsbetrieb ist die Kaufmännische Krankenkasse - KKH (KKH), eine gesetzliche Ersatzkasse mit Hauptsitz in Hannover. Die KKH wurde am 10. März 1890 in Halle gegründet und beschäftigt heute rund 4.000 Mitarbeitende.[1] Das Geschäftsgebiet umfasst die gesamte Bundesrepublik Deutschland. Mit 106 Geschäftsstellen in verschiedenen Bundesländern und etwa 1,6 Millionen Versicherten zählt die KKH zu den größten und leistungsstärksten Krankenkassen des Landes. Der Auftraggeber des Projekts ist der Fachbereich "Ambulante Versorgung und Abrechnung" der KKH. Dieser Bereich ist für alle Themen rund um die Abrechnung von Leistungen durch Vertragsärzte verantwortlich.

1.3 Projektziel

Das Ziel des Projekts ist es, die bestehende Übergangslösung der KKH so anzupassen, dass die Abrechnung von Hybrid-DRGs auch nach dem 1. Januar 2025 automatisiert verarbeitet werden kann. Hintergrund dieser Anpassung sind Änderungen in der Datenübermittlung, die ab diesem Zeitpunkt unter anderem den Austausch von Fehlernachrichten ermöglichen. In enger Abstimmung mit dem Verband der Ersatzkassen (vdek) wird hierfür ein neuer Datenannahmeprozess implementiert, der die Struktur der Eingangsdaten entsprechend anpasst. Ein zentraler Bestandteil ist die Integration eines Robotic Process Automation (RPA)-gestützten Prozesses, der die Rechnungsstellung und Bezahlung automatisiert und so Verzögerungen und zusätzliche Kosten verhindert. Zudem wird die Meldung des morbiditätsorientierten Risikostrukturausgleichs (MRSA) erweitert, um die Hybrid-DRG-Abrechnungen vollständig und korrekt abzuwickeln.

[1]Vgl. Kaufmännische Krankenkasse - KKH [2024].

1.4 Projektbegründung

Derzeit werden Hybrid-DRG-Abrechnungen über Datenstrukturen verarbeitet, die ursprünglich für andere Zwecke konzipiert wurden. Ein neues, speziell für die Abrechnung von Hybrid-DRGs entwickeltes Datenaustauschverfahren mit passenden Datenstrukturen ist daher erforderlich, um die Abrechnungen effizienter und fehlerfrei maschinell zu verarbeiten. Durch die Optimierung des Prozesses werden Risiken wie Verzugszinsen verringert, während die Automatisierung mittels RPA den Arbeitsaufwand im Fachbereich reduziert und gleichzeitig eine schnelle sowie verlässliche Abwicklung sicherstellt. Ohne die erforderlichen Anpassungen an der RPA-Übergangslösung wäre deren Funktion nicht mehr gewährleistet. Dadurch würden die Aufgaben wieder zurück in die manuelle Sachbearbeitung fallen. Da jedoch die durch die Übergangslösung frei gewordenen Personalkapazitäten inzwischen für andere Aufgaben verplant sind, würden für die manuelle Bearbeitung nicht mehr ausreichend Ressourcen zur Verfügung stehen.

1.5 Projektschnittstellen

Im Rahmen des Gesamtprojekts zur Hybrid-DRG-Abrechnung sind mehrere Schnittstellen definiert, an denen verschiedene Mitarbeiter und Abteilungen beteiligt sind. Das Projekt wird durch eine Projektleitung gesteuert, die die übergreifende Koordination der einzelnen Aufgaben übernimmt. Der Fachbereich spielt eine zentrale Rolle in der Definition der Anforderungen, um sicherzustellen, dass die Lösung den spezifischen Bedürfnissen entspricht und ist zudem für die Abnahme des Gesamtprozesses am Projektende verantwortlich. Externe Entwickler wurden beauftragt, um den neuen Dateneingangsprozess sowie die zugrundeliegende Datenstruktur zu entwerfen und zu implementieren. Diese neue Struktur bildet die Grundlage für die Anpassung des Roboters, der die Abrechnungen automatisiert verarbeitet. Der Quellcode der Übergangslösung für den Roboter wird zentral im GitLab-Repository verwaltet, das als Plattform für die Versionierung und Verwaltung des Codes dient.

1.6 Projektabgrenzung

Aufgrund des umfangreichen Gesamtprojekts zur Hybrid-DRG-Abrechnung ist eine klare Unterteilung in spezifische Teilaufgaben notwendig. Diese Projektdokumentation konzentriert sich ausschließlich auf die Anpassung der bestehenden Übergangslösung an den neu implementierten Dateneingangsprozess und die geänderte Struktur der Eingangsdaten. Andere wesentliche Projektbestandteile wie die Abstimmung mit dem vdek, die vollständige Implementierung des neuen Datenannahmeprozesses, die Konzeption und Umsetzung des maschinellen Rechnungsstellungsprozesses sowie die Erweiterung der MRSA-Meldung werden von anderen Projektmitgliedern übernommen und sind explizit nicht Teil dieser Dokumentation.

2 Projektplanung

2.1 Projektphasen

Für die Umsetzung des Projekts standen insgesamt 40 Stunden zur Verfügung, die auf die verschiedenen Phasen verteilt wurden, um den gesamten Projektablauf strukturiert abzubilden. Eine Übersicht der groben Zeitplanung für die Hauptphasen ist in Tabelle 1 dargestellt. Die detaillierte Aufschlüsselung aller Projektphasen und der zugehörigen Aufgaben findet sich im Anhang A1: Detaillierte Zeitplanung auf Seite i.

Projektphase	Geplante Zeit
1. Analyse & Konzeption	11 h
2. Entwicklung, Implementierung & Test	19 h
3. Dokumentation	10 h
Gesamt	**40 h**

Tabelle 1: Zeitplanung

2.2 Abweichungen vom Projektantrag

Da das Projekt in enger Abstimmung mit dem Projektteam und nicht als Einzelarbeit durchgeführt wurde, konnte die Zeitplanung im Projektantrag nur nach bestem Wissen und ohne feste Garantie festgelegt werden. Im Verlauf der Planung und Umsetzung führten veränderte Einschätzungen der erforderlichen Aufwände zu Anpassungen in der Zeitplanung. Die aktualisierte Zeitplanung ist in Abschnitt 8.1 Soll-/Ist-Vergleich enthalten. Zudem musste im RPA-Prozess bedingt durch die Änderung der Datenstruktur auch die Prüfung der Geldbeträge angepasst werden. Obwohl dies im ursprünglichen Antrag nicht explizit erwähnt wurde, findet dieser Aspekt im weiteren Verlauf der Dokumentation ausführlichere Behandlung.

2.3 Ressourcenplanung

Verwendete Ressourcen sind im Anhang A2: Verwendete Ressourcen auf Seite ii aufgelistet. Zusätzlich zu allen Hard- und Softwareressourcen wurde auch das benötigte Personal aufgenommen. Im Hinblick auf anfallende Kosten lag der Fokus darauf, dass die Nutzung der Software kostenfrei ist oder die Lizenzen dem Unternehmen bereits zur Verfügung stehen. Dadurch ließen sich die Projektkosten minimal halten. Unter anderem kam für die Modellierung unterschiedlicher Diagramme diagrams.net zum Einsatz.

2.4 Entwicklungsprozess

Bevor das Projekt starten konnte, erfolgte die Wahl eines passenden Entwicklungsprozesses, der die Vorgehensweise für die Umsetzung festlegte. Die Entscheidung fiel auf das Wasserfallmodell, da die Anforderungen im Vorfeld genau definiert wurden. Im Wasserfallmodell erfolgt die Abarbeitung der Phasen sequenziell, wobei jede Phase erst vollständig abgeschlossen sein muss, bevor die nächste beginnt. Diese lineare Struktur ermöglicht eine klare Abgrenzung der Projektphasen und reduziert den Bedarf an nachträglichen Anpassungen. Die einzelnen Aufgaben im Gesamtprojekt ließen sich gut voneinander abgrenzen und isoliert testen, wodurch der Abstimmungsaufwand im Projektteam gering blieb. Bei komplexeren Abhängigkeiten wäre ein agiler Ansatz geeigneter gewesen.

3 Analysephase

3.1 Ist-Analyse

Wie zuvor beschrieben, nutzt die bestehende Abrechnungsinfrastruktur der KKH für Hybrid-DRG-Abrechnungen eine veraltete Datenstruktur, die ursprünglich für andere Zwecke entwickelt wurde und die spezifischen Anforderungen dieser Abrechnungen daher nur unzureichend erfüllt. Aufgrund des steigenden Datenvolumens und des neuen Datenaustauschformats ist eine dringende Optimierung erforderlich. Aus diesem Grund wurden die notwendigen Anpassungen der Verarbeitungslogik umfassend analysiert. Diese Analyse erforderte enge Abstimmungen mit dem Fachbereich, um die herausgearbeiteten Anpassungen zu besprechen und daraus resultierende Entscheidungen gemeinsam abzustimmen.

Im Rahmen der Ist-Analyse wurde eine detaillierte Übersicht der aktuell verwendeten Datenfelder erstellt. Mithilfe einer Excel-Tabelle wurden alle relevanten Datenfelder hervorgehoben, um die bestehenden Prozesse besser zu verstehen und zukünftige Anpassungen gezielt planen zu können. Die Tabelle befindet sich im Anhang A3: Ist-Analyse auf Seite iii. Darüber hinaus erfolgte eine intensive Einarbeitung in die derzeitige Übergangslösung, um ihre Struktur und Funktionalität zu analysieren und potenzielle Schwachstellen im Hinblick auf die Hybrid-DRG-Abrechnungen zu identifizieren. Diese fundierte Analyse bildet die Basis für die Entwicklung der optimierten Lösung.

3.2 Wirtschaftlichkeitsanalyse

Wie bereits in Abschnitt 1.2 Projektziel erläutert, ist die Umsetzung des Projekts aufgrund veränderter Datenübermittlung notwendig. In diesem Abschnitt wird zusätzlich untersucht, ob die Realisierung auch aus wirtschaftlicher Perspektive sinnvoll ist.

3.2.1 „Make or Buy“-Entscheidung

Dieses Projekt erforderte die Entwicklung einer Individuallösung, die auf die speziellen Anforderungen und gesetzlichen Vorgaben der Hybrid-DRG-Abrechnung abgestimmt ist. Der Markt für vorgefertigte RPA-Lösungen ist in diesem Bereich begrenzt, da gesetzliche Krankenkassen sehr spezifische Anforderungen haben, die durch Standard- oder Branchensoftware oft nicht vollständig erfüllt werden. Hinzu kam die Anforderung, den Prozess reibungslos in die bestehenden Systeme zu integrieren. Dazu zählt das Kernsystem der KKH, das ebenfalls eine Eigenentwicklung und kein standardisiertes Produkt ist. Aus diesen Gründen ist die Nutzung einer Standardlösung keine Option.

Die Entscheidung für eine Eigenentwicklung ermöglicht der KKH die volle Kontrolle über den Quellcode und künftige Anpassungen. Dies sorgt für langfristige Flexibilität und ermöglicht es, schnell und unabhängig auf Änderungen oder Erweiterungen zu reagieren. Gleichzeitig stärkt die individuell entwickelte Lösung die Wettbewerbsposition der KKH, indem sie spezifische Anforderungen besser umsetzen und sich dadurch von anderen Krankenkassen abheben kann.

3.2.2 Projektkosten

Im Folgenden werden die Kosten für die Umsetzung des Projekts kalkuliert. Dabei werden neben den Personalkosten für die Entwicklerin und andere Projektbeteiligte auch die Kosten für die eingesetzten Ressourcen (siehe Abschnitt 2.3 Ressourcenplanung) berücksichtigt.

Die Berechnung erfolgt auf Basis der von der Personalabteilung vorgegebenen Stundensätze, die sich aus dem Gehalt und den Sozialaufwendungen des Arbeitgebers zusammensetzen. Ein Mitarbeiter der IT-Abteilung ist mit einem Stundensatz von 35,00 € eingeplant. Für Auszubildende liegt der Stundensatz bei 10,00 €, während für Mitarbeiter des Fachbereichs ein Satz von 25,00 € festgelegt wurde. Darüber hinaus ist für die Nutzung der technischen Ressourcen ein pauschaler Betrag von 10,00 € angesetzt. Zusätzlich sind jährliche Betriebs- und Wartungskosten von 1.400,00 € für diese RPA-Automation vorgesehen.

Vorgang	Zeit	Kosten pro Stunde	Kosten
Entwicklungskosten	30h	10,00€ + 10,00€ = 20,00€	600,00€
Erstellung des Lastenhefts	1h	25,00€ + 10,00€ = 35,00€	35,00€
Hilfestellung bei Problemen	5h	35,00€ + 10,00€ = 45,00€	225,00€
Abnahme	2h	60,00€ + 10,00€ = 70,00€	140,00€
			1.000,00€

Tabelle 2: Kostenaufstellung

3.2.3 Amortisationsdauer

Im weiteren Verlauf erfolgt die Berechnung der Amortisationsdauer, um die Wirtschaftlichkeit des Projekts zu bewerten. Dabei steht die im Abschnitt 3.2.2 Projektkosten berechnete Gesamtsumme den erzielten Kosteneinsparungen gegenüber. Der Break-Even-Point markiert den Zeitpunkt, an dem die gesamten Investitionskosten durch die erzielten Einsparungen vollständig gedeckt sind. Dieser Punkt bestimmt die Amortisationszeit des Projekts.

Die Automatisierung des Abrechnungsprozesses reduziert den manuellen Aufwand im Fachbereich erheblich. Insbesondere bei der Bearbeitung der Abrechnungen, die nun nicht mehr manuell durchgeführt werden müssen.

Vorgang	Anzahl pro Jahr	Alte Zeit pro Vorgang	Neue Zeit pro Vorgang	Einsparung pro Jahr
Bearbeitung der Rechnung	2500	5 min	0,5 min	11.250 min
				11.250 min

Tabelle 3: Zeiteinsparung

Zur Berechnung der Kosteneinsparungen wird die erzielte Zeiteinsparung im Fachbereich zugrunde gelegt. Bei einer jährlichen Zeitersparnis von 11.250 Minuten ergibt sich eine Kosteneinsparung von:

$$\frac{11.250\ Min/Jahr}{60} \times (25{,}00 + 10{,}00)\ €/h \approx 6.562{,}50\ €/Jahr$$

Unter Berücksichtigung der jährlichen Betriebs- und Wartungskosten von 1.400,00 €, die im Abschnitt 3.2.2 Projektkosten erwähnt wurden, ergibt sich die Amortisationsdauer wie folgt:

$$\frac{1.000{,}00€}{(6.562{,}50 - 1.400{,}00)\ €/Jahr} \approx 0{,}19\ Jahre$$

Somit würde sich das Projekt innerhalb von rund 0,19 Jahren (also etwa 2,28 Monaten) amortisieren. Da die Abrechnungen auch in den kommenden Jahren weiterhin durchgeführt werden müssen, zeigt diese Berechnung, dass sich das Projekt aus wirtschaftlicher Sicht schnell rentiert. Die genaue Darstellung des Amortisationsverlaufs findet sich in der Grafik im Anhang A4: Amortisation auf Seite iv.

3.3 Nicht-monetäre Aspekte

Wie schon in Abschnitt 1.2 Projektziel beschrieben, ist die veränderte Datenübermittlung für die Durchführung der automatisierten Abrechnungen zu Hybrid-DRG klar definiert. Diese

Änderungen machen die Umsetzung des Projekts unumgänglich, sodass der wirtschaftliche Aspekt nur eine untergeordnete Rolle spielt.

Ein weiterer Vorteil des Projekts ist die Verringerung der Abhängigkeit von individuellem Fachwissen. Die Automatisierung des Abrechnungsprozesses reduziert die Notwendigkeit, dass umfassende Kenntnisse aller manuellen Arbeitsschritte ausschließlich im Fachbereich vorhanden sein müssen. Darüber hinaus können die Mitarbeiter durch die gewonnene Zeitersparnis effizienter eingesetzt werden und sich auf andere Aufgaben konzentrieren.

3.4 Anwendungsfälle

Während der Analysephase wurde ein Unified Modeling Language (UML)-Use-Case-Diagramm erstellt, um einen Überblick über die fachlichen Anwendungsfälle zu gewinnen und die Logik der Abläufe besser zu verstehen. Dieses Diagramm zeigt alle wesentlichen Funktionen, die durch den RPA-Prozess automatisiert werden. Eine detaillierte Ansicht des Use-Case-Diagramms befindet sich im Anhang A5: Use-Case-Diagramm auf Seite v. Das Diagramm dient dazu, die verschiedenen Anwendungsbereiche klar voneinander abzugrenzen und schafft somit die Grundlage für die schrittweise Umsetzung der Änderungen in der Implementierungsphase.

3.5 Lastenheft

Am Ende der Analysephase wurde in Zusammenarbeit mit dem Fachbereich ein Lastenheft erstellt, das sämtliche fachlichen Anforderungen des Auftraggebers an den zukünftigen RPA-Prozess enthält. Das vollständige Lastenheft ist im Anhang A6: Lastenheft auf Seite v zu finden. Auf dieser Grundlage wurde anschließend das Pflichtenheft erstellt, welches im Abschnitt 4.7 Pflichtenheft näher beschrieben wird.

4 Entwurfsphase

4.1 Zielplattform

Es handelt sich bei dem Projekt um die Entwicklung eines automatisierten Abrechnungsprozesses zur Verarbeitung von Hybrid-DRG-Daten mithilfe der RPA-Plattform UiPath. Die UiPath-Produkte werden auf einem virtuellen Windows Server betrieben, wobei der UiPath Orchestrator zur zentralen Verwaltung und Überwachung der robotergesteuerten Abläufe eingesetzt wird, um eine skalierbare und zuverlässige Prozesssteuerung sicherzustellen. Zur effizienten Speicherung und Abfrage relevanter Daten dient ein Microsoft SQL Server, der bereits in der bestehenden Infrastruktur etabliert ist. Die Datenbank läuft in einem ausfallsicheren Cluster, der einfache Redundanz bietet, sodass der Orchestrator bei Serverausfall eines Datenbankservers ohne Unterbrechung auf den zweiten Server umschaltet.

Die Entwicklung erfolgt überwiegend in einem Low-Code-Ansatz durch die Verwendung vorgefertigter Aktivitäten auf einer grafischen Benutzeroberfläche. Die Plattform bietet dabei im Kern eine Integration von C#. Dies ermöglicht sowohl komplexere Logiken als auch die direkte Einbindung von C#-Code, wodurch die Integration von Validierungs- und Geschäftslogiken flexibel und effizient gestaltet wird. Zudem ist das Kernsystem direkt an diesen Prozess angebunden und greift auf Webanwendungen im Browser Microsoft Edge zu, während eine Db2-Datenbank als zentrale Datenquelle dient.

4.2 Architekturdesign

Die Architektur der angepassten RPA-Lösung für die Hybrid-DRG-Abrechnung ist auf eine besonders stabile und leistungsfähige Systemstruktur ausgelegt. Zentral dabei ist die SQL-Server-Datenbank, die speziell auf die Anforderungen der Hybrid-DRG-Abrechnung angepasst wurde und somit eine schnelle sowie zuverlässige Datenverarbeitung sicherstellt. Die Steuerung und Verwaltung der Prozesse erfolgt über den UiPath Orchestrator, der eine Echtzeitüberwachung ermöglicht und bei Bedarf manuelle Eingriffe zulässt. Eine Übersicht der Überwachung im Orchestrator befindet sich im Anhang A7: Echtzeitüberwachung auf Seite vi. Die Prozesslogik ist gezielt in C# implementiert und deckt die spezifischen Automatisierungsanforderungen passgenau ab. Nach der Dokumentation in UBLE und der Validierung der Abrechnungsdaten in ADA werden diese an das SAP-System übergeben, das die endgültige Zahlung des Rechnungsbetrags abwickelt und so den automatisierten Abrechnungsprozess vollständig integriert.

Der Orchestrator übernimmt die Koordination der Datenerfassung, die Validierung und die Verarbeitung der Abrechnungen, wodurch eine exakte und effiziente Abwicklung gewährleistet wird. Diese Architektur ist nicht nur flexibel und erweiterbar, sondern schafft auch eine solide Grundlage für künftige Anpassungen und die Integration zusätzlicher Datenquellen. Dank der Verarbeitung der Abrechnungen als separate Transaktionen lassen sich einzelne Aufträge gezielt erneut abarbeiten, falls es zu Fehlern kommen sollte. Dadurch werden die Effizienz und die Fehlerbehebung zusätzlich gesteigert. Die Architektur ist im Anhang A8: Architektur auf Seite vii abgebildet.

4.3 Datenmodell

Auf Grundlage der erarbeiteten Anwendungsfälle und Anforderungen im Lastenheft wurde ein Entity-Relationship-Modell (ERM) entwickelt, welches im Anhang A9: Entity-Relationship-Model auf Seite viii zu finden ist. Es stellt die relevanten Entitätstypen und ihre Beziehungen für die automatisierte Verarbeitung von Hybrid-DRG-Daten dar. Die zentralen Entitätstypen umfassen

„Person“, „Rechnung“, „Hybrid-DRG“ und „Fehlerprotokoll“. Der Entitätstyp „Person“ enthält Attribute wie „Personen-ID“, „Geburtsdatum“, „Versicherungsstatus“ und „Name“ als wesentliche Stammdaten für den Abrechnungsprozess. Über die Beziehung „hat" ist „Person“ mit „Rechnung“ verbunden, wodurch ersichtlich ist, welche Rechnungen einer Person zugeordnet sind.

„Rechnung“ ist das Herzstück des Abrechnungsprozesses und beinhaltet Attribute wie „Rechnungs-ID“, „Betrag“, „Datum“, „Stornierungsstatus“ und „Personen-ID“. Über die Beziehung "umfasst" ist „Rechnung“ mit „Hybrid-DRG“ verbunden, sodass jede Rechnung eine Hybrid-DRG-Position enthalten kann. Der Entitätstyp „Hybrid-DRG“ repräsentiert die Leistungsarten und umfasst Attribute wie „Hybrid-DRG-ID“ und „Anzahl“, wodurch die Zuordnung und Quantifizierung spezifischer Leistungen innerhalb einer Rechnung ermöglicht wird.

Für die Dokumentation und Analyse von Fehlern im Prozess gibt es den Entitätstyp „Fehlerprotokoll“, der mit „Rechnung“ über die Beziehung "wird dokumentiert in" verknüpft ist und das Attribut „Fehlercode“ zur Erfassung von Abweichungen umfasst. Das ERM unterstützt durch die klare Struktur der Entitätstypen und Beziehungen die automatisierte Verarbeitung und Fehlerbehandlung im RPA-Prozess.

4.4 Geschäftslogik

Die Abrechnung der Hybrid-DRG-Daten wird regelmäßig durch den Fachbereich durchgeführt und ist Grundlage für die gesetzlich vorgeschriebene Abrechnung sowie für die RPA-Lösung. Um den Ablauf besser darzustellen, wurde ein Diagramm nach Business Process Model and Notation (BPMN) erstellt, das im Anhang A10: BPMN-Diagramm auf Seite ix zu finden ist.

Zu Beginn überprüft der fachliche Dispatcher regelmäßig, ob neue Abrechnungsdaten in der Datenbank vorhanden sind. Sobald neue Daten identifiziert wurden, übernimmt der Roboter das Einlesen und Validieren der Datensätze. Die Validierung umfasst unter anderem die Prüfung der Personen-ID, der Gültigkeit einer Krankenversicherung sowie weiterer Abrechnungsparameter. Nach erfolgreicher Validierung wird die Zahlung in UBLE dokumentiert und im Anschluss in die Anwendung ADA übertragen, um die Zahlung durchzuführen. ADA gibt die Informationen danach an das SAP-System der KKH weiter, welches die Überweisung des Rechnungsbetrags durchführt. Zuletzt erfolgt eine Zusammenstellung der Abrechnungen, die dem Fachbereich in einem strukturierten Report zur Verfügung steht.

4.5 Maßnahmen zur Qualitätssicherung

Zur Qualitätssicherung im Projekt kommen mehrere gezielte Maßnahmen zur Anwendung, um die Zuverlässigkeit und Stabilität der RPA-Lösung sicherzustellen. Der UiPath Analyzer wird als

statisches Analyse-Tool eingesetzt und prüft Automatisierungsprozesse auf Übereinstimmung mit definierten Best Practices und unternehmensspezifischen Standards. Dieser Analyzer untersucht u. a. Variablenbenennungen, Scope-Definitionen, ungenutzten Code und Performance-Aspekte, um potenzielle Schwachstellen frühzeitig zu identifizieren und zu beheben. Durch diese strukturierte Prüfung lassen sich Fehlerquellen schon vor der Testphase erkennen, wodurch ein robuster Entwicklungsstandard erhalten bleibt. Zusätzlich führen erfahrene Entwickler regelmäßige Code-Reviews durch, um sicherzustellen, dass die Automation den Standards entspricht und sowohl funktionale als auch betriebliche Anforderungen erfüllt. Weiterhin dienen Komponententests der Qualitätssicherung der einzelnen Prozessbausteine.

Tägliche Rücksprachen bieten die Möglichkeit, den aktuellen Entwicklungsstand zu besprechen, offene Fragen zu klären und Herausforderungen gemeinsam anzugehen. Die Ergebnisse dieser Meetings fließen direkt in den Entwicklungsprozess ein, wodurch die Funktionalität der Anwendung und die Code-Qualität kontinuierlich verbessert wird. Darüber hinaus nutzen die enge Zusammenarbeit und der aktive Wissensaustausch im Team dem Fachwissen aller Entwickler. Dies ermöglicht eine schnelle Problemlösung, reduziert Verzögerungen und schafft eine solide Grundlage für die kontinuierliche Weiterentwicklung des Projekts.

4.6 Deployment

Für das Projekt erfolgt die Entwicklung der Automatisierungslösung mithilfe von UiPath. Dabei übernimmt der UiPath Orchestrator die Steuerung und Verwaltung der Roboter. Die Entwicklungsumgebung besteht aus UiPath Studio, unterstützt durch GitLab zur Versionskontrolle. Der Source Code der Automatisierung wird regelmäßig in das GitLab-Repository hochgeladen, wodurch eine kontinuierliche und strukturierte Weiterentwicklung gewährleistet bleibt. Zusätzlich kommt ein Jenkins-Build-Server zum Einsatz, um den Build-Prozess für die Automatisierungssoftware zu automatisieren und die verschiedenen Versionen in den Bereitstellungsprozess zu integrieren. Nach erfolgreichem Build-Prozess orchestriert XL-Release die Bereitstellungsschritte, um einen reibungslosen und planbaren Deployment-Prozess sicherzustellen.

Der UiPath Orchestrator spielt eine zentrale Rolle im Deployment-Prozess: Er empfängt die neuesten Prozessversionen und koordiniert das Deployment auf die vorgesehenen Roboter. Er verwaltet zudem die robotergestützten Prozesse über ein rollenbasiertes Benutzer- und Zugriffsmanagement und erstellt Log-Dateien zur Nachverfolgbarkeit und Wartbarkeit. Eine Darstellung des Deployment-Prozesses ist im Anhang A11: Deployment auf Seite x.

4.7 Pflichtenheft

Zum Abschluss der Entwurfsphase wurde das Pflichtenheft erstellt. Grundlage für das Pflichtenheft waren die Anforderungen, die in enger Zusammenarbeit mit dem Fachbereich formuliert und im Abschnitt 3.5 Lastenheft dokumentiert wurden. Das Pflichtenheft dient als Prüfgrundlage, um final zu betrachten, ob alle Anforderungen erfolgreich umgesetzt wurden und das Projektziel erreicht wurde. Eine vollständige Fassung des Pflichtenhefts befindet sich im Anhang A12: Pflichtenheft auf Seite xi.

5 Implementierungsphase

5.1 Implementierung der Anpassungen im Hauptworkflow

Basierend auf den Ergebnissen der Ist-Analyse und der Einarbeitung in den Code der Übergangslösung sowie den erstellten Diagrammen wurden die notwendigen Anpassungen an die neue Datenstruktur und den neuen Dateneingang während der Implementierungsphase umgesetzt. Es ist wichtig zu wissen, dass die KKH bei der Automation mit UiPath nach dem Dispatcher-Performer-Prinzip arbeitet. Der Dispatcher erstellt die Aufträge, die im Orchestrator als Queue-Items angelegt werden. Eine Darstellung der Queue des Orchestrators ist im Anhang A13: Orchestrator-Queue auf Seite xii zu sehen. Anschließend holt der Performer diese Aufträge aus der Warteschlange und verarbeitet sie in seinem Workflow.

Im ersten Schritt erfolgte die Anpassung des Performers. Dazu wurden in UiPath Studio im Workflow zur Ermittlung der Verarbeitungsdaten die betroffenen Felder in der Structured Query Language (SQL)-Abfrage geändert oder entfernt, um der neuen Struktur gerecht zu werden. Diese Änderungen zogen sich durch den gesamten Workflow, weshalb auch in den nachfolgenden Aktivitäten des Workflows alle zugehörigen Variablen angepasst wurden. Variablen und Argumente, die nicht mehr benötigt wurden, entfielen, um die Übersichtlichkeit zu wahren. Abbildungen der Variablen und Argumente sind im Anhang A14: Variablen Ermittlung Verarbeitungsdaten und A15: Argumente Ermittlung Verarbeitungsdaten auf den Seiten xii und xiii zu sehen. Die Logik der Verarbeitungsvariablen blieb grundsätzlich gleich. Eine Ausnahme bildet die Variable zur Stornierung, die von einer numerischen Prüfung auf eine Boolean-Prüfung umgestellt wurde, um festzustellen, ob die relevanten Felder zur Stornierung einen Wert enthalten.

Anschließend erfolgte im Cleanup-Workflow, der das Trimmen der Werte übernimmt, die Anpassung der Variablennamen. Zusätzlich musste das Quartal anhand des Rechnungsdatums ermittelt und in einer Variable gespeichert werden, da die alte Datenstruktur hierfür eine separate Spalte enthielt.

Für den Test zur Ermittlung der Verarbeitungsdaten wurden die entsprechenden Variablen auf die neuen Felder abgestimmt und eine zusätzliche Log Message-Aktivität eingefügt, um die Werte im JavaScript Object Notation (JSON)-Format auszugeben. Daraufhin wurden auch im ProcessItem-Workflow sämtliche Variablen an die neuen Bedingungen angepasst. Dieser stellt den Ablauf des Roboters als Flussdiagramm dar und beinhaltet die Aktivitäten zur fachlichen Verarbeitung (siehe Anhang A16: Logik Hauptprozess auf Seite xiv). Die neuen Variablen sind im Anhang A17: Variablen Hauptprozess auf Seite xvi zu finden. Zudem wurde auch dort die Stornierungskomponente so umgestellt, dass zur Prüfung nun ein Boolean-Wert statt eines Zahlenwertes verwendet wird.

5.2 Implementierung der Anpassungen in der Preisprüfung

Im Workflow zur Überprüfung, ob der Preis zur abzurechnenden Leistungsart und der Anzahl der Hybrid-DRGs passt, erfolgte eine Vereinfachung der alten Logik (siehe Anhang A18: Ursprüngliche Preisprüfung auf Seite xvii). Früher ermittelten die vier Primärschlüssel aus den Eingangsdaten des Auftrags die zugehörigen Hybrid-DRGs. Diese prüften zeilenweise, ob der Preis zur jeweiligen Leistungsart im Abrechnungszeitraum korrekt war. Jetzt prüft eine If-Bedingung direkt, ob die Anzahl der Hybrid-DRGs gleich 1 ist. Trifft dies zu, erfolgt wie zuvor die Validierung des Preises und das Ergebnis wird als Boolean gespeichert; andernfalls erfolgt die Speicherung von false. Das Ergebnis und die Anzahl der Hybrid-DRGs werden anschließend als Log Message ausgegeben. Die aktuelle Prüfung ist ebenfalls im Anhang A19: Aktuelle Preisprüfung auf Seite xix dokumentiert. Zum Vergleich des Preises mit der abzurechnenden Leistungsart hat der Fachbereich eine Excel-Tabelle mit den passenden Werten bereitgestellt. Diese Tabelle liegt im Orchestrator in einem Storage Bucket, auf den der Prozess zur Prüfung zugreift. Eine Abbildung des Storage Buckets ist im Anhang A20: Storage Bucket auf Seite xx zu sehen.

5.3 Implementierung der Anpassungen in den Dispatchern

Im Gesamtprozess übernimmt ein speziell für Hybrid-DRG entwickelter fachlicher Dispatcher die regelmäßige Prüfung auf neue Abrechnungsaufträge. Sobald neue Aufträge bekannt sind, prüft das System diese innerhalb der Db2-Datenbank des Kernsystems und überträgt sie in eine standardisierte Db2-Tabelle. Anschließend kommt das Dispatcher-Performer-Prinzip in UiPath zum Einsatz. Über den Trigger-Table-Dispatcher als zentralen Kanal werden die Aufträge aus der Db2-Tabelle in den Orchestrator überführt und als Queue-Items in der Warteschlange angelegt. Im fachlichen Dispatcher-Prozess (siehe Anhang A21: Logik Hybrid-DRG-Dispatcher auf Seite xx) wurden hierfür die neuen Primärschlüssel angepasst und veraltete SQL-Bestandteile, die auf nicht mehr relevante Hybrid-DRG-Werte verwiesen, entfernt. Auch der im

Trigger-Table-Dispatcher bereits vorhandene Subprozess für Hybrid-DRG wurde entsprechend an die neuen Primärschlüssel angepasst, um eine reibungslose Verarbeitung sicherzustellen. Dieser Subprozess ist im Anhang A22: Logik Trigger-Table-Dispatcher auf Seite xxi abgebildet.

5.4 Testen der Automation

Für die Tests wurden vom Fachbereich vorgegebene Testdaten manuell mit DBeaver in die Datenbank geschrieben. Diese Daten ermöglichten die Durchführung von Komponententests im Hauptprozess für den Workflow zur Ermittlung der Verarbeitungsdaten und die Prüfung, ob der Preis zur jeweiligen Leistungsart passt. Beide Workflows funktionierten auf Anhieb fehlerfrei. Die beiden Tests sind in den Anhängen A23: Test Ermittlung Verarbeitungsdaten und A24: Test Preisprüfung auf den Seiten xxii und xxiii zu sehen.

Beim anschließenden Test des fachlichen Dispatcher-Prozesses trat zunächst ein Fehler aufgrund nicht vorhandener Klammern auf. Des Weiteren erforderte das SQL zum Einfügen der Daten in die Tabelle eine WHERE-Bedingung, was zu einem Fehler führte. Da im SQL-Befehl eine WHERE-Klausel notwendig war, wurde WHERE 1=1 eingefügt. Danach lief der Prozess ohne Probleme und erzeugte die erwarteten fünf Aufträge zu den Testdaten.

Der Test des Trigger-Table-Dispatchers funktionierte ebenfalls einwandfrei. Anschließend erfolgte die Überprüfung der erzeugten Queue-Items im zum Projekt passenden Ordner im UiPath Orchestrator. Bevor der Roboter getestet werden konnte, mussten ihm erst die nötigen Schreibrechte für die KKH-Webanwendungen UBLE und ADA gegeben werden. Die Einträge des Roboters sind in den Anhängen A25: Erfassung der Leistungsart in UBLE, A26: Auftrag in ADA öffnen und A27: Befüllen der ADA-Bearbeitungsmaske auf den Seiten xxiv und xxv detailliert beschrieben. Nach dieser Vorbereitung konnte der Hybrid-DRG-Prozess, also der Performer, vollständig getestet werden und lief ohne Fehler.

Zum Abschluss wurden alle Änderungen in den drei Prozessen über Sourcetree committet (siehe Anhang A28: Commit in Sourcetree auf Seite xxv), sodass sie nun in GitLab für alle RPA-Entwickler sichtbar sind und eine Qualitätssicherung durch einen weiteren Entwickler durchgeführt werden kann.

6 Abnahmephase

Die Abnahme dieses Teilprojekts erfolgte in zwei Schritten. Zunächst wurden die abgeschlossene Analyse und Konzeption gemeinsam mit dem Auftraggeber abgestimmt. Anschließend folgte die finale Abnahme nach der Implementierung und Testphase durch den Fachbereich und einen Entwickler. So konnte der gesamte Entwicklungsprozess kontinuierlich durch interne Freigaben begleitet und gesichert werden.

7 Dokumentation

Da die RPA-Entwickler sowohl die Ausführung als auch die Überwachung des Prozesses eigenständig übernehmen, ist keine zusätzliche Dokumentation erforderlich. Die Protokollierung über Log-Dateien stellt alle notwendigen Informationen bereit, welche detaillierte Einblicke in die Prozessabläufe und potenzielle Fehlerquellen ermöglichen. Ergänzend dazu fungiert das GitLab-Repository als zentrale Plattform zur Speicherung und Verwaltung des gesamten Quellcodes sowie zur Nachverfolgung von Änderungen und Versionen. Durch die Kombination von Log-Dateien und Repository wird sichergestellt, dass alle relevanten Daten zur Prozessverfolgung und -weiterentwicklung dokumentiert sind. Für diesen RPA-Prozess wurde daher lediglich die für das Abschlussprojekt benötigte Projektdokumentation erstellt.

8 Fazit

Abschließend lässt sich festhalten, dass das Projekt eine große Entlastung für den Fachbereich darstellt. Die Automatisierung reduziert Fehler, wodurch die gewonnene Zeit für andere Aufgaben nutzbar ist. Darüber hinaus steigert der wirtschaftliche Nutzen des Projekts dessen Attraktivität und unterstützt seine erfolgreiche Umsetzung.

8.1 Soll-/Ist-Vergleich

Die Analyse- und Konzeptionsphase des Projekts ließ sich durch eine strukturierte und effiziente Planung um zwei Stunden verkürzen. So konnten die Anforderungen schnell erfasst und klar definiert werden. Im Gegensatz dazu benötigten die Entwicklungs-, Implementierungs- und Testphasen zwei Stunden mehr als ursprünglich geplant. Diese zusätzliche Zeit war notwendig, um auf die komplexeren Anforderungen und Anpassungen im Prozess einzugehen. Auf diese Weise konnte sichergestellt werden, dass alle funktionalen Anforderungen vollständig umgesetzt wurden. Die gewonnenen Stunden kamen der Implementierungsqualität zugute und ermöglichten umfangreiche Tests, wodurch ein fehlerfreier Betrieb des Roboters gesichert ist.

Projektphase	Geplant	Tatsächlich	Differenz
1. Analyse & Konzeption	11 h	9 h	-2 h
2. Entwicklung, Implementierung & Test	19 h	21 h	+2 h
3. Dokumentation	10 h	10 h	0 h
Gesamt	**40 h**	**40 h**	

Tabelle 4: Soll-/Ist-Vergleich

8.2 Lessons Learned

Die Umsetzung dieses Abschlussprojekts bot der Autorin wertvolle Erfahrungen in allen Phasen der Projektarbeit, von der Analyse über die Entwurfs- bis zur Implementierungsphase. Besonders bereichernd war die enge Zusammenarbeit und der stetige Austausch mit dem Fachbereich, der eine tiefe Einarbeitung in die Anforderungen und Herausforderungen der Hybrid-DRG-Automatisierung ermöglicht hat. Die dabei gesammelten Erfahrungen im Bereich Prozessautomatisierung, insbesondere im Zusammenhang mit der Nutzung des Dispatcher-Performer-Prinzips, sind für zukünftige Projekte von großem Nutzen. Durch die Analyse des bestehenden Gesamtprozesses und die Konzeption der notwendigen Anpassungen konnte das Prozessverständnis entscheidend vertieft werden. Auch technologische Kompetenzen, wie die Anwendung von C# zur Anpassung und Erweiterung des Codes, die Zusammenarbeit in GitLab zur Versionierung und Dokumentation sowie die Nutzung von SQL-Servern für die Datenverarbeitung wurden weiter vertieft und gezielt erweitert.

8.3 Ausblick

Obwohl alle Anforderungen des Projekts erfolgreich umgesetzt wurden, besteht die Möglichkeit für zukünftige Erweiterungen. Der kontinuierliche Austausch zwischen dem Fachbereich und den Entwicklern gewährleistet, dass Anpassungen zeitnah identifiziert und umgesetzt werden können. So könnte beispielsweise das Fehlerprotokoll um zusätzliche Meldungen ergänzt werden, um die Fehlerdiagnose zu optimieren. Zudem sind weitere Funktionen zur Verbesserung der effizienten Ressourcennutzung der Automation denkbar, die im Einklang mit den sich ändernden Anforderungen des Fachbereichs stehen. Zusätzlich können im produktiven Betrieb durch die hohe Fallzahl häufig Situationen auftreten, die im Vorfeld nicht berücksichtigt werden konnten. In solchen Fällen lassen sich die Prüfungen nachträglich erweitern und an die neuen Gegebenheiten anpassen.

9 Literaturverzeichnis

Kaufmännische Krankenkasse - KKH. *Karriere - Mitarbeiteranzahl der KKH.* 2024. https://www.kkh.de/karriere (Zugriff am 04. November 2024).

10 Anhang

A1. Detaillierte Zeitplanung

Analyse & Konzeption	**11 h**
1. Durchführung der Ist-Analyse	1 h
2. Durchführung der Wirtschaftlichkeitsanalyse inkl. Amortisationsrechnung	2 h
3. Überblick Anwendungsfälle inkl. Erstellung eines Use-Case-Diagramms	1 h
4. Unterstützung des Fachbereichs bei Erstellung des Lastenhefts	2 h
5. Verstehen der Datenbankstruktur inkl. Erstellung eines ERMs	1 h
6. Erstellen eines BPMN-Diagramms	1 h
7. Erstellung des Pflichtenhefts	2 h
8. Abnahme durch Auftraggeber	1 h
Entwicklung, Implementierung & Test	**19 h**
1. Neuimplementierung der Datenermittlung	6 h
2. Anpassung der Preisprüfung	4 h
3. Anpassung der Dispatcher	4 h
4. Komponententests der einzelnen Workflows	2 h
5. Gesamttest der RPA-Automation	2 h
6. Abnahme durch Auftraggeber	1 h
Dokumentation	**10 h**
1. Erstellung der Projektdokumentation	10 h
Gesamt	**40 h**

Tabelle 5: Detaillierte Zeitplanung

A2. Verwendete Ressourcen

Hardware

- Büroarbeitsplatz mit Laptop

Software

- UiPath Studio – Entwicklungsumgebung
- UiPath Orchestrator – Management-Tool zur Überwachung, Steuerung und Verwaltung von RPA-Prozessen und -Ressourcen
- GitLab – Versionsverwaltung
- Sourcetree – Grafische Benutzeroberfläche für Git
- Windows 10 Pro – Betriebssystem
- diagrams.net – Programm zur Erstellung von Diagrammen und Modellen
- Microsoft Visio – Programm zur Erstellung von Diagrammen und Modellen
- Microsoft Excel – Tabellenkalkulationsprogramm
- DBeaver – Datenbank-Management-Tool
- Virtueller Windows Server – Ausführung der UiPath-Produkte
- Microsoft SQL Server – Relationales Datenbankmanagementsystem zur Speicherung und Verwaltung der Daten
- Jenkins-Build-Server – Automatisierung von Software-Builds und Tests
- XL-Release – Release-Management-Tool zur Automation und Koordination vom Softwarebereitstellungsprozess

Personal

- Auszubildende Fachinformatikerin für Daten- und Prozessanalyse – Umsetzung des Projekts
- Anwendungsentwickler – Hilfestellung bei Problemen und Abnahme
- Mitarbeiter des Fachbereichs – Festlegung der Anforderungen und Abnahme

A3. Ist-Analyse

Column Name	Data Type	Description		Column Name	Data Type	Description
FLLBKV	CHAR	Bereich Kassenärztl. Vereinig		FGEDBAZR	DATE	Letzter Tag Abrechnungszeitr.
FBBDARJ	SMALLINT	Abrechnungsjahr		FGEKESR	SMALLINT	Kn. Einzel-/Sammelrechnung
FLSABQU	SMALLINT	Abrechnungsquartal		FLLIKRST	CHAR	IK Rechnungssteller
FBBLNRDS	INTEGER	Lfd. Nr. Datensatz		FBHRNR	CHAR	Rechnungsnummer
FKRKSEFN	SMALLINT	Storno-Kn. EFN DA BAAV		FGEDRECH	DATE	Rechnungsdatum DA
FGEKVHV	CHAR	Verarbeitungskennzeichen DA		FGEKZEMP	CHAR	IK Zahlungsempfänger DA
FGELNRGV	CHAR	Lfd. Nr. des Geschäftsvorf. D		FGENRKOR	SMALLINT	Korrekturzähler Abrechnungsf.
FGEIKSEN	CHAR	IK Sender Daten DA		FLLKFNLV	CHAR	Kn. FHL vorhanden
FGEIKEMP	CHAR	IK Empfänger Daten DA		FEUNRRB	SMALLINT	Lfd. Nr. Regreßbearbeitung
FGEIDSR	DECIMAL	Sammelrechnungs-ID DA		FGOIKDAN	CHAR	IK Datenannahmestelle
FLVLLANR	CHAR	Lebenslange Arztnummer		FGEKVBZA	CHAR	Vertragsbereich Zahnarzt
FLLBSTNR	CHAR	Arzt-Betriebsstättennummer		FLZZANRV	CHAR	Zahnarztnummer VPS
FLLIKLE	CHAR	Institutionskn. (IK) LE		FGEKVS1N	SMALLINT	Kn. Vers.status DA 1.St. num
FLLBSTNU	CHAR	Arzt-Betriebsstättennr. über		FVEKBPER	CHAR	Kn. besondere Personengruppe
FAZLANR	CHAR	Lebenslange Arztnr. (LANR) 1-7		FVEKBPE2	CHAR	Kn. besondere Personengruppe 2
FAZGRSCL	CHAR	Arztgruppenschlüssel		FVEKDMP1	SMALLINT	DMP-Kennzeichen
FLKKUBVG	CHAR	Unfallkennzeichen/BVG		FLKKVDAT	CHAR	Verarbeitungskennzeichen Datei
FGEKIASN	CHAR	Art Inanspruchnahme DA		FTMDVER	DATE	Verarbeitungsdatum
FLLKVER	CHAR	Vertragskennzeichen DA		FLLKBARE	CHAR	Kn. Batchverarbeitung Regress
FLTVNRLE	CHAR	VNR Leistungsbezieher		FLLDERZ	DATE	Datum Erzeugung
FGEIKKK	CHAR	IK Krankenkasse DA		FPEID	CHAR	Benutzer-ID
FLLNAME	CHAR	Name Versicherter		FGEDSEFN	DATE	Storno-Datum EFN DA
FLLVNAME	CHAR	Vorname Versicherter		FLLNRVTG	CHAR	Vertragsnummer §293a SGB V
FPNDGEB	INTEGER	Geburtsdatum		FTKABRNR	CHAR	Abrechnungsnummer eZahn
FPNPID	INTEGER	Personen-ID (PID)		FTKZZANR	CHAR	zentrale Zahnarztnr. Worldline
FLLKPIDZ	CHAR	Kn. PID-Zuordnung		FGELNRAR	INTEGER	Lfd. Nr. Abrechnungsinfo DA
FGEKGES	SMALLINT	Kn. Geschlecht DA		FGKUWANR	CHAR	Arztnummer (überweisend)
FANPLZZ	CHAR	Postleitzahl Anschrift		FLLIKLE	CHAR	Institutionskn. (IK) LE
FLMKLADA	CHAR	Länderkennzeichen DA		FGENRGEB	CHAR	Gebührennummer DA
FLLTNID	CHAR	Teilnehmer-ID laut Vertrag		FGEIDGEB	CHAR	Gebührennummer-ID DA
FLTGBKMJ	SMALLINT	KVK gültig bis Jahr/Monat		FGETARBG	CHAR	Abrechnungsbegründung
FGEKVSHZ	CHAR	Kn. Versichertenstatus DA		FGEANGEB	INTEGER	Anzahl Abrechnung. Gebührennr
FGEKZUST	SMALLINT	Kn. Zuzahlungsstatus		FGEWEGEB	DECIMAL	Wert Gebührennummer
FGEBGAGN	DECIMAL	Gesamtbetr. abger. Geb.nummern		FLLPZG	DECIMAL	Punktzahl Gesamt
FGEBGADK	DECIMAL	Gesamtbetr. abger. Dialysekost		FLLDIASK	DECIMAL	Dialysesachkosten
FGEBGASK	DECIMAL	Gesamtbetr. abger. Sachkosten		FGEBSACH	DECIMAL	Sachkosten DA
FGEBGGGZ	DECIMAL	Gesamtbetr. gel. ges. Zuzahl.		FGETSACH	CHAR	Bezeichnung Sachkosten DA
FGEBGGVZ	DECIMAL	Gesamtbetr. gel. vertr. Zuzah.		FGEDLEIS	DATE	Datum Leistungserbringung DA
FGEBGMIN	DECIMAL	Gesamtbetrag Minderungsbeträg		FGEULEIS	INTEGER	Uhrzeit Leistungserbringung DA
FGEBGGNN	DECIMAL	Gesamtbetrag Gebührennr. nett		FGEDLTL	DATE	Letzter Tag Leistungserbr. DA
FKDISOWK	CHAR	Währungskürzel		FGEIDRG	CHAR	Info zur DRG Gebührennummer D
FGEDAAZR	DATE	1. Tag Abrechnungszeitraum				

Wird in neuer Datenstruktur durch andere Werte/Prüfungen ersetzt

Wird in neuer Datenstruktur nicht mehr benötigt

Wird in neuer Datenstruktur übernommen

Abbildung 1: Ist-Analyse

A4. Amortisation

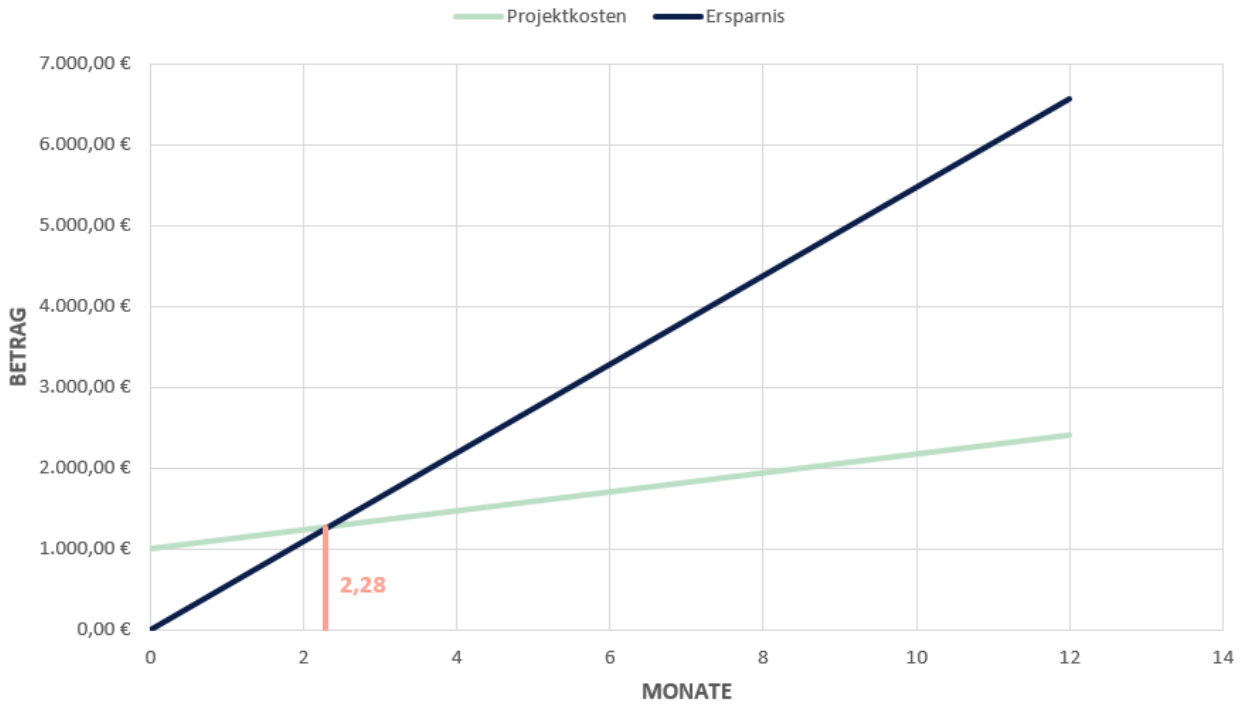


Abbildung 2: Amortisation

A5. Use-Case-Diagramm

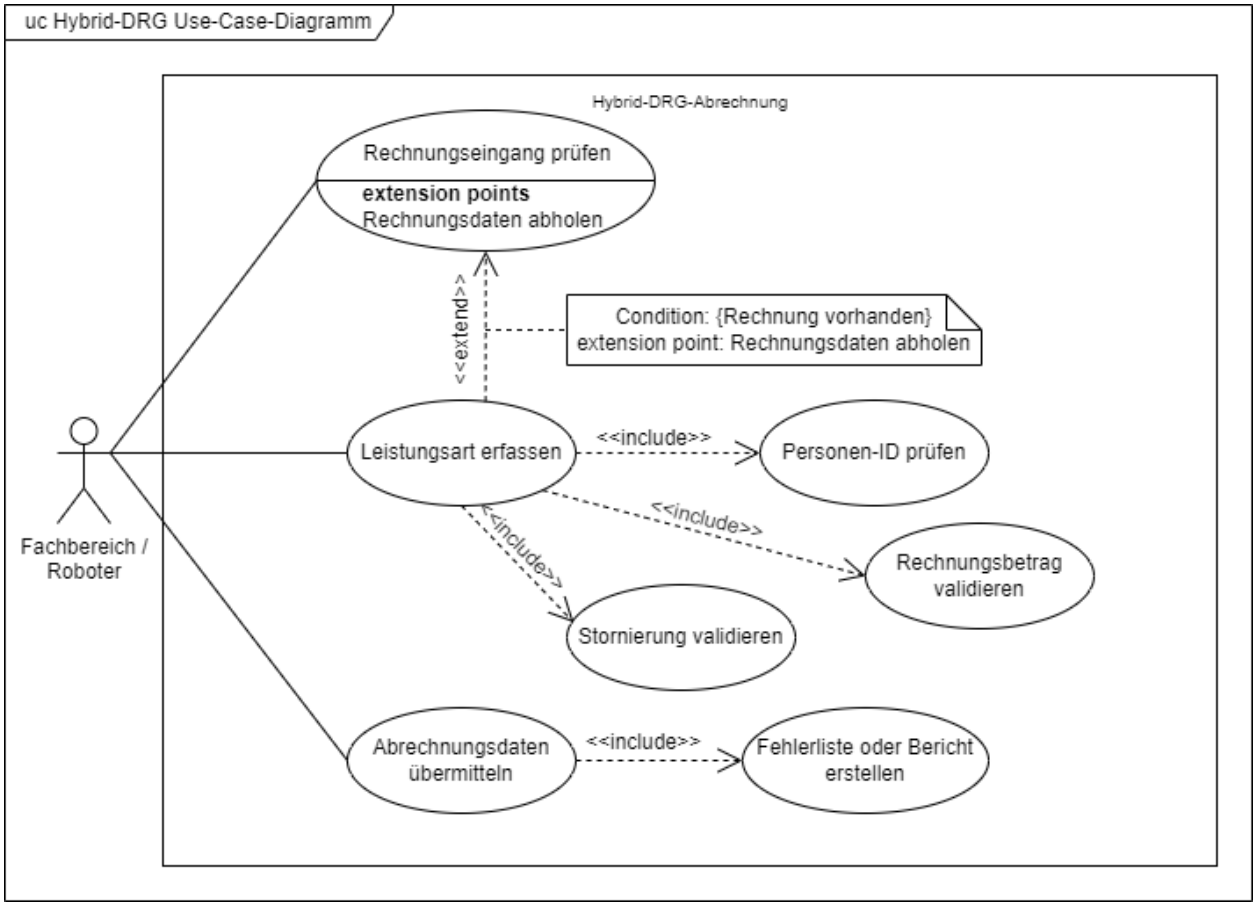


Abbildung 3: Use-Case-Diagramm

A6. Lastenheft

Der RPA- Prozess muss folgende Anforderungen erfüllen:

1. Verarbeitung der Daten

   Der RPA-Prozess muss …

   1.1. … in festgelegten Intervallen überprüfen, ob neue Rechnungen eingegangen sind.

   1.2. … sicherstellen, dass in jeder Rechnung in der Spalte „Personen-ID“ ein Wert enthalten ist.

   1.3. … überprüfen, ob eine Stornierung der Rechnung vorliegt.

   1.4. … verifizieren, dass der Rechnungsbetrag größer als 0 ist.

   1.5. … prüfen, ob die Person zum Leistungszeitpunkt eine gültige Krankenversicherung aufweist.

   1.6. … die zu abrechnende Leistungsart in das System UBLE erfassen.

   1.7. … die relevanten Abrechnungsdaten in das System ADA übermitteln.

1.8. … bei auftretenden Fehlern den Prozess stoppen, den entsprechenden Fehlercode erfassen und die betroffenen Rechnungsdaten in einem Fehlerprotokoll dokumentieren.

2. Darstellung der Daten

   Der RPA-Prozess muss …

   2.1. … regelmäßige Übersichten der verarbeiteten Rechnungen tabellarisch erstellen.

   2.2. … für fehlerhafte Abrechnungen eine verständliche Fehlermeldung und zugehörige Fehlercodes in einer separaten Fehlerliste tabellarisch speichern.

3. Sonstige Anforderungen

   Der RPA-Prozess muss …

   3.1. … die Berichte über die erfolgreich abgerechneten und fehlerhaften Rechnungen auf einem zentralen Laufwerk speichern, auf das nur der Fachbereich Zugriff hat.

   3.2. … die Berichte in Form von Excel-Dateien bereitstellen, um eine übersichtliche Darstellung und einfache Weiterverarbeitung der Abrechnungs- und Fehlerdaten zu ermöglichen.

   3.3. … mit GitLab versioniert werden.

A7. Echtzeitüberwachung

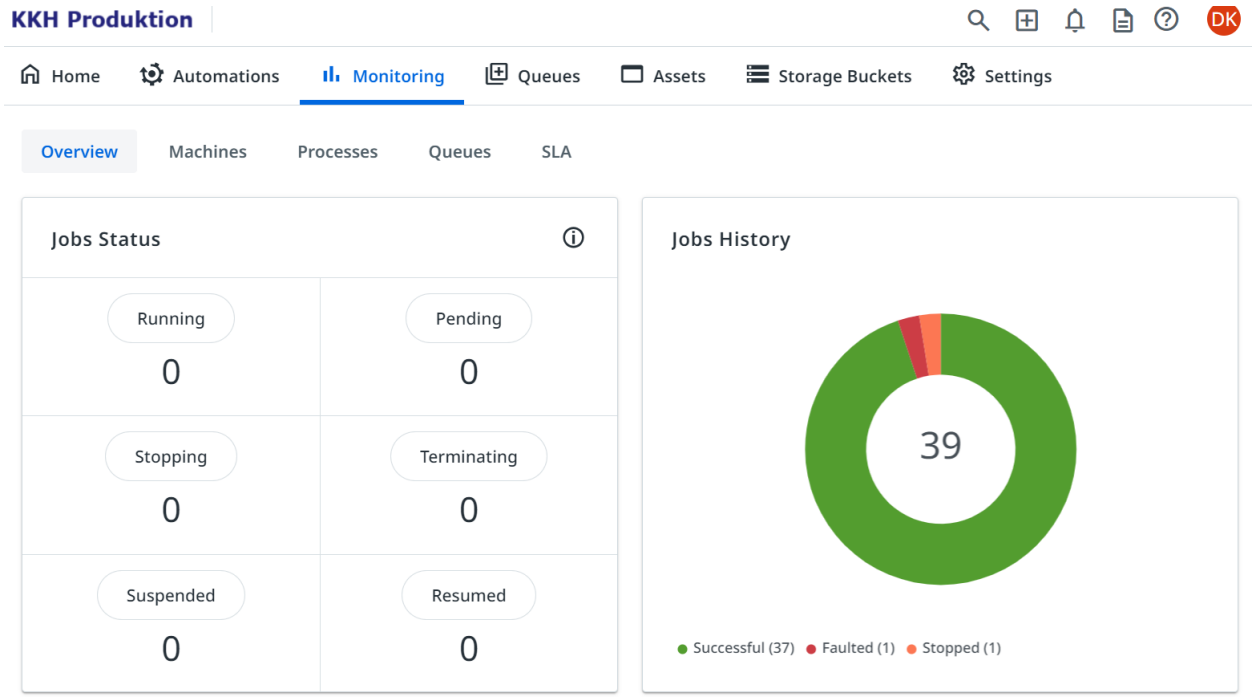


Abbildung 4: Echtzeitüberwachung

A8. Architektur

Abbildung 5: Architektur

A9. Entity-Relationship-Model

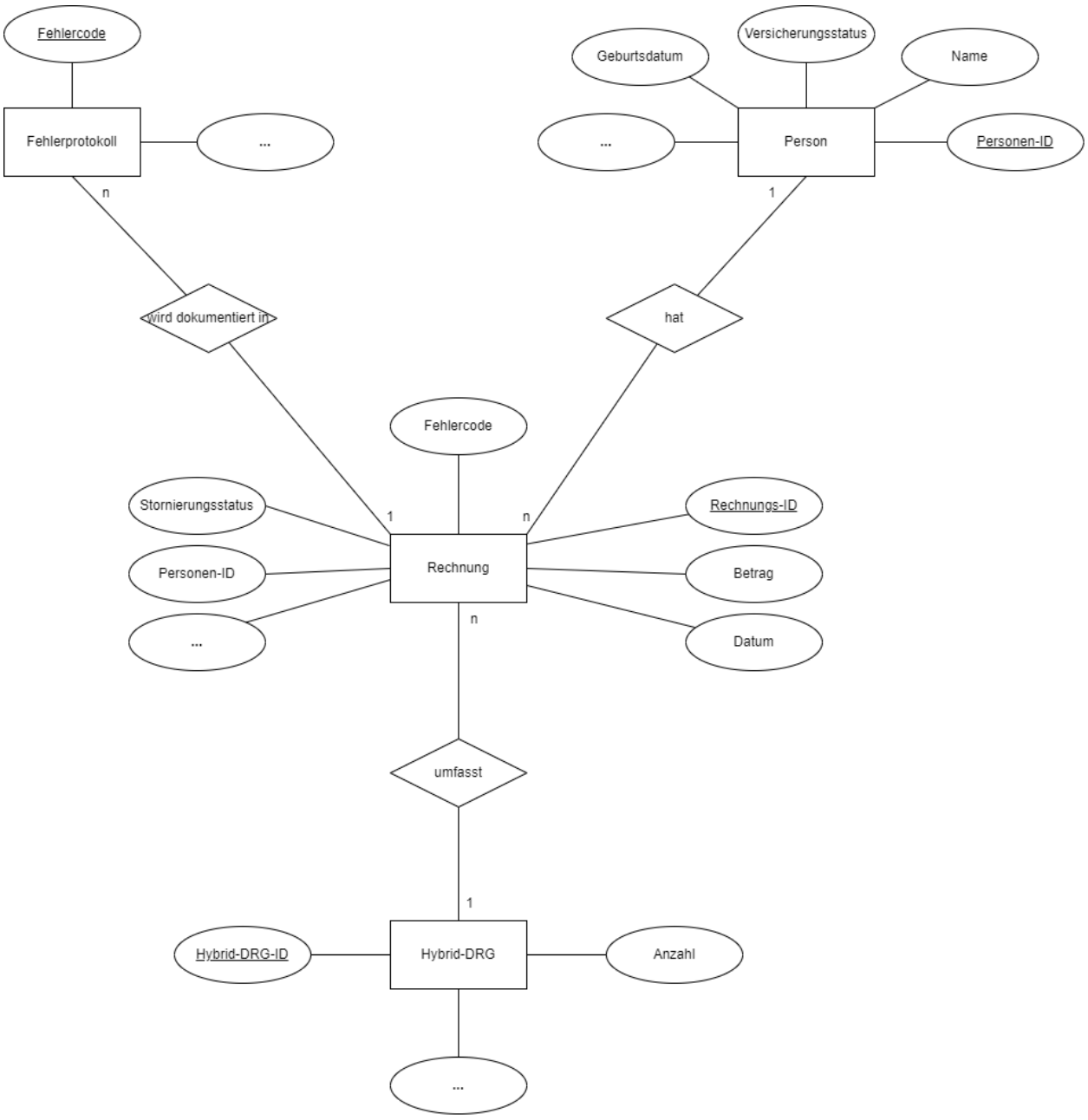


Abbildung 6: Entity-Relationship-Model

A10. BPMN-Diagramm

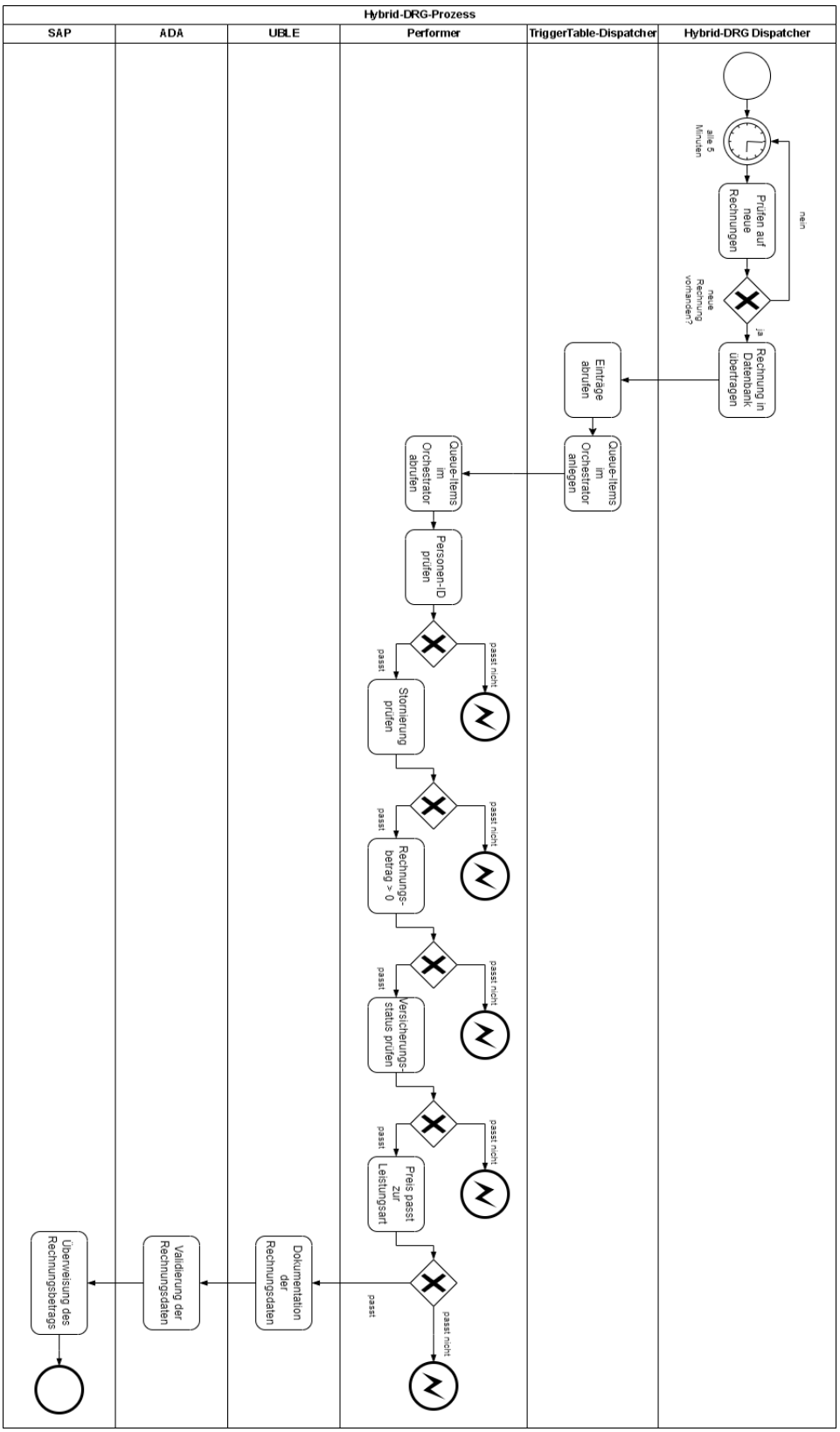


Abbildung 7: BPMN-Diagramm

A11. Deployment

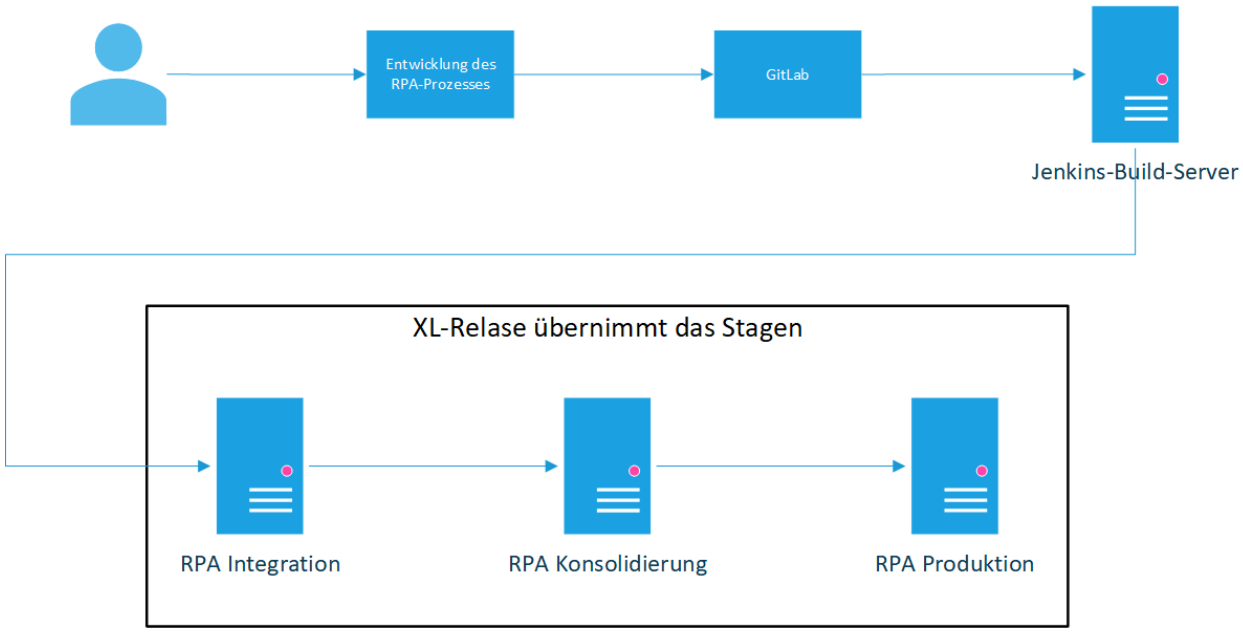


Abbildung 8: Deployment

A12. Pflichtenheft

Zielbestimmung

1. Plattform
   1.1. Die Automatisierung wird mit UiPath Studio entwickelt und durch den UiPath Orchestrator bereitgestellt.
   1.2. Die Implementierung der Automatisierungslogik erfolgt in C#.
   1.3. Die Automation wird nur im internen Netzwerk des Unternehmens verfügbar sein.
   1.4. GitLab wird für das Source-Code-Management und die Versionsverwaltung verwendet.
   1.5. Die automatisierten Prozesse werden regelmäßig überwacht und aktualisiert, um eine hohe Betriebssicherheit zu gewährleisten und Anpassungen an Änderungen in den Anforderungen vorzunehmen.
2. Datenbank
   2.1. Datenbankzugriffe und -abfragen erfolgen über UiPath Studio per SQL im Code.
   2.2. Import und Export der relevanten Daten werden über die passenden Aktivitäten im Workflow verarbeitet.
   2.3. Die Daten für die Prozessdurchführung werden in einem zentralen SQL-Datenbankserver des Unternehmens gespeichert.
3. Geschäftslogik
   3.1. Die Geschäftslogik wird in modularen Workflows und Automatisierungssequenzen abgebildet, die zentral auf dem Orchestrator verwaltet werden.
   3.2. Tests der Automatisierungskomponenten stellen sicher, dass die Geschäftslogik korrekt und fehlerfrei umgesetzt wird.
   3.3. Berichte und Fehlerlisten werden als Excel-Dateien generiert und für den Fachbereich in einem definierten Netzwerkverzeichnis abgelegt.
4. Qualitätssicherung
   4.1. UiPath Analyzer überprüft den Code auf Best Practices und identifiziert Optimierungspotenzial.
   4.2. Tägliche Meetings dienen der Synchronisation und Diskussion offener Fragen im Entwicklerteam.
   4.3. Regelmäßige Code-Reviews werden vom Entwicklerteam durchgeführt, um die Qualität und Effizienz der Implementierung sicherzustellen.
   4.4. Tests zur Qualitätssicherung werden durchgeführt, um die Funktionsfähigkeit aller wesentlichen Teile der Automatisierung sicherzustellen.

A13. Orchestrator-Queue

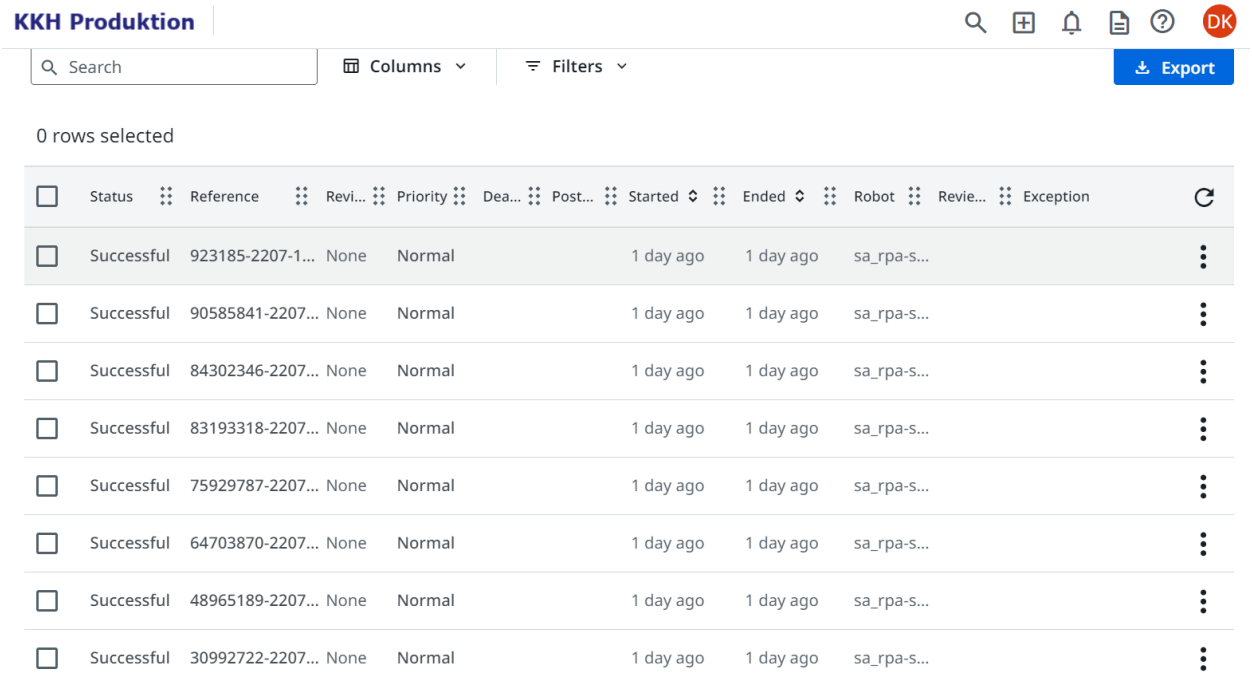


Abbildung 9: Orchestrator-Queue

A14. Variablen Ermittlung Verarbeitungsdaten

Name	Variablentyp	Bereich
dt_sqlAbholung	DataTable	Ermittlung_Verarbei...
str_sqlAbholung	String	Ermittlung_Verarbei...
str_inputFLLLNDL	String	Ermittlung_Verarbei...
str_inputFLLNANR	String	Ermittlung_Verarbei...
str_vornameVersicherter	String	Ermittlung_Verarbei...
str_nameVersicherter	String	Ermittlung_Verarbei...
datetime_gebVersicherter	DateTime	Ermittlung_Verarbei...
str_lebenslangeArztNr	String	Ermittlung_Verarbei...
str_arztBetriebstaettenNr	String	Ermittlung_Verarbei...
str_vnrLeistungsbezieher	String	Ermittlung_Verarbei...
int_abrechnungsjahr	Int32	Ermittlung_Verarbei...
int_abrechnungsquartal	Int32	Ermittlung_Verarbei...
Variable erstellen		

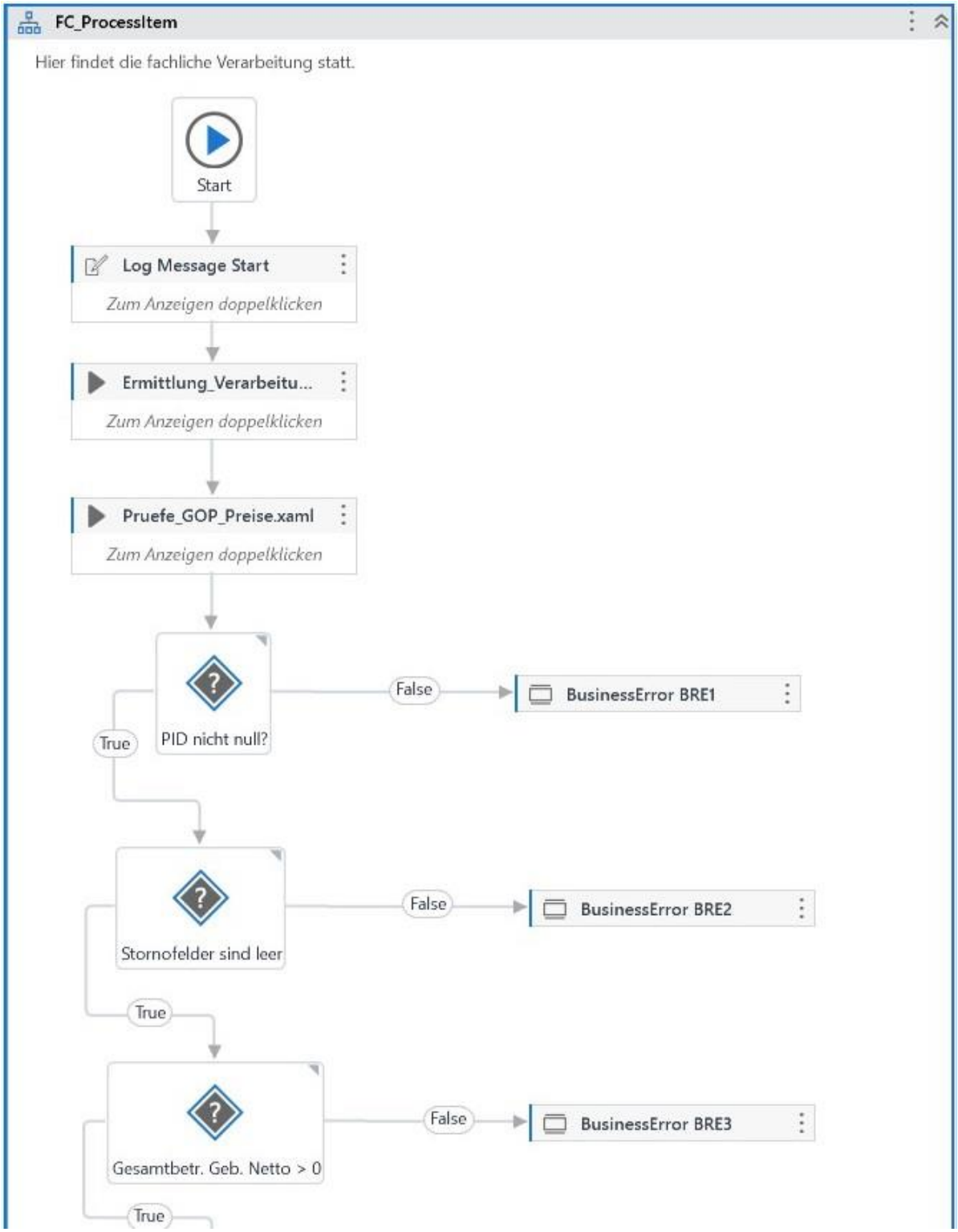
Abbildung 10: Variablen Ermittlung Verarbeitungsdaten

A15. Argumente Ermittlung Verarbeitungsdaten

Name	Richtung	Argumenttyp
out_str_zahlungsempfaenger	Aus	String
out_str_rechnungssteller	Aus	String
out_datetime_rechnungsdatumDA	Aus	DateTime
out_str_rechnungsNr	Aus	String
out_null_int_pid	Aus	Nullable<Int32>
out_datetime_ersterTagAbrZeitr	Aus	DateTime
out_datetime_letztTagAbrZeitr	Aus	DateTime
out_datetime_verarbeitungsDat	Aus	DateTime
out_dbl_rechnungsBetrag	Aus	Double
out_str_hybridDRG	Aus	String
out_int16_anzahlDRG	Aus	Int16
out_bool_storno	Aus	Boolean
Argument erstellen		

Abbildung 11: Argumente Ermittlung Verarbeitungsdaten

A16. Logik Hauptprozess



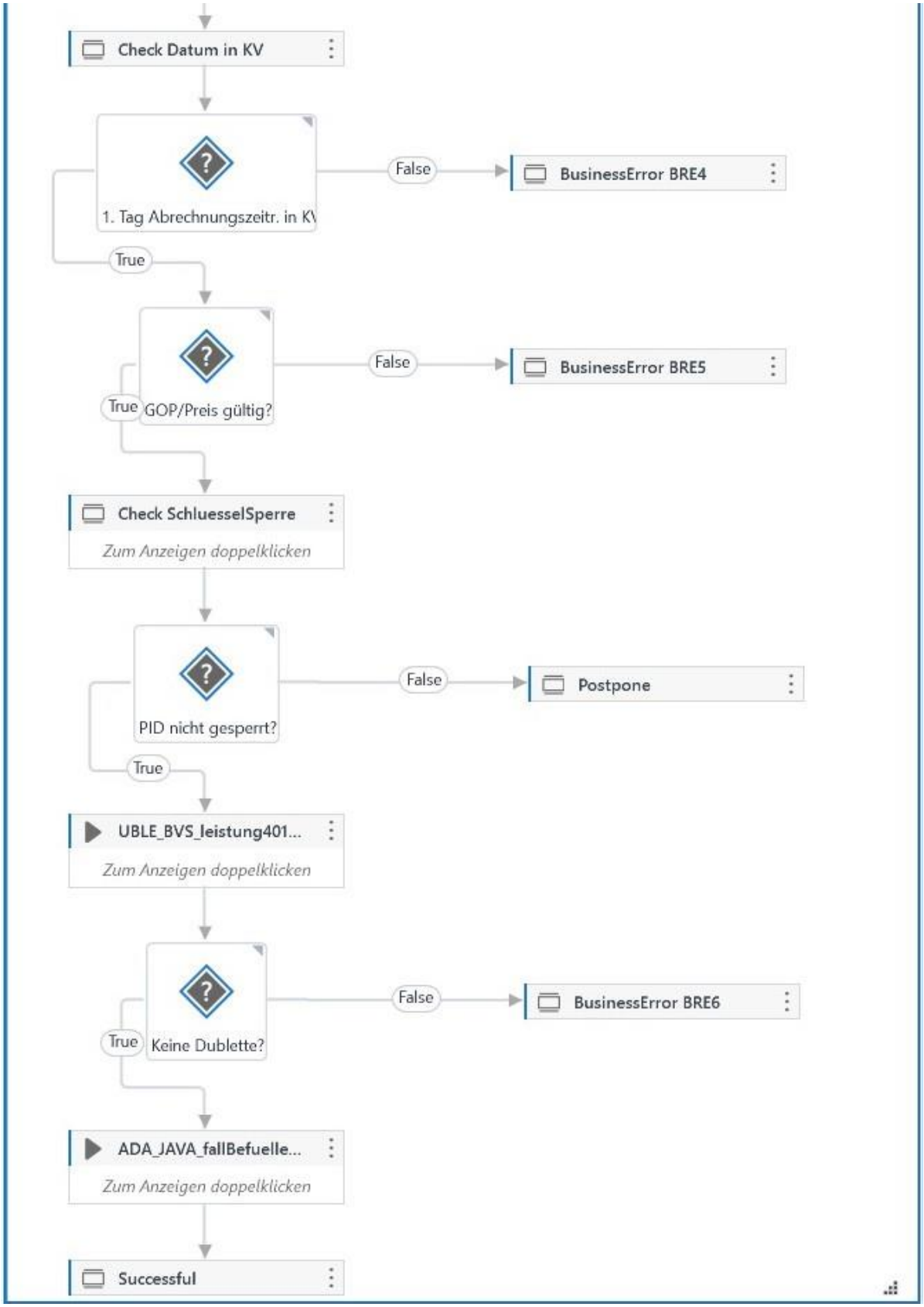


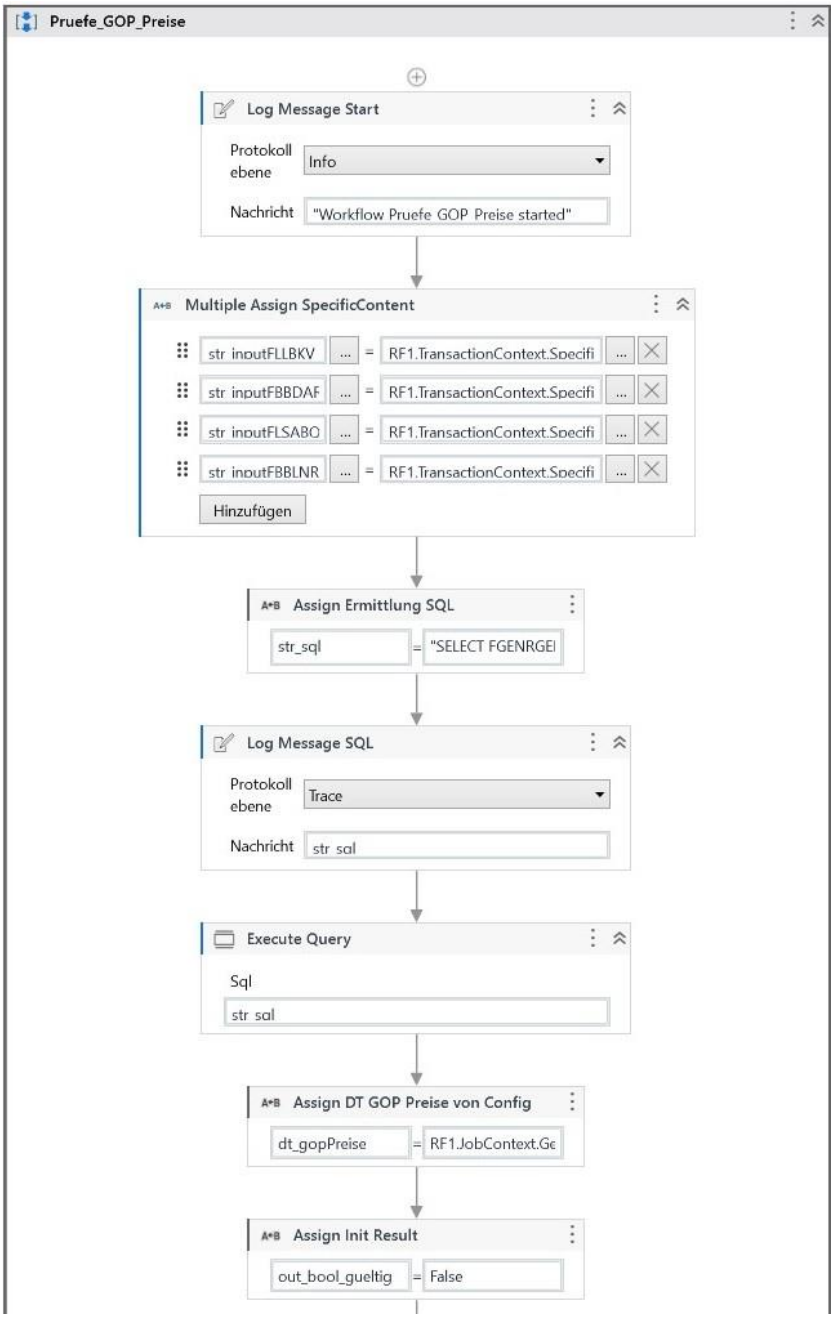
Abbildung 12: ProcessItem

A17. Variablen Hauptprozess

Name	Variablentyp	Bereich
datetime_ersterTagAbrZeitr	DateTime	FC_ProcessItem
datetime_letztTagAbrZeitr	DateTime	FC_ProcessItem
str_zahlungsempfaenger	String	FC_ProcessItem
str_rechnungssteller	String	FC_ProcessItem
bool_dublette	Boolean	FC_ProcessItem
datetime_verarbeitungsDat	DateTime	FC_ProcessItem
str_rechnungsNr	String	FC_ProcessItem
datetime_rechnungsdatumDA	DateTime	FC_ProcessItem
bool_datumLiegtInKV	Boolean	FC_ProcessItem
null_int_pid	Nullable<Int32>	FC_ProcessItem
bool_gopPreisGueltig	Boolean	FC_ProcessItem
bool_schluesselGesperrt	Boolean	FC_ProcessItem
dbl_rechnungsBetrag	Double	FC_ProcessItem
str_hybridDRG	String	FC_ProcessItem
int16_anzahlDRG	Int16	FC_ProcessItem
bool_storno	Boolean	FC_ProcessItem
Variable erstellen		

Abbildung 13: Variablen Hauptprozess

A18. Ursprüngliche Preisprüfung



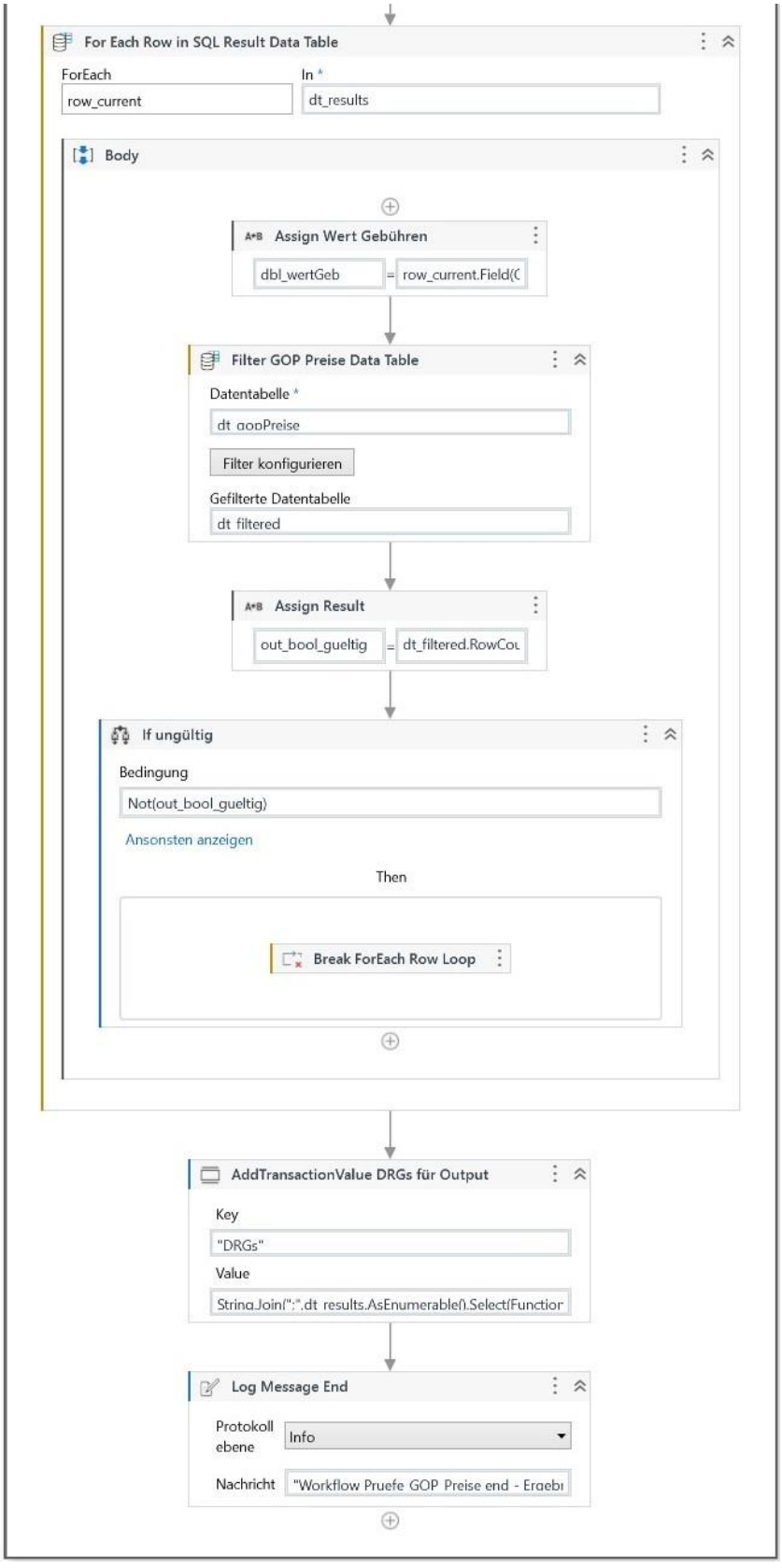


Abbildung 14: Preisprüfung alt

A19. Aktuelle Preisprüfung

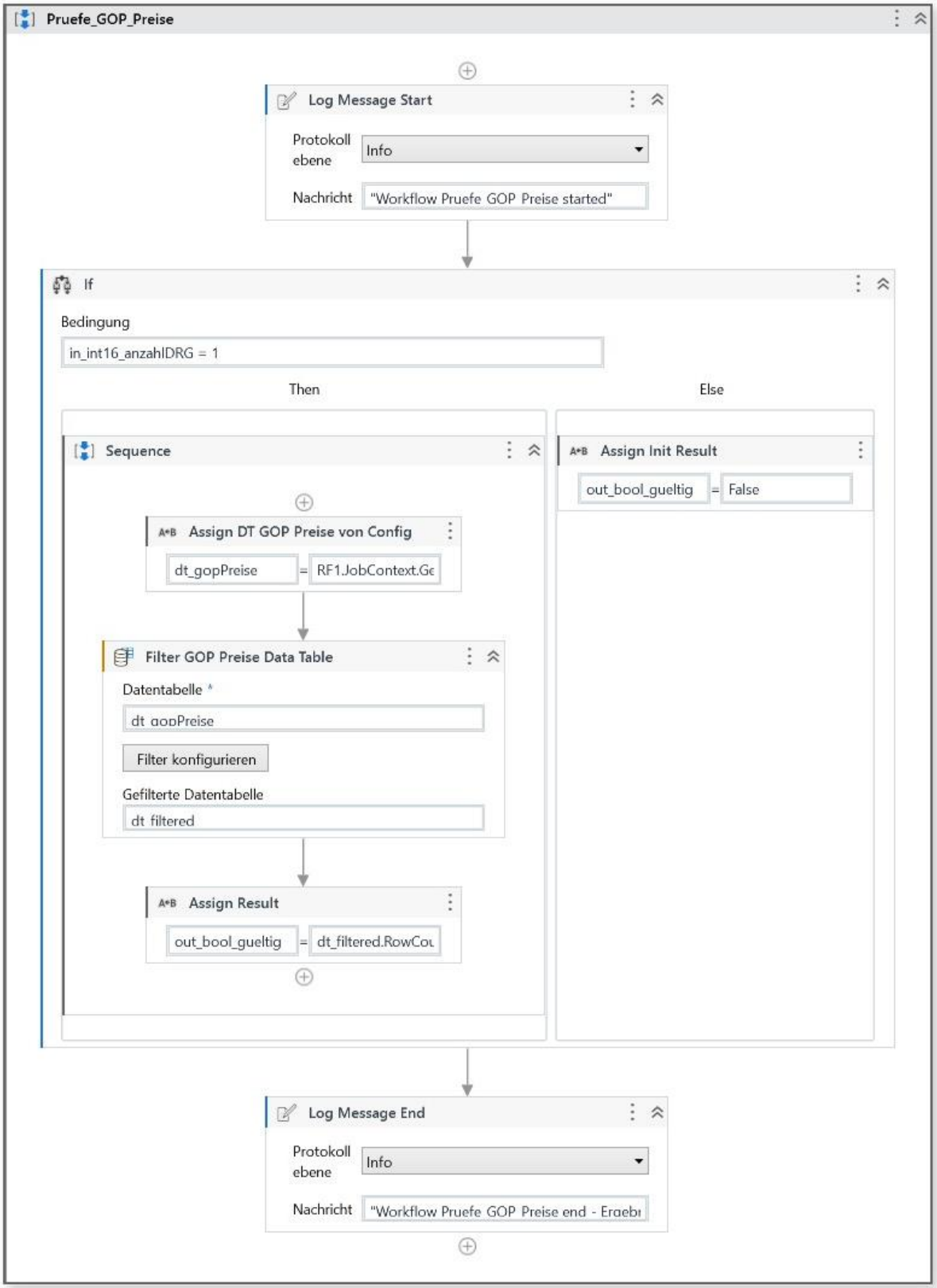


Abbildung 15: Preisprüfung neu

A20. Storage Bucket

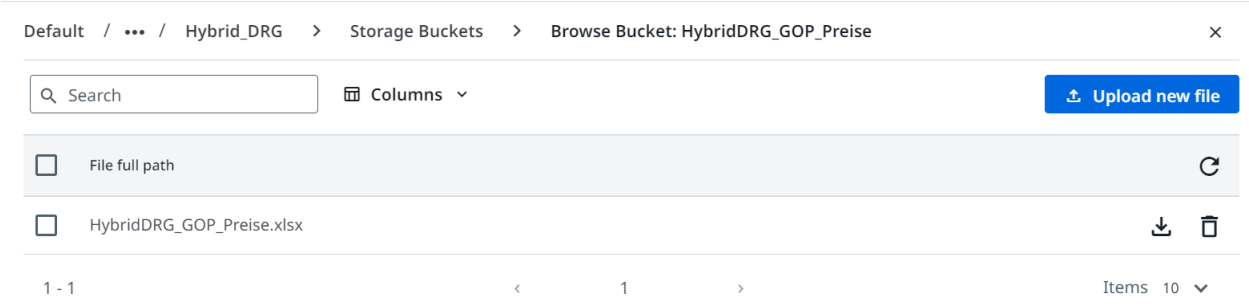


Abbildung 16: Storage Bucket

A21. Logik Hybrid-DRG-Dispatcher

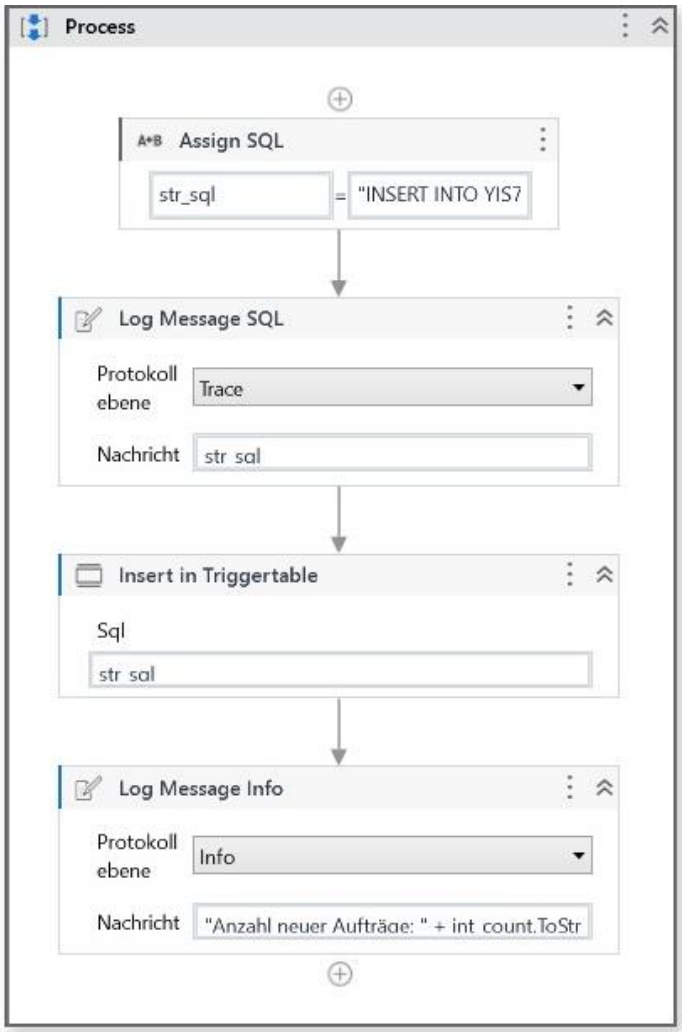


Abbildung 17: Logik Hybrid-DRG-Dispatcher

A22. Logik Trigger-Table-Dispatcher

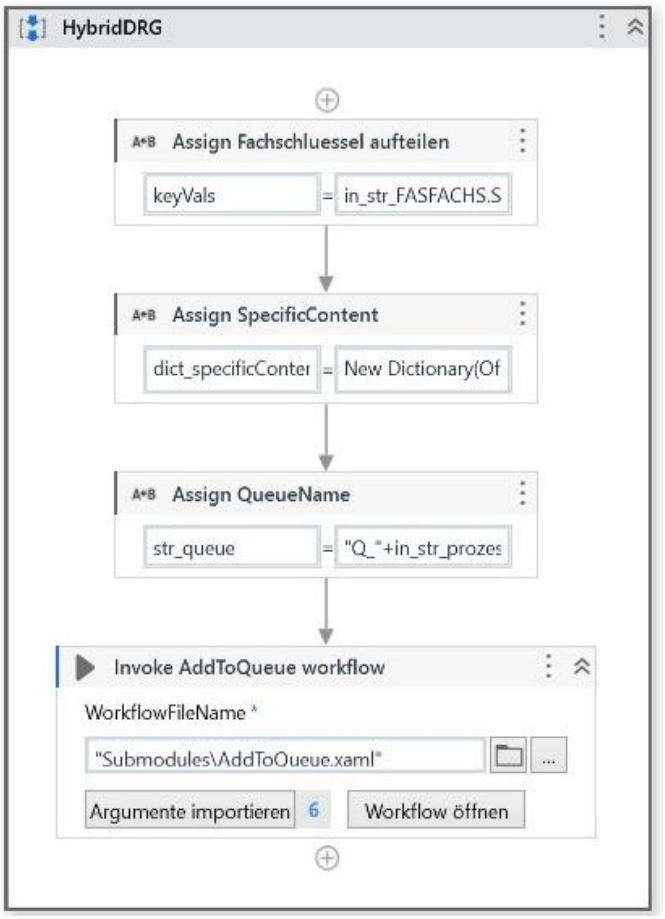


Abbildung 18: Logik Trigger-Table-Dispatcher

A23. Test Ermittlung Verarbeitungsdaten

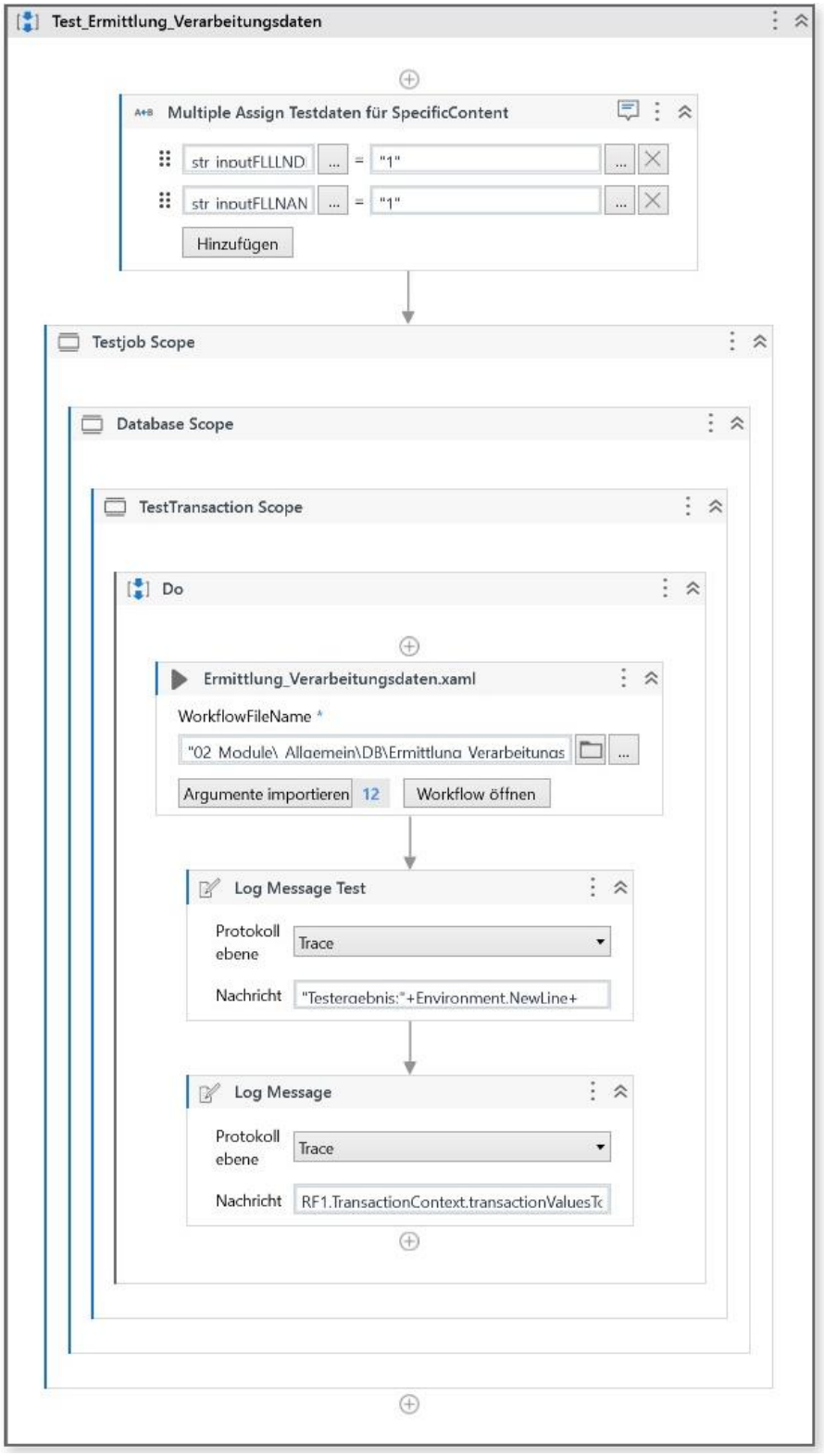


Abbildung 19: Test Ermittlung Verarbeitungsdaten

A24. Test Preisprüfung

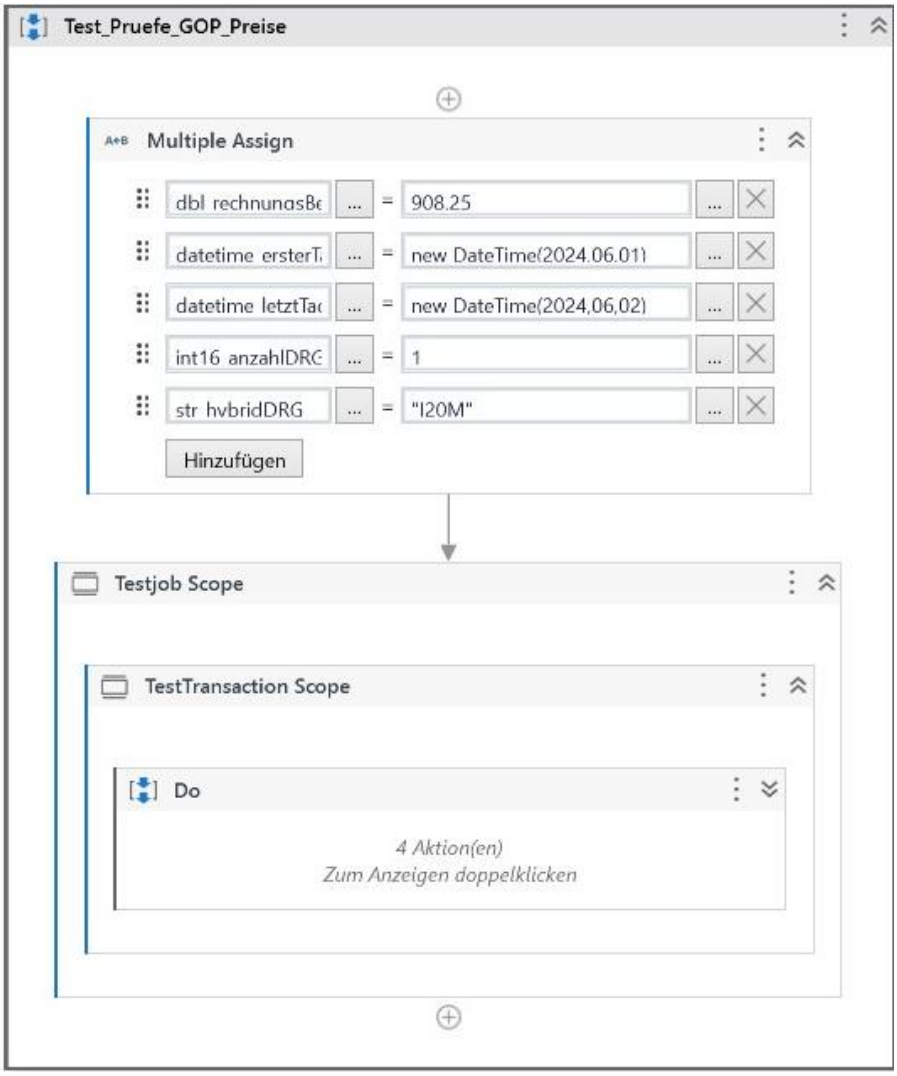


Abbildung 20: Test Preisprüfung

A25. Erfassung der Leistungsart in UBLE

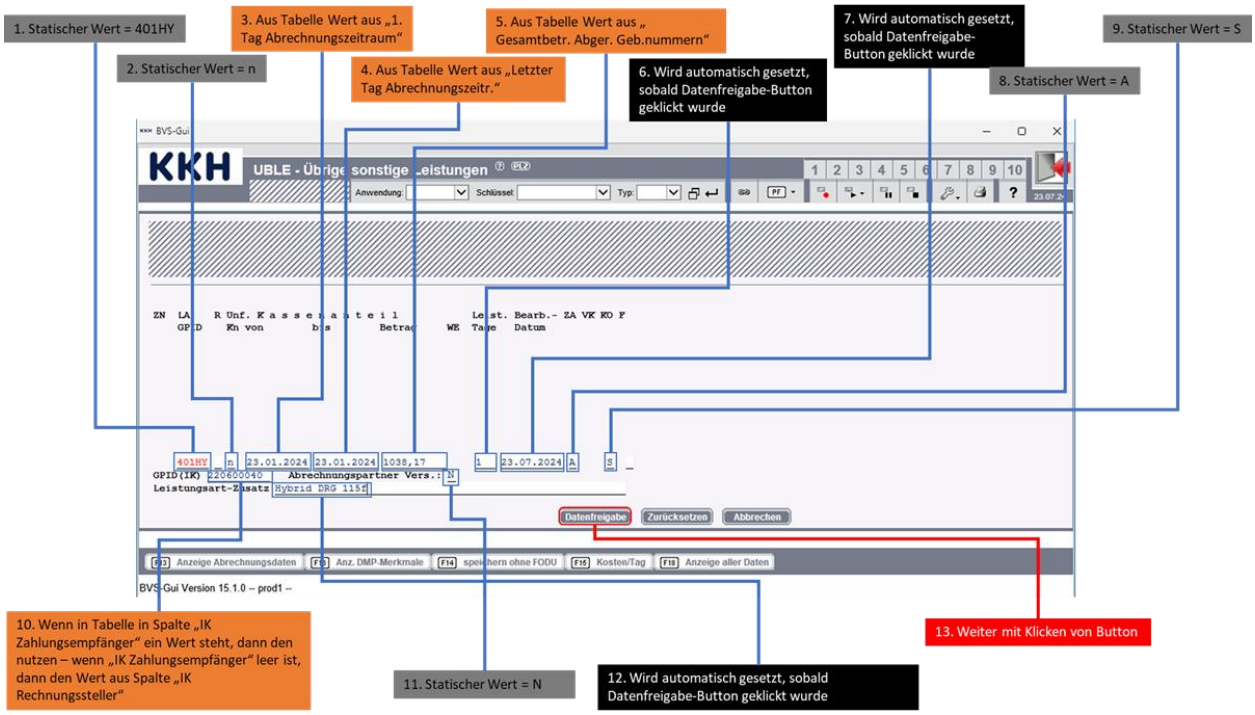


Abbildung 21: Erfassung der Leistungsart in UBLE

A26. Auftrag in ADA öffnen

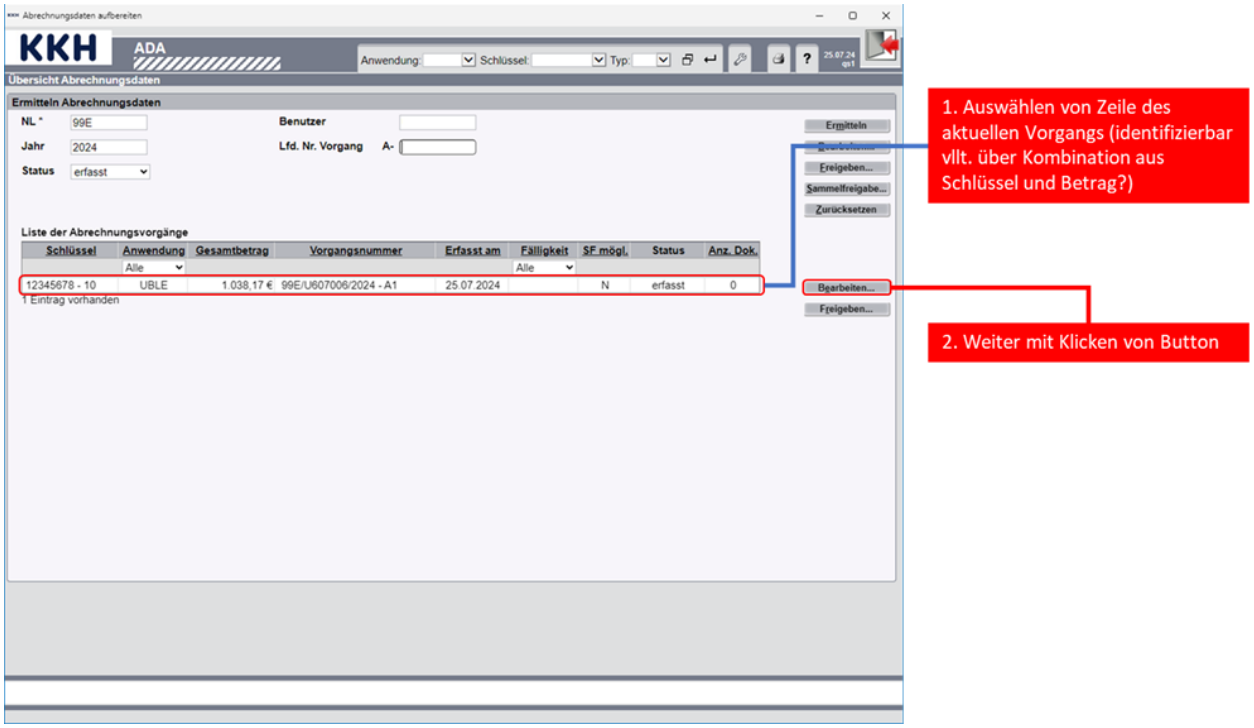


Abbildung 22: Auftrag in ADA öffnen

A27. Befüllen der ADA-Bearbeitungsmaske

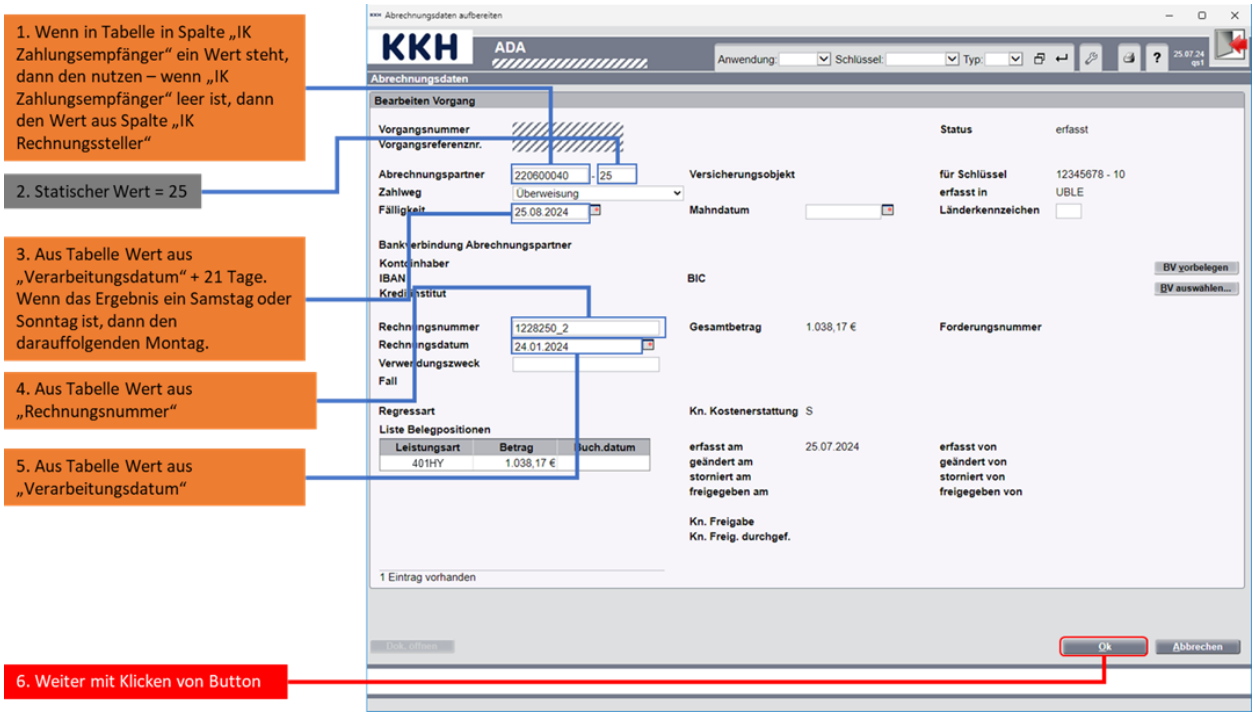


Abbildung 23: Befüllen der ADA-Bearbeitungsmaske

A28. Commit in Sourcetree

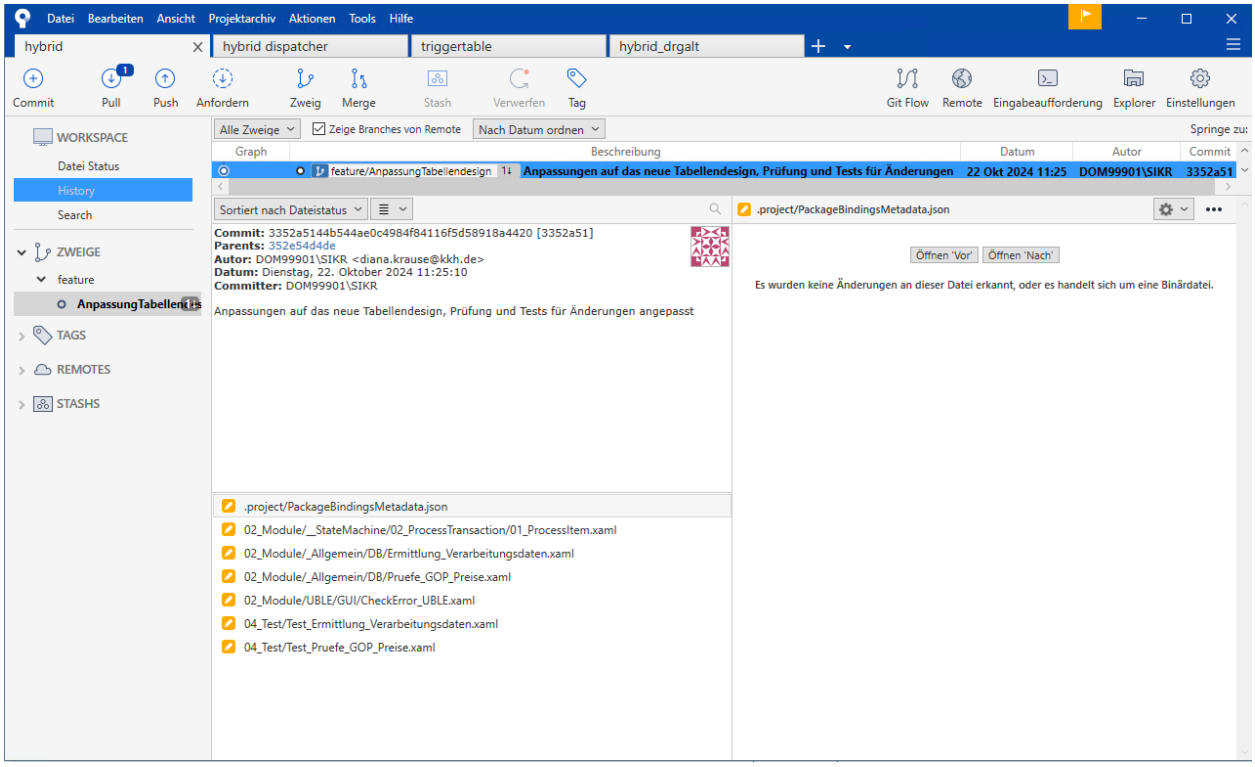


Abbildung 24: Commit in Sourcetree

Bestätigung über die betriebliche Projektarbeit