Bachelorthesis

Entwicklung einer Orchestrierungskomponente zur Ablaufsteuerung von ETL-Prozessen mit KNIME

zur Erlangung des akademischen Grades

Bachelor of Science

eingereicht im Fachbereich Mathematik, Naturwissenschaften und Informatik an der Technischen Hochschule Mittelhessen

von

Pia Georgiew

15. Mai 2022

Referent: Prof. Dr. Frank Kammer

Korreferent: Michael Markert

Erklärung der Selbstständigkeit

Hiermit versichere ich, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie die Zitate deutlich kenntlich gemacht zu haben.

Gießen, den 15. Mai 2022 Pia Georgiew

Inhaltsverzeichnis

Abstract

Eine Orchestrierung übernimmt die automatisierte Ausführung von komplexen Prozessen, in dem bestehenden Anwendungsfall von ETL-Prozessen. Anhand von Anforderungen, definiert in der Konzeptionsphase, wird ein Vergleich zwischen einer selbst entwickelten Orchestrierung und der Workflow-Management-Plattform Airflow gezogen. Während Airflow deutlich mehr Features und Flexibilität bietet, ist der Integrationsaufwand sehr hoch. Die eigens in KNIME entwickelte Orchestrierung fügt sich in die bestehende Systemarchitektur ein und erfüllt durch die Neuentwicklung alle spezifisch bestehenden Prozessanforderungen.
Während der Prozessierung der Datenanlieferungen werden Metriken gesammelt, um die Transparenz zu erhöhen. Zur Anzeige der Metriken wird ein Monitoring mit Tableau entwickelt.
Durch die Verwendung einer Orchestrierung konnte der händische Aufwand während der Prozessierung deutlich gesenkt werden. Zusätzlich ist sichergestellt, dass im belieferten Data Warehouse die Datenkonsistenzen nicht verletzt werden. Mit einer verbesserten Orchestrierung und einem Monitoring kann außerdem die Datenqualität der verarbeiteten Daten gesteigert werden, indem unter anderem die Richtigkeit und die Aktualität der Daten verbessert wird.

1 Einführung

Das Datenvolumen im Jahr 2020 umfasste schätzungsweise 50 Zetabyte. [Bmw20] Das ist ungefähr so viel wie auf einer Billionen Blu-ray Discs gespeichert werden könnte. Allerdings kann das Potenzial, was all diese Daten bieten, nicht vollständig ausgeschöpft werden, da Daten nicht immer gesammelt und homogenisiert werden. Nur in einem gleichartigen Format können große Datenmengen sinnvoll ausgewertet werden. Die Daten werden zum Beispiel analysiert, um neue Markttrends zu erkennen, ein Mittel dafür ist künstliche Intelligenz. Die Auswertungsmethoden können allerdings noch so gut sein, eine entscheidende Rolle spielt immer auch die Qualität der Daten, auf deren Basis ausgewertet wird. Das Homogenisieren kann unter anderem mittels sogenannten ETL-Prozessen realisiert werden. ETL steht für die Extraktion, die Transformation und das Laden. In einem ETL-Prozess werden Daten, kommend aus einer Datenquelle, aufbereitet und dann in das eigene Datenbasis eingepflegt. Dieser Prozess umfasst viele Teilschritte und die neu hinzugekommenen Daten müssen regelmäßig verarbeitet werden. Das regelmäßige automatisierte Aufrufen der Prozesse kann von einer sogenannten Orchestrierung übernommen werden.

1.1 Begriffsdefinition Orchestrierung

Der Begriff Orchestrierung hat seine Herkunft in der Musik. Hier beschreibt er die Ausarbeitung einer Gesamtkomposition mit vielen beteiligten Instrumenten und Sängern. In der Informatik steuert eine Orchestrierung statt den Musizierenden die ablaufenden Prozesse. Dabei kann es sich um verschiedenste Prozesse handeln. Ein starker Gebrauch ist unter anderem im Cloud-Umfeld zu beobachten. [Ran15]
Das Ziel hinter einer Orchestrierung besteht darin, komplexe Prozesse unterteilt in Workflows automatisiert ablaufen zu lassen. Automatisiert bedeutet in diesem Fall, dass die Orchestrierung nur eine menschliche Interaktion benötigt, um die gesamte Datenanlieferung zu prozessieren. Daher kann die zu entwickelnde Orchestrierung auch als 1-Click-Orchestrierung bezeichnet werden.

Eine Orchestrierung hat also die Aufgabe, die Workflows, in denen die Logik zur Abarbeitung der ETL-Prozesse enthalten ist, in der richtigen Reihenfolge aufzurufen. Vor dem Aufruf eines Workflows müssen Aufrufbedingungen überprüft werden. In den Workflows können unterschiedlichste Aufgaben bearbeitet werden, für die Orchestrierung ist es unbedeutend welche.
Eine Lösungsmöglichkeit für eine Orchestrierung im Folgenden als Einführungsbeispiel. Da das Prinzip der Ablaufsteuerung in jeder Orchestrierung identisch ist, wird eine generalisierte Orchestrierungskomponente entwickelt. Eine Orchestrierung besteht also aus der Orchestrierungskomponente und ihrer prozessspezifischen Konfiguration. Bei den zu steuernden Prozessen handelt es sich im konkreten Anwendungsfall um ETL-Prozesse. In Abbildung 1.1 ist der Ablauf in einer Orchestrierung noch einmal visuell dargestellt.

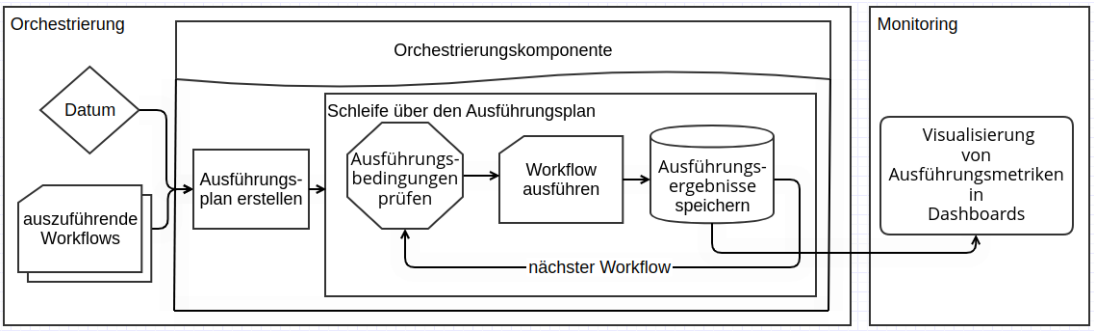

Abbildung 1.1: Überblick über die Aufgaben von Orchestrierung und Monitoring

Es wird die Orchestrierungskomponente aufgerufen in der die Orchestrierungslogik enthalten ist. Eingegeben in diese wird unter anderem das Ausführungsdatum und die auszuführenden Workflows. Zusätzlich kann zwischen verschiedenen Ausführungsbedingungen gewählt werden, wie zum Beispiel der Voraussetzung, dass alle Workflows am vorherigen Tag erfolgreich ausgeführt worden sind. In dem betrachteten Fall wird von einer täglichen Ausführung ausgegangen.

Das Entwicklungsziel besteht darin die Datenanlieferungen zu automatisieren und die Datenqualität zu erhöhen. Ob und wie dies erreicht werden kann, wird in dieser Abschlussarbeit thematisiert.

1.2 Vorgehensplanung

Bevor mit der Entwicklung der neuen Orchestrierung begonnen werden kann, müssen die Anforderungen, die es an die Orchestrierung gibt, herausgearbeitet werden. Dazu

wird in diesem Kapitel zuerst das bestehende Problem beschrieben und danach ein Ziel formuliert. Zusätzlich wird das Vorgehen während der Entwicklung festgelegt.

Problembeschreibung: Da jeden Tag Daten aus mehreren Datenquellen zusammengetragen werden, laufen jeden Tag mehrere ETL-Prozesse. Für jeden von diesen gibt es bisher eine eigene Ablaufsteuerung. Diese Ablaufsteuerungen haben die Aufgabe, die Workflows, in denen der ETL-Prozess abläuft, in der richtigen Reihenfolge aufzurufen. Dadurch dass die Daten durch viele Instanzen laufen, kommt es immer wieder zu verschiedensten Fehlern. Wenn es bei einer der vorherigen Instanzen zu Verzögerungen kommt, dann führt dies dazu, dass beim Anlauf der Orchestrierungen noch keine Daten vorhanden sind. Die Extraktion aus der Datenquelle schlägt fehl. Sobald die Datenanlieferung zur Verfügung steht, kann die Orchestrierung erneut gestartet werden. Zeitintensiv wird es, wenn sich die Tabellenstruktur oder eine Wertausprägung ändert. Dadurch, dass die Datenanlieferungen aus einem sich ständig entwickelnden System kommen, gibt es immer wieder unbekannte Wertausprägungen. Es kann zum Beispiel vorkommen, dass die zulässige Wertmaximallänge, festgelegt im Datenbankschema, überschritten wird. Dies führt dazu, dass der Workflow in einem Zwischenstatus stecken bleibt. Es muss nun händisch ein konsistenter Zustand hergestellt werden. Dieser Zustand ist erreicht, wenn die Ablaufsteuerung fertig abgearbeitet ist und dadurch die Datenlieferung vollständig in den bestehenden Datensatz integriert wurde.
Um die fehlerhafte Datenanlieferung abzuarbeiten, muss zuerst das eigentliche Problem gefunden und beseitigt werden. Dafür müssen die angelieferten Daten, der Workflow und/oder die jeweilige Tabelle bearbeitet werden. Eine Datenanlieferung besteht meist aus mehreren Tabellen. Für jede dieser Tabellen müssen außerdem die, durch den Abbruch, noch fehlenden Prozessschritte im Workflow nachgezogen werden. Danach werden die restlichen Workflows einzeln gestartet. Alternativ können auch alle bisherigen Änderungen gelöscht werden. Die Ablaufsteuerung wird dann neu gestartet. Dieser Prozess kostet viel Zeit und blockiert die Bearbeitung anderer Aufgaben.
Um Verzögerungen entgegenzuwirken soll eine automatisiertere Orchestrierung zum Einsatz kommen. Mit einem darauf aufgesetzten Monitoring sollen die Data Engineers zudem schneller Fehler in den Daten finden können. Auch starke Veränderungen der Datenmengen und Verarbeitungszeiten können auf Fehler beim Datenlieferanten hinweisen.

Zielsetzung: Ziel dieser Arbeit ist es, mögliche Lösungen zu vergleichen und dann eine Orchestrierung zu entwickeln, welche die Ablaufsteuerung von verschiedensten ETL-Prozessen übernehmen kann. Die Orchestrierungen (Eingabeknoten + Orchestrierungskomponente) sollen einen hohen Automatisierungsgrad aufweisen. Neben der Orchestrierungskomponente soll die Infrastruktur für ein Monitoring der ETL-Prozesse

geschaffen werden. Abschließend soll die Steigerung der Datenqualität durch die Neuerungen bewertet werden.

Vorgehensweise: Die Vorgehensweise zur Erreichung der Zielsetzung, kann in drei Phasen unterteilt werden. Begonnen wird mit der Anforderungsaufnahme, danach folgt die Konzeptionsphase und die Entwicklungsphase. Zur Anforderungsaufnahme wird zuerst die bestehende Systemarchitektur analysiert. Der Fokus liegt hier auf allen Prozessen die etwas mit der Abarbeitung der ETL-Prozesse zu tun haben. Außerdem wird protokolliert, wie das händische Vorgehen zum Wiederanlauf nach Fehlern aussieht. Mittels der daraus gewonnen Erkenntnisse werden Anforderungen definiert, welche die Orchestrierung erfüllen soll. Anhand der festgelegten Anforderungen wird in Phase Zwei ein Orchestrierungskonzept erstellt werden. Mit dem Konzept als Basis, kann dann ein erster funktionaler Prototyp entwickelt werden, damit beginnt die Entwicklungsphase. Die Entwicklung erfolgt in einer vollständigen Testumgebung. In der Testumgebung sind alle beteiligten Instanzen nachgebildet. Dazu gehört natürlich auch die Orchestrierung. Diese ruft, eigens für diesen Zweck angelegte, Workflows auf. Nach Fertigstellung einer Version werden sich Testfälle überlegt und anhand dieser die neue Version überprüft. Nach dem erfolgreichen Abschluss der Tests, wird die Orchestrierung von der Testumgebung in die Produktivumgebung migriert. Änderungen im Produktivsystem werden durch Snapshots in KNIME versioniert.
Die Entwicklung des Monitoring-Dashboards erfolgt lokal mit Tableau, einer Datenvisualisierungssoftware. Nach Fertigstellung wird es auf dem Tableau-Server der Abteilung publiziert. Neue Features werden als neue Versionen auf den Server gepusht.
Die Produktivsetzung der Orchestrierung soll mit der ersten lauffähigen Version erfolgen. Dabei soll sich auf die Steuerung einer ETL-Strecke beschränkt werden, um die Inbetriebnahme und Fehlersuche zu vereinfachen. Durch den frühen Einsatz im Produktivumfeld, wird sich erhofft, dass die Anforderungen und damit auch die Orchestrierung stetig verbessert und weiterentwickelt werden können. Nach und nach soll dann die Umstellung aller Orchestrierungen auf die neue Komponente erfolgen. Dafür müssen jeweils auch die Workflows in denen der ETL-Prozess abläuft leicht angepasst werden. Es wird in der Logik der Orchestrierung verankert sein, dass sie täglich läuft. Daher müssen alle Workflows in einem täglichen Schedule Daten prozessieren. Das Datum wird am Anfang in die Workflows eingegeben. Diese Vorgehensweise ist neu und muss daher erst in den bestehenden Strukturen ergänzt werden.

2 Hintergrund

In diesem Kapitel wird, die in der Systemarchitektur verwendete Software, KNIME vorgestellt. Die ETL-Prozesse werden mit KNIME entwickelt und ausgeführt. Anschließend folgt eine kurze Einführung in Tableau, was ebenfalls in der Systemarchitektur für Auswertungen genutzt wird und zur Entwicklung des Monitorings verwendet werden kann. Um einen besseren Einblick in die ETL-Prozesse zu bekommen folgt, nach einer generellen Einführung in ETL-Prozesse, zudem die Vorstellung des ETLs des Kundenmanagementsystems. Am Ende des Kapitels wird der Begriff "Datenqualität" definiert und die beeinflussenden Faktoren beschrieben.

2.1 KNIME

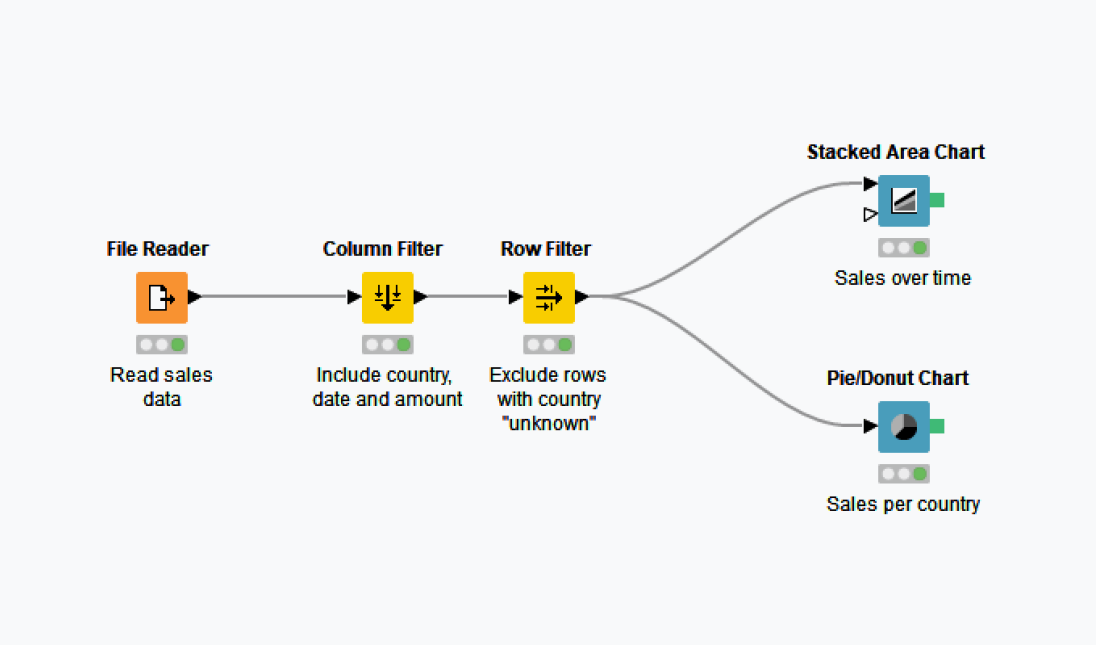

Abbildung 2.1: Beispielworkflow in KNIME [KNIoD]

KNIME ist eine Software zur interaktiven Datenverarbeitung und Analyse von großen Datenmengen. Das Prinzip von KNIME ist, dass Datenanalyseaufgaben in kleinere Einzelschritte zerlegt werden. Ein Einzelschritt wird in KNIME als Node dargestellt. Ein Beispiel mit mehreren verketteten Nodes ist in Abbildung 2.1 dargestellt. Die Nodes

müssen vor der Nutzung nur konfiguriert werden, die eigentliche Programmierarbeit beschränkt sich auf ein Minimum. Dadurch eignet sich KNIME auch für Teams mit weniger ausgeprägten Programmierkenntnissen.
Nodes verändern die eingegebene Tabelle entsprechend ihrer Funktion. Es könnte sich zum Beispiel um einen Row-Filter handeln, mit diesem können alle Zeilen mit einem ausgewählten Wert herausgefiltert werden. Ein Node hat immer einen Ausführungsstatus, eine Symbol und optional eine Beschreibung. Durch die ständige Visualisierung der Abläufe, können Ausführungen schnell nachvollzogen werden. Nodes können aneinandergekettet werden und so die eingegebene Tabelle bearbeiten oder auch Variablen weitergeben und verändern. Das Besondere an KNIME ist, dass es darauf ausgelegt ist, Tabellen zu verarbeiten. Die Bearbeitung funktioniert daher spalten- oder zeilenweise. Außerdem gibt es die Möglichkeit, mehrere Tabellen zu konkatenieren und Schleifen laufen zu lassen.
Die verketteten Nodes werden in Workflows gespeichert. Um die Übersichtlichkeit in den Workflows zu steigern, können Komponenten verwendet werden. Komponenten fassen mehrere Nodes in sich zusammen, im Workflow wird nur noch die Komponente angezeigt. Eine weitere Funktion der Komponente besteht darin das innere Abzukapseln, um die Menge an sichtbaren Variablen möglicht gering zu halten. In- und Output müssen daher explizit gesetzt werden. [Sil20] Auf dem KNIME Server können Workflows zu festen Zeitpunkten ausgeführt werden. Dies ist zum Beispiel bei den Orchestrierungen der Fall, welche jeden Morgen starten. [Kra20] Fast alle Funktionalitäten des Server können auch über eine REST API angesteuert werden. Die REST API bildet eine Schnittstelle nach außen, was anderen Programmen ermöglicht mit diesem zu interagieren und zum Beispiel Workflows zu starten.[Kha21] [Mei15] [Ber07]

2.2 Tableau

Mit der Analytics-Plattform Tableau werden Daten visualisiert dargestellt. Das Ziel ist es, so neue Informationen und Schlüsse aus den Daten zu ziehen. Durch die Visualisierung ist es auch einfacher Trends zu erkennen. In Diagrammen werden die aus einer Tabelle kommenden Daten ausgewertet. Dafür wird ausgewählt, welche Spalten oder Zeilen auf der Y- und X-Achse angezeigt werden sollen. Tableau unterstützt die Aggregation der Daten auf verschiedenen Stufen. Danach wählt Tableau die am wahrscheinlichsten passende Darstellungsart aus, zum Beispiel ein Balkendiagramm. Die Darstellungsart kann beliebig verändert werden. Eine Filterung der ausgewählten Felder ist möglich. Als Filter kommen Felder aus der eingespeisten Tabelle oder berechnete Felder und Variablen infrage. Die erstellten Diagramme werden in Blättern gegliedert und in Dashboards anzeigt. [TaboD]

2.3 ETL

Um Daten auswerten zu können, muss gewährleistet sein, dass sie korrekt sind und eine einheitliche Form haben. Ansonsten kann es bei der Datenauswertung zu Fehlinterpretationen kommen. Um dies zu vermeiden, müssen die angelieferten Daten in ETL-Prozessen transformiert werden. ETL steht für die Extraktion, die Transformation und das Laden der Daten. Der Bereich des Data Engineerings befasst sich mit allen Aufgaben, die zur Schaffung einer gesammelten und korrekten Datenbasis anfallen. Das bedeutet unter anderem den Aufbau und Betrieb von ETL-Strecken. Die Datenbasis wird im Fachjargon als Data Warehouse betitelt. Ein Data Warehouse ist ein Informationssystem, das Daten aus verschiedenen Quellen gesammelt für Auswertungen zur Verfügung stellt.[Bau13] Redshift ist eine Data Warehouse-Cloud die in der verwendeten Systemarchitektur zum Einsatz kommt. Die Daten aus den ETL-Strecken werden am Ende der Verarbeitung in Redshift integriert. [AmaoDb] [AmaoDd]
Die angelieferten Daten werden an das benötigte Format angepasst und sind nach Prozessende vollständig in das Data Warehouse eingebettet. "ETL processes constitute the backbone of a DW architecture, and hence, their performance and quality are of significant importance for the accuracy, operability, and usability of data warehouses. ETL processes involve a large variety of activities (a.k.a. stages, transformations, operations) organized as a workflow."[Kar13]

Der Ablauf einer ETL-Strecke kann sich je nach Anwendungsfall unterscheiden. Eine ETL-Strecke beginnt aber im Regelfall mit der Extraktion der Daten aus einer Datenquelle und das Laden dieser in die Staging-Tabellen. Die Staging-Tabellen, auch Landing-Zone genannt, enthalten immer nur eine Datenanlieferung. Nach einer Transformation werden die Daten in die Raw-Tabelle geladen. Diese enthalten die gesammelten gesamten Rohdaten. Die Transformation besteht daraus den Datensatz zum Beispiel auf Datenkonsistenz und Korrektheit zu überprüfen, dabei werden unter anderem Duplikate aussortiert. Daten im Data Warehouse sind konsistent, wenn die Datenanlieferungen in chronologisch korrekter Reihenfolge in das Data Warehouse integriert wurden sind. Dafür muss die vorgegebene Ausführungsreihenfolge eingehalten werden. Dies ist wichtig, da wenn Einträge aktualisiert werden garantiert werden muss, dass die Einträge nur in chronologischer Reihenfolge überschrieben werden. Nun werden die Daten noch von der Raw-Tabelle in die Refined-Tabelle geladen. In diesem Schritt werden die Daten weiter transformiert, korrigiert und in einigen Fällen auch aggregiert. Ein Beispiel für eine Transformation wäre, dass das Geburtsdatum eines Kunden in seine Altersklasse umgewandelt wird. [Hum19]

Die einzelnen Prozessschritte werden, gegliedert in Workflows, abgearbeitet. Durch die große Menge an Daten die in Data Warehouses verarbeitet werden, ist der Aufbau von zuverlässig laufenden ETL-Strecken mit einem großen Aufwand verbunden. Um die zu investierende Zeit im Tagesgeschäft so gering wie möglich zu halten, wird versucht die ETL-Prozesse möglichst intelligent und robust zu gestalten. Die Orchestrierungen der ETL-Prozesse sollen dabei helfen.

2.4 ETL-Prozess des Kundenmanagementsystems

Das Kundenmanagementsystem, kurz KMS, ist ein Bestandssystem der Deutschen Bahn. In diesem werden sowohl Stamm- als auch Bewegungsdaten gespeichert. Stammdaten sind eher statisch, sie beschreiben Daten die sich während operativen Prozessen selten verändern. Ein Beispiel für Stammdaten im KMS sind die Kunden. "Im Gegensatz zu den zustandsorientierten Stammdaten sind Bewegungsdaten abwicklungsorientiert und entstehen in Geschäftsvorfällen (z.B. Auftragseingang oder Materialentnahme)."[Leg07] Zu den Bewegungsdaten zählen in KMS unter anderem die Auftragsdaten. Interagiert ein Kunde mit der bahn.de-Website und bucht eine Zugverbindung, wird diese Interaktion über den KMS-ETL-Prozess in die KMS Auftragstabellen eingespielt. Die Daten des KMS werden in mehrere Tabellen unterteilt. Dazu gehören, neben Tabellen für die Aufträge, auch Tabellen, die die Kunden beschreiben oder Informationen über Gutscheine und Newsletter beinhalten. Das Prinzip hinter der Anlieferung ist immer das gleiche, daher wird der ETL-Prozess im Folgenden am Beispiel der Auftragsdaten beschrieben. Während des ETL-Prozesses werden die Stage-, Raw- und Refined-Tabellen mit den neuen Daten befüllt. Alle Tabellen haben ein ähnliches Schema, wobei zwischen Staging- und Raw-Bereich Spalten die keinen informativen Mehrwert bieten aussortiert werden. Die Auftragsnummer wird in allen drei Tabellen als Primärschlüssel verwendet. Neben der Auftragsnummer wird zum Beispiel der Preis, die Kundennummer und eine Beschreibung mit angeliefert. Die Beschreibung enthält im Regelfall den Start- und den Zielbahnhof der gebuchten Verbindung.

Das Ablaufdiagramm, in Abbildung 2.2, zeigt den Datenanlieferungsprozess des KMS. Die Abarbeitung des ETL-Prozesses verteilt sich auf mehrere Workflows. Es gibt je einen Workflow für die Datenanlieferung und eine erste Aufbereitung, für die Verarbeitung von der Staging-Tabelle zur Raw-Tabelle und von der Raw-Tabelle in die Refined-Tabelle. Zur Vereinfachung wurde sich bei der Darstellung auf die Auftragsdaten beschränkt. Alle weiteren angelieferten Kategorien folgen demselben Schema.

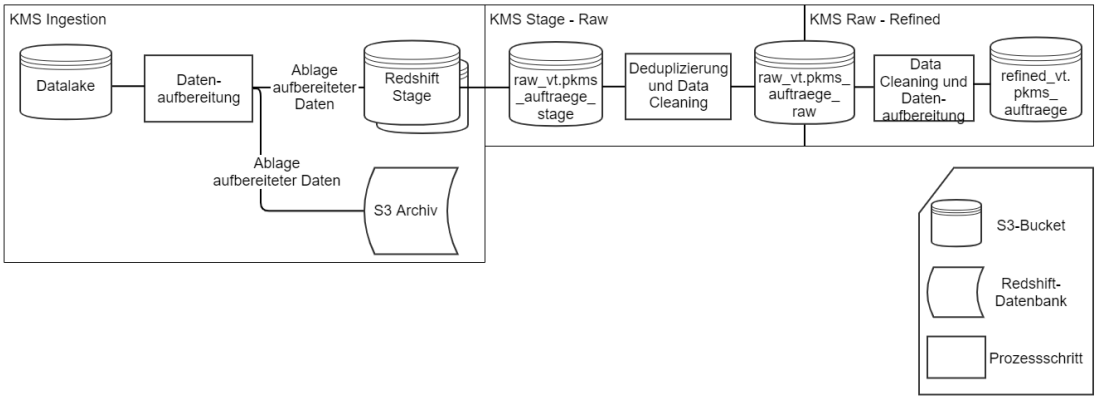

Abbildung 2.2: Ablauf des KMS ETL-Prozesses

Ingestion Der ETL-Prozess beginnt mit dem Verbindungsaufbau zum Datalake. Ein Datalake ist ein sehr großer Datenspeicher. Er enthält Daten aus unterschiedlichsten Datenquellen und nimmt Daten mit verschiedensten Strukturen auf. Dabei kann es sich um Rohdaten, aber auch um bereits bearbeitete Daten handeln.[Sin19] Im Fall von KMS sind die Daten bereits vorprozessiert. Trotzdem müssen sie noch einmal an die Anforderungen des Zielsystems angepasst werden.
Die Datenanlieferung wird aus dem Datalake extrahiert. Es handelt sich um eine Tageslieferung. Die Daten sind dabei immer am Vortag angefallen und werden optimalerweise direkt am nächsten Tag eingespeist.
Bei den Auftragsdaten beschränkt sich die erste Reformatierung auf die Zeitstempel und Daten. Danach werden die Dateien zur Archivierung im CSV-Format in S3 abgespeichert. S3 steht für "Simple Storage Service" und ist ein Objektspeicher-Service. In einem Objektspeicher wird das gesamte Objekt betrachtet und zusammenhängend gespeichert. Dies eignet sich besonders für große statische Dateien. Durch die Archivierung kann im Fehlerfall auch im Nachhinein noch auf die Daten zugegriffen werden. [AmaoDa]
Außerdem werden die Staging-Tabellen in Redshift befüllt.

Stage - Raw Im Stage-Raw Workflow werden die Daten aus dem Staging-Bereich transformiert, bereinigt und dann in die Raw-Tabelle integriert. Das Ziel der Raw-Tabelle ist es die Daten nutzbar, korrekt und vollständig zur Verfügung zu stellen. Mit KNIME werden SQL-Queries erstellt und dann in Redshift ausgeführt. Dadurch müssen die Tabellen nicht auf den KNIME-Server geladen werden. Das wirkt sich positiv auf die Performance aus.
Ein erster Bereinigungsschritt besteht darin, dass IDs herausgefiltert werden, die im Zielsystem keine Bedeutung mehr haben. Außerdem werden Nullwerte behandelt und die Felder zum Beispiel mit "-1" gekennzeichnet. Ein weiterer Bereinigungsschritt besteht in dem aussortieren der Duplikate. In der Staging-Tabelle gibt es noch keinen Primärschlüssel, über welchen ansonsten ein Duplikat definiert wird. Daher werden

Zeilen nur herausgefiltert, wenn sie in Gänze identisch sind. Als letztes wird der Bearbeitungszeitpunkt an die Tabelle angespielt. Im Fehlerfall können so alle Zeilen identifiziert werden, die in einer bestimmten Anlieferung enthalten waren.
Eine Zeile in der Staging-Tabelle kann entweder zu einem neuen Auftrag gehören oder einen bestehenden Auftrag aktualisieren. Daher werden zwei SQL-Queries formuliert. Die Update-Query verbindet über einen Inner Join die Staging- und die Raw-Tabelle. Damit werden alle Einträge zu bestehenden Auftragsnummern gefunden. In der Anfüge-Query werden die Staging und Raw-Tabelle über einen Left-Outer-Join verbunden. Danach wird die Schnittmenge zwischen beiden aussortiert. Dadurch können alle neuen Einträge identifiziert und dann angehängt werden. In einem Transaction Node werden die Queries In-Database ausgeführt. Das besondere an diesem Node ist, dass die Änderungen nur gemacht werden, wenn es keine Fehlermeldungen gab. Ansonsten werden alle Änderungen zurückgerollt. Dadurch bleibt die Datenkonsistenz, gemäß des ACID-Prinzips, gewährleistet. Dieses enthält Regeln zur Durchführung von Transaktionen in Datenbanken. "ACID steht hierbei für Atomarität, Konsistenzerhaltung, Isolation und Dauerhaftigkeit." [Kö12]

Raw - Refined In dem Raw-Refined-Workflow wird die Refined-Tabelle aktualisiert. Das Vorgehen ähnelt dabei sehr stark dem im Stage-Raw-Workflow. Die Daten werden zusätzlich bearbeitet und korrigiert. Damit wird versucht weitere Informationen aus den Daten zu erhalten und die Datenqualität weiter zu steigern. Im Fall der Auftragsdaten wird primär das Beschreibungsfeld untersucht. Aus diesem kann in vielen Fällen der Start- und Zielbahnhof geparst werden. Nach dem Erhalt der Bahnhöfe wird dann noch ein Flughafen und ein International Flag gesetzt. Der Informationsgehalt der Auftragsdaten kann so erhöht werden.
Nach erfolgreicher Integration der Daten in das Data Warehouse, können sie für Auswertungen und Visualisierungen von Data Scientists verwendet werden.

2.5 Datenqualität

Die im Data Warehouse enthaltenden Daten werden für weitere Auswertungen genutzt und auf Basis der Ergebnisse werden viele Entscheidungen getroffen. Daher ist es wichtig die Datenqualität der Daten zu kennen und zu verbessern. Auf die Frage was Datenqualität ist und wie man sie misst, gibt es allerdings verschiedenste Antworten. Dies liegt daran, dass Daten sehr heterogen sind und sich in ihren Anwendungszwecken stark unterscheiden. Die Entscheidung ab wann Daten qualitativ sind, ist immer relativ und schwer zu prüfen. [Lei08] Um den Begriff trotzdessen besser einzurahmen wird Datenqualität oft an ihrer "fitness for use" gemessen. Nach diesem Konzept, sind Daten qualitativ sobald sie bereit zur Nutzung durch ihre Nutzer sind. [Wan96] [Har10] In der

zuvor beschriebenen Datenverarbeitung ist dieser Punkt erreicht, wenn die Daten in die Refined-Tabellen integriert wurden sind.
Zusätzlich zu dem "fitness for use"-Konzept wurden Qualitätsmerkmale zur Einordnung der Datenqualität erarbeitet. Die Qualitätsmerkmale sind in der folgenden Tabelle abgebildet. [Nau07] Zusätzlich enthält die Tabelle eine Spalte mit selbst herausgearbeiteten Fehlern, die die Datenqualität während der Datenverarbeitung negativ beeinflussen.
Die intrinsische Datenqualität beschreibt primär die Qualität eines einzelnen Werts, also zum Beispiel einer Kundennummer. Werte mit einer hohen intrinsischen Datenqualität, sollten korrekt, genau und objektiv sein, das steigert auch die Glaubhaftigkeit. Fehler, die die intrinsiche Datenqualität verschlechtern sind zum Beispiel falsche Werte, kryptische Werte, unbrauchbare und redundante Daten. Redundanzen können auch durch Mehrfacheinspielungen von Daten verursacht werden.
Die kontextuelle Datenqualität bewertet die Daten im Gesamten und im Bezug zueinander. Ein Aspekt der kontextuellen Datenqualität ist die Vollständigkeit, sie wird durch fehlende Werte gemindert. Probleme während der Datenverarbeitung wie fehlerhafte Dateien oder Verbindungsprobleme mit dem Quellsystem haben einen negativen Einfluss auf die Aktualität der Daten.
Die dritte Kategorie beschreibt die repräsentationelle Datenqualität. Diese beschreibt Qualitätsmerkmale, die aus Sicht des Konsumenten auf die Daten bedeutend sind. Auch auf diese Kategorie haben redundante Daten und Mehrfacheinspielungen einen negativen Effekt, da sie die Interpretierbarkeit vermindern. Vor allem aber schematische Änderungen der Daten sind auftretende Fehler, welche die konsistente Darstellung der Daten gefährden.
Die Zugriffsqualität umfasst die Merkmale der Verfügbarkeit und der Zugriffssicherheit. Die ETL-Anlieferungen beeinflussen dieses Qualitätsmerkmal kaum.
Die genannten Fehlerarten können in zwei Kategorien gegliedert werden. Die erste

Tabelle 2.1: Merkmale der Datenqualität mit den beeinflussenden Fehlern

Datenqualität	Qualitätsmerkmal	Fehler
intrinsische Datenqualität	Richtigkeit, Glaubhaftigkeit, Genauigkeit, Objektivität, Reputation	redundante Daten, falsche Werte, kryptische Werte, unbrauchbare Daten, Mehrfacheinspielungen
kontextuelle Datenqualität	Vollständigkeit, Aktualität, Mehrwert, Relevanz, Datenmenge	fehlende Werte, Quellsystem nicht erreichbar, fehlerhafte Dateien
repräsentationelle Datenqualität	Konsistenz der Darstellung, Interpretierbarkeit, Verständlichkeit, Knappheit der Darstellung	schematische Änderung der Daten, Mehrfacheinspielungen, redundante Daten
Zugriffsqualität	Verfügbarkeit, Zugriffssicherheit	

umfasst Systemfehler. Systemfehler beeinflussen nicht nur einen Wert, sondern eine gesamte Datenanlieferung. Zu den Systemfehlern gehören fehlerhafte Dateien, Mehrfacheinspielungen, die schematische Änderung der Daten und Verbindungsprobleme zum Quellsystem.
Die zweite Kategorie besteht aus inhaltlichen Fehlern. Diese haben ihre Ursprünge meistens schon früher in der Verarbeitungskette. Es muss versucht werden diese zu finden, um die Datenqualität zu steigern. Inhaltliche Fehler sind fehlende, falsche oder kryptische Werte sowie redundante und unbrauchbare Daten.

3 Konzept

In diesem Kapitel werden zur Erreichung der Zielsetzung konkrete Vorgaben und Pläne erarbeitet. Es werden Anforderungen an Orchestrierungen und ihr Monitoring festgelegt. Außerdem wird die 1-Click-Orchestrierung, die selbst entwickelte Orchestrierung in KNIME, mit Apache Airflow verglichen. Der Vergleich erfolgt mit Hilfe der zuvor definierten Anforderungen.

3.1 Anforderungen

In diesem Abschnitt werden die Anforderungen, die an die zu entwickelnde Orchestrierung und das Monitoring gestellt werden, vorgestellt. Die Wichtigkeit der einzelnen Anforderungen ist in den Tabellen absteigend sortiert. Die erste Tabelle umfasst die grundlegenden Anforderung zur Entwicklung einer funktionierenden Orchestrierung. Danach folgt eine Tabelle mit zusätzlichen Anforderungen um den Automatisierungsgrad zu erhöhen. Die dritte Tabelle enthält die Anforderungen an das Monitoring.

Tabelle 3.1: Allgemeine Anforderungen an eine Orchestrierung

Anforderung	Beschreibung
Kaum Interaktionen im Tagesgeschäft	Eine Orchestrierung sollte geplant anlaufen und wenn keine Fehler auftreten ihre Ausführung ohne zusätzliche Interaktion abschließen.
Überprüfung von Ausführungsbedingungen	Abhängigkeiten der Workflows müssen definiert und während der Ausführung überprüft werden. Dazu gehören Abhängigkeiten innerhalb der Workflows, aber auch von anderen Prozessen im Data Warehouse.
Integrierbar in die Systemarchitektur	Die Anzahl der verwendeten Systeme ist möglichst klein zu halten. Dies erhöht die Stabilität und mindert den Wartungsaufwand.
Einfach in der Handhabung	Die Schnittstellen zum Benutzer sollten verständlich und intuitiv gestaltet sein. Es muss Möglichkeiten zum Erstellen, Verwalten und Überwachen der ausgeführten Prozesse geben. Dies kann über eine grafische Benutzeroberfläche realisiert werden.

Tabelle 3.2: Zusätzliche Anforderungen an eine Orchestrierung mit hohem Automatisierungsgrad

Anforderung	Beschreibung
Flexible Ausführung der Workflows	Um jeden Ausführungsfall abzudecken, sollte eine Orchestrierung neben Tagesausführungen auch Mehrtages- und Teilausführungen realisieren. Auch eine Möglichkeit zur Einzelauswahl von Workflows ist wünschenswert.
Sinnvolle Interaktionen im Fehlerfall	Es ist wichtig, über Fehler detailliert zu informieren und den händischen Aufwand bei der Reprozessierung so klein wie möglich zu halten.
Sicherstellung der Datenkonsistenz	Der Worst-Case bei der Datenverarbeitung besteht darin, die Daten zu verfälschen. Es muss vor jeder Ausführung überprüft werden, ob alle Ausführungsvoraussetzungen eingehalten worden sind.
Wiederverwendbar	Eine Orchestrierung sollte verschiedenste Datenanlieferungsprozesse steuern können.
Performante Ausführung der Workflows	Unter Berücksichtigung der Ausführungsbedingungen ist eine parallelisierte Ausführung vorteilhaft.

Tabelle 3.3: Anforderungen an ein Monitoring

Anforderung	Beschreibung
Schaffung von Transparenz	Ein Monitoring soll neue Einblicke in den Prozessablauf geben. Dafür müssen Veränderungen in den Daten nach jedem Prozessschritt gezählt und als Metriken dokumentiert werden.
Erhöhung der Vergleichbarkeit	Die gesammelten Metriken müssen aufbereitet werden und dem Nutzer in verständlicher Form zur Verfügung gestellt werden. Ein ähnliches Schema dient der besseren Vergleichbarkeit. Die Datenbasis, mit der Vergleiche angestellt werden können, sollte möglichst groß sein.
Integrierbar in die Systemarchitektur	Die Anzahl der verwendeten Systeme ist möglichst klein zu halten. Dies erhöht die Stabilität und mindert den Wartungsaufwand.
Prognosen erstellt auf Basis der historischen Daten	Viele Datensätze enthalten sich wiederholende Muster, die auch Aufschluss über die Korrektheit der Daten geben können. Eine Erstellung von Prognosen kann beim Verständnis dieser Muster helfen.
Monitoring der Laufzeiten	Die Laufzeiten der Workflows können Hinweise über den Verarbeitungserfolg und die Datenmenge geben. Eine außergewöhnlich lange Verarbeitungszeit gibt Anlass zu Nachforschungen.

3.2 Orchestrierung

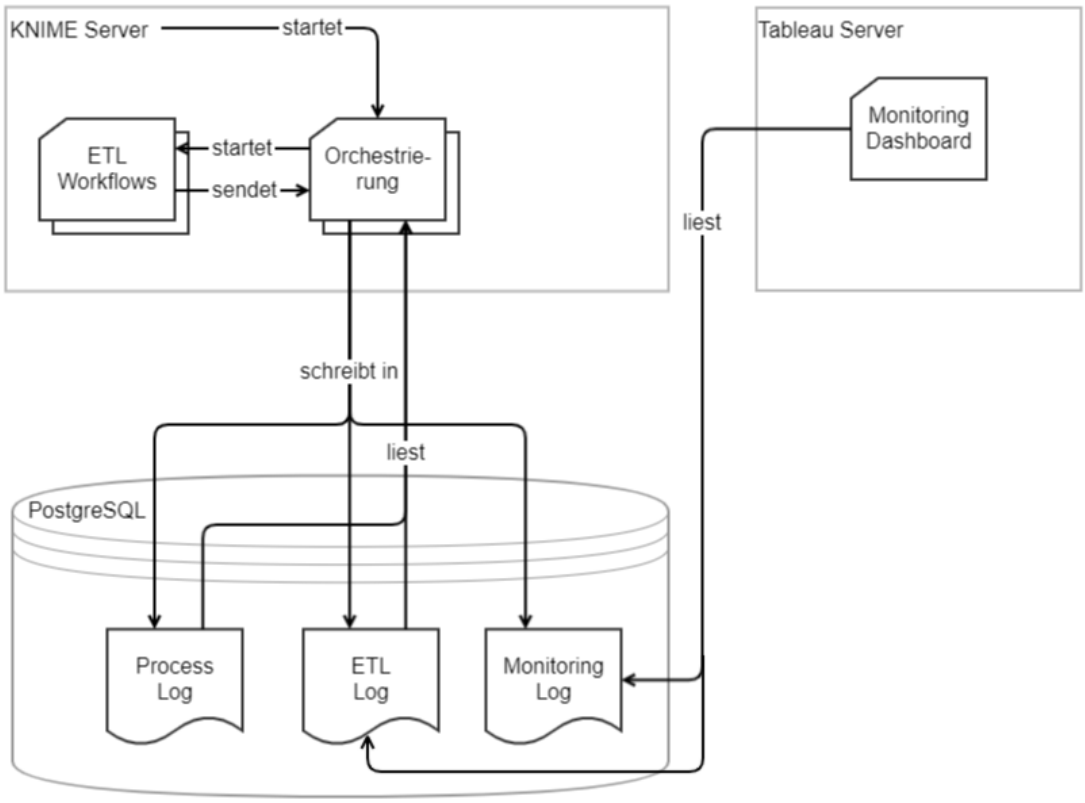

Abbildung 3.1: Architekturdiagramm der Komponenten der Orchestrierungen

In den nächsten beiden Abschnitten wird auf die geplante Umsetzung von grundlegenden Vorgaben zur Orchestrierung eingegangen.

Vorgabe: An bestehende Systeminfrastrukur anknüpfen: Von Anfang an sollte die Orchestrierung darauf ausgerichtet werden die bestehende Systeminfrastruktur mitzubenutzen. Da alle ETL-Prozesse in KNIME laufen, besteht die Anforderung auch die Orchestrierung mit KNIME zu entwickeln. Als Logging-Datenbank wurde historisch bereits PostgreSQL genutzt. Selbiges gilt für die Entwicklung des Monitorings mit Tableau. In Abbildung 3.1 ist das Zusammenspiel der Orchestrierung mit den verschiedenen Komponenten abgebildet. Aus diesem ist erkennbar, dass die Orchestrierungen vom KNIME Server gestartet werden. Der KNIME-Server übernimmt die Rolle des Schedulers. Ein Scheduler weist die benötigten Ressourcen zum Ausführen der Aufgaben zu und startet danach den Workflow. Er führt auch alle ETL-Prozesse aus, die über die Orchestrierung gestartet werden. Die Orchestrierung wiederum liest dann den ETL-Log aus, in diesem Log befinden sich die Einträge zu den Ausführungen der Workflows. Mit Hilfe dieses Logs wird in der Orchestrierung ein Ausführungsplan erstellt. Entsprechend des Plans werden die ETL-Workflows gestartet. Diese senden dann Verarbeitungsinformationen zurück an die Orchestrierung, welche die Informationen in die Logs überträgt. Nachdem

alle Workflows der Orchestrierung für einen Tag aufgerufen wurden sind, wird der Process-Log geschrieben. In diesem wird täglich je ein Eintrag für den Ausführungsstand der Orchestrierungen ergänzt. Den Process-Log gibt es schon länger und er dient einer schnellen Übersicht über die Ausführungen. Die Orchestrierungskomponente hat das Schreiben des Logs nur übernommen. Im dritten Log, dem Monitoring-Log, werden Informationen, die in den Workflows über die verarbeiteten Daten gesammelt werden, festgehalten. Im Monitoring-Dashboard, welches auf dem Tableau-Server liegt, werden die geloggten Metriken angezeigt und ausgewertet.

Vorgabe: ETL-Prozesse täglich ausführen: Die ETL-Prozesse laufen täglich, daher besteht die Vorgabe, dass die Orchestrierung diese täglich aufruft. Um nachvollziehen zu können, dass die Workflows tatsächlich täglich ausgeführt werden, fällt die Entscheidung die Ausführungsreihenfolge mittels eines Logs zu dokumentieren (ETL-Log in 3.1). Der Primärschlüssel bildet sich aus dem Orchestrierungsnamen, der Workflow-ID und dem Datum. Dies forciert die Anforderung, dass es jeden Tag nur genau einen Eintrag pro Workflow geben darf. Bei einer Mehrfachausführung wird der Eintrag aktualisiert. Das Schreiben der Logs übernimmt die Orchestrierungskomponente. So kann sichergestellt werden, dass für alle geplant aufzurufenden Workflows Logeinträge verfasst werden. Würden die Workflows selbst in die Logging-Tabelle schreiben, würden die Einträge im Fehlerfall nicht dokumentiert werden, da der Workflow nicht vollständig ausgeführt wurden wäre. Ein weiterer Vorteil ist die striktere Trennung zwischen Abarbeitung der Datenanlieferung und dem Orchestrierungsprozess.

Bevor mit der Planung zur Umsetzung der Anforderungen an die Orchestrierung begonnen werden kann, müssen zusätzlich noch einige Implementierungsentscheidungen getroffen werden.

Implementierungsentscheidung: Komponente oder Workflow: Die Implementierung der Orchestrierungen soll zur Übersichtlichkeit von der Konfiguration getrennt sein. Dafür kann entweder ein neuer Workflow genutzt werden oder die Nodes, welche die Funktionalität abbilden, in einer Komponente gebündelt werden. Eine Komponente kann wie ein Template vervielfacht und dann als unveränderliche Kopie ausgeführt werden. Beide Varianten können auf dem KNIME-Server abgelegt und geteilt werden. Die Benutzung eines zweiten Workflows würde durch einen Call-Workflow-Node in der Orchestrierung gelöst werden. Dieser Node ruft dann mit einer Konfigurationstabelle den Orchestrierungsworkflow auf. Dies ist etwas umständlich und führt dazu, dass als Konfiguration nur eine Tabelle, aber keine Variablen übergeben werden können. Bei der zweiten Lösungsmöglichkeit mit einer Komponente gestaltet sich das Einreichen von Variablen und Tabellen unproblematisch, somit kann das Format, in dem Daten

in die Komponente eingereicht werden, frei gewählt werden. Während bei der Lösung mit einem zweiten Workflow, liegend auf dem KNIME-Server, dieser Workflow jeweils aufgerufen und eine Instanz von ihm gestartet wird, können Komponenten verlinkt werden. Dabei wird eine Referenz auf das Original auf dem KNIME-Server erstellt, die Komponente liegt aber als Kopie mit im Orchestrierungsworkflow. Sie wird in diesem mit ausgeführt, kann aber im Workflow nicht verändert werden. Aufgrund der höheren Flexibilität bei Eingaben und einer Ausführung mit in den Orchestrierungsworkflows, wird die Funktionalität in einer Komponente eingebettet.

Implementierungsentscheidung: Sortierung der Workflows: Die Workflows, welche von der Orchestrierung gesteuert werden, sind abhängig voneinander. In der Konfigurationstabelle werden diese Abhängigkeiten angegeben, indem jeweils die Workflow-IDs der Workflows, von denen ein Workflow abhängig ist, aufgelistet werden (siehe Abbildung 4.1). Die Workflows sollen in der richtigen Reihenfolge aufgerufen werden. Es müssen alle Workflows ausgeführt worden sein, von denen ein Workflow abhängig ist, bevor dieser aufgerufen werden darf. Die daraus entstehende Verkettung beschreibt einen gerichteten Graphen. Gerichtete Graphen haben immer mindestens eine Wurzel und bestehen aus Knoten und Kanten. Die Kanten (vorausgesetzter Workflow zu Workflow) besitzen seine Richtung und verbinden die Knoten (Workflows) miteinander. Die Workflows müssen entsprechend ihrer Abhängigkeiten im Graphen sortiert werden. Eine Sortierung, bei der die Abhängigkeiten eingehalten werden, ist die topologische Sortierung. Der zu sortierende Graph kann nicht sortiert werden, wenn er einen Kreis enthält. In diesem Fall kann die Anforderung, dass alle Vorgänger ausgeführt worden sein müssen, nicht eingehalten werden. Damit die Workflows topologisch sortiert werden können, muss es sich also um einen azyklischen gerichteten Graphen (DAG) handeln, dies ist vor der Sortierung zu prüfen. Nach der Sortierung ist eine Ausführungsreihenfolge erstellt und es kann mit der Ausführung begonnen werden. [Har15]

3.3 Monitoring

Dadurch, dass mit der Orchestrierungskomponente eine generalisierte Ablaufsteuerung vorhanden ist, ist es machbar, die ETL-Prozesse mit einem Monitoring auszustatten. In einem ETL-Prozess ist es wichtig Monitoring-Metriken zu sammeln. Die Transparenz des ETL-Prozesses steigt und es wird sich erhofft, dass Fehler schneller erkannt und dann beseitigt werden können. Anhand der Metriken kann eine "normale" Datenanlieferung definiert werden. Erst danach können Abweichungen vom Normalbetrieb erkannt werden. Das Ziel hinter dem Monitoring ist es, die Datenqualität zu steigern.

Implementierungsentscheidung: Auswahl von Metriken: Zuerst muss sich in Absprache mit dem Team darauf festgelegt werden, welche Metriken gesammelt werden sollen. Die Metriken sollen die Veränderungen in und zwischen den Stage-, Raw- und Refined-Tabellen abbilden. Dafür müssen als Erstes die Veränderungen identifiziert werden. Da ETL-Prozesse abgearbeitet werden, sind die Prozessschritte immer ähnlich. Die Datenanlieferungen werden zuerst auf Duplikate untersucht und diese dann herausgefiltert. Danach wird die Anlieferung in das Data Warehouse eingepflegt. Es werden neue Einträge eingefügt (Inserts) und bestehende Einträge aktualisiert (Updates). Nach jedem Prozessschritt sollen die entsprechenden Metriken gesammelt werden. Neben diesen Metriken sind zusätzlich die Veränderungen der Raw- und Refined-Tabellen interessant. Daher werden die Raw- und Refined-Tabellen vor und nach dem Einfügen gezählt. Da die Staging-Tabelle immer nur eine Datenlieferung enthält, werden von dieser keine Tabellenmetriken benötigt.
Bei den Workflows, die in die Staging-Tabellen schreiben, werden also die Zeilen der Download-Tabelle gezählt, dann die Duplikate und die Inserts in die Staging-Tabelle. Bei den Workflows, die in die Raw-Tabellen oder die Refined-Tabellen schreiben, sind die Inserts, die Updates, die Duplikate und die Eintragsanzahl der Raw- und Refined-Tabellen vor und nach dem Einfügen interessant. Nach Duplikaten wird in den Raw- und Refined-Tabellen gesucht, da Redshift Primärschlüssel aus Performance Gründen nicht forciert. [AmaoDc]
Mit den Metriken können einige Informationen über die Orchestrierungen gesammelt werden. Dazu gehören die Veränderungen der Inserts und Updates auf Tagesbasis. Sind die Inserts und Updates in die Raw-Tabelle ungleich den Inserts ins Staging, weist dies auf einen Datenverlust hin. Außerdem kann ein erhöhtes Datenvolumen zum Beispiel auf viele Duplikate hinweisen. Ein deutlich verringertes Datenvolumen gibt Anlass zur Fehlersuche im Quellsystem.

Implementierungsentscheidung: Integration der Abfragen in die ETL-Workflows: Da die Metriken zwischen den Prozessierungsschritten gesammelt werden müssen, müssen die SQL-Abfragen zum Erhalt der Metriken in die Workflows integriert werden. Dies soll möglichst unauffällig passieren, damit der eigentliche Zweck des Workflows klar ersichtlich bleibt. In die Workflows wird daher jeweils eine "Pre-Execution-Komponente" und ein "Post-Execution-Komponente" eingefügt, in denen die Nodes gebündelt liegen. Um eine möglichst flexible Struktur zu gewährleisten, gibt der Workflow am Ende eine Tabelle mit den gesammelten Metriken aus. Diese Tabelle soll immer das gleiche Format haben, aber prinzipiell verschiedenste Metriken abbilden können. Daher fällt die Entscheidung auf eine Spalte mit Keys und eine Spalte mit den zugehörigen Values. Die Tabelle wird dann in die Write-Komponente der Orchestrierung gereicht und im Monitoring-Log (Beispiel in Abbildung 4.6) ergänzt.

Implementierungsentscheidung: Dashboards: Um die Metriken zu visualisieren, wird ein Dashboard mit Tableau erstellt. Das Dashboard stellt die Informationen, die aus den Metriken gewonnen werden können, möglichst gut vergleichbar dar. Es gibt eine Seite zur Übersicht über die Ausführungshistorie mit Ausführungsstatus der Workflows. Außerdem werden die gesammelten Metriken pro Workflow angezeigt. Dies umfasst die Inserts, Updates und Duplikate auf die bearbeitete Tabelle. Eine dritte Übersicht zeigt die Einfügemengen in die Tabellen für je ein Subtopic. Ein Subtopic umfasst die Workflows, die je eine Strecke von Staging-Tabelle zu Raw-Tabelle zur Refined-Tabelle bearbeiten. Die Subtopics müssen dafür in der Konfigurationstabelle festgelegt worden sein. Bei allen Ansichten ist die betrachtete Zeitperiode einstellbar.

3.4 Alternativimplementierung mit Apache Airflow

Eine Möglichkeit, sich den Aufwand einer Eigenentwicklung zu sparen, besteht im Verwenden von kommerziell verfügbarer Software. Zur Abwägung, ob sich eine Eigenentwicklung lohnt, wird zusätzlich eine Orchestrierung mit Apache Airflow in Betracht gezogen. In diesem Unterkapitel wird Apache Airflow, näher vorgestellt und danach mit der selbst entwickelten Lösung verglichen.

Airflow ist eine Software zur Orchestrierung von Prozessen. Mit Airflow können Work-

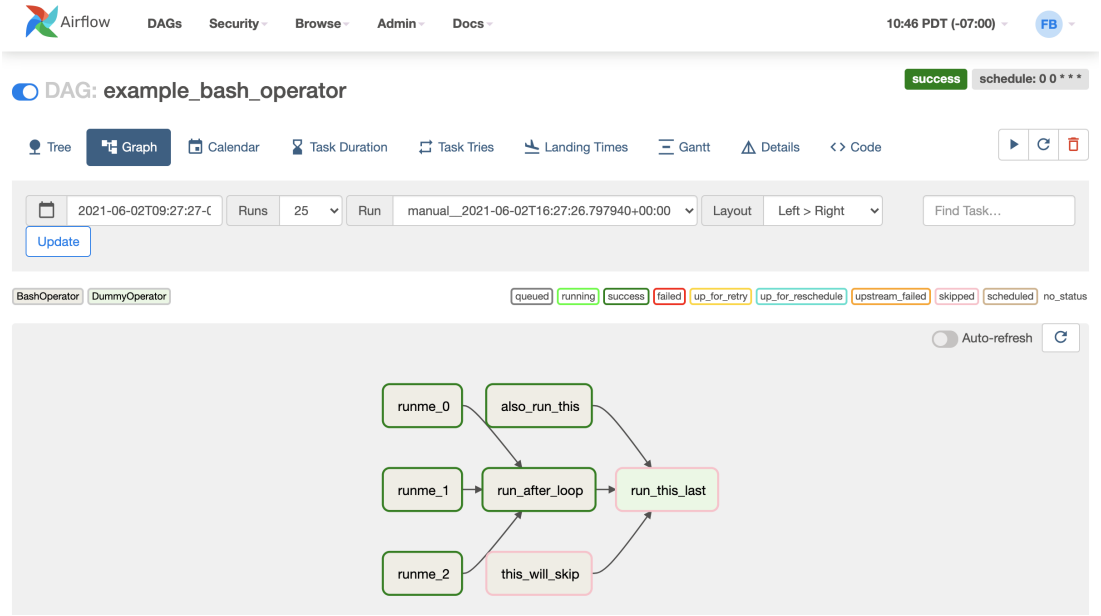

Abbildung 3.2: User Interface von Airflow[AiroDi]

flows als Tasks angelegt, verwaltet und überwacht werden. Dafür wird eine grafische Verwaltungsoberfläche zur Verfügung gestellt. Airflow speichert, ähnlich wie auch die Orchestrierungskomponete, Prozesse mit ihren Tasks als gerichtete azyklische Graphen ab. Ein Knoten stellt eine Task dar. Im UI von Airflow, sichtbar in Abbildung 3.2,

werden die DAGs angezeigt. Dort kann der Ausführungsstatus der einzelnen Tasks, sowie zum Beispiel die Ausführungsdauer, eingesehen werden. Bevor die Tasks aber ausgeführt werden können, müssen sie in Python-Scripten erstellt und spezifiziert werden. Der große Vorteil dabei liegt darin, dass sämtliche Python-Funktionen zur Erstellung von Workflows genutzt werden können. Dadurch können auch komplexe und dynamische Workflows realisiert werden. [AiroDa]

Airflow besteht aus mehreren Komponenten, welche in Abbildung 3.3 dargestellt sind.

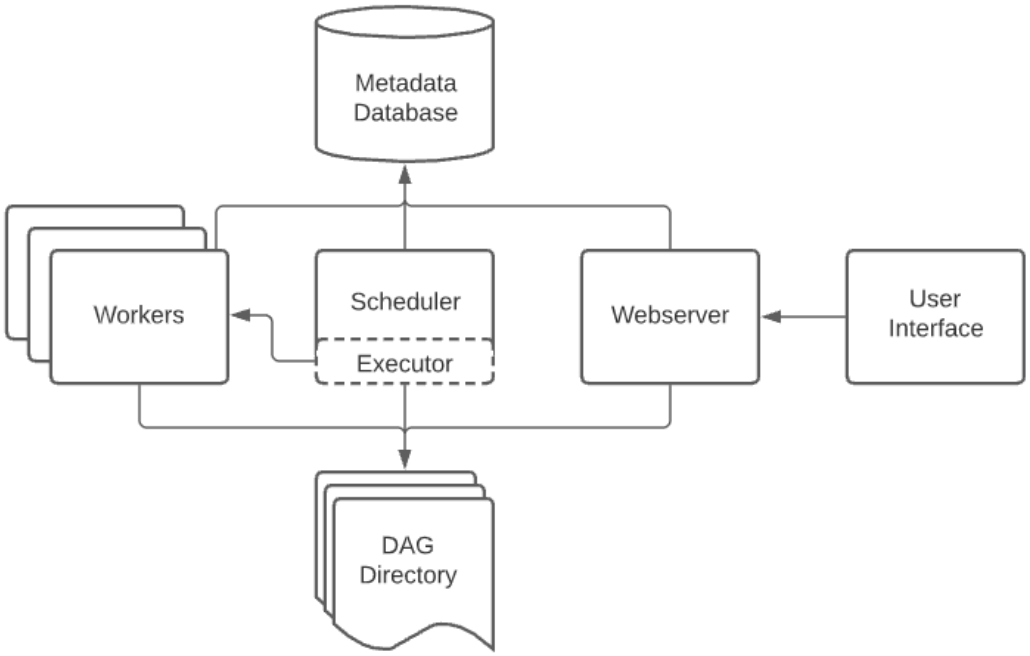

Abbildung 3.3: Systemarchitektur von Airflow [AiroDb]

Der Scheduler startet geschedulte Tasks und überträgt diese an den Executer. Dieser führt den Programmcode aus. Es gibt verschiedene Executer die genutzt werden können, prinzipiell sind sie aber alle für das Ausführen der Tasks verantwortlich. Mit dem UI, welches auf dem Webserver läuft, können die DAGs gestartet und verwaltet werden. Beim Auftreten von Fehlern während der Prozessierung können über das UI schnell nähere Informationen eingeholt werden. Die Konfigurationsdateien für die Tasks und DAGs werden in einem Ordner, dem DAG Directory, gesammelt abgelegt. In der Metadata Database können von den Komponenten anfallende Metadaten gespeichert werden. [AiroDb]

Um Apache Airflow in die bestehende Systemarchitektur zu integrieren, kann die REST API des KNIME-Servers genutzt werden. Mit dieser können Workflows gestartet und Ausführungsergebnisse übertragen werden. Die Workflows werden dann über Python-Scripte von Airflow gestartet werden. Die Python-Scripte enthalten alle benötigten Informationen, zum Beispiel Ausführungsbedingungen, zum DAG. Zusätzlich kann

eine E-Mail Adresse zur Benachrichtigung im Fehlerfall angegeben werden. Auch die Anzahl an Ausführungsversuchen, das Startdatum und der Ausführungsintervall sind konfigurierbar. Zum Anlegen der Workflows können Tasks genutzt werden, sie sind die Knoten des DAGs. Ebenfalls in den Python-Files, kann die Ansteuerung der REST API des KNIME-Servers stattfinden. Auch Abhängigkeiten zwischen den Tasks können festgelegt werden. Die erstellten DAGs können über das UI geprüft und verwaltet werden. [AiroDh] [Kha21] [Mei15]

Airflow selbst sammelt Metriken über die Tasks, wie zum Beispiel die Ausführungszeiten. Für dieses Taskmonitoring gibt es eine Visualisierung in dem UI. Abgesehen davon können mit Airflow keine Metriken über die Abläufe in den Workflows gespeichert und visualisiert werden. Stattdessen empfiehlt Airflow die Verwendung von zusätzlicher Software. StatsD, ein Netzwerkdienst, wird empfohlen zum Zusammentragen der Metriken. Über statsD können die Metriken an Prometheus gesendet und dort gespeichert werden. Zur Visualisierung kann Grafana genutzt werden. Ist diese Verarbeitungspipeline eingerichtet, können die gesammelten Metriken auf einer Website eingesehen werden. [AiroDg] [ProoD] Da im Architekturstack bereits Tableau verwendet wird, würde auch in Kombination mit Airflow Tableau als Monitoring-Plattform genutzt werden. Tableau bietet zusätzlich mehr Flexibilität beim Erstellen von Dashboards als Grafana.

3.5 Vergleich der 1-Click-Orchestrierung mit Apache Airflow

In diesem Unterkapitel soll die Frage geklärt werden, welche der vorgestellten Lösungen verwendet werden soll. Dafür wird jeweils bewertet wie gut die Anforderungen umgesetzt wurden sind. Diese Einordnung wird dann für jede Anforderung noch einmal erklärt und verglichen.

Tabelle 3.4: Bewertung der Orchestrierungen anhand der gewichteten Anforderungen

Anforderung	1-Click-Orchestrierung	Airflow	Gewichtung
Kaum Interaktionen im Tagesgeschäft	5 / 5	5 / 5	20%
Überprüfung von Ausführungsbedingungen	4 / 5	5 / 5	20%
Integrierbar in die Systemarchitektur	5 / 5	0 / 5	15%
Einfach in der Handhabung	5 / 5	4 / 5	10%
Flexible Ausführung der Workflows	3 / 5	5 / 5	10%
Sinnvolle Interaktionen im Fehlerfall	4 / 5	5 / 5	10%
Sicherstellung der Datenkonsistenz	5 / 5	5 / 5	5%
Wiederverwendbar	5 / 5	5 / 5	5%
Performante Ausführung der Workflows	2 / 5	5 / 5	5%
Gesamtpunktzahl	4,35 / 5	4,15 / 5	

Tabelle 3.5: Bewertung der Monitoringlösungen anhand der gewichteten Anforderungen

Anforderung	1-Click-Orchestrierung + Tableau	Airflow + Tableau	Gewichtung
Schaffung von Transparenz	5 / 5	5 / 5	25%
Erhöhung der Vergleichbarkeit	5 / 5	5 / 5	25%
Integrierbar in die Systemarchitektur	5 / 5	4 / 5	20%
Prognosen	0 / 5	0 / 5	20%
Monitoring der Laufzeiten	0 / 5	4 / 5	10%
Gesamtpunktzahl	3,5 / 5	3,7 / 5	

Erklärung der vorgenommenen Gewichtung: Die Gewichtung der einzelnen Anforderungen ergibt sich aus der Wichtigkeit der Anforderung für das Projekt und der Komplexität der Umsetzung der Anforderung. Die grundlegenden Anforderungen für eine funktionierende Orchestrierung (Kaum Interaktionen im Tagesgeschäft, Überprüfung von Ausführungsbedingungen) werden am höchsten gewichtet, da sie das Grundgerüst der Orchestrierung beinhalten und daher sehr viel Umsetzungszeit benötigen.
Mit 15 Prozent ist das nächstwichtige Feature die Integration in die Systemarchitektur. Da die ETL-Prozesse in KNIME laufen und implementiert sind, ist es wichtig eine Orchestrierung zu nutzen die ebenfalls mit KNIME interagieren kann. Dies ist eine spezifische Anforderung und die Integration von systemfremden Orchestrierungen kann sehr zeitaufwändig sein. Die Orchestrierungen sollen Arbeit abnehmen und stetig funktionieren, das kann leichter garantiert werden, wenn die verwendeten Komponenten gut zusammenspielen. Das macht ein System zusätzlich auch robuster. Es ist also von großer Bedeutung, inwiefern die Orchestrierung integrierbar ist, da hiervon der erforderliche Projektaufwand abhängt.
Stufenweise werden die Anforderungen jeweils fünf Prozent weniger gewichtet. Die mit zehn Prozent gewichteten Anforderungen sind trotzdem wichtige Features einer Orchestrierung. Aber auch mit relativ wenig Arbeitsaufwand können bei diesen Features ein paar Mindestanforderungen, wie die Weiterleitung der Fehlermeldungen per E-Mail, umgesetzt werden.
Die letzten drei Anforderungen werden jeweils mit fünf Prozent gewichtet. Bei den Anforderungen zur Sicherstellung der Datenkonsistenz und der Wiederverwendbarkeit ist die Umsetzung wenig komplex, daher die niedrige Gewichtung. Anders sieht es bei einer performanten Ausführung aus. Diese ist anspruchsvoll umzusetzen, wird aber in dem bearbeiteten Projekt nicht dringend benötigt und daher auch nicht hoch gewichtet.

In der zweiten Tabelle zum Monitoring verteilen sich die Gewichtungen nach dem gleichen Prinzip. Die am höchsten gewichteten Anforderungen umfassen das Grundgerüst zur Erstellung eines Monitorings und sind daher besonders wichtig und in der Umsetzung zeitintensiv. Die Integrierbarkeit in die Systemarchitektur ist weiterhin eine für das

Projekt essenzielle Anforderung, da das Monitoring, wie auch die Orchestrierung, in eine bestehende Architektur eingefügt werden sollen. Gleich gewichtet mit der Integrierbarkeit werden Prognosen auf Basis von historischen Daten. Das Monitoring funktioniert ohne Prognosen immer nur im Zusammenspiel mit einer Beurteilung der Metriken durch Mitarbeiter. Da das Gesamtsystem aber möglichst automatisiert funktionieren soll, ist es wichtig nicht bei der Einordnung der Metriken aufzuhören. Die Anforderungen zur Schaffung des Grundgerüsts sind trotzdem noch höher gewichtet, da erst Messdaten aus der Vergangenheit zur Verfügung stehen müssen, bevor Prognosen erstellt werden können. Daraus ergibt sich eine Gewichtung mit 20 Prozent. Auf Grund der schnellen Umsetzbarkeit wurde das Monitoring von Laufzeiten nur mit zehn Prozent gewichtet.

Die Bewertungen der einzelnen Anforderungen werden im Folgenden begründet.

Kaum Interaktionen im Tagesgeschäft: Die 1-Click-Orchestrierungen laufen täglich zu einer festen Uhrzeit an. Das Starten übernimmt der KNIME-Server, welcher die Orchestrierungen als Jobs anlegt und ausführt. Da durch den ETL-Log der letzte Ausführungsstand bekannt ist, erstellt die Orchestrierung einen Ausführungsplan mit allen fehlenden Ausführungen bis zum maximalen Ausführungstag. Dieser wird als Input mit in die Orchestrierungskomponente eingegeben und entspricht bei einer geschedulten Ausführung dem Ausführungstag.
In Airflow wurde die Startzeit während der Konfiguration des DAGs festgelegt. Mit der Angabe "catchup" im Konfigurationsfile kann die Nachziehen von fehlenden Ausführungen bis zum maximalen Ausführungstag aktiviert werden. Neben diesen grundlegenden Einstellungen bietet Airflow noch viele weitere Überprüfungs- und Automatisierungsfunktionen, die ein händisches Eingreifen ersetzen. [AiroDh] Beide Lösungen erfüllen diese Anforderung vollständig.

Überprüfung von Ausführungsbedingungen: Die 1-Click-Orchestrierung kann unterschiedliche Ausführungsbedingungen prüfen. Dazu gehört die Prüfung der Einhaltung der Ausführungsreihenfolge. Dafür werden vor dem Workflowaufruf die Ausführungsergebnisse der vorausgesetzten Ausführungen im Log überprüft. Die Definition der Abhängigkeiten erfolgt über eine Angabe der vorausgesetzten Workflow-IDs in der Eingabetabelle. Diese ist in Abbildung 4.1 abgebildet. Abhängigkeiten zu anderen Orchestrierungen können über eine Variable angegeben werden. Identifiziert werden die Orchestrierungen über ihr Topic, welches ein die Orchestrierung beschreibender String ist. Neben den Voraussetzungen vom Ausführungstag können auch Vorraussetzungen vom vorherigen Ausführungstag überprüft werden. Es besteht die Möglichkeit, Voraussetzungen in drei Abstufungen zu definieren. Das Ziel, was mit dieser Prüfung verfolgt wird, ist die Sicherstellung der Datenkonsistenz. Da aber verschiedene Orchestrierungen unterschiedliche Anforderungen zur Erhaltung der Datenkonsistenz haben, gibt es diese drei Stufen. Es kann festgelegt

werden, dass alle Workflows der Orchestrierung am vorherigen Tag erfolgreich ausgeführt werden müssen. Etwas weniger strikt ist eine Ausführungserlaubnis bei erfolgreicher Ausführung desselben Workflows am vorherigen Tag. Auch eine Ausführungserlaubnis unabhängig von vorherigen Status kann erteilt werden.
Die eben schon genannten Ausführungsbedingungen können ebenfalls von Airflow überprüft werden, auch wenn die Implementierung etwas anders aussieht. Die Angabe erfolgt hier im Konfigurationsfile über die jeweiligen Flags. Mit dem Flag "depends_on_past" wird angegeben, dass die vorherige Ausführung des DAGs vollständig durchgelaufen sein muss, bevor eine neue Ausführung gestartet werden darf. Die Überprüfung von orchestrierungsübergreifenden Abhängigkeiten, also anderen DAGs, ist mit Trigger- und Wartefunktionen umgesetzt worden. So werden DAGs erst gestartet, wenn ein anderer DAG erfolgreich durchgelaufen ist. Während die Orchestrierungskomponente vorausgesetzte Topics nur überprüft und fehlschlägt sobald eine Voraussetzung nicht erfüllt ist, werden DAGs in Airflow erst gestartet, wenn der vorausgesetzte DAG erfolgreich durchgelaufen ist. Dies führt in der Praxis zu flexibleren Ausführungen, weswegen Airflow besser bewertet wurde als die 1-Click-Orchestrierung. [AiroDd]

Integrierbar in die Systemarchitektur: Die Orchestrierungskomponente nutzt viele bereits bestehende Prozesse. Dadurch ist diese optimal an die bestehende Systeminfrastruktur angepasst. Sie nutzt vor allem die Möglichkeiten, die durch den KNIME Server zur Verfügung gestellt werden. Dazu gehört der Scheduler, der die Orchestrierung startet, aber auch Fehlermeldungen an die Nutzer per E-Mail weiterleitet. In der Orchestrierungskomponente werden bereits entwickelte Komponenten zur Authentifizierung genutzt. Auch eine spätere Weiterentwicklung der Orchestrierung ist möglich.
Mit Airflow wird neue Software in die bestehende Systemarchitektur integriert. Der KNIME Server wird dann nur noch die Workflows und die Rechenkapazität zur Verfügung stellen. Alle weiteren Funktionen werden in Airflow ausgelagert sein. Da das Ziel aber darin besteht, die bestehenden Strukturen zu nutzen und die Heterogenität des Systems klein zu halten, erfüllt Airflow diese Anforderung nicht.

Einfach in der Handhabung: Durch die Nutzung der bereits bekannten Software fällt der Einstieg in die Orchestrierung Teammitgliedern leicht. Zusätzlich steht ein Template zum Anlegen von neuen Orchestrierungen zur Verfügung. Da die Berührungspunkte des Anwenders mit der Orchestrierung so klein wie möglich gehalten werden, müssen nur wenige Angaben zur Konfiguration gemacht werden. Falls ein tieferes Verständnis der Komponente erreicht werden soll, kann durch die visuelle Nutzeroberfläche, gegeben durch KNIME, der Ablauf der Komponente besonders gut nachvollzogen werden. Die meisten Nodes verfügen über eine kurze Beschreibung ihrer Funktion, bei weiteren Fragen gibt es eine Dokumentation. Die Eingabetabelle ist erweiterbar und auch das

zeitweilige Deaktivieren von Workflows ist möglich. Um den Ausführungsstand genauer zu prüfen, kann auf das Monitoring-Dashboard oder den ETL-Log zugegriffen werden. Das Aufsetzen von Airflow und vor allem die Integration in die bestehenden Prozesse benötigt einiges an Zeit. Hilfe dabei bietet die ausführliche Dokumentation von Airflow. Auch die Einstiegshürde zum Erlernen von Airflow, einer neuen Software, liegt höher als bei der bekannten Software KNIME. Sobald diese aber überwunden wurde, punktet Airflow mit einem UI, welches die DAGs und ihre Ausführungen visualisiert. Auch die Reprozessierung kann über das UI gestartet werden.

Flexible Ausführung der Workflows: Dadurch, dass das Logging auf der Workflowebene stattfindet, also jede Ausführung dokumentiert wird, können fehlgeschlagene Workflows leicht identifiziert und reprozessiert werden. Die Orchestrierungskomponente kann also von einer Tagesausführung nur Teile auswählen und erneut ausführen. Auch mehrtägige Ausführungen sind problemlos möglich. Da festgestellt worden ist, dass Fehler häufig am Ausführungstag noch behandelt werden, bestand die Anforderung für den Reprozessierungfall am Ausführungstag einen gesonderten Mechanismus zu entwickeln. Auch wenn bereits alle Workflows der Tagesausführung erfolgreich prozessiert wurden sind, werden sie bei einem erneuten Start der Orchestrierung noch einmal ausgeführt. Dies wurde eingebaut, um falls Fehler aufgetreten sind, die während der Prozessierung nicht aufgefallen sind, die Fehlerbehandlung durch eine einfache Reprozessierung zu unterstützen. Airflow bietet feingranularere Steuerungsmöglichkeiten, die fehlenden Workflows können einzeln ausgewählt und dann ausgeführt werden. Da dies in der Orchestrierungskomponente nicht möglich ist, der Punktabzug für die 1-Click.Orchestrierung.
Airflow bietet neben den Einzelausführungen noch weitere nützliche Ausführungsmöglichkeiten. Die Workflows können zum Beispiel für eine gewünschte Zeitperiode erneut ausgeführt werden. [AiroDc]

Sinnvolle Interaktionen im Fehlerfall: Beide Systeme senden im Fehlerfall den Stack-Trace per E-Mail an die verantwortlichen Nutzer. Dies hilft bei einer schnellen Identifikation des vorliegenden Problems. Die Orchestrierungskomponente unterstützt den Wiederanlauf indem sie, nachdem sie gestartet wurde, alle Workflows die noch nicht erfolgreich prozessiert wurden sind ausführt. Selbiges geht auch mit Airflow. Airflow bietet außerdem die Möglichkeit Tasks anzulegen, die nur ausgeführt werden, wenn eine oder alle vorherigen Tasks fehlgeschlagen sind. Dadurch können Aktionen direkt an fehlgeschlagene Ausführungen gebunden werden, was es erleichtert automatisierte Vorgänge zur Fehlerbehebung oder Dokumentation zu erstellen. Aufgrund dieses Features wird Airflow in dieser Kategorie besser bewertet als die Orchestrierung in KNIME.

Sicherstellung der Datenkonsistenz: Workflows werden von der Orchestrierungskomponente erst aufgerufen, wenn sie vorher erfolgreich auf ihre Ausführungserlaubnis hin überprüft wurden sind. Dafür werden die vom Nutzer angegebenen Ausführungsbedingungen überprüft. Auch bei dieser Anforderung ist das genaue Logging von Vorteil. Mittels des ETL- Logs kann immer sicher gestellt werden, dass Workflows nicht mehrfach oder in einer chronologisch falschen Reihenfolge ausgeführt werden. Airflow kann so konfiguriert werden, dass die Datenkonsistenz sichergestellt wird. Mit dem Parameter "depends_on_past" kann einer Task die Ausführungsbedingung zugewiesen werden, dass die letzte Ausführung von selbiger Task erfolgreich gewesen sein muss, um sie erneut aufrufen zu dürfen. Damit wird eine chronologisch sortierte Ausführungsreihenfolge eingehalten. Ein Unterschied zwischen Airflow und der Orchestrierungskomponente ist, dass Workflows in jeglicher Kombination wiederholt ausgeführt werden können. [AiroDe] Dies bringt mehr Flexibilität, kann die Datenkonsistenz im Data Warehouse aber auch gefährden. In der Orchestrierungskomponente geht das in diesem Ausmaß nicht.

Wiederverwendbar: Die Orchestrierungskomponente läuft völlig unabhängig von den eigentlich aufzurufenden Workflows. Sie wird erst durch ihren Input spezifiziert. Airflow kann ebenfalls zur Steuerung von verschiedensten Prozessen verwendet werden. Es wird durch die Angaben zu den DAGs und Tasks in den Pythondateien konfiguriert.

Performante Ausführung der Workflows: Die Ausführungen in der Orchestrierungskomponente laufen in einer Schleife ab. Während der Entwicklung wurde sich gegen eine parallelisierte Ausführung entschieden, da der damit verbundene Implementierungsaufwand sehr groß ist und viele ETL-Prozesse von Ausführungsergebnissen vorheriger Workflows abhängig sind. Airflow hingegen unterstützt die parallelisierte Ausführung. Zusätzlich gibt es unterschiedliche Executor, welche die Tasks aufrufen, die verwendet werden können. Auch ein eigenhändig entwickelter Executer, könnte genutzt werden. Da Airflow performantere Lösungen zur Verfügung stellt, ein deutlicher Punktabzug für die Orchestrierung. [AiroDf]

Nachdem nun die Orchestrierungen verglichen wurden, folgt der Vergleich der Monitoringlösungen.

Schaffung von Transparenz: Um die Transparenz während des Verarbeitungsprozesses zu erhöhen, müssen die Datenflüsse dokumentiert werden. Dazu zählen Metriken über die Veränderungen in den Tabellen. Als auch das Zählen von Inserts, Updates und Duplikaten. Die Umsetzung dessen erfolgt im Workflow selbst, da dort die Daten bearbeitet werden. Die gesammelten Metriken werden dann an die Orchestrierungen

gesendet. Bei der 1-Click-Orchestrierung mit KNIME gibt es einen Monitoring-Log, in dem die Metriken von der Orchestrierung eingetragen werden. Da Airflow keine eigene Monitoringlösung anbietet, werden in diesem Szenario die Metriken in den ETL-Workflows durch eine Write-Log-Komponente in einen Monitoring-Log eingetragen. Die erste Variante ist besser, da eine striktere Trennung der Orchestrierungsaufgaben, in diesem Fall dem Schreiben der Logs, von den ETL-Prozessen stattfindet. [AiroDg]

Erhöhung der Vergleichbarkeit: Ein Vergleich fällt leichter, wenn die Metriken nicht in Tabellenform vorliegen, sondern visualisiert dargestellt werden. In der 1-Click-Orchestrierung und in Airflow wird dafür Tableau verwendet. Die Darstellungsform sollte leicht zu überblicken sein, Balkendiagramme eignen sich dafür gut. Eine Datenanlieferung kann besser eingeordnet werden, wenn sie im Vergleich mit vergangenen Anlieferungen gezeigt wird. Daher sollten die in der Vergangenheit gesammelten Metriken weiter zur Verfügung stehen. Alle Metriken der Ausführungen sind im Monitoring-Log abgespeichert. Das Vorgehen für beide Varianten gleich, daher schneiden sie gleich gut ab.

Integrierbar in die Systemarchitektur: Die Metriken werden bei beiden Lösungen in einem Log gespeichert und mit Tableau dargestellt. Tableau wurde gewählt, da es in der bestehenden Architektur bereits genutzt wird. Es wird auf bestehende Infrastruktur zurückgegriffen, daher bekommen beide Orchestrierungslösungen die volle Punktzahl.

Prognosen: Anhand von Prognosen wird die Einordnung, ob eine Datenanlieferung im Normalbereich liegt, automatisiert. Das könnte mit Schwellwerten, die auf Metriken aus der Vergangenheit basieren, berechnet werden. Sowohl in der Orchestrierungskomponente, als auch in Airflow, ist dies bisher nicht umgesetzt, könnte aber in Zukunft umgesetzt werden.

Monitoring der Laufzeiten: Laufzeiten werden in der Orchestrierung nicht überwacht. Die Umsetzung dessen ist aber möglich. In Airflow hingegen werden die Ausführungszeiten der Workflows überwacht und es gibt ein in das UI integriertes Monitoring mit dem diese angezeigt werden können. Die Metriken können allerdings nicht mit im Tableau-Dashboard angezeigt werden. Da eine möglichst vollständige Prozessbetrachtung erreicht werden soll, gibt es dafür einen Punkt Abzug.

Neben den fachlichen Anforderungen an Orchestrierungen müssen noch einige zusätzliche Auswahlkriterien betrachtet werden, um eine Entscheidung zwischen der Idividuallösung und Apache Airflow zu fällen.

Entwicklungskosten: Die Entwicklungskosten können nur grob abgeschätzt werden und sind nicht allgemeingültig, sondern kalkuliert für den thematisierten Anwendungsfall, der ETL-Prozessierung in KNIME. Um die Kalkulation zu erstellen, wurden die anstehenden Aufgaben zeitlich abgeschätzt und addiert. Da beide Orchestrierungslösungen Tableaudashboards als Monitoring nutzen, ist der Zeitaufwand zur Umsetzung des Monitorings für beide identisch. Die Kostenkalkulation kann der Tabelle 3.6 entnommen werden.

In diesem Absatz folgen noch einige Erklärungen zu der erstellten Kostenrechnung. Der Entwicklungsprozess beginnt mit der Bereitstellung der Infrastruktur. Zum Betrieb von Airflow, muss ein Airflow-Server aufgesetzt werden. Dies wird mit 10 Arbeitsstunden taxiert. Die Eigentwicklung erfolgt in KNIME, welches bereits eingerichtet ist. Danach müssen die Anforderungen an die Orchestrierung gesammelt werden. Dieser Schritt läuft bei beiden Orchestrierungen identisch ab und sollte circa 30 Stunden in Anspruch nehmen. Während in der 1-Click-Orchestrierung Konzepte zur Umsetzung der Anforderungen erstellt und implementiert werden müssen, können die meisten Anforderungen in Airflow durch die Konfiguration der DAGs umgesetzt werden. Die Konfiguration benötigt deutlich weniger Zeit als die Entwicklung. Trotzdem benötigt es bei einer Lösung mit Airflow ungefähr 80 Stunden zur Implementierung. Neben der Konfiguration, muss die Interaktion von Airflow und KNIME ermöglicht werden. Dabei muss unter anderem eine sichere Möglichkeit zur Übermittlung von Anmeldedaten geschaffen werden. Nachdem die Orchestrierungen fertig entwickelt wurden, müssen sie in einer Testumgebung getestet und danach im Produktivumfeld eingesetzt werden. Dabei finden sich, der Erfahrung nach, noch Fehler die behoben werden müssen. Für diesen Prozess werden bei Airflow 10 Stunden eingeplant, da die Orchestrierung als bestehende Softwarelösung bereits funktionieren sollte und vor allem das Zusammenspiel überprüft werden muss. Eine neue selbst entwickelte Anwendung enthält mehr Fehler und daher wird doppelt so viel Zeit für Tests und Anpassungen eingeplant.

Zuzüglich, zu der Zeit zur Umsetzung der Orchestrierung, kommt die Entwicklungszeit zur Erstellung des Monitorings. Da die Lösung mit Tableau für beide Orchestrierungen fast identisch ist, wird die selbe Kalkulation für beide Orchestrierungen genutzt. Es werden 10 Stunden zur Anforderungsanalyse veranschlagt und 80 Stunden für die Implementierung hauptsächlich zur Anpassung der Workflows und Entwicklung der Dashboards. Auch das Monitoring muss danach getestet werden, wofür 10 Stunden

Tabelle 3.6: Angesetzte Kalkulation der einzuplanenden Entwicklungszeit in Stunden

Arbeitspaket	Airflow	1-Click-Orchestrierung	Monitoring
Architektur bereitstellen	10	0	0
Anforderungsanalyse	30	30	10
Implementierung	100	170	80
Testing	10	20	10
Gesamt	150 Stunden	220 Stunden	je + 100 Stunden

veranschlagt werden.
Der Arbeitsaufwand zur Implementierung der Orchestrierung mit Monitoring in KNIME und Tableau wird auf 8 Wochen, also 320 Stunden, geschätzt. Eine Lösung mit Airflow und Tableau wird auf circa 5 Wochen, also 250 Stunden, geschätzt, wobei die meiste Zeit für die Schnittstellenansteuerung und das Setup des Monitorings benötigt wird. Berechnet mit einem Stundenlohn von 80 Euro, belaufen sich damit die Entwicklungskosten für die eigenentwickelte 1-Click-Orchestrierung auf ungefähr 25.600 Euro. Dadurch, dass die bestehende Open-Source-Software Airflow zur Realisierung genutzt werden kann, fallen für die Umsetzung etwas weniger Kosten an. Diese belaufen sich, ebenfalls bei einem Stundenlohn von 80 Euro, auf 20.000 Euro. Das ist die circa 20% weniger Kosten, als bei einer Eigenentwicklung und damit etwas günstiger. Airflow benötigt allerdings einen eigenen Server, was zu laufenden Kosten führt. Die Höhe der anfallenden Kosten ist abhängig vom Anbieter.

Wartbarkeit/Erweiterbarkeit: Die Orchestrierungskomponente ist in KNIME implementiert und dokumentiert wurden. Dadurch, dass das gesamte Team gute Kenntnisse in KNIME hat und die Funktionalitäten der Orchestrierung auf das essentielle beschränkt sind, ist die Orchestrierung gut wartbar und erweiterbar. Es steht den Nutzern frei Änderungen durchzuführen, da sie die alleinigen Nutzer sind. Apache Airflow ist ein Open-Source-Projekt mit vielen Beteiligten. Es wird viel genutzt, stetig weiterentwickelt und verbessert. Auftauchende Probleme werden, sofern sie als wichtig eingestuft werden, auch ohne eigenes Zutun behandelt. Die Lenkung der Entwicklung auf gewünschte Features wäre durch eine Eigenbeteiligung möglich. Die Möglichkeit Einfluss zu nehmen, ist aber viel geringer als bei der eigenständig entwickelten Lösung. Airflow bietet viele Funktionalitäten von sich aus an und kann durch Packages zusätzlich erweitert werden. Packages können auch selbst entwickelt und integriert werden, falls Bedarf besteht. Die verwendeten Programme müssen regelmäßig durch Updates aktualisiert und gewartet werden. Fehler können häufiger durch die höhere Anzahl eingesetzter Systeme entstehen. Auch ist ein fehlerfreier Weiterbetrieb nach jedem Update aufwendig zu prüfen.

Resümee: Beide Orchestrierungen erfüllen die grundlegenden Anforderungen an eine Orchestrierung. Sie erleichtern den Nutzern das Tagesgeschäft, indem die ETL-Prozesse automatisiert abgearbeitet werden. Es werden genügend Möglichkeiten zur Verfügung gestellt, Ausführungsbedingungen anzugeben und so die Abhängigkeiten abzubilden. Die Orchestrierungskomponente ist zugeschnitten auf das bestehende System entwickelt worden und passt daher gut zur Gesamtarchitektur. Airflow hingegen passt nicht zum bestehenden System. Dadurch das Nutzer von KNIME kaum Programmierkenntnisse benötigen, ist der Einstieg und die Benutzung der Orchestrierung anfängerfreundlicher. Airflow hingegen muss neu erlernt werden. Der Vorteil bei Airflow liegt aber im UI,

welches die Ausführungen und viele Funktionen visualisiert. Die tägliche Interaktion mit Airflow ist daher angenehm.
Das Einhalten der chronologischen Verarbeitungsreihenfolge überprüfen beide Orchestrierungen. Dadurch, dass Airflow eine händisch angestoßene Reprozessierung immer erlaubt, kann die Datenkonsistenz trotzdem verletzt werden. Dafür wird der Nutzer aber weniger eingeschränkt. In Airflow ist es auch möglich Workflows einzeln oder in frei wählbaren Teilmengen erneut zu prozessieren. Die Orchestrierungskomponente unterstützt dies in dieser Form nicht. Ein weiteres Feature, was nur von Airflow angeboten wird, ist das Tasks angelegt werden können, die nur ausgeführt werden, wenn die vorherigen Tasks fehlgeschlagen sind. Dies bietet Möglichkeiten die Fehlerbehandlung weiter zu automatisieren. Den ersten Schritt zur Unterstützung bei der Fehlerbehandlung gehen aber beide Orchestrierungen, indem sie Fehlermeldungen gesammelt an den Nutzer senden. Außerdem wird bei einem Wiederanstoß die Ausführung am ersten fehlgeschlagenen Workflow fortgesetzt. Die Orchestrierungen sind beide wiederverwendbar, da nur die Konfiguration angepasst wird, nicht aber die Programme an sich. Eine parallelisierte Ausführung unterstützt nur Airflow, die Workflows in der Orchestrierungskomponente werden seriell ausgeführt.
Die Basis zur Erstellung eines Monitorings, das Sammeln der Metriken, läuft in jedem Fall gleich ab. In die Workflows müssen nach jedem Prozessschritt Abfragen integriert werden. Danach werden die Metriken an die Orchestrierungen gesendet. Beide Orchestrierungslösungen setzen beim Monitoringdashboard auf die bereits vorhandene Software Tableau, mit der eine übersichtliche Darstellung der Metriken realisiert werden kann. Airflow unterstützt zusätzlich ein Taskmonitoring mit dem die Laufzeiten überwacht werden können. Das Logging der Laufzeiten findet in der Orchestrierungskomponente nicht statt, könnte aber noch integriert werden. Prognosen müssten in die Tableau-Dashboards noch implementiert werden.
Die Entwicklungskosten für die 1-Click-Orchestrierung sind etwas höher als die Kosten zur Nutzung von Airflow. Die 1-Click-Orchestrierung benötigt allerdings, dadurch dass sie ebenfalls in KNIME läuft weniger Wartung und ist leichter erweiterbar.

Zusammengefasst erfüllen beide Orchestrierungen die grundlegenden Anforderungen ähnlich gut. Die Anforderungen zur Erreichung eines hohen Automatisierungsgrades werden von Airflow in fast allen Punkten erreicht. Die Orchestrierungskomponente bietet deutlich weniger Flexibilität, was negativ in mehreren Anforderungen auffällt. Trotz dessen besitzt die Orchestrierungskomponente einen mittleren Automatisierungsgrad. Im Monitoring bieten beide Orchestrierungen ähnliche Features, wobei Airflow zusätzlich auch Metriken über die Tasks an sich sammelt. Das Problem mit der Implementierung in Airflow liegt darin, dass Airflow nicht an die bestehende Systeminfrastruktur anknüpfen kann. Die Ansteuerung der Workflows ist umständlich, da dies über die KNIME API realisiert werden muss. Dadurch können die mitgebrachten Vorteile von KNIME, wie

der Scheduler, nicht ausgenutzt werden.
Wie auch in den Tabellen am Anfang des Vergleich sichtbar war, ist ein Unterschied zwischen den Orchestrierungen vorhanden. Vor- und Nachteile wiegen sich allerdings bei beiden ungefähr aus, mit leichtem Vorteil für die eigenentwickelte Orchestrierung. Die zu entwickelnde Orchestrierung soll regelmäßig laufende ETL-Prozesse steuern. Obwohl die auszuführenden Workflows stetig weiterentwickelt werden, verändern sich nicht täglich und sind daher eher statisch. Eine Orchestrierung mit einem mittleren Automatisierungsgrad ist daher für die gesuchten Zwecke vollkommen ausreichend. Die langfristige Nutzbarkeit durch den robusten Aufbau der Orchestrierungskomponente, mit einer einfachen Möglichkeit, neue Features zu integrieren und das ohne hohe Wartungskosten, überwiegen in diesem Fall die geringere Flexibilität und die etwas höheren Initialkosten.
Aufgrund der genannten Punkte wird sich für die Eigenentwicklung entschieden. Im nächsten Kapitel wird es um die Implementierung der Orchestrierungskomponente gehen.

4 Orchestrierungskomponente

Die Orchestrierungskomponente soll das Handling der täglich laufenden ETL-Prozesse vereinfachen, in dem der händische Aufwand minimiert wird. Dafür werden die Workflows, in denen der ETL-Prozess stattfindet, mit ihren Abhängigkeiten in die Komponente hineingereicht. Es können verschiedenste ETL-Prozesse orchestriert werden. Die Datenanlieferungen werden, identifiziert durch das jeweilige Datum, tageweise abgearbeitet. Eine Orchestrierung läuft immer für einen ETL-Prozess mit seinen jeweiligen Workflows. Jede Orchestrierung muss unter einem Topic zusammengefasst werden. In dem ETL-Log wurde die bisherige Ausführungshistorie dokumentiert. Anhand dieser werden bei Anlauf der Orchestrierung die Workflows ausgewählt, welche ausgeführt werden müssen. Die Ausführungsreihenfolge wird mit den in der Eingabetabelle angegebenen Abhängigkeiten berechnet und dann nach dem Ausführungsdatum sortiert. Danach werden die Workflows einzeln in einer Schleife abgearbeitet. Für jeden Workflow wird überprüft, ob dieser ausgeführt werden darf oder nicht. Dafür werden die Abhängigkeiten und je nach Konfiguration noch weitere Voraussetzungen abgefragt. War die Prüfung erfolgreich, wird der Workflow ausgeführt und das Ausführungsergebnis in den Log eingepflegt. Es gibt die Möglichkeit, Monitoring Metriken in den Workflows zu sammeln und dann im Monitoring Dashboard anzeigen zu lassen.

4.1 Umsetzung

Die Abbildung 4.1 zeigt einen Orchestrierungsworkflow, welcher die Konfigurationsnodes und die Orchestrierungskomponente enthält. Die Abbildungen 4.2 und 4.3 enthalten die ausgeklappte Orchestrierungskomponente. Die einzelnen Prozessierungsschritte werden im Folgenden genauer erklärt.

Component Input: Die Komponente braucht zur Ausführung einige Informationen. Dafür werden die Konfigurationsnodes (Abbildung 4.1) angepasst und als Input in die Komponente gegeben. Dazu gehört das Ausführungsdatum. Es handelt sich dabei um das maximal betrachtete Datum, bis zu dem fehlende Workflows ausgeführt werden. Neben dem Datum können auch verschiedene Voraussetzungen für einen Workflowauf-

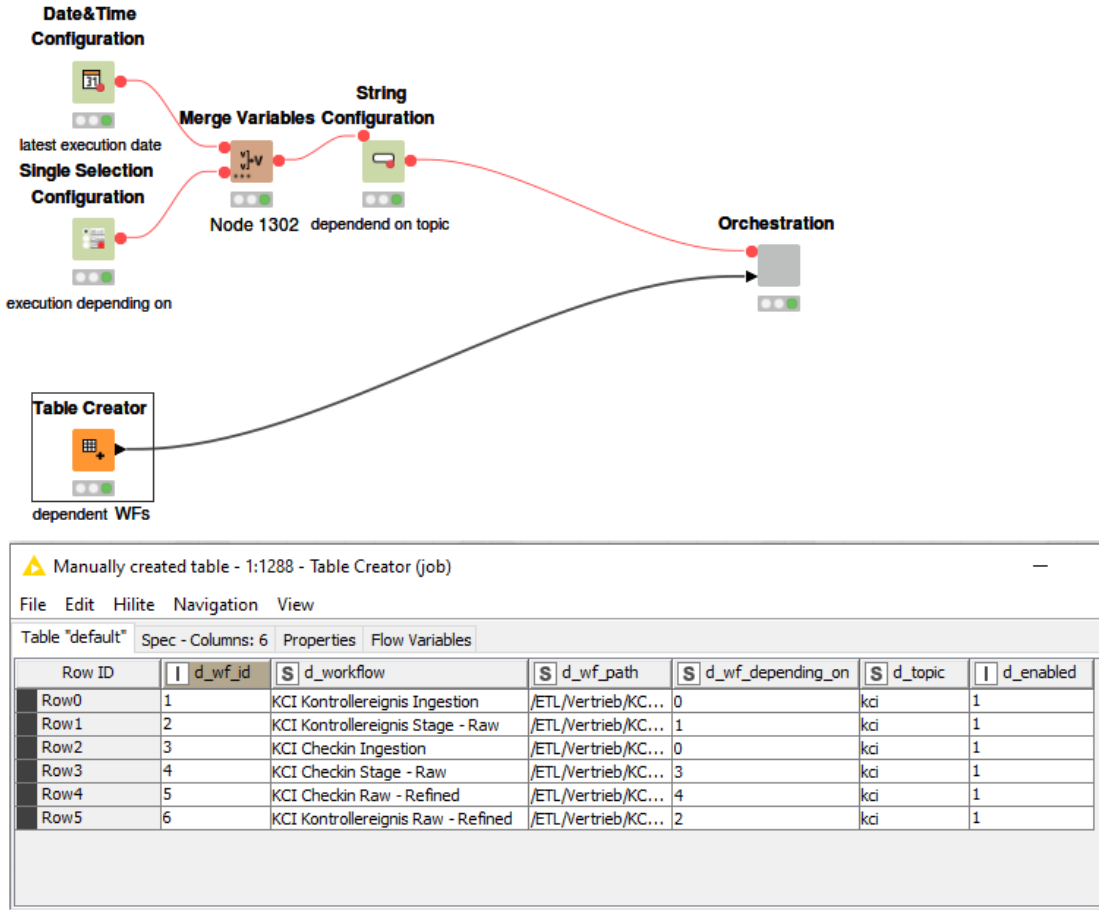

Abbildung 4.1: Orchestrierungsworkflow mit Konfigurationsnodes und der Orchestrierungskomponente

ruf festgelegt werden. Da einige ETLs aufeinander aufbauen, gibt es die Möglichkeit Topicabhängigkeiten anzugeben. Die Topics müssen ausgeführt worden sein, bevor die Workflows der Orchestrierung aufgerufen werden können. Als Drittes wird eine Tabelle mit den Workflowinformationen hineingereicht. Eine Zeile besteht immer aus einer Workflow-ID (diese muss innerhalb der Tabelle einzigartig sein), dem Namen, dem Pfad zum Workflow und einem Topic. Ein Topic umfasst einen ETL-Prozess. Jeder Workflow kann außerdem Abhängigkeiten zu anderen Workflows haben. Auf Basis der angegebenen Abhängigkeiten wird im Dependent-Workflows-Metanode später die Ausführungsreihenfolge berechnet. Workflows können außerdem deaktiviert werden, sie werden dann nicht mehr ausgeführt. In der letzten Spalte der Konfigurationstabelle kann ein Suptopic festgelegt werden. Dies wird nur gebraucht, wenn auch Monitoring Metriken gesammelt werden. Ein Subtopic soll die Workflows umfassen, die je eine Strecke von Staging-Tabelle zu Raw-Tabelle zu Refined-Tabelle bearbeiten. Durch diese Zuordnung können später im Monitoring Veränderungen in den Daten von der Staging-Tabelle bis in die Refined-Tabelle nachvollzogen werden.

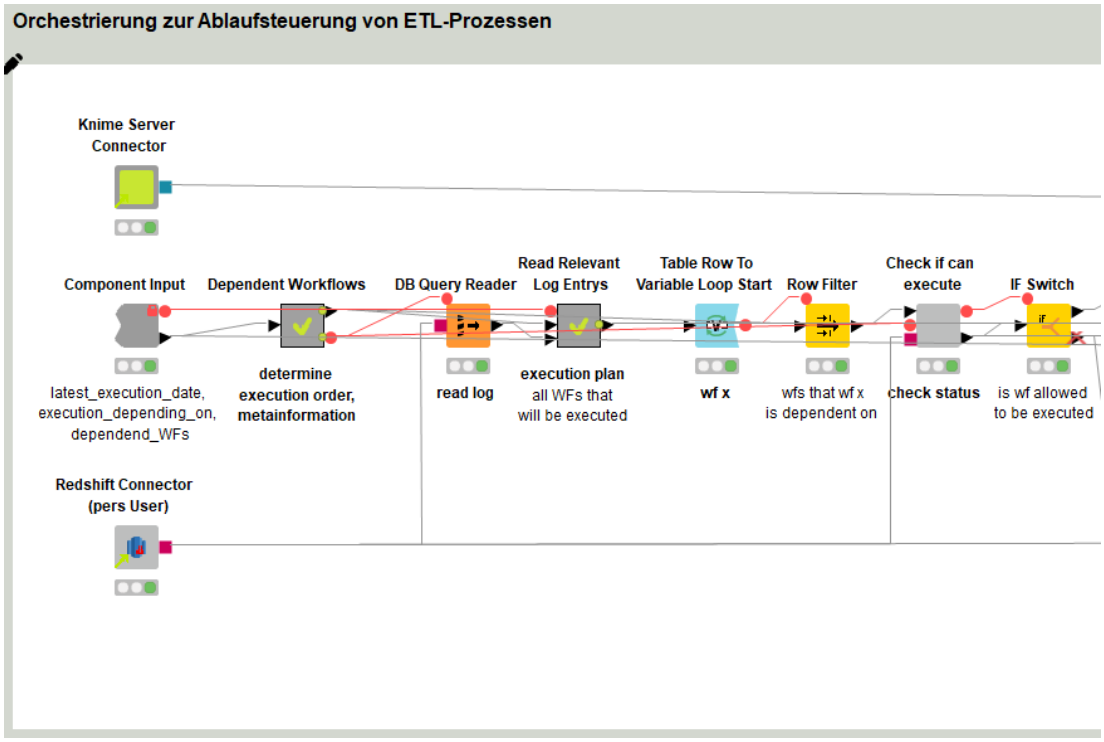

Abbildung 4.2: Aufbau der Orchestrierungskomponente Teil 1

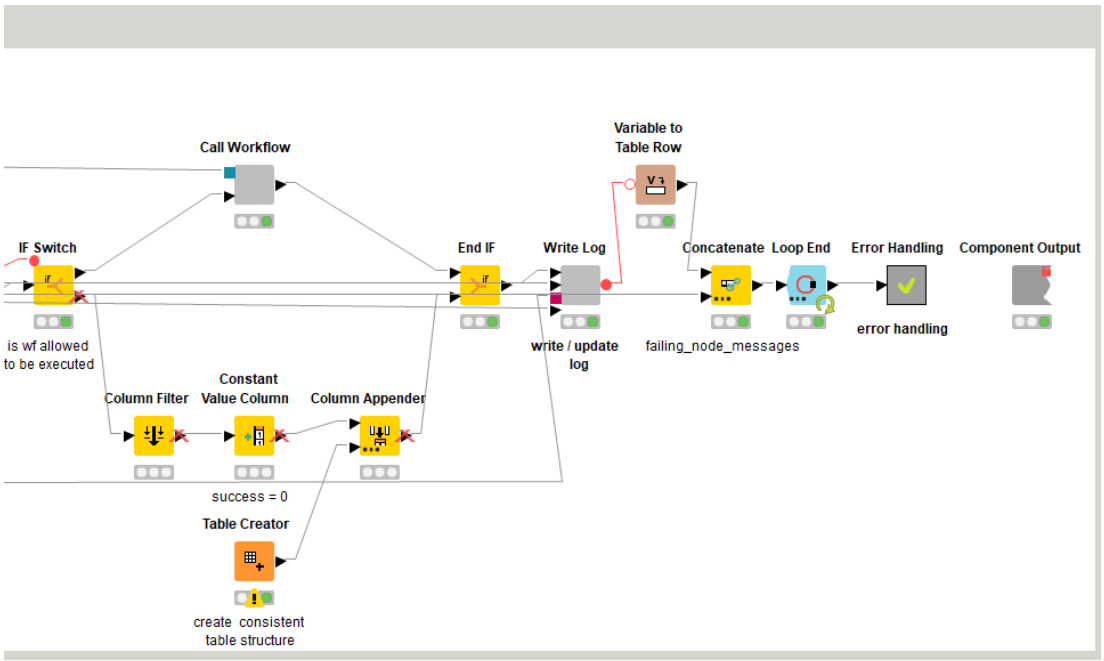

Abbildung 4.3: Aufbau der Orchestrierungskomponente Teil 2

Dependent Workflows: In diesem Teil der Orchestrierung werden deaktivierte Workflows ausgefiltert. Außerdem wird die Ausführungsreihenfolge der Workflows festgelegt, eine nähere Beschreibung dieses wichtigen Prozessschrittes folgt im nächsten Abschnitt.

```
1  import networkx as nx
2  import pandas as pd
3
4  edges = []
5  execution_order = []
6  output = []
7  G = nx.DiGraph()
8
9  for d_wf_id in input_table_1['d_wf_id']:
10     G.add_node(d_wf_id)
11
12 for index, row in input_table_1.iterrows():
13     if(int(row['d_wf_depending_on']) != 0):
14         edges += [(row['d_wf_depending_on'],row['d_wf_id'])]
15
16 G.add_edges_from(edges)
17
18 if not(nx.is_directed_acyclic_graph(G)):
19     raise Exception("Dependent Workflows: Die angegebenen Workflows"
20         + "enthalten einen Zyklus in Ihren Abhängigkeiten!!")
21
22 execution_order = (list(nx.topological_sort(G)))
23
24 for d_wf_id in execution_order:
25     for index, row in input_table_1.iterrows():
26         if(row['d_wf_id'] == d_wf_id):
27             output.append(row)
28
29 output_table_1 = pd.DataFrame(output,
30         columns=['d_wf_id','d_workflow','d_wf_path','d_topic','d_enabled','d_wf_depending_on'])
```

Abbildung 4.4: Sortierung der Workflows mittels eines DAG

Festlegen der Ausführungsreihenfolge mit einem DAG: Während der Planung wurde sich für die Nutzung eines DAGs, welcher topologisch sortiert wird, entschieden. Zur Umsetzung wurde Python und das Package NetworkX genutzt. NetworkX eignet sich zur Erstellung und Sortierung von Graphen. [NetoDa] In Abbildung 4.4 kann der Code zur Sortierung nachvollzogen werden. Mit NetworkX wird in Zeile 7 ein gerichteter Graph angelegt. Danach können die Workflows, abgebildet durch Knoten, eingefügt werden. Für jede Zeile der Konfigurationstabelle werden nun die Kanten eingefügt. Eine Kante geht immer vom vorausgesetzten Workflow zum folgenden Workflow. Alle Workflows, die keine Abhängigkeiten haben, wurden mit "0" gekennzeichnet, diese bilden die Wurzel des Graphen. In Zeile 18 bis 20 wird geprüft, ob der erstellte Graph einen Kreis enthält. Falls dies der Fall ist, wird eine Exception geworfen, da die Ausführung nicht gestartet werden kann. Handelt es sich aber um einen azyklischen Graphen, wird dieser in Zeile 22 topologisch sortiert. Dafür wurde auf eine Funktion von NetworkX zurückgegriffen. Ab Zeile 24 wird mit Hilfe des Packages Pandas der Output erstellt. Ausgegeben wird die umsortierte Inputtabelle. Im nächsten Schritt wird der Ausführungsplan erstellt und nach der topologisch sortierten Vorlage sortiert. [NetoDb]

Read relevant log entries: In diesem Metanode werden die zu bearbeitenden Workflows bestimmt und sortiert. Ausgeführt werden die Workflows, die bis zum Ausführungstag noch keinen Log-Eintrag haben. Dafür wird eine Validierungstabelle mit allen möglichen Log-Einträgen erstellt. Die Workflows können zu unterschiedlichen Zeiten in den ETL-

Prozess integriert wurden sein. Daher wird eine Validierungstabelle erstellt, in der die Einträge für jeden Workflow einzeln generiert werden, jeweils mit dem Startdatum der ersten Ausführung. Die Validierungstabelle wird dann mit dem realen Log abgeglichen. Alle fehlenden Einträge werden dem Ausführungsplan angefügt. Zum Ausführungsplan werden außerdem Workflows hinzugefügt, die noch nicht erfolgreich gelaufen sind, also im Log den Status 0 haben. Eine Ausnahme gilt bei den Workflows am Ausführungstag. Diese werden auch gefunden, wenn sie bereits vollständig erfolgreich ausgeführt wurden. Falls Fehler erst nach der Prozessierung auffallen, kann die Orchestrierung nach Behebung des Fehlers so erneut angestoßen werden. Am Ende wird der Ausführungsplan nach dem Datum und der Ausführungsreihenfolge sortiert.

Check If Can Execute: In dieser Komponente werden die Workflows einzeln eingespeist. Es wird überprüft, ob der Workflow aufgerufen werden darf. Dabei wird sichergestellt, dass die abhängigen Workflows bereits erfolgreich prozessiert wurden und somit die Abhängigkeiten des Ausführungstages erfüllt sind. Workflows mit der Abhängigkeit "0" sind von keinem anderen Workflow abhängig. Je nach Nutzerkonfiguration werden ebenfalls Voraussetzungen für den Vortag überprüft. Die strengste Ausführungsbedingung erfordert eine erfolgreiche Ausführung von allen Workflows des Topics am vorherigen Tag. Als Alternative kann die erfolgreiche Ausführung desselben Workflows am vorherigen Tag geprüft werden. Es kann ebenfalls keine Bedingung überprüft werden. Es gibt mehrere Einstellungen, da sich die Anforderungen der ETLs vor allem mit Blick auf die Datenkonsistenz ebenfalls unterscheiden. In dieser Komponente werden zusätzlich noch, falls gesetzt, die Topicabhängigkeiten überprüft. Topicabhängigkeiten gelten für ein Topic mit all seinen Workflows. Die Daten für die Überprüfung kommen aus dem Process-Log,in dem für jedes Topic der Tagesstatus festgehalten wird. Überprüft werden alle Workflows, die selber keine Abhängigkeiten haben. Um die Anzahl an unnötigen Überprüfungen zu minimieren, werden also nur die Wurzelknoten im DAG geprüft. Nur wenn alle Voraussetzungen erfüllt sind, wird die Ausführungserlaubnis erteilt. Im Fehlerfall werden die Fehlermeldungen gesammelt und mit aus der Komponente gegeben.

Call Workflow: In dieser Komponente wird der Workflow aufgerufen und auf dem Server ein Job zur Ausführung angelegt. Dafür muss eine Verbindung zum KNIME Server bestehen. Falls die Ausführung nicht erfolgreich war, werden die Fehlermeldungen abgespeichert. Gab es Fehlermeldungen wird der Erfolgsstatus auf 0 ansonsten auf 1 gesetzt. In den Workflows können Monitoring Metriken gesammelt werden. Falls das der Fall ist, werden diese als Tabelle ausgegeben. Die Tabelle wird dann in die Write-Log-Komponente gegeben, um im Monitoring-Log eingetragen zu werden.

Write Log: Am Ende jedes Schleifendurchlaufs werden die Ausführungsergebnisse in die Logs eingepflegt. Es wird entweder ein bestehender Eintrag aktualisiert oder ein neuer Eintrag hinzugefügt. Der Process-Log wird nur aktualisiert, wenn alle Workflows des Topics am Ausführungstag bereits ausgeführt wurden sind. Genauere Erklärungen zum Logging folgen im nächsten Unterkapitel.

Error Handling: Falls Fehler auftreten, werden die Fehlermeldungen aus der "Check If Can Execute"-Komponente und aus den aufgerufenen Workflows konkateniert und in einer E-Mail ausgegeben.

4.2 Logging

wf_path	workflow	executi...	success	wf_id	insert_date	topic	update_date
/ETL/Vertrieb/...	PKMS SFTP Download	2022-04-14	1	1	2022-04-14T15:06:10	pkms	2022-04-15T15:00:31
/ETL/Vertrieb/...	PKMS SFTP Ingestion	2022-04-14	1	2	2022-04-14T15:06:10	pkms	2022-04-15T15:04:59
/ETL/Vertrieb/...	PKMS Auftraege Ingestion	2022-04-14	1	21	2022-04-14T15:06:10	pkms	2022-04-19T13:05:17
/ETL/Vertrieb/...	PKMS Kunden Stage - Raw	2022-04-14	1	12	2022-04-14T15:06:11	pkms	2022-04-19T09:28:14
/ETL/Vertrieb/...	PKMS Kunden Raw - Refined	2022-04-14	1	13	2022-04-14T15:06:11	pkms	2022-04-19T09:29:30
/ETL/Vertrieb/...	PKMS Auftraege Stage - Raw	2022-04-14	1	6	2022-04-14T15:06:11	pkms	2022-04-19T13:07:15
/ETL/Vertrieb/...	PKMS Auftraege Raw - Refined	2022-04-14	1	7	2022-04-14T15:06:12	pkms	2022-04-19T13:08:45

Abbildung 4.5: Beispieleinträge im ETL-Log

Der ETL-Log dient dazu, immer den aktuellen Status der Workflows zu dokumentieren. So ist es möglich, fehlende Tage nachzuziehen und die Datenkonsistenz zu garantieren. Durch die Einführung des Logs kann außerdem sichergestellt werden, dass im Fehlerfall nicht immer ein kompletter Tagesdatensatz erneut prozessiert werden muss. Im ETL-Log wird für jede Workflow-Tag-Kombination ein Eintrag angelegt. Der ETL-Log kann in Abbildung 4.5 am Beispiel des KMS-ETL-Prozess (im Bild PKMS) nachvollzogen werden. In der "success"-Spalte wird der Ausführungsstatus festgehalten. Auch der Ausführungszeitpunkt wird gespeichert. Die "update_date"-Spalte wird befüllt, falls ein Workflow mehrmals läuft. Dies kommt unter anderem vor, wenn der Workflow bei der ersten Ausführung keinen erfolgreichen Ausführungsstatus hatte. Neben dem ETL-Log schreibt die Orchestrierungskomponente auch den Process-Log. In diesem wird für jedes Topic ein Tagesstatus festgehalten. Wenn Topicabhängigkeiten überprüft werden müssen, wird in diesem Log nachgeschaut. Außerdem werden die Informationen im Monitoring genutzt. Für die Monitoring-Metriken aus den Workflows gibt es zusätzlich noch einen Monitoring-Log (Abbildung 4.6). In diesem sind in Key-Value-Paaren die Metriken gespeichert. Zur Zuordnung der Key-Value-Paare werden auch Informationen über den Workflow, das Datum und das Subtopic festgehalten.
Da die Orchestrierungskomponente häufig den aktuellsten Stand des ETL-Logs benötigt, wird dieser während jeder Ausführung mehrmals eingelesen. Um das Einlesen des Logs performant zu gestalten, wird niemals der gesamte Log eingelesen. Stattdessen werden

topic	wf_id	workflow	executi...	data_d...	key	value	subtopic
pkms	7	PKMS Auftraege Raw - Refined	2022-04-14	2022-04-13	refined_insert_count	180000	auftraege
pkms	7	PKMS Auftraege Raw - Refined	2022-04-14	2022-04-13	refined_before_count	900000	auftraege
pkms	7	PKMS Auftraege Raw - Refined	2022-04-14	2022-04-13	refined_after_count	1080000	auftraege
pkms	7	PKMS Auftraege Raw - Refined	2022-04-14	2022-04-13	refined_duplicate_count	0	auftraege
pkms	6	PKMS Auftraege Stage - Raw	2022-04-14	2022-04-13	raw_update_count	0	auftraege
pkms	6	PKMS Auftraege Stage - Raw	2022-04-14	2022-04-13	raw_insert_count	190000	auftraege
pkms	6	PKMS Auftraege Stage - Raw	2022-04-14	2022-04-13	raw_before_count	1000000	auftraege
pkms	6	PKMS Auftraege Stage - Raw	2022-04-14	2022-04-13	raw_after_count	1190000	auftraege
pkms	7	PKMS Auftraege Raw - Refined	2022-04-14	2022-04-13	refined_update_count	0	auftraege
pkms	6	PKMS Auftraege Stage - Raw	2022-04-14	2022-04-13	raw_duplicate_count	0	auftraege
pkms	21	PKMS Auftraege Ingestion	2022-04-14	2022-04-13	stage_insert_count	200000	auftraege

Abbildung 4.6: fiktive Beispieleinträge im Monitoring-Log

die SQL-Abfragen so weit wie möglich spezifiziert. Dazu gehören Einschränkungen bezüglich des Topics, dem Datum und wenn möglich dem Workflow.

5 Monitoring

Durch die Standardisierung der ETL-Workflows auf eine tägliche Datenverarbeitung und die generalisierte Ablaufsteuerung, wurde die Grundlage für ein Monitoring geschaffen. Das Monitoring soll den Datenanlieferungsprozess überwachen,dokumentieren und visualisieren. Das Ziel dahinter ist es die Datenqualität zu steigern. Dies kann erreicht werden, indem Abweichungen in den angelieferten Datenmengen oder außergewöhnlich viele aussortiere Daten sichtbar werden. Fehler werden allerdings nur aufgezeigt, die Berichtigung der Daten muss vom Nutzer übernommen werden.
In diesem Kapitel geht es um den Entwicklungsprozess und das daraus entstandene Monitoring. Dieses wird seither im Produktivumfeld eingesetzt.

5.1 Integration in den Datenanlieferungsprozess

Im ersten Schritt wurden in die Workflows, die den ETL-Prozess abarbeiten, die Abfragen zum Erhalt der Metriken eingebaut. Da das Monitoring nur neben den eigentlichen Aufgaben des Workflows läuft, werden eine "Pre-Exection-Komponente" und eine "Post-Execution-Komponente" eingefügt. In dem "Pre-Exection-Komponente" werden die neu einzufügenden Daten gezählt, also die Inserts und Updates. Außerdem wird die Zeilenanzahl der unbearbeiteten Tabelle abgefragt. In der "Post-Execution-Komponente" werden die bearbeitete Tabelle und die Duplikate gezählt. Die SQL-Abfragen sollten, um die Verarbeitungsgeschwindigkeit zu erhöhen, möglichst In-Database erfolgen. Das heißt, dass die zu zählende Tabelle nicht extrahiert werden muss, sondern die SQL-Query an den Server, auf dem die Tabellen gespeichert sind, geschickt und dort ausgeführt wird. Die Querys um die Updates in einer Tabelle zu zählen hier als Pseudo-Code-Beispiel:

```
1 SELECT COUNT ("temp_table" . "primary_key")
2   FROM (
3     (SELECT * FROM "staging_table") AS "staging_temp_table"
4       INNER JOIN (SELECT * FROM "raw_table") AS "raw_temp_table"
5         ON "staging_temp_table" . "primary_key" =
6           "raw_temp_table" . "primary_key"
7     ) AS "temp_table";
```

Die Duplikate werden mit der folgenden SQL-Abfrage gezählt.

```
1 SELECT SUM("count")
2   FROM (
3     SELECT "primary_key", COUNT(*) - 1 AS "count"
4       FROM
5         SELECT * FROM "raw_table"
6           GROUP BY "primary_key"
7             HAVING COUNT(*) > 1)
```

Da es, in den von der Datenquelle heruntergeladenen Tabellen, noch keinen Primärschlüssel gibt, muss dort die gesamte Zeile übereinstimmen, um als Duplikat gezählt zu werden. Als GROUP BY-Bedingung werden daher alle Spalten genutzt.
Die Metriken werden daraufhin gesammelt in den Container-Output-Node gegeben. Dieser Node reicht die Daten zurück an den Aufrufer, also an die Orchestrierung.

Die Orchestrierungskomponente ist so gebaut, dass das Sammeln von Monitoringinformationen ein optionales Feature ist. Falls aber Metriken gesammelt wurden, werden sie über den "Call Workflow"-Node in die "Write Log"-Komponente gegeben. Dort werden entweder neue Einträge erstellt oder, falls schon Einträge mit dem gleichen Primary Key in der Monitoring-Log-Tabelle bestehen, diese aktualisiert. Neben den Metriken werden zusätzlich noch Informationen über den Workflow in den Log geschrieben.

Der Log hält die gesamten Monitoringmetriken. Eine Zeile besteht dabei immer aus der Metrik gespeichert als Key-Value-Paar und aus Informationen über den Workflow wie Name, ID, Topic und Subtopic. Außerdem dem Ausführungsdatum und da die Daten zeitverzögert ausgeführt werden, einem Feld mit dem Datum, an dem die Daten angefallen sind. Da auf Basis des Keys später das Monitoring aufgebaut wird, ist es wichtig, wie dieser genau aussieht. Die Namen der Keys sollten möglichst selbst erklärend sein. Daher sind sie nach dem Muster "ReifegradTabelle_Metrik_Zählart" (z.B. stage_download_count) aufgebaut.

5.2 Dashboards

Das Monitoring-Dashboard besteht aus drei Ansichten. Das erste Dashboard, in Abbildung 5.1, soll einen schnellen Überblick über den Ausführungsstatus der Workflows geben. Der angezeigte Zeitraum kann mittels einens Schiebereglers angepasst werden. Wird ein Workflow zur näheren Betrachtung ausgewählt, wird zur zweiten Ansicht, dargestellt in Abbildung 5.2, gesprungen. Diese zeigt die Inserts, Updates und Duplikate pro Tag. Workflow, Thema und betrachteter Zeitraum sind konfigurierbar. Die dritte

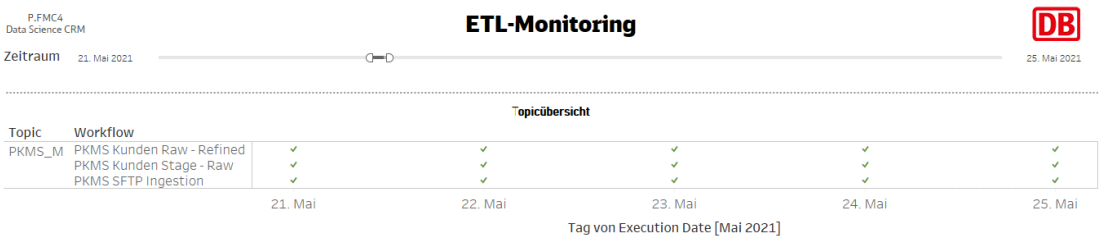

Abbildung 5.1: Dashboard zur Topicübersicht mit Ausführungsstatus

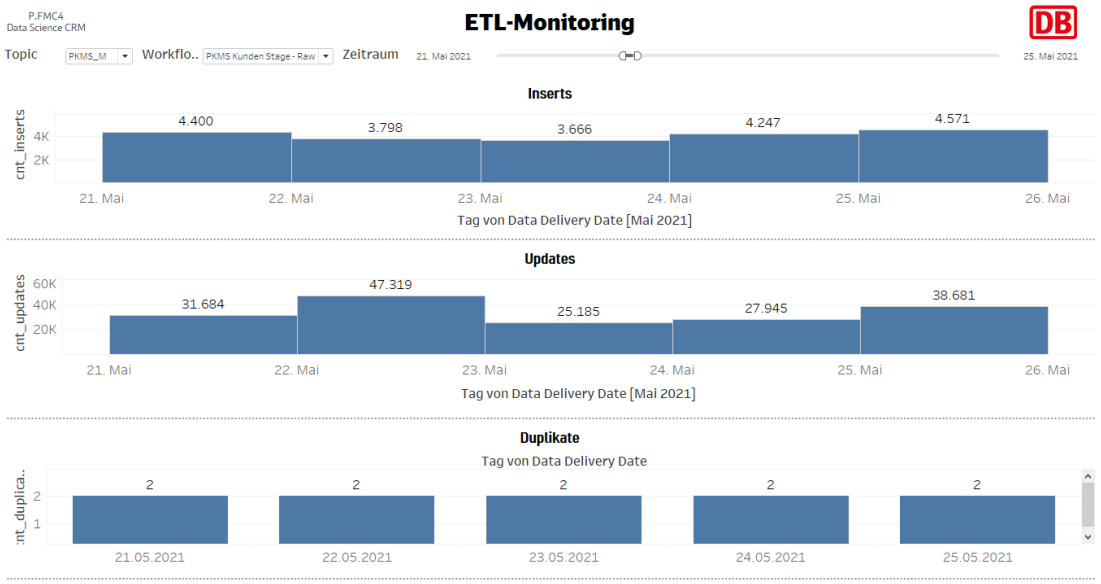

Abbildung 5.2: Detailliertes Dashboard zur Ansicht der Monitoringmetriken

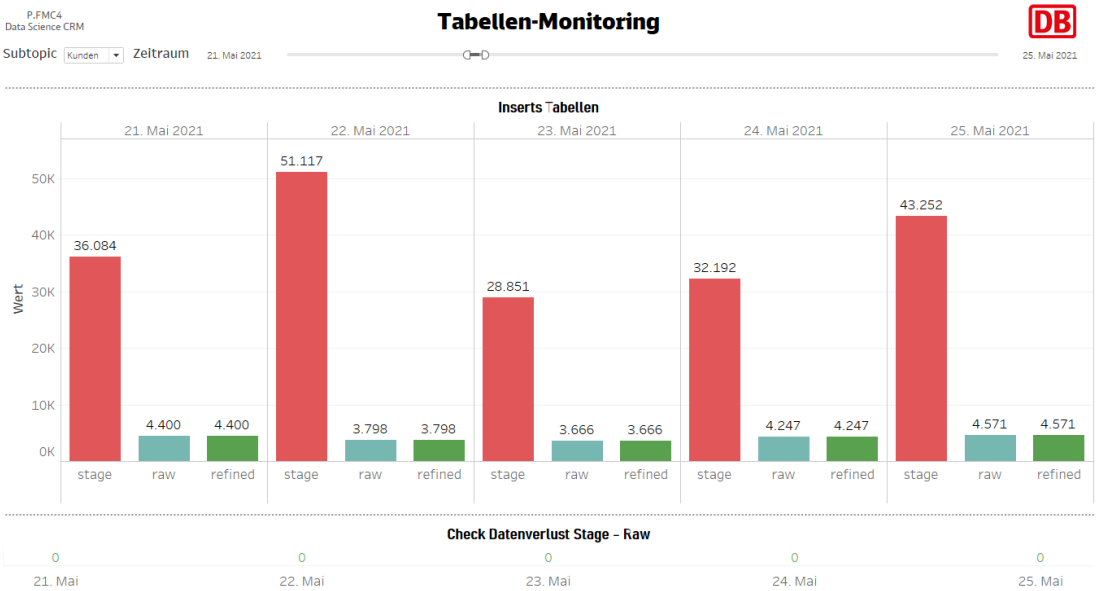

Abbildung 5.3: Dashboard mit Metriken über die Tabellen eines Subtopics

Ansicht, sichtbar in Abbildung 5.3, zeigt die Inserts in die Stage-, Raw- und Refined-

Tabellen zu einem Subtopic. Außerdem wird angezeigt, ob es einen Datenverlust gab.

Umsetzung: Zur Erstellung des Übersichtsdashboards (Abbildung 5.1) wird der ETL-Log ausgelesen. Die Spalten aus der Tabelle werden in Tableau angezeigt und können der Darstellung hinzugefügt werden. Das Datum bildet die Spalten und in den Zeilen stehen das Topic und der Workflow. Die Ausprägungen des Ausführungsstatus wurden mit Symbolen belegt. Eine fehlgeschlagene Ausführung wird mit einem roten Kreuz gekennzeichnet. Die rote Signalfarbe fällt dem Nutzer sofort ins Auge und signalisiert Handlungsbedarf. Die Variablen "Topic" und "Workflow" werden mit einem Klick auf eine der dargestellten Zeilen als Filter auf die anderen Dashboards gesetzt. Zusätzlich wird zum zweiten Dashboard gesprungen. Die angezeigten Metriken sind dann bereits nach dem entsprechenden Workflow gefiltert.
Das zweite Dashboard besteht aus drei Diagrammen. Diese zeigen je eine der gesammelten Metriken über den ausgewählten Zeitraum. Zur Integration der Metriken wurde je ein berechnetes Feld erstellt. Diese überprüfen die Keys der Monitoringtabelle auf ihre Metrik, zum Beispiel "inserts". Alle Metriken, die Inserts in Tabellen beschreiben, werden gefunden und, aufgespalten auf den Entstehungstag der Daten, angezeigt. Die Menge der gefundenen Felder wird dann mit Filtern weiter eingeschränkt, diese können frei vom Nutzer gewählt werden. Sie gelten aber immer für alle drei Diagramme im Dashboard. Der eigentliche Zweck besteht im Anzeigen der Werte für einen Workflow, trotzdem sollen die Nutzer möglichst frei agieren können, da immer auch noch unbekannte Anwendungszwecke entstehen können.
Das dritte Dashboard zeigt den Verlauf der Inserts in die Tabellen über einen wählbaren Zeitraum. Zur Realisierung wurden für jede Tabelle (Stage, Raw, Refined) berechnete Felder erstellt. Diese zeigen alle Inserts im Monitoring-Log für je eine Tabellenart. Zusätzlich werden Datenverluste angezeigt. Ein Datenverlust besteht, wenn die Inserts und Updates in die Raw-Tabelle nicht den Inserts in die Staging-Tabelle entsprechen. Ein größerer Datenverlust bedeutet, dass außergewöhnlich viele Zeilen aussortiert wurden. Das könnte an einem Prozessierungsfehler oder einer Datenanlieferung mit schlechter Qualität liegen. Es ist als nachzuprüfender Hinweis zu verstehen. Bei kleineren Abweichungen kann es sich um Zeilen handeln, die während der Transformation aussortiert wurden sind, dies ist ein üblicher Teil des ETL-Prozesses.

Interpretation: Abbildung 5.1 zeigt, dass die Workflows des Topics "PKMS_M " im ausgewählten Zeitraum erfolgreich ausgeführt wurden sind. Der beschriebene KMS ETL-Prozess (Kapitel 2.5) heißt in den Dashboards zwar "PKMS_M " es handelt sich thematisch aber um den gleichen ETL-Prozess. In Abbildung 5.2 sind nähere

Informationen zum Stage-Raw Workflow zu sehen. So wurden täglich zwischen 3666 und 4571 neue Kunden in die Raw-Tabelle eingefügt. Weitaus mehr Einträge, zwischen 25185 am 23. Mai und 47319 am 22. Mai, wurden aktualisiert. Diese Schwankungen liegen im normalen Bereich. Was in diesem Fall normal ist, muss durch einen Data Engineer beurteilt werden. Zu diesen Zahlen spannend zu wissen ist, dass der 22. Mai ein Samstag war. Am 24. Mai war Pfingstmontag und damit Feiertag. Die hohen Zahlen am Samstag und Dienstag, den 25. Mai, könnten also mit dem verlängerten Wochenende zusammenhängen. Das Monitoring der Datenanlieferungen, spiegelt also auch immer das Kundenverhalten wieder. Das sollte bei der Interpretation der Metriken berücksichtigt werden.
Neben den Inserts und Updates zeigt die Übersicht auch noch Duplikate an. Gezählt wird die Menge der Duplikate in der Raw-Tabelle. Solange wie Duplikate nicht gelöscht werden, werden sie in dieser Übersicht angezeigt. Im angezeigten Zeitraum sind zwei Duplikate in der Raw-Tabelle identifiziert wurden.
Ruft der Nutzer das dritte Dashboard (Abbildung 5.3) auf, sind die Inserts über die Tabellen hinweg sichtbar. Auffällig ist, dass viel mehr Einträge in die Staging-Tabelle geschrieben werden als in die Raw- und Refined-Tabellen. Dies liegt darin, dass in den Kundentabellen Einträge oft aktualisiert werden. Alle neuen Kunden, die in die Raw-Tabellen eingefügt werden, werden auch in die Refined-Tabellen eingefügt. Einen Datenverlust gab es nicht. Die gezeigten Anlieferungen konnten problemlos verarbeitet werden. Um dies zu beurteilen, müssen die gesammelten Metriken allerdings immer von einem Data Engineer betrachtet werden, da die Interpretation nicht vom Monitoring übernommen wird.

Funktionsprüfung: Das Monitoring soll auf seine Funktionalität geprüft werden, daher wurde ein vergangenes ungewöhnliches Prozessierungsszenario nachgebildet. Um die Daten im Data Warehouse mit denen des Data Lakes abzugleichen, wurden alle noch gespeicherten bisher prozessierten Stammdaten erneut prozessiert. Dies geschah über einen Zeitraum von circa einem Monat. Die für den ETL-Prozess genutzten Workflows wurden so angepasst, dass sie die Monitoringmetriken sammeln. Im zweiten Schritt wurden die Daten ab Anfang März 2021 noch einmal in einer Testumgebung prozessiert. Mit einem funktionierenden Monitoring dürften so große zusätzliche Datenmengen direkt auffallen. Wie sich dies auf die Menge der angelieferten Daten niedergeschlagen hat, ist in Abbildung 5.4 abgebildet.
In dem Dashboard sind für jeden Tag die Inserts in die Tabellen zu sehen. In Abbildung 5.3 sind normale Datenanlieferungen zu sehen. Was sofort auffällt, ist, dass es einen starken Anstieg der angelieferten Datenmengen ab dem 10.03 gibt. Ab diesem Tag wurden bis zum 13.04 zwischen 6 bis 20-mal so viele Daten verarbeitet. Danach fallen die Inserts wieder in den Normalbereich von vorher zurück. Hätte es zu diesem Zeitpunkt schon ein Monitoring gegeben, wäre diese Veränderung ersichtlich im Monitoring dokumentiert

worden. Die Funktionsprüfung wurde bestanden.

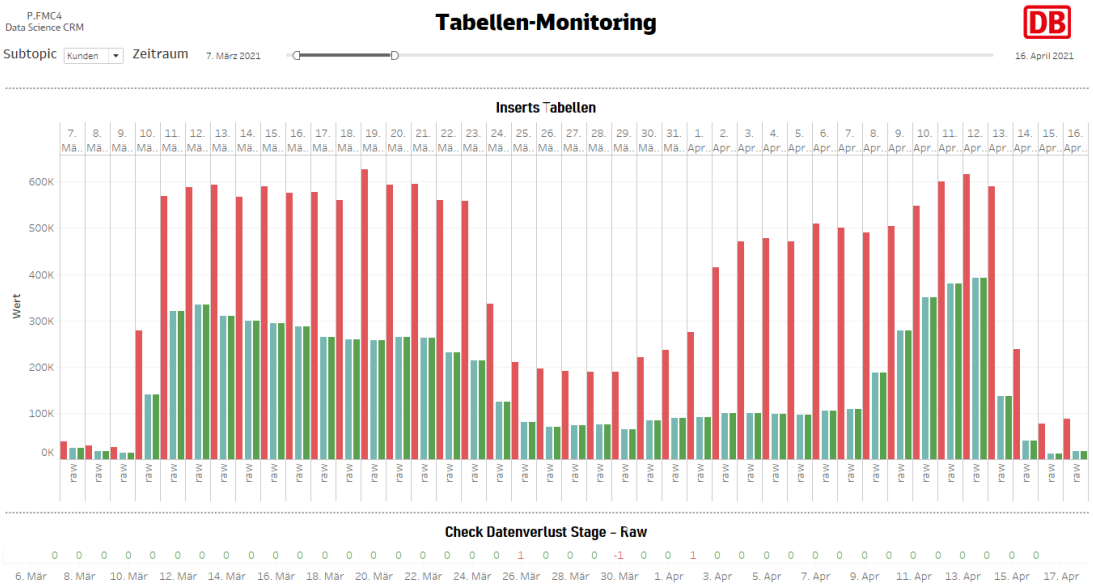

Abbildung 5.4: Reprozessierung der Daten aus dem Data Lake sichtbar im Tabellendashboard

6 Evaluierung

Nachdem die Orchestrierung und das Monitoring ausführlich vorgestellt worden sind, soll in diesem Kapitel die Frage beantwortet werden, was sich verändert hat. Dabei wird es auch darum gehen, ob sich die Datenqualität mit der neuen Orchestrierung und dem neuem Monitoring verbessert hat. Außerdem werden Ideen zur Weiterentwicklung der Komponente und des Monitorings vorgestellt.

6.1 Orchestrierung

Veränderung zu vorher: Bevor die Veränderung beurteilt werden kann, einige Informationen zum vorherigen Stand. Es gab keine Orchestrierungskomponente, stattdessen für jeden ETL-Prozess eine Ablaufsteuerung, in der die Workflows aufgerufen wurden. Die allgemeinen Anforderungen an eine Orchestrierung konnten im Großen und Ganzen mit den Ablaufsteuerungen erfüllt werden. Auch Fehlermeldungen wurden gesammelt und ausgegeben.

Um die Veränderungen besser aufzuzeigen, werden sie anhand des Vorgehens bei Fehlern veranschaulicht. Das Vorgehen im Fehlerfall kann in drei Fälle unterteilt werden. Bei externen Fehlern, also zum Beispiel Verbindungsprobleme zum Data Lake, muss der Fehler gemeldet und dann von den Verantwortlichen gelöst werden. Danach bestand das Vorgehen darin, die Orchestrierung zurückzusetzen und neu zu starten.
In der Orchestrierungskomponente wird die Ausführungshistorie dokumentiert und daher kann die Orchestrierung nach der Fehlerbeseitigung erneut gestartet werden. Sie muss nicht mehr zurückgesetzt werden.
Es gibt nicht nur externe Fehler, sondern auch Prozessierungsfehler in den Workflows. Zur Beseitigung mussten nach einer Fehleranalyse die angelieferten Daten oder der ETL-Workflow angepasst werden. Danach konnte der fehlgeschlagene Teil der Orchestrierung zurückgesetzt und neu gestartet werden.
Die Bearbeitung des ETL-Workflows oder der Daten muss auch bei einer Nutzung der Orchestrierungskomponente erfolgen. Die Orchestrierung würde nach dem Starten aber eigenständig alle fehlenden Workflows ausführen.
Die aufwendigste Fehlerklasse bilden die Schemaänderungen. Bei einer Schemaänderung

müssen die genutzten Stage-, Raw- und Refined-Tabellen angepasst werden. Bevor die Orchestrierung neu gestartet werden kann, muss der Download-Schritt in der Orchestrierung entfernt werden, da dieser in diesem Fall bereits erfolgreich durchgelaufen ist. Die Komponente würde nur die fehlgeschlagenen Workflows ausführen und es muss kein Prozessschritt entfernt werden, da Teilausführungen unterstützt werden.
Im Falle das Fehler nicht direkt am selbigen Tag beseitigt werden konnten, mussten die Orchestrierungen für jeden fehlenden Tag seit dem ersten Fehler neu gestartet werden. Bei acht Orchestrierungen und einem Fehler, der über drei Tage besteht, also zum Beispiel am Samstag auftritt und Montags behandelt wird, müssen bereits 24 Prozesse eingestellt und gestartet werden. Dies benötigt einiges an Zeit und tritt regelmäßig auf. Die Orchestrierungskomponente führt die zusätzlich fehlenden Ausführungen ohne weitere Interaktionen mit aus, dieser Schritt fällt damit weg. Besonders nach längeren Ausfällen lohnt sich also die Verwendung der Orchestrierungskomponente.

Neben einer vereinfachten Fehlerbehandlung bringt die Verwendung der Orchestrierungskomponente noch weitere Vorteile. Dazu zählen Vorteile die sich durch die Nutzung der einheitlichen Komponente ergeben. Diese ist wiederverwendbar für alle Orchestrierungen und leicht wartbar. Anpassungen zum Sicherstellen der Funktionalität oder zum Weiterentwickeln können schnell und leicht Orchestrierungsübergreifend implementiert werden.
Auch das Logging ermöglicht eine deutliche Verbesserung der Orchestrierungen, denn dadurch können alle Ausführungen nachvollzogen werden. Dies führt dazu, dass die Datenkonsistenz, trotz erhöhter Ausführungsflexibilität eingehalten werden kann, da immer auf den letzten Ausführungsstand zugegriffen werden kann.

Weiterentwicklungsideen: Trotz all der genannten Verbesserungen bleiben noch immer viele Ideen, wie die Orchestrierung noch besser werden könnte. Während des Vergleichs mit Airflow wurden viele nützliche Funktionalitäten von Airflow vorgestellt, die auch in die 1-Click-Orchestrierung integriert werden könnten. Die Möglichkeit frei Workflows auszuwählen, die ausgeführt werden sollen, gehört zu diesen. Mit einer Eigenauswahl wird dem Nutzer die Möglichkeit gegeben, jeglichen Ausführungsfall abzubilden.
Ein weiteres nützliches Feature besteht in Workflows, die im Fehlerfall ausgeführt werden. Mit diesen können Fehler besser analysiert werden, indem Statistiken über diese erstellt werden. Das kann bei der Prävention von Fehlern helfen. Die zweite Möglichkeit, die mit solchen Workflows entsteht, ist die, dass häufig auftretende Fehler automatisiert behandelt werden könnten und die Orchestrierung dann zum Beispiel automatisiert neu gestartet werden könnte. Automatisierte Wiederanläufe sind eine einfache Möglichkeit, mit verspäteten Datenanlieferungen und Verbindungsproblemen zum Data Lake umzu-

gehen.
Bei der Bewertung der Orchestrierung anhand der definierten Anforderungen schnitt sie im Punkt "Performante Ausführung" schlecht ab. Um dies zu verbessern, könnte an einer parallelen Ausführung von Workflows gearbeitet werden. Der Vorteil liegt in einer schnelleren Abarbeitung der Datenanlieferungen. Momentan werden die Workflows mittels des DAGs nur entsprechend ihrer Abhängigkeiten sortiert. Mit der Abbildung als Graph könnte aber auch festgestellt werden, welche Workflows parallel ausgeführt werden können. Das parallelisierte Aufrufen der Workflows würde einen Umbau der Orchestrierungskomponente voraussetzen, da die Ausführungen momentan in einer Schleife ablaufen.

6.2 Monitoring

Veränderung zu vorher: Erst durch die Orchestrierungskomponente wurde eine tagesbasierte Verarbeitung der ETL-Daten in allen Orchestrierungen eingeführt. Vorher gab es keine Möglichkeit alle Prozesse zu überwachen, stattdessen wurden die Daten wenn benötigt über SQL-Abfragen auf die Tabellen genauer untersucht. Dadurch konnten stichprobenartig nähere Einblicke in das Data Warehouse erfolgen. Das neue Monitoring ermöglicht einen leicht zugänglichen Überblick über die laufenden Prozesse. Durch die Implementierung in Tableau wurde eine Umgebung gewählt, die von gesamten Team genutzt wird. Das Monitoring steht ständig zur Verfügung um Fragen bezüglich Ausführungen eigenständig zu prüfen. Ein Monitoring bringt nichts, wenn es von niemanden interpretiert und verwendet wird. Damit der Nutzen des Monitoring noch gesteigert werden kann, werden im nächsten Abschnitt Weiterentwicklungsideen thematisiert.

Weiterentwicklungsideen: Ein Feature was relativ einfach umgesetzt werden kann, ist die Messung der Laufzeiten der Workflows. Laufzeiten sind ein Indiz für die Funktionalität des Workflows. Extreme Schwankungen nach oben oder unten können auf Probleme während der Verarbeitung hindeuten. Außerdem können Workflows identifiziert werden, die besonders lange zur Verarbeitung brauchen. Diese können dann auf Optimierungsmöglichkeiten überprüft werden.
Auch weitere Informationen über die Ausführungen der ETL-Workflows, wie Fehlermeldungen, könnten mit in das Monitoring integriert werden. Umso ganzheitlicher die laufenden Prozesse betrachtet werden, je einfacher können Probleme identifiziert und interpretiert werden. Nach einem Fehler und einer Fehlerbehandlung könnten die Metriken einer Datenanlieferung auffällig sein, da der übliche Prozessablauf verändert wurde und der Nutzer kann sich dies später nicht mehr erklären. Werden die aufgetretenen Fehler mit ins Monitoring integriert, ist es leicht möglich dies auch im Nachhinein noch

nachzuvollziehen.
Die nächste Idee ist deutlich aufwendiger umzusetzen, denn es geht um Schwellenwerte für neue Datenanlieferungen. Um die Schwellenwerte zu erstellen, wird eine Menge an vergangen Metriken benötigt, auf die zugegriffen werden kann. Schwellenwerte könnten für die Menge der Anlieferungen, den Anteil der Updates und neuen Inserts in die Tabellen berechnet werden. Erschwert wird diese Berechnung dadurch, dass diese Mengen vom Fahrtverhalten der Kunden beeinflusst werden. Dieses verändert sich über das Jahr und ist abhängig vom Wochentag. Mit Schwellenwerten könnten automatisiert Anomalien in den Datenanlieferungen erkannt und darüber dann informiert werden. Damit könnte das Monitoring den Schritt von einer reinen Darstellung hin zur Interpretation gehen, was Mitarbeiter entlastet und eine stetige Prüfung garantiert.

6.3 Steigerung der Datenqualität

Die Datenqualität im Data Warehouse kann mit vielen verschiedenen Methoden verbessert werden. Es ist wichtig die Daten auf möglichst vielen Abstraktionsebenen automatisiert zu prüfen und zu verbessern, da bei einer so großen Datenmenge ansonsten nur eine sehr geringe Anzahl an Einträgen überprüft werden kann. Der frühestmögliche Eingriffspunkt liegt bei der Einspeisung der Daten in das Data Warehouse, also in den ETL-Prozessen. Um die Datenflüsse besser zu verstehen und zu überwachen, wurde ein Monitoring auf die ETL-Prozesse aufgesetzt. Es handelt sich dabei um ein Monitoring der Datenanlieferungen, diese haben aber einen großen Einfluss auf das Data Warehouse, da sie die Tabellen speisen und verändern.
Wann Daten qualitativ sind, kann an einigen Kriterien wie der Richtigkeit, der Vollständigkeit oder der Aktualität der Daten festgemacht werden. Die Datenqualität wird durch auftretende Fehler verschlechtert. Eine ausführliche Erklärung der Datenqualität und ihren Merkmalen ist im Kapitel 2.5 zu finden.
Die Aktualität konnte durch die Integration der Orchestrierungskomponente verbessert werden, indem Tabellen täglich aktualisiert werden. Dies war vor der Integration nicht bei allen Orchestrierungen gegeben. Eine weitere Verbesserung konnte durch die automatisierten Retrys am Folgetag erreicht werden. Im Fehlerfall, vor allem bei Verbindungsproblemen, verkürzt sich die Zeit, bis das Data Warehouse wieder den aktuellsten Stand hat. Da auch ohne ein händisches Eingreifen am nächsten Tag die fehlenden Datenanlieferung prozessiert werden. Auch nach längeren Ausfällen auf Grund von schwerwiegenderen Fehlern, ist es einfacher und schneller möglich die Datenintegration abzuschließen, da fehlende Tagesanlieferungen automatisiert nachgezogen werden. Insgesamt kann die Aktualität durch die Verwendung der Orchestrierung gesteigert werden, da es seltener und kürzer zu Verzögerungen kommt. Das beeinflusst auch die

Zuverlässigkeit der Daten im Data Warehouse positiv, da das Data Warehouse häufiger zur festgelegten Uhrzeit aktualisiert wurden ist.
Vor allem das Monitoring bietet ein großes Potenzial die Datenqualität stetig zu verbessern. Ab wann Metriken auffällig sind, muss noch von Mitarbeitenden bewertet werden, sobald Metriken im Monitoring aber auffällig sind, sollte die entsprechende Datenanlieferung überprüft werden. Mit dem Monitoring kann nur auf Fehler aufmerksam gemacht werden, die tatsächliche Verbesserung muss ebenfalls durch Mitarbeitende umgesetzt werden. Das Monitoring zeigt Mitarbeitenden aber, wo Fehler unterlaufen sein könnten, was einen großen Mehrwert mit sich bringt. Ungewöhnliche Datenanlieferungen fallen auf und falls Fehler die Ursache für Auffälligkeiten sind, können diese schneller beseitigt werden. Ohne Monitoring wären sie vielleicht nie entdeckt worden, so kann mit einem Monitoring tatsächlich die Richtigkeit der Daten verbessert werden. Zusätzlich findet stetig eine Überprüfung auf Redundanzen in den Tabellen statt, was ebenfalls die Richtigkeit der Daten im Data Warehouse steigert. Mit der Verwendung der Orchestrierung kann außerdem sichergestellt werden, dass es keine Mehrfacheinspielungen mehr gibt, da alle Ausführungen in einem Log festgehalten werden. Mehrfacheinspielungen können zu einer großen Menge an Duplikaten führen.
Ein Merkmal von qualitativen Daten, die Datenmenge, kann mit dem Monitoring sehr gut überwacht werden. Die Anzahl der angelieferten Zeilen wird gespeichert, aber auch die Datenmenge nach jedem weiteren Verarbeitungsschritt wird gezählt. Die Datenanlieferungen enthalten neue Einträge und Updates zu bestehenden Einträgen, auch dieses Verhältnis wird dokumentiert. Starke Veränderungen geben auch hier Anlass zu einer näheren Untersuchung des Prozesses.
Auch die Interpretierbarkeit der Daten kann mit einem Monitoring verbessert werden. Dadurch, dass die Menge an Informationen über einzelne Datenanlieferungen steigt, können die Daten besser eingeordnet werden. Außerdem kann das gesamte Team leicht überprüfen wie aktuell das Data Warehouse momentan ist. Diese Information kann in Auswertungen der Daten wichtig sein und wurde bisher oft von den Mitarbeitenden erfragt.
Zeilen mit einem niedrigen Informationsgehalt werden während der Verarbeitung ausgefiltert. Bisher konnte dieser Aussortierungsprozess nicht überprüft werden, mit dem Monitoring wurde dies geändert. Außergewöhnlich viele aussortierte Werte stellen eine Auffälligkeit dar und die entsprechende Datenanlieferung sollte überprüft werden.
Mit der Orchestrierung kann die Datenqualität hauptsächlich verbessert werden, indem Systemfehler vermieden oder schneller behoben werden. Mit dem Monitoring hingegen können inhaltliche Fehler gefunden werden, da Metriken über die Datenanlieferungen gesammelt werden und neue Einblicke in die Verarbeitungsprozesse gelingen.
Es gibt einige Verbesserungen der Datenqualität, die durch die Verwendung der Orchestrierungskomponente gepaart mit dem Monitoring entstehen. Die tatsächliche Ausbesserung muss häufig weiter durch Mitarbeitende vorgenommen werden, Fehler können aber schneller identifiziert werden, was bei großen Datenmenge mit vielen Fehlerquellen

durchaus hilft.

7 Fazit

Vor dem Einsatz der Orchestrierungskomponente hatte jeder ETL-Prozess eine eigene Ablaufsteuerung. Mit der Generalisierung dieser konnten neue Features implementiert werden und der Wartungsaufwand in Zukunft gesenkt werden. Die Orchestrierungen laufen seit mehreren Monaten produktiv. Eine Verbesserung ist vor allem nach längeren Ausfällen sichtbar, da die Prozessierung aller fehlenden Tagesanlieferungen automatisiert erfolgt und nicht wie vorher ein Orchestrierungsdurchlauf nur eine Tagesanlieferung abarbeiten kann.
Das Vorgehen bei Fehlern ist trotz neuer Orchestrierungskomponente ähnlich geblieben, Fehler müssen nach der Identifikation händisch behandelt werden. Eine Veränderung hat es aber beim Wiederanlauf gegeben, denn durch das Logging der bisherigen Ausführungen ist der aktuelle Ausführungsstand stets bekannt und bei Teilausführungen werden nur die fehlenden Workflows, ab der ersten fehlerhaften Ausführung, ausgeführt. Dies führt zu weniger Fehlern, die entstehen, da das Auswählen der fehlenden Prozesse und händische Zurücksetzen der Orchestrierungen nicht mehr nötig ist.
Das Monitoring bietet zur Überwachung der ETLs ein vollkommen neues Feature. Bisher handelt es sich zwar um eine reine Darstellung der Metriken, später können aber auch Schwellenwerte eingebaut werden. Durch eine stetige automatisierte Vermessung der verarbeiteten Daten bietet das Monitoring jetzt schon die Möglichkeit, unerwartete Ausprägungen in den Datenanlieferungen schneller zu erfassen und zu überprüfen. Dazu zählt das plötzliche An- oder Absteigen der angelieferten Datenmengen. Aber auch die Qualität der Daten wird durch das Zählen der herausgefilterten Zeilen in den Workflows, der Duplikate und der Updates in Teilen überwacht. Trotzdem bietet das Monitoring nur einen kleinen Einblick in die tatsächliche Datenqualität, da eher Metriken über die Menge der Daten, als über die Daten im Einzelnen, gesammelt werden. Nur weil die Metriken aus Datenanlieferungen unauffällig sind, darf nicht davon ausgegangen werden, dass es keine Fehler gibt. Werden Auffälligkeiten identifiziert, ist dies aber ein klares Zeichen, die prozessierten Daten genauer anzuschauen und die Ursache für die Auffälligkeiten zu finden. Die Daten im Data Warehouse und die Datenanlieferungen müssen also trotz Monitoring stets kritisch hinterfragt werden.
Das Anfangs gesteckte Ziel eine Orchestrierung mit hohem Automatisierungsgrad zu entwickeln, konnte, wie im Vergleich mit Airflow sichtbar wurde, nicht erreicht werden (weitere Informationen in den Tabellen 3.4 und 3.2). Andere Orchestrierungen wie Airflow bieten dem Nutzer mehr Flexibilität bei der Ausführung von Workflows und ermöglichen

zum Beispiel eine parallelisierte Ausführung. Die Orchestrierungskomponente konnte einen mittleren Automatisierungsgrad erreichen, dies reicht für den Anwendungszweck aus. Die Weiterentwicklung der Orchestrierungskomponente zur Implementierung der zusätzlichen Features von Airflow lohnt sich nach Betrachtung der Kosten nicht, da die anfänglich gestellten Anforderungen bereits umgesetzt wurden sind. Wird viel Flexibilität bei der Prozessorchestrierung benötigt und ist eine Integration von Airflow in ein System leicht möglich, empfiehlt sich die Nutzung von Airflow. Der Anschluss eines Monitorings an Airflow ist möglich, aber es aufwändiger als bei der 1-Click-Orchestrierung. Es ist mit einer höheren Ausfallquote und einem höheren Wartungsaufwand zu rechnen als bei einer Eigenimplementierung. Außerdem entstehen laufende Kosten durch den benötigten Airflow-Server. Dafür sind die Entwicklungskosten bei einer Eigenentwicklung höher. Zur Entscheidungsfindung sollten die definierten Anforderungen an die Orchestrierung nach den benötigten Vorgaben gewichtet und dann bewertet werden. Mit in den Prozess sollte außerdem einfließen, ob es bereits Software gibt die benötigte Features umsetzt. Diese können dann mitgenutzt werden. Im Fall der Orchestrierungskomponente konnte auf den KNIME-Server als Scheduler und auf die bestehende PostgreSQL-Datenbank zurückgegriffen werden.
Durch die Dokumentation von getätigten Ausführungen und die Verwendung eines Monitorings kann die Datenqualität der Datenanlieferungen überprüft und gesteigert werden. Mehrere Aspekte, die die Datenqualität beeinflussen, konnten verbessert werden, dazu zählt die Aktualität der Daten. Aber auch die Richtigkeit der Daten kann unter anderem durch eine regelmäßige Prüfung auf Duplikate verbessert werden. Die Menge an leicht zugänglichen Informationen über Datenanlieferungen, wie der Zeitpunkt der letzten Aktualisierung des Data Warehouses, steigt ebenfalls durch das Monitoring.

Insgesamt gesehen können durch den Einsatz der Orchestrierungskomponente Prozesse generalisiert und automatisiert werden. Dies führt zu einer Entlastung der Mitarbeiter und ermöglicht außerdem die Integration des Monitorings. Mit diesem kann die Datenqualität der angelieferten Daten transparent überprüft und, sobald nötig, Prozessabläufe angepasst werden.

Literaturverzeichnis

[AiroDa] Airflow: Apache Airflow Documentation — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/index.html

[AiroDb] Airflow: Architecture Overview — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/concepts/overview.html

[AiroDc] Airflow: DAG Runs — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/dag-run.html#re-run-dag

[AiroDd] Airflow: DAGs — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/concepts/dags.html#latest-only

[AiroDe] Airflow: DAGs — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/concepts/dags.html#trigger-rules

[AiroDf] Airflow: Executor — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/executor/index.html

[AiroDg] Airflow: Logging and Monitoring Architecture — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/logging-monitoring/logging-architecture.html

[AiroDh] Airflow: Tutorial — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/tutorial.html

[AiroDi] Airflow: UI — Airflow Documentation (o.D.), URL https://airflow.apache.org/docs/apache-airflow/stable/ui.html

[AmaoDa] Amazon: Amazon Simple Storage Service S3 – Cloud Online-Speicher (o.D.), URL https://aws.amazon.com/de/s3/

[AmaoDb] Amazon: AWS | Amazon Data Warehouse Redshift – Datenverwaltung (o.D.), URL https://aws.amazon.com/de/redshift/

[AmaoDc] Amazon: Definieren von Tabelleneinschränkungen (o.D.), URL https://docs.aws.amazon.com/de_de/redshift/latest/dg/t_Defining_constraints.html

[AmaoDd] Amazon: Was ist ein Data Warehouse? | Wichtige Konzepte | Amazon Web Services (o.D.), URL https://aws.amazon.com/de/data-warehouse/

[Bau13] Bauer, Andreas und Günzel, Holger: *Data-Warehouse-Systeme: Architektur, Entwicklung, Anwendung* (2013)

[Ber07] Berthold, Michael: *KNIME: The Konstanz Information Miner* (2007), URL http://nbn-resolving.de/urn:nbn:de:bsz:352-opus-64487

[Bmw20] Bmwi: Auf einen Blick (2020), URL https://www.bmwi.de/Redaktion/DE/Schlaglichter-der-Wirtschaftspolitik/2020/09/kapitel-1-7-auf-einen-blick.html

[Har10] Harrach, Hakim: *Risiko-Assessments für Datenqualität*, Vieweg+Teubner (2010), URL http://link.springer.com/10.1007/978-3-8348-9720-6

[Har15] Hartmann, Peter: *Graphentheorie*, Springer Fachmedien Wiesbaden (2015), S. 263–294, URL http://link.springer.com/10.1007/978-3-658-03416-0_11

[Hum19] Hummeltenberg, Wilhelm: ETL — Enzyklopaedie Der Wirtschaftsinformatik (2019), URL https://wi-lex.de/index.php/lexikon/informations-daten-und-wissensmanagement/business-intelligence/etl/

[Kar13] Karagiannis, Anastasios; Vassiliadis, Panos und Simitsis, Alkis: Scheduling Strategies for Efficient ETL Execution (2013), Bd. 38(6): S. 927–945, URL https://linkinghub.elsevier.com/retrieve/pii/S0306437912001561

[Kha21] Khan, Wali: The KNIME Server REST API (2021), URL https://www.knime.com/blog/the-knime-server-rest-api

[KNIoD] KNIME: KNIME Getting Started Guide (o.D.), URL https://www.knime.com/getting-started-guide

[Kra20] Kraus, Nicole: Ein Leitfaden für Lehrende zum Einstieg in das Thema Data Mining (2020), URL http://publications.rwth-aachen.de/record/781502/

[Kö12] Köppen, Veit; Gropengiesser, Francis; Kuhlemann, Martin; Saake, Gunter und Sattler, Kai-Uwe: Datenbanken in Der Cloud - Transaktionsmanagement Für Database as a Service (2012), URL https://www.researchgate.net/publication/230661909_Datenbanken_in_der_Cloud_-_Transaktionsmanagement_fur_Database_as_a_Service

[Leg07] Legner, Christine und Otto, Boris: *Stammdaten-Management*, Lange (2007), URL https://www.alexandria.unisg.ch/67534/

[Lei08] Leitner, Paul: Datenqualitätsanalyse beim Ladevorgang in Data Warehouses am Beispiel der ETL-Prozesse der OÖGKK (2008): S. 148, URL http://www.dke.uni-linz.ac.at/rest/dke_web_res/publications/theses/MT0807/MT0807_copy.pdf

[Mei15] Meinl, Thorsten: Giving the KNIME Server (a) REST (2015), URL https://www.knime.com/blog/giving-the-knime-server-a-rest

[Nau07] Naumann, Felix: Datenqualität (2007), URL https://gi.de/informatiklexikon/datenqualitaet

[NetoDa] NetworkX: Introduction — NetworkX 2.8 Documentation (o.D.), URL https://networkx.org/documentation/stable/reference/introduction.html

[NetoDb] NetworkX: Topological_sort — NetworkX 2.7.1 Documentation (o.D.), URL https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.dag.topological_sort.html

[ProoD] Prometheus: Grafana | Prometheus (o.D.), URL https://prometheus.io/docs/visualization/grafana/

[Ran15] Ranjan, Rajiv; Benatallah, Boualem; Dustdar, Schahram und Papazoglou, Michael P.: Cloud Resource Orchestration Programming: Overview, Issues, and Directions (2015), Bd. 19(5): S. 46–56, URL https://ieeexplore.ieee.org/abstract/document/7230217

[Sil20] Silipo, Rosaria: Metanode or Component - What's the Difference? (2020), URL https://www.knime.com/blog/metanode-or-component

[Sin19] Singh, Ajit und Ahmad, Sultan: Architecture of Data Lake (2019), Bd. 5: S. 411–414, URL 331890045_Architecture_of_Data_Lake

[TaboD] Tableau: Was ist Tableau? (o.D.), URL https://www.tableau.com/de-de/why-tableau/what-is-tableau

[Wan96] Wang, Richard Y. und Strong, Diane M.: Beyond Accuracy: What Data Quality Means to Data Consumers (1996), URL https://www.tandfonline.com/doi/abs/10.1080/07421222.1996.11518099

Abbildungsverzeichnis

Tabellenverzeichnis