Leitfaden zur Digitalisierung und Automatisierung von Geschäftsprozessen

Gliederung

1. Einleitung

Abstract

Im Zuge der Digitalisierung entstehen immer größere Datenmengen. Verkaufszahlen, Absatzkennzahlen, Produktmerkmale oder Kundendaten, täglich müssen mehr Daten verarbeitet werden. Gerade für kleine und mittelständische Unternehmen kann diese Datenflut zu einem erheblichen Problem werden. Externe Dienstleister bieten starre Branchenlösungen oder teure Individuallösungen. Branchenlösungen mögen zwar für kleine und mittelständische Betriebe bezahlbar sein, jedoch birgen sie langfristig signifikante Gefahren. Unternehmensprozesse müssen sich zwangsläufig nach den vorgefertigten Funktionalitäten richten, anstatt organisch und unternehmensspezifisch wachsen zu können. Wachstumsabhängige Prozesse können nur eingeschränkt oder vielleicht garnicht umgesetzt werden, weil die eingekaufte Software sie nicht abbilden kann. Dem Reiz einer geringen Investition für eine Branchenlösung steht bald die Unflexibilität der Unternehmensausrichtung gegenüber. Teuere Individuallösungen ermöglichen zwar die Abbildung unternehmensspezifischer Funktionalitäten, sind jedoch nur für Großunternehmen erschwinglich. Der vorliegende Leitfaden soll Unternehmern ermutigen und aktiv unterstützen diesem Dilemma eigenständig beizukommen. Der Fokus liegt dabei auf der anwendungsorientierten Einführung in die Transformation von stetig wiederkehrenden analogen Arbeitsabläufen in automatisierte digitale Prozessroutinen. Der Leser soll von der ersten Konkretisierung seiner Idee bis hin zur eigenständigen Umsetzung, Einführung und Anwendung geführt werden. Für die verschiedenen Etappen existiert bereits eine umfangreiche Auswahl an Literatur und Übungsmaterial. Der Leitfaden hat die Aufgabe den Ablauf der Entwicklung darzustellen, die einzelnen Schritte detailliert zu erläutern und das notwendigen Fachwissen aufzubereiten. Im Gegensatz zu zahlreichen anderen Leitfäden lautet das ausgegebene Ziel hier, dem Leser den Weg der eigenständige Entwicklung und Einführung digitaler Strukturen aufzuzeigen. Die Grenzen des Machbaren werden dabei ausschließlich durch Ihre Motivation gesetzt. Aus Erfahrung soll an dieser Stelle schon folgendes betont werden: Keine Angst vorm Programmieren. Lesern, die bereits bei der unmittelbaren Kombination der Worte "selber" und "programmieren" im Begriff sind diesen Leitfaden wieder beiseite zu legen, soll gesagt werden, dass es sich lohnt hier mal weiter zu lesen. Das Level auf dem hier programmiert werden soll ist absolute Grundstufe. Hier soll keine Raketenwissenschaft betrieben werden und das Handy ihres Kollegen werden Sie danach auch nicht hacken können. Das benötigte Werkzeug ist äußerst simple zu erlernen und in der Anzahl überschaubar. Die praxisorientierte Aufbereitung führt Sie genau da entlang wo Fragen entstehen könnten und werden. Genau diese anfänglichen Hürden sollen gemeinsam genommen werden. Ziel ist es Sie mit so wenig wie möglich Aufwand in die Thematik einzuführen und einen schnellen Mehrwert für Ihr Unternehmen zu gestalten. Jetzt also garnicht erst lang nachdenken sondern weiterlesen.

Motivation

Digitalisierung. Es ist eines der Prädikate unserer modernen Gesellschaft geworden. In kaum einen Bereich unseres täglichen Lebens hat der digitale Wandel nicht längst Einzug gehalten oder bestimmt ihn sogar bereits gänzlich. Onlinehandel, ... Alle Aufgaben sollen immer schneller, effizienter und kostengünstiger abgewickelt werden. Die Digitalisierung wird stets als der große Heilsbringer angepriesen. Ein Rechner führt zahlreiche Prozesse gleichzeitig aus und das in einer Geschwindigkeit die ein Mensch niemals erreichen könnte. Sinkende Prozesskosten, eine planbare Kontinuität und die nahezu unbegrenzte Reproduzier- und Skalierbarkeit einer digitalen Routine sind nur drei Beispiele von unzähligen Vorzügen die die Digitalisierung für die Wirtschaft bietet. Bei all der Euphorie sollte jedoch ein prüfender Blick auf die Schattenseite einer scheinbar für alle gewinnbringenden Entwicklung nicht vergessen werden. Die digitale Welt ist schnell. Wer nicht hinterherkommt, der wird mindestens genauso schnell abgehängt. Der Handel wird zunehmend auf allen Ebenen digitaler. Ganz gleich ob firmeninterne Prozesse oder die Schnittstellen zum Kunden, die Grenzen zwischen der physischen und digitalen Welt verschwimmen zunehmend oder scheinen mitunter ganz zu verschwinden. Genau an dieser Schnittstelle entsteht eine immer größer werdende Nachfrage. Denn wer sich auf der digitalen Seite nicht auskennt muss für externe Dienstleister tief in die Tasche greifen, um weiter mitspielen zu können. Seit geraumer Zeit ist ein klarer und zugleich besorgniserregender Trend zu erkennen. Digitale Dienstleister lassen sich ihre Lösungen immer großzügiger bezahlen. Statt einer monatlich festen Miete, was nebenbei gesagt für Softwareprodukte ein akzeptables Konezpt ist, sind prozentuale

Umsatzbeteiligungen keine Seltenheit mehr. Digitale Dienstleister sind daher nichtmehr als Investition sondern als laufende Kosten anzusehen. Die Digitalisierung ist durch ihre Relevanz für unser gesellschaftliches und wirtschaftliches Leben inzwischen Gegenstand einer breit gefächerten Forschung [1]. Eine für Unternehmer wichtige Schlussfolgerung ist, dass die reine Investition in Software und Digitalisierung keineswegs ausreicht um einen wirtschaftlichen Mehrwert zu generieren. Vielmehr ist die Kompatibilität zwischen Software und Unternehmen entscheident. Und genau darin liegt die Crux. Denn für die Anpassung beider Seiten aneinander gibt es vermeintlich nur zwei Möglichkeiten. Entweder das Unternehmen passt sich an oder die Software. Es wurde bereits darauf hingwiesen, dass die Anpassung der Software in den meisten Fällen nur von einer externen Fachfirma vorgenommen werden kann. Für Unternehmen mit einem begrenzten Budget bleibt demnach nur die Anpassung der bestehenden Prozesse des Unternehmens an die Software. Das beschneidet die Individualität des Unternehmens und reduziert dessen Vorsprung gegenüber Wettbewerbern. Das wirft abermals die Frage auf, welche Breite und Tiefe eine Investition in Software und Digitalisierung für die Entwicklung eines Unternehmens darstellen kann. Hier soll aber noch eine dritte Möglichkeit vorgestellt werden, die Distanz zwischen Branchensoftware und den individuellen Unternehmensprozessen aus eigener unternehmerischer zu überwinden. Was fehlt sind individuelle Bausteine, die die Lücke zwischen Software und Prozessen füllen. Genau diese kleinen, maßgeschneiderten Bausteine bilden den Kern des hier vorgstellten Konzeptes. Für die einführende Motivation soll dies an einem kurzen Besipiel skizziert werden, auf welches später noch sehr detailliert eingegangen wird. Stellen Sie sich bitte folgende Situation vor. Ein Unternehmen muss für die Versendung von Speditionsgütern eine Sendung bei einem Spediteur anmelden. Die erforderlichen Sendungsdaten liegen im Warenwirtschaftssystem des Unternehmens vor. Ein Mitarbeiter muss sich nun den Auftrag anschauen. Um einen Versandtarif zu bestimmen muss er die Maße und das Gesamtgewicht aller in der Bestellung enthaltenen Artikel bestimmen. Danach muss er im Onlineportal des Versanddienstleisters die Kundendaten (Lieferadresse, etc.)eintragen sowie den von ihm bestimmten Tarif auswählen. Klingt erstmal nicht schwer. Dazu braucht ein geübter Mitarbeiter ca. 5 min pro Sendung. Das macht er jedoch nicht nur einmal am Tag sondern 50 Mal. Das sind mehr als 4 Stunden täglich. Wenn das Unternehmen (derzeit 50 Sendungen pro Tag) mit einem Wachstum von 20% plant, ist bereits nach 4 Jahren der erste weitere Mitarbeiter notwendig um den Prozess weiter aufrecht zu erhalten. Nach weiteren knapp 3 Jahren bedarf es schon 3 Mitarbeiter um den gleichen Prozess durchzuführen. Eine Automatisierung dieses Prozesses schafft Abhilfe. Eine vorgefertigte Funktion für diesen Prozess gibt es in dem Warenwirtschaftssystem nicht. Für eine Automatisierung bleibt also nur eine Erweiterung. Diese ist individuell und dementsprechend kostspielig. Weiterhin muss jede nachträgliche Anpassung preislich einkalkuliert werden. Diese Lösung wäre zwar unternehmensspezifisch und würde den Prozess in seinem theoretischen Ablauf erhalten, jedoch ist eine zukünftige Änderung nur durch den Softwaredienstleister möglich. Dieses Beispiel zeigt das Dilemma. Trotz einer vermeintlich individuellen Lösung sinkt die Flexibilität und die Abhängigkeit vom externen Dienstleister wächst. Für das beschriebene Problem haben wir eine eigene Lösung entwickelt und in den Produktionsablauf eingebunden. Wir haben abgesehen von den Lohnkosten keine Ausgaben und erhalten uns für die Zukunft die Flexibilität und Unabhängigkeit. Das eben beschriebene Beispiel wird Ihnen im weiteren Verlauf oft begegnen. Anhand solcher realen Praxisbeispiele wird das Vorgehen im Folgenden sehr detailliert erläutert. Die Entwicklung solcher Lösungen gehören natürlich nicht zum ständigen Tagesgeschäft. Aber genau darin liegt auch die Chance für Unternehmen, ihren Mitarbeitern Abwechslung in Form von spannenden und anspruchsvollen Projekten bieten zu können. Nicht jeder möchte einen Vollzeitjob als Softwareentwickler ausüben. Aber für interessierte Fachkräfte stellt es eine willkommene Herausforderung dar, ihre Fähigkeiten unter Beweis zu stellen und das Unternehmen in dem sie arbeiten im Rahmen eigenständiger Projekte mit zu gestalten. Denn um das Unternehmen weiterzuentwickeln braucht es Mitarbeiter mit der Fähigkeit entwickeln zu können. Im Rahmen der Sicherung und Gewinnung von Fachkräften spielt die Bindung des Arbeitnehmers zum Unternehmen eine entscheidende Rolle. Es ist nachgewiesen, dass die Identifikation von Mitarbeitern mit dem Unternehmen durch die emotionale Bindung zu eigenständigen Projekten gestärkt wird. [2] Diese Möglichkeit sollten Unternehmen nicht verkennen. Denn im Wettbewerb um qualifizierte Arbeitskräfte entscheiden nicht nur das richtige Gehalt und aussichtsreichen Karrierechance. Die Möglichkeit der persönlichen Verwirklichung, erfüllende und abwechslungsreiche Aufgaben sowie die sinnhafte Nachhaltigkeit der Ergebnisse sind der Entlohnung und der Karriere in den meisten Branchen gleichzusetzen. Im Wettbewerb zwischen ländlichem und städtischem Raum um qualifizierte Arbeitskräfte sind diese Faktoren für Unternehmen aus dem produzierenden Gewerbe von entscheidender Bedeutung. Denn es sind nicht unbedingt die grundsätzlich verschiedenen Lebensfaktoren die über den Wohnort von Mitarbeitern entscheiden. Die Mobilität reduziert den Einfluss der räumlichen Distanz zwischen Wohnort und Arbeitsplatz auf die Wahl des Arbeitgebers. Nur weil ein Unternehmen in einem

ländlichen Gebiet ansässig ist, heißt das nicht, dass es keine qualifizierten Fachkräfte gewinnen kann. Vielmehr sind es die eben benannten Kriterien die über die Attraktivität eines Arbeitgebers entscheiden. Wenn man Mitarbeiter sucht die gestalten können und wollen, dann muss man ihnen auch genau diese Möglichkeit schaffen. Im Zuge der Unternehmensdigitalisierung bieten sich genau solche Chancen. Als Arbeitgeber muss man den entsprechenden Rahmen schaffen und Anreize setzen. Der Sinn besteht ja gerade darin klugen und kreativen Köpfen einen Raum zu geben, die mit ihrer Kompetenz das Unternehmen weiterentwickeln. Dazu ist es als Geschäftsführer nicht unbedingt zwingend notwendig selbst die einzelnen vorausliegenden Schritte des Weges zu sehen oder zu verstehen, solange man sich dem langfristigen Ziel bewusst ist. Projektorientierte Arbeitstellen sind oft nicht im produzierenden Gewerbe zu finden. benötigt Platz, welcher in der Stadt nur begrenzt vorhanden und dementsprechend teuer ist. Demgegenüber sind digitale Dienstleister verstärkt im städtischen Raum zu finden. Die Arbeitskräfte Immer weiter gerät man durch die scheinbare Alternativlosigkeit in eine langfristige und tiefgreifende Abhängigkeit. Diese macht sich jedoch oft erst nach geraumer Zeit bemerkbar. Wenn man für eine individuelle Lösung hohe Investitionskosten hatte, kehrt man dem Dienstleister eben nicht so einfach den Rücken, wenn eine funktionelle Erweiterung weitere Kosten verursacht die monatliche Gebühr angehoben wird. Die realen Kosten eines Wechsels des Softwareanbieters umfassen neben den vordergründigen Investitionskosten Laut einer Studie verbringen wir 3,1h pro Tag vor einem Bildschirm. Doch nur ein Bruchteil der Menschen weiß überhaupt wie das Gerät wirklich funktioniert, vor dem sie ein Achtel des Tages verbringen. Nicht umsonst gehört die Welt inzwischen denen, die wissen wie die digitale Welt funktioniert.

[1] Marko Kohtamäki, The relationship between digitalization and servitization: The role of servitization in capturing the financial potential of digitalization, Technological Forecasting and Social Change, Volume 151, February 2020, 10.1016/j.techfore.2019.119804, [https://www.sciencedirect.com/science/article/abs/pii/S0040162519313356]

[2] M. Wastian et al., Commitment und Identifikation mit Projekten, Springer Medizin verlag Heidelberg 2012, DOI 10.1007/978-3-642-19920-2 [https://link.springer.com/chapter/ 10.1007/978-3-642-19920-2_10]

2. Allgemeines Vorgehen

Das nachfolgende Kapitel beschäftigt sich mit dem schematischen Vorgehen bei der Transformation analoger in digitale, automatisierte Geschäftsprozesse. Es sollen grundlegende Kenntnisse der Prozessmodellierung und der Lösungsentwicklung vermittelt werden.

Das Kapitel gliedert sich in 4 Teile:

1. Lösungsentwicklung und Modellierung eines Prozesses
2. Ausgangspunkt: Analoger Geschäftsprozess
3. Ziel: Digitaler, automatisierter Geschäftsprozesses
4. Automatisierung: Datenverarbeitung innerhalb der Prozessabwicklung

Ordnen wir an dieser Stelle die Thematik dieses Leitfadens noch einmal in einen Geschäftsalltag ein. So gut wie jedes Unternehmen hat ein Warenwirtschaftssystem. Dieses bietet neben einer Datenbank eine Palette an Funktionen an. Mit diesen können die gespeicherten Daten bearbeitet werden. Daneben gibt es, je nach Preis des Systems, weitere Funktionen die auf vorgefertigte Art und Weise Aufgaben erledigen können. Ist ein Prozess unternehmensspezifisch, dann muss zur Entwicklung zumeist ein externer Dienstleister beauftragt werden. Das ist erstens sehr teuer und zweitens auch zeitraubend. Wir wollen solche unternehmensspezifischen Geschäftsprozesse selber konzipieren, entwickeln und implementieren. Selbst wenn Sie nach der Konzipierung feststellen, dass die nachfolgende Programmierung zu anspruchsvoll ist, haben Sie dennoch bereits einen wichtigen Beitrag zur Entwicklung geleistet. Machen Sie sich bewusst, dass ein externer Dienstleister sich immer zunächst mit ihrem Unternehmen beschäftigen muss um einen neuen Prozess optimal in den Geschäftsalltag integrieren zu können. Das kostet zusätzlich Zeit und Geld, obwohl ihre Mitarbeiter das eigene Unternehmen schon bis ins Detail kennen.
Verwandeln Sie dieses bereits bestehende Wissenspotential in eine Erhöhung des Geschäftserfolgs.

2.1. Konzipierung und schematische Modellierung des Geschäftsprozesses

Jedes Unternehmen ist in seinen Prozessen einzigartig. Um dennoch alle wissbegierigen Leser unabhängig ihrer Problemstellung ansprechen zu können, soll ein allgemeines schematisches Vorgehen für die Modellierung von Prozessen vorgestellt werden. Es soll dazu dienen komplexe Problemstellungen in kleine Bausteine zu gliedern und für die Entwicklung von Lösungswegen ein allgemeines strukturiertes Vorgehen an die Hand zu geben. Das allgemeine Schema lässt sich aufgrund der hohen Flexibilität auf jedes individuelle Problem anwenden.
Die Konzeptionsphase eines Prozesses ist der erste und gleichermaßen wichtigste Schritt auf dem Weg zur Entwicklung einer effizienten Lösung. Ein Fehler im Konzept kann sich während der weiteren Arbeitsschritte zu einem großen Problem fortpflanzen. Nehmen Sie sich daher für die Konzipierung ausreichend Zeit. Je besser ein Konzept entwickelt und je tiefer es durchdacht wurde, desto weniger Arbeit haben Sie im weiteren Verlauf der Umsetzung.

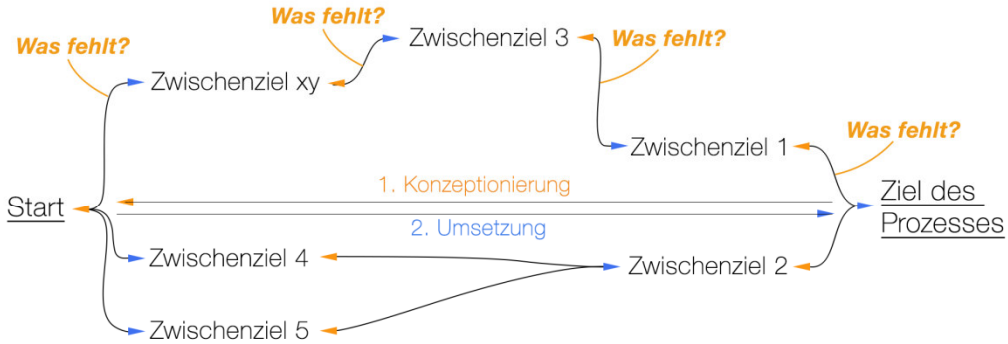
Erfahrungsgemäß stellt die Modellierung des Lösungsweges bei komplexen Aufgabenstellungen eine große Hürde dar. Das Ziel scheint weit weg zu sein und ein Lösungsweg ist nicht direkt erkennbar.

Ein Unternehmen möchte einen neuen Geschäftsprozess entwickeln. während der Entwicklungsphase sind wichtige Fragen zu klären:

- Kann der Prozess überhaupt umgesetzt werden und wie sieht der Ablauf aus?
- Welchen Aufwand bringt die Umsetzung mit sich?
- Wie effizient ist der Prozess?
- Handelt es sich um einen durchweg digitalen Prozess?
- Bietet der Geschäftsprozess Automatisierungspotential?

Konzept der offenen Modellierung

Bei der Beantwortung dieser Fragen kann Ihnen das Konzept der offenen Modellierung behilflich sein. Eine schematische Darstellung ist der Abbildung xy zu entnehmen. Auf das Konzept der offenen Modellierung wird in der Schule gar nicht oder nur unzureichend eingegangen. Es soll in der Mathematik durch Aufgabentypen indirekt vermittelt werden. Den größten Anteil des Lehrinhaltes machen jedoch Routineaufgaben aus. In diesen sind der Start und das Ziel des Problems fest vorgegeben. Die Struktur des Lösungsweges ist für einen gleichen Aufgabentyp identisch. Um solche Probleme zu lösen, müssen die Schüler*innen lediglich bereits bekannte Zusammenhänge abrufen. Das vermittelt jedoch in keiner Weise die Fähigkeit anders gelagerte Probleme lösen zu können. Es ist eine reine Wiedergabe von vorgegebenem Inhalt. Den Schüler*innen wird eine aufgabenbezogene Strategie vorgegeben. Diese gibt die einzelnen Lösungsschritte vor. Was die Schüler*innen aber nicht machen müssen ist sich selbstständig einen neuen Lösungsweg zu überlegen. Es wird ihnen nicht beigebracht, wie sie eigenständig komplexe Aufgabe in logisch aufeinanderfolgende Lösungsschritte zerlegen. Die Fähigkeit unbekannte und anders strukturierte Lösungswege zu entwickeln und zu modellieren wird bei Routineaufgaben nicht gefordert und somit auch nicht gefördert. Dabei trägt das Lernen am Problem einen viel größeren Beitrag zur Entwicklung des strategischen und methodischen Denkens bei. Aus diesem Grund soll dieses enorm wichtige Konzept der offenen Modellierung an dieser Stelle noch einmal anschaulich aufbereitet werden.



Die Methode gliedert sich in zwei Phasen. Zuerst wird der Lösungsweg entwickelt. In Abbildung zeigen die orangefarbenen Pfeile die Richtung der Konzipierung (von rechts nach links: Ziel -> Start). Während der Konzipierung werden die Zwischenschritte des Lösungsweges vom „Ziel des Prozesses“ logisch aufeinander aufbauend in Richtung „Start“ erarbeitet. Dabei wird das Problem automatisch in logisch aufeinander aufbauende und einfach zu lösende Zwischenschritte zerlegt.
indem Sie folgendermaßen Vorgehen:

Fragen Sie sich:

Was ist das Ziel des Prozesses (Zwischenschrittes)?

Wenn sie das Ziel (den Zwischenschritt) identifiziert haben, ist die nächste Aufgabe herauszufinden, welche Information fehlt, um das Ergebnis des Prozesses (oder des Zwischenschrittes) zu schließen. Stellen Sie sich daher die Frage:

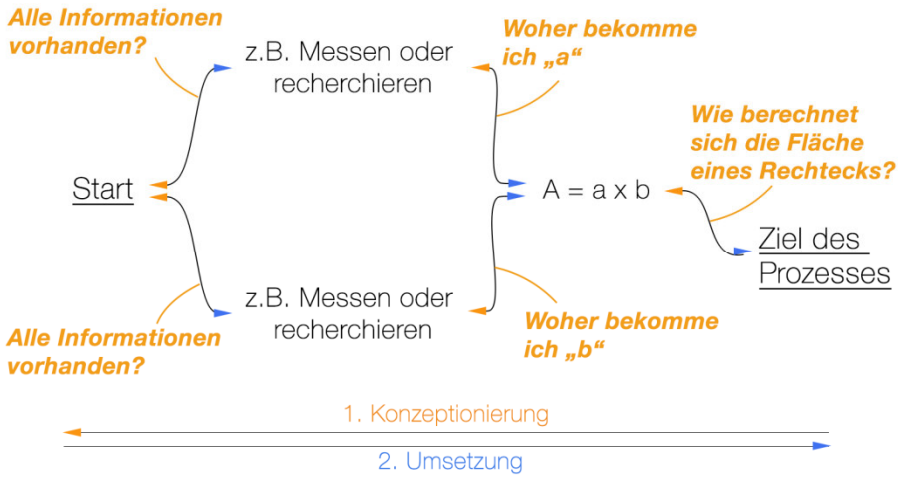
Welche Informationen fehlen um das Ziel (Zwischenziel) zu erreichen?

Umgangssprachlich können Sie auch fragen: „Was fehlt…um die Frage zu beantworten?“
Wiederholen Sie den letzten Schritt solange bis sich keine neuen Lösungsschritte mehr ergeben und alle bisherigen Zwischenziele geschlossen beantwortet werden können. Sämtliche Informationen und Daten die zur Erfüllung aller Zwischenziele notwendig sind sollten am Ende der Konzeptionsphase bekannt sein.

Ist das der Fall, folgt die zweite Phase. Die identifizierten Informationen und Daten werden dem Modell folgend in Richtung der blauen Pfeile zur Lösung (von links nach rechts: Start -> Ziel) der einzelnen Zwischenziele und schließlich zur Lösung des Prozesses zusammenfügt. Das scheint im ersten Moment sehr abstrakt, ist aber für komplexe Probleme ein empfehlenswertes Vorgehen. Das Konzept mutet in der Theorie sehr abstrakt an. Daher wollen wir es nachfolgend in einem Beispiel zur Anwendung bringen. Dieses wird zunächst sehr einfach gehalten sein. Es soll zunächst nur das Vorgehen veranschaulicht werden. Die Vorteile im Rahmen komplexen Aufgaben werden wir an späteren Beispielen besprechen.

Beispiel 1: Aufgabe: Berechnung der Fläche einer Europalette

Es soll die Fläche einer Europalette berechnet werden. Die logische erste Frage wäre: Wie berechnet sich denn die Fläche eines Rechtecks? Man sucht sich also die Formel für die Flächenberechnung eines Rechtecks heraus. Die nächste Frage wäre: Woher bekommt man denn die beiden Variablen „a“ und „b“? Diese kann man entweder durch Messen ermitteln oder durch Recherchieren herausfinden. Danach hat man alle notwendigen Informationen. Man kann nun die
Zwischenziele entlang des entworfenen Pfades wieder zurück in Richtung „Ziel“ gehen. Wie
bereits gesagt ist das Beispiel viel zu einfach um es mittels eines Schemas lösen zu müssen. Aber es veranschaulicht das abstrakte Vorgehen zunächst auf eine sehr einfache Weise.



Beispiel 2: Aufgabe: Automatische Generierung von Speditionssendungslabeln

Das zweite Beispiel ist deutlich umfangreicher. So kann jeder Schritt der Entwicklung am selben Beispiel nachvollzogen werden können. Das Ziel ist es die Daten von Speditionssendungen (Kunden-, Artikel-, Sendungsdaten) automatisch zu verarbeiten. Dazu sind auf Grundlage der eben genannten Informationen zahlreiche Zwischenschritte zu lösen und Entscheidungen zu treffen. Eine übergeordnete Einteilung ist den nachfolgenden Punkten zu entnehmen:

- Bestimmung des Sendungsumfangs (Abmäße, Gewicht)
- Zuordnung der Sendung zu einem Versandtarif (Entscheidung anhand von Abmaß, Gewicht und Versandpreis der verschiedenen Speditionsdienstleister)
- Überprüfung, Berichtigung und gegebenenfalls Aussortierung von fehlerhaften Datensätzen
- Erstellung eines Sendungsdatensatzes anhand der Vorgaben des Speditionsdienstleisters

Entwickelt man diesen Prozess nun nach dem oben vorgeschlagenen Schema, werden die
Vorteile schnell deutlich. Der hohe Umfang der zu berücksichtigenden Informationen macht es ungemein anspruchsvoll während der Konzeptionsphase alle erforderlichen Details zu
berücksichtigen. Genau an dieser Stelle zeigt sich einer der Vorteile. Aufgrund dessen, dass man das Konzept vom finalen Prozess aus rückwärts entwickelt, stößt man zwangsläufig immer auf alle fehlenden Informationen. Auch alle eventuell entstehenden konzeptionellen Lücken werden so identifiziert und können umgehend geschlossen werden. Nachdem das Konzept erfolgreich entwickelt wurde, können wir uns nun Gedanken über die Automatisierung machen.

Vorteile des Konzeptes der offenen Modellierung

Nach der Untersuchung von zwei verschieden gelagerten Beispielen gestaltet sich das Konzept schon anschaulicher. Die folgenden Charakterisierungen können ausgehend von den erarbeiteten Ergebnissen abgeleitet werden:

Das Konzept…

- **…ist simpel.** Die simple und feste Struktur des Frage-Antwort-Prinzips hält den Umfang der zwischengelagerten Lösungsschritte einfach und überschaubar.
- **…folgt immer dem gleichen Ablauf (Start-Ziel, Frage-Antwort).** Unabhängig von der Komplexität der Aufgabe bleibt das Vorgehen unverändert einfach. Taucht während der Entwicklung des Lösungsweges eine neue Barriere auf, wissen Sie sofort was zu tun ist: Frage -> Antwort.
- **…ist universell anwendbar.** Wenn es eine Lösung für ein Problem gibt, kann sie auf diesem Wege gefunden werden.
- **…ist flexibel.** Welche Natur die Lösung eines Zwischenschrittes aufweist, ist vollkommen ohne Belang. Es kann eine Zahl sein, eine Gleichung oder auch nur eine Ja/Nein Entscheidung.
- **…ist effizient.** Bei der Entwicklung und Lösung der Zwischenschritte werden ausschließlich relevante Informationen berücksichtigt. Das spart Entwicklungszeit und hält den Lösungsweg schlank. Die Art des Vorgehens (Frage-Antwort) sorgt automatisch für eine übersichtliche Struktur des Modells.
- **…ist strukturgebend.** Durch die Entwicklung der Zwischenschritte entsteht ein nachvollziehbarer und übersichtlicher Lösungsweg. Die Zwischenziele stellen Prozessknoten dar, sodass auch nachträglich Anpassungen des Lösungsweges vorgenommen werden können.

2.2. Ausgangspunkt: Analoger Geschäftsprozess

Bleiben wir weiter nah an der Praxis und fahren mit einem weiteren Beispiel fort. Ein Händler verkauft eine Vielzahl an Produkten. Alle Informationen zu einem Artikel werden in einem Warenwirtschaftssystem verwaltet, bearbeitet und gespeichert. Der Unternehmer möchte zur Kalkulation des Verkaufspreises den Deckungsbeitrag berechnen. Eine erschwingliche Branchenlösung eines Warenwirtschaftssystems errechnet zumeist nur den Deckungsbeitrag aus einem gespeicherten Einkaufs- und Verkaufspreis. Innerbetriebliche Kosten wie z.B. Einlagerung, Weiterverarbeitung, Verpackung, Versand und weiterführende betriebliche Kosten (Support, Reklame, etc) werden dagegen nicht berücksichtigt. Die beispielhaft genannten Produktionskosten bilden jedoch einen signifikanten Anteil des Deckungsbeitrags und sind zur Kalkulation des Verkaufspreises entscheidend.

Der einfache aber ungenaue Weg ist, alle Kosten zusammen zu rechnen und auf alle Artikel gleichermaßen umzulegen. Das führt zu einer Verwässerung der unterschiedlichen Anteile des Deckungsbeitrages. Scheinbar rentable Artikel verursachen in der Produktionskette unbemerkt Kosten. Sie werden in der Gesamtbilanz von rentablen Artikel getragen, wodurch es im Nachgang nicht mehr möglich ist den Deckungsbeitrages eines Produktes realistisch anzugeben.

Bei einer schmalen Angebotspalette und einem kurzen Produktionsweg lässt sich der Deckungsbeitrag vielleicht noch händisch berechnen. Doch ab einer gewissen Produktvielfalt sowie einem längeren Produktionsweg wird die Sache schnell sehr aufwendig. Natürlich wissen erfahrene Händler welches Produkt wertvoll für das Unternehmen ist und welches man lieber aus dem Sortiment streicht. Doch bedarf es sehr viel Erfahrung um mit hoher Sicherheit eine solche Bewertung ohne eine genaue Rechnung vornehmen zu können. Das Unternehmen begibt sich dadurch in eine gefährliche Abhängigkeit von seinem Personal. Der Händler möchte das vermeiden und will daher einen Geschäftsprozess entwickeln, der schnell und vollumfänglich den Deckungsbeitrag aller Artikel seiner Produktpalette liefert.

Wir werden uns das Vorgehen für die Entwicklung des Geschäftsprozesses an einem Beispiel erarbeiten. Wir betrachten einen Artikel „A“. Der Einkaufspreis soll 10€ betragen. Wir wollen nachfolgend den mehrstufigen Deckungsbeitrag kalkulieren. Das Beispiel soll exemplarisch und verständlich sein. Wir werden nicht bis ins letzte Detail gehen, sondern nur die wichtigsten Kosten berücksichtigen, die vermutlich in jedem Unternehmen mit einem ähnlichen Produktionsablauf anfallen. Vieles wird beim Lesen eine Wiederholung von betriebswirtschaftlichem Grundwissen sein. Wir entwickeln schließlich einen Geschäftsprozess. Nur gehen wir

dann einen Schritt weiter und digitalisieren und automatisieren diesen. Für die Entwicklung müssen wir jedoch bei den teilweise trockenen Grundlagen beginnen. Gleichermaßen werden wir das Beispiel nutzen, um das oben vorgestellte Vorgehen zur Lösungsentwicklung und Modellierung eines Prozesses weiter in die praktische Anwendung zu bringen. Erfahrungsgemäß empfiehlt es sich den ersten Entwurf handschriftlich zu erstellen. Es geht noch nicht um die Form. Bevor man sich am PC in der Formatierung verliert, ist es immer ratsam erste Entwürfe mit Papier und Stift zu machen.

Die nachfolgend betrachteten Kosten entstehen in vielen Unternehmen. Wie eingangs angedeutet ist jedes Unternehmen individuell. Sehen Sie die Beispiele daher als einleitende, einfache Vorschläge. Der Umfang der einfließenden Kostenparameter und die Komplexität der Berechnung lässt sich ohne weiteres für anderes gelagerte Problemstellungen erweitern. Versuchen sie beim Lesen der nachfolgenden Kostenbeispiele schon einmal zu berücksichtigen, dass alle Informationen die zur Berechnung der Kosten notwendig sind, bereits im Warenwirtschaftssystem gespeichert sind. Würden Sie ein externes Unternehmen beauftragen diesen Prozess für Sie zu entwickeln, würde dieses auch nichts Anderes machen als mit ihren Daten zu arbeiten. Nutzen sie, genau wie das Wissen ihrer Mitarbeiter, auch dieses Datenpotential lieber selbst um ihr Unternehmen voran zu bringen. Wir werden uns gemeinsam anschauen was dafür notwendig ist.

Die Abbildung zeigt die schrittweise Entwicklung der Lösung des Prozesses. Das Ziel unseres Beispiels war die Berechnung des mehrstufigen Deckungsbeitrages. Weiter oben wurde bereits gezeigt, dass es gilt sich die richtigen Fragen zu überlegen. Beispielsweise wäre die erste gute Frage:

Aus welchen Kosten setzt sich der mehrstufige Deckungsbeitrag zusammen?

Anschließend gilt es eine Antwort darauf zu finden. Wir überlegen uns also welche Kosten ein Artikel vom Wareneingang bis zum Warenausgang verursacht. Es bietet sich an, sich bei der Kategorisierung der Kosten am Produktionsfluss zu orientieren. Damit ergibt sich die folgende Einteilung:

- Lagerkosten
- Verpackungskosten
- Versandkosten
- Gemeinkosten

Die Summe aus diesen Kosten ergibt den mehrstufigen Deckungsbeitrag. Das weitere Vorgehen ist eine konsequente Wiederholung. Es gilt die erarbeiteten Zwischenschritte zu lösen.

Formulieren Sie dazu wieder eine sinnvolle Frage. Suchen Sie nach einer Antwort. Wenn sich aus der Antwort neue Zwischenschritte ergeben, dann suchen Sie wieder eine Lösung und so weiter. Sie sind fertig, wenn auf alle Fragen eine geschlossene Antwort vorliegt. Wir haben die 4 Kostenfaktoren des Deckungsbeitrages identifiziert und wollen nachfolgend das Frage-Antwort- Spiel üben. Ein kleiner Tipp: Je besser die Frage, desto einfacher wir es eine Antwort zu finden.

Lagerkosten
An dieser Stelle üben wir gleich noch einmal das eben erarbeitete Vorgehen zur Lösungsentwicklung. Wir wollten zunächst eine Frage (Abbildung: F-1) formulieren, die zur Lösung des Ziels (oder Zwischenziels) führt. Diese könnte im vorliegenden Fall lauten:

Wie errechnen sich die Lagerkosten für einen Artikel?

Die Antwort (Abbildung: A-1) auf die Frage 1 findet sich sehr schnell. Umgangssprachlich könnte man formulieren: Eine gewisse Anzahl eines Artikels steht für eine gewisse Zeit auf einem Lagerplatz der eine. bestimmte Betrag kostet. Es fließen daher 3 Variablen in die Gleichung ein: die Standzeit, die monatlichen Kosten für einen Lagerplatz und die Anzahl der Artikel auf einem Lagerplatz.

Lagerkosten pro Artikel = Kosten pro Lagerplatz & Monat / Kubatur der Artikel pro Lagerplatz * Standzeit

Wir haben damit eine Antwort (A-1) gefunden, wie sich die Lagerkosten für einen Artikel berechnen. Wir müssen nun schauen, ob alle Variablen der Gleichung bekannt sind. Wenn die Informationen vorliegen, können wir sie als bekannt deklarieren. Falls sie jedoch noch unbekannt sind, müssen Sie den ersten Schritt einfach wiederholen und weitere Fragen formulieren.

Beginnen wir mit den Kosten pro Lagerplatz & Monat. Genau wie bei einer Mietwohnung kann der Preis pro Quadratmeter als Quotient aus Kosten und Fläche berechnet werden. Je nach Typ des Lagers bietet es sich an eine andere Bezugsgröße zu wählen. Beispielsweise ist die reine Grundfläche bei einem Hochwarenregallager als Bezugsgröße unzweckmäßig. Hier bietet sich eher die Anzahl der Lagerplätze an. Wir halten daher für unser Beispiel fest:

Kosten pro Lagerplatz & Monat = Gesamtkosten pro Monat / Anzahl der Lagerplätze

Die Anzahl der Lagerplätze setzen wir als bekannt voraus. Die monatlichen Gesamtkosten wollen wir zumindest noch einmal aufschlüsseln um eventuell versteckte Kosten zu identifizieren. Die nächste richtige Frage (F-6) wäre daher:
Welche Kosten fallen für das Lager an?

Das ist in jedem Unternehmen anders. In unserem Beispiel fallen die folgenden Kosten (A-6) an: Miete, Zins auf langfristige Darlehen, Personal, Reparaturen, Rücklagen für Investitionen und Zinskosten auf das in Form von Ware gebundene Kapital. Diese Positionen müssen durch den Ertrag des Artikels gedeckt werden. Übersteigen die Kosten der Lagerhaltung für einen Artikel dessen Ertrag, so ist er unrentabel. Die regelmäßigen Kosten für Miete und Personal sollten vollumfänglich bekannt sein. Unregelmäßige Kosten für Reparaturen können ausgehend von bisherigen Reparaturmaßnahmen geschätzt oder zunächst durch Recherche und Abschätzung festgelegt werden. Rücklagen für zukünftige Investitionen können mittels einer vorherigen Kalkulation sinnvoll durch einen monatlichen Betrag berücksichtigt werden (Abschriften). Die letzte Position sind die Zinskosten. Die eingelagerte Ware stellt gebundenes Kapital dar. Die entgangenen Zinsen sind bei der Berechnung des Deckungsbeitrags als Kosten zu berücksichtigen. Im Hinblick auf die Liquidität ist es daher ratsam, eher auf Artikel mit einer hohen Absatzgeschwindigkeit zu setzen. Auch wenn die Marge des Artikels „A“ vielleicht auf den ersten Blick weniger vielversprechend wirkt als eine deutlich höhere Marge eines Artikels „B" Wir können die Gesamtkosten daher als bekannt annehmen. Die Anzahl der Lagerplätze (F-7) ist in unserem Fall ebenfalls bekannt (A-7). Die monatlichen Kosten für einen Lagerplatz sind nun vollumfänglich kalkulierbar.

Schauen wir uns als nächstes die m^3 der Artikel pro Lagerplatz an. Da die Produkte unterschiedlich groß sind, variiert das Volumen auf einem Lagerplatz.

Zuletzt untersuchen wir noch die Verweilzeit des Artikels im Lager. Jeder kennt die klassischen Ladenhüter. Je länger ein Artikel eingelagert wird, desto höher sind die Lagerhaltungskosten.

Für die Kalkulation der Lagerkosten pro Artikel konnten wir alle gestellten Fragen beantworten und die erforderlichen Informationen bereitstellen. Wir führen das Rechenbeispiel des Artikels „A“ fort. Der Lagerbestand des Artikels „A“ beträgt 100 Stück. Auf einen Lagerplatz passen 10 Stück. Ziel ist es, den minimalen Anteil am Deckungsbeitrag des Artikels für dessen Lagerhaltungskosten zu berechnen.

Es sind bei Wareneingang 10 Lagerplätze mit diesem Artikel belegt. Die laufenden Kosten für einen Lagerplatz nehmen wir mit 0,33€ an (siehe Tabelle). Bei einer geplanten Absatzmenge von 3 Stück pro Monat und unter Berücksichtigung einer linearen Abnahme der Lagerhaltungskosten bis zum Ausverkauf des Artikels „A“, ergibt sich der notwendige Anteil am Verkaufspreis zu 2.75€ pro Stück.

Positionen	Einheit	Zahlenbeispiel Artikel „A“
Lagergröße	[Palettenstellplätze]	4500
Laufende Lagerkosten (Miete, Reparatur, etc.)	[€/Monat]	€ 1,500.00
Laufende Kosten je Lagerplatz	[€/Monat]	€ 0.33
Nettowarenwert	[€]	€ 10.00
Zinskosten auf gebundenes Kapital (5%)	[€]	€ 0.50
Lagerbestand Artikel „A“	[Stück]	100
Artikelmenge pro Lagerplatz	[Stück]	10
Absatzmenge	[Stück/Monat]	3
Durchschnittlicher Deckungsbeitrag Lagerhaltungskosten bis zum Ausverkauf	[€/Stück]	€ 1.11

Kann diese Menge nicht verkauft werden, muss beispielsweise über eine Anpassung des Verkaufspreises der Absatz erhöht werden. Sollte der Artikel weiterhin unrentabel sein, gilt es ihn schnellstmöglich los zu werden. Ob ein Artikel mindestens den Deckungsbeitrag für seine Lagerhaltungskosten erwirtschaftet, ist sicher noch recht schnell zu errechnen, wenn man die Lagerkosten kennt. Jedoch soll der Deckungsbeitrag immer aktuell für jeden Artikel im Lager mit allen Kostenpositionen zur Verfügung stehen. Zusätzlich ist die von uns aufgestellte Rechnung recht einfach gehalten. In der Praxis müssen zusätzliche Einflüsse berücksichtigt werden. Die Fülle der Kosten macht es schon erheblich aufwendiger zu einem Ergebnis zu kommen. Wenn Sie die Zeit eines Mitarbeiters beanspruchen, haben sich vermutlich sämtliche Kostenpositionen schon wieder geändert, wenn er fertig ist. Der Prozess muss daher sehr schnell sein. Genau liegt einer der Vorteile der Automatisierung eines Geschäftsprozesses. Dann spielt es keine Rolle mehr, für wie viele Produkte die Rechnung durchgeführt wird oder wie häufig.

Verpackungskosten

Wir haben den durch lauf im Frage-Antwort-Spiel jetzt zwei Mal geübt und sollten schon ein wenig sicherer sein. Beginnen Sie wieder damit sich eine sinnvolle Frage zu überlegen.

Wie berechnen sich die Verpackungskosten für einen Artikel?

In unserem Beispiel möchten wir exemplarisch 3 Positionen betrachten: das Verpackungsmaterial, die Maschinenkosten und die Personalkosten. Die Antwort lautet daher:

Verpackungskosten pro Artikel = Materialkosten + Personalkosten + Maschinenkosten

Beginnen wir mit den Materialkosten. Diese hängen fast ausschließlich von den Abmaßen eines Artikels und dem Verpackungsdesign ab. Ein großer Artikel benötigt selbstverständlich viel mehr Verpackungsmaterial als ein kleiner Artikel. Manche Produkte sind vom Hersteller bereits versandfertig verpackt. In diesem Fall entfällt sowohl die Packzeit als auch das Verpackungsmaterial und damit ein signifikanter Teil der Personalkosten. Egal ob Sie die Verpackung mit einer Spezialmaschine auf das Produkt zuschneiden oder verschieden große Einheitskartonagen verwenden, je größer das Produkt, desto mehr Pappe wird benötigt. Es gilt also eine Formel für die benötigte Menge Pappe herzuleiten. Bei der Verwendung von Schnittmustern liegt eine solche Gleichung meist schon vor.

Die Abmaße des Artikels sind im Warenwirtschaftssystem gespeichert. Die Materialkosten berechnen sich durch den Materialeinsatz multipliziert mit dem Quadratmeterpreis der Pappe.

Ein wesentlicher Anteil der Kosten wird durch die Packzeit verursacht. diese ist für jeden Artikel unterschiedlich. Dennoch ist es schwierig für jedes Produkt eine feste Packzeit festlegen zu können. Solange ihre Packstraße nicht vollautomatisch betrieben wird, ist es empfehlenswert mit Durchschnittszeiten zu rechnen. Sollte ein Artikel deutlich länger dauern als andere, kann man für diesen speziellen Artikel eine eigene reale Packzeit messen.
Dies kann mittels eines Tests der Packzeit von 2 bis 3 Mitarbeitern und der Berechnung des Durchschnittswertes ermittelt werden. Sendungen die überhaupt nicht verpackt werden müssen, sollten auch eine eigene reduzierte Bearbeitungszeit erhalten. Die Packzeit sollte ein artikelspezifischer Parameter sein und ebenfalls im Warenwirtschaftssystem gespeichert werden. Die Personalkosten für einen Artikel errechnen sich aus dem Zeiteinsatz multipliziert mit den Personalkosten.

Die letzte Kostenposition in der Verpackung ist die Maschine. Der finanzielle Einsatz für die Maschine sollte pauschal über alle Aufträge gehen, die verpackt werden. Ein Vorschlag wäre dabei mit der bereits gespeicherten Packzeit zu arbeiten. Wenn in den Attributen eines Artikels eine Packzeit gespeichert ist, dann wird er für die Kalkulation der Maschinenkosten berücksichtigt. Die Gesamtkosten für den Betrieb der Maschine wird auf die Anzahl der Aufträge die verpackt werden aufgeteilt.

Maschinenkosten pro Artikel = Monatliche Maschinenkosten / Verpackte Aufträge
Wir wollen unser Beispiel fortführen. Die Tabelle enthält die relevanten Parameter zur Kalkulation des Beitrags der Verpackung zum Deckungsbeitrag des Artikel „A“. Das Ergebnis setzt sich aus der Summe der Kostenpositionen für die Verpackung (Material-, Zeit-, und Maschinenkosten) zusammen. Das Zustandekommen der Einzelkosten ist anhand der Formeln nachzuvollziehen.

Positionen	Einheit	Zahlenbeispiel Artikel „A“
Materialeinsatz	[qm]	1.5
Quadratmeterpreis Verpackungsmaterial	[€/qm]	€ 0.20
Materialkosten	[€]	€ 0.30
Packzeit	[Minute]	1
Personalkosten	[€/Minute]	€ 0.17
Personalkosten	[€]	€ 0.17
Maschinenkosten pro Artikel (gemittelt)	[€/Stück]	€ 0.05
Deckungsbeitrag Verpackungskosten	[€/Stück]	€ 0.52

Versandkosten

Frage: Wie errechnen sich die Versandkosten?

Versandkosten pro Artikel = Sendungskosten pro Artikel + Personalkosten pro Artikel

Die Versandkosten sind ein relativ überschaubarer Posten. Jedes Paket wird anhand der Größe und des Gewichts in eine Versandkostenkategorie eingeordnet. Die Personalkosten umfassen lediglich die Beladung der Paketbrücken. Die Zeit die diese Aufgabe erfordert muss über eine angemessene Dauer gemessen und durch die Anzahl der Sendungen geteilt werden. Ein sinnvoller Zeitraum wäre zum Beispiel 1 Monat, während dieser Zeit werden durchschnittlich 18.000 Pakete versendet. Damit kann der Zeitaufwand pro Paket ermittelt werden. Für zukünftig steigende Sendungsmengen können dann die Personalkosten bequem kalkuliert werden, da sie pro Packstück ermittelt wurden. Die Tabelle erweitert unser Beispiel des Artikels „A“ um die Versandkosten.

Schritte: Import, Verarbeitung und Export. Die nachfolgende Tabelle fasst die wichtigsten Merkmale der drei genannten Bausteine kompakt zusammen.

Positionen	Einheit	Zahlenbeispiel Artikel "A"
Versandkosten/Stück	[€]	6,15
Zeitaufwand Brückenbeladung /Monat	[Stunden]	40
Anzahl Sendungen /Monat	[Stück]	18000
Personalkosten/Monat	[€]	2100
Deckungsbeitrag Verpackungskosten	[€/Stück]	6,18

Gemeinkosten

Im Sinne der Übersichtlichkeit unseres Beispiels vereinfachen wir den eigentlichen Umfang und beschränken die Gemeinkosten auf die personellen Aufwendungen, die direkt mit der Produktionskette verknüpft sind und keinem der bisherigen Positionen zugeordnet werden können. Zu Beginn wieder die geeignete Frage:

Wie errechnen sich die Gemeinkosten?

Die Personalkosten müssen auf alle Sendungen verteilt werden. Die Antwort auf die Frage nach den Gemeinkosten lautet daher:

Gemeinkosten pro Artikel = Personalkosten / Anzahl Aufträge

In dieser Gleichung sind die Personalkosten noch unbestimmt. Sicher überlegen Sie sich schon ganz intuitiv die nächste Frage. Wie setzen sich die Personalkosten zusammen. Antwort: Sie ergeben sich aus der Summe der Kosten für die Bereiche Disposition, Einkauf und Support.

Personalkosten = Personal Disposition + Personal Einkauf + Personal Support

Für die Anzahl der Aufträge legen wir wieder das durchschnittliche Paketvolumen eines Monats von 18.000 Sendungen zu Grunde. Wir erhalten damit zusätzliche Kosten von 59 Cent pro Artikel.

Personalkosten	**Einheit**	**Zahlenbeispiel Artikel "A"**
Disponent	[€]	2500
Support (Zwei volle Stellen)	[€]	4400
Einkauf	[€]	3800
Anzahl Aufträge/Monat	[Stück]	18000
Deckungsbeitrag Verpackungskosten	[€/Stück]	0,59

Zusammenfassung der Kostenpositionen

Wir haben nun den Prozess entwickelt und alle notwendigen Informationen und Zusammenhänge identifiziert. Die übergeordneten Kostenfaktoren wurden im Verlauf bereits einzeln berechnet. Zusätzlich zum Einkaufspreis des Artikels setzt sich der Deckungsbeitrag wie in Tabelle dargestellt zusammen. Abschließend fügen selbige zusammen und erhalten so den gesuchten mehrstufigen Deckungsbeitrag des Beispielartikels „A“.

Kostenposition (pro Artikel) Einheit Zahlenbeispiel Artikel „A“
Lagerhaltungskosten [€] € 1.11
Verpackungskosten [€] € 0.52
Versandkosten [€] € 1.39
Gemeinkosten [€] € 0.58
Mehrstufiger Deckungsbeitrag [€/Stück] € 5.22

Der errechnete Deckungsbeitrag von 5.22€ muss mindestens auf den Einkaufspreis addiert werden, damit das Unternehmen die Betriebskosten decken kann.

2.3 Ziel: Digitaler Prozess

Das Ziel der Umwandlung eines analogen in einen digitalen Prozess besteht nicht allein in der Abbildung bestehender Abläufe auf einer technischen Plattform. Vielmehr geht es darum, die Effizienz zu steigern, Fehlerquellen zu minimieren und die Prozessqualität insgesamt zu erhöhen. Ein digitaler Prozess zeichnet sich durch strukturiert erfassbare Daten, klar definierte Schnittstellen sowie automatisierte Übergänge zwischen einzelnen Bearbeitungsschritten aus.

Digitale Prozesse bieten die Möglichkeit, Informationen in Echtzeit zu erfassen, zu verarbeiten und weiterzugeben. Dadurch können nicht nur interne Arbeitsabläufe beschleunigt, sondern auch externe Partner und Systeme besser eingebunden werden. Ein wesentliches Merkmal eines digitalisierten Prozesses ist die Reproduzierbarkeit und Nachvollziehbarkeit der einzelnen Arbeitsschritte. Jeder Vorgang ist eindeutig dokumentiert, was die Qualitätssicherung sowie die spätere Auswertung und Optimierung wesentlich erleichtert.

Darüber hinaus ermöglichen digitale Prozesse eine höhere Skalierbarkeit. Während analoge Prozesse bei zunehmendem Volumen oft an personelle oder organisatorische Grenzen stoßen, lassen sich digitalisierte Abläufe einfacher erweitern, kopieren oder an veränderte Rahmenbedingungen anpassen. Dies macht Unternehmen flexibler und reaktionsfähiger in einem zunehmend dynamischen Marktumfeld.

2.4 Automatisierte Datenverarbeitung innerhalb der Prozessabwicklung

Die automatisierte Datenverarbeitung bildet das funktionale Herzstück eines jeden digitalisierten Geschäftsprozesses. Aufbauend auf der zuvor durchgeführten Konzipierung und Modellierung sowie der klaren Zieldefinition im Rahmen des digitalen Soll-Prozesses erfolgt in dieser Phase die tatsächliche technische Umsetzung der Prozesslogik. Sie dient dazu, Eingabedaten strukturiert aufzunehmen, logisch zu verarbeiten und daraus programmatisch gesteuerte Abläufe zu entwickeln, die ohne manuelles Eingreifen funktionieren. Ziel ist es, auf Basis eines stabilen Regelwerks eine fehlerfreie, wiederholbare und nachvollziehbare Verarbeitung sicherzustellen – unabhängig von Datenvolumen, Zeitdruck oder operativen Störungen. Die automatisierte Datenverarbeitung übersetzt damit die theoretisch geplanten Prozessschritte in ausführbaren Code, der sich flexibel in unterschiedliche Anwendungsszenarien integrieren lässt.

Ziel ist es, die im Modell beschriebenen Bearbeitungsschritte systemseitig abzubilden – nachvollziehbar, wiederholbar und fehlerfrei. Damit dieser Anspruch erfüllt werden kann, muss der Übergang von der modellhaften Beschreibung in ein ausführbares Regelwerk systematisch erfolgen. Dazu gehören unter anderem die Trennung von Eingabedaten und Verarbeitungslogik, die Festlegung klarer Bedingungen für Entscheidungen sowie der Aufbau wiederverwendbarer Funktionsbausteine. Diese Modularisierung erleichtert nicht nur die Wartung und Weiterentwicklung der Prozesse, sondern sorgt auch für Transparenz in der Ablauflogik und reduziert potenzielle Fehlerquellen in der späteren Anwendung.

In der betrieblichen Praxis beginnt die automatisierte Verarbeitung meist mit dem Einlesen strukturierter Eingangsdaten – beispielsweise aus Bestellsystemen, Warenwirtschaft oder CRM-Plattformen. Diese Daten werden anschließend durch ein zentrales Regelwerk verarbeitet, das vordefinierte Logiken, Prüfungen und Berechnungen anwendet. Dies kann von einfachen Formatprüfungen über komplexe Preis- oder Versandkalkulationen bis hin zu automatisierten Zuordnungen in Lager- oder Logistikprozesse reichen. Die resultierenden Ergebnisse werden entweder direkt in Folgeprozesse übergeben oder in weiterverarbeitbare Formate (z. B. Etiketten, XML-Dateien oder Datenbanken) exportiert.

Ein zentrales Charakteristikum der automatisierten Datenverarbeitung ist die vollständige Entkopplung vom menschlichen Eingreifen: Sobald die Eingangsdaten vorliegen und ein definierter Auslöser – etwa ein Zeitintervall, ein Ereignisstatus oder ein externer Trigger – eintritt, beginnt die selbstständige Abarbeitung des Prozesses. Innerhalb dieser vollautomatischen Verarbeitung übernehmen Programme sämtliche Bearbeitungsschritte, die zuvor durch Menschen durchgeführt wurden: Sie prüfen auf Vollständigkeit und Plausibilität, führen berechnete Verzweigungen aus, transformieren oder konsolidieren Daten, speichern Zwischenergebnisse und übermitteln Resultate an nachgelagerte Systeme. Der gesamte Ablauf läuft dabei in exakt derselben Weise ab – unabhängig davon, ob er einmal täglich oder tausendfach pro Stunde ausgeführt wird.

Um diese Automatisierung sicher und robust zu gestalten, sind entsprechende Kontrollmechanismen und Fehlerbehandlungen integraler Bestandteil jeder Datenverarbeitungseinheit. Validierungen bei der Datenübernahme stellen sicher, dass nur vollständige und korrekt formatierte Informationen in den Verarbeitungsprozess gelangen. Zusätzlich sollten Ausnahmeregeln definiert werden, die bei Abweichungen greifen – etwa die automatische Benachrichtigung zuständiger Stellen bei kritischen Fehlern oder das Anhalten eines Teilprozesses zur späteren manuellen Prüfung. Auch Logging-Funktionen, also die systematische Protokollierung von Prozessschritten und -ergebnissen, sind essenziell, um die Nachvollziehbarkeit und Transparenz der Automatisierung sicherzustellen.

Darüber hinaus ist die Automatisierung ein Enabler für Reporting- und Auswertungsfunktionen. Jeder verarbeitete Schritt kann dokumentiert, versioniert und für spätere Analysen gespeichert werden. So entsteht eine lückenlose Prozesshistorie, die es erlaubt, Ursachen für Abweichungen nachzuvollziehen, Engpässe frühzeitig zu erkennen oder systematisch Qualitätsverbesserungen einzuleiten. Besonders im Rahmen von Audits oder bei der Einhaltung von Compliance-Vorgaben ist diese Nachvollziehbarkeit von unschätzbarem Wert. Gleichzeitig bilden diese Daten die Grundlage für kontinuierliche Optimierungsprozesse – sowohl auf technischer Ebene (z. B. durch Performance-Messungen) als auch auf organisatorischer Ebene (z. B. durch die Identifikation ineffizienter Abläufe).

Schließlich ist die automatisierte Verarbeitung auch Voraussetzung für die nächste Stufe der Prozessdigitalisierung: die Integration selbstlernender Systeme und künstlicher Intelligenz. Nur wenn die grundlegenden Prozessschritte konsistent, sauber dokumentiert und technisch durchgängig umgesetzt sind, lassen sich darauf aufbauend adaptive Algorithmen einsetzen, die Muster erkennen, Vorhersagen treffen oder Entscheidungen vorschlagen können. Damit wird die Automatisierung nicht nur zum Werkzeug operativer Effizienz, sondern zur strategischen Grundlage datengetriebener Unternehmensentwicklung.

Gleichzeitig erfordert die Etablierung automatisierter Prozesse eine kontinuierliche Weiterentwicklung sowohl der technischen als auch der organisatorischen Rahmenbedingungen. Schnittstellen müssen regelmäßig auf Kompatibilität geprüft, Sicherheitsstandards aktualisiert und Prozesse flexibel an neue Anforderungen angepasst werden können. Dies gelingt nur, wenn Automatisierung nicht als einmaliges Projekt, sondern als dynamischer Bestandteil einer lernenden Organisation verstanden wird. Die Investition in automatisierte Datenverarbeitung ist damit nicht nur ein Effizienzgewinn, sondern ein strategisches Bekenntnis zur nachhaltigen digitalen Transformation des Unternehmens.

3. Grundlagen der Programmstruktur

Bisher haben wir in diesem Kapitel das Konzept der offenen Modellierung kennen gelernt und einen Lösungsweg für die Kalkulation eines Beispielartikels „A“ modelliert. Der nächste Schritt ist, dass wir das Modell in einen Programmcode umsetzen. Dazu wollen wir uns zunächst einen Überblick über den Aufbau des Codes verschaffen.

3.1. Import der Daten

Der Input der Daten ist der erste Schritt während eines Programmdurchlaufs. Der Import stellt die Schnittstelle zum Programm dar. Über diese werden Daten zur Verarbeitung eingespeist. Datensätze können Fehler aufweisen, welche im Zuge des Imports herausgefiltert werden müssen. Unsaubere Daten führen in den allermeisten Fällen zum Absturz der Software. Im nächsten Kapitel werden wir uns mit der Fehlerabsicherung beim Programmieren beschäftigen. Eine solche Absicherung gegen Fehler führt meistens dazu, dass der Datensatz übersprungen und ohne Verarbeitung aussortiert werden muss. Das ist nur bei groben Fehlern sinnvoll. Handelt es sich dagegen um kleinere Unsauberkeiten, wird empfohlen diese schon beim Import zu korrigieren. Das steigert die Stabilität der Software signifikant und trägt zu besseren Resultaten bei. Die bisher behandelten Beispiele haben uns schon gezeigt, wo ein weiterer Vorteil des Konzeptes der offenen Modellierung liegt. Auf der linken Seite der Abbildung … sind alle entwickelten Zwischenschritte von bekannten Informationen und Daten abhängig. Durch die Verwendung des Konzeptes der offenen Modellierung haben wir am Schluss der Entwicklung des Lösungsweges automatisch alle Informationen und Daten identifiziert, welche zur Lösung des Problems notwendig sind. Weiterhin haben wir dadurch sichergestellt, dass die Verfügbarkeit gewährleistet ist. Wäre bei einer Information noch unklar wie sie bereitgestellt werden kann, hätten wir weitere Zwischenschritt entwickeln müssen. Das hätten wir so lange machen müssen,
bis die neuen Zwischenschritte alle durch verfügbare Informationen gelöst werden können.

2 Grundlegendes Vorgehen
Es gibt verschiedene Möglichkeiten auf Daten in einer Datei zuzugreifen. CSV-Dateien eignen sich sehr gut um große Datenmengen zur Verfügung zu stellen. Sollte Ihnen der Begriff „CSV-Dateien“ bisher noch nicht begegnet sein, dann hier eine kurze Erläuterung:
Eine CSV Datei (Comma-separated values) ist eine Textdatei die ausschließlich einfache Zeichenketten beinhaltet. Innerhalb der Datei erfolgt eine feste logische Trennung der Daten mittels verschiedener Sonderzeichen wie zum Beispiel:

- Kommas
- Semikolons
- Tabstopps
- Leerzeichen
- etc.

So können zum Beispiel Tabellenkalkulationsprogramme (Excel, Numbers, etc.) jedes Auftreten eines Kommas zwischen zwei Werten als den Beginn einer neuen Spalte interpretieren. Gleichzeitig ist diese Art der Datei aufgrund der ausschließlichen Verwendung von Zeichenketten so einfach gehalten, dass sie universell verwendbar ist. Aus diesem Grund spielt sie bei der Programmierung eine sehr bedeutende Rolle, denn „vor“ und „hinter“ dem eigentlichen Programmcode müssen Daten bereitgestellt werden, was in Form eben beschriebener CSV Dateien bewerkstelligt werden kann. An dieser Stelle ein kleines Beispiel aus der Praxis: Die Übertragung von Produktdaten an einer Schnittstelle eines Verkaufsportals wie z.B. Amazon mit solchen CSV Dateien. Es gibt unzählige weitere Beispiele, auf die an dieser Stelle jedoch nicht eingegangen werden soll. Wenn Sie einmal darauf achten, wird Ihnen dieses Dateiformat an vielen Stellen begegnen.

SPEDITION.CSV

```
1  30678282;22.06.2021;Giovanni;Costanzo;Augartenstrasse 3;75015;bretten;DE;15317829;
2  30678283;22.06.2021;Olaf;Schneider;Hofheim 7;02837;Dorsten;DE;15317830;z2kxhh9mfh7
3
```

Diese Abbildung zeigt exemplarisch eine CSV-Datei mit Kundendaten. Die Trennung der Daten innerhalb eines Kundendatensatzes erfolgt durch ein Semikolon. Ein Zeilenumbruch kennzeichnet den Beginn eines neuen Datensatzes. Beim Import wird die ganze Datei zunächst als eine große Zeichenkette eingelesen. Anhand von Semikola und Zeilenumbrüchen wird die große Zeichenkette in die einzelnen Informationen geteilt. Nach dem Einlesen der Datei in das Programm werden die separierten Zeichenketten den entsprechenden Attributen zugeordnet. Den Vorgang des Imports werden wir im Kapitel „Programmierung“ genauer behandeln.

CSV-Dateien sind sehr einfach aufgebaut. Eines sollten Sie jedoch beachten, ein beliebter Stolperstein ist die Zeichencodierung. Es gibt je nach Codierung verschiedene Arten wie ein Computer ein Zeichen interpretiert. Sicher ist Ihnen das auch schon einmal passiert. Eine Datei in dem lauter Sonderzeichen vorkamen. Hier müssen Sie unbedingt aufpassen, dass Sie die Daten im richtigen Format zur Verfügung stellen.

Die Erreichbarkeit der Daten ist ein weiteres Kriterium, das Sie sicherstellen müssen. Ihr Programm muss für den Dateninput auf eine Datei mit einem gültigen Dateipfad zugreifen. Dazu ist es notwendig die Daten aus Ihrem Warenwirtschaftssystem zu exportieren. Je nach Preisklasse der Warenwirtschaft gibt es dafür vorgefertigte Funktionen die einen Export automatisch zu einer bestimmten Uhrzeit durchführen. Durchschnittliche Branchenlösungen stellen solche Routinen zumeist nur gegen einen Aufpreis bereit. Einen manuellen Export bietet jedoch jedes Warenwirtschaftssystem an. Wenn Sie diese Möglichkeit noch nicht genutzt haben, sprechen Sie ihren Ansprechpartner für die Warenwirtschaft einmal darauf an.

3.2 Datenverarbeitung

Nach dem Import sollten alle Daten in gleicher Vollständigkeit und Sauberkeit zur Verfügung stehen. Es wird ausdrücklich empfohlen erst mit der Verarbeitung zu beginnen, wenn die Sauberkeit der Datensätze gewährleistet ist. Legen Sie einen einheitlichen Standard fest. Nur die Daten die den Standard erfüllen werden weiterverwendet. Der Überprüfung der Datensätze sollte daher bereits während des Dateninputs stattfinden. Dem entwickelten Lösungsweg folgend wollen wir nun die Verarbeitung durchführen.

Dazu setzen wir die Zwischenschritte unseres Lösungsweges in Funktionen um. Das erfordert mitunter ein bisschen Kreativität. Die Funktionen gleichen in ihrem Aufbau der übergeordneten Programmstruktur: Input, Verarbeitung und Output. Wir müssen also nichts Neues lernen. Das Konzept der offenen Modellierung erweist abermals seine Vorteile und bildet die Struktur der Funktion bereits ab. Hier schlagen wir schon mal den Bogen zum nächsten Kapitel und bereiten

Grundlegendes Vorgehen

Die Implementierung des Codes vor. Abbildung … zeigt exemplarisch einen Zwischenschritt aus einem weiter oben betrachteten Beispiel. Das Ziel dieses Zwischenschrittes ist die Berechnung des Flächeninhaltes. Dazu sind zwei Inputwerte notwendig: die Variablen a und b. Diese werden während der Verarbeitung miteinander multipliziert. Der berechnete Flächeninhalt ist der Output der Funktion. Oftmals das Ziel des Zwischenschrittes nicht in Form einer fertigen Gleichung definiert sein, sondern in Textform. In diesem Fall müssen wir uns Gedanken machen, wie die Verarbeitung abläuft. Betrachten wir dazu einen Zwischenschritt aus dem zweiten Beispiel.
Als Input werden die Länge, die Breite und die Höhe eines Artikels übergeben. Die Abmessung der Sendung soll der Output sein. Wir können nicht einfach die 3 Maße aller Artikel aufsummieren in der folgenden Gestalt aufsummieren:

Gesamthöhe = Höhe 1 + Höhe 2 + Höhe 3
Gesamtbreite = Breite 1 + Breite 2 + Breite 3
Gesamtlänge = Länge 1 + Länge 2 + Länge 3

Besteht der Auftrag zum Beispiel aus 3 unterschiedlich langen Holzarbeitsplatten, werden diese flach gestapelt verpackt. Die Maße des Pakets stimmen mit zwei Abmaßen des längsten Artikels überein. Nur die Höhe der Sendung ergibt sich durch Addition der einzelnen Höhen.

Grundlegendes Vorgehen

Wir Übersetzen das Ziel des Zwischenschrittes „Abmessungen der gesamten Sendung“ in eine Schrittfolge, die vorgibt wie die Verarbeitung stattfindet:

1. Maximale Länge und Breite über alle Artikel bestimmen -> Länge, Breite des Pakets
2. Das minimale Abmaß über alle Artikel bestimmen
3. Die Höhe aus dem jeweils kleinsten Maß der Artikel summieren -> Höhe des Pakets
4. Gewichte

Die beiden ersten Schritte können wir elegant umsetzen, indem wir das Abmaß jedes Einzelartikels der Größe nach ordnen. Dazu gehen wir mit einer Schleife über alle Artikel und wenden eine Funktion an, die die Zahlen nach ihrer Größe sortiert. Für Schritt 3 ist nur noch die Summe aus den jeweils kleinsten Maßes aller Artikel zu bilden.

3.3 Export der Ergebnisse

Am Ende des Prozesses muss ein Ergebnis ausgegeben werden. Wie das aussieht ist natürlich ganz individuell. Vielleicht ist es nur eine Information darüber, dass die Verarbeitung beendet ist. Oder aber die verarbeiteten Daten müssen in irgendeiner Weise aufbereitet und zur Verfügung gestellt werden. Dies kann z.B. wie beim anfänglichen Import in einer CSV-Datei mit Datensätzen sein, eine PDF die ein Ergebnis mit mehreren Informationen darstellt oder vielleicht auch nur eine einzige Zahl auf dem Bildschirm sein. Der Art und Weise der Bereitstellung der Daten sind dabei keine Grenzen gesetzt.

Automatisierte Datenverarbeitung innerhalb der Prozessabwicklung

Wir haben im Verlauf dieses Kapitels die Struktur, Entwicklung und Umsetzung eines Programms zur Abwicklung eines digitalen Geschäftsprozesses kennen gelernt. Wir haben uns mit den grundlegenden Schritten, dem Dateninput, der Verarbeitung und dem Export auseinandergesetzt. Zur Modellierung des Prozesses wurde das Konzept der offenen Modellierung vorgestellt, um komplexe Problemstellungen in eine Folge von einfach zu lösenden Zwischenschritten zu verwandeln. Sie sollten nun selbst in der Lage sein, einen Geschäftsprozesse zu entwickeln und zu implementieren.

Bisher wurde bei der Behandlung von Prozessen der einmalige Ablauf skizziert. Die Abwicklung war dabei unabhängig von der Häufigkeit der Ausführung und vom Umfang der zu verarbeitenden Daten. Ohne eine Automatisierung müsste der Prozess für jede Durchführung neu angestoßen werden. Daher wollen wir uns zum Schluss dieses Kapitels der Automatisierung der Datenverarbeitung im Rahmen von Geschäftsabläufen widmen. Dazu soll der Begriff noch einmal kurz konkreter in den Kontext dieses Leitfadens gebracht werden. Das Ziel war es analoge Prozesse zu digitalisieren und zu automatisieren. Um den Rahmen dieses Leitfadens einzuhalten sollen an dieser Stelle drei Stufen der Automatisierung eingeführt werden. Diese sollen an einem Beispiel erläutert werden.

Stellen Sie sich folgende Aufgabe vor.
Die Unterste ist die vollumfängliche, manuelle Ausführung der Aufgabe durch einen Mitarbeiter, kurzum keine Automatisierung. Jeder Arbeitsschritt wird nacheinander händisch erledigt. Für jeden neuen Datensatz muss

einen ersten Schritt weg von der analogen Ebene wäre die automatisierte Abarbeitung der Aufgabe, nachdem ein Mitarbeiter den Prozess manuell angestoßen hat.

Einen Prozess zu automatisieren bringt wertvolle Vorteile mit sich:
• Nahezu unbegrenzte Skalierbarkeit (In unseren Fällen nur von der Leistung des Rechners abhängig)
• Hohe Zuverlässigkeit
• Sehr schnelle Verarbeitungsgeschwindigkeit

Beispielsweise sind der Skalierbarkeit kaum Grenzen gesetzt. Wird die Verarbeitung wie in unseren Fällen von einem Rechner übernommen, so ist der Umfang der Daten und die notwendige Verarbeitungszeit lediglich von der Leistung des PC's abhängig. Die Zuverlässigkeit des Prozesses ist eine weitere wichtige Charakteristik. Ein Rechner liefert Ihnen stets genau das, was Sie programmiert haben. Das bringt uns gleich zu den Dingen die unbedingt berücksichtigt werden sollte. Fehlerhafte Ergebnisse resultieren nahezu ausschließlich aus den folgenden beiden Gründen:

• Fehler in der Programmierung
• Unsaubere oder fehlerhafte Eingabedaten

Beide Fehlerquellen wollen wir später noch einmal näher betrachten. An dieser Stelle nur so viel, eine der wichtigsten Aufgaben beim Importieren von Daten in ein Programm ist die Überprüfung der Daten. Die zu verarbeitenden Daten sind immer in der gleichen Art und Form zu übergeben. Das kann z.B. bei Kundendaten kritisch werden. Nicht nur, dass verschiedene Portale die Kundendaten anders strukturieren (Anzahl der Adresszeilen, Straße und Hausnummer zusammen oder getrennt), man muss auch mit Fehlern seitens der Kunden rechnen. Ein häufiges Beispiel sind dabei die Angabe der Telefonnummer mit einem Slash oder einem Bindestrich. Solche Daten müssen beim Importieren unbedingt überprüft werden. Das ist zum Glück recht einfach, auch wenn es ein wenig Fleiß erfordert. Denn letztendlich sind dafür nur Kenntnisse im Umgang mit Zeichenketten notwendig. Dieses Thema wird später im Kapitel Programmierung intensiv behandelt. Nicht alle Daten kommen von außerhalb des Unternehmens. Für die Sauberkeit von Artikeldaten ist zum größten Teil das Unternehmen selbst verantwortlich. An dieser Stelle soll mit Nachdruck empfohlen werden, dass unsaubere Datenbanken unbedingt zu vermeiden sind. Eine ungepflegte Datenbank hat genauso weitreichende negative Folgen wie ein unaufgeräumtes
Warenlager. Nur weil Datenbanken digital sind und man nicht physisch über den Dreck stolpert, heißt das nicht, dass man die Sauberkeit der Daten vernachlässigen kann. Wenn Sie vorhaben ihre Prozesse zu automatisieren, werden ungepflegte Daten sehr schnell zu großen Problemen. Zunächst soll jedoch das prinzipielle Vorgehen der Digitalisierung und anschließenden Automatisierung von analogen Prozessen in die folgenden drei Schritte eingeteilt werden. Der finale Ablauf des automatisierten Prozesses stellt sich daher wie folgt dar:

Die eigentliche Automatisierung wird letztendlich durch „Schleifen" bewerkstelligt. Diese sind in ihrer Häufigkeit flexibel und laufen einfach über alle übergebenen Datensätze. Daher spielt es keine Rolle, ob 10, 100 oder mehr Datensätze verarbeitet werden. Im Großen und Ganzen gibt es 3 Typen von Schleifen. Diese werden wir uns im Kapitel „Programmierung" näher ansehen.

Die Implementierung der Schleifen und notwendigen Bedingungen ist sehr einfach. Eine Routine für die Verarbeitung der Daten zu schreiben ist in erster Linie Fleißarbeit. Wenn Sie ein gutes ordentliche Klassen- und Ablaufdiagramme erstellt haben, ist das Programmieren auf dem hier erforderlichen Level fast nur noch Schreibarbeit. Nehmen sie sich daher lieber etwas mehr Zeit bei der Planung. Eine saubere und durchdachte Darstellung des Programms in einem Diagramm wird Ihnen sehr viel Mühe und Nerven sparen.

An dieser Stelle soll noch einmal eine kurze Einordnung des Begriffs der Automatisierung vorgenommen werden.

Zum ersten die automatische Abarbeitung einer Aufgabe im Sinne der Selbstständigkeit. Stellen Sie sich vor, Sie legen in Ihrem Programm fest, dass es ganz von selbst jeden Morgen um 7.00 Uhr eine bestimmte Aufgabe erledigt, ohne dass der Prozess in irgendeiner Weise manuell angestoßen oder beeinflusst werden muss.

4. Allgemeine Grundlagen der Programmierung

Natürlich existiert bereits zahlreiche Literatur die in Programmiersprachen und die Entwicklung von Applikationen einführen. Jedoch sind Fachbücher allgemein gehalten und liefern daher nicht schnell genug Antworten auf ganz spezielle Probleme oder Vorhaben. Aus diesem Grund soll dieser Leitfaden Ihnen schnell helfen einen Überblick übe die wichtigen Kenntnisse zu erlangen, die Sie zur Umsetzung ihres Vorhabens brauchen. Darüber hinaus werden sicher weitere Themen für Sie interessant werden. Diese sind bewusst nicht hier enthalten, um die wichtigen Lektionen im Vordergrund zu halten und nicht durch zu viel Informationen zu verstecken. Bei tiefergehendem Wissensbedarf oder weiterführendem Interesse soll auf auf sehr gute, bereits existierende Fachliteratur verweisen werden.

4.1 Einordnung der Programmiersprache „Swift“

Wie bereits erwähnt ist Swift eine objektorientierte Programmiersprache. Ziel der Objektorientierung ist es, umfangreiche und komplexe Systeme durch Abstraktion auf einfache, wiederkehrende „Objekte“ herunter zu brechen. Als Beispiel für ein „Objekt“ sei hier der Mensch genannt. Das „Objekt“ Mensch wird durch viele verschiedene Merkmale charakterisiert. Augen- und Haarfarbe, Körpergröße und -gewicht seien hier stellvertretend genannt. Eine Instanz eines solchen Objektes ist dann ein bestimmter Mensch. Bei diesem sind die Eigenschaften auf ganz individuelle Weise ausgeprägt. Blaue Augen, braune Haare, eine Körpergröße von 167 cm und ein Gewicht von 67 kg. Auf diesem Weg der Abstraktion können ganze Systeme dargestellt werden. Die Grundidee ist sehr einfach. Wie immer kann man sich natürlich noch viel tiefer mit dem Thema beschäftigen, für unsere Anliegen reichen aber grundlegende Kenntnisse vollkommen aus.

4.2 Digitalisierung/Automatisierung

4.2.1 Nomenklatur

Für die bessere Nachvollziehung der verwendeten Fachbegriffe soll im Folgenden eine kurze Erläuterung für deren saubere Einordnung in den Kontext dieses Leitfadens sorgen.

1. Klasse

Schlüsselwort: „class“

Die „Klasse“ ist der Erste von zwei Typen um Objekte abzubilden. Für die objektorientierte Programmierung ist sie eine der wichtigsten Säulen. Die Klasse enthält die Attribute und Methoden eines Objektes. Stellen Sie sich als Beispiel wieder das Objekt Mensch vor. Jeder Mensch hat Attribute (Augenfarbe, Haarfarbe, etc.) und Fähigkeiten (Im Sinne der Programmierung als Methoden bezeichnet) wie z.B. Laufen, Schwimmen, Auto fahren, etc.. Die Klasse gibt als allgemein vor, welche Attribute und Methoden jedes Objekt vom Typ „Mensch“ besitzt. Wenn in der digitalen Welt ein ganz bestimmter Mensch abgebildet werden soll, so wird eine Instanz der Klasse Mensch initialisiert. Diese Instanz zeichnet sich durch die individuelle Ausprägung der vorgegebenen Attribute und Methoden aus und unterscheidet sich damit in

seiner Gesamtheit von anderen Instanzen des gleichen Typs. Zu beachten ist, dass die Klasse ein

Referenztyp ist. Das bedeutet, dass bei Zuweisungen oder Übergaben (z.B. als Übergabeparameter an eine Methode) ein Zeiger auf die Instanz kopiert wird, nicht die Instanz selbst. Das ist ein wichtiger Unterschied im Vergleich zu dem gleich folgenden Typ: der Struktur.

Struktur Schlüsselwort:

„struct“

Swift bietet als objektorientierte Programmiersprache neben „Klassen“ einen weiteren Typ von Objekten, die sogenannten „Structures“ (deutsch: Strukturen). Diese unterscheiden sich von den Klassen in einigen Eigenschaften. Im Unterschied zur Klasse sind Strukturen Wertetypen. Bei Zuweisungen oder Übergaben (z.B. an Variablen oder Methoden) wird nicht der Zeiger auf die Instanz kopiert, sondern eine Kopie der Instanz übergeben. Daraus resultiert, dass nicht mehr die ursprüngliche Instanz verändert wird, sondern immer ausschließlich die Kopie. Der Begriff Strukturen wird weiterführend, unter Berücksichtigung der ursprünglichen Definition innerhalb von Swift, für das digitale Gerüst einer Abbildung verwendet. Als Beispiel soll hier das Beispiel eines Objektes „Paket“ dienen. Wie sieht die digitale Variante eines Pakets aus? Dazu wird zunächst eine Struktur „Paket“ erstellt, der bestimmte Eigenschaften gegeben werden, die in der digitalen Version (also der Struktur) als Variablen abgebildet sind. So hat jedes Paket zum Beispiel ein Gewicht. Alle Eigenschaften zusammen werden dann in der Struktur abgebildet. Will man nun ein ganz bestimmtes Paket darstellen, wird eine Instanz der Struktur erstellt, die die spezifischen Informationen eines ganz bestimmten Pakets abbildet.

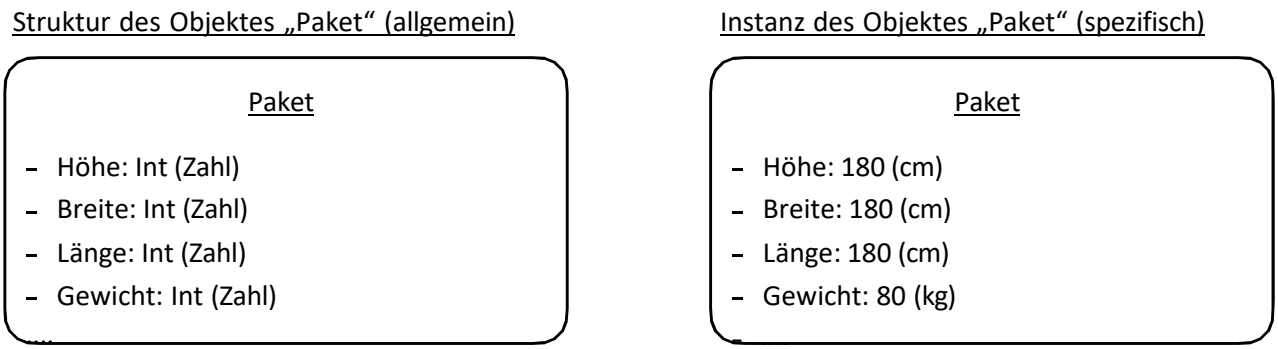


ABBILDUNG 1 | DARSTELLUNG DES STRUKTURBEGRIFFS.

Framework

Ein Framework ist laut Definition ein „Programmiergerüst“. Jedes Framework stellt damit ein Muster für klar abgegrenzte Systeme zur Verfügung. Sie bekommen also schon von Haus aus einen Bausatz an die Hand und müssen nicht alles von Grund auf selbst programmieren. Die Frameworks bestehen wiederum aus einer Ansammlung von Strukturen („Structures“), die die einzelnen Bestandteile des Gerüstes definieren. Von Haus aus enthält XCode unzählige sehr große Frameworks. Wollen Sie zum Beispiel mit Zeichenkette arbeiten so wird Ihnen das Framework

„Foundation“ zur Verfügung gestellt, welches die Struktur „String“ (englisch für Zeichenkette) enthält. Diese liefert Ihnen die grundlegenden Variablen und Funktionen die Sie beim Umgang mit Zeichenketten benötigen. Sie wollen die Länge einer Zeichenkette wissen? Dafür gibt es eine entsprechende Variable. Sie wollen die Zeichenkette manipulieren? Entsprechende Funktionen sind bereits implementiert. Um die für Ihre Anwendungen relevanten Frameworks zu nutzen müssen Sie diese stets zu Beginn importieren.

Compiler

Mit den Befehlen allein kann ein Computer nichts anfangen. Es bedarf einem Compiler, der den Programmcode in Maschinencode übersetzt. Erst dann werden die Befehle des Programms auch von Ihrem Computer ausgeführt. Ein Compiler ist also ein Übersetzer, der zwischen Ihnen (und ihrem Programmcode) und dem Computer. Da es unterschiedliche Programmiersprachen gibt, gibt es auch unterschiedliche Compiler. Sie können also nicht einfach im Editor auf Windows ein Programm schreiben und erwarten, dass es automatisch auch auf einem Mac läuft.

4.2.2 Werkzeugkasten

Im Folgenden Abschnitt sollen die grundlegenden Werkzeuge zum Programmieren zusammengestellt werden. Mit diesen ausgerüstet können wir schon sehr große Schritte machen.

3. Attribute

Die Eigenschaften eines Objektes werden in sogenannten Attributen abgebildet. Wenn wir abermals das Objekt „Mensch“ betrachten, können die weiter oben bereits genannten Eigenschaften wie Augenfarbe, Haarfarbe, Körpergröße, Gewicht, etc. in zwei Typen von Attributen eingeteilt werden. Zum einen die unveränderlichen wie z.B. die Augenfarbe. Diese sind über die gesamte Lebensdauer hinweg konstant. Einmal festgelegt ist die Augenfarbe nicht mehr zu ändern. Dahingegen sind Eigenschaften wie Haarfarbe, Körpergröße und Gewicht mit der Zeit veränderbar, also variabel. Attribute werden demzufolge in Konstanten und Variablen unterteilt.

Konstante
Schlüsselwort: „let“
Beispiel: let augenFarbe = „grün“
Variable Schlüsselwort:
„var“
Beispiel: var haarFarbe = „braun“

Wie der Name deutlich macht, können Konstanten nach der Initialisierung nicht mehr verändert werden. Variablen hingegen können während der gesamten Laufzeit des Programms verändert werden. Man könnte daher auf die Idee kommen einfach alle Attribute als Variable zu deklarieren. Davon soll an dieser Stelle zumindest obligatorisch abgeraten werden. Sie sollten sich während der Konzeptionierung überlegen, ob ein Attribut über die gesamte Laufzeit konstant oder veränderlich sein soll. Denn für Variablen müssen im Hintergrund Funktionen bereit gehalten werden, die beispielsweise zur Veränderung selbiger notwendig sind. Da diese zusätzlichen Funktionen für Konstanten nicht benötigt werden, wird der Code schlanker und das Programm schneller. Ein signifikanter Geschwindigkeitsunterschied ist natürlich erst bei großen Programmen mit sehr viel Code oder umfangreichen Datensätzen spürbar. Auch wenn die hier betrachteten Aufgaben den Prozessor noch lange nicht ins Schwitzen bringen, sollte der Unterschied zumindest einmal genannt werden, falls Sie vor haben z.B. in Schleifen große Datenmengen zu verarbeiten.

3. Funktionen/Methoden

Schlüsselwort: „func“
Syntax: func nameDerFunktion (übergabeParameter1: Datentyp, übergabeParameter2: Dat…..

Ein weiteres unerlässliches Instrument für unseren Werkzeugkasten ist die „Funktion“. Das Objekt „Mensch“ soll, um es einmal umgangssprachlich zu formulieren, schließlich auch gewisse Dinge tun können. Der grundlegende Aufbau einer Funktion ist überschaubar. Die Funktion wird mit dem Schlüsselwort „func“ eingeleitet. Darauf folgt der Funktionsname. Anschließend können der Funktion beliebige Parameter übergeben werden. Diese Parameter können dann bei jedem Aufruf der Funktion unterschiedlich belegt werden und machen eine Funktion damit flexibler einsetzbar. Zur Verdeutlichung sollen die beiden folgenden Funktionen miteinander verglichen werden:

```
var haarFarbe = "blond"

func haarFarbeAendernStatisch(){
    haarFarbe = "braun"
}

func haarFarbeAendernFlexibel(haarFarbeNeu: String){ haarFarbe =
    haarFarbeNeu
}

haarFarbeAendernFlexibel(haarFarbeNeu: "beliebigeHaarfarbe")
```

Die Funktion „haarFarbeAendernStatisch“ ändert immer nur die Haarfarbe von blond zu braun. Dagegen ändert die Funktion „haarFarbeAendernFlexibel“ die Farbe der Haare in die, die als Übergabeparameter übergeben wird. Der Unterschied wird schnell deutlich wenn man sich vor Augen führt, dass man bei der Verwendung der statischen Funktion für jede Haarfarbe eine eigene Methode bräuchte. Die flexible Funktion hingegen kann ohne weitere Anpassungen alle Farben ändern.

Ein weiterer wichtiger Bestandteil von Funktionen sind Rückgabewerte. Sie werden nach den Übergabeparametern festgelegt und sind mit einem Pfeil („->“) einzuleiten. Eine Funktion muss nicht zwangsläufig, wie im eben betrachteten Beispiel, auf externe Variablen zugreifen. Eine Methode kann auch in sich geschlossen sein und ein Ergebnis zurückgeben.

4.3. Beispiel 1: Speditionsmodul

4.3.1 Einführung

4.3.2

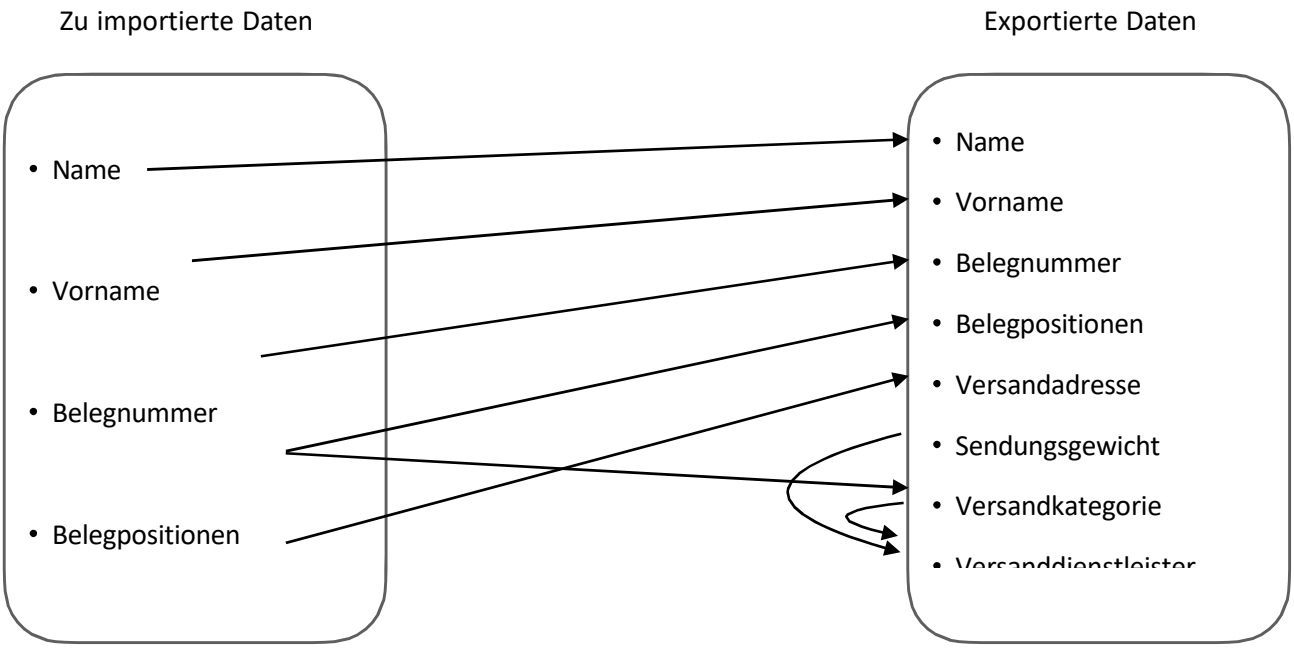
Als Onlinehändler mit Produkten über der Sperrgutgröße müssen täglich zahlreiche Sendungen mit Speditionen verschickt werden. Im Gegensatz zu Paketen müssen Speditionssendungen bisher einzeln in einem eigenen Portal angemeldet werden. Dazu müssen die Kunden- und Auftragsdaten händisch in die Portalmaske eingetragen werden. Bei täglich durchschnittlich 70 Speditionssendungen beträgt der Zeitaufwand schon jetzt ca. 4 Stunden am Tag. Diese Arbeit ist äußerst mühsam, da sich durch die Monotonie und die Routine des immer gleichen Arbeitsablaufs leicht Fehler einschleichen, die weitreichende Konsequenzen haben. Falsche Lieferadressen, oder Auftragsdaten führen schnell zu erheblichen finanziellen Verlusten. Gleichzeitig ist festzustellen, dass dies keineswegs eine befriedigende Aufgabe darstellt. Die Verarbeitung der Daten aller Sendungen des Tages liegt nach der Automatisierung bei unter einer Sekunde. Das wird sich auch bei sehr hohen Zahlen nicht ändern. Je mehr Sendungen in Zukunft verschickt werden, desto effektiver wird das programmierte Tool.

Folgende Gründe waren ausschlaggebend für eine Automatisierung der Aufgabe:

Kriterium	**Hintergrund**
Hochfrequent wiederkehrend	Die Arbeit ist täglich und für mehrere Stunden auszuführen.
Vollumfänglich Reproduzierbar	Der Inhalt der Arbeit ändert sich nicht, der Prozessablauf ist stets der gleiche.
Hohe Zeitersparnis	Derzeit beträgt der Zeitaufwand ca. 4 Stunden (Tendenz stark steigend). Durch die Automatisierung wird sich die Arbeitszeit zukünftig auf ca. 10 Minuten am Tag reduzieren.
Starke psychische Entlastung der Mitarbeiter	Der manuelle Arbeitsanteil durch einen Mitarbeiter beschränkt sich zukünftig auf einen Mausklick um das Programm zu starten und das anschließende Hochladen der erstellten CSV Datei in die Webmaske des Speditionsdienstleisters. Diese Schritte umfassen in Summe ca. 20 Sekunden.
Sehr gute Skalierbarkeit	Mit dem anhaltend hohen Wachstum des Unternehmens wird das Volumen der Speditionssendungen weiter stark steigen. Durch die Automatisierung spielt die Anzahl der Sendungen jedoch in Zukunft keine Rolle mehr, die Verarbeitung aller Sendungen des Tages liegt seitens des Programms bei unter einer Sekunde.

4.3.2. Konzeptionierung

Ziel des Speditionsmoduls ist die automatische Verarbeitung der auftragsrelevanten Daten. Dies umfasst die Kunden-, Beleg-, und Versanddaten. Am Ende des Prozesses soll eine CSV Datei mit den für den Versand benötigten und vom Spediteur vorgegebenen Daten erstellt werden.
Für eine effiziente und geradlinige Verarbeitung der importierten Daten ist es daher bei der Konzeptionierung sinnvoll sowohl das Aussehen der importierten Daten als auch die Struktur der finalen CSV Datei zu berücksichtigen. Die importierten und exportierten Daten bilden zwei separate Datenzustände die es in einfacher Weise zu verbinden gilt.



Aus den genannten Gründen werden im Folgenden zwei verschiedene Vorgehensweisen beschrieben. Im Mittelpunkt steht dabei der Weg den die Daten durch den Programmcode nehmen.

Die erste Möglichkeit ist, nur eine Exportstruktur anzulegen. Die notwendigen Importdaten werden direkt aus der CSV Datei eingelesen und durch die entsprechenden Funktionen zugewiesen und sowie -geordnet. Eine Speicherung der Importdaten in ihrem unveränderten importierten Zustand findet nicht statt. Innerhalb des Codes werden nur die fertig bearbeiteten und zugeordnet Exportdaten gespeichert. Das ist in sofern nachteilig, als dass jegliche Informationen über die Datengrundlage verloren gehen. Ein Vergleich zwischen Import- und Exportdaten zum Beispiel ist anschließend nicht mehr möglich. Auf den ersten Blick scheint der Vorteil weniger Daten speichern zu müssen verführerisch. Jedoch steht dem geringeren Umfang der zu speichernden Variablen entweder ein Verlust der gesamten Datengrundlage oder eine signifikante Hürde des Zugriffes (Jeder Parameter müsste einzeln importiert werden) auf die benötigten Daten gegenüber. Die zweite und deutlich elegantere Lösung ist eine einzige Struktur anzulegen, die sowohl in der Lage ist die Importdaten als auch die Exportdaten abzubilden und über die gesamte Dauer eines Programmzyklus bestehen bleibt. Dadurch ist jederzeit die Möglichkeit gegeben auf alle relevanten Importvariablen durch einen einfachen Aufruf innerhalb der Klasse zugreifen zu können.

Klassendiagramm | Eigenschaften der Klasse Spedition und der Struktur Spezifikation

Spedition

- var checkBoxStates: Array [Bool]
- var adressen: Dictionary [Int:Adresse]
- var fehlerhafteAdressen: Dictionary [Belegfehler]
- var speditionsListe: Dictionary [String:Spezifikationen]
- let gesperrteMetropolenHellmann: Array [String]
- let user: NSUserName
- let distributorMailAdressen: Array [String]
- var processDataString: String
- func importCSV ( ) -> String
- func generateErrorReport ( ) -> Bool, Belegfehler
- func manipulateDimensions ( ) -> String
- func changeColliToKiste ( ) -> String
- func changeDistributorMail ( ) -> String
- func changeCustomerTelefon ( ) -> String
- func changeCustomerAdress ( ) -> [String]
- func zuordnungSets ( )
- func manipulateWeight ( ) -> Double
- func calculateSize ( ) -> [Double]
- func bestimmeVersandDienstLeister ( ) -> String, Double
- func manuelleAenderungVersand ( )
- func assignToDeliveryList ( )
- func exportCSV

Spezifikationen

- let name: String
- let versandKategorie: Int
- let versandform: String
- let laenge: Double
- let breite: Double
- let hoehe: Double
- let volumen: Double
- let gewicht: Double
- let versandpreisHellmann: Double
- let versandpreisHauptvogel: Double
- let versandpreisEmons: Double
- let retourpreisHellmann: Double
- let retourpreisHauptvogel: Double
- let retourpreisEmons: Double
- let gesperrteMetropolenHellmann: [String]
- let versandformAT: String
- let laengeAT: Double
- let breiteAT: Double
- let hoeheAT: Double
- let gewichtAT: Double
- let versandpreisATHellmann: Double
- let versandpreisATEmons: Double
- let versandpreisFRHellmann: Double

- func init ( )

S