

Agiles Projektmanagement SoSe 2020

**Industrie 4.0 Fahrzeugmontage
mit einem autonomen
Roboterschwarm**

29. JUNI

**Jan Ewerszumrode, Viktor Görlitz,
Dennis Hepp, Sören Möller, Nico
Rixe, Roman Sliwinski**



FH Bielefeld
University of
Applied Sciences

Inhaltsverzeichnis

Projektmotivation	3
Zielsetzung	4
Projektorganisation	6
Umsetzung	8
Datenbank	8
Vorranggraph	13
Implementierung	16
Validierung	20
Layout of Workstations	24
Konzept zur FTF-Wegeplanung	27

Projektmotivation

Im industriellen Zeitalter 4.0 stehen Produkthersteller einem hohen Grad an Produktindividualität und der damit einhergehenden enormen Anzahl an Produktvarianten gegenüber. Gleichzeitig entstehen neue Anforderungen bei dem Bezug der dazu benötigten Teile aus einem weltumspannenden Liefernetzwerk. Da die hohe Varianz eine Lagerung aller möglichen Teilevarianten unmöglich macht, müssen die Lieferanten die Teile „Just-in-Time“, das heißt genau zum Zeitpunkt der Produktion an die Montagebänder liefern. Unterschieden werden hierbei Individualteile, die nur für einen bestimmten Auftrag gefertigt werden und Gleichteile, die in einen Großteil der Aufträge vorkommen und so deutlich einfach zu steuern sind.

So werden beispielsweise in der Endmontage von Automobilen die Fahrzeugkarossen entsprechend ihres Kundenauftrags aufs Band gelegt und durchlaufen mehr als 100 Arbeitsstationen. In diesen Stationen werden entsprechend des jeweiligen Auftrags unterschiedliche Teile ins Fahrzeug eingebaut. So z.B. manchmal ein manuelles, ein automatisches oder gar kein Schiebedach. Jede Montagetätigkeit benötigt unterschiedlich viel Zeit und andere Teile aus dem Zulieferernetzwerk. Auf dem Band sind die Arbeitsstationen eng miteinander durch den permanenten Durchfluss der Karossen verbunden und gehen nahtlos ineinander über (Fließbandmontage). Obwohl die Art der Montage hocheffizient insbesondere bezogen auf die Durchlaufzeit der Fahrzeuge ist, kommt es aufgrund der globalen Beschaffung und der hohen Varianz immer häufiger dazu, dass die richtigen Teile nicht zur gleichen Zeit an der entsprechenden Station vorhanden sind. Da ein Anhalten des Montagebandes alle mehr als 100 Montagestationen betreffen würde, werden die Fahrzeuge unfertig weitergeleitet und landen am Ende in der Nacharbeit. Ein manueller, hochineffizienter Prozess in dem mittlerweile bis zu 40% der Fahrzeuge im Schnitt 48 Stunden verbringen. Diese hohe Quote an Nacharbeit zeigt, dass das etablierte, unflexible Fließbandsystem an seine (insbesondere bezogen auf die hochindividuelle Produktion) Grenzen stößt.

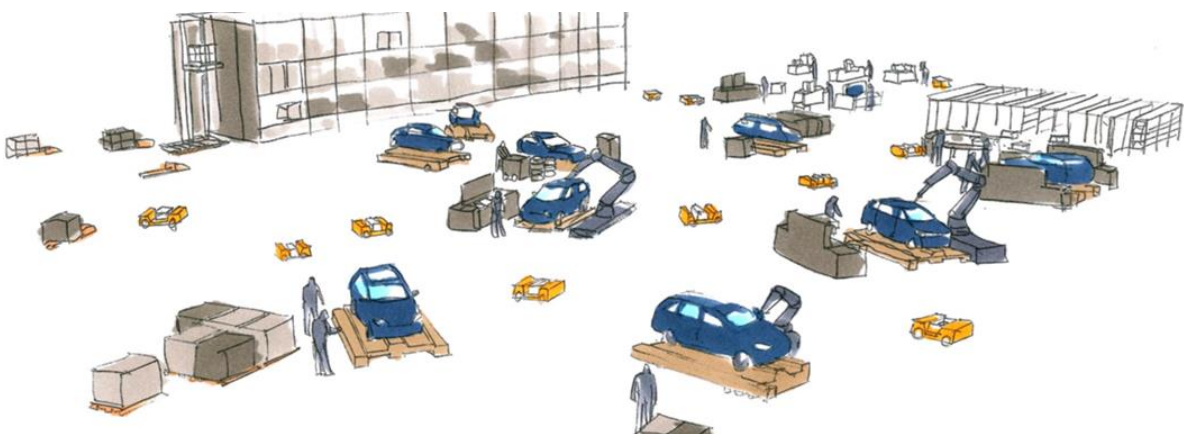


Abbildung 1: Modulare Fertigung für Automobile

Abhilfe soll hier ein flexibleres, intelligenteres Fertigungssystem bringen (vgl. Abbildung 1). Industrie 4.0 basiert auf digitale Echtzeitvernetzung und dezentrales, autonomes Verhalten der Menschen und Maschinen. Störungen (wie die Verspätung von Teilelieferungen) werden bei diesem System mitgedacht und durch das flexible Verhalten als Teil des Normalablaufs behandelt. Die flexible Industrie 4.0 Montage zeichnet sich dadurch aus, dass das Fließband durch Fahrerlose Transportfahrzeuge (FTF) ersetzt wird, dass ein Werkstück aus einem Pufferbereich entnimmt und die notwendigen Arbeitsstationen sukzessive anfährt. Die Stationen sind frei auf der Fläche der Montagehalle verteilt und ermöglichen so einen flexibleren Fertigungsablauf.

Das FTF wird bei Fertigungsbeginn (Auswahl z.B. über den nahenden Liefertermin) mit dem Kundenauftrag verheiratet. Dieser enthält einen sogenannten Vorranggraphen, der die durchzuführenden Montageschritte enthält. Hierbei werden Schritte unterschieden, die frei und unabhängig voneinander ausgeführt werden können und solche, die eine bestimmte Reihenfolge aufweisen müssen. Mit dieser Information kommuniziert das FTF mit den nächsten möglichen Arbeitsstationen und dem Teilelager. Es fragt die Stationen nach freien Montagekapazitäten und ermittelt ob die benötigten Teile vorhanden sind. Nach Abfrage und Auswertung dieser Information entscheidet es und macht es sich auf den Weg. Aus dem Teilelager werden nun ggf. die benötigten Teile zeitgleich (von anderen FTF) an die Stationen gebracht. Bei Gleichteilen kommt eine sogenannte KanBan-Steuerung zum Einsatz, die dafür sorgt, dass immer genügend Teile an der Station verfügbar sind. Individualteile werden erst transportiert, nachdem sich ein FTF für eine Station entschieden hat. Auf diese Weise wird der hochflexible Durchlauf durch die Produktion organisiert.

Um Konflikte im Ringen um die Produktionsressourcen zu organisieren, kann neben der Autonomie der FTF auch eine Autonomie der Arbeitsstationen eingeführt werden. Das FTF mit der Karosse weiß dann aus dem zugehörigen Auftrag den Liefertermin (wann soll das Fahrzeug fertig sein) und den gezahlten Preis. Ziel des autonomen Agenten ist es nun den Liefertermin einzuhalten und dabei so viel Gewinn wie möglich zu machen. Für Montageservices an den Arbeitsstationen fallen für das FTF Kosten an. Der Preis wird von dem FTF mit der Station verhandelt. Die Stationen wiederum wollen für ihre Arbeit möglichst viel Geld bekommen. Das bedeutet, wenn viel Arbeit da ist steigt der Preis und umgekehrt. Durch eine Zwischenlagerung während der Produktion (weil z.B. kein Arbeitsschritt mehr möglich ist) oder nach der Produktion (vor dem eigentlichen Liefertermin) entstehen ebenfalls Kosten, die mit dem Lagerplatzagenten verhandelt werden.

Zielsetzung

Um die Komplexität für den Anfang beherrschbar zu machen, soll die beschriebene Industrie 4.0 Montage für die Fahrradherstellung umgesetzt werden. Das zu

entwickelnde Programm sollte allerdings unabhängig von Anwendungsfall sein und erst durch Modellierung auf ein bestimmtes Szenario übertragen werden.

Im vergangenen Semester wurden in einem ersten Schritt zur Realisierung einer Simulation der beschriebenen Industrie 4.0-Montage einen Produkt-Konfigurator umgesetzt, der es ermöglicht individuelle Produkte zu konfigurieren. Der Konfigurator ist so aufgebaut, dass aus einer festen Liste an Optionengruppen, jeweils eine Option pro Gruppe ausgewählt werden muss (eine Option kann dann z.B. auch „Kein Schiebedach“ sein), um den Auftrag vollständig zu spezifizieren. Für jede Option kann ein Preis hinterlegt werden und der Kunde kann zudem am Ende ab einem bestimmten Datum (Minstdauer) einen Liefertermin auswählen. Am Ende werden Endpreis, die vollständige Konfiguration und der Liefertermin angezeigt. Die Konfiguration sollte dabei auch automatisch (durch ein Programm) erfolgen können.

1.) Die bisherigen Ergebnisse sollen gesichtet und geprüft werden. Gegeben falls soll eine Anpassung nach den Vorgaben des Product Owners durchgeführt werden.




2.) In einem zweiten Schritt soll aus den Aufträgen (sog. Produktbeschreibungen) Stücklisten generiert werden. Stücklisten sind Listen von allen Bauteilen, die zur Fertigung eines konkreten Produktes benötigt werden. Zu jedem Bauteil wird eine Anzahl und eine Stücklistenregel angegeben. Die Regel ist eine boolesche Kombination aus den im Auftrag beschriebenen Optionen. Die Stücklistenauflösung erzeugt dann auf Basis der Aufträge und der Regeln den konkreten Teilebedarf. Wird eine Regel durch einen Auftrag erfüllt, so wird der entsprechende Teilebedarf dem Auftrag hinzugefügt.

3.) Letztendlich sollen in einem dritten Schritt die Arbeitsgänge zur Erstellung eines Auftrags generiert werden. Da jeder Auftrag individuell ist, müssen auch hier auf Basis der gewählten Optionen die entsprechenden Arbeitsschritte automatisch anhand von Regeln bestimmt werden. Hierzu soll ein Layout mit verschiedenen Arbeitsstationen, die jede verschiedene konkrete Arbeitsgänge ausführen können, entworfen werden. Hierbei soll beachtet werden bei welche Optionenkombination der jeweilige Arbeitsschritt durchgeführt werden muss. Zudem muss mit dem Arbeitsgang der jeweilige Teilebedarf aus der Stückliste verbunden werden. Es soll eine Relation zwischen den Arbeitsgängen (Vorranggraphen) definiert werden, die darstellt welcher Arbeitsschritt zwingend vor einem anderen stattfinden muss. Das Ergebnis soll anhand eines Beispielszenarios demonstriert werden.

4.) Zur Weiterführung des Projektes durch nachfolgende Semester des Studiengangs soll bei der Umsetzung auf eine ordnungsgemäße und saubere Programmierung geachtet sowie Wert auf eine vollständige Dokumentation gelegt werden.

Projektorganisation

Da die Projektarbeit im Rahmen des Moduls „Agiles Projektmanagement“ an der FH Bielefeld durchgeführt wurde, wurde eine auf ein Semester reduzierte Variante des SCRUM-Projektmanagement zur Durchführung des Projektes gewählt. Zu Beginn des Projektes wurden fünf Sprints mit je zwei wöchigen Arbeitsphasen festgelegt. Am Ende der jeweiligen Sprints wurde mit sämtlichen Projektbeteiligten ein Sprint-Review-Meeting abgehalten, in dem der Zwischenstand präsentiert und diskutiert wurde. Weiterhin wurden im Anschluss die Aufgaben für den kommenden Sprint festgelegt und priorisiert. Im Folgenden wird die Rollenverteilung dargestellt:

Bezeichnung	Person(en)	Aufgabe(n)
Product Owner	 Prof. Dr.-Ing. Christian Schwede	Projektverantwortlicher, Wertmaximierung, Aufgaben-Priorisierung
SCRUM Master	 Prof. Dr. rer. oec. Thomas Süße	Einführung in SCRUM, Bereitstellung von Res- ourcen, Kommunikationsschnitt- stelle, Konfliktmanagement
SCRUM Entwickler- team	 Jan Ewerszumrode	Konzeptarbeit, Implementierung, Datenbank Optimierung, Dokumentation



Viktor Görlitz



Dennis Hepp



Sören Möller



Nico Rixe



Roman Sliwinski

Bei der Bearbeitung der einzelnen Aufgaben aus dem Backlog durch das Entwicklerteam wurde der Aufgabenaufwand durch Planning-Poker vom Team bewertet. Die Aufgaben wurden anschließend von den Mitgliedern des Entwicklerteams entsprechend ihrer Vorkenntnisse und Kompetenzen bearbeitet und im Anschluss im Team präsentiert, besprochen und optimiert.

Umsetzung

Nachdem zu Beginn des Projektes die Vorarbeiten gesichert und getestet wurden (siehe Zielsetzung 1) wurden im ersten Sprintreview tiefgreifende Veränderungen an den Vorarbeiten beschlossen. So wird lediglich der Datenbankserver beibehalten, jedoch beginnend bei der Datenbankstruktur eine Neuentwicklung durchgeführt. Die durchgeführten Arbeiten reichen somit von der Infrastruktur-Konzeption über die Implementierung von Generatoren bis hin zur Konzeptentwicklung für weiterführende Arbeiten durch anschließende Semester. Im Folgenden werden die Teilergebnisse einzeln präsentiert und erläutert:

Datenbank

Zur Verwaltung von den verschiedenen Datensätzen, welche zur Realisierung des Eingangs beschriebenen Szenarios relevant sind, wurde eine entsprechende Maria Datenbankstruktur realisiert. Diese wurde auf einem Datenbankserver¹ unter dem Namen *Fahrradshop* hinterlegt und kann über einen VPN-Client² aus dem Netzwerk der Fachhochschule Bielefeld erreicht werden.

Um eine erleichterte Kommunikation mit der Datenbank zu ermöglichen wurde auf Funktionen aus dem Python-Modul `db_communication.py` zurückgegriffen. Dieses Modul wurde maßgeblich von dem vorherigen Projektteam entwickelt und nahezu unverändert für die Neustrukturierung der bisher verwendeten Datenbank weitergenutzt.

Insgesamt lässt sich eine Unterteilung der Datenbank in drei Dimensionen abstrahieren:

1. Die verwaltungstechnische Dimension des zugrundeliegenden Szenarios wird durch die Tabellen *Kunde* und *Auftrag* abgebildet. Durch diese lassen sich Kunden- und Auftragsdaten effizient und übersichtlich handhaben.
2. Die konfigurationstechnische Dimension zur Erstellung von einem Fahrrad anhand von den individuellen Vorstellungen eines Kunden wird durch die Datenbanktabellen *Konfiguration*, *Merkmalcluster* und *Merkmale* abgebildet.
3. Die dritte Dimension der Datenbank ergibt sich schließlich aus den Tabellen *Arbeitsschrittgruppe*, *Arbeitsschritt*, *Reihenfolgevorgabe* sowie *Einzelteile*.

¹<http://min-ifm-xdm.ad.fh-bielefeld.de/phpMyAdmin> Zugangsdaten: **User:** root; **Passwort:** FDS-apm1

² Cisco Anyconnect: <https://www.fh-bielefeld.de/dvz/faq/item/125>

Durch sie wird zum einen die technische Sicht auf den Zusammenbau eines Fahrrades dargestellt. Zum anderen wird durch ihr Zusammenwirken auch eine Struktur zur Verfügung gestellt, durch welche sich ein Ablaufplan zur Generierung der einzelnen Arbeitsschritte für das Zusammenbauen eines vom Kunden konfigurierten Fahrrades ableiten lässt.

Im Folgenden werden nun kurz die vorhandenen Tabellen aus der Datenbank vorgestellt und ihre jeweilige Funktion im Gesamtkontext erläutert.

Kunde:

Diese Tabelle beinhaltet die wesentlichen Daten, durch welche sich ein Kunde charakterisieren lässt. Dazu zählen die Attribute *Name*, *Vorname* sowie eine *E-Mail-Adresse*, welche zur Kontaktaufnahme mit dem Kunden dient. Jeder Kunde ist darüber hinaus durch die ID *KundenNr* eindeutig identifizierbar.

Auftrag:

Ein *Auftrag* ist für das vorliegende Szenario durch eine eindeutige *Auftragsnummer* angegeben. Damit verbunden sind sowohl das *Datum* an welchem die Bestellung erfolgte und das Datum zu welchem das Produkt voraussichtlich an den Kunden ausgeliefert werden kann. Ferner ist ein Auftrag an einen bestimmten Kunden gebunden, weshalb zu einem Auftrag auch der Index *KundenNr* für einen bestimmten Kunden gehört.

Konfiguration:

Die Tabelle Konfiguration der Datenbankstruktur *Fahrradshop* verwaltet die von einem Kunden vorgenommen Konfigurationen eines gewünschten Fahrrades. Dafür wird der zu einem Kunden gehörende *Auftrag* mit einem ausgewählten *Merkmal* verknüpft. Somit ist eine vollständige Zuordnung von einem Auftrag mit einem konfigurierten Fahrrad möglich. Zu jeder getroffenen Auswahl von einem bestimmten Merkmal ist, aus Gründen der Eindeutigkeit, der Index *KonfigNr* als Identifikationsnummer angegeben.

Merkmalcluster:

Diese Tabelle bildet die *Merkmalcluster* ab, aus denen sich ein Fahrrad grundsätzlich immer zusammensetzt. Beispiele hierfür sind die Merkmalcluster Lenker, Rahmen oder Pedale. Durch die entsprechenden *Merkmalcluster* können die vom Kunden ausgewählten Merkmale zu einer entsprechenden Gruppe von *Merkmalen* zugeordnet werden.

Als Attribute enthält diese Tabelle eine *Clusternummer* als Primärschlüssel und *Namen* des entsprechenden Clusters. Insgesamt sind die folgenden elf *Merkmalcluster* eines Fahrrades hinterlegt:

Rahmen, Federung, Lackierung, Lenker, Felgen, Radbereifung, Gangschaltung, Sattel, Bremsen, Beleuchtung und Pedalen.

Merkmale:

Die Datenbanktabelle *Merkmale* umfasst sämtliche Konfigurationen, welche ein Kunde für die Zusammenstellung seines gewünschten Produktes wählen kann. Darunter fallen beispielsweise *Merkmale* wie „Bereifung Sport“ oder „Komfort Pedale“. Insgesamt stehen zur Zusammenstellung eines Fahrrades 72 verschiedene *Merkmale* zur Verfügung, welche jeweils einem *Merkmalscluster* zugeordnet sind.

Als Primärschlüssel dient dieser Tabelle der Index *MerkmalNr*, durch den sämtliche Merkmale eindeutig identifizierbar sind.

Die Attribute in dieser Struktur sind einerseits das *Merkmalscluster* zu welchem ein konkretes *Merkmal* gehört sowie andererseits die *Bezeichnung* und ein fiktiv angenommener *Preis* für ein Merkmal.

Arbeitsschrittgruppe:

Hier ist ein Datensatz mit einem Umfang von insgesamt 25 *Arbeitsschritten* hinterlegt, welche sich, aus der Sicht des Projektteams, als sinnvoll erwiesen haben, um ein Fahrrad zu konstruieren. Für die Abbildung des *Vorranggraphen* in der Datenbankstruktur spielt diese Tabelle eine wichtige Rolle (vgl. Abbildung Vorranggraph). Damit besteht die Möglichkeit verschiedene Arbeitsschritte zu einer Gruppe zusammenfassen zu können.

Die Attribute dieser Tabelle sind die *Bezeichnung* der *Arbeitsschrittgruppe*, das *Merkmalscluster*, welches durch die konkreten *Arbeitsschritte* der jeweiligen *Arbeitsschrittgruppe* montiert werden soll und der boolesche Wert *Optional*. Dieser gibt an ob eine bestimmte Gruppe von *Arbeitsschritten* nur bei bestimmten Aufträgen ausgeführt werden muss. Beispielsweise ist dies der Fall bei dem Anbringen eines Dynamos, da dieser unter Umständen nicht verbaut werden muss, da auch die Möglichkeit besteht batteriebetriebene Beleuchtung oder einen Nabendynamo auszuwählen. Schließlich dient auch bei dieser Tabelle der Primärschlüssel *ArbeitsschrittgruppenNr* zur eindeutigen Identifizierung einer speziellen *Arbeitsschrittgruppe*.

Arbeitsschritt:

Durch diese Tabelle der Datenbank ist ein konkreter *Arbeitsschritt* definiert. Dieser setzt sich aus dem Primärschlüssel *SchrittNr* und den folgenden Attributen zusammen. Enthalten ist hier zum einen die *Arbeitsschrittgruppe*, zu welcher ein vorliegender *Arbeitsschritt* gezählt wird. Zum anderen ist auch eine Verbindung zwischen einem *Merkmal* durch *MerkmalNr* und den zugehörigen *Einzelteilen* durch das Attribut *EinzelteilNr* gegeben. Zusammen mit der hier ebenfalls hinterlegten Anzahl an benötigten *Einzelteilen* für ein bestimmtes *Merkmal* können aus dieser Tabelle mit geringem Aufwand Stücklisten erzeugt werden. Schließlich ist auch die *Montagezeit* für den vorliegenden *Arbeitsschritt* in dieser Tabelle hinterlegt.

Reihenfolgevorgabe:

Diese Tabelle dient zur impliziten Festlegung der Reihenfolge der *Arbeitsschritte* im *Vorranggraphen*. Hierdurch sind die zugehörigen *Nachfolger* eines bestimmten *Arbeitsschrittes* entsprechend angegeben.

Als Primärschlüssel dient hierbei eine Vereinigung der Indizes zweier *Arbeitsschrittgruppen*.

Einzelteile:

Die Tabelle Einzelteile beinhaltet alle wesentlichen technischen Komponenten, die zum Zusammenbau eines Fahrrades von Bedeutung sind. Insgesamt sind in dieser Datenbanktabelle bisher 154 Datensätze hinterlegt. Bestandteile

hierbei sind die genauen technischen *Bezeichnungen*, wie beispielsweise „Surly Long Haul Trucker Rahmenkit“ für einen Rahmen.

Als einziges Attribut enthält diese Tabelle die *Bezeichnung* eines *Einzelteiles*, welches durch den Primärschlüssel *EinzelNr* identifiziert werden kann.

Vorranggraph

Ein Fahrrad besteht im Kern aus einem Rahmen, an dem in mehreren Arbeitsschritten verschiedene Objekte montiert werden. Zunächst kann die Gabel und der Gepäckträger montiert werden. Dann kann das Lackieren des bereits zusammengestellten Fahrrads beginnen. Ab diesem Zeitpunkt eröffnen sich mehrere parallele Wege. Sobald sämtliche dieser parallelen Arbeitsstränge absolviert wurden, wird in einem abschließenden „Quality Check“ das Fahrrad auf Vollständigkeit und Korrektheit der Montageschritte überprüft.

Um am Ende ein fertiges Fahrrad zu erhalten, müssen somit eine Anzahl von Arbeits-/ Montageschritten jeweils einmal ausgeführt werden. In manchen Fällen ist hierbei die Reihenfolge der Schritte wichtig, während in anderen Fällen der nächste Schritt aus mehreren Möglichkeiten wählbar ist. Es existieren also für jeden Arbeitsschritt Bedingungen, welche erfüllt werden müssen, bevor ein Arbeitsschritt durchgeführt werden kann.

Beispielhaft lässt sich hier anführen, dass eine Klingel erst an das Fahrrad angebracht werden kann, wenn sich bereits der Lenker am Fahrrad befindet.

Um zu gewährleisten, dass das FTF die Abhängigkeiten berücksichtigt, müssen sämtliche Arbeitsgänge und ihre Abhängigkeiten in einer Datenstruktur abgebildet werden. Als Datenstruktur wurde hierfür ein einfacher Digraph, sprich ein gerichteter Graph ohne Schleifen oder Mehrfachkanten, gewählt. Jeder einzelne Arbeitsschritt ist hierbei ein Knoten des Graphen. Außerdem zeigen die Pfeile die Beziehungen der Arbeitsschritte. Hierbei zeigt ein Vorgänger auf jeden seiner möglichen Nachfolger.

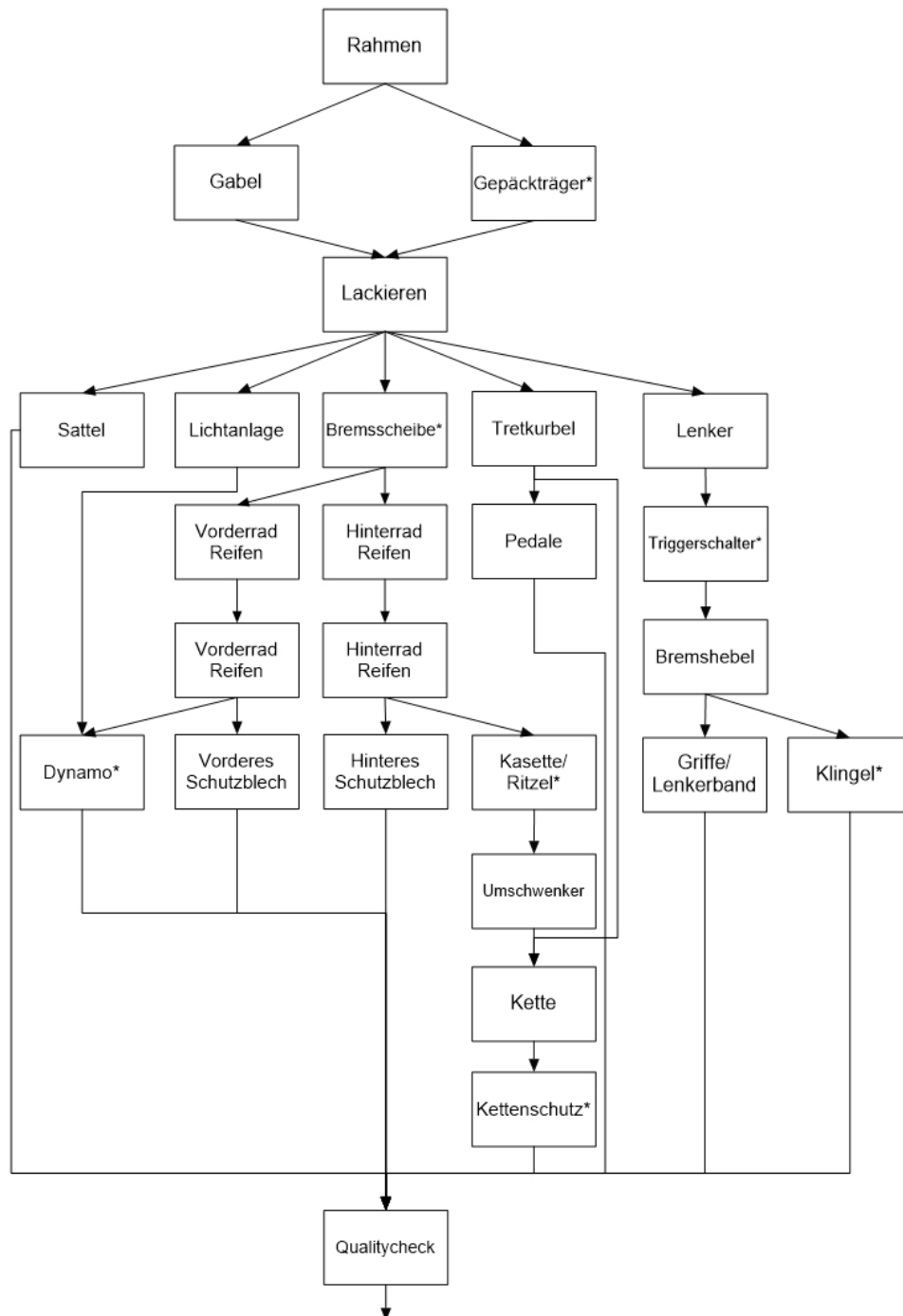


Abbildung 3: Vorranggraph zur Fertigung eines beliebigen Fahrrades

Für das FTF muss lediglich bekannt sein, welche Schritte bereits durchgeführt wurden. Anhand des Graphen lassen sich nun alle möglichen nächsten Schritte ablesen. Die in grün markierten Arbeitsschritten sind bereits durchgeführt. Die blauen Markierungen zeigen an, welche Arbeitsschritte zum jetzigen Zeitpunkt folgen können (vgl. Abbildung 4).

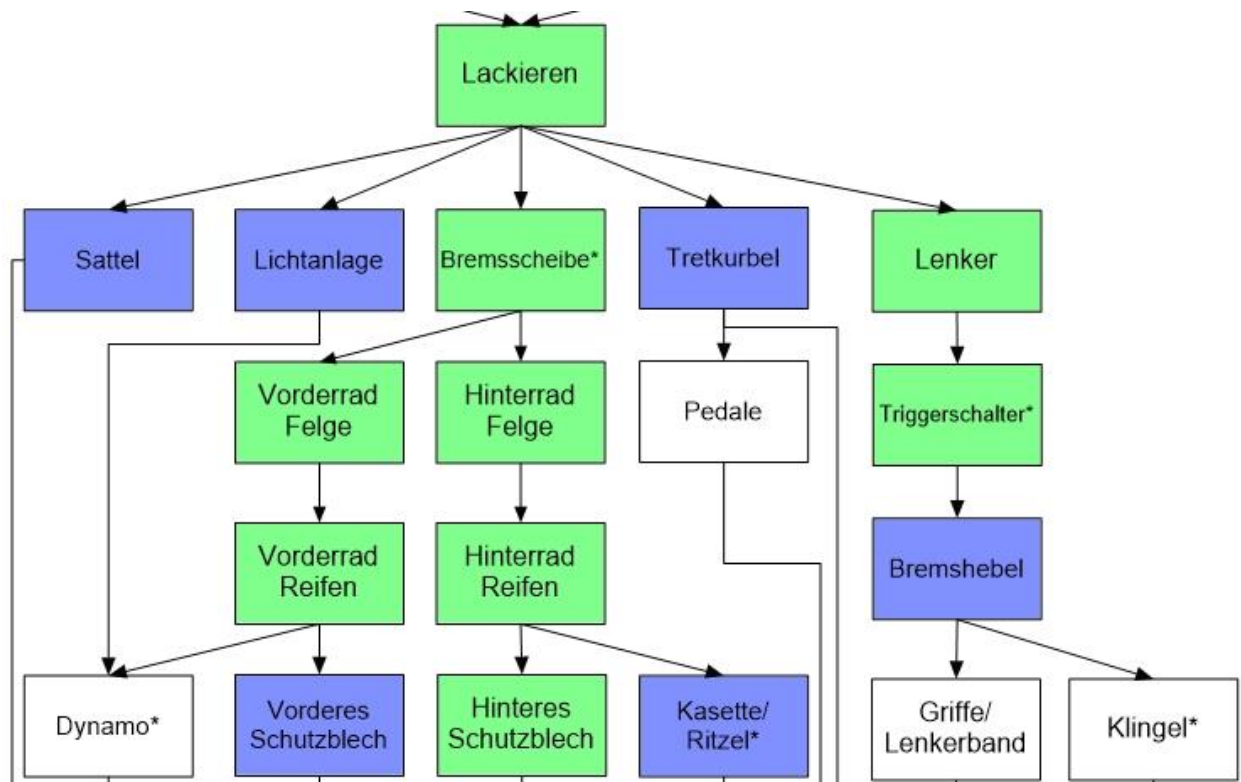


Abbildung 4: Beispielhafte Anwendung des Vorranggraph

Es müssen alle Vorgänger erfüllt sein, damit ein Schritt vollzogen werden kann. Aus diesem Grund kann das Dynamo noch nicht angebracht werden. Es ist zwar bereits der Reifen des Vorderrades montiert worden, aber die Lichtanlage befindet sich nicht am Fahrrad.

Die Umsetzung der Datenstruktur erfolgt auf Datenbankebene. Jeder Arbeitsschritt besitzt eine Schrittnummer und eine Nachfolgernummer. Anhand dieser Nummern lässt sich die Datenstruktur in der Datenbank realisieren.

Implementierung

Nachdem ein Konzept zur Wertschöpfung eines Fahrrades durch den Vorranggraphen sowie ein passendes Datenbankkonzept ausgearbeitet wurde, ist es nun erforderlich diese zu implementieren. Die Implementierung lässt sich unterteilen in

- die Erstellung der Datenbank sowie ihrer Strukturen (Tabelle und Attribute),
- die Festlegung von spezifischen Inhalten aus dem Fertigungskontext (z.B. Lenker von einer bestimmten Marke oder Einzelteil mit bestimmten Eigenschaften),
- die automatische Generierung von zufälligen unspezifischen Daten (z.B. Konfiguration eines Fahrrades oder Kundeneigenschaften),
- die manuelle Generierung selbiger unspezifischer Daten mittels einer Benutzerschnittstelle (UI) und
- die Generierung von Stücklisten und Arbeitsschritten aus den Inhalten der Datenbank.

Abschließend gilt es die aufgeführten Strukturen und Inhalte der Datenbank zu validieren.

Jeder dieser Punkte inklusive der Validierung ist in einem Jupyter-Notebook oder einem Python-Modul umgesetzt. Die Zusammenhänge und Beziehungen der Dateien sind in Abbildung 5 dargestellt.

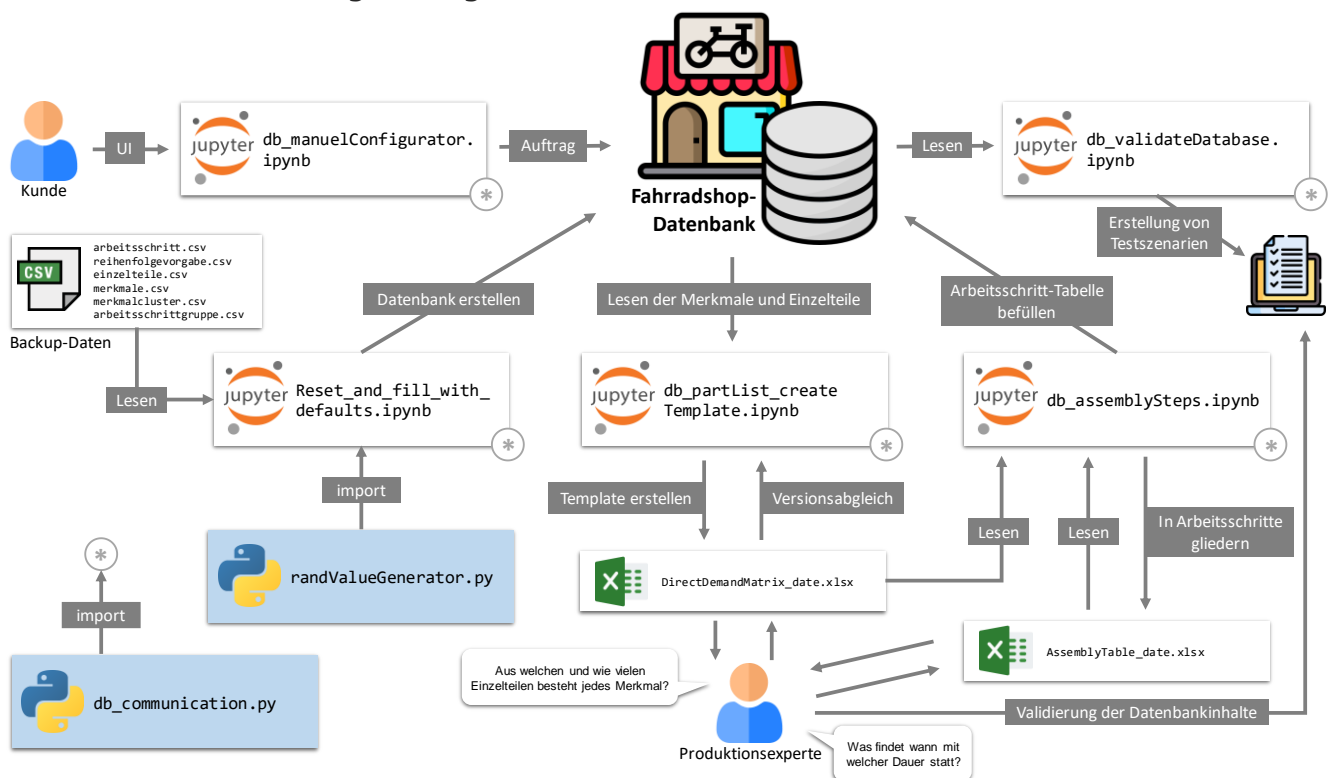


Abbildung 5: Softwarearchitektur der Fahrradshop-Datenbank

In Anlehnung an diese Abbildung werden die einzelnen SW-Module im Nachfolgenden verdeutlicht:

Datei	Beschreibung
Reset_and_fill_ with_defaults.ipynb	In diesem Jupyter-Notebook befindet sich der relationale Aufbau der Datenbank, welcher in Abbildung 2 als ER-Diagramm dargestellt ist. Somit werden im ersten Schritt des Notebooks die Datenbanktabellen mit ihren entsprechenden Attributen und Schlüsseln angelegt. Im zweiten Schritt werden die verschiedenen Backup-Dateien als CSV-Datei eingelesen, welche aus definierten Datensätzen zu den <i>Einzelteilen</i> , <i>Merkmalen</i> sowie <i>Merkmalsclustern</i> gehören. Des Weiteren ist der entwickelte Vorranggraph implizit in den Datensätzen der <i>Arbeitsschrittgruppe</i> und der <i>Reihenfolgevorgabe</i> enthalten. Im dritten Schritt werden generische Datensätze zu den <i>Kunden</i> , <i>Aufträgen</i> und <i>Konfigurationen</i> eines Fahrrades mithilfe des entwickelten Python-Moduls randValueGenerator.py erzeugt. Abschließend werden die eingelesenen und erzeugten Datensätze in die entsprechenden Tabellen der Datenbank geladen.
db_partList_create Template.ipynb	Um die vom Kunden ausgewählten <i>Merkmale</i> mit den <i>Einzelteilen</i> zur Wertschöpfung des Merkmals in Verbindung zu bringen ist eine sogenannte Stückliste vonnöten, welche die Anzahl eines Einzelteils pro Merkmal angibt. Für die Erstellung einer solchen Stückliste wird Expertenwissen aus der Produktion benötigt, welches in Abbildung 5 durch den Produktionsexperten ausgedrückt wird. Die Aufgabe dieses Notebooks besteht somit darin ein Template in Form einer Direktbedarfsmatrix aus den Inhalten der Datenbank zu erzeugen. Zudem beinhaltet das Notebook einen Versionsabgleich, in welchem die vom Produktionsexperten ausgefüllten Verbindungen auf Vollständigkeit und Aktualität überprüft werden.
db_assembly Steps.ipynb	Da die Stückliste bzw. die Direktbedarfsmatrix hohe Ähnlichkeiten und somit Redundanzen zu den <i>Arbeitsschritten</i> aufweisen, wird im ersten Schritt dieses Notebooks, diese Matrix als Excel-Sheet eingelesen. Daraufhin wird die Direktbedarfsmatrix in eine für die Datenbank handelbare Form überführt, wodurch die Arbeitsschritttabelle entsteht. Um die Arbeitsschritte zu vervollständigen, wird nochmals der Produktionsexperte benötigt, welcher zu den Arbeitsschritten eine Zugehörigkeit zu einer

	<i>Arbeitsschrittgruppe</i> herstellen und eine Zeit für den Montageschritt festlegen muss. Abschließend werden die ermittelten Werte der Datenbank in der Tabelle <i>Arbeitsschritte</i> übergeben.
db_manuel Configurator.ipynb	Neben der zufällig generierten Datensätze zu den <i>Kunden</i> , <i>Aufträgen</i> und <i>Konfigurationen</i> durch das Python-Modul <i>randValueGenerator.py</i> , wurde eine manueller Fahrrad-Konfigurator in einem Jupyter-Notebook mittels geeigneter Widgets umgesetzt. Das User-Interface des Notebooks ist in Abbildung 6 dargestellt. Hierbei besteht entweder die Möglichkeit sich als bestehender Kunde einzuloggen oder einen neuen Account anzulegen. Der Unterschied besteht darin, dass beim Login auf einen bestehenden Eintrag in der Kundentabelle zurückgegriffen wird, während bei der Accounterstellung ein neuer Eintrag in dieser Tabelle angelegt wird. Anschließend kann zu jedem Merkmalscluster ein bestimmtes Merkmal ausgewählt werden. Durch einen Klick auf den Button wird der Auftrag abgeschlossen und die jeweiligen Inhalte in die Datenbank übertragen.

Vorname:

Name:

E-Mail:

Options: ☒ Login
☐ Create new account

Konfigurieren Sie Ihr Fahrrad:

Rahmen ▼

Federung ▼

Lackierung ▼

Lenker ▼

Felgen ▼

Radbereifung ▼

Gangschalt... ▼

Sattel ▼

Bremsen ▼

Beleuchtung ▼

Pedalen ▼

Abbildung 6: User-Interface des manuellen Konfigurators

db_validate
Database.ipynb

In diesem Jupyter-Notebook werden Inhalte aus der Datenbank geladen, um hierdurch Testszenarien zu erstellen und somit die Validität der Datenbank sowie dessen Inhalte zu überprüfen. Insgesamt wurden vier unterschiedliche Szenarien in dem Notebook umgesetzt, welche im nachfolgenden vorgestellt werden.

Validierung

Szenario 1:

Es soll ein willkürlicher Auftrag aus der Datenbank ausgewählt werden, dessen Kunde und zugehörige Fahrradkonfiguration ausgegeben werden soll.

Ausgabe:

AuftrNr	Bestelldatum	Lieferdatum	KundenNr
1	2020-02-20	2020-03-05	1
KundenNr	Name	Vorname	Mail
1	Woodrow	Suitt	woodrow.suitt@web.de
KonfigNr	AuftragNr	MerkmalNr	Merkmal Bezeichnung
100	1	100004	Unisex Rennrad Aluminium
101	1	200002	Trekkingfederung Standard
102	1	300002	Weiss uni
103	1	400002	Lenker Komfort
104	1	500006	Felgen Gelaende Pro
105	1	600002	Bereifung Komfort
106	1	700003	Gelaende Kettenschaltung mechanisch
107	1	800005	Rennradsattel Herren
108	1	900003	Scheibenbremse Standard
109	1	1000005	Beleuchtung Pro
110	1	1100002	Trekking Pedale

Ergebnis der Validierung:

Die Verbindung der einzelnen Tabellen untereinander konnte als schlüssig bestimmt werden. Des Weiteren entsprechen die Dateninhalte den Erwartungen an ein Fahrradshop



Szenario 2:

Das in Szenario 1 zufällig ausgewählte Fahrrad soll in seine Einzelteile mittels der Datenbank aufgelistet werden.

Ausgabe:

Merkmal	Einzelteil
Unisex Rennrad Aluminium	VOTEC VRC Framekit
	Tubus Seitenstaender schwarz
	Kein Gepaecktraeger
	Kein Schutzblech
Trekkingfederung Standard Weiss uni	RockShox Paragon Gold TK SA Federgabel
	Weiss Uni Basislack
	2K HS-Klarlack
	2K HS Haerter fuer Klarlack
Lenker Komfort	Silikonentferner
	XLC HB-C02 City-/Trekking-Lenker
	Red Cycling Products Super Ergo Grip
	BBB Loud & Clear BBB-11 Klingel
Gelaende Kettenschaltung mechanisch	XX1 Eagle DUB
	XX1 Eagle Triggershifter
	X-Horizon X-Sync Roller Bearing Clutch Cage Lock
	XX1 Eagle Kette
	XG-1295 Eagle Kassette
	Direct Mount X-Sync
Rennradsattel Herren	Horn Catena A08/48 Kettenschutz
	SQlab 612 Ergowave Sattel S-Tube
Scheibenbremse Standard	Shimano BL-T4000 Bremshebel
	Shimano BR-R7000 Felgenbremse Dual-Pivot Hinterrad CS-51
	Jagwire Mountain Sport
	Shimano Alfine BL-S7000 Scheibenbremse I-Spec II Vorne
Beleuchtung Pro	Shimano Deore SM-RT56 Bremsscheibe 6-loch
	Shimano BH90-JK-SSR
	MonkeyLink MonkeyLight 70 Lux Recharge Vorne
	MonkeyLink MonkeyLight 70 Lux Recharge Hinten
	Dynamo RECHTS 6V/3W Überspannungsschutz Doppelan-schluss Kunststoff
	Leuchtstreifen
Trekking Pedale	Xpedo Detox Pedale
Felgen Gelaende Pro	Tune Race 3.0 MTB Laufradsatz 29 Zoll X-12 XD
	Shimano Nexus DH-C3000-3N Nabendynamo
Bereifung Komfort	Continental Contact Plus Reifen SafetyPlus Breaker 28 Zoll Draht Reflex
	NoTubes Universal Tubeless Ventil 35mm MTB
	Continental MTB 26 Zoll Schlauch

Ergebnis der Validierung:

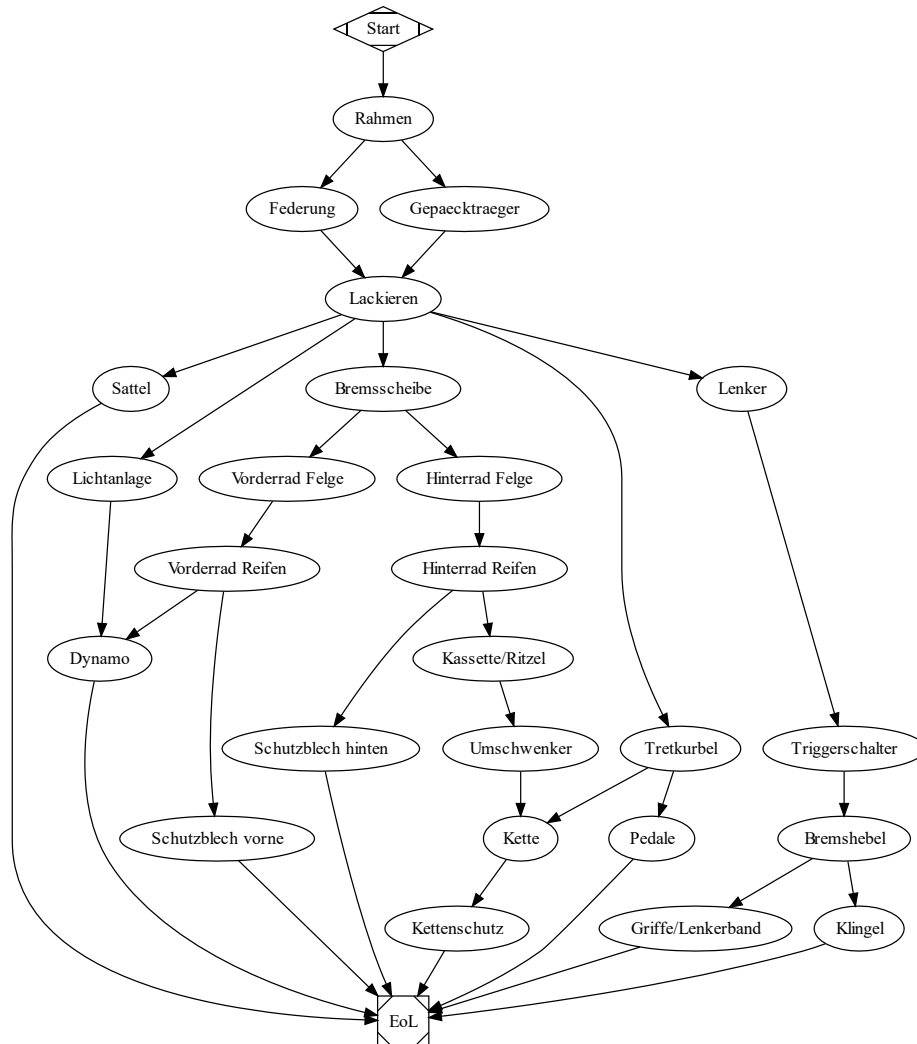
Jedem Merkmal lässt sich eine sinnvolle Kombination an Einzelteilen zuordnen, aus denen es besteht. Des Weiteren ist eine hohe Komplexität des Datensatzes durch die ebenso hohe Varianz für zukünftige Anwendungen gewährleistet.



Szenario 3:

Es sollen die *Arbeitsschrittgruppen* mit der *Reihenfolgeplanung* in Verbindung gebracht werden, um hieraus grafisch den Algorithmus zur Fertigung eines Fahrrades abzuleiten.

Ausgabe:



Ergebnis der Validierung:

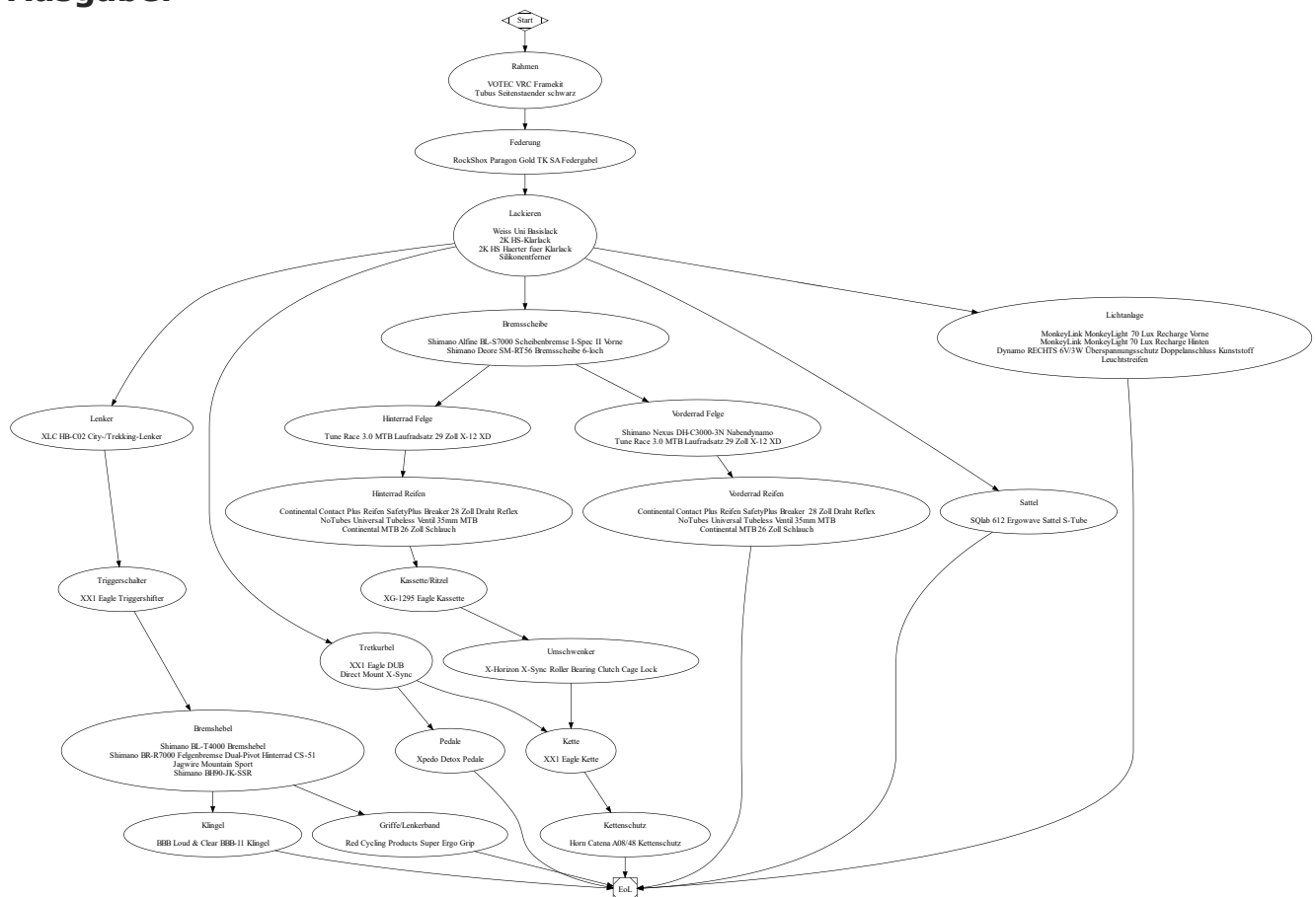
Durch die Inhalte der Datenbank lässt sich ein Graph zeichnen, welcher mit den Überlegungen des konzipierten Vorranggraphen übereinstimmt.



Szenario 4:

Für den zufällig ausgewählten Auftrag soll nun der Fertigungsablauf mit den aufgeführten Einzelteilen angezeigt werden.

Ausgabe:



Ergebnis der Validierung:

Anhand des Vorranggraphen für einen spezifischen Auftrag, lässt sich die Funktionalität eines sinnhaften Fertigungsablaufs und einer sinnhaften Wertschöpfung für Fahrräder nachweisen.



Layout of Workstations

Das Layout of Workstations dient zur Veranschaulichung der Funktionsweise der Fahrradfabrik. In dieser Fabrik wird das Fahrrad, angefangen beim Rahmen, an verschiedenen Arbeitsstationen schrittweise nach Kundenwünschen zusammengesetzt (vgl. Abbildung 3). Die Bauteile werden – wie in der Projektmotivation bereits erklärt – auf einem FTF von Arbeitsstation zu Arbeitsstation transportiert. Das FTF besitzt einen Monitor o.ä., auf dem die auszuführenden Arbeitsschritte an der jeweiligen Arbeitsstation, angezeigt werden. Weiterhin kann der vollständige, geplante Produktionsweg des Fahrrads anhand des Vorranggraphen angezeigt werden.

Die Fabrikhalle besteht aus 7+1 Arbeitsstationen (Workstations), einem Parkplatz mit Platz für bis zu 24 FTFs und einem zentralen Material- und Werkzeuglager. Die systematische Anordnung dieser Bestandteile ist in Abbildung 7 dargestellt. An jeder Arbeitsstation befindet sich ein Puffer-Lager, dass von einem FTF aus dem Zentrallager mit den benötigten Materialien versorgt wird. Des Weiteren ist ein Haltebereich vorhanden, an dem bis zu drei FTFs verweilen können.

Station 1: An dieser Station wird der Rahmen des Fahrrads auf dem FTF fixiert und mit einer Gabel bestückt, optional findet hier die Ausrüstung eines Gepäckträgers statt.

Station 2: Das FTF durchläuft hier – ähnlich wie bei einer Waschstraße – die Station, während Rahmen und Gabel mit Sprühpistolen in der Wunschfarbe eingefärbt werden. Die Sprühpistolen arbeiten gezielt und effizient, weshalb das FTF von Farbspritzern befreit bleibt. Danach gelangt das Bauteil in den Trocknungsbereich der Arbeitsstation. Nach dieser Station hat das FTF die Auswahlmöglichkeit zwischen Station 3, 4, 5 und 6.

Station A: Hier werden parallel die Vorder- und Hinterreifen für die Aufträge hergestellt und bei Bedarf an Station 3 weitergeleitet. Das Vorderrad kann dabei mit einem Nabendynamo bestückt sein und das Hinterrad optional mit einer Kassette/Ritzel. Die Abfolge der Schritte gestaltet sich folgendermaßen: Zuerst wird die gewünschte Nabe genommen und mit Speichen mit der Felge verbunden. Daraufhin kommt der Schlauch auf die Felge, das Ventilrohr wird festgemacht und das Ventil wird eingedreht. Der Schlauch wird etwas mit Luft befüllt, der Mantel kommt auf die Felge und wird anschließend komplett aufgepumpt. Daraufhin wird die Schutzkappe angebracht und das Ventil verschraubt.

Station 3: Diese Workstation rüstet das Fahrrad mit Vorderrad und Hinterrad aus. Dabei werden Scheibenbremsen oder Backenbremsen für vorne und hinten eingebaut. Der Umschwenker für das Schaltwerk wird hier angebracht, sofern es sich bei der Schaltart nicht um eine mit Kassette/Ritzel handelt.

Abhängigkeit: Wenn das FTF vorher bereits Station 5 durchlaufen hat, werden hier die Schaltgriffe mit der Schaltung und die Bremshebel mit den Vorder- und Hinterradbremse verbunden.

Abhängigkeit: Wenn das FTF vorher bereits Station 4 durchlaufen hat **UND** sich ein Nabendynamo im Vorderrad befindet, wird die Lichtanlage hier mit der Energiequelle verbunden.

Station 4: Das entstehende Fahrrad bekommt an dieser Station seine Lichtanlage montiert. Dafür werden die Lichter vorne und hinten sowie Leuchtstreifen angebracht, optional kann hier ein Seitenläuferdynamo angebracht werden.

Abhängigkeit: Wenn das FTF vorher bereits Station 3 durchlaufen hat **UND** sich ein Nabendynamo im Vorderrad befindet, wird die Lichtanlage hier mit der Energiequelle verbunden.

Station 5: Der Lenker wird an dieser Station montiert. Danach werden die Schaltgriffe für die Gangschaltung, die Bremshebel, optional eine Klingel und die Griffe (oder das Lenkerband) am Lenker angebracht.

Abhängigkeit: Wenn das FTF vorher bereits Station 3 durchlaufen hat, werden hier die Schaltgriffe mit der Schaltung und die Bremshebel mit den Vorder- und Hinterradbremse verbunden.

Station 6: An dieser Station bekommt das Fahrrad seinen Sattel.

Station 7: Hier wird das Fahrrad mit der Kurbelgarnitur und dem Innenlager für die Tretkurbel ausgerüstet, danach werden die Pedale verbaut.

Abhängigkeit: Das FTF muss vorher **mindestens** Station 3 durchlaufen haben, damit die Kette auf das Fahrrad gespannt werden kann.

Nach dem Durchlaufen einer Station hat ein FTF die Möglichkeit für eine kurze Zeit (z.B. < 1 Minuten) zu parken. Bei längeren Wartezeiten fährt das FTF zum gekennzeichneten Parkbereich. Je nach der zur Verfügung stehenden Fläche für die Fabrik können an den meisten Arbeitsstationen parallel mehrere Aufträge bearbeitet werden. Die genaue Größe des Parkplatzes richtet sich nach den in Umlauf befindenden FTFs.

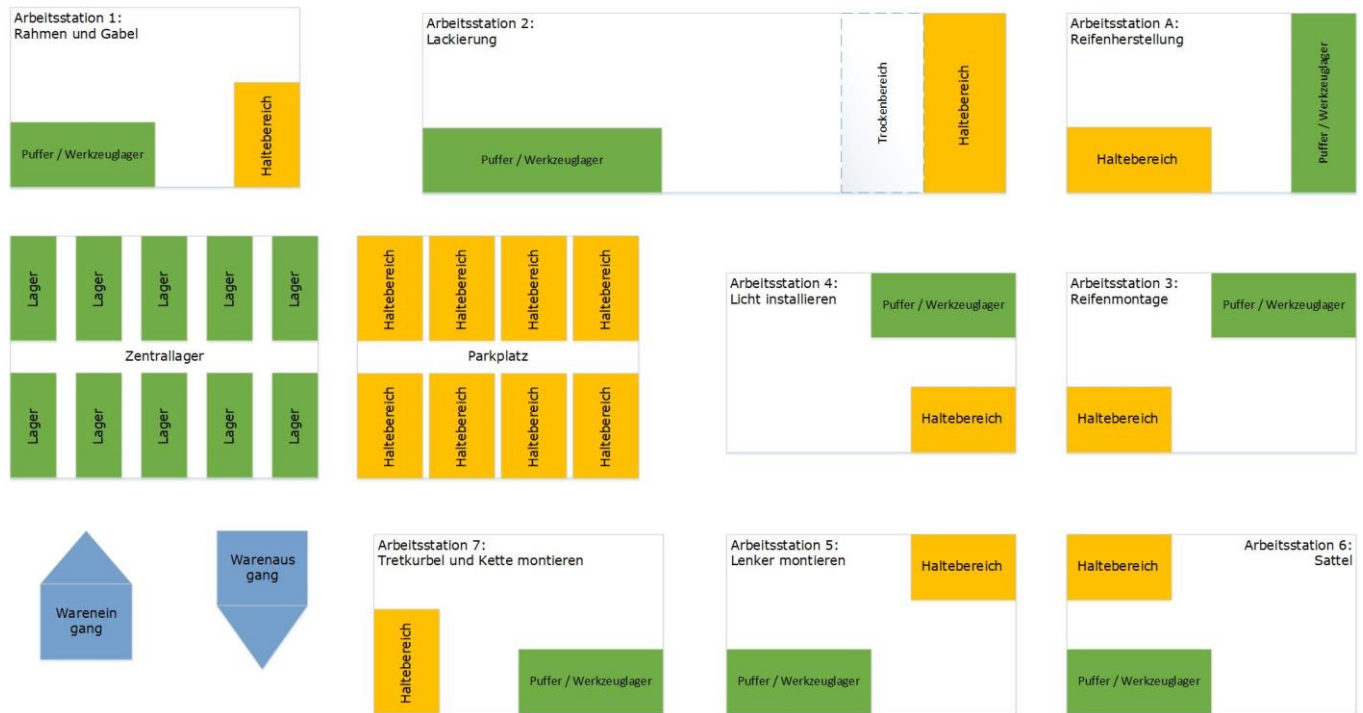


Abbildung 7: Layout of Workstations

Konzept zur FTF-Wegeplanung

Da das FTF zu Produktionsbeginn fest mit einem spezifischen Kundenauftrag verknüpft wird, soll das FTF im Produktionsverlauf die bestmögliche Route bei der Produktion nehmen. Dabei müssen neben einer sinnvollen Arbeitsstationsabfolge natürlich auch die Wege zwischen den Stationen berücksichtigt werden. Hierbei könnten etwa große Distanzen ebenso nachteilig sein, wie Stauverhalten oder ähnliche Faktoren. Darüber hinaus können auch Folgestationen selbst die Routenwahl beeinflussen: Etwa, wenn ein benötigtes Einzelteil gerade nicht lieferbar ist oder auch wenn aktuell ein Auftragsstau an der Station anfällt. Das FTF muss also regelmäßig (z.B. nach jeder Arbeitsstation) prüfen welche Route die geschickteste ist, um seinen Auftrag möglichst effizient zu erfüllen.

Es gibt verschiedene Konzepte zur Erfüllung einer solchen Routenplanung (bspw. autonome Software-Agenten, die mit Stationen verhandeln) gemeinsam haben sie jedoch, dass jedes FTF seinen Auftrag im gesamten Produktionsumfeld so effizient wie möglich abfertigen soll.

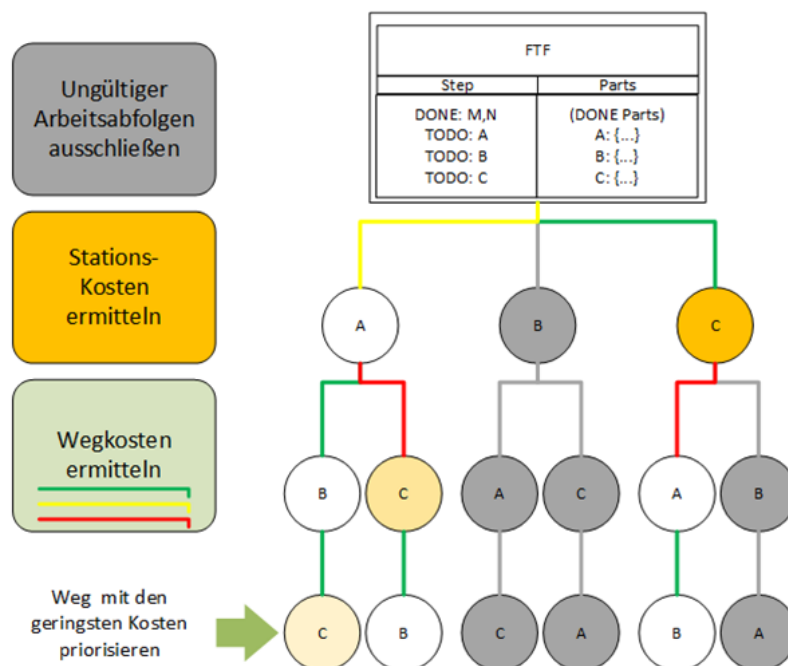


Abbildung 8: Beispiel für effiziente Wegeplanung

Beispiel aus Abbildung 8:

Das FTF muss zur Fertigstellung des Auftrags noch Arbeitsstation A, B und C ansteuern. Da für ein funktionales Endprodukt Station B nach Station A angefahren werden muss sind diese nicht gültigen Abfolgen bereits grau dargestellt.

Das FTF führt nun eine Routenplanung durch: Station C ist in diesem Fall zwar günstig gelegen (geringe Wegkosten – vgl. grüne Verbindung), jedoch herrscht hier momentan ein Auftragsstau und die Station ist stark ausgelastet. Sinnvoll wäre es also Station C erst später anzufahren und zunächst Station A zu besuchen, auch wenn die Wegkosten etwas höher sind. Weiterhin kann Station C auch nach Station B günstig erreicht werden. Entlang der Pfade des Diagramms kann so der aktuell

effizienteste Weg berechnet werden. In der Umsetzung könnten die Einfärbungen durch normierte Werte dargestellt werden, die von Software-Agenten nach jeder abgeschlossenen Station mit den Wegen und Stationen verhandelt werden. Entlang den Pfaden würden dann die Kosten aufsummiert und der Weg mit den geringsten Kosten bzw. der höchstmöglichen Effizienz im Produktionsumfeld gewählt.