Verwaltungs- und Wirtschaftsakademie und Berufsakademie Göttingen
Dr. Helge Fischer

Objektorientierte Analyse und Entwurf eines Webservice für die Optimierung von Produktionsprogrammablaufplänen

Thesis

Steffen Schütz
Sägemüller Str. 38
38678 Clausthal-Zellerfeld

I10.W.002

28. November 2013

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Das Thema dieser Arbeit ist die Objekt-orientierte Analyse und Entwicklung als systematische Vorgehensweise für die Erstellung von komplexen Systemen. Deren weiterer Verlauf die Objekt-orientierte Softwareentwicklung, ist durch die verschiedenen Konzepte durchgängig geprägt. Den Methoden der Objekt-orientierung liegt eine Notation zugrunde die sowohl aus Grafiken wie auch Text besteht. Wesentlich für die Notation ist eine Standardisierung die sich über den Verlauf der letzten 20 Jahre entwickelt hat. Zielsetzung dieser Untersuchung ist die Anwendbarkeit der Objekt-orientierten Konzepte auf die Entwicklung eines konkreten Softwaresystems. Ausgangslage hierfür ist die Vision Produktionsablaufpläne mit Hilfe eines Softwaresystems zu optimieren, das die Besonderheiten einer Service-orientierten Architektur für sich nutzt und somit eine vertikale Integration ermöglicht. Die Erfassung der Ablaufplanung in der Produktion und der damit verbundenen Optimierungsprobleme nehmen dabei einen besonderen Stellenwert ein. Um die verschiedenen Themenpunkte fundiert darstellen zu können wird im Grundlagenteil eine Einführung in relevante Eckepunkte des Produktionsmanagements dargestellt. Die Prozesse und Methoden des Operations Research sowie die wesentlichen die Lösungsansätze für Optimierungsprobleme werden mit einem ersten Blick auf die Ablaufplanung dargestellt. Für ein grundlegendes Verständnis der Systemtheorie sowie der darauf aufbauenden Systemanalyse wird ein weiteres Kapitel grundlegendes Wissen vermitteln. Einen Schwerpunkt wird in den darauf folgenden Kapiteln zu den Themen der Objekt-orientierten Modellieren, sowie der dazu gehörenden Notationen ausführlich behandelt. Der Bereich Grundlagen wird in dieser Thesis mit dem Themenkomplex der Service-orientierten Architektur und deren konkrete Implementierung als Webservice abgeschlossen. Hier werden wesentliche Begriffe, Methoden und auch konkrete technische Details vermittelt. Mit der Ablaufplanung in der Produktion wird die betriebswirtschaftliche Problemstellung um die Optimierung von Produktionsablaufplänen erfasst und ausgewählte Lösungsansätze dargestellt. Dabei werden aufbauend auf einer Einführung zum Themenkomplex und seinen Problemstellungen einzelne Lösungsansätze im Detail erarbeitet. Ein Ausblick auf mögliche Verfahren für komplexere Szenarien wird das Kapitel informativ abschließen. In einem Praxisbeispiel wird die objekt-orientierte Analyse und der Entwurf eines konkreten Webservices auf der Grundlage von PHP dargestellt und somit die Anwendung der in dieser Thesis zusammengetragenen Grundlagen dargestellt.

2 Grundlagen

2.1 Produktionsmanagement

Der Begriff der Produktion in einer Unternehmung umfasst die Herstellung von materiellen und immateriellen Gütern.[1] Die Herstellung lässt sich dabei als Kombinationsprozess in drei Elemente unterteilen.

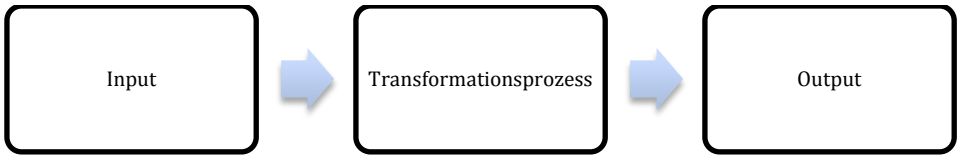

Abb. 1: Kombinationsprozess

Quelle: In Anlehnung an Zäpfel, Günther: Produktionswirtschaft Operatives Produktionsmanagement, S. 2

Das Element Output umfasst alle Enderzeugnisse und Leistungen einer Produktion. Input ist das Element welches aus Produktionsfaktoren besteht die einem Transformationsprozess zugeführt werden. Produktionsfaktoren unterscheidet man nach Elementarfaktoren und dispositiven Faktoren. Zu den Elementarfaktoren zählen Betriebsmittel, Werkstoffe und Menschliche Arbeit. Als zentrales Element zwischen Input und Output stellt der Transformationsprozess die Kombination von Produktionsfaktoren dar.[2]

Das operative Produktionsmanagement stellt mit den Teilgebieten Faktorbereitstellungsplanung, Produktionsprozessplanung und operative Produktionsprogrammplanung ein Führungssystem der Produktion dar. Auf das Input Element der Produktion bezieht sich die Faktorenbereitstellungsplanung die sich in Materialbedarfsplanung, Bestelldisposition, Lagerhaltung und Materialeinkauf unterteilen lässt. Die Produktionsprozess Planung wird dem Produktionselement Transformationsprozess zugeordnet und umfasst die Losgrößenplanung sowie die Maschinenbelegungsplanung. Operative Produktionsprogrammplanung wird dem Produktionselement Output mit den Teilbereichen zur Bestimmung der Produktarten und –mengen, Festlegen der gewünschten Lagerhaltung, sowie Bestimmung von Lieferterminen zugeordnet.[3]

[1] Vgl. Zäpfel, Günther: Grundzüge des Produktions- und Logistikmanagement, S. 1
[2] Vgl. Zäpfel, Günther: Produktionswirtschaft Operatives Produktions-Management, S. 2-7
[3] Vgl. Zahn, Erich, Schmid, Uwe: Produktionswirtschaft, S. 162f.

2.2 Operations Research

Das Wissensgebiet des Operations Research wird über einen komplexen Prozess abgebildet, der sich in seinen Teilschritten mit der Modellbildung und darauf basierenden Entwicklung von Algorithmen zur optimalen Lösung von Entscheidungsproblemen beschäftigt. Als Modell wird eine vereinfachte Abbildung der Realität bezeichnet, das wesentliches von unwesentlichen trennt.[4]

In der Literatur werden als die wichtigsten Problemstellungen des Operations Research Kombinatorische-, Lagerhaltungs-, Ersatz-, Wartezeit- und Konkurrenzprobleme aufgeführt. Die Kombinatorischen Probleme lassen sich in Reihenfolge und Zuordnungsprobleme unterteilen. Bei den Zuordnungsproblemen unterscheidet man zwischen Transportproblemen und der Aufstellung von optimalen Produktionsprogrammen.[5] Der für das Thema dieser Thesis relevante Problembereich der Auftragsreihenfolge beschäftigt sich mit der optimalen Reihenfolge von Aufträgen auf ein oder mehreren Maschinen.[6]

Die praktische Umsetzung des Operations Research als Prozess lässt sich in den Bereich der Konstruktion von Modellen, der mathematischen Umsetzung eines Modells und der Verwendung von Ergebnissen aus der mathematischen Umsetzung unterteilen.[7] Am Anfang des Operations Research Prozess steht das Identifizieren und Analysieren einer Problemstellung. Aus der Analyse folgt die Zielformulierung und Festlegung von Handlungsmöglichkeiten. Eine Zielvorgabe ermöglicht die Formulierung eines mathematischen Modells, für das Daten beschafft werden. Unter Auswahl eines Lösungsverfahrens aus dem Operations Research, wird das mathematische Modell gelöst. Über die dabei verwendeten Daten erhält man eine oder mehrere optimale Lösungen, die mit Blick auf das reale Problem bewertet werden.[8]

Im Operations Research werden die Lösungsverfahren in Teilbereiche unterteilt deren Anzahl sich in der Literatur unterscheidet.[9] Die Lineare Programmierung beschäftigt sich mit Lösung von linearen Problemstellungen. Als Lösungsverfahren dienen graphisch Darstellung und der Simplex-Algorithmus.[10] Nichtlineare Programmierung unterscheidet sich von der linearen Programmierung darin, dass die

[4] Vgl. Domschke, Wolfgang, Drexl, Andreas: Einführung in Operations Research, S. 1-3
[5] Vgl. Ellinger, Theodor, Beuermann, Günter, Leisten, Rainer: Operations Research, S. 8-11
[6] Vgl. Ellinger, Theodor, Beuermann, Günter, Leisten, Rainer: Operations Research, S. 8
[7] Vgl. Gal, Tomas, Horst, Reiner, Heinz, Isermann, Müller-Merbach, Heiner: Einführung in Operations Research 1, S. 23
[8] Vgl. Domschke, Wolfgang, Drexl, Andreas: Einführung in Operations Research, S. 1-2
[9] Vgl. Domschke, Wolfgang, Drexl, Andreas: Einführung in Operations Research, S. 7-8
[10] Vgl. Zimmermann, Hans-Jürgen: Operations Research, S. 69-77

Zielsetzungen oder Nebenbedingungen nicht linear formuliert sind.[11] Probleme die sich aufteilen lassen, so das einzelne Zustände entstehen die von vorgelagerten Zuständen abhängig sind werden mit der dynamischen Optimierung gelöst.[12] Die Entscheidungsbaumverfahren enthalten Berechnungs- und Suchverfahren für die Ermittlung einer optimalen Lösung. Hierzu zählt das Branch & Bound Verfahren.[13] Mit der Netzplantechnik verfügt das Operations Research über Lösungsverfahren zur optimalen Planung von Projekten. Als Projekt versteht man im Operations Research die Teilarbeiten die Zeit beanspruchen und in Beziehung zueinander stehen.[14]

Die strukturierte Untersuchung von Warteschlangen ist Gegenstand der Warteschlangentheorie. Warteschlangen treten auf wenn die Nachfrage nach einem Service für einen Zeitraum größer ist als die Kapazität des Serviceanbieters.[15] Warteschlangenmodelle werden im Operations Research als Prozess dargestellt der sich in die Bereiche Input-Quelle, Warteschlange, Servicemechanismus und Output unterteilt.[16]

In der Spieltheorie werden Entscheidungssituationen untersucht bei der mehrere Akteure, die sogenannten Spieler, an einem ökonomischen Prozess mitwirken.[17] Für komplexe Problemstellungen die keine analytische Modellbildung zulassen wird im Operations Research die Simulation eingesetzt.[18] Die Simulationstechnik ist von den Modell- und Systemexperiment geprägt, welches systematisches Probieren, planmäßiges Suchen und modellmäßigen Experimentieren umfasst.[19] Systematische Suchverfahren werden im Operations Research als heuristische Verfahren bezeichnet, die mit kleinen Rechenaufwand gute Lösungen produzieren welche im Vergleich zu den exakten Verfahren nicht mit Sicherheit eine Optimale Lösung finden.[20] Der Einsatz von heuristischen Verfahren im Operations Research erfolgt unter Bewertung von Rechenaufwand und Güte einer möglichen Lösung.[21]

---

[11] Vgl. Zimmermann, Hans-Jürgen: Operations Research, S 188
[12] Vgl. Ellinger, Theodor, Beuermann, Günter, Leisten, Rainer: Operations Research, S. 12
[13] Vgl. Ellinger, Theodor, Beuermann, Günter, Leisten, Rainer: Operations Research, S. 12
[14] Vgl. Burkhard, Rainer, Neumann, Klaus, Ohse, Dietrich: Grundlagen des Operations Research, S. 168
[15] Vgl. Beckmann, Martin, Gehring, Hermann, Kistner, Klaus-Peter, Schneeweiß, Christoph, Schwödiauer, Gerhard, Zimmermann, Hans-Jürgen: Grundlagen des Operations Research 3, S. 256f.
[16] Vgl. Hillier, Frederick, Liebermann, Gerald: Operations Research Einführung, S.502-503
[17] Vgl. Neumann, Klaus: Operations Research Verfahren Band 1, S. 194
[18] Vgl. Hillier, Frederick, Liebermann, Gerald: Operations Research Einführung, S. 773f.
[19] Vgl. Zimmermann, Werner: Operations-Research Quantitative Methoden zur Entscheidungsvorbereitung, S. 336
[20] Vgl. . Zimmermann, Werner: Operations-Research Quantitative Methoden zur Entscheidungsvorbereitung, S. 150
[21] Vgl. Zimmermann, Hans-Jürgen: Operations Research Methoden und Modelle, S. 271

2.3 Systemanalyse

Die Systemtheorie verfolgt mit einer systemischen Betrachtungsweise eines festgelegten Umfelds eine Modellbildung, die eine Wechselwirkung unterschiedlicher Elemente berücksichtig. Grundlegend wirkt die Systemtheorie abstrahierend auf die Betrachtungsweise der Realität ein.[22]

Zum Begriff System als Grundlage der Systemtheorie lässt sich die Herkunft im griechischen Begriff ‚systema' finden, der ein aus mehreren Elementen zusammengesetztes Ganzes beschreibt.[23] Diese Systemelemente dienen auf unterste Ebene der Beschreibung des Systems und werden über eine zielgerichtete Betrachtung der Realität gebildet. Dabei werden den Systemelementen Eigenschaften und Funktionen zugeordnet. Beziehungen zwischen Systemelementen werden als Systemrelationen bezeichnet und bilden zusammen mit den Elementen die Systemstruktur. Über die Systemgrenze wird die Systemstruktur von der Systemumwelt getrennt. Die Systemumwelt enthält alles was nicht dem System zugeordnet werden kann und steht mit Systeminput, die Wirkung von Umwelt auf das System, und Systemoutput, die Wirkung von System auf die Umwelt, mit der Systemstruktur in Verbindung.[24] In der Systemtheorie werden Systeme nach den Eigenschaften Entstehungsart, Lebensbereich, Außenbeziehung, Konkretisierungsgrad, Abhängigkeit von der Zeit und ob das Verhalten vom Zufall abhängig ist klassifiziert.[25] Diese Eigenschaften weisen konkurrierende Ausprägungen auf. In der Entstehungsart unterscheidet man zwischen künstlichen und natürlichen Systemen. Reale und ideelle Systeme sind Ausprägung im Lebensbereich. Außenbeziehungen zur Systemumwelt werden in geschlossene und offene Systeme unterschieden. Abstrakte und konkret greifbare Systeme beschreiben den Konkretisierungsgrad eines Systems. Die Abhängigkeit von der Zeit wird in statische und dynamische Ausprägung beschrieben. Stochastische Systeme geben ein vom Zufall abhängiges Verhalten, so wie deterministische ein nicht abhängiges Verhalten an.[26]

Die Systemanalyse hat eine hohe Bedeutung als Prozess in der Systementwicklung von Softwaresystemen. Sie wird in Phasen- und Vorgehensmodellen abgebildet die sich in ihrer Ausprägung unterscheiden. Eine Abgrenzung erfolgt zwischen System-

[22] Vgl. Krallmann, Hermann, Schönherr, Marten, Trier, Matthias: Systemanalyse im Unternehmen, S. 60
[23] Vgl. Krallmann, Hermann, Schönherr, Marten, Trier, Matthias: Systemanalyse im Unternehmen, S. 60
[24] Vgl. Krallmann, Hermann, Schönherr, Marten, Trier, Matthias: Systemanalyse im Unternehmen, S. 60-62
[25] Vgl. Heinrich, Gert: Allgemeine Systemanalyse, S. 8
[26] Vgl. Heinrich, Gert: Allgemeine Systemanalyse, S. 8

analyse als vollständiger Entwicklungsprozess, als frühe Phase die sich in Ist-Analyse und Sollkonzept unterteilt sowie einem iterativen Prozess, dessen Phasen parallel ablaufen.[27] Das Vorgehensmodell das die Systemanalyse als vollständigen Prozess betrachtet wird in die Phasen Projektbegründung, Ist-Analyse, Sollkonzept, Realisierung und Implementierung unterteilt.[28] In der Literatur wird der Prozess in unterschiedlicher Ausprägung so dargestellt das auf die Sollkonzept-Phase der Systementwurf folgt.[29]

In der Phase der Projektbegründung werden alle für die Initialisierung wesentlichen Aktivitäten zusammengefasst und umgesetzt. Grundlage für die Projektbegründung ist die Festlegung von Projektzielen und der Systemgrenze. Die Festlegung der Systemgrenze erfolgt durch Abgrenzung des Untersuchungsgebiets gegenüber der Systemumwelt.[30]

Durch den in der Projektgründung entstandenen Projektauftrag wird die Phase der Ist-Analyse gestartet, deren Aufgabe die Dokumentation eines bestehenden Systems sowie die eines neu zu erstellenden Systems ist. Mit einer anschließenden Potentialanalyse werden Schwachstellen analysiert und Verbesserungsvorschläge erarbeitet.[31]

Aufbauend auf den Ergebnissen aus Ist- und Potentialanalyse wird ein Sollkonzept erstellt, das inhaltlich eine Leistungsbeschreibung des neuen Systems liefert. Das Resultat der Leistungsbeschreibung ist ein Pflichtenheft das eine detaillierte Beschreibung des neuen Systems enthält.[32] Ziel des Systementwurfs ist der Entwurf von Lösungen für das im Pflichtenheft beschriebene System. Die Ausprägung dieser Phase ist vom gewählten Vorgehensmodell abhängig.[33]

In der Realisierungsphase wird aufgrund des im Sollkonzept erstellen Pflichtenhefts die Entscheidung über Fremdbezug einer Standardsoftware oder einer Eigenentwicklung der Software getroffen.[34]

Die Implementierungsphase stellt die Auswahl und Umsetzung einer Einführungs-Strategie für das fertige System dar. Eine Einführung die sich gleichzeitig auf alle beteiligten Bereiche eines Systems auswirkt, wird als Big-Bang-Strategie bezeichnet.

---

[27] Vgl. Heinrich, Gert: Allgemeine Systemanalyse, S. 1-5
[28] Vgl. Krallmann, Hermann, Schönherr, Marten, Trier, Matthias: Systemanalyse im Unternehmen, S.135f.
[29] Vgl. Heinrich, Gert: Allgemeine Systemanalyse, S. 3f.
[30] Vgl. Krallmann, Hermann, Schönherr, Marten, Trier, Matthias: Systemanalyse im Unternehmen, S. 143f.
[31] Vgl. Heinrich, Gert: Allgemeine Systemanalyse, S. 25f.
[32] Vgl. Heinrich, Gert: Allgemeine Systemanalyse, S. 47f.
[33] Vgl. Heinrich, Gert: Allgemeine Systemanalyse, S. 50f.
[34] Vgl. Krallmann, Hermann, Schönherr, Marten, Trier, Matthias: Systemanalyse im Unternehmen, S. 178f.

Dem gegenüber stehen die sukzessive und pilotierte Einführungs-Strategie. Merkmal dieser Einführungs-Strategien sind die kleinen, sukzessiven und aufeinander aufbauenden Schritte.[35]

2.4 Objekt-orientierte Modellierung

Die Modellbildung ist in der Wirtschaftsinformatik ein Verfahren zur Beschreibung und vereinfachten Darstellung von realen Systemen. Zielsetzung der Modellierung ist durch Abstraktion über die Elemente und deren Beziehungen Klarheit über die Funktionsweise eines Systems zu schaffen.[36] In der objektorientierten Modellierung wird das Objekt als Grundeinheit betrachtet, über das die Systemelemente abgebildet werden. Ein Objekt kapselt die Daten eines Systemelements und seine spezifischen Operationen die auf den Daten ausgeführten werden.[37] Die objektorientierte Modellierung folgt dabei den Grundkonzepten der Klassen und Objekte, der Vererbung sowie Mehrfachvererbung, dem Polymorphismus, der Objektidentität, den Beziehungen zwischen Objekten und der Abstraktion.[38]

Das Objekt als zentraler Begriff der objektorientierten Modellierung bildet technisch gesehen eine Softwareeinheit die Objektdaten und Funktionen zusammenfasst. Objektdaten werden als Attribute und die Objektfunktionen als Methoden bezeichnet. Zustandsveränderungen und Kommunikation mit der Umwelt erfolgt für ein Objekt über die Methoden.[39] Dabei wird das Nachrichtenkonzept abgebildet, das die Kommunikation zwischen Objekten als Nachrichtenaustausch beschreibt. Das absendende Objekt ruft über eine Nachricht eine gewählte Methode im empfangenden Objekt auf. Inhalt einer solchen Nachricht sind der Name des Empfängerobjektes, der Name einer ausgewählten Methode und eine Parameterliste.[40] Im Grundkonzept der Klasse wird die Zusammenfassung von Objekten mit gleichen Methoden, Verhalten und Attributen beschrieben. Klassen dienen als Schablone für Objektinstanzen und legen somit deren Methoden und Attribute fest.[41] Für die Methoden und Attribute legt die Klasse eine Sichtbarkeit fest. Die Sichtbarkeit regelt die Zugriffsrechte auf die festgelegten Methoden und Attribute. Unterschieden wird zwischen privater, öffentlicher

---

[35] Vgl. Krallmann, Hermann, Schönherr, Marten, Trier, Matthias: Systemanalyse im Unternehmen, S. 183f.
[36] Vgl. Laudon, Kenneth, Laudon, Jane, Schoder, Detlef: Wirtschaftsinformatik Eine Einführung, S. 953f.
[37] Vgl. Laudon, Kenneth, Laudon, Jane, Schoder, Detlef: Wirtschaftsinformatik Eine Einführung, S. 956f.
[38] Vgl. Seidl, Martina, Brandsteidl, Marion, Huemer, Christian, Kappel, Gerti: UML @ Classroom, S. 7-9
[39] Vgl. Schader, Martin, Rundshagen, Michael: Objektorientierte Systemanalyse, S. 15f..
[40] Vgl. Schader, Martin, Rundshagen, Michael: Objektorientierte Systemanalyse, S. 17
[41] Vgl. Schader, Martin, Rundshagen, Michael: Objektorientierte Systemanalyse, S. 18f.

und geschützter Sichtbarkeit. Private Sichtbarkeit lässt nur den Zugriff vom selben Objekt und die öffentliche Sichtbarkeit den Zugriff von allen Objekten auf eine Methode oder Attribut zu. Die geschützte Sichtbarkeit unterscheidet sich von der öffentlichen darin das nur Objekte einer Klasse und deren Unterklassen ein Zugriffsrecht besitzen.[42] Unter Objektidentität versteht man das jedes Objekt, das als eine Instanz einer definierten Klasse existiert, über eine eindeutige Identität im Objektsystem identifiziert werden kann.[43]

Von Vererbung spricht man wenn zwischen zwei Klassen eine hierarchische Beziehung besteht, in der die untergeordnete Klasse von der übergeordneten Basisklasse die Methoden und Attribute übernimmt. Die untergeordnete Klasse bildet durch zusätzliche Methoden und Attribute eine Spezialisierung im Verhältnis zu der Basisklasse.[44] Das Konzept der Mehrfachvererbung beschreibt das eine untergeordnete Klasse von mehr als einer Basisklasse Methoden und Attribute erbt. Dabei entsteht ein vernetztes Klassenmodell.[45] In der Literatur zur objektorientierten Modellierung werden Vererbung und Mehrfachvererbung als Konzept der Generalisierung bezeichnet.[46] Eine abstrakte Klasse liegt vor wenn von einer Basisklasse keine konkreten Objekte instanziiert werden können.[47] Schnittstellen (engl. Interface, Interface Class) sind spezielle Klassen die über eine Menge von Methoden ein für Modellelemente vorgegebenes Verhalten beschreiben. Klassen die das von einer Schnittstelle beschriebene Verhalten anfordern müssen die von der Schnittstelle vorgegebenen Methoden vollständig implementieren.[48]

Unter polymorphen Verhalten versteht man in der objektorientierten Modellierung ein empfangendes Objekt auf eine Nachricht und somit einem Methodenaufruf unterschiedlich reagieren kann.[49] Für Objekte unterscheidet man zwischen statischen und dynamischen polymorphen Verhalten. Statisches Verhalten liegt vor wenn gleichnamige Methoden mit unterschiedlichen Signaturen (Unterscheidung in der Parameterliste) und gleichartigen Verhalten in einer Klasse festgelegt wurden. Dynamisches

[42] Vgl. Balzert, Heide: Lehrbuch der Objektmodellierung Analyse und Entwurf mit der UML 2, S. 266f.
[43] Vgl. Schader, Martin, Rundshagen, Michael: Objektorientierte Systemanalyse, S. 23f.
[44] Vgl. Heinrich, Gert, Mairon, Klaus: Objektorientierte Systemanalyse, S. 24-26
[45] Vgl. Schader, Martin, Rundshagen, Michael: Objektorientierte Systemanalyse, S. 25
[46] Vgl. Balzert, Heide: Lehrbuch der Objektmodellierung, S. 300
[47] Vgl. Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 53
[48] Vgl. Oestereich, Bernd: Die UML-Kurzrefernez 2.3 für die Praxis, S. 66
[49] Vgl. Seidl, Martina, Brandsteidl, Marion, Huemer, Christian, Kappel, Gerti: UML @ Classroom, S. 9

Verhalten liegt vor wenn über das Verhalten eines Objektes auf eine empfangene Nachricht erst zur Laufzeit in einer Objektinstanz entschieden wird.[50]

Mit den Konzepten der Assoziation, Aggregation und Komposition wird in der objektorientierten Modellierungen beschrieben welche Objekte in Beziehungen stehen und wie diese Verbindungen realisiert werden. Im Modell wird dafür die Navigierbarkeit und Multiplizität für eine Assoziation zwischen Objekten angegeben. Die Navigierbarkeit beschreibt die Zugriffsrichtung zwischen Objekten die entweder unidirektional, bidirektional oder nicht spezifiziert sein kann. Über die Multiplizität wird die Wertigkeiten einer Assoziation festgelegt, die beschreibt ob ein Objekt zu einem oder zu einer Vielzahl von Objekten eine Beziehung hat.[51] In der objektorientierten Modellierung wird die Multiplizität in Kann-Assoziationen und Muss-Assoziationen unterschieden. Für die Kann-Assoziation wird eine 0 und für die Muss-Assoziation eine 1 als Untergrenze für die Multiplizität zwischen Objekten angegeben.[52] Assoziationen die zwischen Objekten der gleichen Klasse bestehen werden als reflexive Assoziation bezeichnet.[53] Über Assoziationsklassen werden Assoziationen beschrieben die zusätzlich zu der Navigierbarkeit und Multiplizität gegenüber anderen Objekten eigene Methoden sowie Attribute besitzen.[54]

Aggregation und Komposition sind Spezialfälle der Assoziation. Wenn man die Beziehung zwischen Objekten als „ist ein Teil von“ beschreiben kann bildet sich ein hierarchisches Klassen-Modell das als Aggregation bezeichnet wird. Die Komposition ist eine Form der Aggregation für die zusätzliche Bedingungen gelten. Assoziationen einer Komposition werden als ganze Einheit betrachtet so das Lösch- und Kopiervorgänge alle an einer Komposition beteiligten Objekte zugleich betreffen. Zugriffe auf Teile einer Komposition erfolgen immer über das Aggregatobjekt.[55]

[50] Vgl. Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 64f.
[51] Vgl. Balzert, Heide: Lehrbuch der Objektmodellierung, S. 284-286
[52] Vgl. Balzert, Helmut: Lehrbuch der Softwaretechnik Basiskonzepte und Requirements Engineering, S. 158f.
[53] Vgl. Balzert, Helmut: Lehrbuch der Softwaretechnik Basiskonzepte und Requirements Engineering, S. 161
[54] Vgl. Balzert, Helmut: Lehrbuch der Softwaretechnik Basiskonzepte und Requirements Engineering, S. 162
[55] Vgl. Balzert, Helmut: Lehrbuch der Softwaretechnik Basiskonzepte und Requirements Engineering, S. 171

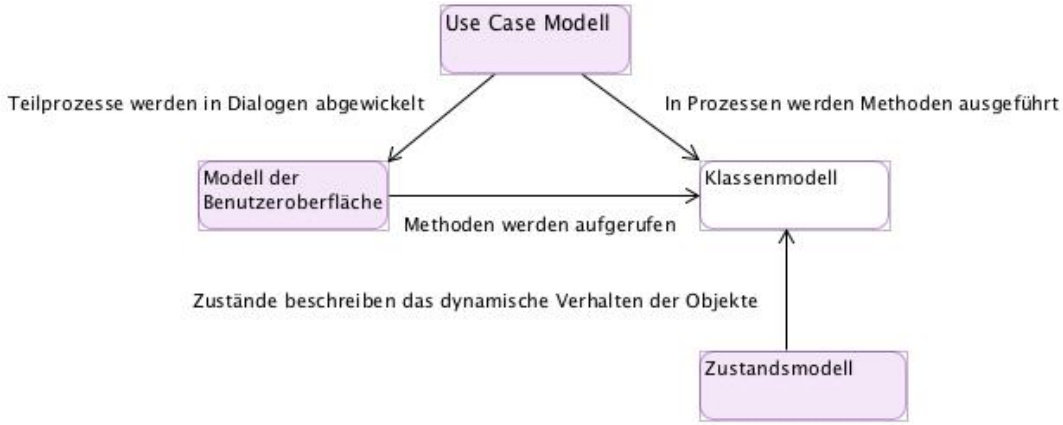

Abb. 2: objektorientierte Vorgehensweise

Quelle: In Anlehnung an Heinrich, Gert, Mairon, Klaus: Objektorientierte Systemanalyse, S. 5

In Abb. 2 wird ein objektorientiertes Vorgehensmodell zur Systementwicklung beschrieben das sich in statische und dynamische Modellbildungsphasen unterscheidet die in direkten Beziehungen stehen. Die dynamischen Modellierungsphasen sind in der Abb. 2 farblich hinterlegt. Die objektorientierte Vorgehensweise weist als ganzheitliche Herangehensweise durch die statischen und dynamischen Phasen eine durchgängige Modellierung, flexible Abstraktionsmöglichkeiten, Verbesserung der Wiederverwendbarkeit der Ergebnisse und Steigerung der Möglichkeiten für Wartung eines Systems auf. Zentrales Element dieser Vorgehensweise ist das bereits beschriebene statische Objektmodell (Klassenmodell) das strukturiert die Daten- und Funktionsmodelle zusammenführt.[56]

Mit der dynamischen Sicht auf ein System wird das interne Verhalten von Objekten und das Verhalten zwischen Objekten sowie die Veränderung ihrer Attributwerte im Verlauf der Zeit modelliert. Die unterschiedlichen Zustände die ein Objekt von der Erzeugung bis zur Zerstörung dynamisch einnimmt werden mit Zustandsdiagrammen modelliert. Im Zustandsmodell werden die Zustandsveränderungen eines Objektes als Reaktion auf Nachrichten die an das Objekt gesendet werden abgebildet.[57] Szenarien die eine Reihe von Ereignissen zwischen mehreren beteiligten Objekten abbilden, werden mit Ereignisfolgediagrammen modelliert. Die so dargestellte Objektinteraktion dient der Validierung der Ergebnisse aus statischer- und dynamischer Objektmodellierung.[58] Abgrenzend zu den Zustandsveränderungen wird im Use Case Modell eine Menge von Anwendungsfällen und Zusammenhänge zwischen den be-

[56] Vgl. Heinrich, Gert, Mairon, Klaus: Objektorientierte Systemanalyse, S. 4f
[57] Vgl. Schader, Martin, Rundshagen, Michael: Objektorientierte Systemanalyse, S. 109
[58] Vgl. Schader, Martin, Rundshagen, Michael: Objektorientierte Systemanalyse, S. 118

teiligten Modellelementen und Akteuren modelliert.[59] Wenn die Anwendungsfälle, Objektmodelle und Zustandsveränderungen festgelegt sind kann das Modell der Benutzerschnittstelle erstellt werden. Das Modell der Benutzerschnittstelle unterteilt sich in Dialog- sowie Eingabe/Ausgabe-Komponenten die für die Gestaltung von Informationsdarstellung und Abläufen verwendet werden. Die Dialogstruktur kann dabei systematisch von dem Klassenmodell abgeleitet werden. Dabei bilden Objekte einer Klasse die Erfassungsfenster und die Attribute einer Klasse die Interaktions- und Eingabeelemente ab.[60]

Für eine Vielzahl wiederkehrender Entwurfsprobleme gibt es in der objektorientierten Modellierung generische Lösungsansätze die sich in der Praxis bewährt haben. Diese Entwurfsmuster (engl. design pattern) werden über Name, Problembeschreibung, Lösungsbeschreibung und Konsequenzen grundlegend beschrieben. Dabei werden die Muster nach Erzeugungs-, Struktur- und Verhaltensmuster klassifiziert. Unter Erzeugungsmuster (engl. creational patterns) fasst man Muster zusammen die ein System unabhängig von der Erzeugung und Repräsentation seiner Objekte machen. Die Strukturmuster (engl. structural patterns) liefern Lösungsansätze für das Zusammensetzen komplexer Strukturen aus Klassen und Objekten. Bewährte Lösungsansätze für komplexe Kontrollflüsse sowie Interaktionen zwischen Objekten und Klassen werden in den Verhaltensmustern (engl. behavioral patterns) behandelt.[61]

2.5 Unified Modelling Language

Die Unified Modelling Language (kurz UML) ist eine von bestimmten Programmiersprachen und Zielplattformen losgelöste Modellierungssprache, die universelle Modellierungskonzepte für das Spezifizieren, Konstruieren, Visualisieren und Dokumentieren von Softwaresystemen liefert.[62] Über das Metamodell der UML wird eine Menge von Elementen sowie deren Verwendung festgelegt. Grundlegend wird in der UML eine Trennung zwischen Modellierungsmethode und Modellierungssprache

[59] Vgl. Oestereich, Bernd: Die UML-Kurzreferenz 2.3 für die Praxis, S. 22
[60] Vgl. Heinrich, Gert, Mairon, Klaus: Objektorientierte Systemanalyse, S. 98f
[61] Vgl. Balzert, Heide: Lehrbuch der Objektmodellierung, S. 356f.
[62] Vgl. Oestereich, Bernd: Analyse und Design mit der UML 2.5, S.239f.

erzielt.[63] Das Metamodell der UML schließt sich als objektorientierter Entwurf der objektorientieren Analyse an.[64]

Mit der UML wird eine grafische Modellierung in Form von Diagrammen umgesetzt, mit denen eine wesentliche Sicht auf die Struktur und Funktionsweises eines Systems erfolgt. Entsprechend werden die Diagramme in die Kategorien Struktur- und Verhaltensdiagramme eingeordnet. Das statische Verhalten eines Systems wird mit Strukturdiagrammen und ein dynamisches Verhalten mit Verhaltensdiagrammen, die mögliche Veränderungen an einem Objekt über den Zeitverlauf beschreiben, modelliert. Zu den Strukturdiagrammen zählen das Klassen-, Objekt-, Paket-, Kompendium-, Kompositionsstruktur-, Verteilungs- und Profildiagramm. In die Kategorie Verhaltensdiagramme werden Anwendungsfall-, Zustands-, Aktivitäts-, Sequenz-, Kommunikations-, Zeit- und Interaktionsübersichtsdiagramm eingeordnet.[65]

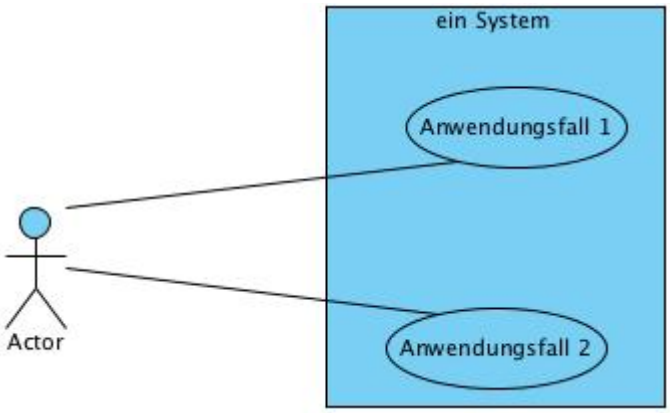

Abb. 3: Notationselemente des Anwendungsfalldiagramms

Quelle: In Anlehnung an Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 244

Das Anwendungsfalldiagramm besteht aus den wesentlichen Elementen des Akteurs, den Anwendungsfällen sowie deren Beziehung untereinander. Ein weiteres Notationselement kann die Darstellung eines Systems sein, das innerhalb seiner Grenzen die Fälle kapselt. Das Anwendungsfalldiagramm liefert keine Beschreibung zu konkreten Abläufen. Es werden vielmehr die Zusammenhänge zwischen Beteiligten Akteuren und den einzelnen Anwendungsfällen dargestellt. Die Abstrakte Sicht soll die Interaktion mit dem System vereinfacht präsentieren.[66]

[63] Vgl. Seidel, Martina, Brandsteidl, Marion, Huemer, Christian, Kappel, Gerti: UML @ Classroom, S. 14f.
[64] Vgl. Balzert, Heide: Lehrbuch der Objektmodellierung, S. 13
[65] Vgl. Seidel, Martina, Brandsteidl, Marion, Huemer, Christian, Kappel, Gerti: UML @ Classroom, S. 16-22
[66] Vgl. Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 243f.

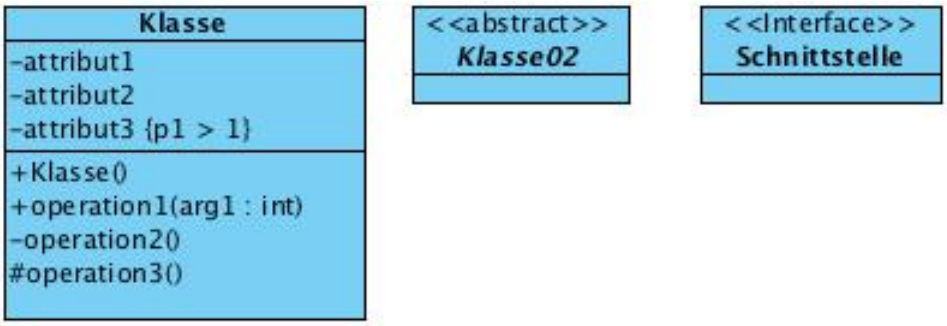

Abb. 4: Notationsmöglichkeiten für Klassen

Quelle: In Anlehnung an Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 273

Mit dem Klassenmodell werden die in einem System verfügbaren Klassen, deren Verhalten und Attribute sowie ihre Beziehungen untereinander abgebildet. Das Notationselement für eine Klasse ist ein aus drei Feldern bestehendes Rechteck. Im ersten Feld wird der Name einer Klasse notierte sowie das Merkmal für abstrakte Klassen und Schnittstellen. Das zweite Feld enthält die Attribute einer Klasse. Funktionen über die das Verhalten einer Klasse implementiert wird, sind im dritten Feld aufgelistet.[67]

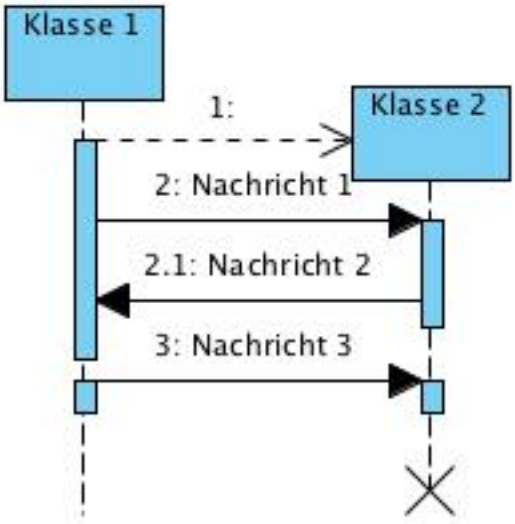

Abb. 5: Grundform des Sequenzdiagrammes

Quelle: In Anlehnung an Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 361

Der Austausch von Nachrichten zwischen einer ausgewählten Menge von Beteiligten in einer einzugrenzenden Situation wird als Sequenz betrachtet. Das Sequenzdiagramm bildet mit seinen Elementen den Nachrichtenaustausch zwischen den Beteiligten, die eine bestimmte Rolle einnehmen, über den Zeitverlauf ab. Der zeitliche Verlauf wird von oben nach unten dargestellt. Über gestrichelte senkrechte Linien werden die Rollen im Sequenzdiagramm dargestellt. Jede Rolle verfügt im Kopf be-

[67] Vgl. Oestereich, Bernd: Analyse und Design mit der UML 2.5, S 271f.

reich über einen rechteckigen Bereich, der den Namen der Rolle enthält. Ein Nachrichtenaustausch wird über Verbindungslinien zwischen den Rollen mit einer Pfeilspitze, die in die Senderichtung zeigt, in das Diagramm eingetragen. Die Unterscheidung zwischen Synchron- und Asynchroner-Nachricht wird über gefüllte und offene Pfeilspitzen dargestellt. Eine Konstruktionsnachricht geht direkt auf den Kopfbereich einer Rolle und zeigt die Erzeugung einer neuen Objektinstanz an. Ein Kreuz am Ende einer gestrichelten Linie zeigt die Destruktion einer Objektinstanz an.[68]

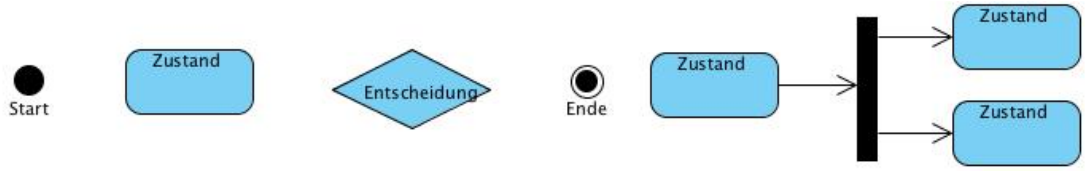

Abb. 6: Notation eines Zustandsdiagramm

Quelle: In Anlehnung an Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 350

Über Zustandsdiagramme werden innerer Zustandsmodelle eines Objektes modelliert. Das innere Verhalten und der Verlauf von Ereignissen werden über die Notationselemente als Zustandsautomat abgebildet. Die Veränderung von Attributen sowie die Zeitspanne zwischen zwei Ereignissen werden als Zustand definiert und im Zustandsdiagram als abgerundetes Rechteck dargestellt. Zwei besondere Zustände sind der Start- und End-Zustand die als gefüllter bzw. teilgefüllter Kreis dargestellt werden. Entscheidungen und Verzweigungen werden mit eigenen Symbolen als Raute und Balken abgebildet.[69]

2.6 Webservices

Charakteristisch für eine Softwarearchitektur sind die Architekturbausteine die durch ihre Beziehungen und Interaktion untereinander die Struktur eines Softwaresystems bilden. Über die veröffentlichten Eigenschaften eines Architekturbausteins werden externe Schnittstellen festgelegt. Softwarearchitekturen werden über differenzierte Sichtweisen unterschieden. Eine Verteilungssicht zeigt auf wo in einer Infrastruktur bestimmte Architekturbausteine implementiert werden. Mit der Kontextsicht wird die Interaktion eines Systems mit seiner Umgebung und den daran beteiligten Schnittstellen beschrieben.[70]

Der Begriff der Serviceorientierten Architektur ist aus Verteilungssicht ein System dessen Elemente verteilt in der Infrastruktur eigenständige Services bereitstellen. Als

[68] Vgl. Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 360-362
[69] Vgl. Oestereich, Bernd: Analyse und Design mit der UML 2.5, S. 350f.
[70] Vgl. Balzert, Helmut: Lehrbuch der Softwaretechnik Entwurf, Implementierung, Installation und Betrieb, S. 23

Service versteht man die Bereitstellung einer Leistung und Handlungsweisen die einem festgelegten Prozess entspricht. Charakteristisch für den Service in der Serviceorientierten Architektur ist eine unabhängige Bereitstellung von dem Nutzer.[71]

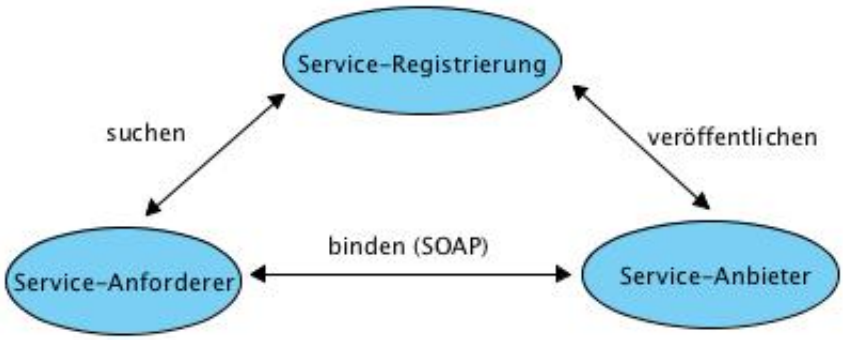

Abb. 7: Serviceorientierte Architektur

Quelle: In Anlehnung an Sommerville, Ian: Software Engineering, S. 563

In Abb. 7 ist das grundlegende Modell einer serviceorientierten Architektur dargestellt, das auf einer Wechselwirkung zwischen drei Elementen basiert. Über den Service-Anbieter wird eine Menge von Services bereitgestellt und über die Service-Registrierung veröffentlicht. Ein Service-Anforderer erhält über eine Service-Registrierung Informationen zur Schnittstellenbeschreibungen und Lokation eines nachgefragten Service. Mit den Informationen aus der Schnittstellenbeschreibung bindet sich der Service-Anforderer an den Service-Anbieter um den nachgefragten Service zu konsumieren.[72]

Webservices bilden eine Implementierung der serviceorientierten Architektur über die grundlegenden Basiskomponenten Kommunikation, Dienstbeschreibung und Verzeichnisdienst ab. Der technische Schwerpunkt liegt bei Webservices auf einer Maschine-Maschine-Kommunikation die über ein Netzwerk abgebildet wird. Die Kommunikation zwischen Server-Anbieter und Service-Nachfrager erfolgt über das XML-basierte Transportprotokoll Simple Object Access Protocol (kurz SOAP). Für die Beschreibung der Webservices über die Service-Registrierung wird das XML-basierende Dokumentenformat Web Service Description Language (kurz WSDL) verwendet.[73]

Mit der Extensible Markup Language (kurz XML) steht eine anwendungsspezifische Auszeichnungssprache für die Abbildung von strukturierten Daten zur Verfügung. Auszeichnungssprachen verfügen über Regeln zum deklarieren von Eigenschaften

[71] Vgl. Sommerville, Ian: Software Engineering, S. 562f
[72] Vgl. Sommerville, Ian: Software Engineering, S. 563f
[73] Vgl. Melzer, Ingo: Service-Orientierte Architekturen mit Web Services, S. 61-63

und der Bedeutung von Textelementen. Charakteristisch für die Auszeichnung von Textelementen sind die Start- und Endmarkierung mit denen die Dokumentinhalte strukturiert dargestellt werden. Als Wohlgeformt gelten XML-Dokumente wenn sie den syntaktisch Regeln der XML folgen. Folgt ein XML-Dokument einer Dokumenttypdefinition dann wird es als gültiges XML-Dokument bezeichnet. Eine Dokumenttypdefinition kann über eine Document Type Description (kurz DTD) oder über ein XML-Schema erfolgen. Die DTD kann innerhalb eines XML-Dokumentes oder als eigenständiges wohlgeformtes XML-Dokument festgelegt werden. Ein XML-Schema muss immer als separates XML-Dokument vorliegen. Zusätzlich zu den in einer DTD festgelegten Elementstruktur und Datentypen werden in einem XML-Schema gültige Wertbereiche für Elemente und Attribute definiert.[74]

Das Simple Object Access Protocol (kurz SOAP) ist ein auf XML basierendes Protokoll für die Integration und Interoperabilität zwischen unterschiedlichen Softwaresystemen. An einer SOAP-Kommunikation sind die Teilnehmer SOAP-Client und SOAP-Server beteiligt. Der SOAP-Client ist eine Anwendung die als Service-Nachfrager über SOAP-Requests Funktionsaufrufe an einen SOAP-Server sendet. Der SOAP-Server ist eine Anwendung die als Service-Anbieter auf Anfragen eines SOAP-Clients mit SOAP-Response Nachrichten reagiert. Ein SOAP-Server besteht aus den drei Komponenten Service-Manager, Deployment Service-List und XML-Translator. Über die Deployment Service-List werden alle Services gelistet die vom Service-Manager bereitgestellt werden. Mit dem XML-Translator verarbeitet der Service-Manger einen SOAP-Request um die Anfrage für die jeweilige konkrete Applikation, die den Service implementiert, bereitzustellen.[75] Nachrichten wie SOAP-Request und SOAP-Response sind XML-Dokumente die aus drei Teilen bestehen. Der SOAP-Envelope stellt das Wurzelelement für das XML-Dokument. Das Wurzelelement kapselt eine SOAP-Nachricht die sich aus SOAP-Header und SOA-Body zusammensetzt. Als optionales Element kann der SOAP-Header Steuerung- und Verarbeitungsinformationen für einen SOAP-Request enthalten, deren Inhaltlicher Aufbau in begleitenden Spezifikationen festgelegt wird. Der SOAP-Body ist für die SOAP-Nachricht zwingend vorgeschrieben und enthält die Nutzlast mit den In-

[74] Vgl. Hansen, Hans Robert, Neumann, Gustaf: Wirtschaftsinformatik 2 Informationstechnik, S. 469-471
[75] Vgl. Masak, Dieter: SOA? Serviceorientierung in Business und Software, S.239-241

formationen aus einem SOAP-Request oder SOAP-Response. Die Nutzlast besteht aus einem wohlgeformten XML-Dokument.[76]

Die Web Service Description Language (kurz WSDL) ist eine XML-Beschreibungssprache mit der standardisiert die von einem Service angebotenen Funktionen und deren Parameter definiert werden. Zusätzlich enthält ein WSDL-Dokument Informationen über die Lokation eines Service. Auf Grundlage eines WSDL-Dokumentes kann ein SOAP-Client dynamisch Klassen erzeugen die als Schnittstelle zum Aufruf des Webservice dienen. Durch die maschinelle Auswertung der WSDL-Dokumente wird eine automatische Integration realisiert.[77] Durch die konzeptionelle enge Bindung zwischen WSDL und Service-Implementierung, kann das WSDL-Dokument direkt aus dem Quellecode generiert werden.[78]

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <wsdl:definitions
3      xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
4      <wsdl:types>
5      </wsdl:types>
6      <wsdl:message name="">
7      </wsdl:message>
8      <wsdl:portType name="">
9      </wsdl:portType>
10     <wsdl:binding name="" type="">
11     </wsdl:binding>
12     <wsdl:service name="">
13     </wsdl:service>
14 </wsdl:definitions>
```

Abb. 8: Schematischer Aufbau einer WSDL-Datei

Quelle: In Anlehnung an Melzer, Ingo: Service-orientierte Architekturen mit Web Services, S.118

Ein Webservice wird mit einem WSDL-Dokument auf funktionaler Ebene abstrakt und auf technischer Ebene konkret beschrieben.[79] In Abb. 8 wird der schematische Aufbau einer WSDL-Datei dargestellt. Die Elemente <wsdl:binding/> und <wsdl:service/> liefern auf technischer Ebene konkrete Informationen wie und wo ein Service angesprochen wird. Über die Elemente <wsdl:types/>, <wsdl:messages/> und <wsdl:portType/> werden abstrakte Informationen zu Datentypen, komplexe Nachrichtenformate sowie die grundlegende Funktionalität für einen Webservice beschrieben.[80] Mit den XML-Elementen <wsdl:include/> und <wsdl:import/> unter-

[76] Vgl. Melzer, Ingo: Service-Orientierte Architekturen mit Web Services, S.87-90
[77] Vgl. Balzert, Helmut: Lehrbuch der Softwaretechnik Entwurf, Implementierung, Installation und Betrieb, S. 265
[78] Vgl. Masak, Dieter: SOA? Serviceorientierung in Business und Software, S. 241
[79] Vgl. Melzer, Ingo: Service-Orientierte Architekturen mit Web Services, S. 116
[80] Vgl. Melzer, Ingo: Service-Orientierte Architekturen mit Web Services, S. 117-122

stützt die WSDL die Modularisierung von WSDL-Dokumenten und die Wiederverwendung von strukturierten Inhalten.[81]

Mit PHP steht für die Entwicklung von dynamischen Web-Anwendungen eine Programmiersprache bereit die direkt in eine Auszeichnungssprache integriert wird. Programminstruktionen werden durch spezifische Start- und Endmarkierungen gekennzeichnet. Als wesentliches Unterscheidungsmerkmal gegenüber weiteren Programmiersprachen benötigt PHP eine serverseitige Laufzeitumgebung die aus PHP-Interpreter und Webserver besteht.[82] Der Sprachumfang von PHP unterstützt die objektorientierte Programmierung vollständig. Dabei orientiert sich der Sprachumfang an der objektorientierten Modellierung mit der UML.[83] Schwerpunkte der Programmierspprache PHP liegen in der Anbindung von Datenbanken, der komplexe Datenaustausch mit Services über definierte Schnittstellen sowie die Verarbeitung von XML-Dokumenten.[84] Zu den definierten Schnittstellen gehört das Web-Application Programming Interface (kurz Web-API) mit der in PHP Service-Anbieter und Service-Nachfrager für Webservices über das Protokoll SOAP implementiert sind.[85] Über die Reflection-API liefert PHP eine Schnittstelle zur eigenen Objekt- und Klassenumgebung über die zur Laufzeit Informationen zur Klassen-Struktur, den enthaltenen Funktionen sowie Attributen und der Code-Dokumentation bereitgestellt werden.[86]

3 Ablaufplanung in der Produktion

3.1 Einführung in die Ablaufplanung

Mit der Produktionsprogrammplanung erfolgt eine mittelfristige operative Planung der Produktion, die sich an einer prognostizierten Nachfrage der Fertigerzeugnisse orientiert.[87] Die daraus abgeleiteten Produktionsprogramme werden in der kurzfristigen operativen Planung durch die Produktionssteuerung in konkrete Fertigungsabläu-

[81] Vgl. Melzer, Ingo: Service-Orientierte Architekturen mit Web Services, S. 124f.
[82] Vgl. Reimers, Stefan, Thies, Gunnar: PHP 5.4 und MySQL 5.5 Das umfassende Handbuch, S.25-30
[83] Vgl. Reimers, Stefan, Thies, Gunnar: PHP 5.4 und MySQL 5.5 Das umfassende Handbuch, S. 147-155
[84] Vgl. Achour, M., Betz, F., Dovgal, A., Lopes, N., Magnusson, H., Richter, G., et al., Was kann PHP, http://www.php.net/manual/de/intro-whatcando.php, 12.04.2012
[85] Vgl. Reimers, Stefan, Thies, Gunnar: PHP 5.4 und MySQL 5.5 Das umfassende Handbuch, S. 780f.
[86] Vgl. Achour, M., Betz, F., Dovgal, A., Lopes, N., Magnusson, H., Richter, G., et al., Einführung Reflections, http://de2.php.net/manual/de/intro.reflection.php, 12.04.2012
[87] Vgl. Schneeweiß, Christoph: Einführung in die Produktionswirtschaft, S. 23

fe geplant.[88] In der Ablaufplanung wird die optimale Durchführung und Reihenfolge der Fertigungsaufträge aus einem festgelegten Produktionsprogramm unter Berücksichtigung von Zeit- und Kostenzielen bestimmt.[89]

Die Ablaufplanung lässt sich durch eine systematische Betrachtung der unterschiedlichen Vorgehensweisen in drei Planungsbereiche unterteilen. Der Planungsbereich der Projektplanung wird bestimmt durch eine komplexe und auftragsorientierte Einzelfertigung. Ein Schwerpunkt der Projektplanung ist die strukturierte Analyse der Einzelschritte sowie deren Terminplanung. Ein Lösungsansatz für die Projektplanung ist die Netzplantechnik. Die Fließbandplanung ist der Planungsbereich in dem ganze Produkte oder Bestandteile von Produkten standardisiert in einer Massenfertigung unter Zeitzwang produziert werden. Wesentlich für die Fließbandfertigung ist die Aufteilung der Produktfertigung in einzelne Elemente mit denen das Fließbandlayout umgesetzt wird. Für die Fließbandfertigung wird eine Optimierung von Taktzeiten und Anzahl der Arbeitsstationen in der Ablaufplanung durchgeführt. Eine weitere Zielsetzung liegt in der Optimierung von Wartezeiten zwischen den einzelnen Arbeitsstationen.[90] Typisch für die Projekt- und Fließbandplanung ist die einmalige Planungsphase vor Beginn der Fertigung. Dem gegenüber steht die kurzfristige Planung im Bereich der Maschinenbelegungsplanung. Die Aufgabe in diesem Planungsbereich besteht in der Festlegung einer optimalen Reihenfolge und Zeitpunkte für die Bearbeitung von Fertigungsaufträgen auf einzelnen Maschinen. Für eine optimale Ablaufplanung werden in diesem Bereich Methoden aus dem Operations Research eingesetzt.[91] Für die Planung der Auftragsfolge unterscheidet man die Maschinenbelegungsplanung in die Problemtypen Flow Shop und Job Shop. Der Problemtyp Flow Shop beschreibt einen Fertigungsbereich mit festgelegter Maschinenfolge. Dem gegenüber steht der Problemtyp Job Shop der unterschiedliche Maschinenfolgen für die Fertigungsaufträge zulässt.[92]

Über die grundlegende Sichtweise der Zuordnung von Aufträgen zu Maschinen lassen sich für eine Optimierung der Ablaufplanung Bestands-, Termin- und Auslastungsorientierte Zielsetzungen festlegen.[93] Für eine optische Darstellung der optimierten Ergebnisse einer Ablaufplanung werden das Auftrags- und Maschinenfolge-

[88] Vgl. Schneeweiß, Christoph: Einführung in die Produktionswirtschaft, S. 259
[89] Vgl. Plümer, Thomas: Logistik und Produktion, S. 215
[90] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 281f.
[91] Vgl. Plümer, Thomas: Logistik und Produktion, S. 215f.
[92] Vgl. Schneeweiß, Christoph: Einführung in die Produktionswirtschaft, S. 262f.
[93] Vgl. Thonemann, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 366-368

diagramm verwendet. Beide Diagramme liefern unter Einbeziehung des zeitlichen Verlaufs eine graphische Darstellung der Ablaufplanung.[94]

3.2 Ablaufplanung mit einer Maschine

3.2.1 Problembeschreibung

Neben den bereits beschriebenen Zielen für eine Optimierung der Ablaufplanung existiert unter Fertigungsbedingungen mit nur einer Maschine die Zielsetzung Umrüstzeiten zu minimieren. Als Rüstzeit wird der Zeitbedarf für die Umrüstung einer Maschine zwischen unterschiedlichen Produktarten ermittelt. Eine Minimierung der Rüstzeiten führt zur Senkung der Rüstkosten und Steigerung der Bearbeitungszeitkapazität.[95] Die Struktur dieser Problemstellung ist vergleichbar mit dem „Traveling Salesman Problem“ aus dem Operations Research, dessen Zielsetzung in der Optimierung eines Rundreiseplans mit Bezug auf Kosten und Zeit besteht. Jeder Ort eines Rundreiseplans muss bei der Optimierung genau einmal besucht werden. Den Abschluss des Reisplans bildet die Rückkehr zum Startpunkt. Lösungen für das Rundreiseproblem lassen sich auf die Ablaufplanung mit einer Maschine anwenden, da der Übergang zwischen zwei Orten vergleichbar mit dem Umrüsten von zwei Aufträgen auf einer Maschine ist. Eine Minimierung der Rundreisezeit entspricht dann der Minimierung von Rüstzeiten.[96]

Die Kosten für einen Umrüstvorgang von Produktart i auf Produktart j wird über die Rüstkostenmatrix d_{ij} dargestellt. Über die binäre Reihenfolgematrix u_{ij} werden die möglichen Reihenfolgen dargestellt. Für den Fall das von Produktart i auf Produktart j gewechselt werden kann wird u_{ij} gleich 1 und sonst gleich 0 gesetzt. Somit wird folgende Zielfunktion formuliert:[97]

$$\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij} * u_{ij} \Rightarrow Min!$$

Bei der Auftragsfolge muss auf eine Produktart immer eine andere Produktart folgen. Daraus ergeben sich die folgenden Nebenbedingungen:[98]

---

[94] Vgl. Schiemenz, Bernd, Schönert, Olaf: Entscheidung und Produktion, S. 191f.
[95] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 286
[96] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 286
[97] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 286f.
[98] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 287

$$\sum_{i}^{n} u_{ij} = 1$$

$$\sum_{j}^{n} u_{ij} = 1$$

In der Auftragsfolge müssen alle Produktarten berücksichtig werden ohne das Teilfolgen mit eigenen geschlossenen Zyklus entstehen. Daraus ergeben sich zusätzlich unter Einführung des Produktartenindex h die folgenden Nebenbedingungen:[99]

$$u_{ij} + u_{ji} \leq 1 \; f\ddot{u}r \; i,j = 1,\dots,n$$

$$u_{hi} + u_{ij} + u_{jh} \leq 2 \; f\ddot{u}r \; i,j = 1,\dots,n$$

3.2.2 Lösungsansatz mit Branch & Bound

Branch and Bound Verfahren ermöglichen die Suche von optimalen Lösungen für ein Optimierungsproblem durch das systematische Aufspalten eines zulässigen Bereichs in zu bewertende Teillösungen. Auf die hieraus entstehenden Teillösungen wird jeweils weiter die Optimierung durch systematische Aufspaltung angewendet, sofern die Teillösung nicht schlechter bewertet wird als die beste bekannte Lösung aus einem weiteren Teilbereich. Eine graphische Modellierung kann über einen Entscheidungsbaum erfolgen, der die Aufspaltung und Bewertung von Teillösungen darstellt.[100] Ein wesentliches Merkmal von Entscheidungsbäumen ist die logische und strukturierte Darstellung von alternativen Lösungen sowie deren Bewertung. Entscheidungsbäume können für die mathematische Modellierung als Matrix dargestellt werden.[101]

Unter Berücksichtigung als Zielwert die minimale Gesamtbearbeitungszeit zu ermitteln wird das Branch and Bound für einen Flow Shop mit einer Maschine in die Phasen Initialisierung, Terminierung, Branching und Bounding unterteilt. In der Initialisierungsphase wird eine gültige Ausgangslösung bestimmt sowie deren Zielwert ermittelt. Unter Terminierung werden die Abbruchbedingungen für den Branch and Bound Algorithmus überprüft. Sofern alle Unterprobleme gelöst oder von der weiteren Aufspaltung ausgeschlossen sind wird der Algorithmus beendet. Über das Branching wird das Problem in Unterprobleme oder ein Unterproblem in weitere Unterprobleme aufgespalten. Entscheidend für das Branching ist die Auswahl einer Verzweigung mit der geringsten Steigerung für den Zielwert um eine gute Lösung zu

[99] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 287f.
[100] Vgl. Ellinger, Theodor, Beuermann, Günther, Leisten, Rainer: Operations Research: S. 169f
[101] Vgl. Eisenführ, Franz, Weber, Martin: Rationales Entscheiden, S. 38f

finden. Um zu entscheiden wie ein Unterproblem weiter behandelt wird, muss in der Phase des Boundings für das betrachtete Unterproblem der Zielfunktionswert berechnet werden. Ist der berechnete Wert größer oder gleich dem Zielfunktionswert der bisher besten Lösung, so liegt eine Lösung vor die nicht weiter betrachtet werden muss. Liegt der Wert unter dem Zielfunktionswert der besten Lösung, so wird die beste Lösung durch die aktuelle betrachtete Lösung ersetzt. Der Algorithmus springt jetzt zurück in die Terminierungsphase und führt, sofern kein Abbruch erfolgt, ein weiteres Branching auf dem betrachteten Unterproblem durch.[102]

Für eine Umsetzung als Rechenlösung wird eine Matrix mit den Elementen d_{ij} benötigt, mit der die Rüstkosten bzw. Rüstzeiten für das Umrüsten von Auftrag i auf Auftrag j erfasst werden. Für Elemente die eine nicht zulässige Kombination d_{ij} darstellen erfolgt die Bewertung mit ∞ (Wert unendlich).[103]

	A_1	A_2	A_3	A_4	A_5	$\sum$
A_1	∞	3	∞	7	∞	
A_2	3	∞	5	10	8	
A_3	∞	2	∞	14	9	
A_4	7	∞	∞	∞	4	
A_5	6	8	6	4	∞	
SM	3	2	5	4	4	**18**

;

	A_1	A_2	A_3	A_4	A_5	ZM
A_1	∞	1	∞	3	∞	1
A_2	0	∞	0	6	4	0
A_3	∞	0	∞	10	5	0
A_4	4	∞	∞	∞	0	0
A_5	3	6	1	0	∞	0
$\sum$						**1**

Für die vorliegende Beispiel-Matrix mit fünf Aufträgen wird eine Zeilen- und Spaltenreduktion durchgeführt. Eine Reduktion der Matrix wird über die Ermittlung der minimalen Werte je Spalte (linke Matrix) und je Zeile (rechte Matrix) eingeleitet. Die Summe aller minimalen Werte liefert die Reduktionskonstante mit einem Wert von 19 Zeiteinheiten (18 + 1). Mit der Reduktionskonstante wird die theoretische Kostenuntergrenze für die Umrüstvorgänge festgelegt die von der zu ermittelnden optimalen Lösung nicht unterschritten werden kann. Dies entspricht dem oben beschriebenen Bounding. Die Minimalwerte je Spalte und Zeile werden in den entsprechenden Spalten und Zeilen von den Elementwerten abgezogen. Somit entsteht in jeder Zeile und Spalte mindestens ein 0-Element.[104]

[102] Vgl. Thonemann, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 386f

[103] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 288

[104] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 288f

	A_1	A_2	A_3	A_4	$\boldsymbol{A_5}$
A_1	∞	0	∞	2	∞
A_2	0	∞	0	6	4
A_3	∞	0	∞	10	5
$\boldsymbol{A_4}$	4	∞	∞	∞	**0**
A_5	3	6	1	0	∞
					$\Uparrow$

$\Leftarrow$;

	A_1	A_2	A_3	A_4	ZM
A_1	∞	0	∞	2	0
A_2	0	∞	0	6	0
A_3	∞	0	∞	10	0
A_5	3	6	1	∞	1
SM	0	0	0	2	

Die Auswahl einer Teilmenge über das Branching wird durch die Bewertung der 0-Elemente der reduzierten Kostenmatrix vorgenommen. Alle 0-Elemente werden mit der Summe aus dem zugehörigen Spalten- und Zeilenminimum bewertet. Das Element mit der höchsten Bewertung wird in die Lösung aufgenommen. In der obigen Matrix wird das Element d_{45} ausgewählt. Somit wird in die optimale Lösung eine Umrüstung von Auftrag 4 zu Auftrag 5 aufgenommen. Durch ein Löschen von Zeile 4 und Spalte 5 wird eine neue Matrix erstellt. Um einen Zyklus in der Lösung zu verhindern wird das Element d_{54} auf den Wert ∞ gesetzt. Da nicht alle Zeilen und Spalten ein 0-Element aufweisen erfolgt eine erneute Iteration für die Zeilen- und Spaltenreduktion, die zu dem neuen Wert von 22 für die Reduktionskonstante führt.[105]

	A_1	$\boldsymbol{A_2}$	A_3	A_4
A_1	∞	0	∞	0
A_2	0	∞	0	4
$\boldsymbol{A_3}$	∞	**0**	∞	8
A_5	2	5	0	∞
		$\Uparrow$		

$\Leftarrow$;

	A_1	A_3	$\boldsymbol{A_4}$
$\boldsymbol{A_1}$	∞	∞	**0**
A_2	0	∞	4
A_5	2	0	∞
			$\Uparrow$

$\Leftarrow$;

	A_1	A_3
A_2	0	∞
A_5	2	0

Für die folgenden Iterationen ist immer ein 0-Element je Zeile und Spalte vorhanden. Somit erfolgt für die jeweilige Matrix eine Bewertung der 0-Elemente. Über die Auswahl der höchsten Bewertung werden die sukzessiv die Elemente d_{32} , d_{14} und d_{21}. Damit ergibt sich eine optimale Lösung für die Auftragsfolge 4-5-3-2-1-4 mit den minimalen Rüstkosten von 22.[106]

[105] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 290f

[106] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 291f

3.3 Ablaufplanung mit zwei und drei Maschinen

3.3.1 Problembeschreibung

Die Fließfertigung mit mehreren Maschinen wird in zwei Varianten unterschieden. In der getakteten Fließfertigung (engl. Assembly Line) laufen alle Aufträge synchronisiert über die Bearbeitungsstationen. Die Fertigungsaufträge werden im gleichen Takt auf den Maschinen gestartet. Dem gegenüber steht die zweite Variante der ungetakteten Fließfertigung (engl. Flow Shop), bei der ein Fertigungsauftrag nach Beendigung auf einer Maschine auf die nächste Maschine wechselt sobald die Folgemaschine für die Bearbeitung frei wird. Eine Flow Shop Fertigung mit beliebiger Anzahl von Maschinen weist eine hohe Komplexität auf. Im folgenden werden die Spezialfälle für einen Flow Shop mit zwei und drei Maschinen, mit der Zielsetzung die Gesamtbearbeitungszeit zu minimieren, dargestellt.[107] Der Begriff Gesamtbearbeitungszeit wird in der Literatur äquivalent zum Begriff der Zykluszeit verwendet. Unter der Zykluszeit wird die Zeit ermittelt, die für die Bearbeitung der gesamten Fertigungsaufträge in einem Produktionsbereich benötigt wird.[108]

$$T = \begin{matrix} \cdot & M_1 & M_2 \\ A_1 & 4 & 8 \\ A_2 & 2 & 3 \\ A_3 & 6 & 1 \end{matrix} = \begin{pmatrix} 4 & 8 \\ 2 & 3 \\ 6 & 1 \end{pmatrix}$$

$$T_{kum} = \begin{pmatrix} 4 & 4+8=12 \\ 4+2=6 & 6+3=9 \\ 6+6=12 & 9+12=18 \end{pmatrix} = \begin{pmatrix} 4 & 12 \\ 6 & 9 \\ 12 & \mathbf{18} \end{pmatrix}$$

Für eine Bearbeitungszeitmatrix T mit den Aufträgen A_i und den Maschinen M_j wird die Zykluszeit über die kumulierte Bearbeitungszeitmatrix T_{kum} ermittelt. Die Zykluszeit wird in der Matrix T_{kum} in der Zelle der letzten Spalte und letzten Zeile dargestellt.[109]

3.3.2 Lösungsansatz nach Johnson

Unter dem Begriff Permutationslösung versteht man alle Lösungen bei denen Fertigungsaufträge auf allen Maschinen in der gleichen Reihenfolge bearbeitet werden. Für einen Flow Shop mit zwei Maschinen ist ein optimaler Ablaufplan immer eine Permutationslösung. Zum bestimmen einer optimalen Lösung aus „I!“ möglichen Permutationen wird der Johnson-Algorithmus verwendet. Die Funktionsweise des

[107] Vgl. Thoneman, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 378f.
[108] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 284
[109] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 291f

Johnson-Algorithmus lässt sich in die drei Phasen Initialisierung Algorithmus, Sortierung der Aufträge und Verkettung der Reihenfolge unterteilen.[110]

$$\begin{array}{cc} \cdot & M_1 \quad M_2 \\ A_1 & 1 \quad 4 \\ A_2 & 4 \quad 1 \\ A_3 & 9 \quad 7 \\ A_4 & 3 \quad 8 \\ A_5 & 10 \quad 4 \end{array}; T = \begin{pmatrix} 1 & 4 \\ 4 & 1 \\ 9 & 7 \\ 3 & 8 \\ 10 & 4 \end{pmatrix}; T_{kum} = \begin{pmatrix} 1 & 5 \\ 5 & 6 \\ 14 & 21 \\ 17 & 29 \\ 27 & \mathbf{33} \end{pmatrix}; J_1 = \{\quad\}; J_2 = \{\quad\}$$

In der Phase der Initialisierung wird aus einem Produktionsprogramm mit mehreren Aufträgen eine Bearbeitungszeitmatrix T sowie die kumulierte Bearbeitungszeitmatrix T_{kum} erstellt. Aus der Matrix T_{kum} wird im obigen Beispiel die Zykluszeit mit einem Wert von 33 Zeiteinheiten abgelesen. Zusätzlich werden die zwei leeren Listen J_1 und J_2 definiert und anschließend durch eine Iteration über alle Aufträge gefüllt.[111]

$$J_1 = \{A_1|A_4\}; J_2 = \{A_2|A_3|A_5\}$$

Aufträge deren Belegungszeit auf Maschine M_1 kleiner ist als auf Maschine M_2 werden der Liste J_1 zugeordnet. Aufträge deren Belegungszeit auf Maschine M_1 größer ist als auf Maschine M_2 werden der Liste J_2 zugeordnet.[112]

$$J_1 = \{A_1|A_4\}; J_2 = \{A_3|A_5|A_2\}$$

In der Phase der Sortierung werden die Aufträge mit ihren Belegungszeiten in der Liste J_1 aufsteigend und in der Liste J_2 absteigend sortiert.[113]

$$R = \{A_1|A_4|A_3|A_5|A_2\}$$

Die letzte Phase im Johnson-Algorithmus ist die Verkettung der beiden Listen zu der optimalen Lösung R.[114]

$$T^* = \begin{pmatrix} 1 & 4 \\ 3 & 8 \\ 9 & 7 \\ 10 & 4 \\ 4 & 1 \end{pmatrix}; T^*_{kum} = \begin{pmatrix} 1 & 5 \\ 4 & 13 \\ 13 & 20 \\ 23 & 27 \\ 27 & \mathbf{28} \end{pmatrix}$$

Aus der Lösung R wird die optimierte Bearbeitungszeitmatrix T^* sowie die kumulierte Bearbeitungszeitmatrix T^*_{kum} erstellt. Im Vergleich zu Ausgangslösung wurde die Zykluszeit auf den Wert von 28 Zeiteinheiten verbessert.[115]

[110] Vgl. Thoneman, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 379
[111] Vgl. Thoneman, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 380
[112] Vgl. Thoneman, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 380
[113] Vgl. Thoneman, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 380f.
[114] Vgl. Thoneman, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 380f.
[115] Vgl. Thoneman, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 380f.

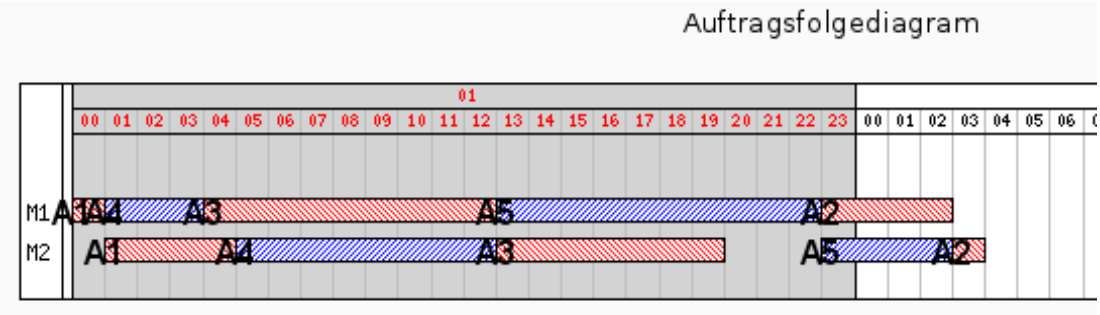

Abb. 9: Auftragsfolgediagramm

Quelle: In Anlehnung an Thonemann, Ulrich: Operations Management, S. 381

Für den Sonderfall mit drei Maschinen kann das Johnson-Verfahren angewendet werden, wenn sich das Problem auf ein Zwei-Maschinen-Problem reduzieren lässt. Hierfür ist eine der beiden Bedingungen zu erfüllen:[116]

$$t_{2,max} \leq t_{1,min} \ \ oder \ t_{2,max} \leq t_{3,min}$$

Sofern die maximale Bearbeitungszeit aller Aufträge für Maschine 2 kleiner oder gleich der kleinsten Bearbeitungszeit aller Aufträge auf Maschine 1 oder Maschine 3 ist, wird mit dem Johnson-Algorithmus die optimale Lösung ermittelt. Im nächsten Schritt wird aus der Bearbeitungszeitmatrix mit drei Maschinen in eine Bearbeitungsmatrix mit zwei Maschinen überführt.[117]

$$\begin{array}{cccc} \cdot & M_1 & M_2 & M_3 \\ A_1 & 6 & 2 & 3 \\ A_2 & 10 & 4 & 5 \\ A_3 & 8 & 3 & 2 \\ A_4 & 7 & 6 & 4 \end{array} ; T = \begin{pmatrix} 6 & 2 & 3 \\ 10 & 4 & 5 \\ 8 & 3 & 2 \\ 7 & 6 & 4 \end{pmatrix} \Rightarrow T^* = \begin{pmatrix} 6+2=8 & 2+3=5 \\ 10+4=14 & 4+5=9 \\ 8+3=11 & 3+2=5 \\ 7+6=13 & 6+4=10 \end{pmatrix}$$

Die Bearbeitungszeitmatrix T^* wird aus der Bearbeitungsmatrix T durch das Addieren der Werte aus Spalte 1 und 2 sowie der Werte aus Spalte 2 und 3 erstellt. Für die Bestimmung der optimalen Lösung wird auf die Matrix T^* der oben beschriebene Johnson-Algorithmus für zwei Maschinen angewandt.[118]

3.4 Ablaufplanung mit mehreren Maschinen

3.4.1 Problemstellung

Für die Flow Shop Fertigung mit beliebiger Anzahl an Maschinen gilt nicht notwendigerweise die Minimierung der Zykluszeit als optimale Lösung. Abhängig von der

[116] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 300
[117] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 300
[118] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 299f.

Zielsetzung kann es vorteilhaft sein die Aufträge auf einer bestimmten Maschine in einer anderen Reihenfolge zu fertigen als auf einer anderen Maschine. Durch die hieraus resultierende Komplexität entsteht eine Vielzahl von Möglichkeiten die sich über $(m!)^a$ bestimmen lassen. Dabei wird m gleich der Anzahl der Maschinen und a gleich Anzahl der Aufträge gesetzt.[119] Als Beispiel ergeben sich bei 4 Maschinen und 3 Aufträgen 1296 mögliche Lösungen. Mit steigender Komplexität ist eine schnelle und exakte Ermittlung einer optimalen Lösung nicht möglich.[120]

3.4.2 Lösungsansätze

Da mit steigernder Auftrags- und Maschinenanzahl die Komplexität und somit die Rechenzeit für die praktische Lösungsfindung steigt, wurden verschiedene Optimierungsverfahren aus dem Operations Research für die Problemstellung des mehrstufigen Ablaufplanungsproblems herangezogen. Zu betrachten sind die Lösungsverfahren mit Prioritätsregeln, Expertensysteme, Branch & Bound Verfahren und Evolutionäre Algorithmen.[121]

Über Prioritätsregeln lassen sich gute oder sogar optimale Lösungen für Problemstellungen ermitteln, die mit analytischen Methoden aufgrund des zu hohen Rechenaufwands nicht zu lösen wären. Mittels Prioritätsregeln werden wartenden Aufträgen vor der Bearbeitung auf einer Maschine Prioritäten zugeordnet mit denen die Rangfolge in der Ablaufplanung bestimmt wird. Für Prioritätsregeln lassen sich zwei wesentliche Gliederungsmöglichkeiten festlegen. Eine Gliederung erfolgt in statische und dynamische sowie in lokale und globale Prioritätsregeln. Die First-Come First-Serve Regel ist eine statische, lokale Regel die besagt das Aufträge in der Reihenfolge bearbeitet werden in der sie vor einer Maschine eintreffen. Ebenfalls zu den statischen, lokalen Regeln zählt die kürzeste Operationszeit Regel, die besagt das der Auftrag mit der geringsten Bearbeitungszeit auf einer Maschine zuerst bearbeitet wird. Als dynamisch, global wird die Critical Ratio Regel eingeordnet, die besagt das für einen Auftrag die noch notwendige Restbearbeitungszeit ins Verhältnis zum vorgegebenen Liefertermin gesetzt wird. Mit der kürzesten Schlupfzeit Regel wird die Position in der Warteschlange für einen Auftrag über die Differenz zwischen Liefertermin und der noch ausstehenden Bearbeitungszeit bestimmt.[122]

---

[119] Vgl. Thoneman, Ulrich: Operations Management Konzepte, Methoden und Anwendungen, S. 382
[120] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 297
[121] Vgl. Plümer, Thomas: Logistik und Produktion, S. 216f.
[122] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 306f.

Aus dem Forschungsbereich der Künstlichen Intelligenz (kurz KI) stammen Expertensysteme, die über Expertenwissen in Form von Regelbasierten Wenn-Dann-Entscheidungen und Fallbeschreibungen Lösungsvorschläge für komplexe Aufgabenstellungen liefern.[123] Zwei Einsatzbereiche für Expertensysteme sind die dynamische Auswahl geeigneter Prioritätsregeln und die Neudisposition in der Fertigungssteuerung.[124]

Das Lomnicki-Verfahren stellt für die Ablaufplanung mit mehreren Maschinen einen Branch and Bound Algorithmus da, der sich vom Modell mit einer Maschine in der Bounding-Technik unterscheidet.[125]

Evolutionäre Algorithmen basieren auf Erkenntnissen aus der Vererbungslehre. Ausgehend von Menge von Ausgangslösungen, die unter Zufall erzeugt werden, erfolgt der Ablauf in drei Schritten. Die Ausgangslösung wird auch als Anfangspopulation bezeichnet. Im ersten Schritt wird die Güte der vorhandenen Lösungen festgestellt. Gute und schlechte Lösungen werden durch Selektion getrennt. Der dritte Schritt erzeugt mit Hilfe von genetischen Verfahren wie Crossing over, Mutation oder Reproduktion neue Lösungsmengen. Die Schritte eins bis drei werden in einer Endlosschleife ausgeführt. Als Abbruchkriterium werden eine Anzahl von Iterationen, abgelaufene Rechenzeit oder eine erreichte Lösungsgüte eingesetzt. Als Endergebnis wird die beste Lösung aus dem Verlauf des Algorithmus bestimmt.[126]

4 Praxisbeispiel

4.1 Systemanalyse

4.1.1 Problem-Analyse

Für die Ablaufplanung in der Produktion soll ein Softwaresystem die Optimierung von Produktionsprogrammplänen durchführen. Das System ist dabei in Eingabe, Verarbeitung und Ausgabe unabhängig und wird über eine standardisierte Kommunikationsschnittstelle mit weiteren Systemen integriert. Verarbeitete Ablaufpläne werden für eine zeitversetzte Verarbeitung zwischen gespeichert. Die Integration in die Systemlandschaft ist flexibel und unabhängig von Programmiersprachen oder

[123] Vgl. Kurbel, Karl: Entwicklung und Einsatz von Expertensystemen, S. 13f.
[124] Vgl. Kurbel, Karl: Entwicklung und Einsatz von Expertensystemen, S. 155
[125] Vgl. Bloech, Jürgen, Bogaschewsky, Ronald, Buscher, Udo, Daub, Anke, Götze, Uwe, Roland, Folker: Einführung in die Produktion, S. 301
[126] Vgl. Boersch, Ingo, Heinsohn, Jochen, Socher, Rolf: Wissensverarbeitung, S. 69-71

Betriebsplattformen sowie Betriebssystemen. Das Optimieren einer Ablaufplanung kann jeder Zeit angepasst und um neue Verfahren transparent erweitert werden.

Aus der oben aufgeführten Vision lassen sich die folgenden Zielsetzungen ableiten. Das Softwaresystem soll als Service-orientierte Architektur implementiert werden. Als Service-Anbieter muss die anstrebte Integration über die Systemgrenze hinaus realisiert werden. Der Serviceanbieter wird über das standardisierte Protokoll SOAP mit seiner Umwelt kommunizieren. Die Service-orientierte Architektur wird durch das gewählte Protokoll als Webservice implementiert. Eine langfristige Speicherung der Daten wird über den Einsatz einer Datenbank umgesetzt. Die Softwarearchitektur des Systems wird als Webapplikation mit Datenhaltung umgesetzt.

4.1.2 System-Spezifikation

Im folgenden wird eine Spezifikation des Systems über die zu behandelnden Anwendungsfälle, der grundlegenden Systemarchitektur und seiner Struktur in Form von Aktivitäten modelliert. Da die Schwerpunkte auf die Bereitstellung eines Service liegen werden die Möglichkeiten eines UI-Interfaces nicht vertieft.

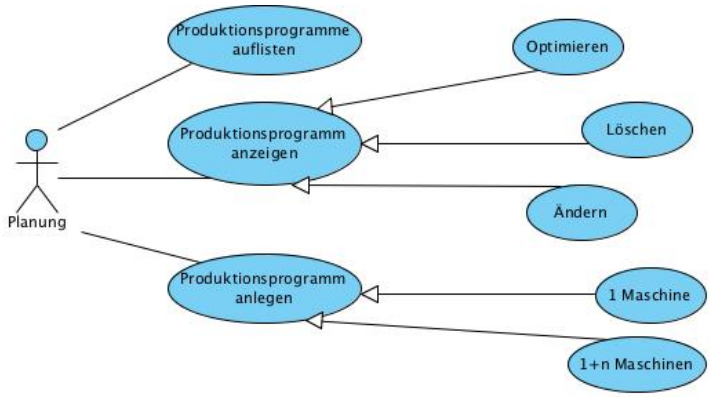

Abb. 10: Use-Case Modell Webservice

Quelle: Eigene Darstellung

Die in Abb. 10 dargestellten Anwendungsfälle werden aus Sicht eines Actors betrachtet der in seiner Funktion der Produktionsplanung einer Unternehmung zugeordnet werden kann. Grundlegend werden die drei Anwendungsfälle Auflisten von Produktionsprogrammen, Detailanzeige eines Produktionsprogramms und Neuanlage eines Produktionsprogramms betrachtet.

Das betrachtete Use-Case Modell aus Abb. 10 liefert die Grundlage für den zu implementierende Funktionsumfang im Webservice, sowie die Grundlage für ein hierauf aufbauendes User Interface.

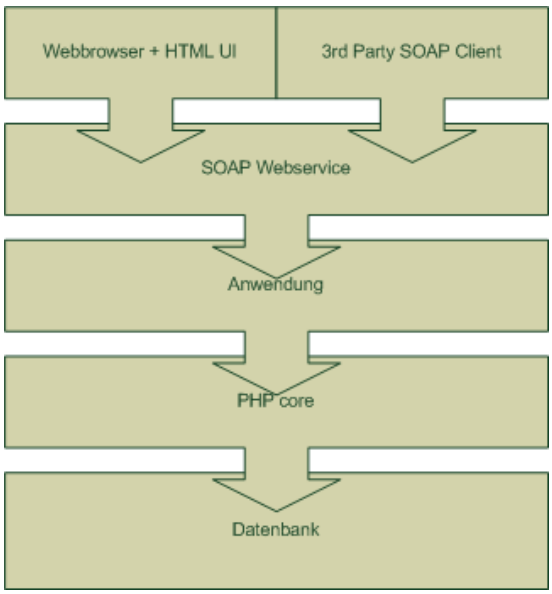

Abb. 11: Softwarestack für ein Webservice

Quelle: Eigene Darstellung

Der in Abb. 11 dargestellte Softwarestack modelliert die technisch umzusetzenden Grundlagen für den Webservice. Basis des Stacks ist eine Datenbank, die der Datenhaltung für zeitlich versetzte und aufeinander folgender Service-Anfrage implementiert. Auf der Datenbank setzt eine Laufzeitumgebung für die Programmiersprache PHP inklusiver deren Kernfunktionen. Die Anwendungslogik wird in PHP programmiert und setzt auf den PHP-Kern auf. Hier werden die für PHP spezifischen Webservice-API's genutzt, die auf der nächsten Ebene des Softwarestacks den konkreten SOAP Webservice mit seinen Funktionen, die im Use-Case-Modell aus Abb. 10 beschrieben wurden, bereit stellt. Über die SOAP-Schnittstelle kommuniziert das System über seine Grenzen mit der Umwelt. Hierzu zählen sowohl Webanwendungen die in HTML implementiert sind wie auch SOAP-Clients in weiteren unabhängigen Systemen.

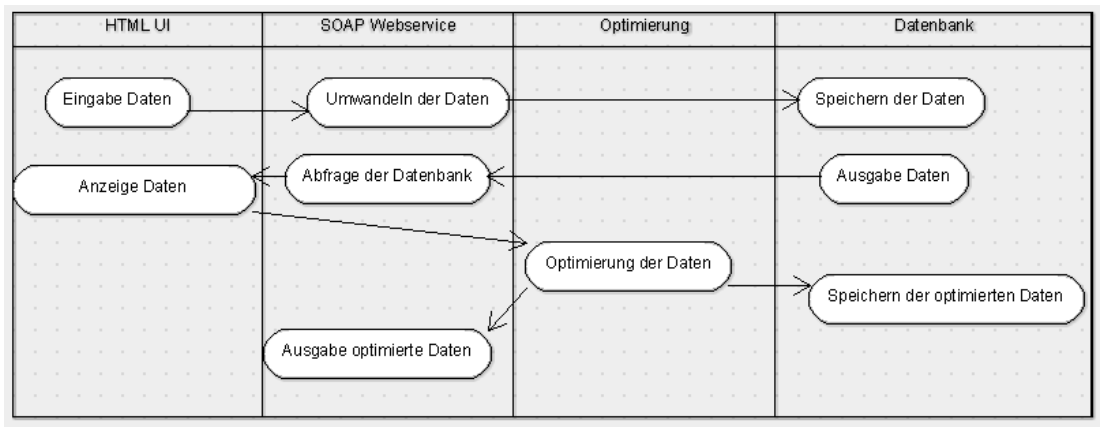

Abb. 12: Aktivitätsdiagramm Webservice-Struktur

Quelle: Eigene Darstellung

Das Aktivitätsdiagramm in Abb. 12 modelliert ein strukturiertes wirken der einzelnen Anwendungsschichten untereinander. Das Diagramm ist in die Bereiche HTML UI (stellvertretend für einen beliebigen SOAP-Client), Webservice, Optimierung und Datenbank partitioniert. Jede Partition bildet eine konkret zu implementierende Anwendungsschicht ab. Für die Client-Schicht ist charakteristisch das sie in keinem Anwendungsfall direkt mit der Optimierungs- oder Datenbankschicht kommuniziert. Diese Unabhängigkeit von Client und Optimierung sowie Datenhaltung ermöglicht eine hohe Wiederverwendbarkeit.

4.2 Objekt-orientierter Systementwurf

4.2.1 Datenmodell des Webservice

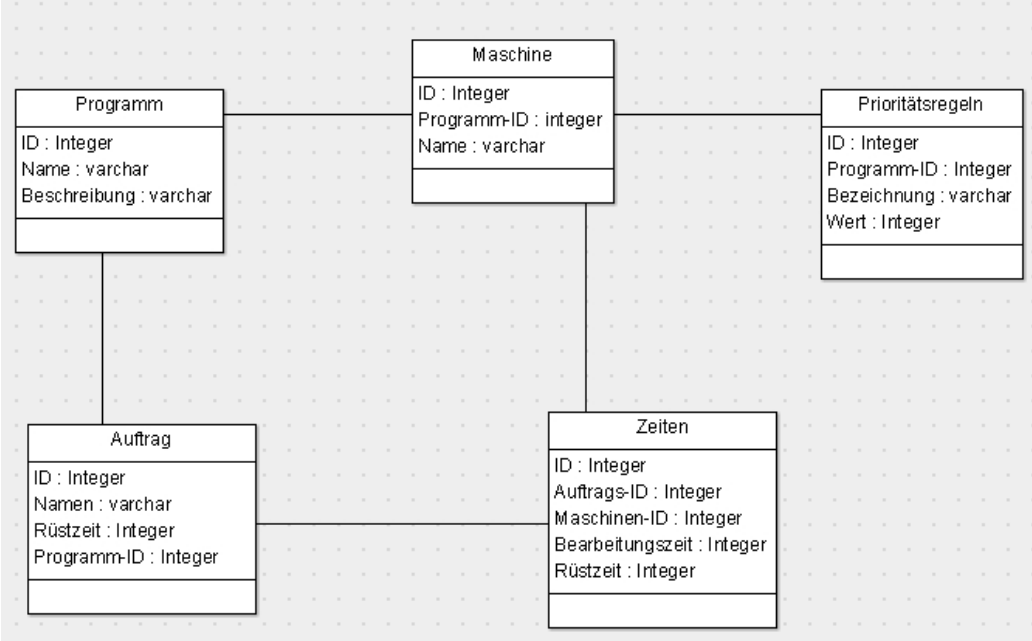

Abb. 13: Datenmodell als Klassendiagramm

Quelle: Eigene Darstellung

Das Datenmodell für den Webservice wird in Abb. 13 mit einem Klassendiagramm vereinfacht modelliert. Die Zielsetzung ist eine Modellbildung der Datenstrukturen wie sie in der konkreten Implementierung zur Laufzeit verarbeitet werden. Aus dem Klassenmodell wird in der Implementierungsphase ein relationales Datenmodell für die Datenbank erstellt. Ausgehend von dem Produktionsprogramm das über eine eindeutige ID, seinen Namen und einer Beschreibung identifiziert wird, lassen sich die Datenstrukturen für die Klassen Auftrag, Maschine, Zeiten und Prioritätsregeln verknüpfen. Die Klasse Zeiten ermöglicht das flexible verwalten von Bearbeitungs-

zeitmatrix und Rüstzeitmatrix je Auftrag und Maschine. Prioritätsregeln werden über eine eigene Klasse für die Mehrmaschinen-Probleme mit dem jeweiligen Programm über die Programm-ID verknüpft.

4.2.2 Klassenmodell der Optimierungsschicht

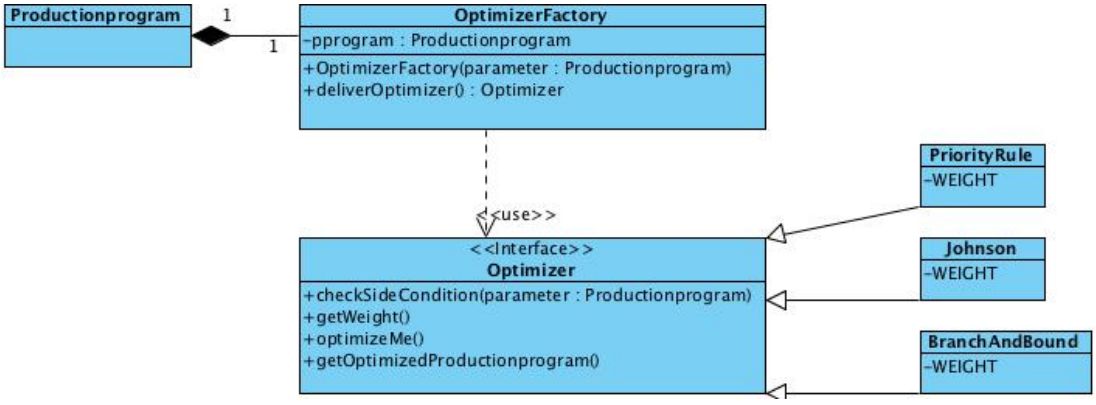

Abb. 14: Klassenmodell der Optimierungsschicht

Quelle: Eigene Darstellung

Die Optimierungsschicht wurde auf Grundlage des objektorientierten Entwurfsmusters „Factory-Pattern“ modelliert. Aufgrund der möglichen Komplexität von Problemstellungen aus der Ablaufplanung und einer Vielzahl von Lösungsmethoden, werden die Details der Optimierung vom Produktionsprogram getrennt um zu verhindern das die Klasse Productionprogram eine Komplexität annimmt die eine Implementierung und Wartung des Codes erschwert. Die Klasse Productionprogram steht in einer 1:1 Beziehung zu der OptimizerFactory, deren Aufgabe die Auswahl von geeigneten Optimierungsalgorithmen ist. Welche Algorithmen im System vorhanden sind und wie diese funktionieren ist für die Klasse Productionprogramm nicht sichtbar. Alle Klassen die einen Optimierungsalgorithmus implementieren müssen das Interface Optimizer implementieren. Somit ist ein einheitliches Verhalten über alle Klassen die ein Optimierungsverfahren darstellen garantiert. Die Auswahl eines geeigneten Algorithmus für ein Produktionsprogramm erfolgt durch die Klasse OptimizerFactory, die über Klassen-Reflektion alle zur Laufzeit geladenen Klassen mit einer Implemntierung des Interface Optimizer überprüft. Die Eignung einer Klasse wird über die statische Methode checkSideCondition überprüft. Sollten mehrere Klassen als Optimierer in Frage kommen, erfolgt die Auswahl über eine Gewichtung die in den jeweiligen Klassen als Attribut WEIGHT festgelegt ist.

4.2.3 Sequenzieller Ablauf eines Webservice-Calls

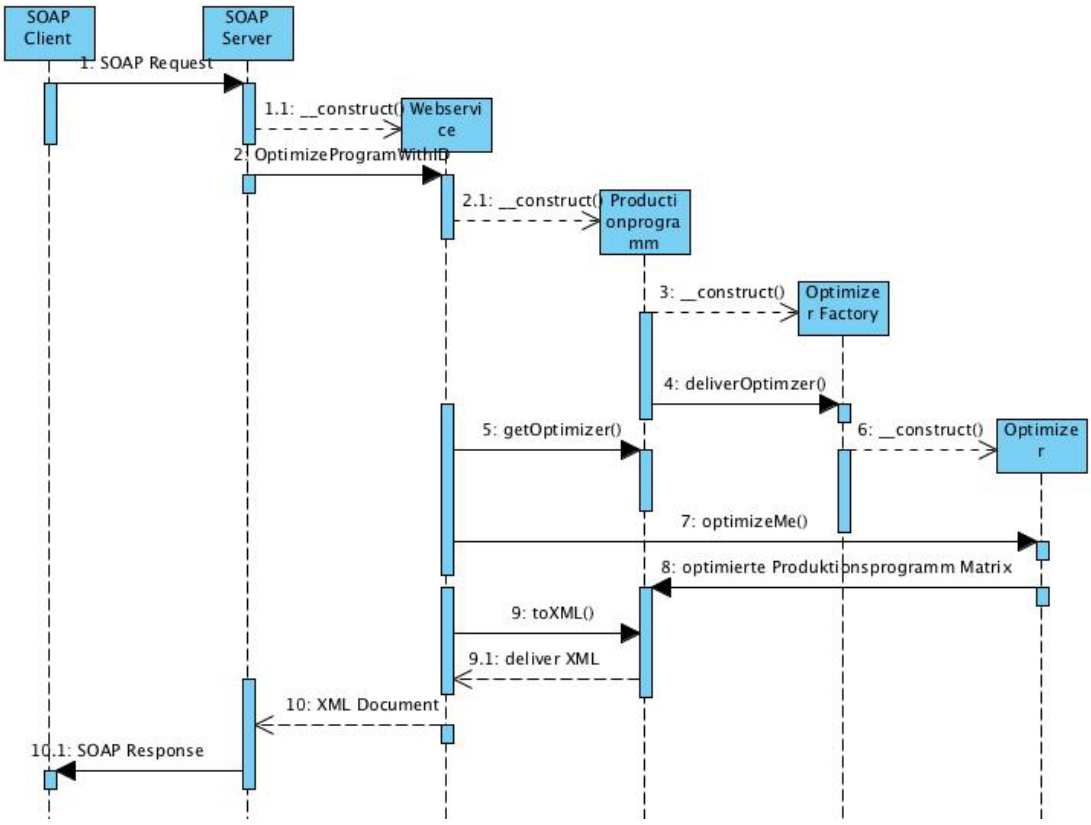

Abb. 15: Sequenzdiagramm Webservice-Call

Quelle: Eigene Darstellung

Der zeitlichen Verlauf einer Service-Anfrage sowie die daran beteiligten Klassen wurden in Abb. 15 mit einem Sequenzdiagramm modelliert. Hier wird verdeutlicht wie die einzelnen Klassen durch synchrone und asynchrone Nachrichten untereinander in zeitlicher Abfolge kommunizieren. Die Abläufe werden im wesentliche durch das bereits in Abschnitt 4.2.2 modellierte Factory-Pattern vorgegeben.

Die Instanz eines SOAP-Servers nimmt die Anfrage nach einem Webservice via SOAP-Request entgegen. Für die Verarbeitung einer spezifischen Service-Anfrage wird eine Instanz der Klasse Webservice erzeugt. An das Objekt vom Typ Webservice wird nun die Nachricht aus dem SOAP-Request in Form eines Funktionsaufrufs gesendet. In der Funktion OptimizeProgramWithID(id:int) wird eine Instanz der Klasse Productionprogram erzeugt, welche die Daten aus dem Produktionsprogramm mit der übermittelten ID hält. Die Instanz der Klasse Productionprogram erzeugt eine Instanz der Klasse OptimizerFactory und sendet anschließend die Nachricht deliverOptimizer() an das Objekt. Hier wird in der Funktion deliverOptimizer() der in Abschnitt 4.2.2 beschriebene Mechanismus ausgeführt um eine Klasse zu identifi-

zieren, die einen auf das Productionprogram anwendbaren Optimierungsalgorithmus implementiert. Für diese Klasse wird eine Instanz erzeugt und als Objekt-Referenz gespeichert. Die Instanz der Klasse Webservice ist nun in der Lage über eine Objekt-Referenz der Klasse Productionprogram eine Nachricht an die Instanz der Optimierungsklasse zu senden. Diese führt dann die eigentliche Optimierung des Produktionsprogramms an den Daten der Productionsprogram-instanz durch. Im Anschluss an die Optimierung erhält das Productionprogram-Objekt die Nachricht toXML() von der Webserivice-Instanz. Dadurch werden die Datenstrukturen des optimierten Produktionsprogramms als XML-Dokument an das Webservice-Objekt gesendet. Über den SOAP-Server wird zum Abschluss der Service-Anfrage das XML-Dokument an den SOAP-Client gesendet.

4.3 Feinentwurf ausgewählter Komponenten

4.3.1 Ein PHP SOAP-Server

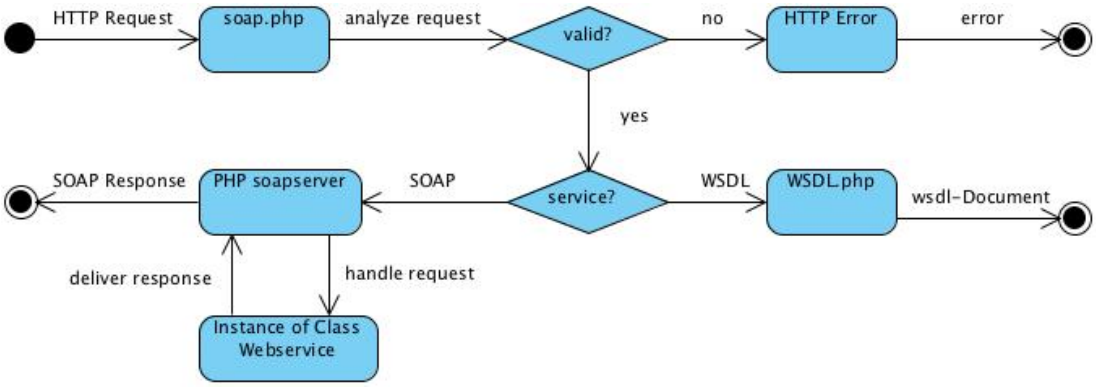

Abb. 16: Zustandsdiagramm PHP SOAP-Server

Quelle: Eigene Darstellung

Aus Sicht der Service-orientierten Architektur stellt ein SOAP-Server den Service-Anbieter auf technischer Ebene mit Hilfe des standardisierten Simple Objekt Access Protocols bereit. Die gewählte Implementierungssprache PHP enthält in seiner Kernfunktionalität bereits eine Implementierung für die Abwicklung von Service-Anfragen und –Antworten. Für die Verwendung dieser Funktion müssen Strukturierte Bedingungen geschaffen werden. In Abb. 16 wird eine solche Struktur mit einem Zustandsdiagramm modelliert. Ausgangspunkt ist ein HTTP-Request der über den PHP Kern an den Programmcode soap.php gesendet wird. Dort erfolgt eine Analyse und Überprüfung der Anfrage auf Korrektheit in Bezug auf Syntax und angefragter Funktionalität. Bei einer Fehlerhaften Anfrage wird der Prozess mit einer Fehlermeldung an den Service-Nachfrager beendet. Fehlerfreie Anfragen werden je nach angefragten Serice unterschieden. Für Anfragen nach einem WSDL-Dokument wird die

Service-Anfrage im Programmcode WSDL.php bearbeitet und der Prozess mit einem WSDL-Dokument das an den Service-Nachfrager gesendet wird beendet. Anfragen die über SOAP eine konkrete Funktion im Webservice abrufen werden über den PHP internen SOAP-Server abgewickelt. Der SOAP-Server arbeitet mit einer Instanz der Klasse Webservice, die alle bereitgestellten Funktionen eines Web-Service enthält, zusammen. Über die Instanz der Klasse Webservice werden die Anfragen und Antworten in XML abgewickelt. Der Prozess endet mit dem SOAP-Response welcher von dem PHP SOAP-Server an den Service-Nachfrager gesendet wird.

4.3.2 Darstellung des Algorithmus nach Johnson

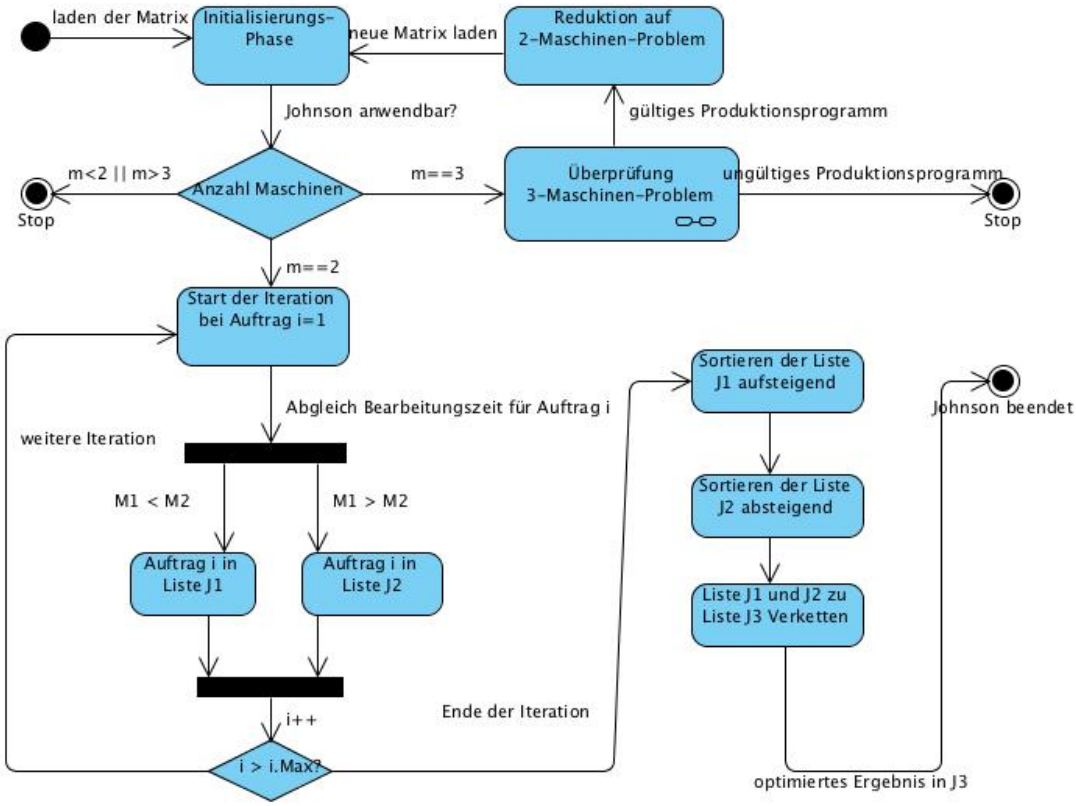

Abb. 17: Zustandsdiagramm für Johnson-Algorithmus

Quelle: Eigene Darstellung

In Abb. 17 wird der bereits in Abschnitt 3.3.2 beschriebene Johnson-Algorithmus mit einem Zustandsdiagramm modelliert. Die Modellierung orientiert sich an der mathematischen Einzelschritten des Algorithmus. Nach der Initialisierungsphase, die das Laden der Bearbeitungszeitmatrix sowie die Initialisierung der Listen J_1 und J_2 enthält, wird die Anwendbarkeit des Johnson-Algorithmus auf das vorliegende Produktionsprogramm überprüft. Für ein 3-Maschinen-Problem erfolgt die Modellierung in einem gesonderten Zustandsdiagramm. Programme für die der Johnson-Algorithmus nicht anwendbar ist führen jeweils zum Abbruch. Für gültige Programme erfolgt die Iteration über alle Aufträge sowie das Füllen und Sortieren der Listen

J_1 und J_2. Der Algorithmus wird mit dem Verketten der beiden Listen J_1 und J_2 zu dem optimalen Ergebnis beendet.

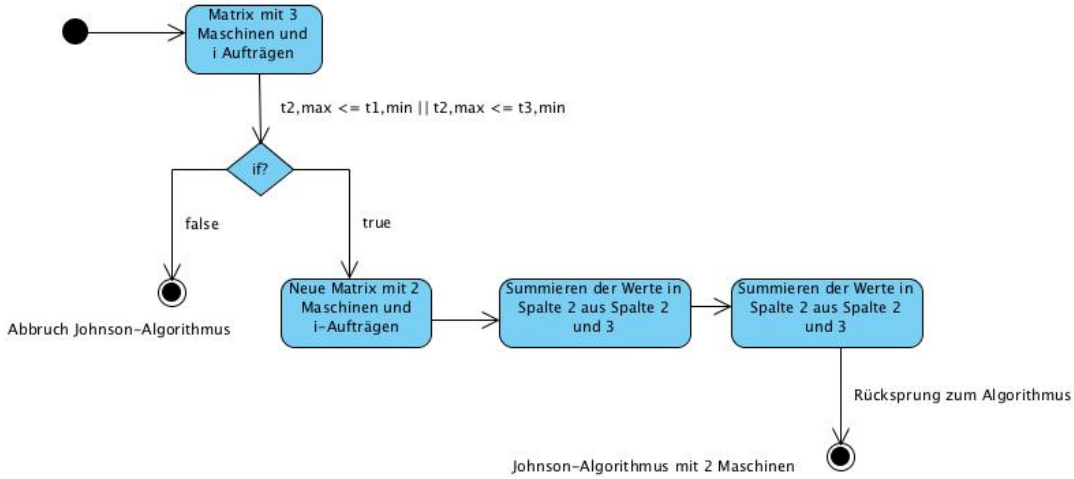

Abb. 18: Zustandsdiagramm Prüfung 3-Maschinenproblem

Quelle: Eigene Darstellung

Das Optimieren eines 3-Maschinen-Problems wird im Johnson-Algorithmus als Sonderfall behandelt und hier in einem gesonderten Zustandsdiagramm modelliert. Der Ablauf entspricht der Beschreibung in Abschnitt 3.3.2. Sofern die Nebenbedingungen nicht zutreffen erfolgt ein Abbruch des Algorithmus. Andernfalls wird das 3-Maschinen-Problem auf ein 2-Maschinen-Problem reduziert und eine neue Matrix im Algorithmus geladen. Im Anschluss erfolgt dann die Optimierung wie sie in Abb. 17 bereits modelliert wurde.

5 Fazit

Das Thema der Untersuchung wurde mit einem umfangreichen Grundlagenteil vorbereitet. Über die Einführung in Aspekte des Produktionsmanagements und den Methoden des Operations Research wurden wesentliche Wissensbereiche erarbeitet, die für das grundlegende Verstehen des Themenkomplex der Ablaufplanung wichtig sind.

Mit der Einführung in die Systemtheorie wurden Grundlagen für die Systemanalyse als Vorgehensmodell der Softwareentwicklung dargestellt. Die Betrachtung einer Aufgabenstellung als System und die Zerlegung in einzelne Elemente, die innerhalb der Systemgrenzen in Beziehungen stehen, eignet sich als Vorgehensmodell. Darauf Aufbauend wurde eine umfangreiche Einführung in die Objekt-orientierte Modellierung sowie der UML als graphische Notation erarbeitet. Es ist erkennbar das Systemanalyse und Objekt-orientierte Modellierung sich ergänzen. Eine Betrachtung ausgewählter Notationen und Modelle der UML legt die Grundlage für das Praxisbeispiel.

Für die grundlegende Betrachtung des Begriffs Webservice wurde eine abstrakte Betrachtungsweise herangezogen, die sich mit der Service-orientierten Architektur als Grundlage für Webservices beschäftigt. Zu einem guten Verständnis wurde hier auch konkrete Themen zu Protokollen und Datenformaten behandelt. Den Abschluss der Grundlagen bildet eine Aussicht auf die Programmiersprache PHP und die besondere Eignung für den Einsatz im Bereich der Webservices.

Eine Einführung in die Ablaufplanung sowie deren Variationen hat den Kern dieser Arbeit eingeleitet. Einfache Probleme mit 1-, 2 und 3 Maschinenproblemen wurden im Detail anhand von Beispielen gelöst. Somit wurde eine umfangreiche Abhandlung der zugrundeliegenden Algorithmen erzielt. Im Verlauf der behandelten Problemstellungen wurde bereits deutlich mit steigender Komplexität die Lösungen nur mit erheblichen Rechenaufwand gelöst werden können. Mit einem Ausblick auf weitere Lösungsansätze, die teilweise aus dem Forschungsbereich der künstlichen Intelligenz stammen, schließt das Kapitel zum Mehrmaschinenproblem ab.

Mit dem Praxisbeispiel wurde die Vision für das angestrebte Softwaresystem formuliert. Aus der Vision wurden konkrete Ziele abgeleitet, die verdeutlichen das eine Ideallösung in der Service-orientierten Architektur mit einem Webservice liegen muss. Die Anwendbarkeit der Objekt-orientierten Modellierung wurde in den Bereichen System-Spezifikation, Systementwurf und Feinentwurf für ausgewählte Syste-

melemente erfolgreich bewiesen. Hierfür wurden ausgewählte Notationen aus der UML eingesetzt.

Eine ideale Weiterführung dieser Arbeit wäre die Betrachtung der Modellgetriebenen Softwareentwicklung als schlüssiger Übergang von Modellierung, Modellierungswerkzeugen hin zu der Implementierung mit einer ausgewählten Programmiersprache.

Literaturverzeichnis

Achour, M., Betz, F., Dovgal, A., Lopes, N., Magnusson, H., Richter, G., et al. (12. 04 2012). *PHP-Handbuch*. (P. Olson, Produzent) Abgerufen am 12. 04 2012 von Einführung Reflections: http://de2.php.net/manual/de/intro.reflection.php

Achour, M., Betz, F., Dovgal, A., Lopes, N., Magnusson, H., Richter, G., et al. (11. 04 2013). *PHP-Handbuch Manual*. (P. Olson, Produzent) Abgerufen am 12. 04 2013 von Was kann PHP?: http://www.php.net/manual/de/intro-whatcando.php

Balzert, H. (2005). *Lehrbuch der Objektmodellierung.* Heidelberg: Spektrum Akademischer Verlag.

Balzert, H. (2009). *Lehrbuch der Softwaretechnik Basiskonzepte und Requirements Engineering.* Heidelberg: Spektrum Akademischer Verlag.

Balzert, H. (2011). *Lehrbuch der Softwaretechnik Entwurf, Implementierung, Installation und Betrieb.* Heidelberg: Spektrum Akademischer Verlag Heidelberg.

Beckmann, M., Gehring, H., Kistner, K.-P., Schneeweiß, C., Schwödiauer, G., & Zimmermann, H.-J. (1987). *Grundlagen des Operations Research* (Bd. 3). (T. Gal, Hrsg.) Berlin: Springer.

Bloech, J., Bogaschewsky, R., Buscher, U., Daub, A., Götze, U., & Roland, F. (2008). *Einführung in die Produktion.* Berlin Heidelberg: Springer-Verlag.

Boersch, I., Heinsohn, J., & Socher, R. (2007). *Wissensverarbeitung.* München: Elsevier GmbH.

Burkhard, R., Neumann, K., & Ohse, D. (1989). *Grundlagen des Operations Research* (Bd. 2). (T. Gal, Hrsg.) Berlin: Springer.

Domschke, W., & Drexl, A. (2005). *Einführung in Operations Research.* Berlin: Springer.

Eisenführ, F., & Weber, M. (2003). *Rationales Entscheiden.* Berlin: Springer-Verlag.

Ellinger, T., Beuermann, G., & Leisten, R. (2003). *Operations Research.* Berlin: Springer.

Gal, T., Horst, R., Isermann, H., & Müller-Merbach, H. (1991). *Grundlagen des Operations Research 1* (Bd. 1). (T. Gal, Hrsg.) Berlin: Springer.

Hansen, H. R., & Gustaf, N. (2005). *Wirtschaftsinformatik 2 Informationstechnik.* Stuttgart: Lucius & Lucius Verlagsgesellschaft mbH.

Heinrich, G. (2007). *Allgemeine Systemanalyse.* München: Oldenbourg Wissenschaftsverlag GmbH.

Heinrich, G., & Mairon, K. (2008). *Objektorientierte Systemanalyse.* München: Oldenbourg Wissenschaftsverlag GmbH.

Hillier, F., & Liebermann, G. (2002). *Operations Research Einführung.* München: Oldenbourg Wissenschaftsverlag GmbH.

Krallmann, H., Schönherr, M., & Trier, M. (2007). *Systemanalyse im Unternehmen.* München: Oldenbourg Wissenschaftsverlag GmbH.

Kurbel, K. (1992). *Entwicklung und Einsatz von Expertensystemen.* Berlin: Springer-Verlag.

Laudon, K., Laudon, J., & Schoder, D. (2010). *Wirtschaftsinformatik Eine Einführung.* München: Pearson Education Deutschland GmbH.

Masak, D. (2007). *SOA? Serviceorientierung in Business und Software.* Heidelberg: Springer-Verlag.

Melzer, I. (2010). *Service-orientierte Architekturen mit Web Services.* Heidelberg: Spektrum Akademischer Verlag.

Neumann, K. (1975). *Operations Research Verfahren* (Bd. 1). München: Carl Hanser Verlag.

Oestereich, B. (2012). *Analyse und Design mit der UML 2.5 Objektorientierte Softwareentwicklung.* München: Oldenbourg Wissenschaftsverlag GmbH.

Oestereich, B. (2009). *Die UML-Kurzreferenz 2.3 für die Praxis.* München: Oldenbourg Wissenschaftsverlag GmbH.

Pinedo, M. L. (2012). *Scheduling Theory, Algorithms, and Systems.* New York: Springer Science+Business Media.

Plümer, T. (2003). *Logistik und Produktion.* München: Oldenbourg Wissenschaftsverlag GmbH.

Reimers, S., & Thies, G. (2012). *PHP 5.4 und MySQL 5.5 Das umfassende Handbuch.* Bonn: Galileo Press.

Schader, M., & Rundshagen, M. (1996). *Objektorientierte Systemanalyse.* Berlin: Springer-Verlag.

Schiemenz, B., & Schönert, O. (2005). *Entscheidung und Produktions.* München: Oldenbourg Wissenschaftsverlag GmbH.

Schneeweiß, C. (2002). *Einführung in die Produktionswirtschaft.* Berlin: Springer-Verlag.

Seidl, M., Brandsteidl, M., Huemer, C., & Kappel, G. (2012). *UML @ Classroom Eine Einführung in die objektorientierte Modellierung.* Heidelberg: dpunkt.verlag GmbH.

Sommerville, I. (2012). *Software Engineering.* München: Pearson Deutschland GmbH.

Suhl, L., & Mellouli, T. (2009). *Optimierungssysteme Modell, Verfahren, Software, Anwendungen.* Berlin: Springer-Verlag.

Thonemann, U. (2010). *Operations Management Konzepte, Methoden und Anwendungen.* München: Pearson Deutschland GmbH.

Zäpfel, G. (1996). *Grundzüge des Produktions- und Logistikmanagement.* Berlin: Walter de Gruyter & Co.

Zäpfel, G. (1982). *Produktionswirtschaft Operatives Produktions-Management.* Berlin: Walter de Gruyter & Co.

Zahn, E., & Schmid, U. (1996). *Produktionswirtschaft.* Stuttgart: v. Lucius Verlagsgesellschaft mbH.

Zimmermann, H.-J. (2005). *Operations Research Medhoden und Modelle.* Wiesbaden: Vieweg & Sohn.

Zimmermann, W. (1995). *Operations-Research Quantitative Methoden zur Entscheidungsvorbereitung.* München: Oldenbourg Verlag GmbH.

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt habe. Ich versichere auch, dass ich bei allen Gedanken, Befunden und anderen Inhalten, die nicht von mir stammen, direkt vor Ort auf die entsprechenden Quellen verwiesen habe. Alle wörtlichen Zitate sind als solche kenntlich gemacht.

Clausthal-Zellerfeld, 28.11.2013, Steffen Schütz