Technische Universität Dortmund
Fakultät Maschinenbau
Lehrstuhl IT in Produktion und Logistik

Masterarbeit

Ansätze zur Integration eines Vorgehensmodells des Data Farmings in Supply Chains

verfasst von:

Mirco Bonnemann

Matrikelnummer: 208454
Studiengang: Maschinenbau

Ausgegeben am: 02.11.2020
Eingereicht am: 19.04.2021

Prüferin: Dr.-Ing. Dipl.-Inform. Anne Antonia Scheidler
Betreuer: M. Sc. Joachim Hunker

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Formelverzeichnis

Quellcodeverzeichnis

Abkürzungsverzeichnis

ASIM	Arbeitsgemeinschaft Simulation
ADOdb	Active Data Objects DataBase
CCD	Central Composite Design
CPU	Central processing unit
CTF	Capture the Flag
DACDAM	Data-farmable Agent-based Cyber Defense Assessment Model
DEDS	Discrete-event dynamic system
DF	Data Farming
DFTOP	Data Farming decision support Tool for Operation Planning
DFS	Data Farming Services
DFSS	Data Farming Services System
DSB	Data Science Board
DTA	Defence Technology Agency
ERP	Enterprise-Resource-Planning
ETL	Extract, Transform, Load
FLOPS	Floating point operations per second
GFLOPS	Giga floating operations per second
GPU	Grapical processing unit
GUI	Grapical user interface
HPC	High-Performance-Computing
IA	Integrationsansatz
ISAAC	Irreducible Semi-Autonomous Adaptive Combat
KI	Künstliche Intelligenz
LHD	Latin Hypercube Design
LHS	Latin Hypercube Sampling
MHPCC	Maui High Performance Computing Center
MANA	Map-Aware Non-Uniform Automata
MSaaS	Modeling & Simulation as a Service
MS Access	Microsoft Access
MS-DOS	Microsoft Disk Operating System
MSG	Modeling & Simulation Group
NATO	North Atlantic Treaty Organization
OFAT	One-Factor-at-a-Time
PB	Plackett-Burman
REST	Representational State Transfer
SC	Supply Chain
SD	System dynamics
SQL	Struktured Query Language
SCM	Supply Chain Management
UML	Unified Modeling Language
USMC	United States Marine Corps

VBA Visual Basic for Applications
VDI Verband deutscher Ingenieure
V&V Verifikation & Validierung
XML Extensible-Markup-Language

1 Einleitung

Einhergehend mit der digitalen Vernetzung von Menschen und Dingen über das Internet hat sich auch das weltweite Konsumverhalten seit dem Beginn der Digitalisierung in den späten 90er Jahren stark verändert [Kah+18, S. 8]. In einem Trendbericht des Umweltbundesamts über die Auswirkungen der Digitalisierung wird die Veränderung der gesellschaftlichen Wirtschafts- und Lebenweise, als Pendant zum technischen Wandel in Richtung „Industrie 4.0“, als „Konsum 4.0“ betitelt [Kah+18, S. 10].

Dabei bestellen nicht nur Endverbraucher verstärkt Waren online, sondern auch alle Akteure im gesamten Wertschöpfungsprozess eines Produktes, der sog. Supply Chain, besitzen über das Internet eine größere Reichweite auf den globalen Markt [Haa17, S. 26] [HN08, S. 1]. In Folge dieses Wandels ist der Materialfluss in den letzten Jahren exponentiell angestiegen [HN08, S. 1]. Gleichzeitig verlangen wachsende Kundenanforderungen flexible Transportkonzepte mit genauer Einhaltung einer möglichst kurzen Lieferzeit [Bec04]. Um diesen Entwicklungen standzuhalten, müssen moderne Supply Chains (SCs) den Zielkonflikt zwischen maximaler Lieferbereitschaft und minimalen Beständen bei der inhärenten Unsicherheit von Angebot und Nachfrage lösen [HN08] [Gop13]. Aus den Bestrebungen zur Erfüllung des Zielkonflikts ist ein kritischer Anstieg in der Komplexität und Dynamik von Logistiksystemen zu erkennen, weshalb eine analytische Auslegung und Bewertung dieser Systeme kaum noch möglich ist [HN08, S. 1] [Law13, S. 1].
Aus diesem Grund hat die Simulationstechnik in der Logistik in den letzten Jahrzenten kontinuierlich an Bedeutung gewonnen [Gut+17, S. 1]. Bei einer Simulation wird im Kontext der Produktion und Logistik ein System in einem vereinfachten Modell abgebildet, um durch Experimente Erkenntnisse zu gelangen, die auf die Wirklichkeit übertragbar sind [Gut+17, S. 22–23]. Dabei besitzt ein Simulationsmodell immer definierte Startparameter, die nach der Ausführung ein Ergebnis liefern [Gut+17, S. 20]. Das primäre Ziel besteht dabei darin, Einflüsse der Startparameter auf die Simulationsergebnisse zu untersuchen, um ein besseres Systemverständnis zu erlangen.
Durch die Verwendung dieser Technik können für Systeme komplexe Wirkzusammenhänge abgeleitet werden ohne die realen Systeme dafür zu verwenden. Ein Nachteil bei der Durchführung von Simulationen besteht allerdings in der Planung von Experimenten, bei denen der Fokus darauf gelegt wird für einen minimalen Aufwand möglichst wenige Parameterkonfigurationen zu simulieren [Gut+17, S. 177]. So werden Vorgehensweisen wie das statische oder dynamische Faktor-Design angewendet, um im Vorfeld bereits eine „intelligente Auswahl zu untersuchender Parameterkombinationen zu treffen“ [Gut+17, S. 177]. Aus einem Versuchsplan, der nicht die gesamte Bandbreite möglicher Simulationsläufe repliziert, kann jedoch nicht das bestmögliche Systemverhalten resultieren.
Während nahezu jedes in der Literatur bekannte Vorgehensmodell für die Anwendung von Simulationstechniken die Experimentenplanung beschreibt, betrachtet keines dieser Vorgehensmodelle die gesamte Bandbreite möglicher Parameterkombinationen [RSW08, S. 29]. Ein Grund dafür liegt wahrscheinlich in der verfügbaren Rechenleistung von Computern zum Zeitpunkt der Veröffentlichung der Vorgehensmodelle. Während Vorgehensmodelle

nach Sargent oder Balci in den 90er Jahren erschienen sind, besaßen die schnellsten Computer dieser Zeit eine Rechenleistung von ca. $60 \cdot 10^6$ Operationen pro Sekunde [RSW08, S. 30–33] [Smi98, S. 1] [Top21]. Aktuelle Supercomputer verfügen mit $122 \cdot 10^{12}$ sekündlichen Rechenoperationen hingegen über zwei Millionen mal so viel Leistung, wodurch für die Simulationstechnik ein enormes ungenutztes Potential vorhanden ist [Top21]. Im Gegensatz zu den klassischen Simulationstechniken der Produktion und Logistik existiert jedoch noch eine Weitere, das sog. Data Farming.

Der Begriff Data Farming stammt ursprünglich aus dem US-militärischem Kontext und verwendet einen interdisziplinären Ansatz der Simulation, Hochleistungsrechnung und statistischen Analyse zur Untersuchung spezifischer Fragestellungen. Die Idee dieser Methode basiert darauf, nicht nur über einzelne Parameterbereiche, sondern über möglichst alle Ausgänge eines Szenarios durch Simulationen Daten zu generieren und auszuwerten [HS16, S. 1]. Die verwendeten Simulationsmodelle sind dabei darauf ausgelegt die Berechnungsdauer einer Replikation durch den Einsatz von Hochleistungsrechnung zu minimieren, um alle möglichen Replikationen der Startparameter sehr effizient in verhältnismäßig kurzer Zeit abzuwickeln.
Der Ansatz des Data Farmings könnte auch alle möglichen Ausgänge spezifischer SC-Simulationen analysieren, um bestmögliche Lösungen für aktuelle SC-Probleme zu erarbeiten. Verglichen mit bekannten Vorgehensmodellen für Simulationsprobleme wäre es möglich sowohl die Anzahl als auch den Bereich zu untersuchender Parameter zu erhöhen, um aus generierten Daten durch umfassende Datenanalysen zuverlässige Aussagen auf die Realität zu schließen.

Aus diesem Potential kann somit das folgende Forschungsthema für die vorliegende Masterarbeit abgeleitet werden. Grundlegend sollen die Einsatzmöglichkeit von Data Farming innerhalb der Supply Chain überprüft werden. Dabei verfolgt diese Masterarbeit zwei Hauptziele.
Das Erste besteht darin, Ansätze für ein Vorgehen zu erarbeiten, um Data Farming als Service innerhalb einer Supply Chain zu integrieren. Dafür gilt es zunächst die Vorgehensweisen und Arten der Simulationstechniken in Supply Chains zu erarbeiten, um für die Integrationsleistung Anforderungen an Data Farming Modelle stellen zu können. Da das Data Farming mit dem militärischem Ursprung in einem komplett anderen Kontext entwickelt wurde, gilt es als weiteres Teilziel das Themengebiet nach bereits existierenden Vorgehensmodellen oder definierten Phasen zur strukturierten Anwendung zu untersuchen. Die erarbeiten Integrationsansätze bilden die Basis für den zweiten Teil der Arbeit, in dem Data Farming an einem fiktiven Szenario einer Supply Chain angewendet wird. Dabei sollen sowohl die formulierten Integrationsansätze des ersten Hauptziels validiert, als auch Potentiale aufgezeigt werden, die sich durch die praktische Anwendung ergeben. Die Arbeitspakete des zweiten Teils orientieren sich dabei strikt an den formulierten Integrationsansätzen. Der Fokus richtet sich auf einen strukturierten praktischen Einsatz von Data Farming für eine typische Problemdomäne einer Supply Chain, bei dem die Planung, Entwicklung und Anwendung eines Simulationsmodells und die anschließend folgende Untersuchung der Ergebnisse im Vordergrund steht.

Zur Erfüllung aller Ziele verfolgt die vorliegende Masterarbeit folgendes methodisches Vorgehen. Zu Beginn werden in Kapitel 2 sowohl die notwendige Grundlage zu Supply Chains als auch zu der Simulationstechnik im produktionslogistischen Kontext geschaffen. Anfänglich gilt es zunächst die Eigenschaften und Effekte von Supply Chains und die Inhalte des Supply Chain Management zu erläutern. In Abschnitt 2.1 werden die Grundlagen der Simulationstechniken im Kontext der Produktion und Logistik beschrieben, die ebenfalls repräsentativ für Simulationen in Supply Chains sind. Dafür gilt es zunächst spezifische Begriffe der Simulationstechnik zu erläutern. Der Fokus liegt dabei in der Definition des System- bzw. Modellbegriffs und der Definition von Experimenten, Simulationsläufen, Parametern und Zielgrößen. In Abschnitt 2.1.1 werden verschiedene Arten von Simulationsmodellen unterschieden, wobei grundsätzlich zeitdiskrete und kontinuierliche bzw. determinstische und stochastische Modelle voneinander abgegrenzt werden. Weiterhin werden in Abschnitt 2.1.2 die wesentlichen Bestandteile einer Simulationsstudie, von der Modellbildung bis zur Ergebnisanalyse, vorgestellt und detaillierter Bezug auf die unterschiedlichen Möglichkeiten einer Experimentenplanung genommen. Im folgenden Abschnitt 2.1.3 werden unterschiedliche Vorgehensmodelle für die systematische Anwendung von Simulationstechniken vorgestellt und miteinander verglichen, um gemeinsame Phasen zu identifizieren. Der letzte Abschnitt 2.1.4 der Simulationsgrundlagen handelt von Kriterien für die Simulationswürdigkeit eines Problemfalls. Im Zuge dessen kann in einer Machbarkeitsstudie abgewägt werden, ob eine Simulation das geeignete Werkzeug für die Lösung eines Problems ist. Weiterhin werden Vor- und Nachteile für den Einsatz von Simulationstechniken formuliert.
Kapitel 3 ist das Ergebnis einer umfassenden Literaturrecherche zum Forschungsgebiet des Data Farmings. Dafür wird in Abschnitt 3.1 die Notwendigkeit und der Ursprung, sowie der historische Kontext mit wichtigen Projekten für die Anwendung von Data Farming erarbeitet. Im folgenden Abschnitt 3.2 werden einheitliche Strukturen für die Entwicklung und Anwendung von Data Farming Modellen vorgestellt, die zusammen mit den Vorgehensmodellen der Produktion und Logistik aus Abschnitt 2.1.3 die Basis für die Erarbeitung der Integrationsansätze bilden. Letztlich wird nach der Bearbeitung des gesamten Forschungsgebiets in Abschnitt 3.3 eine allgemeingültige Definition des Begriffs „Data Farming“ formuliert.
Im nachfolgenden Kapitel 4 werden auf Basis der aufbereiteten Grundlagen aus Kapitel 2 und Kapitel 3 Integrationsansätze zur Erfüllung des ersten Hauptziels formuliert. Dafür werden in Abschnitt 4.1 als Erstes die gemeinsamen Phasen der Vorgehensmodelle aus der Produktion und Logistik mit den Strukturen des Data Farmings verglichen, um potentielle Phasen für ein integriertes Vorgehen zu erkennen. Als Zweites werden in Abschnitt 4.2 Anforderungen an die Phaseninhalten und die Verknüpfung der Phasen gestellt. Abschließend wird in Abschnitt 4.3 schließlich ein Vorgehensmodell auf Basis der geleisteten Integrationsarbeit erstellt.
In Kapitel 5 wird mit dem entwickelten Vorgehen aus Kapitel 4 das zweite Hauptziel bearbeitet, indem der Einsatz von Data Farming an einem praktischen Szenario einer Supply Chain angewendet wird. In Abschnitt 5.1 wird dafür die Entwicklung eines Simulationsmodells beschrieben, während in Abschnitt 5.2 das fiktive Szenario erstellt und simuliert wird. Abschließend werden in Abschnitt 5.3 die Ergebnisse der Simulationsläufe

ausgewertet und in Abschnitt 5.4 ein Fazit in Bezug auf die genannten Zielstellungen gezogen.
Zum Abschluss der Arbeit wird das gesamte Forschungsthema in Kapitel 6 zusammengefasst und ein Ausblick auf weitere offene Aspekte des Forschungsgebiets gegeben.

2 Simulationen in Supply Chains

Trotz des veränderten Konsumverhaltens und den wachsenden Kundenanforderungen an die Flexibilität und Dynamik von Logistiksystemen haben sich die Ziele der Logistik im wesentlichen nicht verändert: „Die sichere Versorgung mit Materialien und Gütern zu optimalen Kosten und Beständen“[Koe11, S. 21]. Im klassischen Sinne sind dafür die sechs R's der Logistik zu erfüllen, die besagen, dass die **r**ichtige Menge der **r**ichtigen Objekte am **r**ichtigen Ort zum **r**ichtigen Zeitpunkt in der **r**ichtigen Qualität und zu den **r**ichtigen Kosten bereitzustellen sind [Koe11, S. 21]. In vielen literarischen Quellen werden auch sieben oder sogar acht R's der Logistik erwähnt, bei denen im Kontext der IT-gestützten Logistik noch die Aspekte des **r**ichtigen Kunden und der **r**ichtigen Informationen erwähnt werden [Aun15]. Bei der zunehmenden Komplexität von Logistiksysteme ist eine unternehmensinterne Sichtweise meistens nicht mehr ausreichend, um den gesamten Materialfluss bis zum Endkunden effektiv zu optimieren. Erforderlich ist die ganzheitliche Betrachtung des gesamten Wertschöpfungsnetzes vom Rohstoffmateriallieferanten bis zum Endkunden, der sog. Supply Chain [Bec04, S. 10].

Supply Chain (SC) bedeutet wörtlich übersetzt Versorgungs- oder Lieferkette und beinhaltet alle Unternehmen, die an der Entwicklung, Erstellung und Lieferung von Erzeugnissen beteiligt sind [Bec04, S. 1]. Nach Beckmann [Bec04, S. 2] repräsentiert die SC „den Fluss von Leistungsobjekten durch ein Netzwerk von Wertschöpfungspartnern, das sich vom Rohstofflieferanten bis zum Endverbraucher erstreckt.“. Die Leistungsobjekte bestehen dabei aus Materialien, Informationen und Finanzen [Bec04, S. 2].
In Abbildung 2.1 ist eine SC prozessorientiert dargestellt. Die Knoten spiegeln dabei die verschiedenen Akteure wieder während die Kanten möglichen Leistungsflüsse zwischen den Akteuren symbolisieren. Die Akteure bestehen in der Regel aus Lieferanten, Herstellern, Groß- bzw. Einzelhändlern und Kunden. Dabei müssen in einem SC-Design aber nicht zwangsläufig alle Arten von Akteuren vorhanden sein. Die ganzheitliche Betrachtung der SC verfolgt das Ziel, alle Leistungsflüsse der Kanten so abzustimmen, dass ein optimales Gesamtergebnis erreicht wird, bei dem die Faktoren Kosten, Zeit und Qualität im Mittelpunkt stehen [Pfo18, S. 338].

Supply Chain Management (SCM) Das SCM ist eine Bezeichnung aller Tätigkeiten zur Planung, Steuerung und Überwachung sämtlicher Material-, Informations- und Finanzflüsse entlang der Lieferkette [Bec04, S. 5]. Dieses integrierte Management verfolgt einen prozess- als auch kundenorientierten Ansatz, bei dem das gesamte Wertschöpfungsnetzwerk auf die Maximierung des Kundennutzens ausgelegt ist [Bec04, S. 10]. Gleichzeitig sollen die Kosten aller Akteure minimiert werden, indem alle intra- und interorganisationalen Leistungsflüsse aufeinander abgestimmt werden.
Eine große Herausforderung dieses Vorhabens besteht in den dynamischen Charakteristiken einer SC, die zu Bedarfsveränderungen führen [Bec04, S. 7]. In Folge dessen werden bis

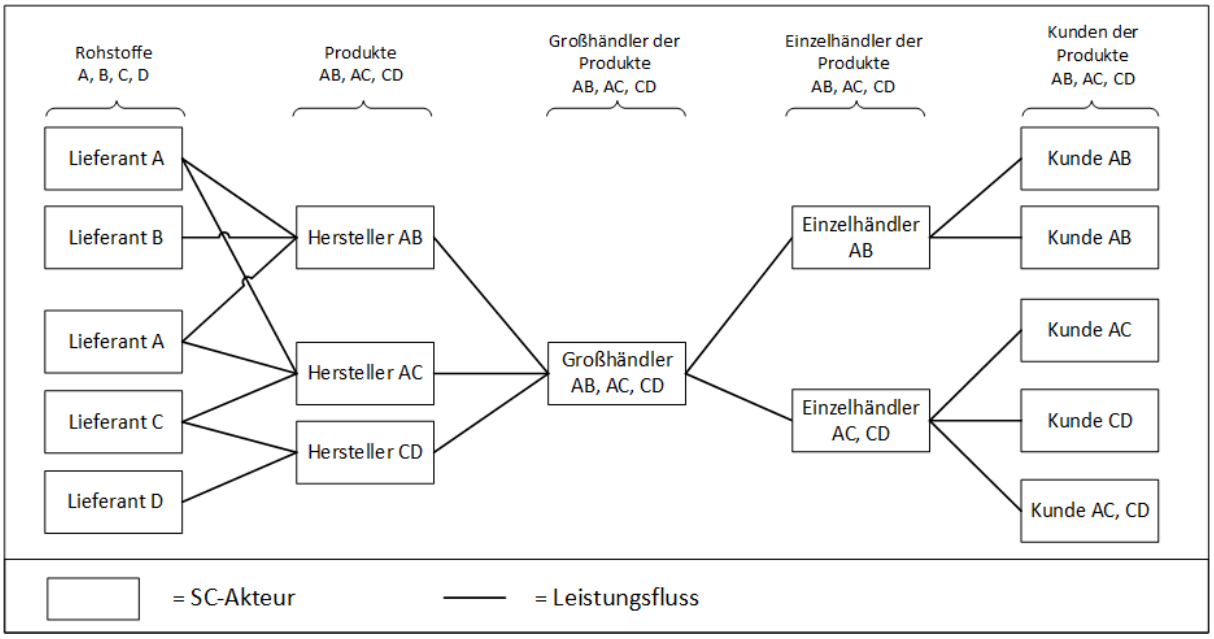

Abb. 2.1: Prozessorientierte Darstellung einer Supply Chain (vereinfacht)

zur Reaktion des vorgeschalteten Lieferanten Rückstände bzw. Überstände aufgebaut, die typischerweise noch durch verzögerte Informationsweitergaben verstärkt werden [Bec04, S. 7]. Diese dynamische Eigenschaft wird Bullwhip-Effekt genannt und ist eines der Kernpunkte, die mit Hilfe eines ausgereiften SCM vermieden werden können. Typische Aufgaben des SCM bestehen demnach in der Reduzierung von Unsicherheiten durch zentralisierte und integrierte IT-Systeme, der Reduzierung von Variabilitäten durch synchronisierte Bestellzyklen oder der Bildung strategischer Partnerschaften [Bec04, S. 9].
Insgesamt bietet eine optimierte SC erheblichen Nutzen für alle beteiligten Akteure. Durch die kundenorientierte Ausrichtung der Geschäftsprozesse kann die Kundenzufriedenheit durch verkürzte Durchlauf- und Lieferzeiten und verbesserte Termintreue enorm gesteigert werden [Bec04, S. 14]. Darüber hinaus kann über einen permanenten Informationsaustausch zwischen den Akteuren und informationsgestützte Bedarfsprognosen eine vollkommene Transparenz über Mengen, Bestands- und Absatzsituationen geschaffen werden [Bec04, S. 15].

Aufgrund der dynamischen Effekte einer SC und der damit verbundenen inhärenten Unsicherheit der Nachfragen beschäftigt sich ein wesentlicher Bestandteil der strategischen Planungsebene mit der Simulation von Material, Informations- und Finanzflüssen, um optimale Ergebnisse prognostizieren zu können. Die Zielsetzungen von SC-Simulationen sind dabei stark abhängig von der zu untersuchenden Fragestellung. So können Simulationstechniken bspw. eingesetzt werden, um Transportrouten und Lagerbestände zu optimieren oder Bedarfsschwankungen zu prognostizieren. In Abschnitt 2.2 wird detaillierter auf die verschiedenen Arten und Anwendungszwecke von SC-Simulationen eingegangen. Im nachfol-

genden Abschnitt wird dafür zunächst Bezug auf die Grundlagen der Simulationstechniken genommen.

2.1 Grundlagen der Simulationstechnik

Gutenschwager et al. [Gut+17, S. 22] beschreiben eine Simulation als „eine Problemlösungsmethode, bei der durch Experimente mit Simulationsmodellen Aussagen über das Verhalten der durch die Modelle beschriebenen Systeme gewonnen werden“ [VDI18, S. 28]. Der Verband deutscher Ingenieure (VDI) konkretisiert den Begriff Simulation für die Produktion und Logistik als das „Nachbilden eines Systems mit dynamischen Prozessen in einem experimentierbaren Modell, um zu Erkenntnissen zu gelangen, die auf die Wirklichkeit übertragbar sind“.
Demnach ist für die Durchführung einer Simulation immer ein Simulationsmodell erforderlich, welches das abstrahierte Verhalten eines realen oder geplanten Systems widerspiegelt [MK11, S. 13]. Das System wird nach DIN IEC Norm 60050-351 als eine „Menge miteinander in Beziehung stehender Elemente, die in einem bestimmten Zusammenhang als Ganzes gesehen und als von ihrer Umgebung abgegrenzt werden“, betrachtet [Deu, S. 21]. Jedes System ist dabei in seinen Elementen begrenzt und besitzt sog. Systemgrenzen, über die das System von seiner Umgebung getrennt wird [Gut+17, S. 11]. „Über definierte Schnittstellen kann ein System an den Systemgrenzen Materie, Energie und Information [...] austauschen“, welche entweder in Form von Eingangsgrößen in das System gelangen oder dieses als Ausgangsgrößen verlassen.[Gut+17, S. 11]
Das Simulationsmodell orientiert sich immer am System und stellt somit „eine vereinfachte Nachbildung eines [...] Systems mit seinen Prozessen“ dar [VDI18, S. 19]. Die Vereinfachung des Systems wird auch als Abstraktion bezeichnet, bei der es gilt einen angemessenen Detaillierungsgrad zu berücksichtigen, welcher je nach System und Untersuchungsfrage unterschiedlich sein kann [MK11, S. 13]. Der Nutzen dieser Simulationsmodelle besteht darin Experimente zu planen und durchzuführen.
Nach dem VDI ist ein Simulationsexperiment die „gezielte empirische Untersuchung des Modellverhaltens über einen bestimmten Zeithorizont durch wiederholte Simulationsläufe mit systematischen Parametervariationen“ [VDI18, S. 29]. Ein Simulationslauf bezeichnet dabei eine Ausführung des Modells mit definierten Parametern zur Nachbildung der Eingangsgrößen des realen oder geplanten Systems. Bei der Durchführung von Experimenten gilt es einen statistischen Versuchsplan aufzustellen, um bereits im Voraus eine geeignete Wahl an Parameterkonstellationen zu treffen [VDI18, S. 32]. Das Ziel dieses Vorgehens besteht darin, den Aufwand durch die Anzahl der notwendigen Simulationsläufe zu reduzieren [Hrd+97, S. 19].
Bei der Planung von Experimenten unterscheidet Mertens [Mer82, S. 21] zwischen What-if- und How-to-achieve-Analysen die in Abschnitt 2.1.2 näher behandelt werden. Während durch What-if-Analysen primär das Systemverhalten bei der Veränderung der Parameter untersucht wird, dient letzteres vielmehr dazu einen oder mehrere Parameter durch gezielte Variationen zu optimieren [Mer82, S. 21] [VDI18, S. 37]. In diesem Kontext stehende

Parameter werden auch Zielgrößen genannt [VDI18, S. 37]. Im Anschluss an ein Experiment liefert eine Simulation im Allgemeinen ein Simulationsergebnis zur Darstellung der „Höhe und des zeitlichen Verlaufs der Zustandsgrößen eines Modells zwischen Anfang und Ende eines Simulationslaufes“ [VDI18, S. 29]. Die Auswertung der Simulation erfolgt durch die statistische Analyse oder die Visualisierung der Ergebnisse, wodurch letztendlich Erkenntnisse auf das reale oder geplante System übertragen werden.

2.1.1 Arten von Simulationsmodellen

Simulationsmodelle werden im Allgemeinen nach ihrem Zeitverhalten, ihrer Zeitmenge und der Abbildung von Zufällen klassifiziert (s. Tabelle 2.1) [Law07, S. 6] [Gut+17, S. 16]. Während statische Simulationen ein System entweder ohne Berücksichtigung der Zeit oder zu einem bestimmten Zeitpunkt betrachten (z. B. Monte-Carlo-Simulation), repräsentieren dynamische Simulationsmodelle das zeitliche Verhalten eines Systems entweder in einer kontinuierlichen oder diskreten Zeitvoranschreitung [MK11, S. 13–14]. Bei kontinuierlichen Simulationen verändert sich der Systemzustand meistens anhand von Differentialgleichungen beständig mit der Zeit [Law07, S. 7]. Zustandsveränderungen, die nur an diskreten Zeitpunkten erfolgen, werden dementsprechend als diskrete Simulationsmodelle klassifiziert [MK11, S. 14]. Wenn ein Systemverhalten durch scheinbar zufällige Ereignisse beeinflusst wird, handelt es sich um stochastische, ansonsten um deterministische Modelle [Law07, S. 6]. Gutenschwager et al. [Gut+17, S. 127] ergänzen, dass die Zufallsvariablen stochastischer Modelle nicht tatsächlich zufällig gewählt werden, sondern durch den Einsatz deterministischer mathematischer Rechenvorschriften erzeugt werden, wodurch beide Klassen gleichermaßen reproduzierbare Ergebnisse liefern. Statische Arten von Simulationsmodellen

Tab. 2.1: Klassifikationskriterien für Simulationsmodelle

Zeitverhalten	Zeitmenge	Abbildung von Zufällen
statisch – dynamisch	kontinuierlich – zeitdiskret	deterministisch – stochastisch

werden im Rahmen dieser Masterarbeit nicht weiter behandelt, weil der Einsatz dieser Modelle keine Relevanz für das Forschungsgebiet besitzt. Auf der Basis der Klassifikationskriterien werden folgende drei dynamische Simulationsmethoden häufig in der Praxis verwendet.

Kontinuierliche Simulationen bilden das Zeitverhalten eines Modells ab, indem sich die Werte der Variablen kontinuierlich mit der Zeit ändern [Law07, S. 109]. Der Verwendungszweck dieser Simulationsart bezieht sich meistens auf chemische, physikalische oder biologische Gesetzmäßigkeiten, dessen Verhalten durch Differentialgleichungen beschrieben werden kann. Ein einfaches Beispiel dieser Simulationsart wäre ein Behälter, der über

einen Zufluss mit Wasser gefüllt und über einen Abfluss geleert wird. Über die Variation der Zu- bzw. Abflüsse und des initialen Wasserstands könnte mit Hilfe einer Simulation der Zeitpunkt in Erfahrung gebracht werden, zu dem der Behälter komplett gefüllt oder entleert ist. Da eine kontinuierliche Simulation bei der Analyse diskreter Materialflüsse innerhalb einer SC hingegen eine untergeordnete Bedeutung besitzt, wird diese im Rahmen der Masterarbeit ebenfalls nicht weiter behandelt [Gut+17].

Diskrete zeitgesteuerte Simulation „Die diskrete zeitgesteuerte Simulation beruht darauf, dass in jedem Simulationsschritt die Simulationszeit um ein vorher festgelegtes konstantes Zeitinkrement Δt erhöht wird“ [MM89, S. 202]. Nach jeder Zeitvoranschreitung folgt dann die Überprüfung und ggf. Veränderung der Zustände. Falls innerhalb eines Intervalls von Δt Ereignisse auftreten, die Zustandsänderungen hervorrufen, werden diese gesammelt und gebündelt am Ende des Intervalls verarbeitet [Gut+17, S. 53]. Bei der Verwendung zeitgesteuerter Simulationen besitzt die Größe des Zeitinkrementes Δt eine wichtige Rolle für die Korrektheit, Genauigkeit und Effizienz des nachgebildeten Systems [MM89, S. 202]. Während aus einem zu klein gewählten Inkrement gerade bei Systemen mit verhältnismäßig langen Totzeiten eine hohe Rechenzeit resultiert, besteht bei zu großen Intervallen die Gefahr von Fehlern durch die verspätete Schaltung der Zustände [Gut+17, S. 53]. Eine angemessene Größe des Zeitinkrementes ist dafür stark von der Anwendung abhängig und kann nach Mattern und Mehl [MM89, S. 202] von Millisekunden bei der Simulation von Chipsätzen bis hin zu mehreren Minuten für Gefechtsfeldsimulationen variieren.

Diskrete ereignisgesteuerte Simulation Eine weitere Art von Simulationsmodellen mit diskreter Zeitvoranschreitung sind die diskreten ereignisgesteuerten oder auch ereignisdiskreten Simulationsmodelle [Gut+17, S. 54]. Die Zustandsänderungen ereignisdiskreter Simulationen werden „über Ereignisse verursacht, die zu einem beliebigen Zeitpunkt eintreten“ und direkt im jeweiligen Moment des Eintrittes verarbeitet werden [Gut+17, S. 54]. Verglichen mit der diskreten zeitgesteuerten Simulation erfolgt die Berechnung neuer Zustände daher nicht in festgelegten Intervallen, sondern nur mit jedem neuen Ereignis [Gut+17, S. 54]. Die Simulationszeit wird dabei immer auf den Zeitpunkt des aktuellsten Ereignisses gesetzt [Gut+17, S. 55]. Daher empfiehlt es sich, Modelle mit einer geringen Anzahl von Ereignissen ereignisdiskret zu simulieren, um die Anzahl von Berechnungen bzw. Rechenzeit einzusparen. In der Praxis werden ereignisdiskrete Modelle meistens im Bereich der Produktion und Logistik für die Simulation von Abläufen oder Anlagen eingesetzt, „die das dynamische Verhalten des Systems unter Verwendung stochastischer Komponenten mit Zustandsänderungen an diskreten Zeitpunkten abbilden“ [MK11, S. 14].

2.1.2 Ablauf einer Simulation

Basierend auf den Begriffen der Simulationstechnik und verschiedenen Arten der Simulationsmodellen gibt der folgende Abschnitt einen groben Überblick für den prinzipiellen Ablauf einer Simulation. Während Abschnitt 2.1.3 detailliert unterschiedliche Ansätze für die Durchführung von Simulationen behandelt, ist der folgende Abschnitt an Abbildung 2.2 der Arbeitsgemeinschaft Simulation (ASIM) angelehnt, weil die beschriebenen Phasen in allen Vorgehensmodellen vertreten sind.

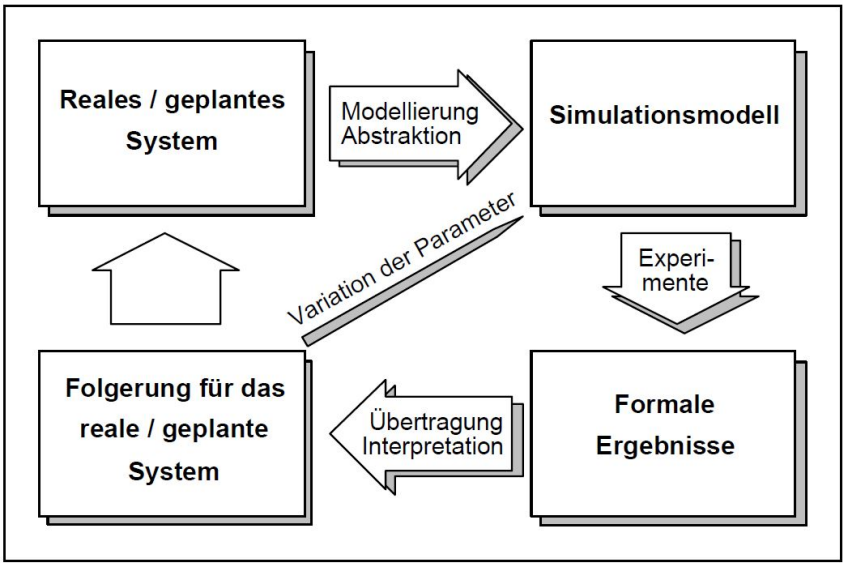

Abb. 2.2: Ablauf einer Simulation [Hrd+97, S. 3]

Modellbildung Für die Durchführung einer Simulation ist immer die Entwicklung eines experimentierfähigen Modells erforderlich, bei der es gilt einen entsprechenden Detaillierungsgrad zu verwenden, welcher ein Maß für die Abstraktion des Modells ist. Zu Beginn der Modellierung gilt es zu überprüfen, welche Systemelemente für die zu untersuchende Fragestellung wirklich relevant für das Modell sind [Gut+17, S. 19]. Der Grundsatz der ASIM lautet, „so abstrakt wie möglich und so detailliert wie nötig“ zu modellieren [Hrd+97, S. 7]. Während untermodellierte Systeme das reale Systemverhalten nicht hinreichend genau wiedergeben, besteht bei zu detaillierten Modellen die Gefahr die Ergebnisse aufgrund vieler Parameter und komplexer Zusammenhänge nicht nachvollziehen zu können [RSW08, S. 70]. Bei der Modellierung ist dabei stets zu überprüfen, ob das Modell das Systemverhalten glaubwürdig genug wiedergibt, um als Entscheidungshilfe dienen zu können [RSW08, S. 1]. Die Überprüfung der Glaubwürdigkeit erfolgt durch die Verifikation & Validierung (V&V) des Modells, die Rabe et al. Rabe, Spieckermann und Wenzel [RSW08, S. 2] als unverzichtbare Bestandteile einer Simulationsstudie bezeichnen. Es steht dabei im Vordergrund, in Erfahrung zu bringen, ob ein Modell einerseits technisch korrekt transformiert wurde und darüber hinaus, ob das Verhalten des abgebildeten Systems

hinreichend genau wiedergegeben wird [RSW08, S. 14–15]. Der Umfang und die Häufigkeit für den Einsatz von V&V-Techniken ist je nach verwendetem Vorgehensmodell einer Simulationsstudie unterschiedlich. Dennoch sollte die Überprüfung der Glaubwürdigkeit des Modells unabhängig von der Vorgehensweise grundsätzlich Vorhanden sein, um Fehler bereits während der Entwicklung frühzeitig zu erkennen und zu beheben und darüber hinaus sicherzustellen, dass aus den Ergebnissen eines Modells keine fehlerhaften Schlüsse gezogen werden [RSW08, S. 3].

Planung und Durchführung von Experimenten Nach der Planung und Implementierung des Modells folgt die Durchführung einer oder mehrerer Simulationsläufe [Ele12, S. 4]. Diese beginnt in der Regel mit der Erstellung eines Experimentenplanes, bei der zwischen What-if- und How-to-achieve-Analysen unterschieden wird [Mer82, S. 21]. What-if-Analysen haben nicht den Zweck eine bestimmte Zielgröße zu erreichen, sondern dienen vielmehr dazu ein besseres Systemverständnis zu erlangen und das Verhalten des realen/geplanten Systems bei der Veränderung der Eingangsgrößen zu prognostizieren. So könnte bspw. überprüft werden, ob eine Arbeitsstation bei der Erhöhung der Eingangstaktzeit nicht überlastet wird. Nach den Aussagen von Gutenschwager et al. [Gut+17, S. 177] ergibt sich die Variation der Parameter für einen Versuchsplan bei What-if-Analysen „klar und direkt“ aus der Aufgabenspezifikation. Nähere Methoden oder Ansätze werden an dieser Stelle jedoch nicht erwähnt.
How-to-achieve-Analysen besitzen hingegen das Ziel, „eine Parameterkonfiguration zu finden, mit der vorab definierte Zielgrößen erreicht werden oder ein möglichst guter Wert für eine Zielgröße erhalten wird“ [Gut+17, S. 177]. Ein großes Problem bei der Planung von Experimenten für How-to-achieve-Analysen besteht in der hohen Anzahl möglicher Parameterkombinationen [Gut+17, S. 177]. Deshalb werden Vorgehensweisen, wie das statische oder dynamische Faktor-Design, zur Reduktion der Simulationsläufe bei einem bestmöglichen Erhalt an Informationen angewendet [Gut+17, S. 177]. In der Terminologie der Experimentenplanung symbolisiert ein Faktor einen Eingangsparameter oder eine strukturelle Annahme des Modells [Law13, S. 629].
Beim **statischen Faktor-Design** werden alle zu untersuchenden Parameterkombinationen bereits vor der Durchführung der Experimente aufgestellt [Gut+17, S. 178]. Law [Law13, S. 632–679] unterscheidet innerhalb des statischen Faktor-Designs drei Ansätze, sortiert nach der Komplexität der Durchführung:

Beim OFAT-Ansatz (One-Factor-at-a-Time) besitzen alle Faktoren eine definierte Anzahl möglicher Ausprägungen, bei der eine Ausprägung als Grundeinstellung des Faktors festgelegt wird. Ausgehend von der Grundeinstellung eines Faktors wird dann jede festgelegte Ausprägung isoliert von den anderen Faktoren variiert [Law13, S. 632].
Law [Law13, S. 632] gibt dafür ein Beispiel, bei dem die zwei Faktoren A und B jeweils die möglichen Ausprägungen „+“ und „-“ besitzen können. Die Grundeinstellung der Ausprägung ist bei beiden Faktoren „-“. Aus einem Versuchsplan dieses Szenarios würden die folgenden Simulationsläufe resultieren:

$$A^-, B^-$$
$$A^+, B^-$$
$$A^-, B^+$$

Die vierte mögliche Kombination des Beispiels wäre ein Simulationslauf der Ausprägungen A^+ und B^+, die allerdings nicht in den Versuchsplan aufgenommen wird, weil diese nur durch die Variation beider Faktoren zustande kommen würde. Der OFAT-Ansatz bietet sich an, wenn die Auswirkungen einzelner Faktoren auf eine oder mehrere Zielgrößen mit wenig Aufwand untersucht werden sollen. Für eine detaillierte Analyse der Auswirkungen ist der OFAT-Ansatz jedoch ungeeignet, weil durch die isolierte Variation keine Wechselwirkungen zwischen den Faktoren gemessen werden können. [Law13, S. 632]

Um deutlich stärker mögliche Interdependenzen zwischen den Faktoren zu messen, eignet sich der deutlich komplexere Ansatz des 2^k-Faktor-Designs, bei dem repräsentierend für jeden Faktor nur die minimale und maximale Ausprägung untersucht werden [Gut+17, S. 178]. Bei k Faktoren mit jeweils zwei Ausprägungen gibt es demnach 2^k mögliche Kombinationen. Die Herausforderung des 2^k-Faktor-Design besteht in der Festlegung der Minimal- bzw. Maximalausprägung der Faktoren [Law13, S. 633]. Für Modelle die mehrere dutzende Parameter besitzen oder falls mehr als zwei Ausprägungen eine Faktors untersucht werden sollen, entstehen in einem Versuchsplan mit dem 2^k-Faktor-Design zu viele Kombinationen, die weiterhin auch als Auslegungspunkte bezeichnet werden [Law13, S. 633]. Zur Reduktion der Auslegungspunkte sollte der dritte Ansatz des 2^{k-p}-Faktor-Design eingesetzt werden.

Im Vergleich zum 2^k-Faktor-Design, auch Vollfaktor-Design genannt, werden im 2^{k-p}-Faktor-Design bzw. Teilfaktor-Design 2^p Auslegungspunkte weniger simuliert als im Vollfaktor-Design [Law13, S. 651]. Für $p = 1$ besitzt der Versuchsplan nur noch halb so viele Auslegungspunkte wie das Vollfaktor-Design und mit jedem weiteren zunehmenden p halbiert sich die Anzahl der Auslegungspunkte weiter [SvH17, S. 30]. Die grundsätzliche Idee teilfaktorieller Versuchspläne besteht darin die Anzahl an Simulationsläufen durch einen möglichst geringen Informationsverlust drastisch zu reduzieren. Dieser Informationsverlust zeigt sich in einer fehlenden Schätzbarkeit von Wechselwirkungen höherer Ordnungen [Mei12, S. 224]. Dadurch, dass in einem vollfaktoriellern Versuchsplan des 2^5-Faktor-Designs alle Haupteffekte bekannt sind, ist ebenfalls die Schätzung von 2-fach, 3-fach oder sogar 4-fach Wechselwirkungen möglich. Da die Erkenntnis über Wechselwirkungen höherer Ordnungen (z. B. 4-fach Wechselwirkung) allerdings kaum von Bedeutung ist, wird bei einem 2^{5-1}-Faktor-Design auf die Schätzung von Wechselwirkungen dritter oder höherer Ordnungen verzichtet, um den Versuchsplan von 32 auf 16 Auslegungspunkte zu reduzieren [Mei12, S. 225]. Der Nachteil dieser Reduktion äußert sich jedoch in der sog. Vermengung. „Mit Vermengung bezeichnet man das Phänomen, dass man Effekte einzeln gar nicht schätzen kann, sondern tatsächlich stets die Summe mehrerer Effekte schätzt“ [Mei12, S. 225]. Wenn sich bei vier möglichen Parametern A, B, C, D die 2-fach Wechselwirkungen AC und BD vermengen, bedeutet es, dass die Ergebnisse der Spaltenprodukte identisch sind. Die Schätzbarkeit der Wechselwirkungen höherer Ordnungen und

Tab. 2.2: Auflösungen von Versuchsplänen [Kle20, S. 144]

Auflösung	Definition	Bewertung
III	Haupteffekt mit 2FWW	kritisch
IV	Haupteffekt mit 3FWW 2FWW mit 2FFW	weniger kritisch
V	Haupteffekt mit 4FWW 2FWW mit 3FFW	unkritisch
VI	Haupteffekt mit 5FWW 2FWW mit 4FFW 3FWW mit 3FWW	unkritisch
2FWW = 2-fach Wechselwirkung, 3FWW = 3-fach Wechselwirkung, usw.		

die Häufigkeit auftretender Vermengungen wird durch die Auflösung klassifiziert, die in Tabelle 2.2 abgebildet ist [Kle20, S. 144]. Je niedriger die Auflösung ist, desto weniger Wechselwirkungen der höheren Ordnungen können geschätzt werden. Im Allgemeinen sind Versuchspläne der Auflösung III kritisch zu bewerten, da 2FWW die Effekte der Faktoren verfälschen können. Versuche der Auflösung IV sind damit verglichen bereits weniger kritisch einzustufen, weil die Vermengungen von Haupteffekten erst mit 3FWW auftritt und diese in der Regel hohe Auswirkung auf die Ergebnisse besitzen. Alle Versuchpläne höherer Auflösungen werden hingegen unkritisch eingeschätzt, weil Vermengungen erst einer hohen Ordnung von Wechselwirkungen auftreten. Für die Erstellung teilfaktorieller Versuchspläne bietet Abbildung 2.3 eine Übersicht für k Faktoren mit m Simulationsläufen bzw. Auslegungspunkten. Verglichen mit dem statischen Faktor-Design zur Auslegung von Versuchsplänen steht beim **dynamischen Faktor-Design** zu Beginn des Experimentes noch nicht fest, wie viele Parameterkonfigurationen simuliert werden [Gut+17, S. 181]. Der Ansatz lautet, einen Simulationslauf so lange mit leicht veränderten Parameterkombinationen zu testen, bis für eine oder mehrere Zielgrößen ein zufriedenstellendes Ergebnis erreicht wurde oder kein besseres Ergebnis gefunden werden kann [Gut+17, S. 181]. Der Einsatz dynamischer Faktor-Designs wird in der Simulationstechnik typischerweise für simulationsgestütze Optimierungen verwendet, deren Ziel es ist, die Parameter eines Modells so einzustellen, dass sich eine möglichst optimale Lösung ergibt [Gut+17, S. 181].

m \ k	3	4	5	6	7	8	9	10	11	12
4	2^{3-1} III									
8	2^3 vollst.	2^{4-1} IV	2^{5-2} III	2^{6-3} III	2^{7-4} III					
16		2^4 vollst.	2^{5-1} V	2^{6-2} IV	2^{7-3} IV	2^{8-4} IV	2^{9-5} III	2^{10-6} III	2^{11-7} III	2^{12-8} III
32			2^5 vollst.	2^{6-1} VI	2^{7-2} IV	2^{8-3} IV	2^{9-4} IV	2^{10-5} IV	2^{11-6} IV	2^{12-7} IV
64				2^6 vollst.	2^{7-1} VII	2^{8-2} V	2^{9-3} IV	2^{10-4} IV	2^{11-5} IV	2^{12-6} IV
128					2^7 vollst.	2^{8-1} VIII	2^{9-2} VI	2^{10-3} V	2^{11-4} V	2^{12-5} IV

Abb. 2.3: Auflösung teilfaktorieller Versuchspläne für k Faktoren und m Auslegungspunkte [Kle20, S. 145]

Interpretation der Ergebnisse und Übertragung von Erkenntnissen Nach der Durchführung der Experimente erfolgt die Auswertung der Ergebnisse und schließlich die Übertragung der Erkenntnisse für das reale oder geplante System. In den meisten Fällen bestehen die Ergebnisse einer Simulationsstudie aus Daten, die sich aus den Eingabeparametern und den daraus resultierenden Zielgrößen zusammensetzen. In einem ersten Schritt werden die Daten durch Verdichtung, Filterung und Verknüpfung in eine andere Form überführt, um Aussagekraft für ursprüngliche Untersuchungsfragen zu erzeugen [Gut+17, S. 193]. Auf der Basis der aufbereiteten Daten gibt die deskriptive Statistik zunächst Hinweise auf Lage- und Streuungsmaße und Korrelationen zwischen Eingangs- und Zielgrößen [Gut+17, S. 194]. Weiterhin ist es hilfreich den zeitlichen Verlauf ausgewählter Zielgrößen zu betrachten, um Schwankungen, Trends oder Ausreißer zu erkennen [Gut+17, S. 195]. Durch die zeitliche Entwicklung der Zielgrößen können auch Zeitpunkte ausfindig gemacht werden, in denen mehrere Abläufe wechselseitig aufeinander warten, die von Rabe et al. [RSW08, S. 195] auch als Deadlock-Situationen benannt werden.
Ein häufig verwendetes Verfahren zur Auswertung von Simulationsstudien ist die Visualisierung, die in der VDI [VDI18, S. 36] als „Umsetzung von Daten oder Abläufen in Bilder in Form von Präsentationsgrafiken, Laufbetrachtungen, Animationen, Monitoring oder grafischem Modellaufbau“ definiert ist. Viele Werkzeuge der deskriptiven Statistik wie z. B. Diagramme, Boxplots und Histogramme, werden innerhalb der Gruppe von Präsentationsgrafiken als eine Art der Visualisierung klassifiziert [VDI18, S. 25]. Der Fokus der visuellen Aufbereitung von Daten liegt auf einer schnellen und präzisen Übermittlung signifikanter Informationen an den Betrachter.
Ein weiteres vielseitig eingesetztes Mittel der Visualisierung ist die Animation, die der „Erzeugung und Präsentation von Bildfolgen“ dient, um Zustandsänderungen und Bewe-

gungen von Modellelementen visuell darzustellen [VDI18, S. 4]. Der VDI [VDI18, S. 4] differenziert zwischen der Online-Animation, die simultan zur Ausführung des Modells stattfindet, oder der Offline-Simulation, bei der die Animation erst im Anschluss anhand der erzeugten Ergebnisdaten erstellt wird.

2.1.3 Vorgehensmodelle der Simulationstechnik

Zur Steigerung des Erfolgs einer Simulationsstudie werden Vorgehensmodelle verwendet, in denen erforderliche Aktivitäten und Produkte (als Ergebnis der Aktivitäten) und auch die Reihenfolge der Bearbeitung definiert sind, um das Projektziel zu erreichen [Ver02, S. 29]. In diesem Kontext definiert Versteegen [Ver02, S. 29] ein Vorgehensmodell als „eine Beschreibung der koordinierten Vorgehensweise bei der Abwicklung eines Vorhabens“. Weiterhin ergänzt er, dass koordinierte Vorgehensmodelle nicht nur einen Plan, sondern auch Erfahrungen benötigen, auf denen dieser Plan basiert [Ver02, S. 29]. Die benötigte Erfahrung zur Entwicklung von Vorgehensmodellen sollte dafür auf vielen Projekten mit unterschiedlichsten Größen aus einer Masse von Unternehmen zurückgreifen, um praktische Gegebenheiten möglichst zuverlässig abzubilden [Ver02, S. 30].
Vorgehensmodelle stammen hauptsächlich aus den Bereich der Softwareentwicklung, bei dem zur Planung, Entwicklung und Einführung von IT-Systemen definierte Phasen und Phasenüberprüfungen, mit möglichen Iterationen zu vorherigen Phasen, befolgt werden [Han10, S. 3–4]. Renommierte Vorgehensmodelle der Softwareentwicklung sind unter anderem das Wasserfall- bzw. V-Modell oder das Scrum-Modell, als Vertreter für eine agile Vorgehensweise [Han10, S. 4, 6, 61]. Analog zu den genannten Modellen werden im folgenden Abschnitt ausgewählte Vorgehensmodelle vorgestellt, die speziell für die Durchführung von Simulationsstudien entwickelt wurden.

Vorgehensmodell nach Law Das Vorgehensmodell von Law (s. Abbildung 2.4) enthält insgesamt zehn sequentielle Phasen zur Beschreibung erforderlicher Aktivitäten. In der ersten Phasen gilt es das Problem zu formulieren und einen Rahmen für alle folgenden Phasen abzugrenzen. Typischerweise werden in der ersten Phase neben der Problemdefinition ebenfalls das Hauptziel und die zur Erfüllung notwendigen spezifischen Fragestellungen erarbeitet [Law13, S. 68]. Weiterhin gilt es, im Projektplan bereits den Modellumfang und die Zielgrößen festzulegen, an denen die Effizienz verschiedener Simulationsläufe gemessen werden kann. Darüber hinaus werden der gesamte Zeitrahmen und die benötigten Ressourcen festgelegt [Law13, S. 68]. Law [Law13, S. 68] erwähnt ebenfalls, dass in der Grobplanung der Studie bereits die Auswahl der verwendeten Software getroffen werden sollte.
In der nächsten Phase gilt es zur Vorbereitung der Modellbildung die Systemstruktur und (falls vorhanden) betriebliche Abläufe zu analysieren [Law13, S. 68]. Aufbauend auf den Erkenntnissen besteht die größte Herausforderung der zweiten Phase in der Wahl eines geeigneten Detaillierungsgrades [Law13, S. 68]. Die Auswahl sollte auf Basis folgender Kriterien erfolgen:

- Projektziel(e)
- Erfolgskriterien
- Verfügbarkeit von Daten
- Computerbeschränkungen
- Meinung der Fachexperten
- Zeit- und Kostenplanung

Das Konzept für das Modell sollte zunächst einfach und übersichtlich starten und nach und nach um weitere notwendige Aspekte ergänzt werden, damit das System nicht detaillierter abgebildet wird, als es die Zielerstellung erfordert [Law13, S. 68]. Neben der Modelldefinition sollten bereits Daten gesammelt werden, aus denen im späteren Verlauf die Werte der Modellparameter spezifiziert werden [Law13, S. 68].
Die folgende Phase gilt der Überprüfung der Annahmen bzgl. der Modelldefinition. Law [Law13, S. 69] empfiehlt dazu einen strukturierten Walkthrough mit allen Projektbeteiligten durchzuführen, damit jeder Fachexperte das Modellkonzept nochmal durchdenkt, bevor es in der vierten Phase realisiert wird.
Diese Realisierung wird von Law [Law13, S. 69] in die Implementierung und die anschließende Verifikation untergliedert. Ersteres bezeichnet die Umsetzung des ursprünglichen Konzeptes in ein reales Modell, entweder durch Programmiersprachen oder mit Hilfe von Simulationssoftware. Programmiersprachen wie C, C++ oder Java bieten dabei den Vorteil, dass sie durch ein breites Anwendungsfeld einen hohen Bekanntheitsgrad besitzen, wodurch qualifiziertes Personal leicht zu finden ist. Weiterhin kann mit sehr geringen Anschaffungskosten eine hohe Programmkontrolle erreicht werden [Law13, S. 69]. Auf der anderen Seite resultiert aus dem Einsatz von Simulationssoftware ein geringer Programmieraufwand durch vorgefertigte Bausteine, die i.d.R. nur noch parametrisiert werden müssen. Dadurch können die Entwicklungsdauer und die damit verbundenen Projektkosten stark reduziert werden. Wie in Abschnitt 2.1.2 bereits erwähnt wird in der anschließenden Verifikation überprüft, ob das reale Modell von der konzeptuellen Beschreibungsart korrekt transformiert wurde. In Phase 5 des Vorgehensmodells werden testweise Simulationsläufe durchgeführt, die in der folgenden sechsten Phase validiert werden. Um zu prüfen, ob das Modell das erwartete Systemverhalten richtig wiedergibt, eignen sich Sensitivitätsanalysen, bei denen der Einfluss der Eingangsgrößen auf die Zielgrößen analysiert wird [Law13, S. 69]. Während das Vorgehensmodell grundsätzlich in sequentielle Phasen eingeteilt ist, sind in Phase 3 oder 6 jedoch iterative Schleifen zur ersten Phase möglich, falls bei der Modelldefinition falsche Modellannahmen getroffen werden oder das reale Modell das geplante Systemverhalten nicht wiedergibt.
Die siebte und achte Phase dienen der Planung und Durchführung von Experimenten unter Berücksichtigung der in Abschnitt 2.1.2 vorgestellten Arten der Versuchsplanung.
Die Analyse und Auswertung der Ergebnisse erfolgt in der vorletzten Phase. Der Fokus der Analyse sollte darauf basieren, die Leistung der Simulationsläufe bestimmter Systemkonfigurationen möglichst aussagekräftig beurteilen und im Anschluss verschiedene Konfigurationen miteinander vergleichen zu können [Law13, S. 70].
In der letzten Phase ist es wichtig das gesamte Projekt verständlich zu dokumentieren und die gewonnen Erkenntnisse in Entscheidungsprozesse zu integrieren [Law13, S. 70].

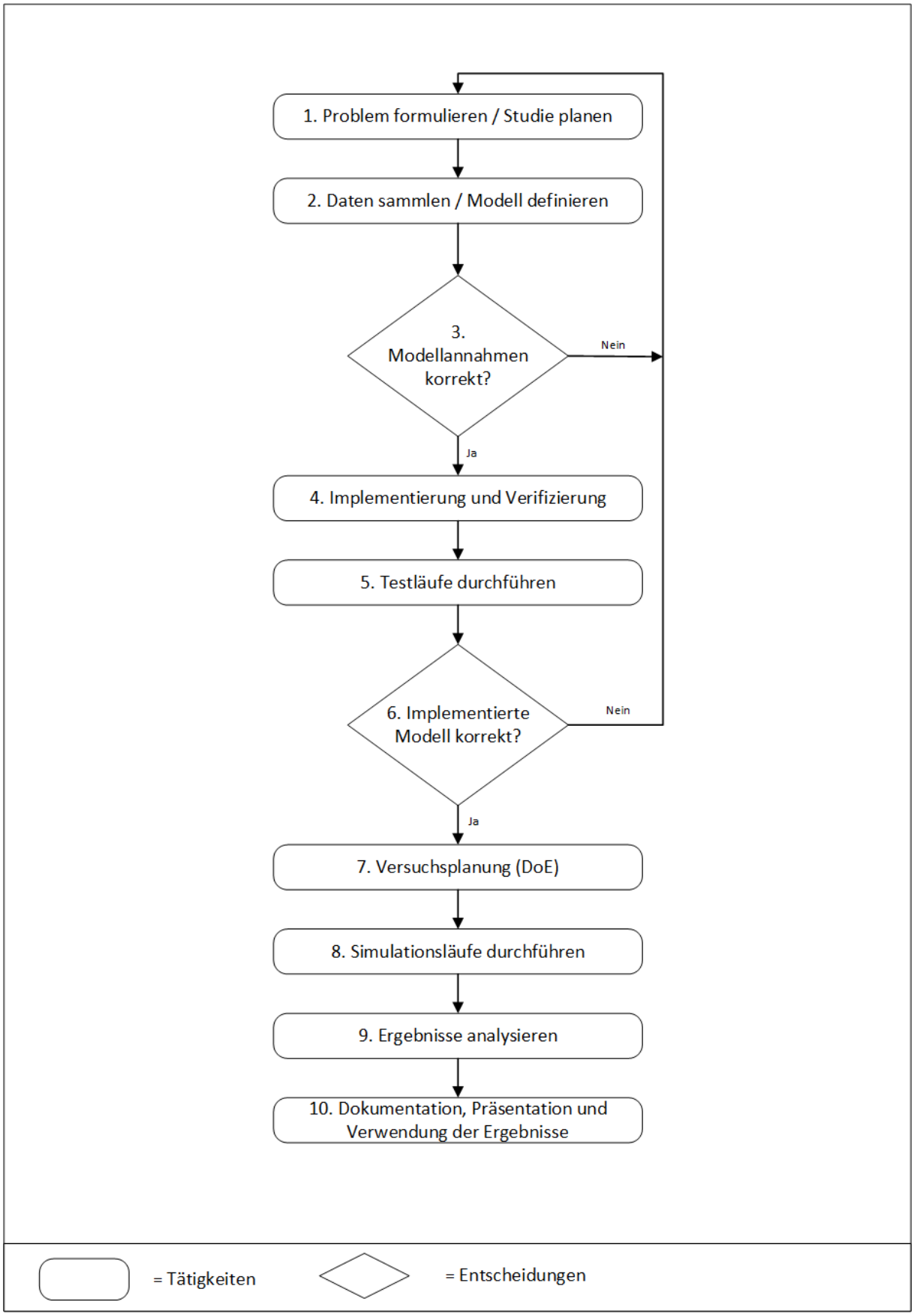

Abb. 2.4: Vorgehensmodell für ereignisdiskrete Simulationen nach Law [Law13, S. 68]

Vorgehensmodell nach Banks Verglichen mit dem Vorgehensmodell von Law gibt es bei dem von Banks viele Parallelen, in der Aufteilung und dem Inhalt einzelner Phasen. Während die V&V-Tätigkeiten im Vorgehensmodell von Law vor und nach Realisierung des Modells angeordnet sind, werden diese Tätigkeiten von Banks gemeinsam im Anschluss an die Modellbildung in der sechsten und siebten Phase ausgeführt [Ban10, S. 37]. Auf diese Weise können vor der Modellübersetzung keine Fehler innerhalb des Modellkonzeptes ausgeschlossen werden. Ein weiterer Unterschied in der Anwendung der V&V besteht in der Stärke der Rückkopplung bei der Feststellung eines Fehlers. Während bei Banks nur iterative Schleifen zurück bis zu den Phasen vorhanden sind, in denen die Fehler typischerweise entstehen, wird bei Law das gesamte Projekt, von der Problemformulierung an, neu durchdacht.
Die Phase für die Planung der Experimente ist in beiden Modellen ähnlich beschrieben, jedoch ergänzt Law nach der Durchführung und Analyse der Experimente eine Tätigkeit zur Überprüfung der Notwendigkeit weiterer Simulationsläufe. Falls die Ergebnisse eines Versuchsplans nicht genügend Aussagekraft zur Beantwortung der Fragestellungen erzeugen können, trägt der Analyst die Entscheidung darüber erneute Experimente durchzuführen oder sogar einen komplett neuen Versuchsplan aufzustellen [Ban10, S. 37].
Wenn keine neuen Experimente erforderlich sind, folgt die Dokumentation und Berichtserstattung der Simulationsstudie in Phase 11. Banks [Ban10, S. 37] unterscheidet zwischen einer ausnahmslos erforderlichen Prozessdokumentation, welche die gesamte Chronik des Projektes mit allen getroffenen Entscheidungen umfasst, und einer Programmdokumentation, die allerdings nur erforderlich ist, wenn verschiedene Analysten das Simulationsprogramm für die Durchführung der Experimente verwenden.
Die letzte Phase der „Implementierung“ beschreibt im Vergleich zu den Vorherigen keine Aktivität, sondern vielmehr einen Zustand, den es zu erreichen gilt. So erwähnt Banks [Ban10, S. 38], dass der Erfolg der Implementierung von der Durchführung der vorherigen elf Phasen abhängig ist. Wenn ein finaler Modellanwender, der in einem Betrieb auch ein Kunde sein kann, über alle Phasen hinweg mit in die Planung und Entwicklung des Modells einbezogen wurde und das Verhalten des Modells und die Effekte zwischen den Eingabe- und Zielgrößen nachvollziehen kann, ist eine erfolgreiche Implementierung bzw. Übergabe gelungen [Ban10, S. 38].

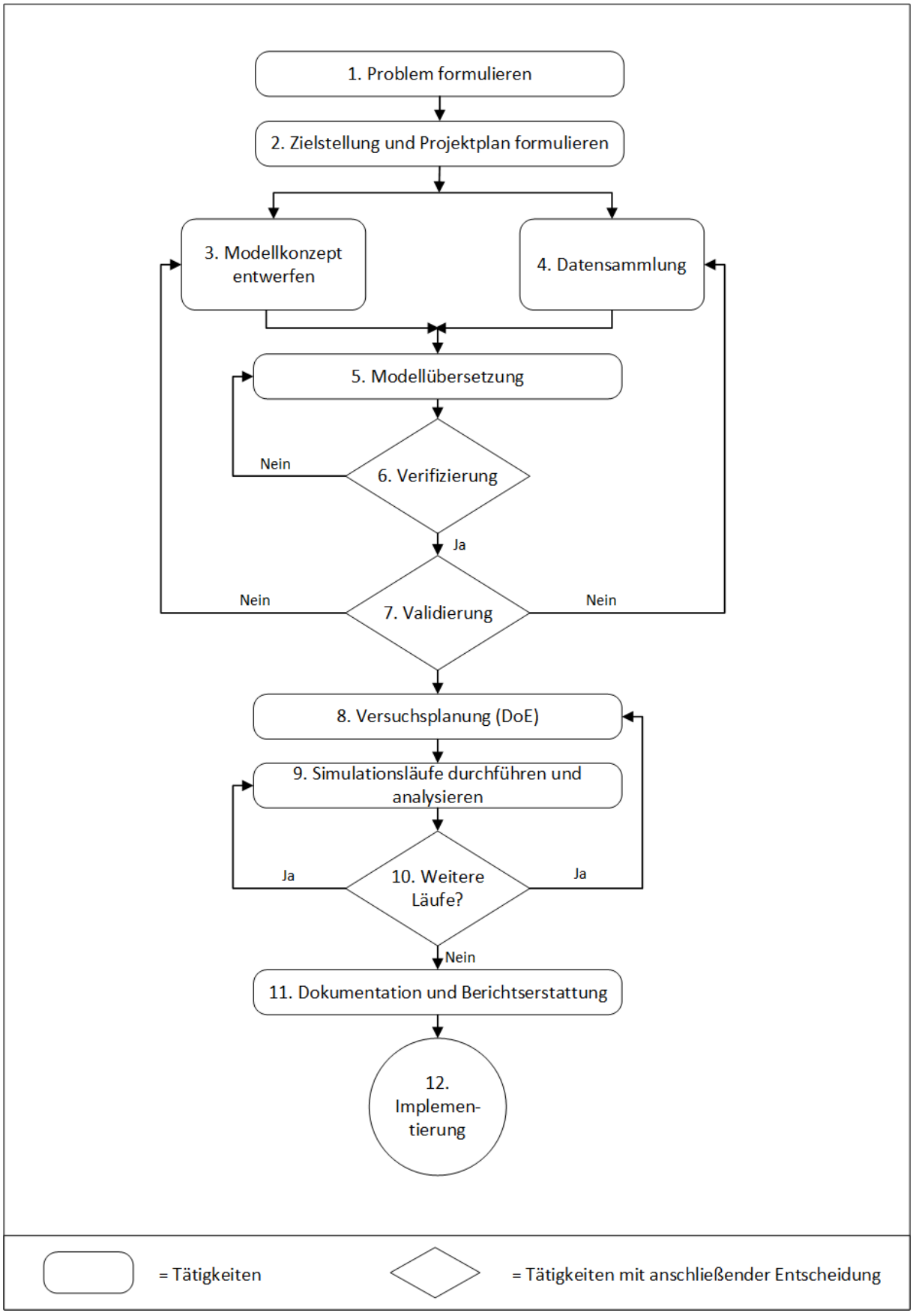

Abb. 2.5: Vorgehensmodell für ereignisdiskrete Simulationen nach Banks [Ban10, S. 35]

Vorgehensmodell nach Sargent Während die Vorgehensmodelle nach Banks und Law eine Simulationsstudie als Ablauf von Tätigkeiten beschreiben, betrachtet Sargent vielmehr die Produkte, die als Resultate einer Tätigkeit entstehen (s. Abbildung 2.6). So definieren Sargent und Mason [SM08, S. 159], dass für eine Simulation die Erstellung folgender drei Produkte erforderlich sind:

- Problemfall (anhand des Systems)
- Konzeptmodell
- Computergestütztes Modell

Die Produkte stehen innerhalb des Vorgehensmodells in Beziehungen zueinander, die durch Tätigkeiten zur Erstellung oder Verifikation und Validierung der Produkte beschrieben werden [SM08, S. 159]. Die Erstellung dieser Produkte wurde von Law und Banks ebenfalls als Element einzelner Phasen bezeichnet. So sind zur Ausarbeitung des *Problemfalls* in dem Vorgehensmodell von Sargent bspw. die Tätigkeiten der Projektplanung, Problembeschreibung und Zieldefinition aus den Phasen von Law und Banks erforderlich. Durch die Systemanalyse und den Tätigkeiten zur Modellbildung, inklusive der Festlegung des Detaillierungsgrades, entsteht aus dem Problemfall schließlich ein theoretisches *Konzeptmodell*, das alle getroffene Annahmen beinhaltet. Durch die Implementierung wird das Konzeptmodell in ein reales *computergestütztes Modell* überführt, woraus aus der Durchführung von Experimenten wiederum Erkenntnisse gewonnen werden, die auf das ursprüngliche *System* übertragen werden.
Eine Besonderheit in der Simulationsbeschreibung nach Sargent und Mason [SM08, S. 159] ist die Anwendung von V&V-Tätigkeiten, die nicht als bestimmte Phase, sondern parallel

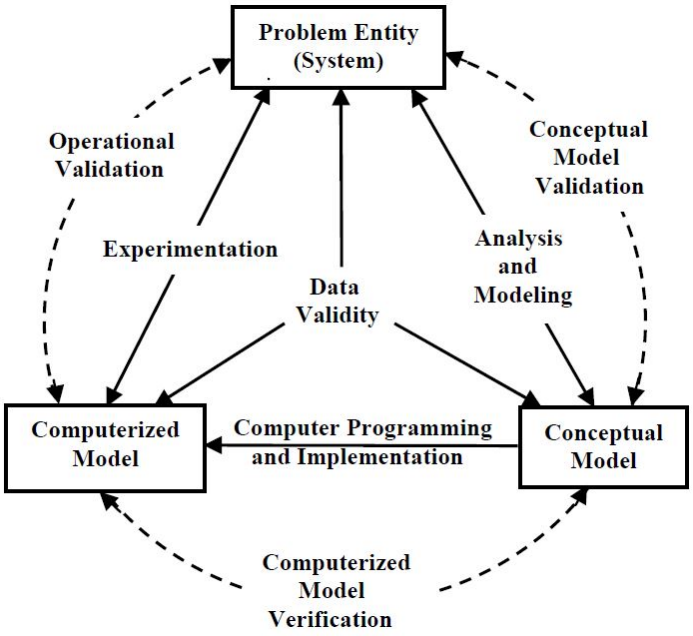

Abb. 2.6: Vorgehensmodell für Simulationen nach Sargent [SM08, S. 159]

zu jeder im Vorgehensmodell vorhandenen Tätigkeit stattfindet. Bei der Systemanalyse und der Modellbildung wird deshalb auch kontinuierlich geprüft, ob das Konzeptmodell als Werkzeug zur Lösung des Problemfalls hinreichend verwendet werden kann. Der Implementierung des Konzeptmodells steht dementsprechend die stetige Verifizierung des computergestützten Modells gegenüber und bei der operationalen Validierung wird gewährleistet, dass die Simulationsergebnisse präzise genug zur Lösung des Problemfalls sind [SM08, S. 159]. Eine weitere beschriebene V&V-Tätigkeit dient der Datenvalidierung, die über alle Aktivitäten gewährleistet, dass die zugrundeliegenden Daten angemessen und richtig sind [SM08, S. 159]. Insgesamt bietet das Vorgehensmodell nach Sargent mehr Freiheiten bei der Gestaltung der Aktivitäten zur Erstellung der beschriebenen Produkte im Vergleich zu Law und Banks.

Vorgehensmodell nach dem VDI In Bezug auf Unternehmen der deutschen Automobilindustrie wie Audi, BMW oder VW werden Simulationsstudien häufig nach dem Vorgehensmodell der ASIM durchgeführt, welches auch in der VDI-Norm 3633 aufgeführt ist (s. Abbildung 2.7)[VDI18, S. 19]. Ähnlich wie auch bei Sargent gliedert sich das Vorgehen in Phasen zur Beschreibung einer Aktivität (runde Klammern) aus denen jeweils ein Produkt bzw. Phasenergebnis entsteht (eckige Symbole). Die V&V des Modells ist keine eigene Phase, sondern ein begleitendes Element für alle Phasenergebnisse [RSW08, S. 38]. Während die Phasen für die *Aufgabendefinition, Systemanalyse, Modellformalisierung, Implementierung* und die *Experimente und Analyse* nacheinander ausgeführt werden, verlaufen jegliche Aufwände zur Datenbeschaffung und -aufbereitung simultan zu den anderen Phasen, müssen allerdings bis zum Ende der Studie abgeschlossen sein [VDI18, S. 33]. Der Grund für die simultane Datenerfassung besteht darin, dass die für die Modellbildung erforderlichen Daten in der Regel nicht die notwendige Vollständigkeit und Konsistenz für die Experimente aufweisen. Dadurch müssen sie erneut einer Plausibilitätskontrolle bzgl. der Systemlast, der Systemstruktur, der funktionalen Abläufe über der Zeit, sowie der hinterlegten Strategie unterzogen werden [VDI18, S. 33]. Während die notwendigen Daten in den bisher vorgestellten Vorgehensmodellen nur als *Systemdaten* oder *Unternehmensdaten* beschrieben wurden, klassifiziert der VDI [VDI18, S. 34] Simulationsdaten in *Systemlastdaten, Organisationsdaten* und *technische Daten*, die in Tabelle 2.3 weiter untergliedert sind. Weiterhin ergänzt der VDI in Bezug auf die Phase der *Systemanalyse* die Vorgehensweisen der Top-down- und Bottom-up-Ansätze [VDI18, S. 23]. Der erste Ansatz beschreibt die Detaillierung des Modells ausgehend vom Ganzen, wohingegen beim Bottom-up-Ansatz zunächst einzelne Elemente der untersten Ebene analysiert und das Gesamtsystem schrittweise durch die Verbindung aller Einzelelemente zusammengesetzt wird [VDI18, S. 23]. Während die Erstellung eines Konzeptmodells auch bei den Vorgehensweisen von Banks und Sargent aufgeführt wird, existiert nach der ASIM noch eine weitere Phase der *Modellformalisierung*, in der als vorbereitenden Schritt für die Implementierung Steuerstrategien formuliert werden [VDI18, S. 31]. Die Implementierung des formalen Modells und die anschließende Durchführung der Experimente und Analyse der Ergebnisse ähneln im Ablauf stark den Vorgehensmodellen von Law, Banks oder Sargent. Aus einem Vergleich aller behandelten Vorgehensmodelle wird die Erkenntnis geschlossen, dass es

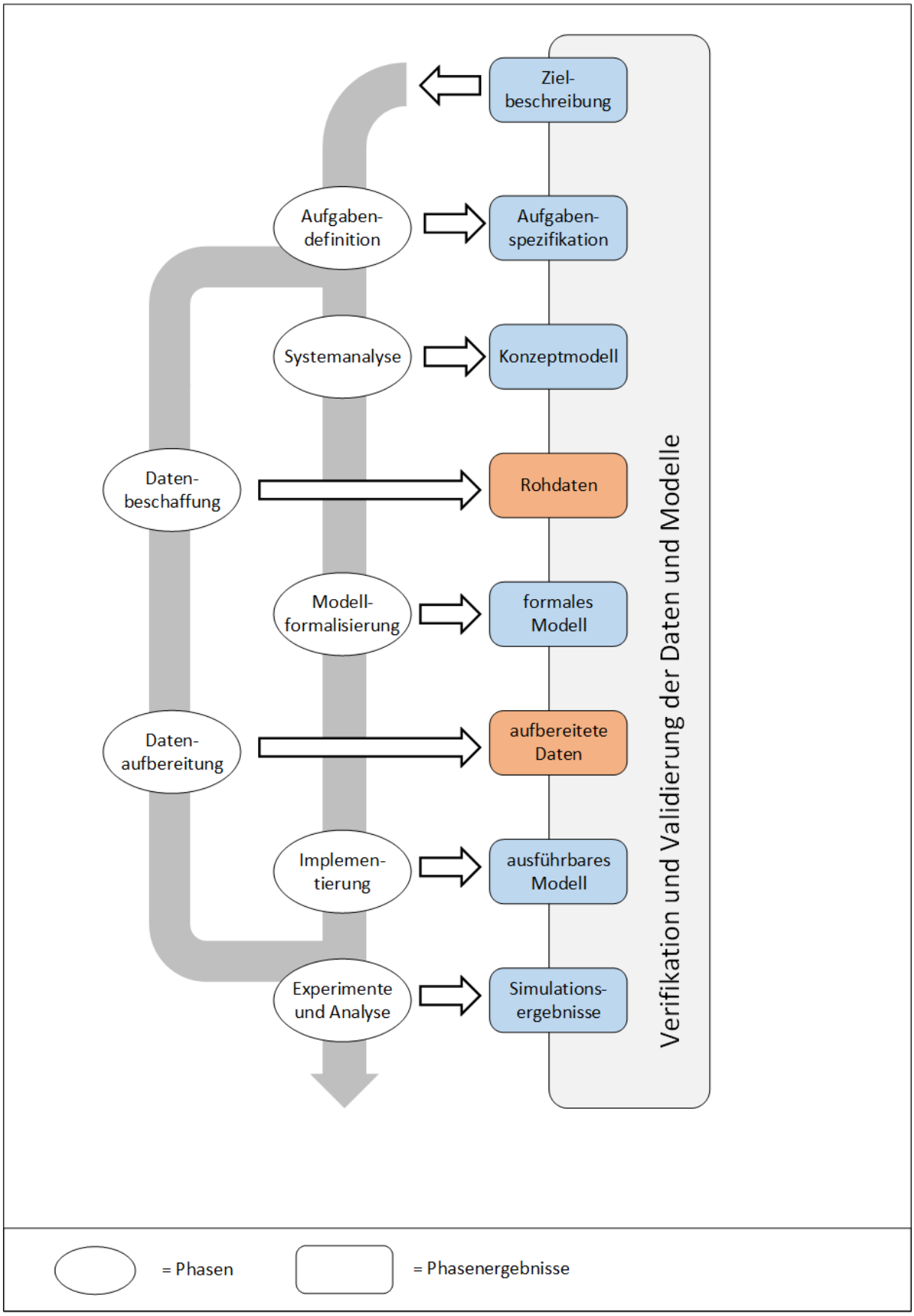

Abb. 2.7: Vorgehensmodell der Arbeitsgemeinschaft Simulation [VDI18, S. 19]

Tab. 2.3: Klassifizierung von Simulationsdaten [VDI18, S. 34]

Art der Daten	Weitere Untergliederung	Beispiel
Systemlastdaten	Auftragseinlastung	Produktionsaufträge Menge Termine
	Produktdaten	Arbeitspläne / Stücklisten
Organisationsdaten	Arbeitszeitorganisation	Pauseregelung Schichtmodelle
	Ressourcenzuordnung	Werker Maschinen Fördermittel
	Ablauforganisation	Strategien Restriktionen Störfallmanagement
Technische Daten	Fabrikstrukturdaten	Layout Fertigungsmittel Transportfunktion
	Fertigungsdaten	Nutzungszeit Leistungsdaten Kapazität
	Materialflussdaten	Fördermittel Nutzungsart Kapazität
	Stördaten	Funktionale Störung Verfügbarkeit

wesentliche Unterschiede in der Anzahl und dem Inhalt einzelner Phasen, dem Umfang und der zeitlichen Einordnung der Datenbeschaffung und der V&V der Simulationsmodelle gibt. Wie aber auch Rabe, Spieckermann und Wenzel [RSW08, S. 29] erwähnen, sind in allen Vorgehensmodellen (mit abweichenden Bezeichnungen und Unterschieden in der Komplexität und dem Umfang der Ausführung) folgende fünf Phasen vorhanden:

- Aufgabenanalyse
- Modellformulierung
- Modellimplementierung
- Modellüberprüfung
- Modellanwendung

2.1.4 Angemessenheit und Potentiale einer Simulation

Simulationstechniken können unter den richtigen Einsatzbedingungen ein enormes Potential entfalten, wodurch neue Systeme (z. B. Fertigungs-, Materialfluss- oder Montagesysteme) effizient geplant und realisiert oder bestehende Systeme durch ein größeres Systemverständnis optimiert werden können [Ban10, S. 23] [Hrd+97, S. 10]. Um dieses bestehende Potential auszuschöpfen zu können, muss der vorliegende Problemfall allerdings einem typischen Einsatzgrund einer Simulation entsprechen, was in der Praxis jedoch nicht immer leicht abzuwägen ist. Banks [Ban10, S. 22] formuliert diesbezüglich elf Einsatzzwecke, für die eine Simulation die geeignete Problemlösungsmethode ist. Die wesentlichen Punkte sind dabei, dass mit dem Simulationsmodell Experimente durchgeführt werden können und Effekte zwischen veränderten Eingabewerten und den Zielgrößen zu erkennen sind [Ban10, S. 22].
Der VDI [VDI14, S. 19] empfiehlt vor der Durchführung einer Simulationsstudie zunächst eine Grundsatzentscheidung zu treffen, bei der zu prüfen ist, ob sich eine Problemstellung anhand bestimmter Gesichtspunkte als simulationswürdig erweist. Dabei sollte das abzubildende System der Problemstellung zunächst eine gewisse Komplexität besitzen, damit keine zeit- und kostenintensive Simulationsstudie durchgeführt wird, wenn auch eine einfache analytische Lösung möglich wäre [VDI14, S. 19]. Typische Eigenschaften komplexer Systeme sind eine Vielzahl von Einflüssen, Abhängigkeiten und nebenläufigen Prozessen, sowie große Datenmengen und eine grundsätzliche Unüberschaulichkeit [VDI14, S. 19]. Es ist allerdings möglich eine bereits bestehende analytische Lösung eines Problems durch eine Simulation zu verifizieren [Ban10, S. 22]. Der VDI [VDI14, S. 20] erwähnt aufgrund des hohen Aufwandes einer Simulationsstudie außerdem, dass sich der Einsatz nur dann als sinnvoll erweist, wenn eine wiederholte Notwendigkeit für die Verwendung des Modells besteht.
Die ASIM beschreibt die Notwendigkeit des Einsatzes einer Simulation speziell für die Produktion und Logistik, „wenn Neuland beschritten wird, die Grenzen analytischer Methoden erreicht sind, komplexe Wirkzusammenhänge die menschliche Vorstellungskraft überfordern, das Experimentieren am realen System nicht möglich bzw. zu kostenintensiv ist und das zeitliche Ablaufverhalten einer Anlage untersucht werden soll“ [Hrd+97, S. 10].
Auf der anderen Seite existieren ebenfalls klare Indizien, wenn sich der Einsatz einer Simulation als ungeeignet erweist. So empfiehlt Banks [Ban10, S. 23] zusätzlich zu den bereits genannten Punkten, Simulationen zu vermeiden, wenn es kostengünstiger ist die Experimente direkt an einem realen System durchzuführen, keine Daten für die Erstellung des Simulationsmodells vorhanden sind oder die Kosten einer Simulationsstudie die möglichen Ersparnisse übersteigen. Letzteres ist dabei schwierig zu beurteilen, da im Rahmen einer Kosten-Nutzen-Analyse die Wirtschaftlichkeit eines Simulationsprojektes abgewägt werden müsste. Während sich die Kosten für die Durchführung einer Simulation normalerweise relativ genau berechnen lassen, ist der Nutzen einer Simulation im Vorfeld jedoch nicht quantifizierbar, weil der Erkenntnisgewinn aus den Ergebnissen anfangs nicht bekannt ist [Gut+17, S. 47].

Bei einer simulationswürdigen Problemstellung bietet eine Simulation folgende Vorteile. Durch die Erstellung des Modells können technische und ablaufbedingte Veränderungen eines realen Systems getestet und bewertet werden ohne den laufenden Betrieb des Systems

zu unterbrechen [Ban10, S. 23] [Ghe08, S. 46]. Auf diese Weise werden kostenaufwändige Veränderungen oder Modifikationen direkt am System vermieden, bei denen der Nutzen nicht gewährleistet ist [BS09, S. 21]. Durch die Durchführung von Experimenten können die Einflüsse bestimmter Parameter auf die Systemleistung und auch Erkenntnisse über Wechselwirkungen zwischen bestimmten Parametern gewonnen werden, wodurch das allgemeine Systemverständis gesteigert wird und Optimierungsansätze bestimmt werden können [Ban10, S. 23]. Eine weitere vorteilhafte Eigenschaft von Simulationsmodellen besteht in der variablen Zuordnung der Länge eines Zeitinkrementes, wodurch Zeit des simulierten Systems komprimiert oder gestreckt werden kann. Grade bei Systemen mit langen Laufzeiten ist dadurch eine große Zeitersparnis möglich [Ban10, S. 23].
Entgegen den genannten Potentialen gibt es allerdings auch Nachteile, die bei einer Simulation zu berücksichtigen sind. So erwähnen Banks und Sokolowski [BS09, S. 21], dass für die Durchführung einer Simulationsstudie eine hohe Qualifikation erforderlich ist. Besonders für die Modellbildung und die Wahl des Detaillierungsgrades ist ein spezielles Training erforderlich, welches größtenteils auf viel Erfahrung beruht [Ban10, S. 24]. So versichert Banks [Ban10, S. 24], dass zwei Modelle des gleichen Systemabbildes unterschiedlicher Entwickler nie komplett identisch funktionieren werden. Zudem sind Simulationsprojekte meistens zeitaufwändig und mit hohen Kosten für qualifizierte Entwickler und erforderliche Software verbunden. Deshalb sollte eine Problemstellung nur mit Hilfe von Simulationen gelöst werden, wenn es wirklich angemessen ist bzw. das Problem simulationswürdig ist.

2.2 Einsatzgebiete für Simulationstechniken in Supply Chains

Eine große Herausforderung für die ganzheitliche Betrachtung und Optimierung einer SC besteht in dem Integrationsgrad der vorliegenden IT-Landschaften mit anderen Akteuren. Selbst wenn das gesamten interorganisationale Datenmanagement eines Unternehmens zentral durch ERP-Systeme (Enterprise-Resource-Planning) realisiert und verwaltet wird, zeigt ein Bericht des Fraunhofer IML, dass sich für eine ganzheitliche Optimierung der Lieferkette im Durchschnitt nur 20 % der erforderlichen Informationen in den eigenen Systemen befinden [Ber14, S. 8]. Klassische ERP-Lösungen und lokale Planungslösungen sind für solche Anwendungszwecke nur im Ansatz oder durch kostenintensive Erweiterungen mit der SC vernetzt [Ber14, S. 8]. Auch wenn deutliche Tendenzen zu integrierten Cloud-Lösungen zu bemerken sind, auf die alle Akteure über serviceorientierte Web-Dienste mittels Internetbrowser zugreifen können, wird der Informationsfluss häufig auch über gewöhnliche Kommunikationswege wie Telefon, Fax oder E-Mail realisiert [Ber14, S. 9]. Um eine SC auf Basis realer Daten zu analysieren bzw. zu optimieren, sind daher innovative integrierte IT-Systeme oder aufwändige Aktivitäten zur Datenerschaffung (bei verteilten Systemen) erforderlich.
Zur Vermeidung hoher Kosten und Aufwände werden in der Praxis simulationsgestützte oder analytische Methoden angewendet. Letzteres eignet sich dabei allerdings nur für triviale Szenarien, weil eine SC einer Vielzahl dynamischer Einflussfaktoren unterliegt, dessen Zusammenhänge meistens zu komplex sind, um sie auf einem analytischen Wege

zu bestimmen [CM11, S. 1]. Simulationsgestütze Methoden erweisen sich hingegen als hilfreiche Werkzeuge, wenn Analysen der realen SC zu teuer, aufwändig oder unpraktisch sind [CM11, S. 1]. Im SCM werden Simulationen daher verwendet, um sowohl strategische als auch taktische und operationale Entscheidungen zu treffen [CM11, S. 4].
Bei einer SC-Simulation bildet das Modell entweder die gesamte oder bestimmte Teile einer SC ab [CM11, S. 1]. Um die Leistungsflüsse des dynamischen Netzwerks möglichst realistisch abzubilden eignen sich grundsätzlich ereignis- oder zeitdiskrete Modelle [CM11, S. 4]. Aus technischer Sicht können sich Simulationsstudien abhängig von dem jeweiligen Simulationstyp und Anwendungsfall, auf die im Folgenden eingegangen wird, in allen anderen Charakteristiken unterscheiden. Kleijnen und Smits [KS03, S. 5–6] beschreiben insgesamt vier Simulationstypen für eine SC-Simulation:

Spreadsheet simulation Wie in beinahe allen Bereichen eines Unternehmens sind Tabellenkalkulationsprogramme wie bspw. Microsoft Excel alltäglich verwendete Mittel für schnelle Analysen überschaubarer Datenmengen. Tabellenkalkulationsprogramme gehören mittlerweile zur Standardsoftware jedes Betriebssystems und werden aufgrund der weiten Verbreitung und der einfachen und intuitiven Bedienung sowohl im privaten als auch industriellen Bereich häufig für Simulationen verwendet [Sei06, S. 1]. Programme wie Microsoft Excel besitzen durch die Gestaltung der grafischen Benutzeroberfläche eine der leichtesten Schnittstellen für den Datenimport und bieten leistungsstarke Werkzeuge für Analysen und statistische Auswertungen, um mathematische und logische Zusammenhänge zwischen Faktoren zu untersuchen [Sei06, S. 1]. Die Vorteile dieser Art von Simulationen sind im Wesentlichen die hohe Verfügbarkeit der Software und die verhältnismäßig geringe erforderliche Qualifikation des Simulationsexperten. Kleijnen und Smits [KS03, S. 5] kritisieren jedoch, dass „Excel"-Simulationen keine ausreichenden Detaillierung abbilden können und daher oftmals zu unrealistisch sind, um komplexe Systeme wie SCs abbilden zu können.

System dynamics (SD) untersuchen die Leistungsflüsse einer SC, um das Systemverständnis über die dynamischen Eigenschaften zu steigern und Synergien zu identifizieren. Der Fokus liegt auf dem Verständnis der Ursachen für ein Systemverhalten. Es werden bspw. die Auswirkungen von Auftragsgrößen, Produktpreisschwankungen, Rationierungen und fehlender Verfügbarkeit auf die Varianz der Nachfrage analysiert, die letztlich zum *Bullwhip-Effekt* führen kann [CM11, S. 6]. SD-Simulationen dienen der Auslegung und Kontrolle von SCs und besitzen den Vorteil, keine detaillierten Informationen oder genaue Daten über die Beziehungen zu anderen Akteuren zu benötigen [CM11, S. 6].

Discrete-event dynamic system (DEDS) sind detaillierter als SD-Simulationen, indem individuelle Ereignisse repräsentiert werden [CM11, S. 6]. Wie bereits in Abschnitt 2.1.1 erläutert, eignen sich ereignisdiskrete Simulationen, um dynamische Systeme mit stochastischen Komponenten abzubilden. Dadurch können Unsicherheiten in Form von zufälligen

Ankunftszeiten von Lieferungen oder unvorhergesehenen Maschinenausfällen simuliert werden [CM11, S. 6]. Zusätzlich zu geplanten Abläufen sind DEDS-Simulationen hilfreich, um das Systemverhalten bei unvorhergesehen Ereignisse zu betrachten und somit Risiken abzuwägen.

Business games Simulationsbasierte Planspiele bieten die Möglichkeit menschliche Entscheidungen in die Simulation einfließen zu lassen. Eine oder mehrere Personen treffen während der Simulation strategische Entscheidungen, die Auswirkungen auf die Simulationsergebnisse haben. Es können also Entscheidungsqualitäten gemessen werden, die aber in erster Linie dazu dienen Entscheidungsträger zu trainieren, anstatt die Erkenntnisse auf reale SCs zu übertragen. Nach Kleijnen und Smits [KS03, S. 5] werden derartige Planspiele zwar in der SC praktiziert, sind aber nicht so stark verbreitet wie die ersten drei Simulationstypen.

Alle vorgestellten Simulationsarten sind vielseitig einsetzbar und können deshalb bei unterschiedlichen Problemstellungen Anwendung finden. Terzi und Cavalieri [TC04] untersuchten in diesem Kontext über 80 SC-Simulationen auf den praktischen Einsatz hin, um Muster in den Problemfällen zu erkennen und empfohlene Vorgehensweisen zu erarbeiten. Die Anwendungsfälle für SC-Simulationen gliedern sich grundsätzlich in die Kategorien *Netzwerkdesign, strategische Entscheidungen* und *Prozessanalysen*.
Simulationen für das *Netzwerkdesign* konzentrieren sich im Wesentlichen auf die Gestaltung der SC durch die Auswahl von Lieferanten, Händlern und Kunden, die durch die Lieferkette versorgt werden sollen [TC04, S. 10]. Ein zentraler Bestandteil ist die Standortplanung neuer Knoten (z. B. Distributionszentren), um den Kundenkreis auszuweiten oder Transportkosten zu minimieren [CM11, S. 4].
Darüber hinaus können Simulationen ebenfalls unterstützend für *strategische Entscheidungen* sein, um die SC auf schnelle Reaktionszeiten, kollaborative Planung oder die Auslagerung von Drittfirmen auszurichten [TC04, S. 10].
Überwiegend werden SC-Simulationen jedoch für die Optimierung der alltäglichen *Prozesse* verwendet, welche für die Leistungsflüsse verantwortlich sind. So empfiehlt sich die Anwendung für Produktionsplanungen und -terminierungen, Bestandsplanungen, Bedarfprognosen, Verkaufplanungen (z. B. durch stochastische Nachfragen) und der Distribution und Transportplanung [TC04, S. 8]. Zusammengefasst beschäftigen sich Prozesssimulationen mit der zeitlichen und geografischen Koordination sämtlicher Materialflüsse eines bestehenden bzw. fertig geplanten Netzwerks unter Berücksichtigung der erforderlichen Informationsflüsse und den daraus resultierenden Finanzflüssen.

Die Analysetechnik der Experimente ist bei allen vorgestellten Simulationstypen und Anwendungsfällen entweder How-to-achieve oder What-if-Analysen mit dem Hintergrund die wirklich kritischen Faktoren zu identifizieren [KS03, S. 6]. Die Versuchsplanung erfolgt meistens nach dem OFAT-Ansatz, um die Sensitivität einzelner Faktoren auf das Ergebnis zu messen [KS03, S. 6]. Große SCs unterliegen aber einer Vielzahl dynamischer Faktoren mit komplexen Wechselwirkungen, wodurch ein sequentieller Analyseansatz einzelner Faktoren

wenig Erfolg verspricht. Selbst triviale Modelle von SCs benötigen als Eingangsfaktoren parametrisierte Produktionszeiten- und raten, Transportdurchsätze und -kapazitäten, Anfangsbestände, Mindestbestände für Bestellungsauslösungen, Bezugsgrößen von Bestellung usw. . Zusätzlich können diese Faktoren in der Praxis für jeden Akteur unterschiedlich sein, wodurch die möglichen Ausgänge eines Szenarios, selbst bei wenigen Faktorstufen, zu hoch ist, um jede Parameterkombination zu simulieren. Im Rahmen von What-if-Analysen mit statischem Faktor-Designs sind 2^k-Faktor-Designs daher ungeeignet für SCs. Nach Kleijnen und Smits [KS03, S. 6] werden in der Praxis stattdessen teilfaktorielle Versuchspläne geringer Auflösungen eingesetzt, um die Anzahl von Läufen zugunsten der Simulationsdauer zu reduzieren. Durch die zunehmende Vermengung von Wechselwirkungen niedriger Ordnungen gehen dadurch aber ebenfalls wertvolle Informationen verloren. Teilfaktorielle Versuchspläne sind zwar eine gute Alternative für Modelle mit vielen Faktoren, aber weniger für komplexe Wechselwirkungen, die bei SCs aber vorhanden sind. In Anbetracht der zukünftig weiter zunehmenden Komplexität und Dynamik von Materialflusssystemen existiert bisher also keine aussichtsreiche Methode, um SCs simulationsbasiert effektiv zu analysieren bzw. zu optimieren. Ein potentieller Ansatz zur Lösung dieses Konfliktes könnte darin bestehen, Data Farming in die Problemdomäne der SC zu integrieren.

3 Anwendung von Data Farming im militärischen Kontext

Die Bezeichnung „Data Farming“ fiel zum ersten Mal im Jahr 1997 und beschreibt anfänglich fundamentale Veränderungen in der Ausführung von Simulationen für Problemdomänen des United States Marine Corps (USMC) [HS16, S. 3]. Der Auslöser für diesen Umbruch war ein Bericht des Data Science Board (DSB), indem die Eignung „traditioneller“ Simulationen für militärische Szenarien in Frage gestellt wurde [BH98, S. 93]. Nach den Aussagen von Brandstein und Horne [BH98, S. 93] waren die damaligen Simulationsmodelle für militärische Szenarien „nicht zukunftsträchtig“ und die Prozesse des USMC befanden sich auf einer „konservativen, lediglich reagierenden“ Basis.
Für militärische Simulationsstudien werden oftmals agentenbasierte Modelle verwendet, bei denen die Aktionen jedes Agenten zweier im Konflikt stehender Einheiten simuliert werden (Rot vs. Blau Szenario). Alle Agenten einer Einheit verfolgen dabei ein zentrales Ziel, können allerdings vollständig autonom mit ihrer Umwelt interagieren, wodurch unterschiedliche Agenten innerhalb der Mikro-Ebene auch unterschiedliche Entscheidungen über ihre nächste Aktion treffen [WR17, S. 13]. Selbst bei einer sehr geringen Anzahl an Agenten trifft jedes Individuum aufgrund stochastischer Parameter nahezu immer eine andere Entscheidung über seine nächste Aktion, wodurch endlos viele Ausgänge einer Simulation möglich sind.
Die von Brandstein und Horne [BH98, S. 93] betitelten „traditionellen“ Simulationen, bei denen durch den Einsatz des statischen bzw. dynamischen Faktor-Designs möglichst wenig Simulationsläufe durchgeführt werden, geben agentenbasierte Modelle daher nur einen kleinen Einblick über die Bandbreite möglicher Ausgänge eines Szenarios. Aus diesem Grund lautete der damalige Ansatz, das Vorgehen bisheriger Simulationsstudien sowohl im Ablauf als auch in der technischen Durchführung so umzugestalten, dass mit dem Einsatz von Hochleistungsrechnung leistungsfähigere Modelle erzeugt werden konnten. Durch die massive Erzeugung großer Datenmengen konnten durch diese Modelle der Einblick auf die Ergebnislandschaft maximiert werden [BH98, S. 93]. Zur Analyse und Auswertung gilt es Werkzeuge einzusetzen, die in der Lage sind in der gesamten Ergebnislandschaft Verteilungen und Ausreißer zu identifizieren, um schließlich Erkenntnisse abzuleiten [Hub+19, S. 1].
Die gesamte abgewandelte Durchführung einer Simulationsstudie wird mit dem Begriff „Data Farming“ (DF) in Verbindung gebracht. Eine genaue Definition des Begriffs ist an dieser Stelle aufgrund unterschiedlicher literarischer Aussagen jedoch unbestimmt, weshalb diese erst nach der Bearbeitung des Themengebietes in Abschnitt 3.3 erfolgt. Eine von Horne formulierte und zunächst zutreffende Beschreibung deklariert DF als eine Komponente, die einen interdisziplinären Ansatz der Simulation, Hochleistungsrechnung und statistischen Analyse zur Untersuchung einer spezifischen Fragestellung verwendet [BH98, S. 95] [HS16]. Die Erstellung des Versuchsplans für DF-Projekte erfolgt dabei immer im Rahmen von What-if-Analysen, um das Verständnis über das zugrundeliegende System möglichst zu maximieren. [HS16, S. 3].

Im folgenden Abschnitt 3.1 werden die wichtigsten Projekte vorgestellt, in denen das DF bis hin zum aktuellen Stand entwickelt wurde. Darauf aufbauend werden in Abschnitt 3.2

ähnliche bzw. gleiche Strukturmuster vergangener DF-Projekte herausgestellt, die selbst als potentielle Phasen eines Vorgehensmodells für die Anwendung von DF innerhalb einer SC dienen oder bei der Gestaltung der Phasen unterstützend wirken können. Im Ausblick auf die praktische Modellentwicklung in Kapitel 5 wird außerdem Bezug auf die Architektur und empfohlenen Hard- und Softwarelösungen von DF-Modellen eingegangen. Eine interessante Anwendungsmöglichkeit besteht darin DF webbasiert und modular als sog. Data Farming Services (DFS) anzubieten. Die Realisierung und Vorteile von DFS werden in Abschnitt 3.2.3 erläutert. Abschließend wird der Begriff Data Farming auf Basis der erarbeiteten Grundlagen präzise definiert und innerhalb einer Simulationsstudie eingeordnet.

3.1 Projekte für den Einsatz von Data Farming

In den folgenden Abschnitten wird die chronologische Entwicklung des DFs erläutert. Die Ursprünge des Forschungsgebietes stammen dabei aus dem USMC. Ab dem Jahr 2010 bis hin zum heutigen Zeitpunkt wird DF auch für zivile Anwendungsfälle im Rahmen der North Atlantic Treaty Organization (NATO) weiterentwickelt.

3.1.1 Erste Entwicklungen innerhalb des Project Albert (1998 - 2006)

Der Beginn der Umstrukturierung US-militärischer Simulationen beschreibt die 1998 gestartete Initiative des USMC mit dem Projektnamen „Project Albert“ [HS16, S. 3]. Der Direktor dieses Projektes war Dr. Gary Horne, der als einer der Initiatoren für die Forschung im Themengebiet des DFs gilt [Law05]. Zunächst startete Project Albert als nationales Projekt, wurde aber im Jahr 2002 durch den Beitritt Australiens und später Neuseelands und Deuschlands international ausgeweitet [HS16, S. 1]. Innerhalb des Project Albert wurden eine Reihe von Modellen, Werkzeugen und wissenschaftlichen Methoden verwendet, um Fragen im Interesse militärischer Entscheidungsträger zu erkunden [HJ02, S. 4]. Die Mission bestand darin, den Einfluss immaterieller Faktoren auf den Entscheidungsprozess von Befehlshabern während einer Kampfsituation zu untersuchen [HJ02, S. 4]. Dabei standen zum Beispiel folgende Fragestellungen im Vordergrund:

- Wann ist eine zentrale/dezentrale Befehlsführung von Vorteil bzw. Nachteil?
- In welchen Situationen sind taktische Manöver gegenüber dem Einsatz von Feuerkraft vorzuziehen?
- Welche Rolle spielen Vertrauen, Motivation oder weitere immatrielle Faktoren in einem Kampfgebiet?

[HL01, S. 2]

Für die Analyse dieser Fragestellungen wurden ausschließlich agentenbasierte Modelle verwendet, in denen immer zwei im Konflikt stehende Einheiten simuliert wurden. Die Modelle zeichneten sich darin aus, dass sie möglichst klein und kompakt gehalten und sehr abstrakt konzipiert wurden, damit viele Simulationsläufe in einer kurzen Zeit durchgeführt werden konnten [Mas08, S. 1]. Im Rahmen des Project Albert wurden mehrere bestehende agentenbasierte Modelle um die DF-Komponente ergänzt oder komplett neu entwickelt. Im Folgenden werden die Modelle ISAAC und MANA als bedeutende Meilensteinen des Project Albert für die Anwendung von DF vorgestellt.

Irreducible Semi-Autonomous Adaptive Combat (ISAAC) Bereits vor dem Beginn des Project Albert entwickelt das USMC das ISAAC-Modell. ISAAC ist im Ursprung ein rudimentäres Microsoft Disk Operating System (MS-DOS) basiertes zeitdiskretes Modell, das in C programmiert wurde [HL01, S. 132] [Ila97, S. 16].
Das dargestellte Szenario des Modells ist eine klassisches Capture the Flag (CTF) zwischen Team Rot und Blau auf einem individuell festgelegten zweidimensionalen Terrain [Ila97, S. 26]. Jeder Agent verfolgt dafür das übergeordnete Ziel die Flagge des Gegners in die eigene Basis zu befördern, besitzt allerdings eigene physische und persönliche Eigenschaften, mit denen er dezentral agieren kann [HL01, S. 182]. Diese Eigenschaften werden zu Beginn eines Simulationslaufs initial parametrisiert, können jedoch innerhalb der Simulation durch äußere Umstände sowohl positiv als auch negativ beeinflusst werden. In einem Handbuch zum ISAAC-Modell beschreibt Ilachinski [Ila97, S. 30] Gewichtungsvektoren, wie z. B. die Anzahl befreundeter/gegnerischer überlebender Agenten oder die Seite des Flaggenbesitzes, die Einflüsse auf die offensive oder defensive Haltung einzelner Agenten besitzen. Das übergeordnete Ziel des Modells besteht lautet, Verhaltensmuster auf der Mikroebene zu identifizieren und für reale Szenarien abzuleiten [Ila97, S. 7]. Bei der Modellierung wurde ein relativ hoher Abstraktionsgrad verwendet, um den Fokus lediglich auf das Verhalten der Agenten zu legen [HL01, S. 156]. Die verwendete Ausrüstung und unterschiedliche Eigenschaften von Schusswaffen blieben aus diesem Grund unberücksichtigt [Ila97, S. 6].

Im ursprünglichen Entwicklungsstand von ISAAC erfolgte die Ausführung des Modells lokal, wodurch für die Berechnung nur die Leistung der internen Hardware zur Verfügung stand. Für die Zuweisung von Parameterwerten des Szenarios, der Agenten und des Terrains wurden dat-Dateien verwendet, die zu Beginn eines Simulationslaufes ausgelesen wurden [Ila97, S. 54]. Die Planung von Experimenten mit einem Experimentenplan, analog zu Abschnitt 2.1.2, war vor den Entwicklungen im Rahmen des Project Albert nicht möglich, weil die Parametrisierung und Ausführung einzelner Läufe ausschließlich manuell erfolgte. Aus den Bemühungen innerhalb des Projektes wurde das ursprüngliche ISAAC-Modell um die DF-Komponente modifiziert. In Kooperation mit dem Maui High Performance Computing Center (MHPCC) wurde eine webbasierte Schnittstelle entwickelt, über die festgelegte Parametervarianten mit dem Einsatz von Hochleistungsrechnung simuliert und anschließend als Ausgabe bereitgestellt wurden [HL01, S. 132]. Nach den Aussagen von Horne und Leanardi [HL01, S. 132–133] gelang es auf diese Weise Millionen von Simulationsläufen über das MHPCC zu berechnen. Zu diesem frühen Zeitpunkt der DF-

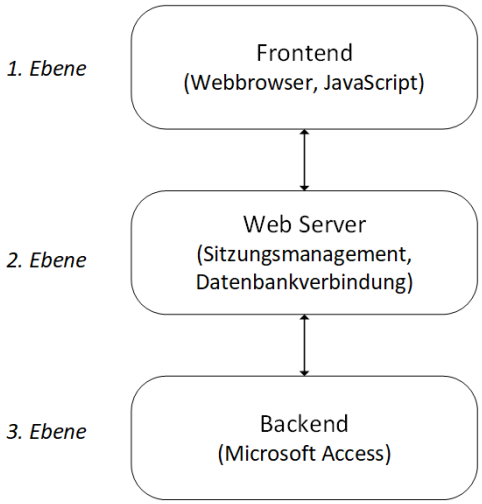

Abb. 3.1: Architektur der ISAAC Datenanalyse [HL01, S. 141]

Entwicklung konzentrierte sich das Projektteam stärker auf die Erzeugung von Daten als auf das Management oder die Analyse der Datenmengen, wodurch als Rückgabe der Simulation zunächst für jeden Lauf einfache ASCII-Dateien im einem Ordnerverzeichnis gespeichert wurden [HL01, S. 133–134].
Die Auswertung der ASCII-Dateien erfolgte daher entkoppelt im Anschluss an die Erzeugung der Daten, durch die Verwendung einer dreistufigen Frontend-Backend-Architektur (s. Abbildung 3.1) [HL01, S. 138]. Auf der ersten Ebene werden Benutzereingaben auf einer Weboberfläche validiert und verarbeitet und daraus entstehende Datenbanktransaktionen an den Web-Server der zweiten Ebene weitergeleitet. Dieser stellt eine Verbindung zu einer Microsoft Access Datenbank her, die als Speicherort für die Simulationsdaten dient. Durch manuelle Interaktionen können die ASCII-Dateien der auszuwertenden Simulationsläufe über eine Weboberfläche ausgewählt werden. Diese werden ausgelesen und in der Datenbank abgespeichert. Durch SQL-Abfragen (Struktured Query Language) der Datenbank werden die Rohdaten in einer transformierten Form für den Anwender in der Weboberfläche dargestellt.
Resümierend war es möglich, das ISAAC-Modell soweit zu modifizieren, dass massive generiert werden konnten. Die zielgerichtete Analyse und Auswertung der Daten stellte sich jedoch als eine große Herausforderung dar. Horne und Leanardi [HL01, S. 142] beschreiben den damaligen Entwicklungsstand als unvollendet und betonen, dass sowohl das verwendete Modell als auch die anschließende Analyse der erzeugten Daten stärker in einen DF-Prozess zu integrieren sind.

Map-Arare Non-Uniform Automata (MANA) Nach dem Beitritt Neuseelands im Jahr 2002 entwickelte die neuseeländische Defence Technology Agency (DTA) den sog. MANA, der eine Klasse von Modellen für unterschiedliche Simulationsszenarien verwendet [And13, S. 2]. Ähnlich wie auch bei ISAAC handelt es sich um abstrakte agantenbasierte

Modelle, die sämtliche physikalische Bedingungen zugunsten einer schnellen Laufzeit unberücksichtigt lassen [And13, S. 2]. Die DTA vertritt den Standpunkt, die menschliche Natur nicht durch eine greifbare mathematische Größe beschreiben zu können. Dem Anwender sollte stattdessen die Möglichkeit gegeben werden, Persönlichkeitstypen der Agenten zu definieren, indem bpsw. Bewegungswünsche der Agenten stärker gewichtet werden können [HJ03, S. 16]. Die verwendeten Modelle des MANA besaßen zwar einen geringeren Detaillierungsgrad im Simulationsablauf, boten allerdings wesentlich mehr Einstellung als ISAAC. So stehen Optionen für folgende Kategorien zu Auswahl:

- Map/Terrain
- Greifbare Eigenschaften der Agenten (Geschwindkeit, Ausdauer, Trägheit)
- Persönlichkeiten der Agenten
- Schusswaffenoptionen
- Sensorische Fähigkeiten

[And13, S. 2–4]

In den durch den MANA simulierten Szenarien konnten neben roten und blauen Agenten auch neutrale Agenten parametrisiert werden, wodurch auch Rettungs- und Versorgungsszenarien für Zivilisten innerhalb eines feindlichen Gebietes simuliert werden konnten. Während des 6. internationalen Project Albert Workshops simulierte eine logistische Arbeitsgruppe auf diese Weise ein Szenario, indem ein Konvoi in einem städtischen Gebiet eine Gruppe von Zivilisten mit Vorräten versorgt [HJ03, S. 15–17]. Im Versuchsplan des Szenarios wurden vier verschiedene Parameter mit jeweils vier Werten variiert. Jeder Lauf wurde aufgrund des stochastischen Modelldesigns 16.385 mal wiederholt, wodurch insgesamt 4,2 Millionen Versuchsläufe durchgeführt wurden [HJ03, S. 18]. Die Berechnung erfolgt ebenfalls über das MHPCC, welches die Simulationsergebnisse für anschließende Auswertungen zur Verfügung stellte. Die verwendete Architektur zur Simulation des Szenarios und die Methode der Ergebnisauswertung geht nicht aus dem Jahresbericht des Maneuver Warfare Science 2003, dem Entwicklungsfortschritt des Project Albert, hervor. Das aufgestellte Konzept, die Auswertungen und die Simulationsläufe innerhalb eines Prozesses zu etablieren, wurde zu diesem Zeitpunkt noch nicht realisiert, weil die Auswertung der Ergebnisse wie auch bei ISAAC von der Experimentendurchführung separiert wurde.

Project Albert beschreibt im Bereich des DFs den Ursprung der Entwicklung. Aufgrund der rasanten internationalen Ausbreitung des Projektes war bereits damals ein allgemeines Interesse für das Forschungsgebiet vertreten. Letztlich konnten durch das Projekt zwar die Potentiale der DF-Komponente aufgedeckt werden, es fehlten für die Modelle wie bei ISAAC und MANA allerdings zunächst eine präzise Beschreibung methodischer Vorgehensweisen. Horne und Meyer [HM05, S. 2] formulierten für dieses Anliegen die sog. Data-Farming-Schleife, um das Vorgehen einer DF-Simulationsstudie auf der Metaebene zu beschreiben. Dieser Ansatz zur Etablierung des DFs ist eines der bedeutendsten Ziele des Project Albert gewesen. Auch wenn die Data-Farming-Schleife ein grobes Konzept für den Aufbau und die Anwendung der Komponente definierte, bot sie dennoch viel

Interpretationsspielraum für die Gestaltung detaillierter Arbeitspakete. In Abschnitt 3.2 wird näherer Bezug auf die strukturierte Anwendung der DF-Komponente genommen. Zusätzlich widerlegte Dr. Gary Horne aus den Erkenntnissen des Projektes die ursprüngliche Annahme, jedes mögliche Szenario eines agentenbasierten Modells betrachten zu können. Aufgrund der verwendeten stochastischen Parameter zur Simulation individueller Bewegungsmuster gibt es endlos viele Möglichkeiten für den Verlauf eines Szenarios, sodass eine reale Landschaft niemals in voller Form simuliert werden kann [Law05]. Für zukünftige Bestrebungen korrigiert Horne seine These daher wie folgt: „You can not really predict anything, but if you look at enough possibilities, you can begin to understand“ [Law05]. Das Project Albert endete 2006 aufgrund der auslaufenden finanziellen Unterstützung vom USMC. Das Forschungsgebiet wurde aufgrund des allgemeinen Interesses aber weiter verfolgt und im Rahmen jährlicher internationaler Workshops wurden neue Entwicklungen und Erfolge präsentiert. Aus diesen Anstrengungen entstanden innerhalb der NATO die Studien MSG-088, MSG-124 und MSG-155 der Modeling & Simulation Group (MSG), die prägend für den aktuellen Stand des Forschungsgebietes sind.

3.1.2 Fortschritte innerhalb der NATO (2010-2017)

Den Beginn der Weiterentwicklung von DF innerhalb der NATO beschreibt die Studie MSG-088 von 2010 bis 2013 [Reu+18, S. 5]. Das Ziel war es, die DF-Komponente durch die Unterstützung der NATO zu standardisieren. In einem NATO-Bericht des MSG-088 wird das Problem geschildert, dass sich die Vorgehensweisen und Architekturen der Studien je nach Nation unterscheiden, wodurch Herausforderungen für den Austausch internationaler Projektteams entstanden [NAT14, S. 51]. Für die Lösung dieses Problems fokussierte sich die MSG auf die einheitliche Definition von Begriffen und die Standardisierung von Vorgehensweisen und Architekturen für Simulationsstudien [NAT14, S. 51]. Als Resultat dieser Arrangements gelang es der MSG eine Simulationsstudie mit dem Einsatz von DF in folgende sechs Bereiche zu gliedern:

1. Rapid Scenario Prototyping
2. Model Developement
3. Design of Experiments
4. High Performance Computing
5. Analyse and Visualization
6. Collaboration

[HS16, S. 14]

Jede Phase wird dabei klar von den anderen abgegrenzt und es werden sowohl Software- als auch Hardwareempfehlungen für die technische Realisierung geäußert. Basierend auf Hornes Aussage, dass durch DF zwar eine große Ergebnislandschaft generiert werden soll, es allerdings nicht möglich ist, jedes mögliche Szenario zu untersuchen, wurden gerade im Bereich der Versuchsplanung auch alternative Designs in Betracht gezogen,

die nicht jede Parameterkombination simulieren [NAT14, S. 63]. Auch für die beiden Kernkomponenten des DFs, der Hochleistungsrechnung und der anschließenden Analyse der erzeugten Daten, werden Anforderungen und Realisierungsmöglichkeiten geäußert [NAT14, S. 85–93]. Neben den sechs Bereichen werden auch explizite Schritte für die Durchführung von Simulationsstudien unter der Verwendung der DF-Komponente beschrieben und darüber hinaus auch visuell in einem Flussdiagramm veranschaulicht [NAT14, S. 60]. Eine detaillierte Erarbeitung von Strukturen und Vorgehensweisen für die systematische Durchführung von DF-Studien erfolgt in Abschnitt 3.2.
Die hauptsächlichen Entwicklungen zielten insofern darauf ab eine einheitliche Basis zu schaffen, um Data Farming zukünftig als Prozess zu etablieren. Neben den strukturellen Fortschritten gab es jedoch auch thematische Weiterentwicklungen. Statt agentenbasierte Modelle und die DF-Komponente im militärischen Kontext ausschließlich für Konflikte und Gefechtssituationen zu erforschen, wurden auch Potentiale für die Katastrophenhilfe erkannt [Reu+18, S. 5]. Dafür wurden Szenarien untersucht in denen das Militär bspw. im Falle eines Erdbebens möglichst effektiv logistische Versorgungsnetzwerke erstellt und Evakuierungsketten bildet [Reu+18, S. 5]. Durch die Simulation sollten Engpässe identifiziert und Möglichkeiten zur Vermeidung gefunden werden [Reu+18, S. 5]. Die darauf folgende Studie MSG-124 beschreibt die DF-Fortschritte zwischen 2013 und 2017 mit dem übergeordneten Bestreben die DF-Komponente in die NATO-Entscheidungsprozesse zu integrieren [Reu+18, S. 7]. Dafür verwendet die Studie als Basis das aufgestellte DF-Konzept der MSG-088 [Reu+18, S. 7].

Die MSG entwickelte das sog. Data Farming decision support Tool for Operation Planning (DFTOP) als unterstützendes Tool für NATO-Entscheidungsprozesse, das sich selbst durch eine enge Verknüpfung mit der NATO-Prozesslandschaft auszeichnet [NAT18, S. 43]. DFTOP ist kein Modell und beschreibt auch kein spezielles Szenario, sondern bietet vielmehr die Ansätze einer Plattform, um DF modular für den Benutzer anzubieten. Der große Nachteil bisheriger Studien bestand immer darin, dass in jeder Phase das gesamte Know-How aller Projektbeteiligten erforderlich war. Die Idee des DFTOP-Tools besteht deshalb in der Erstellung von Modulen und der Einbindung dieser in Workflows [NAT18, S. 53]. Zusätzlich wird zwischen den Benutzerrollen Datentechniker, Datenanalyst und Entscheidungsträger unterschieden [NAT18, S. 53]. Während die Datentechniker für die Entwicklung der Module für die Analyse und Visualisierung zuständig sind, können Entscheidungsträger auf einer höheren Ebene agieren und die vorgefertigten Module zur Erstellung von Workflows verwenden. Zum einen werden dadurch Zuständigkeiten klar zugewiesen und zum anderen können erstellte Module und Workflows für unterschiedliche DF-Projekte verwendet werden, wodurch eine konstante Qualität der Ergebnisse gewährleistet wird [NAT18, S. 53].

Weiterhin wurden im Laufe der MSG-124 das Data-farmable Agent-based Cyber Defense Assessment Model (DACDAM) entwickelt, welches die DF-Komponente und agentenbasierte Modellierung verwendet, um Hackerangriffe auf Netzwerke zu simulieren, und Fortschritte im Bereich der Cyberabwehr zu gewährleisten [Reu+18, S. 7]. Das hauptsächliche Ziel war es den Entscheidungsträgern aufzuzeigen, welche Protokolle, Topologien und

Konfigurationen die sichersten Netzwerke erzeugen [NAT18, S. 17].
Das DACDAM beinhaltet eine definierte Anzahl an Routern, Servern, Subnetzen und Clients, die mit Sensoren ausgestattet sind, um Hackerangriffe zu erkennen. Die Hacker verfolgen verschiedene Strategien in der Art und dem Ausmaß ihrer Angriffe. Je nach Anzahl der detektierten Angriffe ist der Systemadministrator in der Lage einzelne Subnetze oder das gesamte Netz abzuschalten, um weitere Schäden möglichst zu reduzieren. [Reu+18, S. 10] Die Besonderheit des DACDAM besteht darin, dass es als erstes Modell mit dem DFTOP verknüpft angewendet wurde und sich diese Kombination insofern als geeignet erwies, dass sie als Entscheidungshilfe für die NATO im Bereich der Cyberverteidigung eingesetzt wird [NAT18, S. 134].

3.1.3 Aktueller Stand der Technik

Seit 2017 beschreiben die Entwicklungen der MSG in der Studie MSG-155 den heutigen Stand der Technik [Reu+18, S. 7]. Die gegenwärtige Studie baut die Idee der Modularisierung von DF-Projekten mit dem Ziel weiter aus, die gesamte Modellierung und Simulation als webbasierten Service anzubieten (Modeling & Simulation as a Service (MSaaS)) [Reu+18, S. 7]. Im Rahmen dieser Entwicklung gilt es die DF-Komponente ebenfalls modular zu gestalten, woraus der Begriff der DFS entsteht [Hub+19, S. 2]. Der tiefere Sinn sowohl die Simulation als auch das DF als Web-Service anzubieten besteht in der Entkopplung der technischen Realisierung von DF mit den jeweils unterschiedlichen szenariospezifischen Modellen. DFS sind ein ausgereiftes Netz von Werkzeugen, um DF in der NATO flächendeckend zu etablieren und eine einfache und intuitive Steuerung der Simulation für alle Anwender zu gewährleisten [Reu+18, S. 7]. In Abschnitt 3.2.3 wird näherer Bezug zu den verschiedenen Arten und Architekturen zu DFS genommen.

Ein weiterer interessanter Aspekt lautet, DF-Projekte mit dem Einsatz künstlicher Intelligenz (KI) zu verknüpfen. Projekte dieser Art werden in der NATO nicht von der MSG betreut, sondern seitens des technischen Teams für Informationssystemtechnologien, und finden daher parallel zur MSG-155 Studie statt [Reu+18]. So könnten bspw. neuronale Netze dazu beitragen, im Team Rot gegen Blau Szenario eine lernende Komponente einzubinden, wodurch sich die Agenten der Einheiten aus den Erkenntnissen vergangener Experimente stetig verbessern [Reu+18, S. 14].
Darüber hinaus wird ebenfalls daran gearbeitet durch KI manuelle Eingriffe zwischen der Erzeugung und der Auswertung der Daten zu automatisieren [Reu+18, S. 14]. Seit dem Project Albert wird die strikte Trennung der Datenerzeugung und der anschließenden Analyse und Visualisierung zwar kritisiert, aber auch durch die Verwendung von DFS werden diese Phasen sequenziell und nicht iterativ innerhalb der Experimente durchgeführt. Die Ergebnisse eines Simulationslaufes haben daher keine Auswirkung auf die Parametervariation des nächsten Laufes, wie es zum Beispiel beim dynamischen Faktor-Design der Fall ist. Reus et al. [Reu+18, S. 14] beschreiben dahingehend Möglichkeiten, um KI für automatische Analysen zu verwenden, die während eines Experimentes Auswirkungen auf die Startwerte des nächsten Laufes haben.

3.2 Strukturen von Data Farming Modellen

Wie aus den Entwicklungen des USMC und der NATO hervorgeht ist DF ein Themengebiet an dem mittlerweile über zwei Jahrzente praxisbezogen geforscht wird. Innerhalb dieser Zeit wurden für DF-Projekte Systematiken für die Durchführung einer Simulationsstudie und Strukturen zur technischen Gestaltung von DF-Modellen entwickelt, die sich stets an neuen technischen Möglichkeiten orientieren. Im folgenden Kapitel werden die bekannten Strukturen zur Durchführung von DF-Simulationsstudien als Leitbild vorgestellt, um bestimmte Elemente auf die Problemdomäne von SCs übertragen zu können.

3.2.1 Die Data Farming Schleife

Die ersten Strukturen zur Durchführung einer DF-Studie wurden gegen des Project Albert auf der Basis der Best Practice aus Simulationsstudien mit Modellen wie dem ISAAC-Modell oder dem MANA formuliert. Nach Aussagen von Horne und Meyer [HM05, S. 2] existierten bis zu diesem Zeitpunkt keine klar definierten Standards für DF Systeme, worauf sie DF im Jahr 2005 erstmals als iterativen Prozess deklarierten, der durch die Data-Farming-Schleife beschrieben wird (s. Abbildung 3.2). Die gesamte Schleife untergliedert sich dabei wiederum in die zwei kleineren, ebenfalls iterativen Schleifen, der *Experiment*

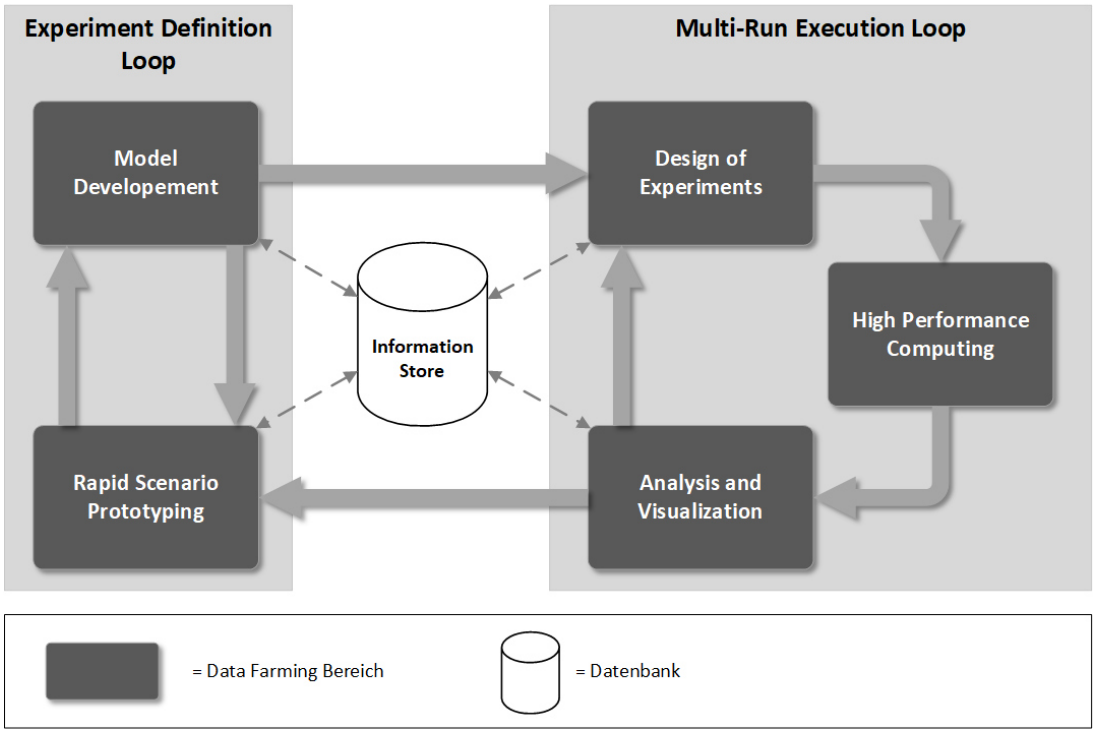

Abb. 3.2: Data Farming als iterativer Prozess (in Anlehnung an [HM05, S. 2])

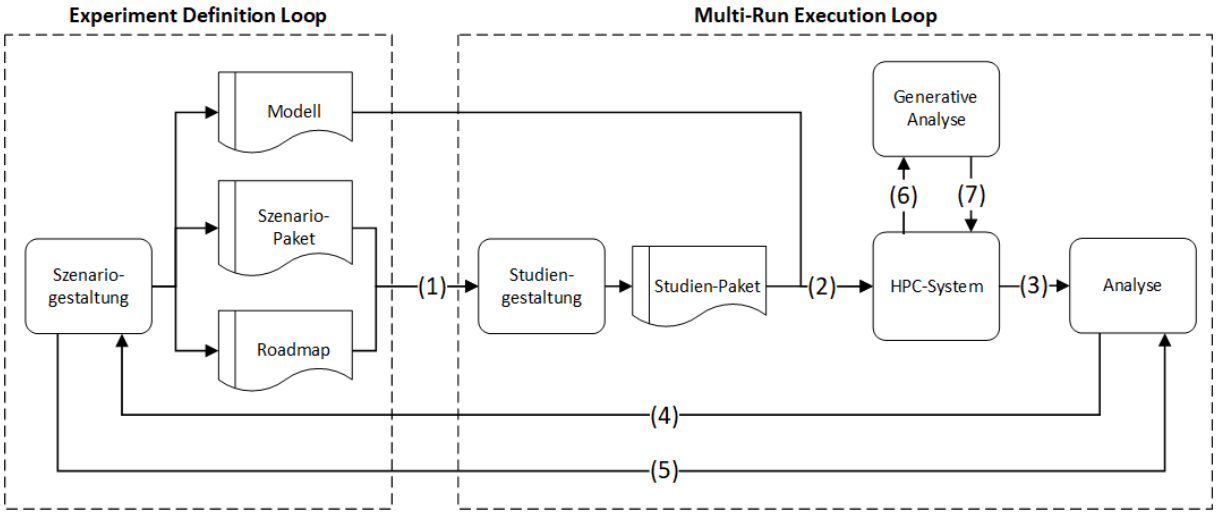

Abb. 3.3: Komponenten und Datenflüssen des Data Farming Prozesses (in Anlehnung an [HM05, S. 2])

Definition Loop und *Multi-Run Execution Loop* [HM05, S. 1]. Ersteres beschreibt die Entwicklung und Spezifikation des Modells auf die zu beantwortende Fragestellung, letzteres die Experimentenplanung, und -durchführung mit dem Einsatz von Hochleistungsrechnung und die anschließende Analyse und Visualisierung der Ergebnisse.
Zwischen den Schleifen gibt es einen Speicher, indem alle notwendigen Informationen der ersten Schleife abgelegt und für die Durchführung der Experimente ausgelesen werden. Für die Schnittstellengestaltung zum Speicher existieren keine Restriktionen. Während der Informationsfluss zwischen der *Experiment Definition Loop* und der *Multi-Run Execution Loop* anfänglich noch über einfache ASCII- oder XML-Dateien stattgefunden hat, werden für komplexere Projekte leicht zugängliche On-Premises- bzw. Cloud-Datenbanken empfohlen [HM05, S. 3].
Jeder einzelne Baustein der Data-Farming-Schleife beschreibt dabei einen der sechs definierten DF-Bereiche im Rahmen der MSG-088 Studie. Zum Zeitpunkt der Veröffentlichung der Data-Farming-Schleife im Laufe des Project Albert existierte noch keine detaillierte Aufgliederung über erforderliche Aktivitäten und Produkte innerhalb der Bereiche, wodurch diese bis zur genaueren Beschreibung in Abschnitt 3.2.2 zunächst als Blackbox betrachtet werden. Während Abbildung 3.2 ein Ansatz zur Illustration des DF-Prozesses ist, stellen Horne und Meyer [HM05, S. 5] in Abbildung 3.3 ebenfalls Komponenten und erforderliche Datenflüsse zur Realisierung des Prozesses dar. Die hauptsächliche Aufgabe der *Experiment Definition Loop* ist die Szenariogestaltung und besteht darin folgende drei Objekte zu erstellen bzw. zu generieren, mit denen eine auf hochleistungsbasierende Experimentendurchführung grundsätzlich erst ermöglicht wird (Abbildung 3.3 - (1)):

- Ein ausführbares Modell
- Eine Szenario-Paket
- Eine Roadmap

[HM05, S. 3]

Zusätzlich zu der Modelldefinition aus Kapitel 2 fügen Horne und Meyer [HM05, S. 3] speziell für den DF-Einsatz hinzu, dass ein Modell eine Verbindung aus Software und Hardware ist, die auf HPC-Knoten bzw. -Cluster (High-Performance-Computing) ausführbar ist und Ergebnisse über alle Simulationsläufe in Form von Daten generiert. Das Szenario-Paket kann eine XML-Datei (Extensible-Markup-Language) sein oder aus anderen Dateiformaten bestehen und enthält alle globalen Parameter des Szenarios Horne und Meyer [HM05, S. 3]. Gerade in der Problemdomäne militärischer agentenbasierter Modelle gibt es immer ein definiertes Terrain für ein Team Rot gegen Blau Szenario, bei dem die Eigenschaften, die Größe und ggf. Hindernisse genau im Szenario-Paket definiert sind. Die Roadmap ist eine Datei mit der genauen Beschreibung der im Szenario-Paket enthaltenen Parameter. Darüber hinaus werden auch die Attribute beschrieben, über die in der Experimentendurchführung Daten erzeugt werden [HM05, S. 3].

In der *Multi-Run Execution Loop* wird das Szenario-Paket und die Roadmap eingelesen und mit weiteren notwendigen Informationen, wie z. B. Benutzereingaben für das Faktor-Design der Experimentenplanung, wird daraus ein Studien-Paket generiert (s. Abbildung 3.3 - (1)) [HM05, S. 4]. Das Studien-Paket enthält somit eine Beschreibung des gesamte Szenarios, den Aufbau der Experimentenplanung mit den verwendeten Parametern und den Variationsbereichen und ggf. einen Algorithmus zur Erzeugung zufälliger Startwerte für stochastische Parameter [HM05, S. 4]. Horne und Meyer [HM05, S. 4] erwähnen, dass die Generierung des Studien-Pakets oftmals automatisch durch Algorithmen erfolgt, aber auch manuell implementiert werden kann. Wichtig ist dabei nur, dass alle erforderlichen Daten innerhalb einer Datei gebündelt werden bzw. von der Hochleistungskomponente leicht von einem Speicherort abgerufen werden können (s. Abbildung 3.3 - (2)).
Die benötigte Hardware für Hochleitungsberechnungen stammt in seltensten Fällen aus eigenen Ressourcen, weshalb die Lösung der Aufgaben entweder auf Supercomputer oder sog. HPC-Cluster außerhalb des eigenen Netzwerks verlagert wird [NAT14, S. 85]. Das Prinzip dahinter lautet, alle notwendigen Informationen für die Simulation in einem Studien-Paket bereitzustellen, um externe HPC-Hardware mit der Durchführung der Simulationsläufe oder des gesamten Experimentes zu beauftragen. Die Rückgabe der Simulationsergebnisse erfolgt je nach Projekt entweder über Dateien oder über direkte Einträge in einer Datenbank seitens der HPC-Hardware [HM05, S. 5].
Im Anschluss wird die gesamte Landschaft der erzeugten Daten analysiert. Dazu können verschiedene statistische Werkzeuge verwendet werden, die entweder auf die Datenquelle der Simulationsdaten zugreifen oder diese durch einen Datei-Import erhalten (s. Abbildung 3.3 - (3)) [HM05, S. 5].
Es ist auch möglich Simulationsläufe mit Ausreißern oder anderweitig interessanten Ergeb-

nissen nochmals in einer Online-Animation visualisiert nachzuvollziehen, um eine größere Einsicht über die Entstehung der Zielgrößen zu erlangen. Horne und Meyer [HM05, S. 5] beschreiben Möglichkeiten, um entsprechende Startparameter für das Modell manuell einzustellen und in einem grafischen Modus ohne die Hochleistungskomponente auszuführen (s. Abbildung 3.3 - (4)). Viele Modelle besitzen dafür unterschiedliche Ausführungsmodi. Das ISAAC-Modell besitzt einen interaktiven Modus mit einer Visualisierung, in der selbst während der Modellausführung Änderungen an Laufzeitparametern zulässig sind und einen Datenerhebungsmodus, für die gezielte Erzeugung von Daten vieler Simulationsläufe [Ila97, S. 39]. Die Ergebnisdaten können sich je nach Ausführungsmodus unterscheiden, wodurch aus einem interaktiven Modus meistens detaillierte Ergebnisse resultieren, da die Laufzeitperformance in diesem Fall nebensächlich ist (s. Abbildung 3.3 - (5)).
Weiterhin beschreiben Horne und Meyer [HM05, S. 5] die Komponente der generativen Analyse als einen Algorithmus, der als Eingabe die gleichen Ergebnisse wie die Analyse verwendet und bis zur Erfüllung eines Abbruchkriteriums automatisch neue Studien-Pakete für weitere Simulationsläufe erstellt (s. Abbildung 3.3 - (6 & 7)). Das Abbruchkriterium kann eine maximale Anzahl an Iterationen, eine definierte Anzahl an Ergebnissen oder die bestimmte Ausprägung einer Zielgröße sein [HM05, S. 5].
Wie auch innerhalb der iterativen *Experiment Definition Loop* und *Multi-Run Execution Loop* gibt es auch zwischen den Schleifen so lange erneute Rückkopplungen, bis die zu untersuchenden Fragestellungen mit der Analyse der Ergebnisse beantwortet werden kann (s. Abbildung 3.2). Dafür sind während den Iterationen sowohl Veränderungen am Szenario als auch am Modell möglich.

3.2.2 Die sechs Bereiche

Im Rahmen der Studie MSG-088 wurde die Grundlage des DF-Konzeptes erarbeitet, wie es von dem USMC und der NATO heute noch verwendet wird. Das größte Ergebnis dieser Studie ist eine genaue Definition der sechs DF-Bereiche, die im Folgenden nacheinander beschreiben werden.

1. Rapid Scenario Prototyping Die ersten beiden DF-Bereiche *Rapid Scenario Prototyping* und *Model Development* sind die beiden Elemente der iterativen *Experiment Definition Loop*, wodurch diese „Hand in Hand“ arbeiten und daher nur schwer voneinander abgegrenzt werden können [HS16, S. 4].
Grundsätzlich ist ein Szenario in diesem Kontext als eine präzise inhaltliche Beschreibung des vorliegenden Sachverhalts zu verstehen, während das Modell die erforderliche technische Komponente zur Ausführung und Untersuchung des Szenarios ist. Im Beispiel militärischer Simulationen ermöglicht ein agentenbasiertes Modell die technischen Voraussetzungen, damit sich einzelne Agenten in einem definierten Terrain eigenständig fortbewegen können, um gemeinsam eine übergeordnete Zielstellung zu erfüllen. Die gesamten Eingangs- und Zielgrößen, die Anzahl der Agenten und unterschiedlicher Einheiten und das übergeordnete Ziel werden innerhalb des ersten DF-Bereichs durch das Szenario definiert. Mit einem

Modell können daher unterschiedliche Szenarien untersucht werden. Der wichtigste Schritt besteht darin, die Fragestellung der Studie so zu formulieren, dass sie mittels Simulation beantwortet werden kann [NAT14, S. 29]. Darüber hinaus sind messbare Zielgrößen zu definieren und falls erforderlich, sollten in dieser frühen Phase bereits benötigte Daten für die Studie identifiziert und gesammelt werden [NAT14, S. 29–31]. Das Ziel des *Rapid Scenario Prototypings* besteht darin, alle Aspekte des Szenarios in das Modell zu implementieren [NAT14, S. 29]. In vielen Projekten des USMC und der NATO wurden bereits vorhandene Modelle lediglich an das neue Szenario angepasst. Falls jedoch die Entwicklung eines komplett neuen Modells erforderlich ist werden innerhalb des *Rapid Szenario Prototypings* erstmals nur wichtige Rahmenbedingungen getroffen. Die Implementierung des Szenarios erfolgt in diesem Fall erst im Anschluss an die Modellentwicklung, wodurch die iterative Gestaltung dieser beiden Bereiche schlüssig ist. In Abbildung 3.4 sind die detaillierten Tätigkeiten des *Rapid Szenario Prototypings* in Anlehnung auf dem Abschlussbericht der MSG-088 Studie in einem UML-Aktivitätsdiagramm (Unified Modeling Language) dargestellt, worin gut zu erkennen ist, dass der Bereich *Model Developement* wie ein Wrapper vom *Rapid Szenario Prototyping* umhüllt ist. Begleitend zu der Szenariogestaltung ist die Erstellung einer detaillierte Szenarienbeschreibung erforderlich, die alle relevanten Informationen zur realisierten Ausführung des Szenarios mit dem Modell enthält [NAT14, S. 33]. Die Szenariobeschreibung wird im Bericht der NATO als lebendes Dokument bezeichnet und ist ein Meilenstein der ersten iterativen Schleife, wodurch eine strikte Versionskontrolle empfohlen wird [NAT14, S. 31]. Zusätzlich ist zum Abschluss der ersten Schleife ebenfalls die Dokumentation des sog. Basisszenarios mit der Beschreibung der verwendeten Eingangsparameter während der Implementierung erforderlich [NAT14, S. 33].

Zur Realisierung einer „schnellen“ Szenariengestaltung beschreiben Horne und Meyer [HM05, S. 5] für den militärischen Bereich die Verwendung GUI-basierter Terraineditoren (Grapical user interface). Dadurch kann das Terrain eines Szenarios leicht in einer visualisierten Darstellung erstellt bzw. angepasst werden und über einen Export direkt ein neues Szenario-Paket mit den veränderten Metadaten des Terrains generiert werden.

Die Zuständigkeit für den erfolgreichen Abschluss des *Rapid Szenario Prototypings* obliegt einer intensiven Zusammenarbeit zwischen dem Analyseteam, den Modellexperten und Fachexperten, die für die inhaltliche Formulierung des Szenarios verantwortlich sind.

Abschließend wird im NATO-Bericht eine Checkliste zur Verfügung gestellt, deren Punkte alle innerhalb des ersten DF-Bereichs abgeschlossen sein müssen:

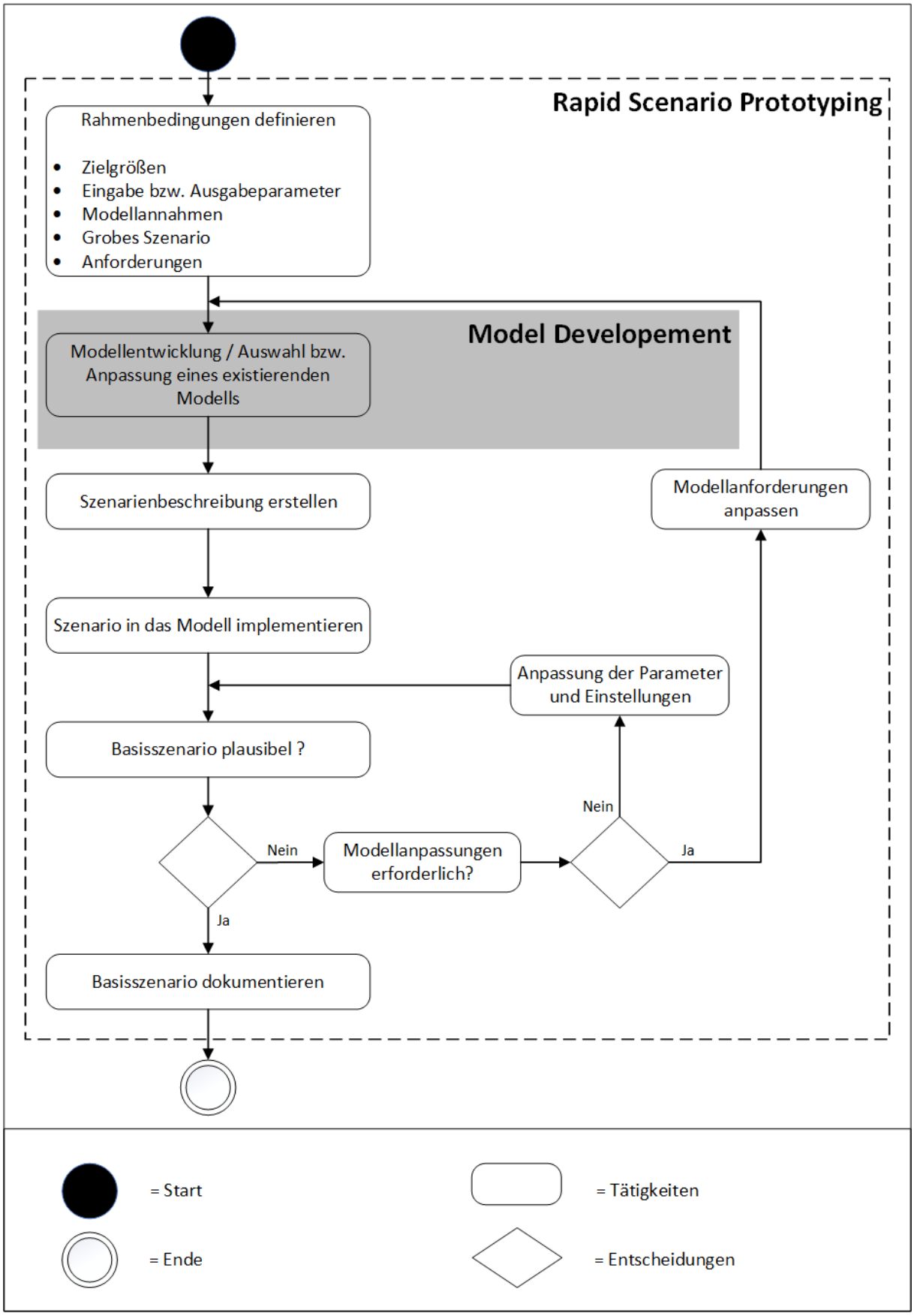

Abb. 3.4: Iterative Vorgehensweise der Experiment Definition Loop (in Anlehnung an [NAT14, S. 30])

1. Sind die Analysefragen klar definiert?
2. Wurden Zielgrößen, Eingangs- und Ausgangsparameter dokumentiert?
3. Sind Szenarioideen oder Einstellungen dokumentiert?
4. Ist ein ausgewählte Modell verfügbar?
5. Ist für das ausgewählte Modell eine Simulation verfügbar?
6. Ist das Analyseteam für das Rapid Szenario Prototyping verantwortlich?
7. Sind alle Szenarioinformationen in einer versionierten Szenariobeschreibung dokumentiert?
8. Sind die benötigten Daten verfügbar?
9. Sind Fachexperten des Szenarios verfügbar?
10. Ist ein Modellexperte für die Implementierung des Szenarios verfügbar?
11. Ist das Basisszenario plausibel?
12. Existiert eine Dokumentation des Basisszenarios?

[NAT14, S. 35]

2. Model Developement Die Modellentwicklung beschreibt die Realisierung der zuvor formulierten Rahmenbedingungen unter Berücksichtigung verwendeter Software, der verfeinerten Funktionsweise des Modells mit wichtigen Algorithmen und der Beschreibung erzeugter Ausgaben. Horne und Leanardi [HL01, S. 1] charakterisieren DF-Modelle anhand folgender vier Eigenschaften:

- Transparenz
- Geschwindigkeit
- Adaptierbarkeit
- Anwenderfreundlichkeit

Zur Erfüllung der Charakteristiken erfolgt die Modellierung im Allgemeinen auf einer sehr hohen Abstraktionsebene [Law05]. Der Grad der Abstraktion wird dafür bereits im ersten Bereich des *Rapid Szenario Prototypings* mit dem entsprechenden Bezug zu den Analysefragen festgelegt. In der Modellentwicklung gilt es, den geplanten Abstraktionsgrad/Detaillierungsgrad möglichst ohne Abweichungen zu realisieren.
Stark abstrahierte Modelle besitzen den Vorteil, viele reale Gegebenheiten nicht zu berücksichtigen, wodurch während der Ausführung verhältnismäßig wenige Bedingungen abgefragt werden müssen und komplexe Algorithmen vermieden werden können. In Bezug auf die schnelle Ausführbarkeit erfolgt die Umsetzung der Modelle durch Programmiersprachen, da die Implementierung im Vergleich zu moderner Simulationssoftware besser auf das behandelte Szenario zugeschnitten werden kann [Ila97, S. 35]. Hinsichtlich diesen Aspektes wurden die entwickelten Modelle während der MSG-Studien ausschließlich in Programmiersprachen wie Java, C++, Python oder Delphi implementiert [NAT14, S. 47]. Simulationssprachen besitzen zwar vorgefertigte Bausteine, in denen aber viele Bedingungen geprüft werden, die für das behandelte Szenario oftmals nicht von Interesse sind. Dies hätte eine negative Auswirkung auf die Laufzeit des Modells.
Im Vergleich zu den bekannten Simulationsstudien aus Kapitel 2 sollte die Funktionalität eines DF-Modells auch losgelöst von der Operationsumgebung, wie zum Beispiel User-Interfaces sein, damit die eigentliche Simulation unter Verwendung von HPC-Komponenten extern ausgelagert werden kann [NAT14, S. 41]. Dies bedeutet nicht, dass das Modell ohne zusätzliche Benutzereingabe auskommen muss, sondern lediglich, dass keine direkten Referenzen auf die Eingabefelder bestehen. Das gesamte Modell muss wie eine modulare Funktion betrachtet werden, die theoretisch von jeder Stelle im Quellcode mit den entsprechenden Eingabeparametern aufgerufen werden kann.
Der bedeutendste Unterschied zu klassischen Simulationen besteht allerdings nicht in der Modularität, sondern vielmehr in der Automatisierung. Im NATO-Bericht wird ausdrücklich erwähnt, dass DF-Modelle dazu in der Lage sein müssen die Simulationsläufe kompletter Versuchspläne ohne menschliche Eingriffe (eng. *human-in-the-loop*) reproduzierbar auszuführen [NAT14, S. 50]. Mit einer manuellen Experimentenausführung ist die massive Erzeugung von Daten in einem zufriedenstellenden Zeitraum nicht möglich.

Die Entwicklung von DF-Modellen erfolgt in erster Linie durch die Modellexperten, die jedoch permanent von den Fachexperten unterstützt werden sollen, um die geplanten inhaltlichen Aspekte erfolgreich umzusetzen [NAT14, S. 40].

3. Design of Experiments Die Versuchsplanung ist eine äußerst wichtige Komponente für das DF, um die mögliche Einsicht auf die Ergebnisse effektiv zu begrenzen [San10]. Paradoxerweise ist diese Methode essenziell, obwohl sie genau das Gegenteil zu dem beschreibt, was DF eigentlich bewirken soll. Nämlich die Sicht auf die mögliche Ergebnislandschaft zu erweitern.
Das Ziel des DFs besteht allerdings nicht nur in der massiven Erzeugung von Daten, sondern darüber hinaus sollen unter den aufgenommen Daten möglichst aller Parameterkombinationen, die wenigen Läufe ausfindig gemacht werden, die ein optimales Systemverhalten prognostizieren. Im Umkehrschluss bedeutet dies, das der größte Teil der erzeugten Daten nur erforderlich ist, um entsprechende Parameterkombinationen für interessante Erkenntnisse ausschließen zu können. Durch die richtigen Versuchspläne können diese nicht signifikanten Simulationsläufe nicht mit absoluter, aber mit hoher Wahrscheinlichkeit bereits im Voraus ausgeschlossen werden, wodurch weniger Parameterkombinationen simuliert und somit Zeit eingespart werden kann. Während Horne [Law05] zu Beginn des Project Albert noch annahm, mit der entsprechenden Rechenleistung stets jede Parameterkombination simulieren zu können, kam spätestens gegen Ende des Projektes die Erkenntnis über die Notwendigkeit von Versuchsplänen. Das Ziel einer Versuchsplanung besteht darin eine realisierbare Anzahl an Simulationsläufen mit einem möglichst hohen Informationsgewinn zu finden [Reu+18, S. 6]. Dabei gilt es den Zielkonflikt zwischen der Anzahl an Versuchsläufen und der Simulationszeit optimal zu lösen. Für das DF agiert die HPC-Komponente unterstützend zugunsten der Simulationszeit, wodurch in der Versuchsplanung tendenziell eine größere Anzahl an Läufen simuliert werden kann als bei gewöhnlichen Simulationen. Agentenbasierte DF-Modelle charakterisieren sich durch viele Faktoren, die oftmals auf mehr als zwei Faktorstufen simuliert werden. Bei bereits 12 verschiedenen Faktoren mit 5 Faktorstufen müssten in einem vollfaktoriellen Design ca. 244 Millionen Simulationsläufe durchgeführt werden. Aus diesem Grund wurden in der MSG-088 Studie der praktische Einsatz alternative Designs evaluiert. In Abbildung 3.5 wurde die Verwendbarkeit von Versuchsplänen für DF-Studien für empfohlene Faktoranzahlen und Komplexitäten der Simulationsmodelle in einem Diagramm eingeordnet. Aufgrund großer Unterschiede verschiedener Simulationsmodelle ist das dargestellte Diagramm zwar eine grobe Auswahlhilfe auf Basis von Best Practices, stellt jedoch keinesfalls einen allgemeingültigen Leitpfaden dar. In Abschnitt 2.1.2 wurden für gewöhnliche Simulationen bereits voll- bzw. teilfaktorielle Versuchspläne verschiedener Auflösungen vorgestellt, die in Abbildung 3.5 ebenfalls alle innerhalb des unteren linken Quadranten zu finden sind. Das vollständige 2^k Faktor-Design wird von Kleijnen et al. [KSC03, S. 27] auch als „grobes Gitter“ betitelt, weil jeweils nur zwei Faktorstufen betrachtet werden. Als Pendant wird auch das feine Gitter (m^k) in Betracht gezogen, bei dem durch die Untersuchung beliebig vieler Faktorstufen (m) eine detaillierte Analyse für komplexere Modelle möglich ist [KSC03, S. 27]. Dadurch ist es möglich, Linearitäten in der Ausprägung von Zielgrößen zu beweisen oder zu widerlegen.

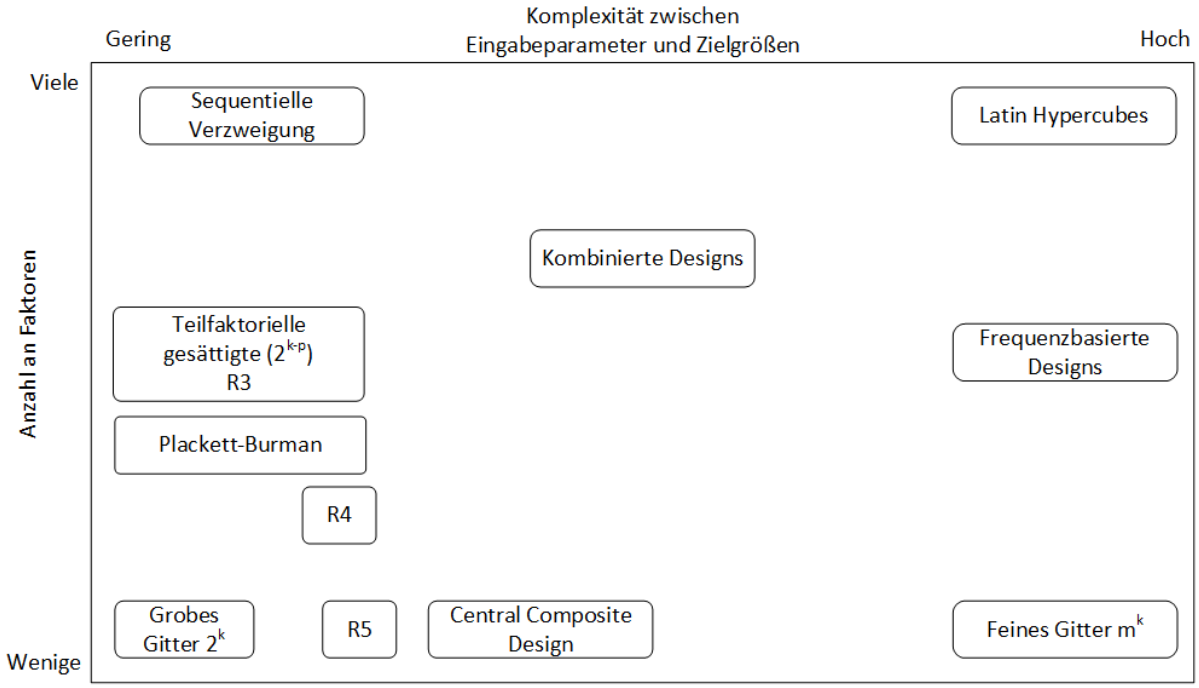

Abb. 3.5: Empfohlene Designs für Data Farming Studien (in Anlehnung an [KSC03, S. 26])

Generell erweisen sich vollfaktorielle Designs für die Untersuchung für mehr als eine „Hand voll“ Faktoren aber als ungeeignet, weil die Anzahl der erforderlichen Simulationsläufe exponentiell mit den Faktoren ansteigt [KSC03, S. 27].
Zur Reduzierung der Versuchsläufe können die ebenfalls im Grundlagenteil vorgestellten Teilfaktor-Designs der Auflösungen III, IV und V verwendet werden (R3, R4, R5). Je geringer die Auflösung, desto geringer ist auch die Anzahl an Simulationsläufen, wobei aber durch die zunehmende Vermengung von Wechselwirkungen niedriger Ordnungen auch wertvolle Informationen verloren gehen. Aus diesem Grund eignen sich Versuchspläne geringerer Auflösung zwar für eine größerer Anzahl von Faktoren, aber weniger für komplexere Modelle weil die Haupteffekte nur vermengt mit Wechselwirkungen analysiert werden können. Versuchspläne der Auflösung III werden auch gesättigte Versuchspläne genannt, weil jeder Effekt mit genau einem Faktor belegt ist [Kle20, S. 145]. Dadurch wird die Anzahl der Versuchsläufe auf ein notwendiges Minimum reduziert und entspricht der Anzahl der Faktoren [KSC03, S. 20]. Gesättigte Versuchspläne eignen sich nicht für die detaillierte Analysen zwischen den Faktoren und Zielgrößen und noch weniger für die Untersuchung von Wechselwirkungen, sondern eher für ein grobes Durchleuchten *(eng. „screening“)* der Ergebnisse für unterschiedliche Parameterkombinationen [Kle20, S. 144].
Eine besondere Form von Gitterdesigns ist das sog. Plackett-Burman (PB) Design und ist nach Siebertz et al. [SvH17, S. 51] das beste Design für zweistufige Faktoren. In der Regel wird es in der Auflösung III angewendet und besitzt im Vergleich zu den teilfaktoriellen Versuchsplänen den Vorteil, dass 2FWW nicht komplett mit Hauptfaktoren vermengt sind,

sondern lediglich zu einem gewissen Anteil. „Wechselwirkungen schlagen also nicht zu 100 % in die Haupteffektberechnung durch, verfälschen aber alle Haupteffekte der nicht an der jeweiligen 2FWW beteiligten Faktoren“ [SvH17, S. 33]. Kleppmann [Kle20, S. 157] bezeichnet die anteilige Verfälschung als „Verschmieren“ der 2FFW. Wenn die entsprechende 2FWW der Faktoren nur einen kleinen Einfluss besitzt, ist dieses Verschmieren ein Vorteil, weil keiner der untersuchten Faktoren wesentlich verfälscht wird. Andernfalls werden die Faktoren durch eine einflussreiche 2FWW zu stark manipuliert, wodurch der gesamte Versuchsplan unbrauchbar werden könnte [Kle20, S. 157]. Bei erfolgreicher Anwendung ist es mit dem Einsatz des PB Design allerdings möglich einen Versuchsplan mit bspw. 11 Faktoren mit 12 Versuchen zu erstellen, wobei in diesem Fall weniger als 0,6 % aller möglichen Kombinationen simuliert werden müssten [SvH17, S. 34].

Wesentlich geeignetere Versuchspläne für DF-Studien werden mittels sequentieller Verzweigungen, frequenzbasierten Designs oder sog. Latin Hypercubes erstellt. Sequentielle Verzweigungen basieren auf Kleijnens [Kle15, S. 136] Annahme, dass in Modellen mit vielen Eingaben nur wenige Faktoren einen signifikanten Einfluss auf die Zielgrößen haben. Mittels sequentieller Verzweigung sollen diese wenigen signifikanten Faktoren identifiziert werden. Voraussetzung ist allerdings, dass durch bereits durchgeführte Versuche die Haupteffekte von Faktoren bekannt sind und sich die Ausprägung der Zielgrößen durch einfache Metamodelle beschreiben lassen [Kle09, S. 6]. Metamodelle betrachten das eigentliche Simulationsmodell als Blackbox und approximieren das Verhalten dieser Blackbox durch die Herleitung einer mathematischen Beziehung zwischen Faktoren und Zielgrößen [NAT14, S. 64]. Sequentielle Verzweigungen setzen das Prinzip der Vererbung voraus [Kle15, S. 168]. Wenn der Haupteffekt eines Faktors keinen signifikanten Effekt auf eine Zielgröße besitzt, dann besitzt auch keine Wechselwirkung höherer Ordnung des Faktors einen Einfluss [Kle15, S. 168]. Wie der Name schon beschreibt, wird diese Methode sequentiell ausgeführt. Zunächst werden alle Faktoren als Gruppe betrachtet. Wenn die Gruppe einen signifikanten Einfluss auf die Zielgrößen besitzt, wird sie in zwei weitere Gruppen unterteilt, die wiederum erneut auf signifikante Einflüsse untersucht werden. Die Faktoren nicht einflussreicher Gruppen bleiben dabei unberücksichtigt. Durch die Wiederholung dieses Vorgehens werden auf der untersten Ebene nur signifikante Faktoren identifiziert, die für die Erstellung detaillierter Versuchspläne verwendet werden können. [HAN03, S. 566]
Sequentielle Verzweigungen sind darauf ausgelegt, aus Modellen vieler Faktoren die wirklich wichtigen Faktoren zu extrahieren, setzen dafür aber eine vergleichsweise triviale lineare und quadratische Beschreibung zwischen Ein- und Ausgabe voraus [Kle15, S. 168]. Aus diesem Grund ist diese Methode in der oberen linken Ecke in Abbildung 3.5 angeordnet.

Eine weitere interessante Planung von Experimenten besteht mittels Einsatz frequenzbasierter Designs, in denen ausgewählte Faktoren während eines Laufes sinusförmig zwischen einem definierten Minimal- und Maximalwert schwingen [KSC03, S. 28]. Wenn durch den Ausschlag der schwingenden Faktoren ein verändertes Verhalten der Zielgrößen beobachtet werden kann, ist ein signifikanter Einfluss vorhanden [KSC03, S. 28]. Der Ansatz frequenzbasierter Designs wird in der Praxis verwendet, um Metamodelle zweiter oder dritter

Ordnung herzuleiten, die in der Lage sind komplexere Zusammenhänge zwischen Faktoren und Zielgrößen zu beschreiben [KSC03, S. 29].

Die von Kleijnen, Sanchez und Cioppa T. M. [KSC03, S. 24] empfohlene geeignetste Methode für gleichermaßen komplexe Modelle und hohe Anzahlen von Faktoren ist die Verwendung eines LHD (Latin Hypercube Design). Ein LHD ist eine $n_r * n_f$ Matrix X^{LHD} mit n_r Läufen und n_f Faktoren, bei der jede Spalte der Matrix aus einer der zufälligen Permutationen der Zahlen 1, 2, 3,...,n_r besteht [SvH17, S. 205]. Aus einem LHD wird anschließend ein Latin Hypercube Sampling (LHS) erstellt, „indem von jedem Wert des LHD eine Zufallszahl aus dem Bereich [0,1) abgezogen wird und anschließend jeder Wert durch n_r geteilt wird“ [SvH17, S. 205]. Dadurch wird für jeden Tupel der Matrix X^{LHS} ein Wert zwischen 0 und 1 erzeugt, der mit dem möglichen Faktorraum multipliziert werden kann, um letztlich die Ausprägung der Faktoren zu erhalten. In Abbildung 3.6 ist für Erstellung eines LHD bzw. LHS für das Beispiel $n_r = 10$ und $n_f = 2$ dargestellt. Im Vergleich zu Gitterdesigns entsteht die Anzahl der Läufe nicht aus der Anzahl der

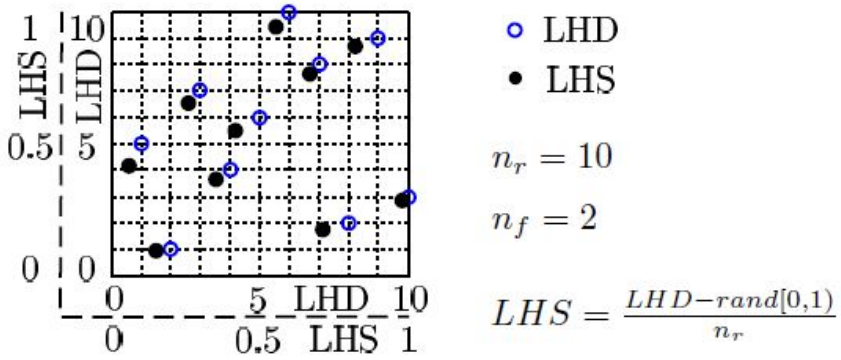

Abb. 3.6: Latin Hypercube Design/Latin Hypercube Sampling [SvH17, S. 205])

Faktoren und der Auflösung des Versuchsplans, sondern kann als fester Wert bestimmt werden. Je höher n_r gewählt wird, in desto kleinere Schichten wird der Faktorraum für das LHS aufgeteilt, wodurch detaillierte Analysen komplexer Modelle ermöglicht werden [SvH17, S. 205].
Die grundlegende Konstruktionsmethode von LHD garantiert allerdings keine gleichverteilten und korrelationsfreien Testfelder, wie anhand der blauen Permutationen der Faktoren in Abbildung 3.7 zu sehen ist [SvH17, S. 206]. Aufgrund der zufälligen Permutationen sind für ein LHD $n_r^{n_f}$ verschiedene Testfelder unterschiedlicher Qualitäten möglich [SvH17, S. 206]. Ye [Ye98] stellt dafür ein Verfahren vor, mit dem paarweise Orthogonalität zwischen jeweils zwei Faktoren gewährleistet wird, wodurch die unabhängige Bestimmung aller Haupteffekte garantiert wird.
LHDs zählen zu den sog. „space-filling“-Designs, weil beliebig viele Faktorstufen untersucht werden können. Im NATO-Bericht der MSG-088 wird das Vorhaben beschrieben, möglichst gleichverteilte orthogonale Latin Hypercubes zu verwenden, mit denen ein schneller Überblick über den gesamten Faktorraum erstellt werden kann [NAT14, S. 64].

Als letzte Methode wird die Kombination verschiedener Versuchspläne vorgestellt. Gerade für die Entwicklung von Metamodellen sollen nur Auswirkungen signifikanter Faktoren

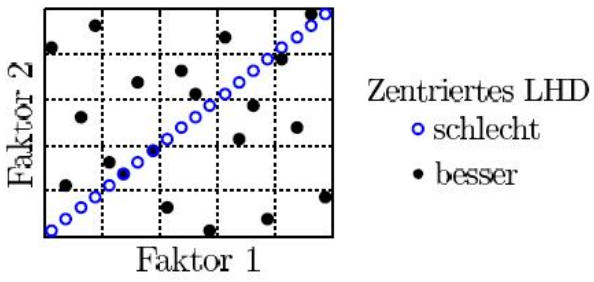

Abb. 3.7: Vergleich guter und schlechter Latin-Hypercube Designs [SvH17, S. 206])

(Entscheidungsfaktoren) berücksichtigt werden. Der Ansatz besteht darin, die restlichen nicht signifikanten Faktoren (Rauschfaktoren) mit einem weniger detailreichen Versuchsplan und weniger Faktorstufen zu simulieren als die Entscheidungsfaktoren. Kleijnen et al. [KSC03, S. 29] stellen dafür Ansätze vor, ein 2^{k-p} teilfaktorielles Design für Entscheidungsfaktoren zu verwenden, und empfehlen für Rauschfaktoren gesättigte bzw. nahezu gesättigte Versuchspläne. Eine ebenfalls gute Möglichkeit für eine überschaubare Anzahl von Entscheidungsfaktoren besteht in der Verwendung eines Central Composite Design (CCD) mit der Kombination eines LHDs für die Rauschfaktoren [KSC03, S. 29]. Beim CCD werden fünf Faktorstufen der Auflösung VI oder V untersucht. Damit zählt es zu den detaillierten Versuchsplänen, verglichen mit den zweistufigen Faktorplänen der Auflösung III bis V [SvH17, S. 41].
Insgesamt sind auf diese Art und Weise viele kombinierte Architekturen möglich, aus denen unterschiedlich gute Ergebnisse resultieren. Eine detaillierte Einordnung dieser Methode in das Diagramm aus Abbildung 3.5 ist daher stark vom Anwendungsfall abhängig. Insgesamt besteht das Ziel kombinierter Designs darin, mit vielen Parametern umzugehen und einen detaillierten Blick ausschließlich auf signifikante Faktoren zu ermöglichen. Dadurch wird der Verlust von Haupt- und Wechselwirkungen der Rauschfaktoren bewusst in Kauf genommen, wodurch nicht so viele Informationen gewonnen werden wie bei den Latin Hypercubes, frequenzbasierten Designs oder feinen vollfaktoriellen Versuchsplänen (m^k).

4. High Performance Computing Nachdem in den ersten zwei DF-Bereichen ein ausführbares Modell entwickelt wurde und im dritten Bereich ein Versuchsplan aufgestellt wurde, gilt es die Simulation in einem angemessenen Zeitrahmen durchzuführen. Der folgende Abschnitt klärt dafür was genau unter dem Begriff Hochleistungsrechnung verstanden wird und wie sich der Begriff im Verlauf der DF-Entwicklungen gewandelt hat. Eine HPC-Komponente kann nach Horne und Schwierz [HS16, S. 10] sowohl ein einzig leistungsstarker Rechner mit einem Multi-Core Prozessor sein als auch ein gesamtes Netzwerk von tausenden vernetzten Prozessoren. Bei der ersten Variante wird mehr darauf gesetzt durch leistungsstarke CPUs (Central processing unit) die Simulationszeit eines Laufes zu reduzieren. Bei der Zweiten werden die Läufe der Simulation hingegen auf ein HPC-Cluster aufgeteilt, um durch die Parallelisierung der Berechnungen Zeit einzusparen.

In beiden Varianten besteht die Hardware der HPC-Komponente immer aus Prozessoren, Arbeitsspeicher, Festplatten und einem Netzwerkzugang [HS16, S. 10].
Für die Leistungsbewertungen von Prozessoren existieren viele verschiedene Kennzahlen. Eine von Smith [Smi98, S. 1] vorgestellte Kennzahl sind die sog. Floating point operations per second (FLOPS) für die Anzahl an Gleitkommaberechnungen, die eine CPU pro Sekunde ausführen kann. Die Simulationsdauer ist hardwarespezifisch und abhängig von der Anzahl der verwendeten CPUs und FLOPS jeder CPU. Prozessoren mit mehreren Kernen können eine Rechenaufgaben auch innerhalb der CPU auf die Kerne verteilen, wodurch ein Vielfaches der FLOPS ausgeführt werden kann, als auf einem Kern [GO21].
Als die ersten Modelle im Jahre 2001 während des Project Albert zum Einsatz kamen, besaßen gewöhnliche Computer damaliger Zeit nicht genügend Kapazitäten, um die Anfragen in einer vertretbaren Zeit auszuführen [HL01, S. 18]. Da für die Experimentenplanung bereits damals mehrere 100.000 Läufe erforderlich waren, wurden die Simulationen im MHPCC verarbeitet [HL01, S. 18], dass zu diesem Zeitpunkt 4000 Giga-Flops GFLOPS ausführen konnte [HJ02, S. 5].
Im Vergleich zu aktuellen leistungsfähigen Prozessoren besitzt der AMD Threadripper 3990X, der im ersten Quartal 2020 auf dem Markt erschienen ist, eine Leistung von 3732 GFLOPS [Gad21]. Die Rechenleistung ist in knapp 20 Jahren so stark angestiegen, dass heutzutage eine einzelne CPU fast die gleiche Leistung besitzt wie ein gesamtes Rechencenter aus dem Jahr 2001. In Tabelle 3.1 werden die technischen Daten eines Intel Pentium 4 aus dem Jahr 2005 mit dem aktuellen AMD Threadripper 3990X verglichen. Beide Prozessoren gelten als führende Produkte ihrer Zeit. Während der Markt bis zur

Tab. 3.1: CPU Vergleich Intel Pentium 4 vs. AMD Threadripper 3990X [AMD21] [Che+05] [Gad21] [Int21]

Bezeichnung	Erscheinungsjahr	Takt (GHz)	Kerne	GFLOPS
Intel Pentium 4	2005	3,2	1	6,4
AMD Threadripper 3990X	2020	2,9	64	3732,0

Veröffentlichung des Intel Pentium 4 ausschließlich von Einzelkernprozessoren dominiert wurden, begann ab 2006 der Verkauf revolutionärer Multikernprozessoren [GO21]. Da die Erhöhung der Taktfrequenz ab einem bestimmten Wert zu größeren Problemen in der Wärmeabfuhr führt, wurde ein preisgünstigeres Verfahren entwickelt, um mehrere Kerne in einer CPU zu integrieren [GO21]. Der AMD Threadripper 3990X besitzt insofern zwar einen geringeren Basistakt als der Intel Pentium 4, ist jedoch durch die 64 Kerne in der Lage 64 arithmetisch-logische Einheiten parallel für eine Simulation zu verwenden [GO21]. Im Vergleich zum Intel Pentium 4 können mit dem AMD Threadripper 3990X über 580 mal so viele sekündliche Rechenoperationen durchgeführt werden. Durch die Vernetzung derartig leistungsfähiger Hardware besitzt der derzeitig schnellste Supercomputer im Oak Ridge National Laboratory (ORNL) in Tennessee einen Durchsatz von 122,3 TFLOPS durch ein Cluster aus 4356 CPUs und GPUs (Grapical processing unit) [Top21]. Dadurch

entsteht ein enormes Potential für die Simulation komplexer DF-Modelle mit einer hohen Anzahl an Parametern und Effekten.

Die Aufgabe des DF-Bereichs *High-Performance-Computing* besteht zunächst in der Auswahl geeigneter HPC-Ressourcen und darüber hinaus in der erfolgreichen Integration in die bestehende Systemarchitektur. Die HPC-Ressource muss dazu in der Lage sein alle Experimentdaten aus einer Speicherquelle zu laden, das Modell mit simulationsspezifischen Eingabeparametern zu versorgen, die Simulationsläufe als einzelne Jobs auszuführen und die Ergebnisse in einer definierten Form und einem festgelegten Speicherort zu deponieren [Hub+19, S. 8]. Aus der Kombination eines ausführbaren Computermodells, der Experimentenplanung und der HPC-Ressource entsteht ein sog. „data farmable" Modell, das wie folgt definiert wird:

> *„A data farmable model is a computable model whereby the inputs can be modified programmatically, i.e., through some computer program, and an instance of that model can be started programmatically, e.g., from a command-line interface without a Grapical User Interface (GUI) [NAT14, S. 87]."*

Das „data farmable" Modell ist zusammengefasst ein autarkes Computermodell, das nach einem manuellen Startaufruf in der Lage ist ein gesamtes Experiment ohne menschliche Eingriffe durchzuführen. Als Grundlage wird das Studienpaket und das ausführbare Computermodell als Ergebnisse der *Experiment Definition Loop* verwendet (Abbildung 3.3). Aus den im Studienpaket enthaltenen Informationen über das verwendete Design zur Versuchsplanung werden über einen Algorithmus zunächst die Parameter aller Simulationsläufe in einem Speicher abgelegt [NAT14, S. 90]. Dies kann entweder über SQL-Transaktionen in eine Datenbank oder die Erstellung von ASCII- oder XML-Dateien erfolgen [NAT14, S. 90]. Prinzipiell ist es vorteilhaft in der gesamten Architektur das gleiche Speicherformat zu verwenden, um die Programmierung zahlreicher Schnittstellen zu vermeiden. Das Zielformat soll einer Matrix gleichen, in der die Faktoren als Attribute abgespeichert sind, die in jedem Datensatz die Parameterwerte eines Laufes enthalten sind. Bei komplizierten Designs (z. B. kombinierte Designs oder Designs mit erforderlichen Metamodellen) sind vorher zusätzliche Implementierungen der entsprechenden Algorithmen erforderlich, auf die im Rahmen der Masterarbeit kein detaillierter Bezug genommen wird.
Im Anschluss wird für jeweils einen Simulationslauf ein sog. Job für die HPC-Ressource erzeugt, welcher die jeweiligen Parameterwerte des Experimentenplans und alle weiteren erforderlichen Variablen zur Ausführung des Modells beinhaltet [NAT14, S. 92]. Alle erforderlichen Transaktionen, inklusive der Planung und des Managements der Jobs für die HPC-Ressource, werden von der DF-Software ausgeführt, welche das *data farmable* Modell, den Experimentenplan, die Jobs und die HPC-Ressource wie „Klebstoff" miteinander kombiniert. Die Funktionen der Software und die Gestaltung der Schnittstellen sind vom Projekt abhängig. Die Programmierung der DF-Software ist daher immer auf die Architektur zugeschnitten und erfolgt speziell für jedes Szenario durch das Projektteam. Die Zuständigkeit für die Auswahl der HPC-Ressource und der anschließenden Entwicklung des „data farmable" Modells liegt bei den Modellexperten.

5. Analysis and Visualization Der Bereich *Analysis and Visualization* beschreibt den Einsatz statistischer Werkzeuge und Präsentationstechniken mit dem Ziel die generierte Datenmenge übersichtlich zusammenzufassen, wichtige Informationen hervorzuheben und letztlich eine Entscheidungsgrundlage für Stakeholder schaffen [NAT14, S. 109].
Das zu erfüllende Teilziel besteht darin, die Ausprägung der Zielgrößen zu verstehen und die Auswirkungen der Parameter auf die Zielgrößen nachzuvollziehen. Dafür empfiehlt die MSG einen Top-Down-Ansatz, indem zunächst die gesamte Ergebnislandschaft betrachtet wird und darauf basierend interessante Bereiche oder Schwellenwerte für Faktoren detaillierter analysiert werden [NAT14, S. 115]. Für die Erfüllung der Teilziele sollten durch das gewonnene Systemverständnis optimale Parameterwerte für zukünftige Experimente prognostiziert werden [NAT14, S. 115].
In der Praxis werden dazu unter anderem folgende statistische Werkzeugen und Verfahren angewendet:

- Lage- und Streuungsmaße
- Zeitreihenanalysen
- Charakterisierung von Verteilungen
- Hypothesentests
- Korrelations- und Regressionsanalysen

[NAT14, S. 113]

Darüber hinaus werden unter dem Stichwort *Data Mining* auch Techniken vorgestellt, um Muster in Daten zu erkennen und Metamodelle zu erstellen. Während *Data Mining* das Suchen nach vergrabenen Informationen in einer großen Datenmenge beschreibt (*„Miners seek valueable buried nuggets“*), wird durch DF eine Datenmenge in einer Art und Weise erstellt, um möglichst maximalen Ertrag zu generieren (*„Farmers cultivate to maximize yield“*) [San10]. Für die Anwendung von Data Mining Techniken wird demnach eine große Datenmenge vorausgesetzt, die mit DF erzeugt wird.
Für die übersichtliche Darstellung der Ergebnislandschaft eignen sich dreidimensionale *Fitness Landscapes*, in denen jeweils zwei Parameter und eine Zielgröße grafisch dargestellt werden [HL01, S. 21] (s. Abbildung 3.8). Dies ermöglicht dem Benutzer die Stabilität der Region zu beurteilen. Es können Regionen ungewöhnlicher Daten ausfindig gemacht werden, in denen die Oberfläche der Landschaft besonders rau oder glatt erscheint [HL01, S. 21].
Für n-dimensionale Daten eignen sich für einen großen Einblick ebenfalls Streudiagramme oder Streudiagramm-Matrizen. In der Streudiagramm-Matrix in Abbildung 3.9 werden in der Hauptdiagonalen die Namen der Parameter und Zielgrößen aufgetragen, wobei sich in einer Zelle (i, j) jeweils das Streudiagramm der i-ten und j-ten Variable befindet [Wol16, S. 243]. Im Vergleich zu den Fitness-Landscapes besteht der Vorteil, dass auch kombinierte Effekte von Parametern auf die Zielgrößen analysiert werden können. Neben der direkten statistischen Analyse der Daten können auch Werkzeuge zur Visualisierung verwendet werden, um entweder den Ablauf einzelner Simulationsläufe im Detail zu verstehen oder Stakeholder eines Projektes durch aufschlussreiche Animationen die Erkenntnisse der statistischen Analyse zu präsentieren [NAT14, S. 112].
Für agentenbasierte Modelle eignet sich besonders gut die Verwendung von Playbacks als

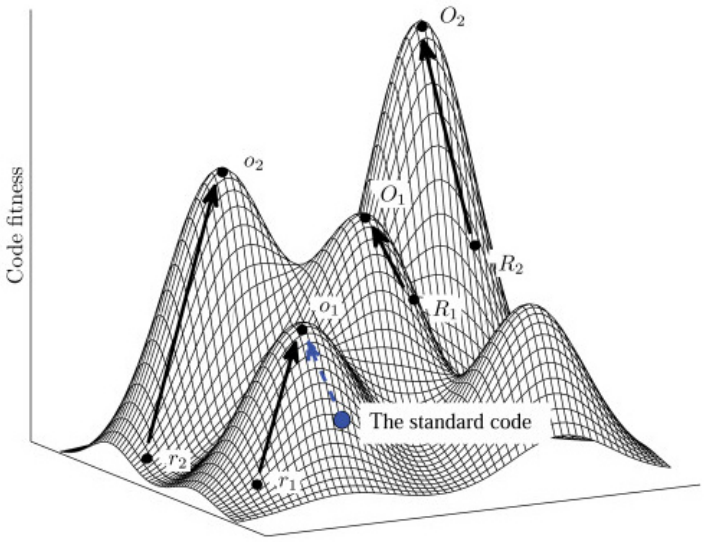

Abb. 3.8: Beispiel eines Fitness Landscape Diagramms [NWK07, S. 15]

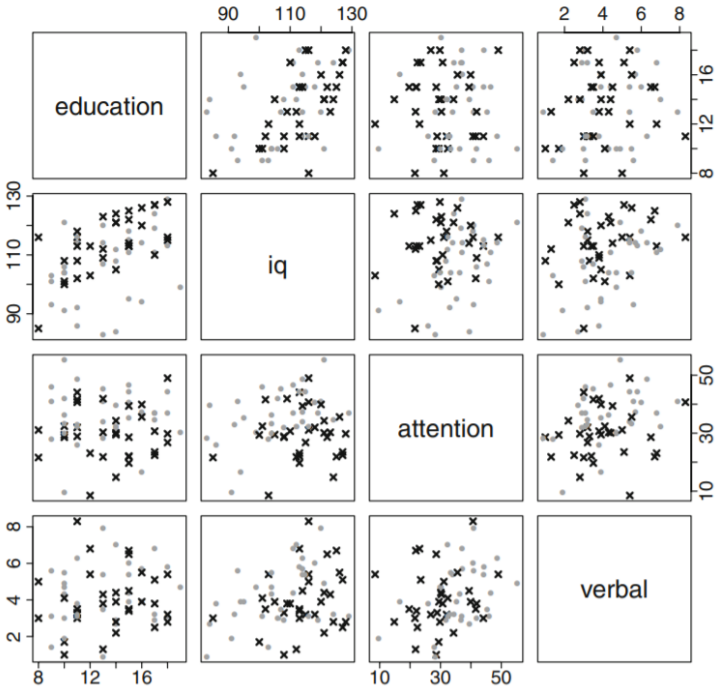

Abb. 3.9: Beispiel einer Streudiagramm-Matrix [Wol16, S. 243]

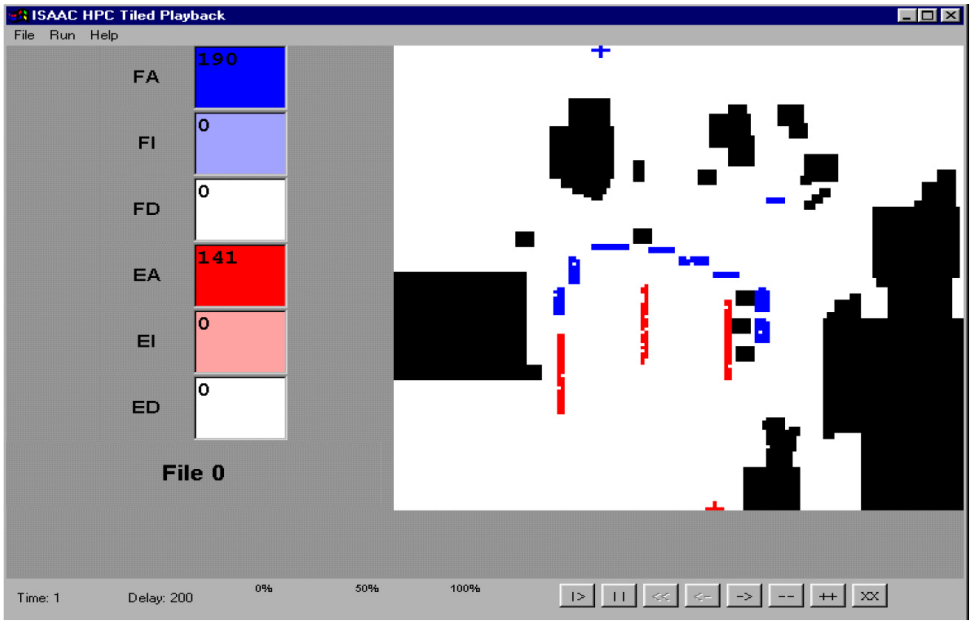

Abb. 3.10: Playback-Editor des ISAAC [HL01, S. 19]

grafische Visualisierung einer Simulation, in denen die Aktionen der Agenten innerhalb des Terrains im zeitlichen Verlauf dargestellt werden (s. Abbildung 3.10) [HL01, S. 19]. Dadurch können die Gründe für die Entstehung von Anomalien auf der untersten Ebene eines interagierenden Agenten untersucht werden. Im besten Fall bietet ein guter Playback-Editor ähnliche Funktionen eines Musikplayers zum Abspielen, Pausieren und Vor- bzw. Zurückspulen der Simulation [NAT14, S. 32].

Jedes Werkzeug/Verfahren zur Analyse und Visualisierung der generierten Datenmenge besitzt unterschiedliche Vorteile in der Anwendung. Es gibt daher kein allgemeines Anwendungsmuster für Analystechniken, wodurch die Auswahl stark von den jeweiligen Untersuchungsfragen abhängt. Horne und Schwierz [HS16, S. 13] formulieren deshalb folgende zehn Fragen für die erfolgreiche Evaluierung der Analysephase:

1. Wie groß war die Streuung der Zielgrößen über das gesamte Experiment?
2. Welche zufälligen Variationen wurden nur über die zufälligen Replikationen beobachtet?
3. Sind Ausreißer vorhanden?
4. Sind Korrelationen zwischen Zielgrößen vorhanden?
5. Welche Faktoren waren am einflussreichsten?
6. Sind signifikante Wechselwirkungen vorhanden?
7. Wo liegen interessante Regionen und Schwellenwerte?
8. Sind einige Ergebnisse kontraintuitiv?
9. Welche Parameterkombinationen waren am besten?
10. Gibt es Parameterkombinationen, die mehrere Ziele erfüllen?

Basierend auf der Entwicklung im Rahmen der NATO wird von der MSG folgende Software für die Analyse großer Datenmengen empfohlen:

- JMP
- S-PLUS
- SPSS
- Minitab
- Stata
- MATLAB
- Microsoft Excel

[NAT14, S. 116]

6. Collaboration Der letzte Bereich *Collaboration* wird von Horne und Schwierz [HS16, S. 13] als Hauptgrundsatz des DFs bezeichnet. Der Bereich der Kollaboration resümiert die gesamte Schleife des DFs zusammen mit weiteren Fachexperten des Forschungsgebietes und beschreibt damit den weltweiten Austausch von Erfahrungen und Expertise [HS16, S. 13] [NAT14, S. 141]. Der sechste Bereich ist daher nicht essentiell für eine einzige Studie, aber mit dem Blick auf das große Ganze gerichtet basiert die gesamte Entwicklung seit dem Project Albert auf der Zusammenarbeit und dem regelmäßigen Austausch internationaler Projektteams. Die Kollaboration wird unter anderem durch jährliche Workshops gefördert, bei denen sich Fachexperten modellierter Szenarien mit sachkundigen interdisziplinären Modellexperten und Datenanalysten austauschen, um neue Erkenntnisse zu gewinnen [HS16, S. 13]. Nach Hornes Aussagen ist diese Kombination von anwendungsspezischem Fachwissen und der technischen Expertise zur Planung, Gestaltung und Durchführung einer DF-Studie der Schlüssel zu realistischen und aussagekräftigen Ergebnissen [HS16, S. 16]. Die Kolloboration bezieht sich darüber hinaus aber ebenfalls auf den Austausch von DF-Strukturen, der Präsentation interessanter Ergebnisse und auf den Einsatz und die Verwendung unterschiedlichster Hardware und Software, die für DF-Studien empfohlen werden.

3.2.3 Data Farming as a Service

Die Entwicklung von DFS begann 2017 während der MSG-155 Studie und ist auch zum gegenwärtigen Zeitpunkt ein aktuelles Forschungsthema der NATO, um DF durch Modularisierung flächendeckend, ausgereift und intuitiv für Anwender bereitzustellen [Hub+19, S. 2]. Durch die Architektur der DFS entstehen sowohl in der technischen Realisierung als auch in der Durchführung einer Studie wesentliche Vorteile. Aufgrund des modularen Aufbaus wird die Austauschbarkeit einzelner Services gewährleistet. Unter dem Stichwort der Kollaboration können DF-Experten jederzeit zwischen der eigenen Implementierung von Services oder der Verwendung bereits bestehender Services entscheiden [Hub+19, S. 6]. Zudem kann die Entwicklungszeit neuer Module durch die Möglichkeit simultaner

Bearbeitungen enorm reduziert werden [Hub+19, S. 6].
Seitens der Anwender besteht der größte Benefit in der Verwendung eines zentralen webbasierten DFS-Portal, über das alle Funktionen, von der Planung einer Studie bis zur Einbindung sämtlicher Analysewerkzeuge, einfach und intuitiv aufgerufen werden können. Der Anwender benötigt daher kein spezifisches Wissen bzgl. der Architektur, wodurch die Durchführung von Studien prinzipiell für eine größere Zielgruppe ermöglicht wird. In Tabelle 3.2 sind die möglichen drei Arten von DFS dargestellt [Reu+18, S. 8]. Die Inter-

Tab. 3.2: Arten von Data Farming Services

Name	Beschreibung
Business services	Modularer Aufbau der DF Bereiche
Repository services	Steuern der Transaktionen der *Business services* auf die Speicherquellen
Cross sectional support services	Unterstützung und Ermöglichung für die Verwendung der *Business-* und *Repository services*

aktion der DFS orientiert sich stark an die sechs DF-Bereiche und ist in Abbildung 3.11 dargestellt. Die dunkelblauen Kästchen stellen die *Business services* dar, während die grünen Kästchen als Speicherquellen die *Repository services* repräsentieren. Die *Business services* enthalten vier der sechs DF-Bereiche, in denen die Modellentwicklung und die Kollaboration ausgeschlossen sind.
Während die Kollaboration ohnehin keine technische Umsetzung beinhaltet, wird die Modellbildung nicht als DFS unterstützt, weil diese oftmals in eigenständiger Modellierungs- bzw. Simulationssoftware erfolgt. Insofern wird für eine DFS-Architektur bereits die vorherige Entwicklung eines Modells vorausgesetzt, wodurch der Service *Model Execution* entsteht. Das *Rapid Szenario Prototyping* ist als *Business Service* grau gefärbt, weil es durch die fehlende Modellbildung nicht den Umfang beinhaltet, der in den sechs Bereichen beschrieben wird. In der DFS-Architektur beinhaltet das *Rapid Szenario Prototyping* lediglich die Veränderung initialer Modellparameter.
In Bezug auf den aktuellen Stand der Technik bestehen die Speicherquellen für die *Repository services* fast ausschließlich aus einer bzw. mehreren Datenbanken. Im Hinblick auf die industrielle Verwendung des DFs beschreiben Reus et al. [Reu+18, S. 8] die Absicht, als Speicherquellen direkt die Unternehmensdatenbanken zu verwenden. Auf diese Art wird der Zugriff auf erforderliche Stammdaten ohne aufwändige ETL-Prozesse (Extract, Transform, Load) ermöglicht. Dadurch ergibt sich auch der Begriff der *Business services*, weil die DF-Funktionen direkt mit der IT-Unternehmenslandschaft verbunden sind. Der lauffähige Betrieb der DFS wird durch die hellblauen *Cross sectional support services* ermöglicht, in denen Zugriffe autorisiert und nutzergruppenspezifische Rechte erteilt und verwaltet werden [Reu+18, S. 8]. Alle Services sind als REST-Web Services (Representational State Transfer) implementiert, in denen die Kommunikation durch HTTPS und OAuth2 Protokolle gesichert ist.

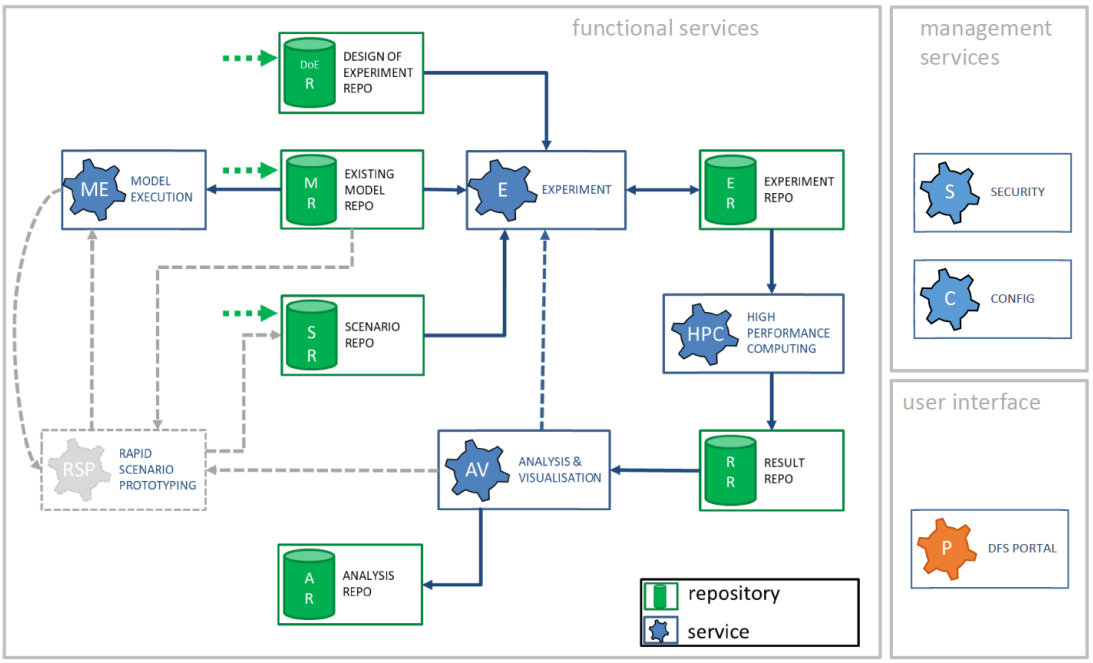

Abb. 3.11: Überblick der Data Farming Services [Hub+19, S. 8]

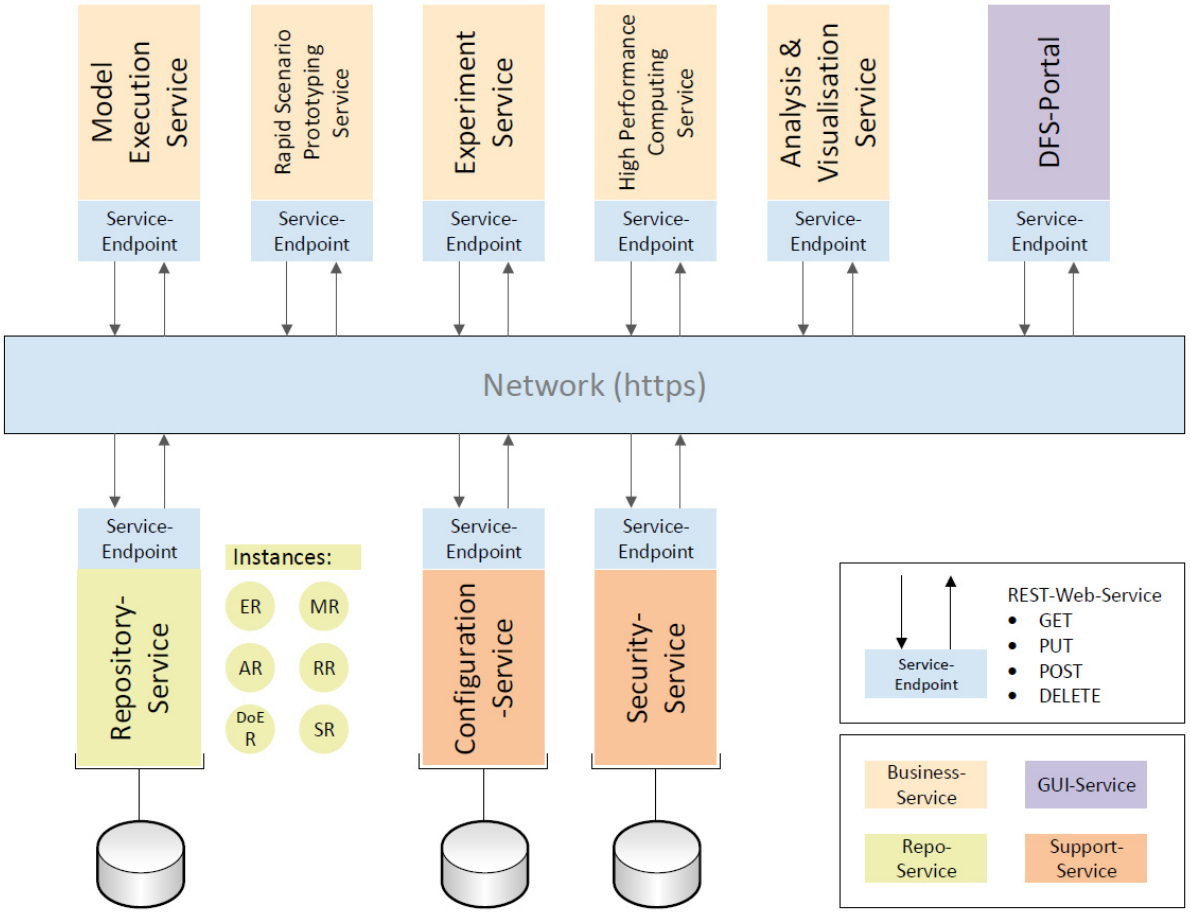

Abb. 3.12: Architektur der DFS [Hub+19, S. 9]

In Abbildung 3.12 ist die Architektur der DFS inklusive der Schnittstellengestaltung der REST-Web Services zur gesicherten HTTPS-Weboberfläche, schematisch dargestellt. Die Aufgabe der Service-Endpoints besteht darin die Kommunikation der einzelnen Services zum Web-Server durch die REST-Anfragen GET, POST, PUT oder DELETE zu ermöglichen. Die Interaktion mit dem Benutzer findet ausschließlich über das DFS-Portal statt [Hub+19, S. 7]. Jeder Benutzer kann sich in dem DFS-Portal mit seinem eigenen Account einloggen und erhält ein übersichtliches Dashboard mit allen möglichen Funktionen und einer Historie vergangener Experimente [Hub+19, S. 7]. Die wesentlichen Funktionalitäten bestehen in der Anpassung von Parametern für ein neues Szenario, dem Erstellen von Experimenten mit individuellen Versuchsplänen, der Durchführung von Experimenten und der anschließenden Analyse und Visualisierung der Ergebnisdaten [Hub+19, S. 7].

3.3 Definition des Data Farmings

In der Literatur zu den Ursprüngen des DFs, während des Project Albert und den Weiterentwicklungen der MSG in der NATO, gibt es unterschiedliche, teilweise kontroverse Definitionen des Begriffs „Data Farming“. Daher gilt es für den Rahmen der Masterarbeit eine allgemeingültige Definition zu verwenden, die sowohl für den Kontext militärischer agentenbasierter Simulationsstudien als auch für die Problemdomäne von SCs geeignet ist. Einer der ersten Definitionsansätze stammen von Horne und Brandstein aus dem Jahr 1998, in denen DF als eine neue Komponente betitelt wird:

> *„We call this new component Data Farming“* [BH98, S. 93]

In diesem frühen Entwicklungsstadium wurde der Begriff als eine Blackbox betrachtet, dessen Potentiale ungewiss waren. Dennoch existierten Anforderungen, die sich im wesentlichen auf die Fortschritte für die Entwicklung agentenbasierter Modelle, der Steigerung der Rechenleistung und der Analyse und Visualisierung großer Datenmengen bezogen [BH98, S. 93]. Als Ergebnis des Project Albert wird die Gestaltung der Komponente mit der Veröffentlichung der DF-Schleife von Horne und Meyer präziser formuliert:

> *„Data Farming is a methodology and capability that makes use of high performance computing to run models many times.“* [HM05, S. 1]

Der Begriff wird als Methode definiert, welche zur Erzeugung von Datenmengen bereits die Verwendung von Modellen und den Aspekt der notwendigen Rechenleistung beinhaltet, aber noch nicht die Analyse der Daten berücksichtigt. Erst mit der Veröffentlichung der sechs DF-Bereiche wird diese ebenfalls essentieller Bestandteil der Definition:

> *„Data Farming is a process that was developed to aid decision-makers in answering questions that are not addressed by traditional modeling and modeling processes. Data farming uses an inter-disciplinary approach that includes modeling and simulation,*

> *high performance computing, and statistical analysis to examine questions of interest with large number of alternatives.“* [HS16, S. 1]

Aufgrund der detaillierten Beschreibung und Abgrenzung der Bereiche und der Zuweisung von Zuständigkeiten in der MSG-088 Studie definiert die MSG DF als Prozess, der hauptsächlich darauf ausgelegt ist Entscheidungsträgern Antworten zu ermöglichen. Diese Definition kann in der Problemdomäne von SCs jedoch nur Gültigkeit besitzen, wenn der aktuelle Stand der DF-Entwicklung den Anforderungen eines Prozesses entspricht.

Nach der DIN EN ISO 9000 ist ein Prozess definiert als ein *„Satz zusammenhängender oder sich gegenseitig beeinflussender Tätigkeiten, der Eingaben zum Erzielen eines vorgesehenen Ergebnisses verwendet“* [Deu15, S. 78]. Zusätzlich zu definierten Ein- und Ausgaben werden im industriellen Kontext weitere Anforderungen an einen Prozess in der DIN EN ISO 9001 aufgeführt:

- die Aufgaben der Prozessbeteiligten
- die Methoden, wie welche Tätigkeiten auszuführen sind
- die Ressourcen, die zur Verfügung stehen müssen
- die Messgrößen, mit denen der Erfolg des Prozesses gemessen wird
- die Risiken, die mit dem Prozess verbunden sind

In den Berichten der MSG wird das methodische Vorgehen zur Durchführung einer DF-Studie zwar auf einer Metaebene beschrieben. Präzise sequentielle Schritte in Form von Vorgehensmodellen oder Ablaufdiagrammen sind allerdings nicht zu allen Phasen enthalten. Weiterer wichtige Aspekte sind Messgrößen, die als Indikatoren dienen, um die Prozessfähigkeit zu beurteilen. In Simulationsstudien gibt es keine einheitliche Kennzahl zur Messung des Erfolgs. Diese Indikatoren sind vielmehr projektspezifisch auf der Basis der jeweiligen Zieldefinition abzuleiten. Auf potentielle Risiken von DF-Studien wird in Berichten der MSG ebenfalls keinen Bezug genommen. Gerade für die Realisierung von DFS bei denen direkte Zugriffe auf Unternehmensdatenbanken erfolgen sollen, müssten Vorkehrungen getroffen werden, um bspw. Risiken für Datenverluste auszuschließen. Weiterhin sind die Folgen für reale/geplante Systeme abzuwägen, wenn aus eine DF-Studie falsche Erkenntnisse abgeleitet werden.
Insgesamt sind in Bezug auf das Vorgehen und der Zuweisung von Zuständigkeiten bereits strukturelle Ansätze vorhanden, die auf einen Prozess hinweisen. Die Anforderungen an verfügbare Ressourcen, klar definierte Messgrößen und potentielle Risiken werden durch den aktuellen Stand der DF-Entwicklung nicht ausreichend erfüllt, um DF als Prozess in der Unternehmenslandschaft zu etablieren.
Im Grundsatz beschreibt DF das „Wie“, also die Tätigkeiten, die auszuführen sind, um aus Modellen unter Verwendung hoher Rechenleistung eine möglichst große Ergebnislandschaft zu generieren und zu analysieren, damit Erkenntnisse abgeleitet werden können. Eine größtenteils zutreffende Definition für dieses Bestreben formulierte Kusiak bereits im Jahr 2005:

> *„Data farming is concerned with methods and processes used to define the most appopriate features for data collection, data transformation, data quality assessment and data analysis“* [Kus06, S. 280]

Nach Kusiaks Ansicht befasst sich DF mit geeigneten Methoden und Prozessen zur Erfassung, Transformation, Bewertung und Analyse von Daten. Auch wenn DF aus genannten Gründen bisher noch nicht als Prozess definiert werden kann, ist das für zukünftige Entwicklungen aber nicht ausgeschlossen, wodurch die Definition zukunftsfähig ist. Insgesamt ist die Definition von Kusiak aber sehr allgemein formuliert, da kein expliziter Bezug zur Simulation, dem Modellbegriff oder dem Einsatz der Hochleistungsrechnung genommen wird. Theoretisch ist aber auch die Möglichkeit gegeben DF unabhängig von Simulationen und Modellen und ohne Bezug zu realen Systemen zu praktizieren, indem lediglich fiktive Daten durch Algorithmen generiert werden. Kusiaks Definition bezieht sich deshalb nur auf das notwendige Minimum für die Beschreibung, wodurch eine hohe Allgemeingültigkeit entsteht. Im abstrahierten Sinne geht es aber genau darum: Eine große Masse von Daten erzeugen und anschließend Informationen und Wissen ableiten. Mit den sechs DF-Bereichen entwickelte die MSG zwar eine Methode zur Anwendung von DF, prinzipiell sind aber auch andere Vorgehensweisen möglich.

4 Integrationsansätze für Data Farming Services in einer Supply Chain

Das Hauptziel dieser Arbeit besteht in der Ausarbeitung von Ansätzen für ein Vorgehensmodell, um DF innerhalb von SCs zu integrieren. Angesichts der geschilderten Datenarchitektur moderner SCs in Abschnitt 2.2 sind deutliche Trends in Richtung organisationsübergreifenden Cloud-Datenbanken zu erkennen. Das Potential dieser zunehmenden Vernetzung und der gemeinsamen Datenbasis soll daher als Ansatzpunkt verwendet werden, um DFS in SCs zu integrieren. Die Vision lautet, den Nutzen von DF gleichermaßen für alle Akteure der SC über Web-Services zugänglich zu machen. Durch zunehmende Zusammenarbeit könnten mit Hilfe eines DFS-Portals optimale Systemverhalten für verschiedene Steuerungsstrategien identifiziert werden. Auf diese Weise würde die gesamte Lieferkette koordiniert ein gemeinsames übergeordnetes Ziel verfolgen, bei dem die Charakteristiken dynamischer Effekte bekannt sind und effektiv minimiert werden können.
Für die Umsetzung des Vorhabens müssen die Eigenschaften von DF-Modellen auf die Problemdomäne von SCs übertragen werden. Die dazu erforderlichen Integrationsansätze gliedern sich in zwei Stufen. In Abschnitt 4.1 werden die Integrationsansätze der ersten Stufe formuliert, indem die Vorgehensweisen für Simulationsstudien aus Abschnitt 2.1.3 mit denen des DFs aus Abschnitt 3.2.1 in einer Konzeptmatrix verglichen werden. Das übergeordnete Ziel ist es, notwendige Phasen für den Einsatz von DFS in SCs zu erkennen. Die Durchführung bzw. das Vorgehen innerhalb einzelner Phasen bleibt hierbei zunächst unberücksichtigt. Es steht lediglich im Fokus ein grobes Konstrukt für eine Vorgehensweise nach dem Top-Down-Ansatz zu gestalten. Darauf aufbauend werden in Abschnitt 4.2 Anforderungen an die Phaseninhalte, bzgl. der Durchführung und der erforderlichen Hardware und Software, aufgestellt. Falls die bisherigen Methoden und technischen Mittel einzelner Phasen nicht den aufgestellten Anforderungen entsprechen, werden Integrationsansätze der zweiten Stufe abgeleitet.

4.1 Integrationsansätze Stufe I - Phasenvergleich

In Abschnitt 2.1.3 wurden vier verschiedene Vorgehensmodelle für den Einsatz der Simulationstechnik im Bereich der Produktion und Logistik vorgestellt, die weiterhin als *klassische Vorgehensmodelle* bezeichnet werden. Benannte Modelle nach Banks und Law ähneln sich dabei stark im Aufbau und beschreiben primär sequentielle und simultane Tätigkeiten einer Simulationsstudie. Im Gegensatz zur prozessorientierten Betrachtung im Unternehmensumfeld konzentrierte sich Sargent vielmehr auf die technische Umsetzung des Simulationsmodells, wodurch die Schritte des Vorgehensmodells nach den Phasenergebnissen gegliedert sind. Die ASIM vereint diese beiden Ansätze, indem sowohl die Tätigkeiten als auch daraus resultierende Ergebnisse im Vorgehensmodell aufgeführt werden. In allen Modellen sind allerdings die fünf Elemente *Aufgabenanalyse, Modellformulierung, Modellimplementierung, Modellüberprüfung* und *Modellanwendung* vertreten

Tab. 4.1: Vergleich der Data Farming Bereiche mit den Elementen klassischer Vorgehensmodelle

Elemente klassischer Vorgehensmodelle	Data Farming Bereiche
Aufgabenanalyse	
Modellformulierung	
Modellimplementierung	Rapid Scenario Prototyping Model Development
Modellüberprüfung	
Modellanwendung	Design of Experiments High Performance Computing Analysis and Visualization

(vgl. Abschnitt 2.1.3). Auch fünf der sechs formulierten DF-Bereiche der MSG können grundsätzlich den fünf gemeinsamen Elementen klassischer Vorgehensmodelle zugeordnet werden, wie in Tabelle 4.1 gegenübergestellt wird. Der Bereich *Collaboration* wird in dieser Darstellung nicht berücksichtigt, da er keine definierten Tätigkeiten für einzelne Projekte beinhaltet. Das *Rapid Szenario Prototyping* und *Model Development* beschreiben das iterative Vorgehen für die *Modellimplementierung*, während die Bereiche *Design of Experiments, High Performance Computing* und *Analysis and Visualization* zur Modellanwendung zugeordnet werden können. Es ist deutlich zu erkennen, dass Tätigkeiten zur *Aufgabenanalyse, Modellformulierung* und *Modellüberprüfung* von der MSG weniger ausführlich behandelt werden. Der Grund dafür ist, dass Projekte im Kontext der Produktion und Logistik stärker von den Prozessen der Unternehmen abhängig sind und deshalb zeitlichen und monetären Faktoren unterliegen. Präzise Aufgabenabgrenzungen, Projektplanungen und Modellkonzeptionierungen besitzen deshalb eine größere Bedeutung, um Kunden und Vorgesetzte über den aktuellen Projektstatus zu unterrichten. Die Arbeiten der MSG besitzen in dieser Hinsicht vielmehr einen „forschenden Charakter“, der sich, wie auch das Vorgehensmodell nach Sargent in Abschnitt 2.1.3, primär auf die technische Umsetzung, also der Implementierung und Anwendung des Modells, bezieht.
Die folgenden Abschnitte orientieren sich an den fünf grundlegenden Elementen klassischer Vorgehensmodelle, die ebenfalls repräsentativ für die Simulationsstudien in SCs sind. Es gilt, die klassischen Vorgehensmodelle mit der Durchführung von DF-Studien zu vergleichen, um einerseits gemeinsame Phasen ausfindig zu machen und darüber hinaus Ansätze zur Integration zu formulieren, wenn eine Phase nicht gleichermaßen in beiden Vorgehensweisen vorhanden ist.

4.1.1 Aufgabenanalyse

Die wesentlichen Phasen der Aufgabenanalyse bestehen bei klassischen Vorgehensmodellen aus einer Problemformulierung, der daraus folgenden Zielstellung und der Aufstellung eines

Projektplans (s. Tabelle 4.2). Die Problemformulierung und die Zielstellung sind zentrale Bestandteile aller klassischen Vorgehensmodelle, wohingegen die Problemformulierung für das DF weniger Relevanz besitzt (Tabelle 4.2 - Ph. 1.1). Das Ziel des DFs ist es, unabhängig von den Problemstellen eines Systems, eine große Ergebnislandschaft zu erstellen. Die Problemstellung besitzt deshalb keinen Einfluss auf die Durchführung der Studie. Folglich lässt sich daraus der erste Integrationsansatz (IA) ableiten:

IA St.1/Nr.1 Eine Problemformulierung ist für DF-Studien nicht notwendig, weil sie keinen Einfluss auf die Versuchsplanung besitzt. Zudem ist auch Optimierungspotential vorhanden, wenn keine Probleme eines realen Systems bekannt sind.

Die Zielstellung ist hingegen auch für das DF zwingend erforderlich, um generierten Daten im Hinblick auf ein bestimmtes Ziel auszuwerten (Tabelle 4.2 - Ph. 1.2). In der Anfangsphase des DFs wird deshalb auch mehrmals erwähnt, dass mit DF Fragen untersucht werden, die mit klassischen Simulationen nicht beantwortet werden können (vgl. Kapitel 3). Die Aufstellung dieser Untersuchungsfragen ist das Ergebnis einer ausführlich durchdachten Zielstellung. Weiterhin kann durch das Festlegen der Zielsetzung das Erreichen oder Nichterreichen und somit auch die Wirtschaftlichkeit einer Studie beurteilt werden [VDI14, S. 21].
Eine weitere wichtige Tätigkeit der *Aufgabenanalyse* ist die Aufstellung eines Projektplans, die ebenfalls in den Modellen nach Banks, Law und der ASIM enthalten ist (vgl. Abschnitt 2.1.3). Während Banks und Law den Projektplan als Inhalt der Phase zur Problemformulierung beschreiben, werden derartige Tätigkeiten im Modell der ASIM vor der Studie im Rahmen einer Grundsatzentscheidung durchgeführt. In der Literatur von Sargent ist die Aufstellung eines Projektplans nicht explizit gefordert (Tabelle 4.2 - Ph. 1.3). Für die Integration ist die Phase der Projektplanung im industriellen Bereich aber unverzichtbar. Deshalb lautet der zweite Integrationsansatz wie folgt:

IA St.1/Nr.2 Ein Projektplan ist im industriellen Kontext zwingend erforderlich, um die Phasen einer Studie zeitlich zu terminieren, Budgetfreigaben für Software und Hardware zu erhalten oder qualifizierte Mitarbeiter für die Studie zu arrangieren bzw. freizustellen.

Tab. 4.2: Phasen der Aufgabenanalyse

	Vorgehensmodelle / Vorgehensweise				
Phasen	**Banks**	**Law**	**Sargent**	**ASIM**	**MSG DF**
1. Aufgabenanalyse					
1.1 Problemformulierung	☑	☑	☑	☑	
1.2 Zielstellung	☑	☑	☑	☑	☑
1.3 Projektplan	☑	☑		☑	

4.1.2 Modellformulierung

In allen vorgestellten klassischen Vorgehensmodellen enthält die Modellformulierung die Entwicklung des Modellkonzeptes (s. Tabelle 4.3 - Ph. 2.1). Das Konzept beinhaltet die Planung von Eingangsparametern und Zielgrößen, die Festlegung des Detaillierungsgrades und grundsätzlichen Funktionsweisen und Steuerungsstrategie (vgl. Abschnitt 2.1.3).
Die ersten beiden DF-Bereiche beschreiben hingegen direkt die Implementierung des Modells, ohne die vorherige Entwicklung eines Konzeptes. Dieses ist für die Validierung aber zwingend erforderlich, um bspw. zu überprüfen, ob der vereinbarte Detaillierungsgrad technisch umgesetzt werden konnte. Das Konzept dient daher als Indikator für die erfolgreiche Implementierung und darüber hinaus als Absicherung für Projektbeteiligte im Falle eines Fehlers. Das Modellkonzept ist für das Zielmodell aus den genannten Gründen zwingend erforderlich.

IA St.1/Nr.3 Ein Modellkonzept ist für die Validierung des realisierten Modells zwingend erforderlich.

In den klassischen Vorgehensmodellen folgt unmittelbar nach der Konzeptionierung des Modells die Implementierung. Ein zentraler Bestandteil von DF-Modellen ist hingegen der Einsatz der HPC-Komponente, wodurch eine komplexere Architektur entsteht. Zusätzlich soll das Frontend und damit alle Aktionen der DFS über Web-Services realisiert werden, die ebenfalls über Schnittstellen mit dem Modell und dem Backend interagieren müssen. Während viele Simulationen nach klassischen Vorgehensweisen mit einer ausgereiften Simulationssoftware (z. B. Arena, AutoMod, Dosimis-3) durchgeführt werden, bei denen von der Implementierung bis zur Analyse alle notwendigen Funktionen in der Software enthalten sind, erweisen sich diese integrierten Lösungen für das DF als ungeeignet. Auf die Gründe dafür wird im folgenden Abschnitt 4.1.3 eingegangen. Die Architektur eines DF-Modells für SCs umfasst daher unterschiedliche Komponenten verschiedener Plattformen, die alle

durch Schnittstellenformate miteinander kommunizieren müssen. Dieser Ansatz erfordert eine ausführliche Planung der Systemarchitektur, um die allgemeine Kompatibilität des Systems zu gewährleisten. Diese Systemarchitektur wird im erforderlichen Ausmaß in keinem Vorgehensmodell und auch nicht in den NATO-Berichten beschrieben (s. Tabelle 4.3 - Ph. 2.2).

IA St.1/Nr.4 Es ist eine Phase für die Aufstellung der Systemarchitektur zur Strukturplanung des gesamten IT-Systems erforderlich. Darin enthalten sind die Planung für die Realisierung des Front- und Backends, der Einsatz der HPC-Komponente und die Schnittstellengestaltung inklusive erforderlicher Datenformate.

Tab. 4.3: Phasen der Modellformulierung

	Vorgehensmodelle / Vorgehensweise				
Phasen	**Banks**	**Law**	**Sargent**	**ASIM**	**MSG DF**
2. Modellformulierung					
2.1 Modellkonzept	☑	☑	☑	☑	
2.2 Systemarchitektur					

4.1.3 Modellimplementierung

Zentraler Bestandteil der Modellimplementierung ist die Umsetzung des Modellkonzeptes in ein ausführbares Modell. In allen klassischen Vorgehensmodellen aus Abschnitt 2.1.3 werden begleitend zur Implementierung notwendige Stammdaten beschafft und für das Simulationsmodell aufbereitet. Die MSG beschreibt in den sechs DF-Bereichen zwar keine Aktivitäten zur Beschaffung externer Simulationsdaten, in der Architektur der DFS aus Abschnitt 3.2.3 wird aber Bezug auf die Verknüpfung des Modells mit den Unternehmensdatenbanken genommen. Erforderliche Simulationsdaten könnten auf diese Weise direkt für das Modell zugänglich gemacht werden. In Tabelle 4.4 ist die Phase *3.1 Datenbeschaffung/-aufbereitung* aus diesem Grund eingeschränkt vorhanden.

Ein wesentlicher Unterschied zwischen Simulationsstudien klassischer Vorgehensmodelle und DF-Studien ist die Modellierungstiefe während der Implementierung. Bei klassischen Simulationen im Kontext der Produktion und Logistik wird hauptsächlich moderne Simulationssoftware verwendet, die grafisch anschauliche Editoren und Kataloge, mit bereits implementierten Bausteinen besitzen. In vielen Simulationsstudien im produktionslogistischen Kontext beschreibt die Implementierung daher nicht die fundamentale Entwicklung

des Simulationskerns, sondern die Konstruktion und Parametrisierung des Szenarios aus dem Modellkonzept in der Simulationssoftware. Für eine klare Begriffsabgrenzung wird die Erstellung des Szenarios aus dem Modellkonzept mit einem bereits vorhandenen Simulationskern als Phase der Modellierung bezeichnet. Diese Phase ist sowohl in allen klassischen Vorgehensmodellen als auch in DF-Studien präsent (s. Tabelle 4.4 - Ph 3.4). Obwohl moderne Simulationssoftware einem Anwender eine einfache Modellierung mit vielen Funktionalitäten ermöglicht, erweist sie sich für das DF als ungeeignet. Der Grund dafür sind notwendige Anpassungen an dem Simulationskern der Software, um das Modell mit einer HPC-Komponenten zu verknüpfen bzw. das Modell in der HPC-Umgebung auszuführen. Der Simulationskern genannter Software ist in den meisten Fällen aber nicht durch den Anwender veränderbar.
In DF-Studien gilt es aus diesem Grund, vor der Modellierung zunächst den Simulationskern im Rahmen einer Modellentwicklung zu implementieren, um die Fähigkeit des HPC zu gewährleisten (s. Tabelle 4.4 - Ph 3.2). Die Modellentwicklung ist in klassischen Vorgehensmodellen im Kontext der Produktion und Logistik nur eingeschränkt vorhanden, weil die Phase der Implementierung sowohl die Modellentwicklung (Ph. 3.2) als auch die Modellierung (Ph 3.4) zusammen betrachtet und deshalb kein detailliertes Vorgehen zur Implementierung des Modellkerns beschreibt (vgl. Abschnitt 2.1.3 - Law Ph.4).
Da die Struktur eines DF-Modells mit der erforderlichen HPC-Komponente aber eine wesentlich komplexere Architektur besitzt, verfolgt die MSG den ähnlichen Ansatz moderner Simulationssoftware zur stringenten Trennung des Szenarios und Simulationskerns, damit das DFS-Portal auch für andere Szenarien verwendet werden kann. Allerdings wird dabei auch gleichzeitig die erforderliche HPC-Fähigkeit des Modells berücksichtigt.
Im Hinblick auf SCs würde das Modell innerhalb des Simulationskerns die Eigenschaften und Funktionen aller Akteure enthalten und das „Gerüst“ umfassen, mit dem die Akteure miteinander interagieren können. Das Szenario stellt hingegen das „Terrain“ der Lieferkette mit der Anzahl, der Art und den Standorten der Akteuren dar.
In Abbildung 4.1 ist die benötigte Abgrenzung zwischen dem Modell und dem Szenario schematisch am Beispiel einer einfachen SC dargestellt. Das Szenario enthält jeweils einen Lieferanten, einen Hersteller und einen Kunden, die über definierte Parameter miteinander interagieren. So werden im Szenario Parameter, wie die Anzahl der Bestellten Produkte des Kunden, die Produktionsraten des Herstellers oder Transportzeiten festgelegt, während die logische Ausführung und die Eintragungen der Simulationsdaten im Backend vom Modell ausgeht und im Simulationskern enthalten sind. Die Integration von DFS erfordert daher die Entwicklung modularer Szenarien, um bereits bestehende Modelle vielseitig einsetzen zu können (Tabelle 4.4 - Ph. 3.3).

IA St.1/Nr.5 Analog zum ersten DF-Bereich *Rapid szenario prototyping* ist im Vorgehensmodell eine Phase „Szenariogestaltung“ zur Entwicklung, Änderung und Einbindung von Szenarien erforderlich.

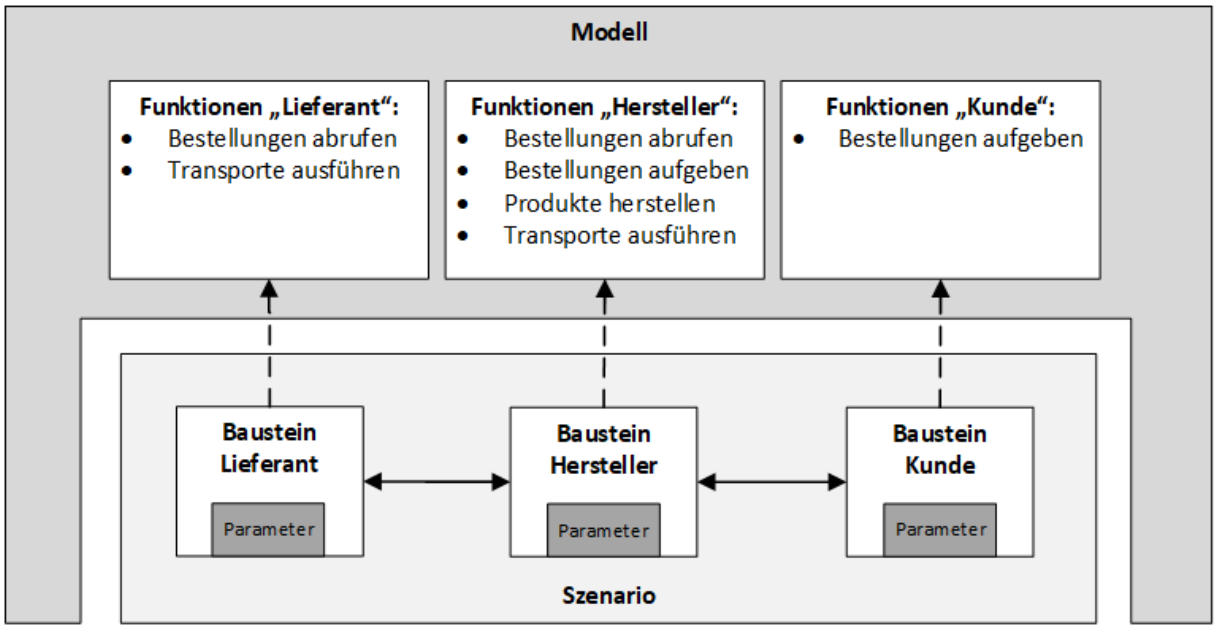

Abb. 4.1: Abgrenzung zwischen Modell und Szenario am Beispiel der Supply Chain

Auch an dieser Stelle ist die deutliche Abgrenzung zwischen der Szenariogestaltung (Ph. 3.3) und der Modellierung (Ph. 3.4) wichtig. Ersteres beschreibt die technische Realisierung aller Funktionen, um Szenarien intuitiv zu erstellen und durch eine Schnittstelle mit dem Modellkern simulieren zu können. Letzteres ist hingegen die Modellierung des Szenarios aus dem Modellkonzept und folgt im Anschluss an die Modellentwicklung und Szenariogestaltung. Im Anschluss an die Implementierung gilt es das Modell nach der geplanten Systemarchitektur aus **IA St.1/Nr.4** mit dem Frontend, dem Backend und der HPC-Komponente zu vernetzen. Derartige Tätigkeiten sind in den Vorgehensmodellen nach Banks, Law, Sargent oder der ASIM nicht enthalten, weil gewöhnliche Simulationsstudien bereits im Vorfeld nicht so eine komplexe Struktur wie DF-Systeme besitzen (Tabelle 4.4 - Ph. 3.4).

IA St.1/Nr.6 Für die Realisierung der Systemarchitektur aus **IA St.1/Nr.4** wird eine Phase zur Gestaltung sämtlicher Schnittstellen benötigt. Das Ergebnis der Schnittstellengestaltung ist ein funktionsfähiges DFS-Portal.

Tab. 4.4: Phasen der Modellimplementierung

	Vorgehensmodelle / Vorgehensweise				
Phasen	**Banks**	**Law**	**Sargent**	**ASIM**	**MSG DF**
3. Modellimplementierung					
3.1 Datenbeschaffung / -aufbereitung	☑	☑	☑	☑	(☑)
3.2 Modellentwicklung	(☑)	(☑)	(☑)	(☑)	☑
3.3 Szenariogestaltung					☑
3.4 Modellierung	☑	☑	☑	☑	☑
3.5 Schnittstellengestaltung					☑

4.1.4 Modellanwendung

In der Anwendung von Simulationsmodellen sind in allen Vorgehensweisen deutliche Parallelen in den Phasen der Experimentenplanung und -durchführung und der anschließend folgenden Analyse der Ergebnisse zu erkennen (s. Tabelle 4.5). In der Kategorie der *Modellanwendung* müssen daher keine Integrationsansätze der ersten Stufe abgeleitet werden.

Tab. 4.5: Phasen der Modellanwendung

	Vorgehensmodelle / Vorgehensweise				
Phasen	**Banks**	**Law**	**Sargent**	**ASIM**	**MSG DF**
4. Modellanwendung					
4.1 Experimentenplanung	☑	☑	☑	☑	☑
4.2 Durchführung der Experimente	☑	☑	☑	☑	☑
4.3 Ergebnisanalyse	☑	☑	☑	☑	☑

4.1.5 Phasenübergreifende Tätigkeiten

Neben den festen Phasen eines Vorgehensmodells gibt es zusätzlich Tätigkeiten einer Simulationsstudie, die nicht statisch einer einzelnen Phase zugeordnet werden können.

Diese Aufwände gliedern sich in Tätigkeiten zur Verifikation und Validierung, der Dokumentationserstellung und projektinterner bzw. -externer Kollaboration.
Besonders in Bezug auf die Größe und Komplexität eines DFS-Portals besitzen V&V-Tätigkeiten eine hohe Bedeutung, auch wenn die MSG diesen wichtigen Punkt nur oberflächlich behandelt (s. Tabelle 4.6). In einer Projektübersicht der NATO wurden von acht DF-Systemen lediglich drei Modelle teilweise verifiziert und validiert [NAT14, S. 44]. Nach Aussagen der MSG beziehen sich V&V-Tätigkeiten ausschließlich auf die Überprüfung des Szenarios. Elemente zur Validierung der Simulationsdaten, der Ergebnisse oder der Modellarchitektur werden gar nicht berücksichtigt.
Im Vergleich dazu wird die Verifikation und Validierung in klassischen Vorgehensmodellen vorbildlich praktiziert. Während V&V-Tätigkeiten in dem Vorgehen von Banks noch relativ zentral nach der Implementierung des Modells vorgeschrieben sind, empfiehlt Law bereits nach der Konzeptionierung einen ausführlichen Vergleich zur Zielstellung, um die Abstraktionsebene des Modellkonzeptes beurteilen zu können (vgl. Abschnitt 2.1.3). Noch allgegenwärtiger ist die V&V aber in den Modellen von Sargent und der ASIM, bei denen jedes entstehende Phasenergebnis einer Studie ausführlich auf die Richtigkeit hin überprüft wird (vgl. Abschnitt 2.1.3).
Im Hinblick auf klassische Vorgehensweisen sollten auch DF-Systeme hinreichend genau überprüft werden, um Entwicklungsfehler frühzeitig erkennen und beheben zu können. Eine wichtige Phase für den Erfolg bzw. Nichterfolg einer Simulationsstudie ist die Zielstellung (Ph. 1.2), da sie die Basis für alle folgenden Phasen ist. Aus der Zielstellung wird der Detaillierungsgrad für das Modellkonzept abgeleitet, welches wiederum die Grundlage für die Implementierung und damit auch für die Qualität der Simulationsergebnisse ist. In dem integrierten Vorgehensmodell ist daher ein ausführlicher Vergleich zwischen dem Modellkonzept (Ph. 2.1) und der Zielstellung (Ph. 1.2) durchzuführen. Weiterhin gilt es das realisierte Modell nach der iterativen Modellentwicklung (Ph. 3.2) und Szenariogestaltung (Ph. 3.3) zu validieren. Dafür ist zu überprüfen, ob das Simulationsmodell eine hinreichend genaue Abbildung der SC darstellt und ob das geplante Ziel der Simulation mit dem Modell erreicht werden kann. Im Gegensatz zu klassischen Vorgehensweisen ist bei DF-Studien zusätzlich zur V&V des Modells auch die V&V des gesamten IT-Systems nötig. Dafür muss das gesamte System nach der Schnittstellengestaltung (Ph. 3.5) zum einen auf die Funktionsweise überprüft und zum anderen mit der geplanten Systemarchitektur aus Phase 2.2 verglichen werden. Zusätzlich sind erforderliche Stammdaten der SC im Zuge einer permanenten Datenvaldierung auf eine ausreichende Detaillierung und Qualität zu überprüfen. Nach dem Abschluss der Experimente muss ebenfalls beurteilt werden, ob die generierten Ergebnisdaten ausreichend sind, um das Zielsystem auf Basis der gesamten Ergebnislandschaft optimieren zu können. Wenn die Zielstellung nicht erfüllt werden kann, müssen neue Experimente mit einer korrigierten Versuchsplanung durchgeführt werden.

IA St.1/Nr.7 Die Konzeptionierung, Entwicklung und Anwendung von DFS in einer SC erfordert die permanente V&V des Modellkonzeptes, des realisierten Modells, des DF-Systems, der erforderlichen SC-Stammdaten und der generierten Simulationsdaten.

Einer der größten Vorteile modular konstruierter DF-Systeme besteht darin, das Modell für eine Zielstellung nur einmalig zu entwickeln und dieses an beliebige SCs anpassen zu können. Der potentielle Benutzerkreis des DF-Systems könnte dadurch wesentlich größer als bei Simulationsmodellen sein, die nur auf ein spezielles Szenario zugeschnitten sind. Die ausführliche Dokumentation von DF-Systemen ist daher unverzichtbar. Banks unterscheidet in Abschnitt 2.1.3 zwischen Programm- und Prozessdokumentationen. Während Ersteres als Inhalt der Modellentwicklung (Ph. 3.2) in den Integrationsansätzen der zweiten Stufe behandelt wird, muss die Prozessdokumentation begleitend während des Projektes erweitert werden. Der Umfang der Prozessdokumentation sollte sich an die Vorgehensmodelle von Banks oder Law orientieren. Bei Sargent ist die Dokumentation nämlich kein Bestandteil des Vorgehensmodells und in der Norm der VDI bezieht sich die Dokumentation im Modell der ASIM lediglich auf die Durchführung der Experimente (s. Tabelle 4.6). Da eine Prozessdokumentation ebenfalls nicht in den Vorgehensweisen der MSG aufgeführt ist, kann folgender Integrationsansatz abgeleitet werden:

IA St.1/Nr.8 Die Konzeptionierung, Entwicklung und Anwendung von DFS in einer SC erfordert die kontinuierliche projektbegleitende Entwicklung einer ausführlichen Prozessdokumentation, um technische Umsetzungen oder getroffene Entscheidungen rückwirkend nachvollziehen zu können.

Der letzte relevante Punkt, der sich nicht auf einzelne Phasen sondern auf die gesamte Simulationsstudie bezieht, ist die Kollaboration. DF ist, verglichen mit den bisher bekannten Simulationsstudien der Produktion und Logistik, wie sie in Kapitel 2 beschrieben sind, keinesfalls Stand der Technik, wodurch als aktuelles Forschungsthema noch viel unentfaltetes Potential vorhanden ist. Dieses Potential kann durch die Zusammenarbeit und den Austausch von Simulationsexperten unterschiedlicher Nationen und Projekte, wie es im sechsten DF-Bereich beschrieben wird, wirksam gefördert werden. Der Autor empfiehlt daher den Kollaborationsgedanken der MSG auch im produktionslogitischen Kontext weiter zu verfolgen, prognostiziert gleichzeitig aber auch Probleme in der Zusammenarbeit mit Unternehmen anderer SCs aufgrund der vorhandenen Wettbewerbspolitik im unternehmerischen Umfeld. Ein Indiz dafür liefert die Betrachtung der klassischen Vorgehensmodelle, bei denen in keinem Modell eine Phase für den externen Austausch beschrieben wird. Im Anhang, in den Abbildungen A.1 bis A.2, ist die gesamte Konzeptmatrix inklusive aller Integrationsansätze der ersten Stufen zusammenfassend und übersichtlich dargestellt. Die orange markierte Phase *Problemformulierung (Ph. 1.1)* wird in einem Vorgehensmodell für die Integration von DFS aufgrund von **IA St1./Nr.1** nicht aufgenommen. Die blau hinterlegten Bereiche sind Phasen, die entweder nur klassischen Vorgehensmodellen oder in DF-Studien vorhanden sind und durch genannte Integrationsansätze in das Zielmodell mit aufgenommen werden. Die einzige Ausnahme sind die V&V-Tätigkeiten, die in DF-Studien aber in so geringem Maße vertreten sind, dass dort auch deutliche Integrationsansätze erforderlich sind. Die Phasen ohne farbliche Kennzeichnung sind in beiden Arten von Simulationsstudien vorhanden.

Tab. 4.6: Phasenübergreifende Tätigkeiten

	Vorgehensmodelle / Vorgehensweise				
Phasenübergreifende Tätigkeiten	**Banks**	**Law**	**Sargent**	**ASIM**	**MSG DF**
Verifikation und Validierung	☑	☑	☑	☑	(☑)
	Nach 3.4	Nach 2.1, 3.4	Nach jedem Phasen-ergebnis	Nach jedem Phasen-ergebnis	Nach 3.4
Prozessdokumentation	☑	☑		(☑)	
Kollaboration					☑

Phasen des Vorgehensmodells Aus den Integrationsansätzen der ersten Stufe geht ein „grobes“ Gerüst für ein Vorgehensmodell hervor, das in Tabelle 4.7 abgebildet ist. Während sich die Kategorien der Phasen in der Konzeptmatrix aus den Abbildungen A.1 bis A.2 noch an den gemeinsamen Elementen der klassischen Vorgehensmodelle orientieren, wurden die Kategorien *Modellformulierung* und *Modellimplementierung* des Vorgehensmodells in *Konzeptionierung* und *Realisierung* umbenannt. Dies erfolgt aus dem Grund, dass DFS nicht allein aus dem Modell, sondern aus einem komplexen Data Farming Services System (DFSS) bestehen, in dem das Modell neben der HPC-Komponente und dem DFS-Portal nur ein Teil des Systems ist.

Tab. 4.7: Phasen des Vorgehensmodells zur Integration von DFS

Kategorie	Phase
Aufgabenanalyse	**1.** Zielstellung
	2. Projektplan
Konzeptionierung	**3.** Modellkonzept
	4. Systemarchitektur
Realisierung	**5.** Datenbeschaffung / -aufbereitung
	6. Modellentwicklung
	7. Szenariogestaltung
	8. Modellierung
	9. Schnittstellengestaltung
Modellanwendung	**10.** Experimentenplanung
	11. Durchführung der Experimente
	12. Ergebnisanalyse

4.2 Integrationsansätze Stufe II - Anforderungen

Die potentiellen Phasen für ein Vorgehensmodell, die in Tabelle 4.7 als Ergebnis der Integrationsansätze Stufe I ausgearbeitet wurden, werden für den folgenden Abschnitt als Grundlage verwendet. Für die Integrationsansätze der Stufe II werden Anforderungen an die Inhalte der jeweiligen Phasen gestellt. Die Kategorien der Anforderungen gliedern sich in ablaufbedingte und technische Anforderungen bzgl. der Funktionen, der Hardware und der Software des DFSS. Falls die Phasen, die auch in klassischen Vorgehensmodellen enthalten sind, nicht den Anforderungen für DFS entsprechen, werden Integrationsansätze der zweiten Stufe abgeleitet.

4.2.1 Aufgabenanalyse

Während der *Aufgabenanalyse* sind ausschließlich ablaufbedingte Anforderungen von Bedeutung, weil zunächst nur der Planungsaspekt im Vordergrund steht. Bei der Zielstellung (Ph. 1) besteht eine der größten Herausforderungen des Projektteams in einer genauen Definition von Zielgrößen, anhand derer das Ergebnis der Simulation gemessen werden kann. Im Vergleich zu klassischen Simulationen sind die Zielwerte der Zielgrößen nicht von Bedeutung. In Bezug auf die Zielgrößen lautet der Ansatz von DF zunächst „so gut wie

möglich“. Festgelegte Zielwerte sind bspw. für simulationsgestütze Optimierungen relevant, weil sie als Abbruchkriterium der Simulation verwendet werden können. In DF-Modellen hat ein zuvor festgelegter Zielwert keinen Einfluss auf das Erreichen oder Nichterreichen des Ziels.
Eine hingegen erforderliche Zielstellung ist die Festlegung der Gewichtungen (bei mehreren Zielgrößen), gerade wenn sich diese gegenseitig beeinflussen. Wenn in einer SC-Simulation sowohl die Lieferzeit als auch die Transportkosten betrachtet werden, wird es wahrscheinlich keine Lösung mit optimalen Ergebnissen für beide Zielgrößen geben. Eine reaktive SC bietet zwar minimale Lieferzeiten, kann dieses Ziel aber nur durch zusätzliche und nicht voll ausgelastete Transporteinheiten bewältigen, wodurch hingegen die Kosten der Lieferung steigen. Es muss daher entschieden werden, welche Zielgröße mit welcher Gewichtung priorisiert wird. Zusätzlich sollte aus allen Zielgrößen ein wirtschaftliches Gesamtziel hergeleitet werden können. In Bezug auf den industriellen Kontext und das SCM sollte das Gesamtziel in monetären Kennzahlen formuliert werden. Die Gewichtung der Zielgrößen und das Gesamtziel sind für die Konzeptionierung, Realisierung und auch für die Durchführung der Experimente zunächst nicht von Bedeutung. Im Hinblick auf die Ergebnisanalyse werden diese aber benötigt, um das beste Ergebnis identifizieren zu können.

Die zweite Phase beinhaltet die Aufstellung des Projektplans (Ph. 2), dessen Anforderungen im wesentlichen auf die gesamte Terminierung des Projektes, die Zuweisung von Rollen und Zuständigkeiten und die Budgetplanung für Hardware, Software und Personal bezieht. In Bezug auf die Integrationsansätze der ersten Stufe gilt es im Projektplan auch die Zuständigkeit für die Programm- bzw. Prozessdokumentation festzulegen. In Tabelle 4.8 sind die Anforderungen der Aufgabenanalyse abgebildet. Diese werden durch die Beschreibung der Zielstellung und des Projektplans klassischer Vorgehensmodelle ausreichend erfüllt, wodurch für die Aufgabenanalyse keine Integrationsansätze erforderlich sind.

Tab. 4.8: Anforderungen der Aufgabenanalyse

Phasen		Anforderungen für DFS in einer SC		Anforderungen erfüllt?
		Ablaufbedingte Anforderungen	Technische Anforderungen	
Aufgabenanalyse	1. Zielstellung	• Definition und Gewichtung von Zielgrößen • Gesamtziel muss wirtschaftlich beurteilt werden können	-	☑
	2. Projektplan	• Terminierung einzelner Phasen • Zuweisung von Rollen und Zuständigkeiten • Budgetplanung	-	☑

4.2.2 Konzeptionierung

Die Konzeptionierung erfolgt, genau wie auch anschließend die Realisierung, von „innen nach außen“. Der Kern des Systems ist das Modell, welches zunächst über das Modellkonzept (Ph. 3) ausgelegt wird. Typische Eigenschaften, wie die Geschwindigkeit und Transparenz von DF-Modellen, müssen bereits in der Konzeptionierung berücksichtigt werden. Um die Geschwindigkeit eines Simulationslaufs zu erhöhen, können entweder die zur Verfügung stehenden Rechenressourcen erhöht werden oder es kann die Komplexität des Problems reduziert werden. Zur Nutzung der maximalen Leistung empfiehlt es sich jedoch beide Faktoren zu beeinflussen. Während die Rechenleistung über das HPC gesteigert wird, empfiehlt es sich die Regeln und Strategien des Systemverhaltens so einfach wie möglich zu formulieren, um einerseits Transparenz zwischen den Faktoren und den Zielgrößen zu gewährleisten und darüber hinaus Berechnungszeiten einzusparen. Je detaillierter und realitätsnäher ein System dafür abgebildet wird, desto komplexer wird das Systemverhalten und dementsprechend höher die Simulationszeit eines Laufs. Während die ASIM den Leitpfaden aufstellt „so abstrakt wie möglich“ zu modellieren, muss die Vorgabe für das DF lauten „eine hohe Abstraktion zu garantieren“:

IA St.2/Nr.1 Die Konzeptionierung des DF-Modells muss auf einer **hohen** Abstraktionsebene erfolgen.

Im Anschluss an das Modellkonzept werden in der Systemarchitektur (Ph. 4) alle weiteren notwendigen Komponenten und deren Verknüpfung mit dem Modell geplant, die für den Betrieb des DFS-Portals notwendig sind. In Abbildung 4.2 ist eine vereinfachte Systemarchitektur mit den grundlegenden Funktionen eines DFS-Portals abgebildet. Die Struktur ist in die Bereiche *Frontend, Laufzeit* und *Backend* eingeteilt. Die Laufzeit verarbeitet sämtliche, durch den Benutzer ausgeführte Aktionen und fungiert als Vermittler zwischen dem Frontend und dem Backend. Die wesentlichen Komponenten des Systems sind das DFS-Portal, die HPC-Komponente, das Modell, das Szenario und die SC- bzw. Simulationsdatenbanken.
In der Konzeptionierung der Systemarchitektur besteht die Anforderung darin den Einsatz aller Komponenten und der in Abbildung 4.2 dargestellten Verbindungen zwischen den Komponenten so zu planen, dass theoretisch eine Interoperabilität gewährleistet ist. Dabei sollte zunächst beantwortet werden können in welcher Form und von welchem Anbieter die HPC-Komponente realisiert wird.
Das Angebot von HPC-Komponenten, in Form von Clustern oder Supercomputern, wächst stetig. Für die Entwicklung eines DFSSs sind von kostenlosen Open-Source-Projekte bis hin zu professionellen kostspieligen Services von Microsoft Azure [Mic21] oder Amazon [Ama21] viele Lösungen möglich. Dabei ist die Architektur aus Abbildung 4.2 nur eine vieler möglicher Varianten. Beispielsweise kann der Grad des Outsourcings je nach gewählter HPC-Komponente unterscheiden. Die MSG realisierte die Modellausführung über Batchskripte,

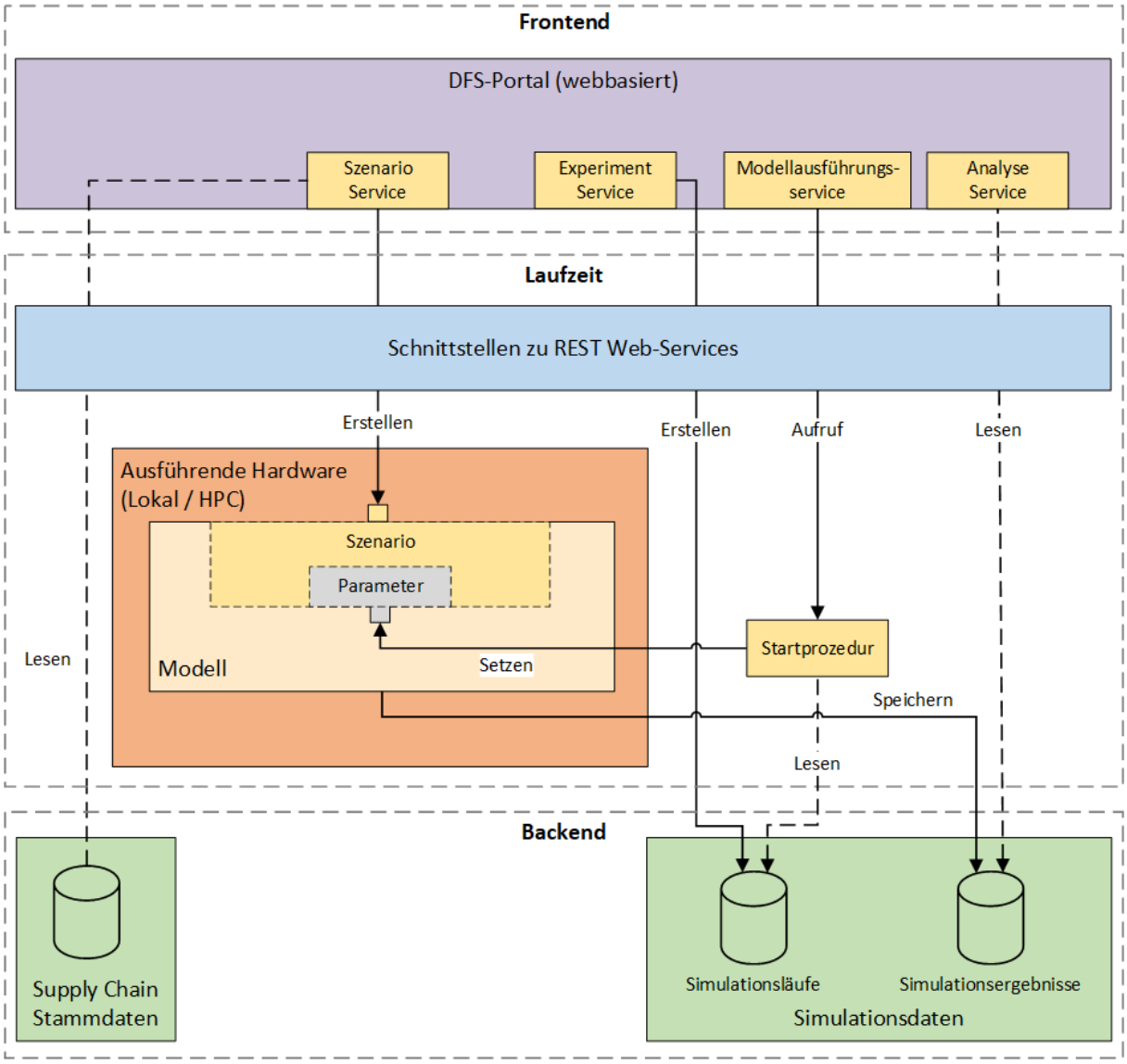

Abb. 4.2: Grundstruktur eines Data Farming Service Systems für eine Supply Chain

die an die HPC-Komponente übermittelt wurden. In MATLAB ist es hingegen möglich sog. „Batch-Jobs“ direkt in der MATLAB-Umgebung als Funktion aufzurufen [Mat21a], während in Microsoft Azure Lösungen möglich sind, bei denen alle Komponenten eines DFSSs innerhalb der Azure-Cloud miteinander kommunizieren können.
Nach der Auswahl einer HPC-Komponente sollte ebenfalls berücksichtigt werden, wie diese „Befehle“ des Frontends erhält und auf welche Weise die Kommunikation mit dem Backend stattfindet. In den ersten DF-Modellen erfolgte die Rückgabe der Ergebnisse durch ASCII-Dateien, die zurück an den lokalen Computer gesendet wurden, von dem die Simulation aus gestartet wurde (vgl. Abschnitt 3.2.1). Je nach Bedarf mussten diese Dateien für die Analyse in einem weiteren Schritt aber entweder in Datenbanken oder

direkt in die verwendete Analysesoftware importiert werden.
Heutzutage sind wesentlich effizientere Schnittstellengestaltungen möglich, bei denen die HPC-Komponente die Ergebnisse direkt in der Datenbank speichert auf die auch die Analysesoftware zugreift. Programme wie Minitab oder MATLAB bieten Web-Apps oder Web-Services an, welche die Verwendung statistischer Werkzeuge ermöglichen ohne, dass die erforderliche Software auf dem lokalen Computer des Anwenders installiert ist [Mat21b] [Add21]. Der Integrationsansatz für die Systemarchitektur lautet daher:

IA St.2/Nr.2 Für die Systemarchitektur müssen verschiedene HPC-Dienste miteinander verglichen und bewertet werden. Zur Gewährleistung der allgemeinen Interoperabilität ist die Durchführung einer Machbarkeitsstudie erforderlich.

Tab. 4.9: Anforderungen an die Konzeptionierung

Phasen		Anforderungen für DFS in einer SC		Anforderungen erfüllt?
		Ablaufbedingte Anforderungen	Technische Anforderungen	
Konzeptionierung	3. Modellkonzept	• Regeln und Strategien für das Systemverhalten so einfach wie möglich formulieren	-	☒
	4. System-architektur	• Planung aller Funktionalitäten des DFS-Portals inkl. der HPC-Komponente • Interoperabilität gewährleisten	-	-

4.2.3 Realisierung

Wie auch in dem Vorgehensmodell der ASIM sollte die Datenbeschaffung und -aufbereitung (Ph. 5) parallel während der gesamten *Realisierung* erfolgen. Grundsätzlich ist die Durchführung der Datenbeschaffung stark von dem Vernetzungsgrad und den IT-Systemen der SC-Akteure und der Zielstellung der Simulationsstudie abhängig. Die erforderlichen Daten für eine SC-Simulation sind im Wesentlichen die Stammdaten (Lage, Sachnummern der Produkte, Preise, Mindest- und Sicherheitsbestände, Anzahl und Art von Transporteinheiten, usw.) der SC-Akteure, die wichtig für die Erstellung des Szenarios sind [Koe11, S. 74]. Die Beschaffung von Stammdaten ist in der Regel unkritisch, weil sie von der Menge her überschaubar und verglichen mit Ereignisdaten (Aufträge, Termine, Mengen, usw.) relativ statisch sind [Koe11, S. 84]. Falls in der betrachteten SC keine interorganisationalen Datenbanken in Form von integrierten Cloud-Lösungen vorhanden sind, können Stammdaten auch leicht über Extract, Transform, Load (ETL)-Prozesse beschafft werden.
Nicht zwangsläufig notwendig, aber hilfreich, sind hingegen aber auch dynamische Ereignisdaten von SCs, um die Parameterbereiche eines Experimentes in der Versuchsplanung

präzise festzulegen. Aufgrund der Dynamik von Prozessdaten würden verwendete ETL-Prozesse nur eine Momentaufnahme zum Zeitpunkt des Exportes widerspiegeln. Wenn die Simulation zu einem späteren Zeitpunkt erneut ausgeführt wird, würden Modellparameter auf Basis extrahierter Prozessdaten nicht dem aktuellen Stand entsprechen.
Wenn stets aktuelle Prozessdaten für die Simulation benötigt werden, sind in einer SC interorganisationale Datenbanken mit einer einheitlichen Datenstruktur erforderlich, die direkt an das DFS-Portal angebunden werden können (s. Abbildung 4.2). Eine ablaufbedingte Anforderung besteht aber darin, zuvor eine klare Erkenntnis über die Art und den Umfang erforderlicher Daten auszuarbeiten (s. Tabelle 4.10). Unabhängig von dem Bedarf von Prozessdaten ist für die Datenbeschaffung eine enge Zusammenarbeit der gesamten SC erforderlich, die in vielen Lieferketten nicht im ausreichenden Ausmaß vorhanden ist (vgl. Abschnitt 2.2).

IA St.2/Nr.3 Ein Ansatz zur Integration ist deshalb die Unterstützung und Förderung kollaborativer Planung mit allen Akteuren einer SC.

Dieser Ansatz ist nicht nur ausschließlich für die Datenbeschaffung, sondern auch für die gesamte Simulationsstudie äußerst wichtig. Alle Akteure sollten über die Planung und Durchführung und auch über Ziele und mögliche Optimierungspotentiale in Kenntnis gesetzt werden. Die Optimierung der Lieferkette muss im kollektiven Interesse stehen, damit die Erkenntnisse einer DF-Studie realisiert werden können.

Die nächste Phase ist die Modellentwicklung (Ph. 6) für die Implementierung des in Abschnitt 4.1.3 erwähnten Simulationskerns. Die Herausforderungen bestehen darin, das im Konzept enthaltene Systemverhalten und den geplanten Detallierungsgrad einzuhalten. Sowohl für die Modellentwicklung als auch für die Gestaltung des Szenarios ist die Zusammenarbeit mit den Fachexperten, also den zuständigen Personen des SCM erforderlich, um Fehler in der Abbildung des Systems zu vermeiden.
Ein DF-Modell unterliegt vielen technischen Anforderungen. Um die erforderliche Geschwindigkeit der Simulation technisch umzusetzen, ist eine „schlanke“ Implementierung erforderlich. Der Begriff „schlank“ impliziert in diesem Kontext, einen auf das Konzeptmodell zugeschnittenen Programmcode, indem nur wirklich erforderliche Bedingungen geprüft werden, um die Performance zu maximieren. Eine weitere Eigenschaft ist die Fähigkeit des Modells über externe Hochleistungshardware ausgeführt werden zu können (HPC-Fähigkeit). Dafür ist nicht nur die Adaptierbarkeit zwischen dem Modell und dem Szenario erforderlich, sondern das gesamte Modell muss unabhängig von spezieller Simulationssoftware auf einer externen Hardware ausgeführt werden können. Wie bereits in Abschnitt 4.1.3 angesprochen wurde, erfüllen bspw. moderne Simulationsprogramme mit modularen Bausteinkonzepten dafür nicht den Anforderungen an die HPC-Fähigkeit, weil der Simulationskern nicht angepasst werden kann. SC-Simulationen wie Business games oder Simulationen mit Tabellenkalkulationsprogramme aus Abschnitt 2.2 sind aufgrund der fehlenden HPC-Fähigkeit ebenfalls nicht für DF-Studien geeignet.

IA St.2/Nr.4 Zur Gewährleistung der HPC-Fähigkeit muss der Simulationskern eines DF-Modells optimal auf das Modellkonzept abgestimmt sein. Zur Erreichung dieser Individualität ist die Implementierung in Programmiersprachen erforderlich.

Aufgrund des modularen Aufbaus von Modell und Szenario ist ein DFSS nach der Realisierung vielseitig einsetzbar und theoretisch ohne große Aufwände auf andere SCs übertragbar. Bei der Vernachlässigung politischer Restriktionen haben DFSSs wahrscheinlich einen höheren Benutzerkreis als klassische Simulationsmodelle. Zum Abschluss der Modellentwicklung ist daher die Verfassung einer Programmdokumentation erforderlich, in der die Eingabefaktoren und Zielgrößen definiert sind und sämtliche Funktionalitäten erläutert werden.

Iterativ mit der Modellentwicklung (Ph. 6) wird auch die Gestaltung des Szenarios (Ph. 7) implementiert. Die Szenariogestaltung enthält in der Architektur aus Abbildung 4.2 den *Szenario Service* und das Einbinden des Szenarios in das Modell. Der Szenario Service ist eine der drei fundamentalen Funktionen des DFS-Portals und bietet einen für den Benutzer intuitiven Szenario-Editor. In Bezug auf eine SC empfiehlt sich für den Szenario-Editor eine grafisch visualisierte Karte, auf die per Drag-and-Drop verschiedene Teilnehmer der SC gesetzt werden können. Weiterhin kann auf Basis von Stammdaten festgelegt werden, welche Lieferanten welche Rohstoffe bereitstellen und welche Hersteller welche Produkte herstellen. Zusätzlich müssen in dem Editor durch Verbindungen gekennzeichnet werden, welche Akteure sich gegenseitig beliefern. Ein fertiges Szenario sollte sich im Aufbau einer prozessorientierten Darstellung einer SC ähneln, wie sie in Abbildung 2.1 bereits abgebildet wurde.
Der Szenario-Editor sollte für die Anwender intuitiv gestaltet werden, damit schnelle Anpassungen der Szenarien möglich sind. Nach der Erstellung muss das visuell anschauliche Szenario in eine Maschinensprache (z. B. XML- oder ASCII-Format) kompiliert werden, um die Interoperabilität mit dem Modell sicherzustellen.

IA St.2/Nr.5 Entwicklung intuitiver Szenario-Editoren, mit denen Anwender schnell neue Szenarien erstellen bzw. bearbeiten und für das Modell bereitstellen können.

Während in der Szenariogestaltung (Ph.7) alle technischen Funktionalitäten bereitgestellt werden, um Szenarien vom Anwender intuitiv erstellen und in das Modell einbinden zu können, werden diese Funktionen während der Modellierung (Ph.8) verwendet, um das Szenario des Modellkonzeptes nachzubilden. An diese Phase gibt es keine technischen Anforderungen, weil dafür das fertige Modell verwendet wird, das aus der iterativen Zusammenarbeit der Modellentwicklung (Ph. 6) und Szenariogestaltung (Ph. 7) entstanden

ist (s. Tabelle 4.10 - Ph. 8). Nach dem Abschluss der Modellierungsphase wurde das Modellkonzept der dritten Phase erfolgreich in die Realität umgesetzt.

Die letzte Phase der Realisierung ist die Schnittstellengestaltung (Ph. 9), in der alle Komponenten der geplanten Systemarchitektur (Ph. 4) lauffähig miteinander vernetzt werden. Während der *Szenario Service* bereits in der Szenariogestaltung (Ph. 7) implementiert werden sollte, umfasst die Schnittstellengestaltung für eine ganzheitliche Kommunikation zwischen dem DFS-Portal und dem Modell auch die Realisierung des *Experiment*, *Modellausführungs* und *Analyse Service* (s. Abbildung 4.2).
Im *Experiment Service* kann ein Anwender den Versuchsplan einer Studie auswählen und die Faktorbereiche und die Anzahl der Faktorstufen für bestimmte Szenarien festlegen. Im Anschluss werden aus dem definierten Experiment die Parameterkombinationen aller Versuchsläufe erstellt und im Backend gespeichert. Die Simulation der geplanten Experimente wird im Anschluss über den *Modellausführungsservice* gestartet. Der *Analyse Service* stellt dem Anwender eine Sammlung statistischer Werkzeuge für die Analyse der Simulationsergebnisse bereit.
Das Ergebnis der Schnittstellengestaltung ist ein lauffähiges DFSS, mit dem Experimente über die HPC-Komponente ausgeführt werden können. Aufgrund der verhältnismäßig komplexen Architektur und der Anzahl vieler Komponenten, verglichen mit klassischen Simulationsmodellen, erfordert die Entwicklung eines DFSSs eine sehr hohe Qualifikation und eine starke Zusammenarbeit aller beteiligten Entwickler des Projektteams.

IA St.2/Nr.6 Für die Implementierung des DFSS sind hochqualifizierte Programmierer erforderlich.

Tab. 4.10: Anforderungen an die Realisierung

Phasen		Anforderungen für DFS in einer SC		Anforderungen erfüllt?
		Ablaufbedingte Anforderungen	Technische Anforderungen	
Realisierung	5. Daten-beschaffung / - aufbereitung	• Klare Erkenntnis über die Art und den Umfang erforderlicher SC-Daten	• Interorganisationale zentrale Datenbanken bei erforderlichen SC-Prozessdaten	☒
	6. Modell-entwicklung	• Einhaltung des Detaillierungsgrades • Verfassung einer Programmdokumentation • Zusammenarbeit mit den Fachexperten	• HPC-fähig • Modularer Aufbau • Hohe Geschwindigkeit • Hohe Transparenz	☒
	7. Szenario-gestaltung	• Zusammenarbeit mit den Fachexperten	• Schnittstellenformate für den Export und Import des Szenarios • Schnelle Anpassungen der Szenarien gewährleisten • Intuitive Gestaltung / Bedienung	☒
	8. Modellierung	• Umsetzung des Szenarios aus dem Modellkonzept	-	☑
	9. Schnittstellen-gestaltung	• Hohe Qualifikation und Zusammenarbeit der Simulationsexperten	• Gewährleistung der Datenbankkommunikation • Erforderliche Hardware für Webserver und On-Premises- / Cloud Datenbanken • Ausführung der Simulationsläufen über die HPC-Komponente	-

4.2.4 Modellanwendung

Zur Vorbereitung auf die Simulationsstudie gilt es in der Experimentenplanung (Ph. 9) einen Versuchsplan mit den Werten der Faktoren jedes Simulationslaufs aufzustellen. Das Ziel besteht darin, einen effizienten Überblick über die gesamte mögliche Ergebnislandschaft zu erstellen (s. Tabelle 4.11). Die in den klassische Simulationen beschriebenen dynamischen Faktor-Designs aus Abschnitt 2.1.2 erweisen sich für DF-Studien als ungeeignet, weil keine Übersicht der Ergebnislandschaft generiert wird, sondern vielmehr innerhalb einer unbekannten Landschaft nach einem definierten Ziel „gesucht“ wird. Der erste Integrationsansatz in Bezug auf die Experimentenplanung lautet daher:

IA St.2/Nr.7 Für die Versuchspläne von DF-Studien eignen sich ausschließlich What-if-Analysen mit statischen Faktor-Designs.

Weiterhin wurde in Abschnitt 3.2.2 im dritten DF-Bereich ebenfalls die Notwendigkeit alternativer Versuchspläne geschildert. In Bezug auf die Komplexität des Modells und

die Anzahl der Faktoren erweisen sich sequentielle Verzweigungen, Latin Hypercubes, frequenzbasierte oder kombinierte Designs als deutlich geeigneter als klassische Vollfaktor- bzw. Teilfaktor-Designs, da ein deutlich besseres Verhältnis zwischen dem Informationsgewinn und der Anzahl erforderlicher Simulationsläufe erreicht werden kann. Welches Design sich am besten eignet, ist dabei vom jeweiligen Anwendungsfall abhängig. Daher empfiehlt es sich für eine Studie mehrere Experimente mit unterschiedlichen Versuchsplänen durchzuführen (s. Tabelle 4.11).

IA St.2/Nr.8 Verwendung alternativer Versuchspläne, die für komplexe Modelle und/oder vielen Faktoren geeignet sind.

Die folgende Durchführung der Experimente (Ph. 10) wird über den *Modellausführungsservice* des DFS-Portals ausgeführt, nachdem die Experimentenplanung abgeschlossen ist. Es werden keine ablaufbedingten Anforderungen an die Modellausführung formuliert, weil dafür nur ein einmaliger Startaufruf vom Anwender erforderlich ist. Umso anspruchsvoller ist hingegen die technische Anforderung an das DFSS, weil die gesamte Simulation nach dem Startaufruf des Anwenders komplett automatisiert ausgeführt werden muss. Das Hauptprogramm setzt dafür die Parameter aus der Datenbank der Simulationsläufe und sendet den Anfrage für die Simulation an die HPC-Komponente.

IA St.2/Nr.9 Das DFSS muss die Durchführung einer DF-Studie komplett automatisiert ohne menschliche Eingriffe bewerkstelligen können.

Die letzte Phase des Vorgehensmodells ist die Ergebnisanalyse (Ph. 11), bei der der wesentliche Unterschied, im Vergleich zu klassischen Simulationen, in der Menge der Simulationsergebnisse liegt (s. Tabelle 4.11). Wie auch in Abschnitt 2.1.2 beschrieben wurde, sollten Datenmengen dieser Art in einem ersten Schritt verknüpft, gefiltert und verdichtet werden, um eine grundsätzliche Übersicht zu schaffen. Diese Aktionen sollten innerhalb der Datenbank über Abfragesprachen wie SQL ausgeführt werden.
Darauf aufbauend sollten auf Basis der Abfragen statistische Werkzeuge angewendet werden, um einerseits die Zielgrößen verschiedener Simulationsläufe miteinander zu vergleichen und darüber hinaus die zeitlichen Verläufe der Faktoren und Zielgrößen innerhalb der Läufe darzustellen. Im Bezug auf die hohen Faktoranzahlen der Modelle sollten auch die von der MSG beschriebenen *Fitness Landscape*-Diagramme und Streudiagramm-Matrizen zum Einsatz kommen. Das methodische Vorgehen und die Art der Analysewerkzeuge entsprechen im Wesentlichen aber den Techniken bereits bekannter klassischer Simulationsstudien. Aus diesem Grund ist die Ableitung weiterer Integrationsansätze nicht erforderlich. Alle Anforderungen an die Phasen des Vorgehensmodells sind mit den entsprechenden Integrationsansätzen der zweiten Stufe im Anhang in den Abbildungen A.3 bis A.5 angehängt.

Tab. 4.11: Anforderungen an die Modellanwendung

Phasen		Anforderungen für DFS in einer SC		Anforderungen erfüllt?
		Ablaufbedingte Anforderungen	Technische Anforderungen	
Modellanwendung	10. Experimenten-planung	• Maximierung der Ergebnislandschaft • Verwendung mehrerer unterschiedlicher Versuchspläne	• Effizienz (hohes Informationsgewinn/Zeit - Verhältnis)	☒
	11. Durchführung der Experimente	-	• Automatisierte Ausführung / Speicherung der Ergebnisse	☒
	12. Ergebnisanalyse	• Bewältigung großer Datenmengen	• Leistungsfähige Analysesoftware • Übersichtliche Statistiken über viele Faktoren und Zielgrößen	☑

4.3 Das Vorgehensmodell zur Integration von Data Farming Services

In Abschnitt 4.1 und Abschnitt 4.2 konnten acht Integrationsansätze der Stufe I und neun Integrationsansätze der Stufe II abgeleitet werden. Aus den Informationen der insgesamt 17 Integrationsansätze ergeben sich klare Vorgaben an die erforderlichen Phasen und den Inhalt jeder Phase. Zusätzlich wurden in Abschnitt 4.1 erforderliche phasenübergreifende Tätigkeiten wie V&V-Tätigkeiten, Dokumentation und Kollaboration in den Kontext einer SC eingeordnet. Aus den gewonnenen Informationen wurde in Abbildung 4.3 ein Vorgehensmodell entwickelt, um DFS in einer SC zu integrieren. Die Planung, Entwicklung und Einführung von DFS ist insgesamt ein großes Projekt, das hohe Qualifikationen auf mehreren unterschiedlichen Themengebieten erfordert. Das gesamte System ist aufgrund der hohen Modularität darauf ausgerichtet nach der einmaligen Implementierung viele SCs durch das Szenario beschreiben zu können, wodurch der Erstaufwand höher ist als die Übertragung des Systems auf eine andere SC. In der Beispielarchitektur aus Abbildung 4.2 ist der *Szenario* und *Experiment Service* zwar direkt mit den Datenbanken der SC verbunden, diese Schnittstelle ist allerdings nur möglich, wenn die IT-Systemlandschaft der SC auf einer ausreichend hohen Ebene integriert ist. Aus diesem Grund ist die direkte Anbindung der SC-Datenbanken optional, wodurch auch die manuelle Eingabe von Stammdaten im *Szenario Service* möglich sein muss. Aufgrund dieser Architektur kann das DFSS autark verwendet werden und ist daher einfacher auf eine andere SC zu übertragen.
Das entwickelte Vorgehensmodell erfordert als Basis keine bereits vorhandenen Simulationsmodelle und beschreibt daher auch alle Tätigkeiten, wenn zuvor noch keine klassischen Simulationstechniken in der betrachteten SC verwendet wurden. Aufgrund der Tatsache, dass Simulationstechniken im Vergleich zu DF aber kein neues Themengebiet in der Industrie sind, wird es bei einer Integration Fälle geben, in denen bereits vorhandene Simulationsmodelle um die DF-Fähigkeit erweitert werden sollen. In diesen Fällen kann das bereits vorhandene Modell als Basis des Projektes verwendet werden, sodass in der Aufstellung des Modellkonzeptes (Ph. 3)), der Modellentwicklung (Ph. 6) und der Model-

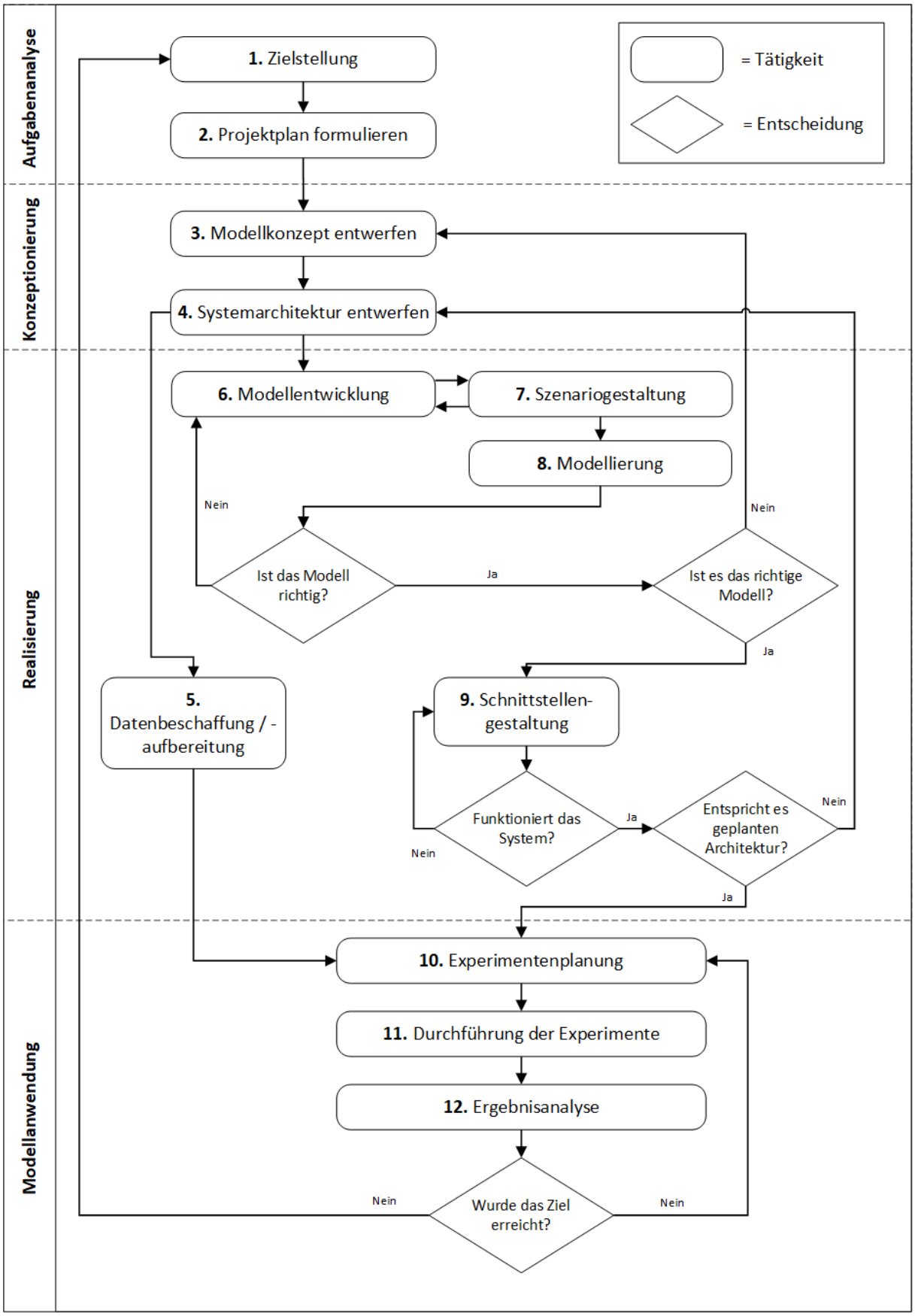

Abb. 4.3: Vorgehensmodell zur Integration von Data Farming Services in einer Supply Chain

lierung (Ph. 8) viele Aufwände entfallen. Anstatt dessen sollten alle Phasen ausführlich bzgl. der aufgestellten Anforderungen validiert werden. So gilt es das Modellkonzept auf einen ausreichend hohen Abstraktionsgrad und die technische Umsetzung des Modells auf die HPC-Fähigkeit und die erforderliche Performance hin zu überprüfen. Falls die Anforderungen an das bestehende Modell nicht erfüllt werden, sind Anpassungen erforderlich. In ungünstigen Fällen ist eine Übersetzung des Modells in eine andere Umgebung oder eine andere Programmiersprache durchzuführen.

5 Einsatz von Data Farming in der Problemdomäne einer Supply Chain

Etablierte Vorgehensmodelle basieren, zusätzlich zur theoretischen Abwicklung eines Vorhabens, auf viel Erfahrung, die durch den praktischen Einsatz der Vorgehensmodelle gewonnen wird (vgl. Abschnitt 2.1.3). Aus diesem Grund wurde das abgeleitete Vorgehensmodell aus Abschnitt 4.3 im folgenden Kapitel als Vorlage verwendet, um DF in der Problemdomäne einer SC praktisch anzuwenden. Die Intention des Vorhabens bestand zum einen in einer grundsätzlichen Machbarkeitsstudie und zum anderen in der Validierung der abgeleiteten Integrationsansätze, soweit es im Rahmen dieser Arbeit möglich war.
In der Machbarkeitsstudie war das Hauptziel die Überprüfung, ob der Einsatz von DF in der Problemdomäne von SCs möglich ist und ob das bestehende Potential des DFs erfolgsorientiert genutzt werden kann. Die Validierung der Integrationsansätze bezieht sich primär auf die technische Realisierung und Anwendung eines DFSS.

In Abschnitt 5.1 wird dafür zunächst die Entwicklung des DFSS analog zu den Phasen der *Konzeptionierung* und *Realisierung* des Vorgehensmodells beschrieben. Im Anschluss wird in Abschnitt 5.2 ein fiktives Szenario einer SC vorgestellt, dessen Zielgrößen durch eine Simulationsstudie optimiert wurden. In der Durchführung der Experimente wurden dabei sowohl allgemein etablierte als auch alternative Designs verwendet, die von der MSG für DF-Studien empfohlen werden. Darauf folgt in Abschnitt 5.3 die Auswertung der Simulationsergebnisse und ein Vergleich der verwendeten Versuchspläne in Hinblick auf die Ausprägungen der Zielgrößen und die erlangten Erkenntnisse über das fiktive Szenario. Im letzten Abschnitt 5.4 folgt ein Fazit, indem der gesamte praktische Einsatz von DF in der Problemdomäne von SCs bewertet wird. Dabei gilt es sowohl die Anwendbarkeit von DF als auch des abgeleiteten Vorgehensmodells zu beurteilen und ggf. widersprüchliche Erkenntnisse zu den theoretischen Ansätzen zu formulieren.

5.1 Entwicklung eines Data Farming Service Systems

Wie in Abschnitt 4.2 bereits erläutert wurde, besteht ein DFSS aus vielen Komponenten, dessen interoperabler Betrieb ein Projektteam mit ausgeprägten Qualifikationen erfordert. Zur Reduktion der Komplexität wurde aus diesem Grund auf die Einbindung einer HPC-Komponente und eines webbasierten Frontends verzichtet. Die HPC-Komponente ist für das DF zwar ein essentieller Bestandteil für die Geschwindigkeit der Simulationsdurchführung, besitzt für die prinzipielle Validierung der Methodik aber keine Relevanz. Webbasierte Frontends bieten hingegen Vorteile für die leichte Zugänglichkeit eines großen Anwenderkreises, sind für den Zweck einer Validierung aber ebenfalls nicht erforderlich.
Die Entwicklung des DFSS erfolgte nach den Phasen des abgeleiteten Vorgehensmodells aus Abbildung 4.3, indem das Funktionsverhalten, die Eingabeparameter und Zielgrößen

in Abschnitt 5.1.1 zunächst theoretisch geplant und darauffolgend in Abschnitt 5.1.2 in die Realität umgesetzt wurde.

5.1.1 Konzeptionierung

Für die Ausarbeitung des Modellkonzeptes bestand die Schwierigkeit darin, ein Systemverhalten zu bestimmten, mit dem sowohl inhärente Problemfelder in SCs untersucht werden können, als auch der Einsatz von DF getestet werden kann. Aus der Schnittmenge beider Kriterien konnte die Anforderung abgeleitet werden, dass das Modell in der Lage sein muss viele unterschiedliche Eingangsparameter zu simulieren. Das Verhalten von SCs ist in den meisten Fällen nur sehr schwierig zu prognostizieren, weil die Reaktion auf eingehende Leistungsflüsse je nach Akteur unterscheidet. Diese tendenziell nachteilige Eigenschaft für die Prognosefähigkeit wurde in das Modellkonzept aufgenommen, indem verschiedene Akteure unterschiedlich parametrisiert werden können. Im Umkehrschluss steigt die Anzahl der Eingabeparameter demnach mit der Anzahl der SC-Akteure des Szenarios.
Grundlegend werden in dem Modell die Material- und Informationsflüsse zwischen allen Akteuren einer SC über einen bestimmten Zeitraum simuliert. In Bezug auf die Einsatzgebiete für Simulationstechniken in SCs in Abschnitt 2.2, entspricht das Modellkonzept dem Simulationstyp einer SD-Simulation. Die Informationsflüsse bestehen dabei aus Bestellungen, die an die nächstgelegenen Akteure entlang der vertikalen Koordination veranlasst werden können. Für jede Bestellung wird als entgegengesetzter Materialfluss ein Transportauftrag generiert. Dabei kann seitens des Anwenders sowohl die Reaktionszeit auf eine Bestellung als auch die Lieferzeit parametrisiert werden.
Der Auslöser für die Leistungsflüsse innerhalb des Systems sind die Kunden einer SC, die in täglichen Abständen eine Bestellung durchführen. Die Stückzahl der Bestellung ist eine Zufallszahl, die normalverteilt um einen definierten Wert variiert. Aus diesem Grund erzeugt ein Simulationslauf mit gleichen Eingangsparametern keine identischen Ergebnisse, wodurch jeder Lauf mit einer angegeben Anzahl an Iterationen durchgeführt werden kann. In der Ergebnisanalyse können auf diese Weise, durch die gemittelte Auswertung aller Iterationen, „stabilere“ Zielgrößen der Simulationsläufe erzeugt werden. Auf die Formulierung der Zielgrößen wird im weiterem Verlauf noch eingegangen. In Bezug auf die Klassifikationskriterien für Simulationsmodelle (vgl. Abschnitt 2.1.1) handelt es sich demnach um eine diskrete zeitgesteuerte Simulation mit einer stochastischen Abbildung von Zufällen.
Um die Anforderungen an einen ausreichend hohen Abstraktionsgrad zu erfüllen, wurden mehrere Aspekte berücksichtigt. Zum einen wurde das minimale Zeitinkrement Δt, in dessen Regelmäßigkeit die Zustände des Modells abgefragt werden, in Tagen festgelegt. Bei einer Betrachtungsdauer von einem Monat werden die Zustände jedes Akteurs folglich auch nur ca. 30 Mal abgefragt. Weiterhin wurde die Annahme getroffen, dass zum Zeitpunkt der Auslösung eines Transportauftrags auch immer eine Transporteinheit zu Verfügung steht. Die Lieferzeit eines Auftrags ist dabei nicht von der geografischen Position des Akteurs abhängig.

Bei der Entwicklung des DFSSs wurde ebenfalls die erforderlichen Modularität und die stringente Trennung zwischen dem Modell und dem Szenario berücksichtigt, indem die prozessorientierte Darstellung einer SC, mit allen Akteuren und Leistungsflüssen, über eine XML-Datei importiert werden kann (vgl. Abbildung 2.1). In dem Szenario kann ein

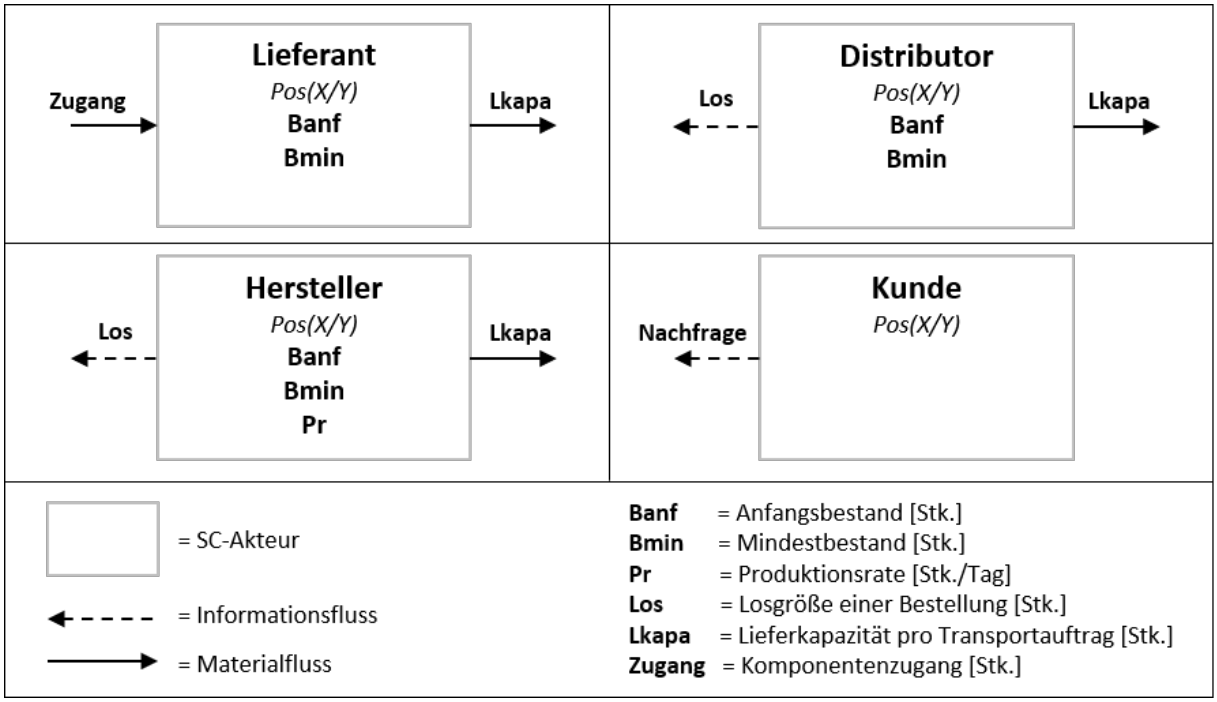

Abb. 5.1: Rollen der Supply Chain Akteure im Simulationsmodell

Akteur mit einer der vier möglichen *Rollen* (Kunde, Distributor, Hersteller, Lieferant) aus Abbildung 5.1 parametrisiert werden. Jeder Akteur besitzt dafür zunächst eine festgelegte x- und y-Koordinate, über die die Distanzen zwischen den Akteuren berechnet werden können. Während der Mittelwert für die Nachfrage der Kunden innerhalb eines Experimentes konstant ist, können die Parameter der Lieferanten, Hersteller und Distributoren in unterschiedlichen Läufen variiert werden. Die Parametrisierung des Anfangsbestandes (**Banf**) zum Simulationsstart, des zulässigen Mindestbestandes (**Bmin**) für eine Bestellungsauslösung und der maximalen Lieferkapazität (**Lkapa**) ist dabei mit Ausnahme des Kunden bei allen Akteuren vorhanden. Falls die Losgröße (**Los**) der Bestellung eines Herstellers oder Distributors größer als die Lieferkapazität des vorgelagerten Akteurs ist, werden mehrere Transportaufträge zu der Bestellung veranlasst. Weitere spezifische Funktionen und Parameter der Akteure werden im Folgenden erläutert:

Der Lieferant ist der letzte Akteur am Ende der Lieferkette und kann aus diesem Grund keine Kante zu einem weiteren „Vorgänger“ besitzen. Ein Lieferant lagert bzw. liefert eine oder mehrere Komponenten, die diesem in der Szenario XML-Datei zugeordnet werden.

Wenn der zulässige Mindestbestand (**Banf**) einer Komponente des Lieferanten unterschritten wird, wird der Bestand dieser Komponente um den festgelegten Komponentenzugang (**Zugang**) erhöht. Anders wie bei dem Hersteller oder Distributor ist dafür keine Bestellung erforderlich, sodass es in einer Simulation auch keine Latenz bis zum Auffüllen des Bestandes gibt.

Der Hersteller besitzt als einziger Akteur die Fähigkeit eingehende Komponenten in eine oder mehrere Produkte umzuwandeln. Die Produkte, die ein Hersteller produzieren kann, werden diesem in der Szenario XML-Datei zugewiesen. Weiterhin muss in dem Szenario definiert werden, welche Komponenten und Stückzahlen für die Herstellung eines Produktes benötigt werden. Bei ausreichendem Bestand kann ein Hersteller die festgelegte tägliche Produktionsrate (**Pr**) umsetzen. Wenn dies nicht der Fall ist, werden nur so viele Produkte hergestellt, bis der vorrätige Bestand ausgeschöpft ist. Wenn der Mindestbestand einer Komponente des Herstellers unterschritten wird, wird eine Bestellung an alle vorgelagerten Akteure ausgelöst, denen diese Komponenten zugewiesen ist, falls zum Zeitpunkt der Abfrage des Zustandes noch keine offene Bestellung mit der gleichen Komponente vorhanden ist. Der Hintergrund dieser Bedingung besteht darin, dass bei einer Lieferzeit/Reaktionszeit von mehreren Tagen nicht täglich neue Bestellungen erstellt werden, obwohl sich die Komponenten/Produkte der ersten Bestellung bereits in der Auslieferung befindet.

Der Distributor überwacht für alle zugewiesenen Produkte und/oder Komponenten die vorrätigen Bestände. Falls der Mindestbestand (**Bmin**) einer Komponente/eines Produktes unterschritten wird, wird eine Bestellung mit der Losgröße (**Los**) an alle vorgelagerten Akteure ausgelöst, denen diese Komponente zugewiesen ist. Das Bestellverhalten eines Distributors ist entspricht dem eines Herstellers.

Der Kunde besitzt, anders als die anderen Akteure, keine internen Bestände. Unabhängig davon, ob zum Zeitpunkt der Zustandsabfrage noch offene Bestellungen des Kunden vorliegen, löst ein Kunde für jeden simulierten Tag eine neue Bestellung aus.

Die Funktionsweise des Modells zielt darauf ab, die Informations- und Materialflüsse einer SC auf einer hohen Abstraktionsebene optimal auf die normalverteilten Kundenachfragen anzupassen. Die Zielgrößen der Simulation sind dafür einerseits die Kundezufriedenheit und andererseits die entstehenden SC-Kosten.
Aufgrund der kundenorientierten Ausrichtung moderner SCs besitzt die erste Zielgröße primäre Bedeutung (vgl. Kapitel 2). In Anbetracht an den Zweck der Simulationsstudie wird die Kundezufriedenheit vereinfacht an dem Verhältnis der gelieferten und bestellten Produkte innerhalb des gesamten Betrachtungszeitraums gemessen (s. Formel (5.1)). Eine hohe Reaktions- bzw. Lieferzeit auf eine Kundenbestellung wirkt sich demnach nicht

negativ auf die Kundenzufriedenheit aus, solange der volle Umfang der Bestellung bis zum Simulationsende an den Kunden ausgeliefert wurde.

$$\text{Kundenzufriedenheit} = \frac{\text{Stk. ausgeliefert}}{\text{Stk. bestellt}} \tag{5.1}$$

Die SC-Gesamtkosten setzen sich aus den Gesamtlagerkosten und den Gesamttransportkosten zusammen (s. Formeln (5.2) bis (5.4)). Die erforderlichen *Lagerkosten [Stk./Tag]* und *Kosten pro Kilometer* werden dabei vom Anwender des DFSS während der Experimentenplanung festgelegt.

$$\text{SC-Gesamtkosten} = \frac{\text{Gesamtlagerkosten}}{\text{Gesamttransportkosten}} \tag{5.2}$$

$$\text{Gesamtlagerkosten} = (\sum_{n=1}^{t} \sum_{n=1}^{akt} \text{Bestände}) * \text{Lagerkosten[Stk./Tag]} \tag{5.3}$$

t = Anzahl der Tage, akt = Anzahl der Akteure

$$\text{Gesamttransportkosten} = \sum_{n=1}^{tr} (\text{Distanz} * \text{Kosten pro Kilometer}) \tag{5.4}$$

tr = Anzahl der Transporte

Damit die genannten Zielgrößen im Rahmen der Ergebnisanalyse erhoben werden können, muss das Simulationsmodell die Daten der Bestellungen, der Transportaufträge und die täglichen Bestände von jedem Akteur im Backend des DFSS speichern. Auf die detaillierte Umsetzung des Modells und den Aufbau des Backends wird im folgenden Kapitel Abschnitt 5.1.2 eingegangen.

5.1.2 Realisierung

Das Frontend des DFSSs wurde in Microsoft Access (MS Access) 2016 in der Skriptsprache Visual Basic for Applications (VBA) implementiert, welches als Backend mit einem SQL-Server kommuniziert. Auf die Verwendung eines lokalen Backends in der MS Access Umgebung wurde bewusst verzichtet, weil der vorhandene SQL-Abfrageeditor für komplexe Abfragen nur eingeschränkte Funktionalitäten bereitstellt. Der erste Schritt bestand darin, die Kommunikation mit dem SQL-Server zu gewährleisten. Für diesen Zweck wurde ein Modul „*Backendkommunikation*“ erstellt, das wesentliche VBA-Funktionen zur Ausführung der SQL-Transaktionen SELECT, INSERT, UPDATE und DELETE beinhaltet. Die VBA-Funktionen dienen dabei als „Vermittler“ zwischen dem Anwender und dem SQL-Server und generieren einen ausführbaren SQL-Code, der über ein ADOdb-Objekt (Active Data Objects DataBase) an den SQL-Server gesendet wird.
Das zentrale Startformular für den Anwender ist in Abbildung B.1 dargestellt und grund-

sätzlich in drei Phasen gegliedert, die sequentiell zu bearbeiten sind. In der ersten Phase wird das erstellte Szenario im Backend gespeichert. Die zweite Phase beinhaltet das Anlegen von Experimenten für angelegte Szenarien zu denen in der dritten Phase Läufe für das Experiment parametrisiert werden können.

1. Szenario auswählen Mit dem Button „Szenario hinzufügen“ öffnet sich ein Auswahlfenster, in dem der Name des Szenarios definiert werden kann und der Pfad der XML-Datei festgelegt wird. In Quellcode 5.1 ist die erforderliche Struktur des Validierungsszenarios dargestellt, auf die in Abschnitt 5.2 noch weiter eingegangen wird.

Die XML-Datei ist in zwei Bereiche gegliedert. Von Zeile 3-28 werden die Eigenschaften und Beziehungen der SC-Akteure definiert, während von Zeile 29-38 Komponenten, Produkte und Stücklisten festgelegt werden.
Jeder Akteur besitzt die fünf Attribute *id, name, idRolle, lngPosX* und *lngPosY*, die für einen erfolgreichen Import unerlässlich sind. Die *id* muss innerhalb der XML-Datei für jeden Akteur eindeutig sein, weil die Beziehungen zu anderen Akteuren und zu den Komponenten über die *id* referenziert wird. Der *name* ist je nach Szenario frei wählbar. Die *idRolle* muss ein ganzzahliger Wert von 1-4 sein und definiert die Rolle des Akteurs in Anlehnung an Abbildung 5.1 (1=Kunde, 2=Distributor, 3=Hersteller, 4=Lieferant). Über die letzten beiden Attribute *lngPosX* und *lngPosY* wird die geografische Position des Akteurs in dem fiktiven Szenario gesetzt. Zusätzlich zu den Attributen kann ein Akteur maximal zwei weitere Unterelemente besitzen. In dem Element *komponenten* wird einem Akteur über die *id* ein *komponentenname* zugeordnet. Den Herstellern dürfen dabei nur die *komponentennamen* der Produkte zugewiesen werden, die sie selber herstellen und nicht die *komponentennamen*, die für die Produktion benötigt werden. Das zweite Unterelement *verbindungen* definiert die Kanten zu den anderen Akteuren. Mit dem Attribut *idnach* wird eine Kante zum nächsten Akteur entlang der Materialflussrichtung gesetzt. Aus diesem Grund besitzen Akteure der *idRolle=1* (Kunde) kein Unterelement *verbindungen*, weil der Materialfluss beim Kunden endet.
Im zweiten Bereich der XML-Datei werden die Komponenten definiert, zu denen im ersten Bereich bereits Beziehungen erstellt wurden. Falls für die Produktion einer *komponente* weitere *komponenten* erforderlich sind, können diese durch das Unterelement *stueckliste* angegeben werden. Eine *stueckliste* enthält die drei Attribute *komponente, komponenteB* und *anzahl*. Das erste Attribut gibt den *komponentennamen* des übergeordneten Elementes an, während die zweiten und dritten Attribute den Namen und die Anzahl der benötigten Komponente beschreiben. Entgegen der sinngemäßen Annahme über die unterschiedliche Bedeutungen der Begriffe „Komponente“ und „Produkt“, besteht im Simulationsmodell hingegen keine Differenzierung. Aus diesem Grund ist ein „Produkt“ in der Simulation ebenfalls eine Komponente, die im Vergleich zu anderen Komponenten lediglich ein oder mehrere *stuecklisten* als Unterelemente besitzen.

Der Import des Szenarios erfolgt in zwei Schritten. In dem ersten Schritt werden die SQL-Abfragen der Quellcodes D.1 bis D.5 verwendet, um die Daten der verschiedenen Ebenen

```
<?xml version="1.0" encoding="UTF-8"?>
<data>
  <akteure>
    <akteur id = "1" name = "Kunde" idRolle = "1"
    lngPosX = "135" lngPosY = "10">
      <komponenten id = "1" komponentenname = "Produkt" />
    </akteur>
    <akteur id = "2" name = "Distributor" idRolle = "2"
    lngPosX = "110" lngPosY = "15">
      <verbindungen id = "2" idnach = "1" />
      <komponenten id = "2" komponentenname = "Produkt" />
    </akteur>
    <akteur id = "3" name = "Hersteller" idRolle = "3"
    lngPosX = "40" lngPosY = "10">
      <verbindungen id = "3" idnach = "2" />
      <komponenten id = "3" komponentenname = "Produkt" />
    </akteur>
    <akteur id = "4" name = "Lieferant Komponente A"
    idRolle = "4" lngPosX = "10" lngPosY = "30">
      <verbindungen id = "4" idnach = "3" />
      <komponenten id = "4" komponentenname = "Komponente A" />
    </akteur>
    <akteur id = "5" name = "Lieferant Komponente B"
    idRolle = "4" lngPosX = "0" lngPosY = "0">
      <verbindungen id = "5" idnach = "3" />
      <komponenten id = "5" komponentenname = "Komponente B" />
    </akteur>
  </akteure>
  <komponenten>
    <komponente komponentenname = "Komponente A" />
    <komponente komponentenname = "Komponente B" />
    <komponente komponentenname = "Produkt">
      <stueckliste komponente = "Produkt"
      komponenteB = "Komponente A" anzahl = "1" />
      <stueckliste komponente = "Produkt"
      komponenteB = "Komponente B" anzahl = "1" />
    </komponente>
  </komponenten>
</data>
```

Quellcode 5.1: Szenario XML-Datei

der XML-Datei in die Tabellen des Datenbankdiagramms einzufügen. Die Tabellen aus Abbildung C.1 fungieren dafür als temporäres Transferverzeichnis (s. Tabellenbenennung: „tbl_tmp“-temporär), um die Daten der XML-Datei in eine relationale Tabellenform zu übertragen. Aus diesem Grund werden die gesamten Tabellen, vor der Ausführung der Abfragen der Quellcodes D.1 bis D.5, vollständig geleert, damit nur Daten des aktuell ausgewählten Szenarios in den Tabellen vorhanden sind. In einem zweiten Schritt werden die Daten der Transfertabellen in das endgültige Backend der Stammdatentabellen (Tabellenbenennung: „tbl_sda“-Stammdaten) übertragen (s. Abbildung C.2). Dabei werden

neue Primärschlüssel für die Akteure und Komponenten des Szenarios vergeben und entsprechende Fremdschlüssel der Verbindungstabellen gesetzt, damit jedes Element des Szenarios neben anderen Szenarien des DFSSs eindeutig identifiziert werden kann.

2. Experiment erstellen Zum Hinzufügen eines neuen Experimentes muss zuvor ein Szenario im DFSS vorliegen. Über den Button „Exp. hinzufügen“ öffnet sich das Formular aus Abbildung B.2, indem alle globalen Parameter gesetzt werden können. Neben der individuellen Auswahl eines Experimentnamens können die in Abschnitt 5.1.1 beschriebenen Eigenschaften, wie die Betrachtungsdauer, die Anzahl der Iterationen pro Simulationslauf, die Reaktions- und Lieferzeit, die Lager- und Lieferkosten und die Mittelwerte für die normalverteilten Nachfragen der Kunden, parametrisiert werden. Nach dem Speichern der Eintragungen wird ein neuer Eintrag in die Tabelle „tbl_eda_exp“ erstellt (s. Abbildung C.3). Die Mittelwerte für die Nachfrage der Kunden werden mit der erzeugten ID des Experimentes als Fremdschlüssel in die Tabelle „tbl_eda_exp_Kunden“ eingetragen.

3. Läufe hinzufügen Auf der Basis eines vorhandenen Experimentes können in der dritten Phase die Versuchsläufe erstellt werden. Die Auswahl des Versuchsplans erfolgt dabei nicht direkt innerhalb des DFSSs, sondern über eine implementierte Schnittstelle für den Import externer Versuchspläne. Über den Button „Simulationsläufe importieren“ wird dafür das Formular aus Abbildung B.3 geöffnet, indem für jeden Akteur der Pfad auf eine Textdatei festgelegt werden kann, welche die Parameter aller Versuchsläufe enthält. Bei der Auswahl der Textdateien sind die Hinweise und die Reihenfolge der erwarteten Faktoren des Formulars zu berücksichtigen (s. Abbildung B.3). Zusätzlich müssen alle ausgewählten Textdateien für die Akteure die gleiche Anzahl an Versuchsläufen besitzen, damit die fehlerfreie Ausführung der Simulation gewährleistet werden kann. Durch den Import der Versuchsläufe werden in die übrigen Tabellen der Versuchsplanung aus Abbildung C.4 alle notwendigen Daten eingetragen, die für den Start der Simulationsschleife erforderlich sind.

4. Simulation durchführen Mit dem Button „Simulation starten“ beginnt die automatisierte Ausführung der Simulation aller Versuchsläufe eines ausgewählten Experimentes. Zur Maximierung der Rechenleistung wurde ein VBA-Klassenmodul für die Akteure implementiert, in denen alle notwendigen Eigenschaften der Akteure gespeichert werden. Auf diese Weise können die erforderlichen Datenbankzugriffe während der Simulation auf ein notwendiges Minimum reduziert werden. Die wesentlichen Eigenschaften und Funktionen der VBA-Klasse **akteur** sind in Algorithmus 1 im Pseudocode dargestellt. Mit der Funktion *setzeParameter* (Zeile 8) werden mit der Angabe des aktuellen Versuchslaufes einmalig alle Parameter der Akteure aus der Datenbank ausgelesen und den Eigenschaften der VBA-Klasse (Zeile 1-8) zugewiesen. Für die Abfrage einer Bestellungsauslösung befindet sich bspw. die Information des Mindestbestandes eines Akteurs in der lokalen Laufzeit innerhalb der VBA-Klasse, sodass nicht jedes Mal über ein ADOdb-Objekt eine neue

Algorithmus 1: VBA-Klasse akteur (Pseudocode)

```
   /* Eigenschaften:                                                    */
 1 id;
 2 rolle;
 3 bestand;
 4 anfangsbestand;
 5 mindestbestand;
 6 losgroesse;
 7 lieferkapazität;
 8 komponentenzugang;
 9 nachfrage;
   /* Funktionen:                                                       */
10 setzeParameter(idLauf);
11 bestandZurücksetzen(idLauf);
12 offeneBestellungen();
13 bestand(intAnzahl);
```

Datenbankverbindung geöffnet werden muss. Die Funktionen in Zeile 11 und 13 sind für das Ändern der Bestände eines Akteurs erforderlich. Mit der Funktion *bestand(intAnzahl)* kann der Bestand einer angegebenen Komponente um den Wert *intAnzahl* erhöht bzw. verringert werden. Die Angabe der Komponenten bleibt dabei aus Übersichtsgründen im Pseudocode unberücksichtigt. Mit der Funktion *bestandZurücksetzen(idLauf)* wird die Eigenschaft *akteur.bestand* auf *akteur.anfangsbestand* gesetzt. Mit der letzten Funktion *offeneBestellungen()* können alle eingehenden unbearbeiteten Informationsflüsse eines Akteurs ausgelesen werden, um falls möglich entgegengesetzte Materialflüsse zu veranlassen.
Nach dem Setzen des Startzeitpunktes für das Experiment wird eine Schleife über alle Versuchsläufe initiiert (s. Algorithmus 2). In Zeile 3-5 werden für alle Klassen der Akteure die Eingangsparameter des aktuellen Laufs gesetzt. Zu Beginn jeder Iteration eines Laufs werden die Bestände eines Akteurs wieder auf den Anfangsbestand (**Banf**) zurückgesetzt. Der Hauptteil der Simulation ist zwischen Zeile 11-42 dargestellt, indem für jeden Tag des Betrachtungszeitraums alle Zustände der Akteure abgefragt werden.
Die Logik des Hauptteils kann prinzipiell in vier Bereiche gegliedert werden. Im ersten Bereich (Zeile 13-20) werden alle eingehende Materialflüsse der Akteure überprüft und ggf. die Bestände der VBA-Klasse aktualisiert. Der zweite Bereich (Zeile 21-25) beschreibt die interne Umsetzung von Beständen durch die Hersteller. Die Funktion *produzieren()* (Zeile 23) beinhaltet dafür ebenfalls die Aktualisierung der Bestände der VBA-Klasse *akteur*. Darauffolgend werden im dritten Bereich alle ausgehenden Materialflüsse in Form von Transportaufträgen veranlasst, falls offene Bestellungen vorliegen und ausreichend Bestände vorrätig sind (Zeile 26-32). Der letzte Bereich (Zeile 33-40) prüft und veranlasst alle ausgehenden Informationsflüsse. Kunden lösen jeden Tag der Simulation eine Bestellung aus (Zeile 33-35), während Hersteller und Distributoren Bestellungen nur durchführen, wenn die Mindestbestände (**Bmin**) der Komponenten/der Produkte unterschritten werden (Zeile 36-40).

Algorithmus 2: Die Simulationsschleife (Pseudocode)

```
1  setzeExperimentStart;
2  wiederhole
3      für alle akteure tue
4          akteur.setzeParameter;
5      Ende
6      für n = 1 bis anzahlIterationen tue
7          setzeIterationStart;
8          für jeden akteur tue
9              akteur.bestandZurücksetzen
10         Ende
11         für t = 1 bis anzahlTage tue
12             für jeden akteur tue
13                 wenn akteur.rolle = Hersteller oder Distributor dann
14                     prüfeEingänge();
15                 Ende
16                 wenn akteur.rolle = Lieferant dann
17                     wenn akteur.bestand ← akteur.mindestbestand dann
18                         akteur.bestand(akteur.komponentenzugang);
19                     Ende
20                 Ende
21                 wenn akteur.rolle = Hersteller dann
22                     wenn akteur.bestand ausreichend dann
23                         produzieren();
24                     Ende
25                 Ende
26                 wenn akteur.rolle != Kunde dann
27                     für jede bestellung bis akteur.offeneBestellungen tue
28                         wenn akteur.bestand → bestellung.stückzahl dann
29                             erstelleTransportauftrag();
30                         Ende
31                     Ende
32                 Ende
33                 wenn akteur.rolle = Kunde dann
34                     erstelleBestellung() mit akteur.nachfrage(normalverteilt);
35                 Ende
36                 wenn akteur.rolle = Distributor oder Hersteller dann
37                     wenn akteur.bestand ← akteur.mindestbestand dann
38                         erstelleBestellung() mit akteur.losgroesse;
39                     Ende
40                 Ende
41             Ende
42         Ende
43     Ende
44 bis bis alle Versuchsläufe simuliert wurden;
```

In den Funktionen *erstelleBestellung(), erstelleTransportauftrag()* werden die Bestellungen und Transporte in die Ereignistabellen der Simulation gespeichert (s. Abbildung C.4). Zusätzlich werden zum Ende jeder Iteration die Bestände aller Komponenten der Akteure ebenfalls in die Tabelle „tbl_exp_Bestände“ gespeichert. Nach dem Ende jeder Iteration, jedes Versuchslaufs und nach dem Abschluss des Experimentes wird jeweils der Endzeitpunkt im Backend gespeichert, um im Rahmen der Auswertungen die Geschwindigkeit des Simulationsmodells bewerten zu können.

5.2 Anwendung des Data Farming Service Systems

Mit der abgeschlossenen Realisierung und dem erfolgreichen Funktionstest des DFSSs wurde eine solide Grundlage entwickelt, mit der im folgenden Kapitel der Einsatz von DF im produktionslogistischen Kontext von SCs validiert wurde. Dazu wird in Abschnitt 5.2.1 ein fiktives Szenario vorgestellt, das in Abschnitt 5.2.2 für die Simulation verschiedener Versuchspläne verwendet wird.

5.2.1 Vorstellung des fiktiven Szenarios

Für den Einsatz von DF wurde als Szenario wissentlich eine triviale Architektur einer SC gewählt, weil der Fokus der Versuchsreihe nicht in der tiefgreifenden Analyse und Optimierung des fiktiven Szenarios, sondern in der grundlegenden Überprüfung der Machbarkeit und der Validierung der Methodik, liegt. Das konstruierte Szenario für die Versuchsreihe ist in Abbildung 5.2 dargestellt und umfasst insgesamt fünf Akteure. Innerhalb der SC

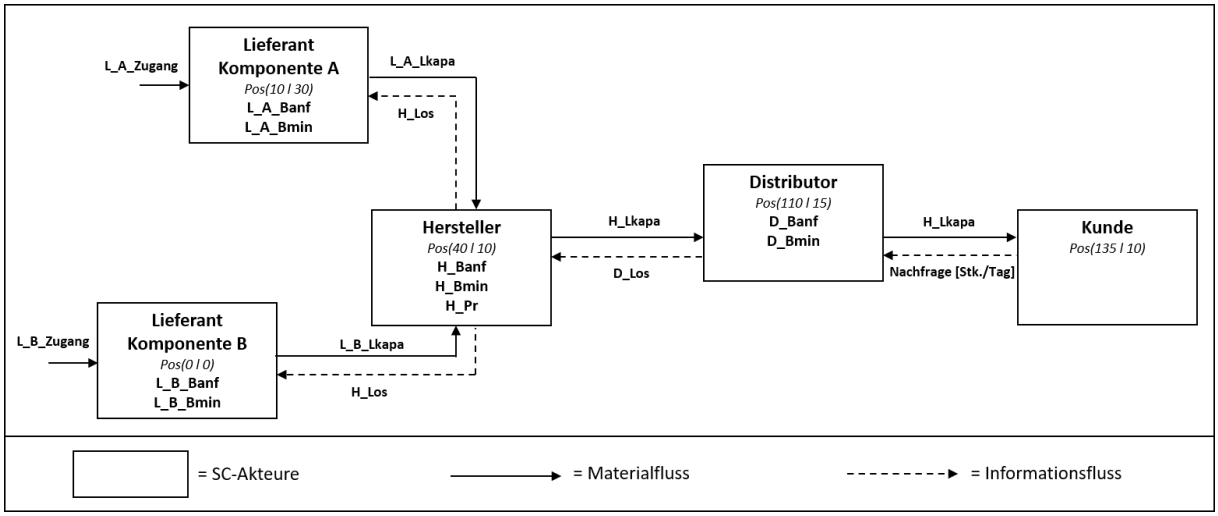

Abb. 5.2: Fiktives Szenario einer Supply Chain

löst ein *Kunde*, von ausschließlich einem *Produkt*, Bestellungen an einen *Distributor* aus. Dieser erhält das nachgefragte *Produkt* wiederum von einem *Hersteller*. Für die Fertigung eines *Produktes* ist jeweils eine *Komponente A* und eine *Komponente B* erforderlich, die dementsprechend entweder bei dem *Lieferant der Komponente A* bzw. dem *Lieferant der Komponente B* bestellt werden können.
Für jeden Akteur können in der Experimentenplanung die Werte für die Eingabeparameter zugewiesen werden, die bereits in der Modellkonzeption in Abbildung 5.1 dargestellt wurden. Selbst bei diesem Szenario mit einer verhältnismäßig geringen Anzahl an Akteuren ergeben sich für die Experimentenplanung 17 verschiedene Faktoren, die für die Simulation parametrisiert werden müssen.

5.2.2 Experimentenplanung und -durchführung

Ein wesentlicher Unterschied von DF-Modellen und klassischen Simulationsmodellen der Produktion und Logistik sind die verwendeten Versuchsdesigns der Experimentendurchführung. Im dritten DF-Bereich wurde daher die Verwendung alternativer Versuchspläne wie z. B. Latin Hypercube Designs, sequentieller Verzweigungen oder kombinierter Designs empfohlen, um das Verhältnis zwischen der Simulationszeit und dem Informationsgewinn zu steigern (vgl. Abschnitt 3.2.2). Auf Basis dieser Empfehlung wurde in Abschnitt 4.2 die Verwendung alternativer Versuchsläufe als Integrationsansatz formuliert (**IA St.2/Nr.8**), obwohl in der gesichteten Literatur kein direkter Vergleich zwischen etablierten teilfaktoriellen Versuchsplänen und den genannten alternativen Versuchsplänen an einem gleichen Modell aufgeführt wird.
Zur Validierung von **IA St.2/Nr.8** wurden daher zwei Experimente mit identischen globalen Modellparametern mit unterschiedlichen Versuchsplänen durchgeführt. Die gesamte Parametrisierung der Experimente, inklusive der unteren und oberen Grenzwerte der variierenden Parameter der Akteure, sind ist in Abbildung 5.3 dargestellt. Zur eindeutigen Zuordnung der Faktoren wurden vor den Parameterbezeichnungen der Akteure die Präfixe „D_“, „H_“, „L_A_“, „L_B_“ eingefügt. Die Betrachtungsdauer der Simulation betrug 30 Tage, die mit jeweils 10 Iterationen wiederholt wurden. Die Lieferzeit eines Transportauftrags betrug einen Tag, wohingegen die Reaktionszeit in Anbetracht der Untersuchungsfrage nicht berücksichtigt wurde (0 Tage). Die Auswahl der Lager- und Lieferkosten basierte auf plausiblen Vermutungen, bei der die Lieferkosten pro gefahrenen Kilometer mit 0,15 € eine stärkere Gewichtung als die täglichen Lagerkosten pro Stück (0,06 €) besaßen.

Versuchsdesigns Als Vertreter für etablierte Designs wurde ein zweistufiger Plackett-Burman (PB) Versuchsplan der Auflösung III verwendet, der in Minitab erstellt wurde [Min21]. PB Versuchspläne können in Minitab mit bis zu 47 Faktoren und maximal 48 Duchläufen erstellt werden, weshalb sie gut zum Screening geeignet sind. Der erstellte Versuchsplan für das Experiment besaß ebenfalls 48 Durchläufe und ist mit den Faktoren im Anhang in Abbildung E.1 abgebildet.

Experimentenplanung					
Betrachtungsdauer [Tage]:	30	Reaktionszeit [Tage]:	0	Lagerkosten [€]:	0,06
Anzahl der Iterationen:	10	Lieferzeit [Tage]:	1	Lieferkosten pro Kilometer [€]:	0,15
Mittlere Nachfrage des Kunden [Stk.]:	15				

Parameter der Akteure					
Distributor	Min.	Max.	**Hersteller**	Min.	Max.
D_Banf	0	100	H_Banf	0	100
D_Bmin	20	70	H_Bmin	20	70
D_Los	20	50	H_Los	20	50
D_Lkapa	20	50	H_Lkapa	20	50
			H_Pr	10	50
Lieferant Komponente A	Min.	Max.	**Lieferant Komponente B**	Min.	Max.
L_A_Banf	0	100	L_B_Banf	0	100
L_A_Bmin	20	70	L_B_Bmin	20	70
L_A_Lkapa	20	50	L_B_Lkapa	20	50
L_A_Zugang	10	50	L_B_Zugang	10	50

Abb. 5.3: Parametrisierung des Szenarios

Für den zum Vergleich konkurrierenden alternativen Versuchsplan wurde ein nahezu gleichverteilter Latin-Hypercube der *Naval Postgraduate School in Monterey* verwendet, der im Bereich Software Downloads im *Center for Data Farming* frei zum Download zur Verfügung steht [Nav21]. Das LHD besitzt mit 47 Versuchsläufen beinahe die identische Anzahl wie der PB Versuchsplan, wodurch die solide Voraussetzungen für einen direkten Vergleich gegeben waren.
Selbst mit nur zwei Stufen pro Faktor besitzt das vorgestellte Szenario mit 17 Faktoren $2^{17} = 131.072$ verschiedene Kombinationen für einen Versuchsplan. In dem PB Versuchsplan wurden mit 47 Läufen dementsprechend nur 0,036 % aller möglichen Ausgänge simuliert.

Performance Wider Erwarten wurden in der Ausführungsdauer der Experimente trotz der beinahe identischen Anzahl an Versuchsläufen erhebliche Unterschiede gemessen. Mit einer Iteration von 10 Wiederholungen wurden in dem Experiment mit dem LHD in 47 Versuchsläufen somit 470 Simulationen des Betrachtungszeitraums in 2086,8 Sekunden durchgeführt. Die durchschnittliche Simulation eines Betrachtungszeitraums dauerte im LHD demnach 4,4 Sekunden. Insgesamt wurden mit dem LHD 145.495 Datensätze generiert, was einer DF-Quote von ca. 70 generierten Datensätzen/Sek. entspricht.
Das Experiment des PB Versuchsplans besaß mit einer Gesamtlaufzeit von 3519 Sekunden für 480 Simulationen eine 68 %ige längere Dauer, obwohl nur ein Versuchslauf mehr

simuliert wurde. Die durchschnittliche Simulation eines Betrachtungszeitraums betrug 7,3 Sekunden und mit 156.739 generierten Datensätzen lag die DF-Quote bei ca. 45 Datensätze/Sek. .
Der Grund für die schlechtere Performance des PB Versuchsplans besteht darin, dass durch das zweistufige Faktor-Design nur die Extrempunkte der Faktoren simuliert werden, um in der anschließenden Analyse möglicherweise signifikante Einflüsse einzelner Faktoren zu erkennen. Durch die Simulation der Extrempunkte entsteht eine höhere Anzahl von Läufen mit „ungünstigen“ Eingabeparametern, bei denen z. B. maximale Anfangsbestände (**Banf**) und Losgrößen für Bestellungen (**Los**) zusammen mit minimalen Lieferkapazitäten (**Lkapa**) simuliert werden. Dadurch müssen pro Bestellung viele Transporte erstellt werden, von denen auf der Seite des empfangenen Akteurs erneut die Zustände abgefragt werden müssen. In Summe müssen viele Schleifen der Simulation öfter durchlaufen werden, wodurch die Performance für das Generieren der Daten abnimmt.
Die Simulation wurde mit einem *AMD Ryzen 7 3700X* 8 Kernprozessor durchgeführt, der während der gesamten Laufzeit der Experimente eine durchschnittliche CPU-Auslastung von lediglich 10 % besaß.

5.3 Ergebnisanalyse

Um die erzeugten Daten der Experimente auswerten zu können, wurden die Rohdaten der Ereignistabellen (s. Abbildung C.4) als Erstes mit der SQL-Abfrage der Quellcodes D.6 bis D.8 aufbereitet, um die Zielgrößen zu ermitteln. In Abbildung 5.4 sind die Streudiagramme der Zielgrößen für beide Experimente dargestellt. Auf der x-Achse sind die SC-Gesamtkosten pro ausgeliefertem Produkt an den Kunden und auf der y-Achse das Verhältnis der bestellten und ausgelieferten Produkte, angeordnet. Ersteres ist dabei der Indikator für die Kundenzufriedenheit (vgl. Formel (5.1)) und letzteres für die Höhe der Gesamtbetriebskosten (vgl. Formel (5.2)). Bei einem guten Versuchslauf würde sich das Verhältnis zwischen den bestellten und ausgelieferten Produkten bei gleichzeitig sehr geringen Gesamtkosten pro ausgeliefertem Produkt dem Faktor 1 nähern. Dies entspricht einer Anordnung in der oberen linken Ecke im dargestellten Streudiagramm in Abbildung 5.4.
Sowohl mit dem PB Versuchsplan als auch dem LHD waren dementsprechend viele gute Versuchsläufe vorhanden. Die Dichte und Anzahl der Läufe mit effektiven Zielgrößen ist bei dem LHD jedoch deutlich höher, wodurch mit diesem Design grundsätzlich bessere Parametereinstellungen für ein Systemverhalten gefunden werden konnten. Weiterhin entspricht die Verteilung der Datenpunkte im Streudiagramm beim LHD deutlich mehr den zugrundeliegenden mathematischen Gesetzmäßigkeiten: Mit der Reduktion der ausgelieferten Produkte werden auch die SC-Gesamtkosten auf weniger Produkte verteilt, wodurch die Kosten pro Stück exponentiell ansteigen. Während die Datenpunkte des LHDs annähernd exponentialverteilt sind, ähneln sich die des PB Versuchsplans vielmehr einer linearen Verteilung mit einer großen Streuweite.
Mit der genannten Priorisierung der Kundezufriedenheit (vgl. Abschnitt 5.1.1) hat der beste Lauf des PB Versuchsplans eine Kundenzufriedenheit von 97 % bei 2,28 € Gesamtkosten

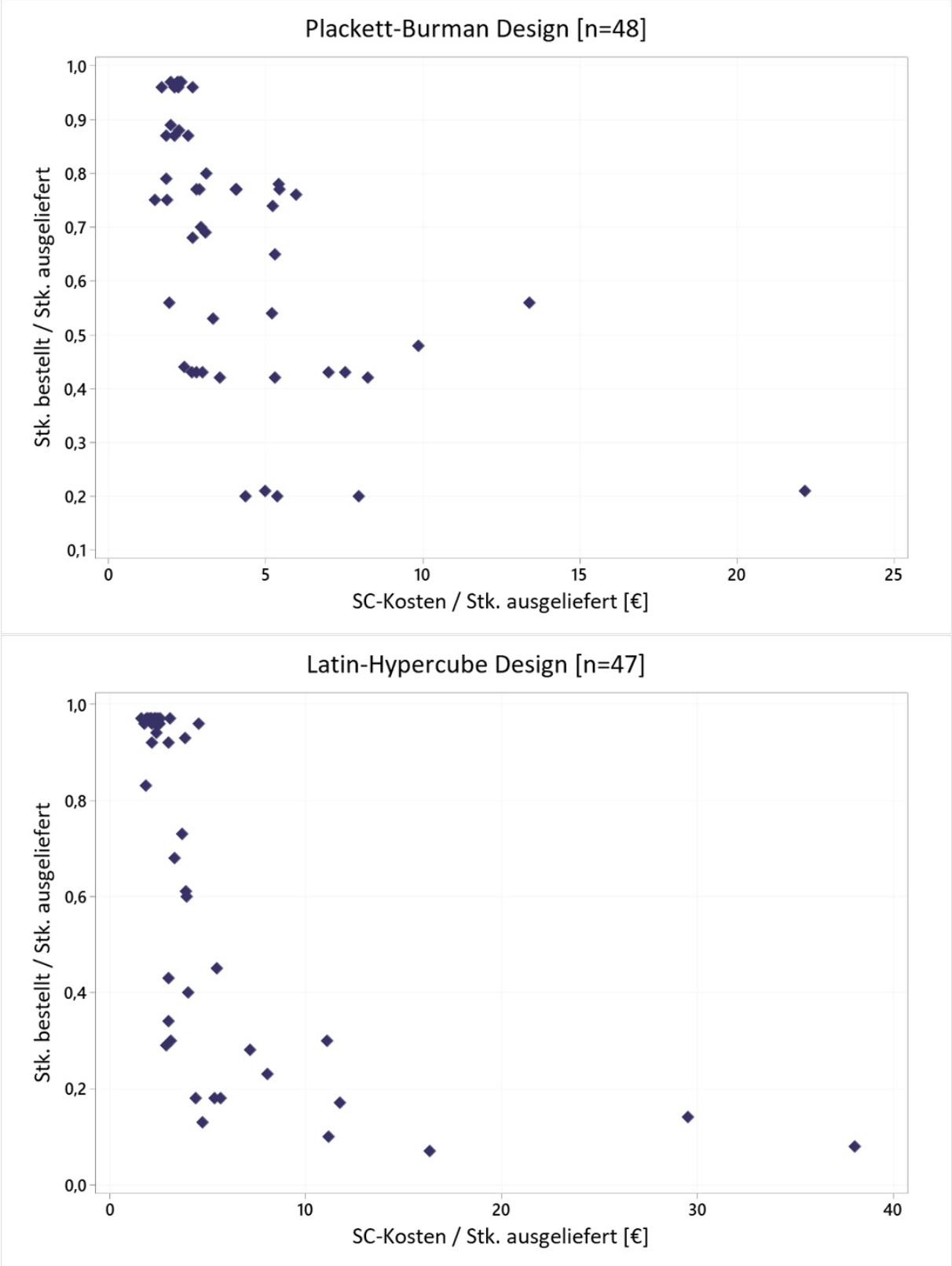

Abb. 5.4: Streudiagramme der Zielgrößen

pro ausgeliefertem Produkt. Mit dem LHD wurde die gleiche Kundenzufriedenheit hingegen mit deutlich geringeren Stückkosten von 1,64 € pro ausgeliefertem Produkt erreicht. Eine Kundenzufriedenheit von 100 % konnte mit den Parametern des Experimentes nicht nicht erzielt werden, weil der Kunde am letzten simulierten Tag immer eine Bestellung ausgelöst hat, die aufgrund der festgelegten Lieferzeit (1 Tag) nicht mehr zugestellt werden konnte. Eine 97 %ige Kundezufriedenheit entspricht daher in beiden Experimenten dem maximal erreichbaren Wert.

Zusätzlich zu dem Vergleich der Zielgrößen wurden auch die von der MSG empfohlenen 3D-Wirkungsflächendiagramme zur Visualisierung der erzeugten Ergebnislandschaften verwendet. Damit konnte überprüft werden, welches Design das Systemverhalten des trivialen Szenarios besser beschreibt. Da eine tiefgehende Analyse jeden Faktors nicht das Ziel der Analyse war, wurde in den 3D-Wirkungsflächendiagrammen die Summe aus den Faktoren der gleichen Gruppe verwendet. In den Abbildungen 5.5 bis 5.6 werden in der horizontalen Ebene der Diagramme aus diesem Grund die Losgrößen und Anfangsbestände bzw. die Losgrößen und Lieferkapazitäten aller Akteure summiert betrachtet. Auf diese Weise konnte der Einfluss einzelner Faktoren zwar nicht mehr gemessen werden, aber es wurde ein grobes Verständnis dafür gewonnen, welche Auswirkungen tendenziell hohe bzw. niedrige Losgrößen, Bestände und Lieferkapazitäten auf die Gesamtkosten haben.

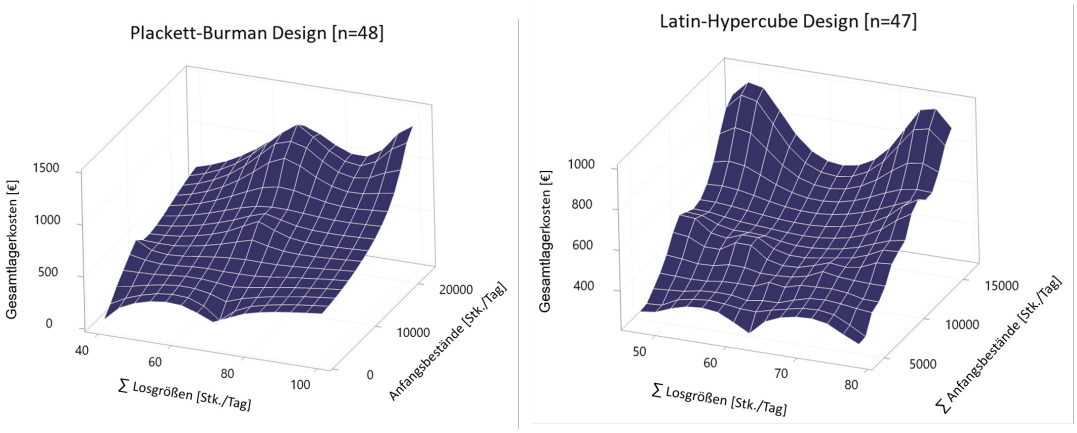

Abb. 5.5: 3D-Wirkungsflächendiagramme: Einfluss der Losgrößen und Bestände auf die Bestandskosten

In Abbildung 5.5 wurden die Einflüsse der Losgrößen und Anfangsbestände auf die Gesamtlagerkosten (vgl.Formel (5.3)) analysiert. Prinzipiell sollten aus insgesamt hohen Anfangsbeständen innerhalb des Systems auch erhöhte Lagerkosten messbar sein. Darüber hinaus befinden sich bei geringen Losgrößen tendenziell weniger Komponenten in in Arbeit befindlichen Transportaufträgen. Komponenten werden während eines Transports nämlich keinem Bestand eines Akteurs zugeordnet. Aus diesem Grund müssten die Bestandskosten bei geringen Losgrößen ansteigen, weil sich im Durchschnitt weniger Komponenten in aktuellen Lieferungen befinden.

In beiden Wirkungsflächen der Versuchspläne ist der Einfluss der Anfangsbestände auf die Gesamtlagerkosten zu erkennen. Der signifikante Unterschied zwischen beiden Ergebnislandschaften sind die Gesamtlagerkosten bei niedrigen Losgrößen und hohen Anfangsbeständen (s. Abbildung 5.5 - hintere linke Ecke). Während in den Ergebnissen des LHDs das beschriebene Systemverhalten eindeutig wiedererkannt werden kann, wurde im PB kein Ausschlag mit niedrigen Losgrößen und hohen Anfangsbeständen registriert. Die Landschaft des LHD besitzt aus diesem Grund mehr Strukturen als die des PB Versuchsplans, wodurch das Verhalten des Modells detaillierter beschrieben wird.

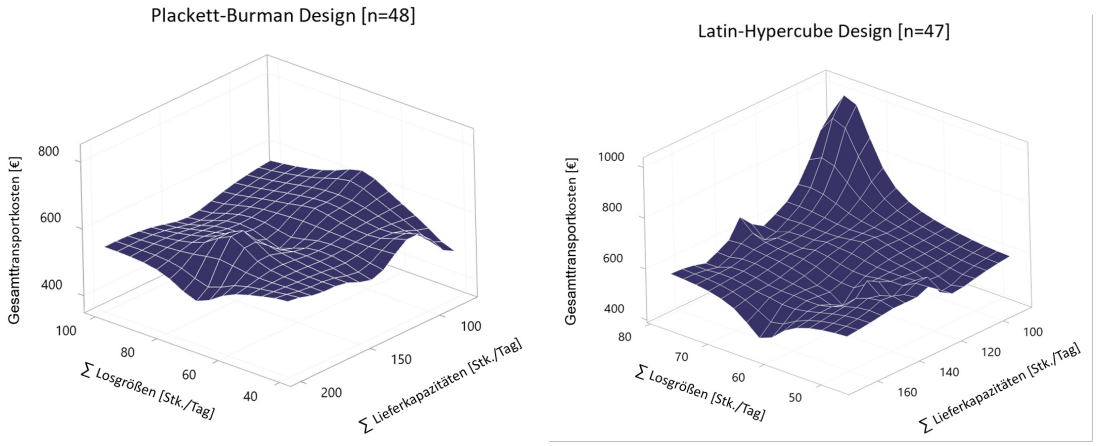

Abb. 5.6: 3D-Wirkungsflächendiagramme: Einfluss der Losgrößen und Lieferkapazitäten auf die Transportkosten

Noch deutlicher kann dieser Effekt in Abbildung 5.6 erkannt werden, in der die gesamten Transportkosten (vgl. Formel (5.4)) in Abhängigkeit von den parametrisierten Losgrößen und Lieferkapazitäten der Akteure wiedergegeben werden. Die höchsten Transportkosten entstehen logischerweise, wenn viele Transporte mit geringeren Kapazitäten veranlasst werden. Zur Einsparung von Transportkosten ist es daher empfehlenswert die Anzahl der Transporte zu reduzieren, indem die Kapazität pro Lieferung erhöht wird. Folglich steigen die Transportkosten in der Simulation bei hohen Losgrößen von Bestellungen und geringen Lieferkapazitäten, weil mehrere Transporte für die Bearbeitung einer Bestellung durchgeführt werden müssen.
Im LHD wird dieses Verhalten mit einem deutlichen Anstieg der Transportkosten bei hohen Losgrößen und geringen Lieferkapazitäten sehr anschaulich dargestellt. Die Landschaft des PB Versuchsplans weist hingegen keine „rauen“ Strukturen auf, aus denen Erkenntnisse gezogen werden könnten. Im direkten Vergleich sind Strukturen beider Wirkungsflächen stark inkongruent zueinander. Die einzige Gemeinsamkeit ist ein leichter Abfall der Transportkosten bei mittleren Losgrößen (ca. 60 Stk./Bestellung) und maximalen Lieferkapazitäten (ca. 160-200 Stk./Lieferung). Das erwartete Systemverhalten wird aber in beiden 3D-Wirkungsflächendiagrammen deutlich präziser durch das Experiment mit dem LHD beschrieben, wodurch bei diesem Design sowohl in Bezug auf die Qualität

der Ergebnisse als auch in der Menge der abzuleitenden Erkenntnisse bessere Daten in der Simulation generiert werden.

5.4 Fazit

Die zugrundeliegende Notwendigkeit für den Forschungsaspekt der vorliegenden Masterarbeit ist der stetige Anstieg der Komplexität und Dynamik von Materialflusssystemen. Da es eine immer größere Herausforderung ist moderne SCs durch analytische Methoden oder etablierte Simulationstechniken zu optimieren, sollte DF als Potentialität für die Lösung des Problems untersucht werden. Die Arbeit verfolgte dabei zwei hauptsächliche Ziele.
Das erste Ziel war es Ansätze für ein Vorgehensmodell zu formulieren, mit dem DFS in die Problemdomäne von SCs integriert werden kann. Zur Erfüllung des Ziels wurden insgesamt 17 Integrationsansätze an die Vorgehensweise für die Entwicklung von DF-Simulationsmodellen und die Durchführung von Simulationsstudien erarbeitet. Basierend auf den Integrationsansätzen wurde ein Vorgehensmodell mit definierten Phasen und Phaseninhalten zur Anwendung von DF im produktionslogitischen Kontext von SCs entwickelt. Die Phaseninhalte unterliegen dafür klaren Anforderungen und nehmen an vielen Stellen direkten Bezug zu den organisatorischen und technischen Gegebenheiten realer SCs. Darüber hinaus werden in den Integrationsansätzen auch notwendige phasenübergreifende Tätigkeiten wie die Verifikation & Validierung, Dokumentationserstellungen oder erforderliche Kollaboration berücksichtigt. Gerade unter dem Stichwort „Kollaboration“ wird im Bereich der Planung, Konzeptionierung und Datenbeschaffung besonders auf die erforderliche inter- als auch die intraorganisationale Koordination entlang der gesamten SC eingegangen.

Das zweite Ziel bestand darin den grundlegenden Einsatz von DF an einem fiktiven Szenario einer SC zu testen und die formulierten Integrationsansätzen, soweit es im Rahmen der Arbeit möglich war, zu validieren. Mit Hilfe des erarbeiteten Vorgehensmodells wurde ein voll funktionierendes DFSS mit allen grundlegenden Funktionen zur Planung und Durchführung von Simulationsstudien konstruiert. Nach **IA St.1/Nr.5** gewährleistet das DFSS die modulare Gestaltung des Szenarios, wodurch die Simulation und Analyse individueller SCs möglich ist. In den durchgeführten Experimenten konnte in einem direkten Vergleich zwischen etablierten und alternativen Versuchsdesigns das große Potential von LHDs bestätigt werden. Mit einer 40 %ig schnelleren Berechnungsdauer pro simulierten Betrachtungszeitraum konnten mit dem verwendeten LHD in allen Kategorien qualitativ hochwertigere Daten generiert werden. Dabei besitzt die entstandene Ergebnislandschaft der Faktoren und Zielgrößen detailliertere Strukturen, die das Verhalten des Modells wesentlich konkreter beschreiben, als die Versuche des PB Versuchsplans. Zusätzlich ist die Anzahl an Versuchsläufen mit positiven Ergebnissen mit dem LHD deutlich höher und im direkten Vergleich der besten Versuchsläufe liefert das LHD ebenfalls ein besseres Resultat.

Durch den praktischen Einsatz von DF und der erfolgreichen Analyse des fiktiven Szenarios konnten viele Integrationsansätze zur technischen Realisierung von DF-Studien validiert werden. Dies beinhaltet die Ansätze an die Verwendung einer hohen Abstraktionsebene bei der Modellierung (**IA St.2/Nr.1**), an die Modellimplementierung in Programmiersprachen (**IA St.2/Nr.4**) und an die notwendige automatisierte Ausführung der Experimente (**IA St.2/Nr.9**). Weiterhin wurde aufgezeigt, dass selbst für die Entwicklung der grundlegendsten Funktionen eines DFSSs viele unterschiedliche Qualifikationen erforderlich sind (**IA St.2/Nr.9**).
Andererseits gibt es aber auch Integrationsansätze, die zwar auf fundierten wissenschaftlichen Grundlagen basieren, im Rahmen der Arbeit allerdings nicht validiert werden konnten. Dementsprechend wurde für die Durchführung der Simulation lediglich eine einzelne CPU verwendet, anstatt die Berechnung durch die Aufteilung auf ein HPC-Cluster zu parallelisieren (**IA St.2/Nr.2**). In Anbetracht der geringen CPU-Auslastung während der durchgeführten Simulation, könnte die Simulationszeit auf diese Weise erheblich reduziert werden, wodurch DF nochmals geeigneter für komplexe SC-Szenarien wird. Darüber hinaus konnten die Integrationsansätze im Zusammenhang mit den Prozessen oder der vorliegenden IT-Landschaft realer SCs nicht überprüft werden. Gerade in Bezug auf **IA St.2/Nr.3** zur Förderung der kollaborativen Planung, könnte es bei unterschiedlichen Akteuren auch entsprechend verschiedene Definitionen über das zu erreichende Ziel geben. Die optimale Reduktion der Gesamtkosten eines Systems bedeutet nicht, dass auch die Kosten für jeden Akteur maximal reduziert werden. Dominierende Akteure der SC werden aus diesem Grund immer versuchen die Lieferkette und somit auch eine geplante DF-Studie zu ihren Gunsten auszurichten. Ob die geforderte Kollaboration des DFs in der Problemdomäne von SCs realisiert werden kann, muss daher in einer praktischen Erprobung an einer realen SC validiert werden.

Abschließend kann der Beschluss gefasst werden, dass der Einsatz von DF nach dem formulierten Vorgehensmodell der erarbeiteten Integrationsansätze möglich ist. Durch erste Experimente und Ergebnisanalysen konnten mit dem verwendeten LHD bereits Erkenntnisse eines fiktiven Szenarios mit 17 Eingabeparametern abgeleitet werden, die unter den gleichen Voraussetzungen bzgl. der Hardware und Simulationsdauer wahrscheinlich nicht durch klassische Simulationsmodelle und Vorgehensweisen entdeckt werden können. Aufgrund der maximalen Flexibilität im Aufbau von Latin Hypercubes können LHDs auch für SCs mit wesentlich mehr Eingangsparametern verwendet werden, weil es möglich ist, die Anzahl der Versuchsläufe individuell festzulegen. So ist es bspw. möglich Latin Hypercubes mit über 500 Faktoren in einem gesättigtem Design mit ungefähr der gleichen Anzahl an Läufen zu simulieren, das von der Komplexität schon eher den realen SCs entspricht. Auch wenn die Einführung und Entwicklung eines serviceorientierten DF-Systems hohe Qualifikationen erfordern und mit viel Aufwand verbunden sind, wurde durch die vorliegende Arbeit das hohe Potential von DF bewiesen, wodurch es als leistungsstarkes Simulationswerkzeug für SCs empfohlen wird.

6 Zusammenfassung und Ausblick

Die beiden Hauptziele der Masterarbeit bestanden darin Integrationsansätze für den Einsatz von DFS in SCs zu erarbeiten und durch den praktischen Einsatz an einem fiktiven Szenario zu validieren. Für die Erreichung dieser Ziele wurden in Kapitel 2 die Grundlagen aktuell verwendeter Simulationstechniken in SCs erarbeitet. Dafür wurden zu Beginn des Kapitels als Erstes die Begriffe SC und SCM definiert und Bezug zu den dynamischen Eigenschaften von SCs genommen, die in Form von verzögerten Informationsweitergaben zum Bullwhip-Effekt führen können. Im folgenden Abschnitt 2.1 wurde im Zusammenhang zur Simulationstechnik der Modell- und Systembegriff beschrieben und die unterschiedlichen Arten und Klassikationskriterien von Simulationsmodellen vorgestellt (Abschnitt 2.1.1). In Abschnitt 2.1.2 wurde darauffolgend der gesamte Ablauf einer Simulationsstudie von der Modellbildung über die Experimentenplanung und -durchführung bis hin zur systematischen Analyse der Ergebnisse beschrieben. Dabei war es wichtig die Bedeutung des Detaillierungs- bzw. Abstraktionsgrad der Modellbildung und etablierte voll- bzw. teilfaktorielle Versuchsdesigns der Experimentenplanung vorzustellen. Im nächsten Abschnitt 2.1.3 wurden die Vorgehensmodelle für Simulationsstudien von Law, Banks, Sargent und der ASIM vorgestellt und die gemeinsamen Phasen der Modelle selektiert, die aus den Elementen der Aufgabenanalyse, Modellformulierung, Modellimplementierung, Modellüberprüfung und Modellanwendung bestehen. In Abschnitt 2.1.4 wurden Kriterien an die Simulationswürdigkeit von Problemen aufgestellt und Vor- und Nachteile für die Verwendung von Simulationstechniken erarbeitet. Im letzten Abschnitt 2.2 wurden explizite Simulationstypen für SC-Simulationen vorgestellt und auf die Experimentenplanung eingegangen. Aufgrund der zunehmenden Komplexität von Materialflusssystemen entsprechen verwendete OFAT-Ansätze und etablierte Versuchspläne nicht den Anforderungen für zukunftsträchtige Optimierungen von SCs, wodurch die Notwendigkeit von DF formuliert wurde.

In Kapitel 3 wurde als Kontrast zu den Simulationstechniken der Produktion und Logistik die Literatur zu DF gesichtet und strukturiert zusammengefasst. In Abschnitt 3.1 wurden dafür zunächst der Ursprung des DFs in den agentenbasierten Modellen des USMC und die wichtigsten Projekte für die DF -Entwicklung im Rahmen der NATO vorgestellt. Während in Abschnitt 3.1 die chronologische Entwicklung im Vordergrund stand, wurde in Abschnitt 3.2 auf die inhaltliche Entwicklung der technischen und ablaufbedingten Strukturen einer DF-Studie eingegangen. Dabei wurde besonderer Bezug auf den MSG-088 Bericht genommen, indem die sechs DF-Bereiche mit sowohl technischen als auch verfahrensspezifischen Kriterien definiert wurden. Zusätzlich wurde das aktuelle Forschungsthema der NATO vorgestellt, um DF über ein webbasiertes DFS-Portal modular für einen großen Benutzerkreis als Service anzubieten. Da innerhalb der Literatur zum Forschungsgebiet unterschiedliche Definitionen des Begriffs „Data Farming“ existieren wurde in Abschnitt 3.3 nach der gesamten Aufarbeitung eine Definition gewählt, die ebenfalls Allgemeingültigkeit für den produktionslogistischen Kontext besitzt.

Im folgenden Kapitel 4 wurden zur Erfüllung des ersten Hauptziels Integrationsansätze für das Vorgehen erarbeitet, DFS in SCs zu integrieren. Die methodische Arbeitsweise bestand zunächst darin, die gemeinsamen Elemente klassischer Vorgehensmodelle (Abschnitt 2.1.3) den sechs DF-Bereichen (Abschnitt 3.2.2) gegenüberzustellen. In Abschnitt 4.1 wurden darauf basierend die Integrationsansätze der ersten Stufe verfasst, aus denen die erforderlichen Phasen eines Vorgehensmodells ausgehen. Als Vertreter für klassische Vorgehensweisen von Simulationsstudien wurden dabei die Vorgehensmodelle nach Law, Banks, Sargent und der ASIM verwendet (Abschnitt 2.1.3). Das entstandene „Gerüst“ des Vorgehensmodells wurde im folgenden Abschnitt 4.2 verwendet, um in Zuge der Integrationsansätze der zweiten Stufe technische und ablaufbedingte Anforderungen der Phaseninhalte aufzustellen. Das Ergebnis des vierten Kapitels ist das dargestellte Ablaufdiagramm des entstandenen Vorgehensmodells aus Abschnitt 4.3.

Das letzte Kapitel 5 beschreibt die praktische Validierung der aus Kapitel 4 erarbeiteten Integrationsansätze. Dafür wurde in Abschnitt 5.1 ein lauffähiges DFSS mit den grundlegenden Services zur Planung und Durchführung von DF-Simulationen entwickelt. Das Frontend des Systems wurde in MS Access implementiert, dass als Backend mit einem SQL-Server kommuniziert. Über eine zentrale Benutzerschnittstelle ist es möglich individuelle Szenarien von SCs in das DFSS zu importieren, Experimente zu planen und unterschiedliche Versuchsläufe bestimmter Designs zu importieren. Die Konzeptionierung (Abschnitt 5.1.1) und Realisierung (Abschnitt 5.1.2) des Systems richtete sich dabei nach den Phasen des Vorgehensmodells aus Abschnitt 4.3. Im Anschluss an die Entwicklung wurden in Abschnitt 5.2 zwei Experimente mit einem fiktiven Szenario einer SC durchgeführt, um empfohlene alternative Versuchspläne des dritten DF-Bereichs mit etablierten Versuchsplänen zu vergleichen. Als Ergebnis erwies sich ein LHD im Gegensatz zu einem zweistufigen PB Versuchsplan der Auflösung III als wesentlich geeigneter für die Absicht des DFs. Mit einer deutlich geringeren Simulationszeit wurden mit dem LHD qualitativ hochwertigere Daten generiert.
Aufgrund der praktischen Umsetzung des DFSSs und der positiven Ergebnisse des Experimentes mit dem LHD konnte ein Großteil der Integrationsansätze erfolgreich validiert und die grundlegende Einsatzmöglichkeit von DF innerhalb der Problemdomäne einer SC bewiesen werden. Weiterhin wurden durch die Masterarbeit wesentliche Potentiale für DF in SCs aufgezeigt.

Da die Entwicklung der Integrationsansätze und des Vorgehensmodells aus Kapitel 4 nur der erste theoretische Schritt einer umfassenden Integrationsphase ist, könnte in zukünftigen Forschungen auf dem Gebiet an vielen Stellen angeknüpft werden. Zunächst sollte dabei die technische Realisierung des DFSSs weiter verfolgt werden. Das Vorgehensmodell ist darauf ausgelegt DF als Service für die Akteure der SCs anzubieten. Dazu ist allerdings ein webbasiertes Frontend und die Weiterentwicklung des *Experiment* bzw. *Analyse Service* nach Abbildung 4.2 erforderlich. Zusätzlich sollten für den *Szenario Service* Editoren entwickelt werden, durch die der Anwender die zu untersuchende SC grafisch visualisiert erstellen kann. Auch wenn die Ergebnisse des Experimentes mit dem LHD zufriedenstellend waren, könnten die Potentiale weiterer alternativer Versuchsdesigns, wie z. B. sequentieller

Verzweigungen oder frequenzbasierter Designs untersucht werden.
Für eine absolut zuverlässige Erkenntnis über das Potential von DF in SCs sollte ein DFSS in der Umgebung einer realen SC implementiert werden. Dadurch kann zum einen der kritische **IA St.2/Nr.3** der erforderlichen interorganisationalen Kollaboration validiert werden und weiterhin überprüft werden, ob die Kostenersparnis durch die Optimierung der SC größer ist als bei dem Einsatz klassischer Simulationstechniken.

Literaturverzeichnis

[Add21] Additive: Minitab-Web-App, 2021. URL: https://www.additive-net.de/de/software/produkte/minitab/minitab/minitab-web-app, 02. 04. 2021

[Ama21] Amazon: Amazon High Performance Computing, 2021. URL: https://aws.amazon.com/de/hpc/, 01. 04. 2021

[AMD21] AMD: AMD Ryzen Threadripper 3990X Prozessor, 2021. URL: https://www.amd.com/de/products/cpu/amd-ryzen-threadripper-3990x, 19. 02. 2021

[And13] Anderson, M. A.: Agent-based modelling in thenew Zealand Defence Force. 2013, URL: https://www.researchgate.net/publication/259604091_Agent-Based_Modelling_in_the_New_Zealand_Defence_Force, 06. 02. 2021

[Aun15] Aunkofer, B.: Die 7+ Dimensionen der Logistik, 17.12.2015. URL: https://www.der-wirtschaftsingenieur.de/index.php/die-7-dimensionen-der-logistik/, 12. 03. 2021

[Ban10] Banks, J., Hrsg.: Discrete-event system simulation, 5. ed., internat. ed. Pearson, Upper Saddle River, NJ, 2010, ISBN: 0-13-815037-0

[Bec04] Beckmann, H., Hrsg.: Supply Chain Management: Grundlagen, Konzept und Strategien, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, ISBN: 978-3-642-62089-8. DOI: 10.1007/978-3-642-17057-7

[Ber14] Bernhard von Bonn: Big Data für eine optimierte Supply Chain – Wie verbessert sich die Planung und Gestalötung von Supply-Chain Netzwerken durch den Einsatz von Big Data Verfahren. Fraunhofer IML, Dortmund, 2014, URL: https://www.industrie40.iml.fraunhofer.de/content/dam/iml/industrie40/de/documents/Studien/140714_Positionspapier_BigData4.pdf, 18. 03. 2021

[BH98] Brandstein, A.; Horne, G. E.: Manueuver War Science 1998 – DataFarming: A Meta-technique for Research in the 21st Century. 1998, URL: https://www.researchgate.net/publication/258375147_Farming_Systems_Research_into_the_21st_Century_The_New_Dynamic, 28. 01. 2021

[BS09] Banks, C. M.; Sokolowski, J. A.: Principles of modeling and simulation – A multidisciplinary approach, Wiley, Hoboken, NJ, 2009, ISBN: 978-0-470-28943-3

[Che+05] Chen, T. u. a.: Cell Broadband Engine Architecture and its first implementation – A performance view, 2005. URL: https://www.ibm.com/developerworks/library/pa-cellperf/

[CM11] Campuzano Bolarín, F.; Mula, J.: Supply chain simulation – A system dynamics approach for improving performance, Springer, London, 2011, ISBN: 978-0-85729-719-8. URL: http://www.loc.gov/catdir/enhancements/fy1208/2011276655-b.html

[Deu] Deutsches Institut für Normung e.V., Hrsg.: Internationales Elektrotechnisches Wörterbuch: Teil 351: Leittechnik. DIN IEC. Beuth Verlag GmbH, Berlin. DOI: 10.31030/2159569

[Deu15] Deutsches Institut für Normung e.V., Hrsg.: Qualitätsmanagementsysteme - Grundlagen und Begriffe. DIN EN ISO. 2015,

[Ele12] Eley, M.: Simulation in der Logistik – Eine Einführung in die Erstellung ereignisdiskreter Modelle unter Verwendung des Werkzeuges "Plant Simulation", Springer-Lehrbuch. Springer Gabler, Berlin und Heidelberg, 2012, ISBN: 978-3-642-27373-5

[Gad21] Gadgetversus: AMD Ryzen Threadripper GFLOPS performance, 19.02.2021. URL: https://gadgetversus.com/processor/amd-ryzen-threadripper-gflops-performance/

[Ghe08] Ghesla, M.: Simulationssoftware in der Logistik - Eine Nutzenanalyse am Beispiel einer Automotive Supply Chain. Technischer Universität Wien, Wien, 2008, URL: https://repositum.tuwien.at/handle/20.500.12708/14438, 24. 01. 2021

[GO21] Geißler, O.; Ostler, U.: Die Herzmuskeln eines Computers – Was ist ein Rechenkern (Core)?, 19.02.2021. URL: https://www.datacenter-insider.de/was-ist-ein-rechenkern-core-a-820808/

[Gop13] Gopalappa, S.: Data Mining for Supply Chain Management, hrsg. von Ecanarys. 2013. URL: https://www.ecanarys.com/Blogs/ArticleID/98/Data-Mining-for-Supply-Chain-Management, 04. 12. 2020

[Gut+17] Gutenschwager, K. u. a.: Simulation in Produktion und Logistik – Grundlagen und Anwendungen, Springer Vieweg, Berlin, Heidelberg, 2017, ISBN: 978-3-662-55745-7. DOI: 10.1007/978-3-662-55745-7

[Haa17] Haas, A.: Intelligence Systeme im Logistik- und Supply Chain Management, Dissertation. Schriftenreihe der HHL Leipzig Graduate School of Management. 2017, ISBN: 978-3-658-21466-1. DOI: 10.1007/978-3-658-21466-1

[HAN03] Hong W., Ankenman; N.: „Controlled sequential bifurcation: a new factor-screening method for discrete-event simulation“. In: *Proceedings of the 2003 Winter Simulation Conference, 2003.* Bd. 1. 2003, 565–573 Vol.1. DOI: 10.1109/WSC.2003.1261470

[Han10] Hanser, Eckhart: Agile Prozesse: von XP über Scrum bis MAP, eXamen.press. Springer, Berlin, 2010, ISBN: 978-3-642-12313-9

[HJ02] Horne, G. E.; Johnson S., Hrsg.: Maneuver Warfare Science 2002, 2002, URL: http://projectalbert.org/files/MWS2002On-line.pdf, 04. 12. 2020

[HJ03] Horne, G. E.; Johnson S.: Maneuver Warfare Science 2003. 2003, URL: http://projectalbert.org/files/MWS2003On-line.pdf, 04. 12. 2020

[HL01] Horne, G. E.; Leanardi, M., Hrsg.: Maneuver Warfare Science 2001, 2001, URL: http://projectalbert.org/files/MWS2001On-line.pdf, 04. 12. 2020

[HM05] Horne, G. E.; Meyer, T.: Data Farming: Discovering Surprise – 2005 Winter Simulation Conference (WSC). Winter Simulation Conference, Association for Computing Machinery und Institute of Electrical and Electronics Engineers, New York, N.Y und Piscataway, N.J, 2005,

[HN08] Hompel, M. t.; Nagel L.: Zellulare Transportsysteme - Den Dingen Beine machen im "Internet der Dinge". Fraunhofer-Institut für Materialfluss und Logistik, Dortmund, 2008,

[Hrd+97] Hrdliczka, V. u. a.: Leitfaden für Simulationsbenutzer in Produktion und Logistik. ASIM, 1997,

[HS16] Horne, G. E.; Schwierz, K.: Sumary of Data Farming. 2016, URL: https://www.researchgate.net/publication/296625383_Summary_of_Data_Farming, 06. 12. 2020

[Hub+19] Huber, D. u. a.: Data Farming Services: Mirco-Services For Facilitating Data Farming in NATO – 2019 Winter Simulation Conference (WSC). Piscataway, NJ, 2019,

[Ila97] Ilachinski, A.: Irreducible Semi-Autonomous Adaptive Combat (ISAAC): An Artificial-Life Approach to Land Warwarfe. Hrsg. von Center for Naval Analyses. 1997, URL: https://www.semanticscholar.org/paper/Irreducible-Semi-Autonomous-Adaptive-Combat-An-to-Ilachinski/5fcbbd494a908448dc5704f8cb2adf2d651b2fc0, 31. 01. 2021

[Int21] Intel: Für die HT-Technologie geeigneter Intel Pentium 4 Prozessor 640, 2021. URL: https://ark.intel.com/content/www/de/de/ark/products/27480/intel-pentium-4-processor-640-supporting-ht-technology-2m-cache-3-20-ghz-800-mhz-fsb.html, 19. 02. 2021

[Kah+18] Kahlenborn, W. u. a.: Die Zukunft im Blick: Konsum 4.0 Wie Digitalisierung den Konsum verändert – Trendbericht zur Abschätzung der Umwelteinwirkungen. Hrsg. von Umweltbundesamt. 2018, URL: https://www.umweltbundesamt.de/publikationen, 21. 12. 2020

[Kle09] Kleijnen, J. P.C.: Factor Screening in Simulation Experiments: Review of Sequential Bifurcation. 2009, DOI: 10.1007/b110059_8. URL: https://www.researchgate.net/publication/226136306, 19. 02. 2021

[Kle15] Kleijnen, J. P. C.: Design and analysis of simulation experiments, Second edition. Bd. 230. International series in operations research & management science. Springer, Cham u. a., 2015, ISBN: 978-3-319-18087-8

[Kle20] Kleppmann, W.: Versuchsplanung – Produkte und Prozesse optimieren, 10. überarbeitete Auflage. Praxisreihe Qualität. München, 2020, ISBN: 9783446463974

[Koe11] Koether, R., Hrsg.: Taschenbuch der Logistik, ger. 4., aktualisierte und erw. Aufl. Fachbuchverl. Leipzig im Carl-Hanser-Verl., München, 2011, 614 S. ISBN: 978-3-446-42512-5

[KS03] Kleijnen, J. P. C.; Smits, M. T.: Performance metrics in supply chain management. In: Journal of the Operational Research Society, 54.5 (2003), S. 507–514. ISSN: 0160-5682. DOI: 10.1057/palgrave.jors.2601539

[KSC03] Kleijnen, J. P.C., Sanchez, S. M.; Cioppa T. M.: A user's guide to the brade new world of designing simulation experiments. 2003, URL: https://www.researchgate.net/publication/4760338_A_user%27s_guide_to_the_brave_new_world_of_designing_simulation_experiments, 18. 02. 2021

[Kus06] Kusiak, A., Hrsg.: Data mining and knowledge discovery approaches based on rule induction techniques – Data Farming: Concepts and methods, unter Mitarb. von Triantaphyllou, E.; Felici, G. Massive computing. Springer, New York, N.Y., 2006, ISBN: 978-0-387-34294-8

[Law05] Lawlor, M.: Data Farming Cultivates New Insights, hrsg. von Armed Forces Communications and Electronics Association. 2005. URL: https://www.afcea.org/content/?q=node/975

[Law07] Law, A. M.: Simulation modeling and analysis, 4th ed. McGraw-Hill series in industrial engineering and management science. McGraw-Hill, Boston, 2007, ISBN: 978-0070667334

[Law13] Law, A. M.: Simulation modeling and analysis, 5th ed. McGraw-Hill series in industrial engineering and management science. McGraw-Hill Education, Dubuque, 2013, ISBN: 978-0-07-340132-4

[Mas08] Mason, S.: Data Farming Around the world Overview – 2008 Winter Simulation Conference (WSC). Piscataway, NJ, 2008,

[Mat21a] Mathworks: Run Script as Batch Job, 2021. URL: https://de.mathworks.com/help/parallel-computing/run-script-as-batch-job.html, 01. 04. 2021

[Mat21b] Mathworks: WSDL (Web Services Description Language), 2021. URL: https://de.mathworks.com/help/matlab/call-wsdl-web-services.html?s_tid=CRUX_topnav, 02. 04. 2021

[Mei12] Meintrup, D.: Statistische Versuchsplanung mit JMP - von der Klassik zur Moderne – Konferenz der SAS-Anwender in Forschung und Entwicklung. Technische Universität Dresden, 2012, URL: http://de.saswiki.org/wiki/KSFE_2012, 12. 01. 2021

[Mer82] Mertens, P.: Simulation, 2., neu bearb. Aufl. Bd. 61. Sammlung Poeschel. Poeschel, Stuttgart, 1982, ISBN: 9783791091242

[Mic21] Microsoft: High Performance Computing in Azure – Innovation durch führende HPC-Cloudlösungen, 2021. URL: https://azure.microsoft.com/de-de/solutions/high-performance-computing/, 01. 04. 2021

[Min21] Minitab: Plackett-Burman-Versuchspläne, 2021. URL: https://support.minitab.com/de-de/minitab/18/help-and-how-to/modeling-statistics/doe/supporting-topics/factorial-and-screening-designs/plackett-burman-designs/, 14. 04. 2021

[MK11] März, L.; Krug, Wi., Hrsg.: Simulation und Optimierung in Produktion und Logistik – Praxisorientierter Leitfaden mit Fallbeispielen, Bd. 130. ASIM-Mitteilung. Springer, Heidelberg, 2011, ISBN: 978-3-642-14536-0

[MM89] Mattern, F.; Mehl, H.: Diskrete Simulation - Prinzipien und Probleme der Effizienzsteigerung durch Parallelisierung. In: Informatik Spektrum, 4 (1989), S. 198–210

[NAT14] NATO: Data farming in support of NATO. Neuilly-sur-Seine Cedex, 2014, URL: https://www.sto.nato.int/publications/, 04. 02. 2021

[NAT18] NATO: Developing Actionable Data Farming Decision Support for NATO, NATO, Research and Technology Organisation, Neuilly-sur-Seine, 2018, ISBN: 978-92-837-2151-2. URL: https://www.sto.nato.int/publications, 05. 02. 2021

[Nav21] Naval Postgraduate School: SEEED Center for Data Farming - Software Downloads, 2021. URL: https://nps.edu/web/seed/software-downloads, 14. 04. 2021

[NWK07] Novozhilov, A., Wolf, Y.; Koonin, E.: Evolution of the genetic code: Partial optimization of a random code for robustness to translation error in a rugged fitness landscape. In: Biology direct, 2 (2007). 02, S. 24. DOI: 10.1186/1745-6150-2-24

[Pfo18] Pfohl, H.-C.: Logistiksysteme – Betriebswirtschaftliche Grundlagen, 9., neu bearbeitete und aktualisierte Auflage. Springer Vieweg, Berlin, 2018, ISBN: 978-3-662-56228-4

[Reu+18] Reus, N. u. a.: Data Farming Services in Support of Military Decision Making – Meeting proceedings : STO-MP-IST-160-P5. NATO, 2018,

[RSW08] Rabe, M., Spieckermann, S.; Wenzel, S., Hrsg.: Verifikation und Validierung für die Simulation in Produktion und Logistik – Vorgehensmodelle und Techniken, Springer, Berlin, 2008, ISBN: 978-3-540-35282-2

[San10] Sanchez, P.: Introduction to Data Farming – CENIC Conference, Monterey. 2010, URL: https://nps.edu/web/seed/papers-presentations, 13. 12. 2020

[Sei06] Seila, A. F.: Spreadsheet simulation – Proceedings of the 2006 Winter Simulation Conference. 2006, URL: https://www.informs-sim.org/wsc06papers/002.pdf, 16. 03. 2021

[SM08] Sargent, R. G.; Mason, S. J.: Verification and validation of simulations models – Proceedings of the 2008 Winter Simulation Conference, IEEE Service Center, Piscataway, NJ, 2008, ISBN: 978-1-4244-2708-6

[Smi98] Smith, J. E.: Characterizing computer performance with a single number. 1998, DOI: 10.1145/63039.63043. URL: https://dl.acm.org/doi/pdf/10.1145/63039.63043

[SvH17] Siebertz, K., van Bebber, D.; Hochkirchen, T., Hrsg.: Statistische Versuchsplanung – Design of Experiments (DoE), 2. Auflage. VDI-Buch. Springer Vieweg, Berlin, 2017, ISBN: 978-3-662-55743-3

[TC04] Terzi, S.; Cavalieri, S.: Simulation in the supply chain context: a survey. In: Computers in Industry, 53.1 (2004), S. 3–16. ISSN: 0166-3615. DOI: 10.1016/S0166-3615(03)00104-0. URL: https://www.sciencedirect.com/science/article/pii/S0166361503001040

[Top21] Top 500: Top 500. The List – Performance Developement, 2021. URL: https://top500.org/statistics/perfdevel/, 22. 12. 2020

[VDI14] VDI, Hrsg.: Simulation von Logistik-, Materialflussund Produktionssystemen - Blatt 1. 2014,

[VDI18] VDI, Hrsg.: Simulation von Logistik-, Materialfluss- und Produktionssystemen. 2018,

[Ver02] Versteegen, G.: Software Management – Beherrschung des Lifecycles, Xpert.press. Springer, Berlin und Heidelberg, 2002, ISBN: 978-3-642-56367-6. DOI: 10.1007/978-3-642-56367-6

[Wol16] Wollschläger, D.: R kompakt – Der schnelle Einstieg in die Datenanalyse, ger. 2. Auflage. Springer-Lehrbuch. Springer Spektrum, Berlin und Heidelberg, 2016, 277 S. ISBN: 978-3-662-49102-7. URL: http://www.springer.com/

[WR17] Weyer, J.; Roos, M.: Agentenbasierte Modellierung und Simulation – Intrument prospektiver Technikfolgenabschätzung. In: 26 (2017), S. 11–16. URL: https://www.tatup.de/index.php/tatup/issue/view/4, 30. 01. 2021

[Ye98] Ye, K. Q.: Orthogonal Column Latin Hypercubes and Their Application in Computer Experiments. In: Journal of the American Statistical Association, 93.444 (1998), S. 1430–1439. ISSN: 01621459. URL: http://www.jstor.org/stable/2670057

Anhang

A Integrationsansätze

Phasen	Vorgehensmodelle / Vorgehensweisen					Integrationsansätze Stufe 1
	Banks	Law	Sargent	ASIM	MSG Data Farming	
1. Aufgabenanalyse						
1.1 Problemformulierung	☑	☑	☑	☑		**IA St.1/Nr.1:** Eine Problemformulierung ist für DF-Studien nicht notwendig, weil sie keinen Einfluss auf die Versuchsplanung besitzt. Zudem ist auch Optimierungspotential vorhanden, wenn keine Probleme eines realen Systems bekannt sind.
1.2 Zielstellung	☑	☑	☑	☑	☑	
1.3 Projektplan	☑	☑		☑		**IA St.1/Nr.2:** Ein Projektplan ist im industriellen Kontext zwingend erforderlich (Zeitplan, Budgetfreigaben für Soft- bzw. Hardware, Freistellung von Mitarbeitern).
2. Modellformulierung						
2.1 Modellkonzept	☑	☑	☑	☑		**IA St.1/Nr.3:** Ein Modellkonzept ist für die Validierung des realisierten Modells zwingend erforderlich.
2.2 Systemarchitektur						**IA St.1/Nr.4:** Strukturplanung des gesamten IT-Systems (Realisierung von Front- und Backend, Einsatz der HPC-Komponente, Schnittstellengestaltungen, Datenformate usw.).
3. Modellimplementierung						
3.1 Datenbeschaffung / -aufbereitung	☑	☑	☑	☑	(☑)	
3.2 Modellentwicklung	(☑)	(☑)	(☑)	(☑)	☑	
3.3 Szenariogestaltung					☑	**IA St.1/Nr.5:** Analog zum ersten DF-Bereich *Rapid szenario prototyping* ist im Vorgehensmodell eine Phase „Szenariogestaltung“ zur Entwicklung, Änderung und Einbindung von Szenarien erforderlich.
3.4 Modellierung	☑	☑	☑	☑	☑	
3.5 Schnittstellengestaltung					☑	**IA St.1/Nr.6:** Für die Realisierung der Systemarchitektur aus **IA St.1/Nr.4** wird eine Phase zur Gestaltung sämtlicher Schnittstellen benötigt. Das Ergebnis der Schnittstellengestaltung ist ein funktionsfähiges DFS-Portal.

Abb. A.1: Integrationsansätze Stufe I (1/2)

Phasen	Vorgehensmodelle / Vorgehensweisen					Integrationsansätze Stufe 1
	Banks	**Law**	**Sargent**	**ASIM**	**MSG Data Farming**	
4. Modellanwendung						
4.1 Experimentenplanung	☑	☑	☑	☑	☑	
4.2 Durchführung der Experimente	☑	☑	☑	☑	☑	
4.3 Ergebnisanalyse	☑	☑	☑	☑	☑	
Phasenübergreifende Tätigkeiten						
Verifikation und Validierung	☑ Nach 3.4	☑ Nach 2.1, 3.4	☑ Nach jedem Phasen-ergebnis	☑ Nach jedem Phasen-ergebnis	(☑) Nach 3.4	**IA St.1/Nr.7:** Die Konzeptionierung, Entwicklung und Anwendung von DFS in einer SC erfordert permanente V&V des Modellkonzepts, des realisierten Modells, des DF-Systems, der erforderlichen SC-Stammdaten und der generierten Simulationsdaten.
Prozessdokumentation	☑	☑		(☑)		**IA St.1/Nr.8:** Die Konzeptionierung, Entwicklung und Anwendung von DFS in einer SC erfordert die kontinuierliche projektbegleitende Entwicklung einer ausführlichen Prozessdokumentation, um technische Umsetzungen oder getroffene Entscheidungen rückwirkend nachvollziehen zu können.
Kollaboration					☑	

Abb. A.2: Integrationsansätze Stufe I (2/2)

Phasen		Anforderungen für DFS in einer SC		Anforderungen erfüllt?	Integrationsansätze Stufe 2
		Ablaufbedingte Anforderungen	**Technische Anforderungen**		
Aufgabenanalyse	1. Zielstellung	• Definition und Gewichtung von Zielgrößen • Gesamtziel muss wirtschaftlich beurteilt werden können	-	☑	-
	2. Projektplan	• Terminierung einzelner Phasen • Zuweisung von Rollen und Zuständigkeiten • Budgetplanung	-	☑	-
Konzeptionierung	3. Modellkonzept	• Regeln und Strategien für das Systemverhalten so einfach wie möglich formulieren	-	☒	**IA St.2/Nr.1:** Die Konzeptionierung des DF-Modells muss auf einer **hohen** Abstraktionsebene erfolgen.
	4. Systemarchitektur	• Planung aller Funktionalitäten des DFS-Portals inkl. der HPC-Komponente • Interoperabilität gewährleisten	-	-	**IA St.2/Nr.2:** Vergleich und Bewertung verschiedener HPC-Dienste und Durchführung einer Machbarkeitsstudien zur Gewährleistung der allgemeinen Interoperabilität.

Abb. A.3: Integrationsansätze Stufe II (1/3)

Phasen		Anforderungen für DFS in einer SC		Anforderungen erfüllt?	Integrationsansätze Stufe 2
		Ablaufbedingte Anforderungen	**Technische Anforderungen**		
Realisierung	5. Datenbeschaffung / - aufbereitung	• Klare Erkenntnis über die Art und den Umfang erforderlicher SC-Daten	• Interorganisationale zentrale Datenbanken bei erforderlichen SC-Prozessdaten	☒	**IA St.2/Nr.3:** Unterstützung und Förderung kollaborativer Planung mit allen Akteuren einer SC.
	6. Modellentwicklung	• Einhaltung des Detaillierungsgrades • Verfassung einer Programmdokumentation • Zusammenarbeit mit den Fachexperten	• HPC-fähig • Modularer Aufbau • Hohe Geschwindigkeit • Hohe Transparenz	☒	**IA St.2/Nr.4:** Zur Gewährleistung der HPC-Fähigkeit eines DF-Modells ist die Implementierung in Programmiersprachen erforderlich.
	7. Szenariogestaltung	• Zusammenarbeit mit den Fachexperten	• Schnittstellenformate für den Export und Import des Szenarios • Schnelle Anpassungen der Szenarien gewährleisten • Intuitive Gestaltung / Bedienung	☒	**IA St.2/Nr.5:** Entwicklung intuitiver Szenario-Editoren, mit denen Anwender schnell neue Szenarien erstellen bzw. bearbeiten und für das Modell bereitstellen können.
	8. Modellierung	• Umsetzung des Szenarios aus dem Modellkonzept	-	☑	-
	9. Schnittstellengestaltung	• Hohe Qualifikation und Zusammenarbeit der Simulationsexperten	• Gewährleistung der Datenbankkommunikation • Erforderliche Hardware für Webserver und On-Premises- / Cloud Datenbanken • Ausführung der Simulationsläufen über die HPC-Komponente	-	**IA St.2/Nr.6:** Für die Implementierung des DFSS sind hochqualifizierte Programmierer erforderlich.

Abb. A.4: Integrationsansätze Stufe II (2/3)

Phasen		Anforderungen für DFS in einer SC		Anforderungen erfüllt?	Integrationsansätze Stufe 2
		Ablaufbedingte Anforderungen	Technische Anforderungen		
Modellanwendung	10. Experimentenplanung	• Maximierung der Ergebnislandschaft • Verwendung mehrerer unterschiedlicher Versuchspläne	• Effizienz (hohes Informationsgewinn/Zeit - Verhältnis)	☒	**IA St.2/Nr.7:** Für die Versuchspläne von DF-Studien eignen sich ausschließlich What-if-Analysen mit statischen Faktor-Designs **IA St.2/Nr.8:** Verwendung alternativer Versuchspläne, die für komplexe Modelle und/oder vielen Faktoren geeignet sind.
	11. Durchführung der Experimente	-	• Automatisierte Ausführung / Speicherung der Ergebnisse	☒	**IA St.2/Nr.9:** Das DFSS muss die Durchführung einer DF-Studie komplett automatisiert ohne menschliche Eingriffe bewerkstelligen können.
	12. Ergebnisanalyse	• Bewältigung großer Datenmengen	• Leistungsfähige Analysesoftware • Übersichtliche Statistiken über viele Faktoren und Zielgrößen	☑	-

Abb. A.5: Integrationsansätze Stufe II (3/3)

B Frontend

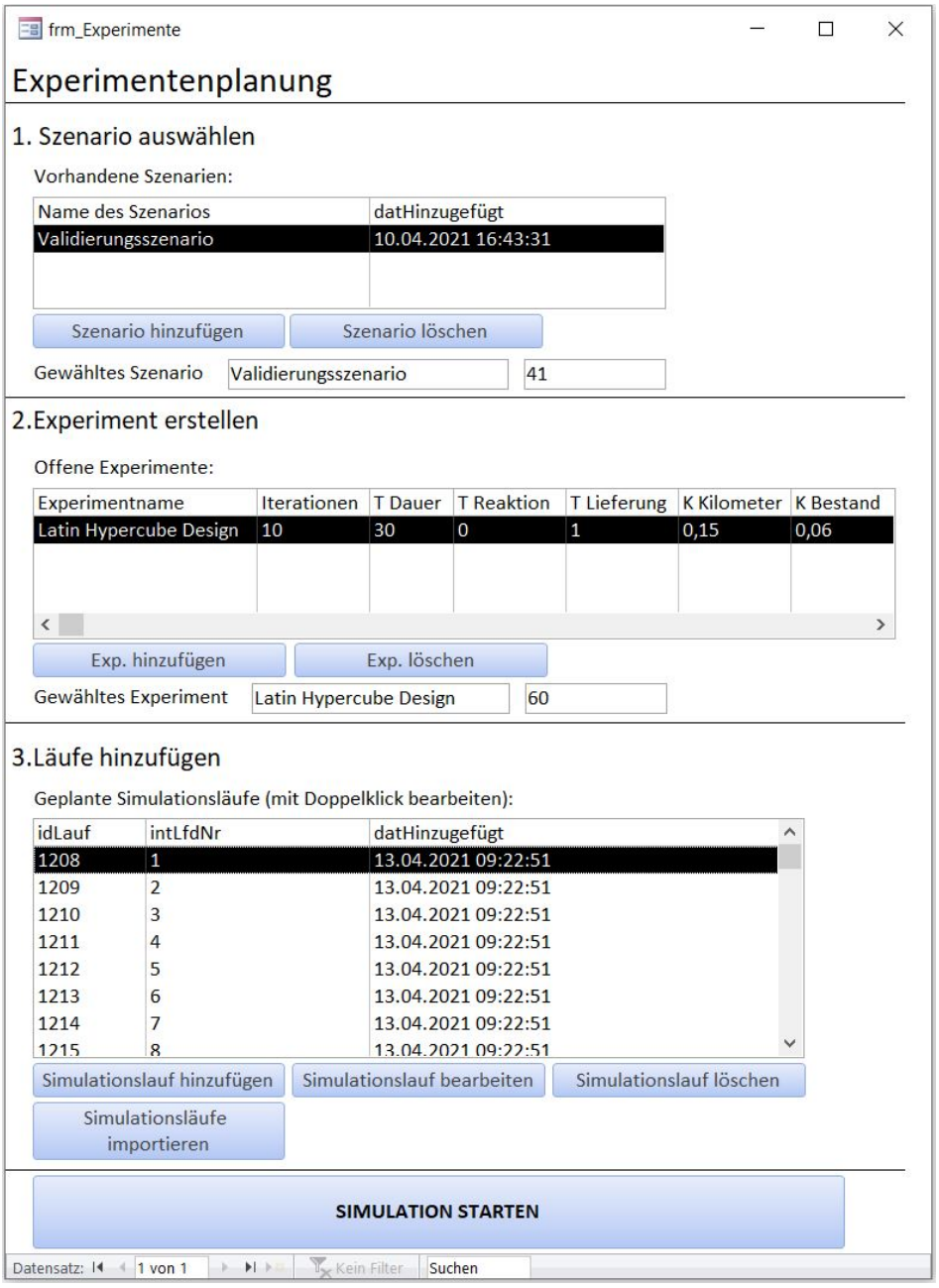

Abb. B.1: Startformular des Data Farming Service Systems

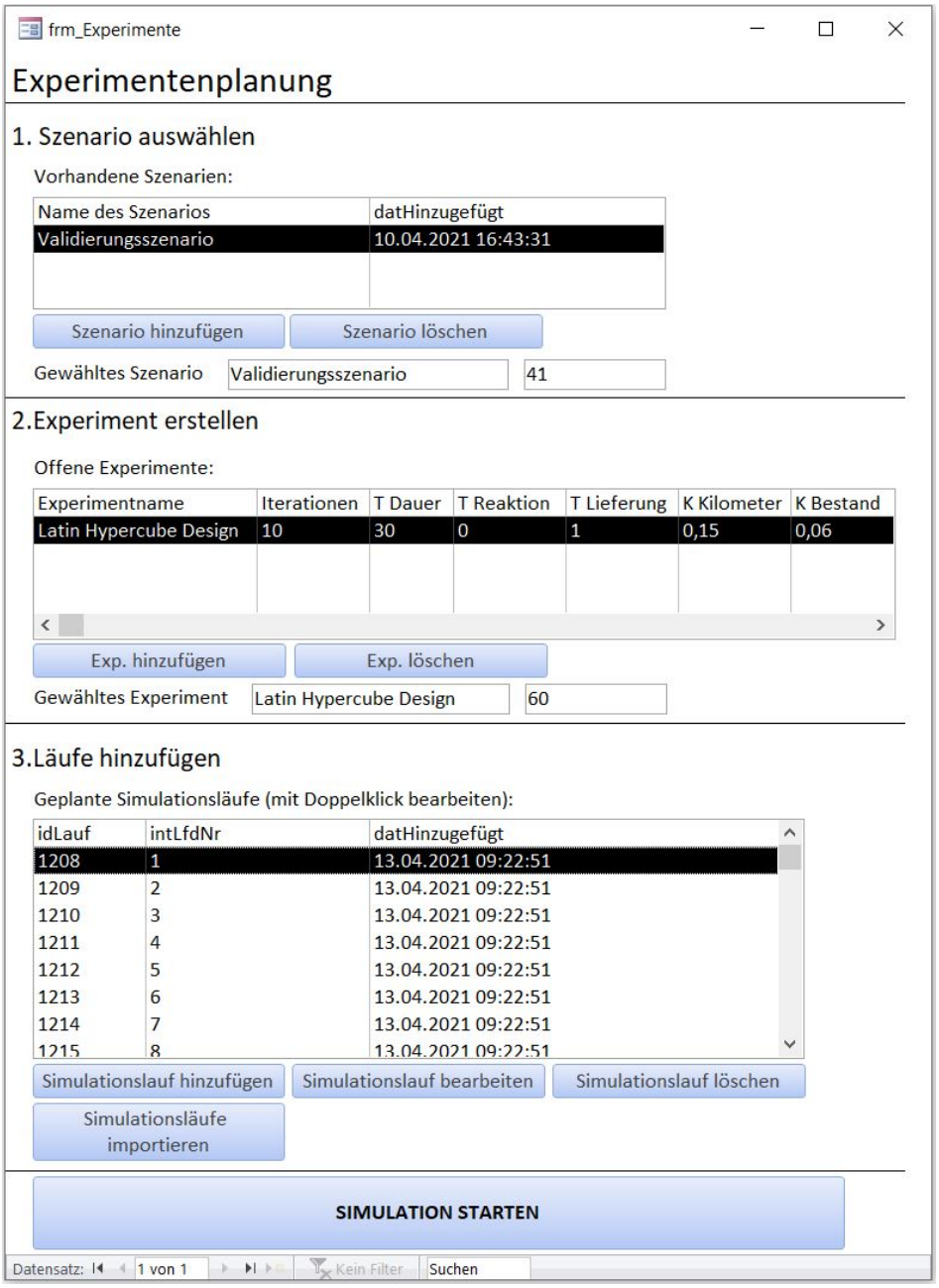

Abb. B.2: Formular zum Hinzufügen eines Experimentes

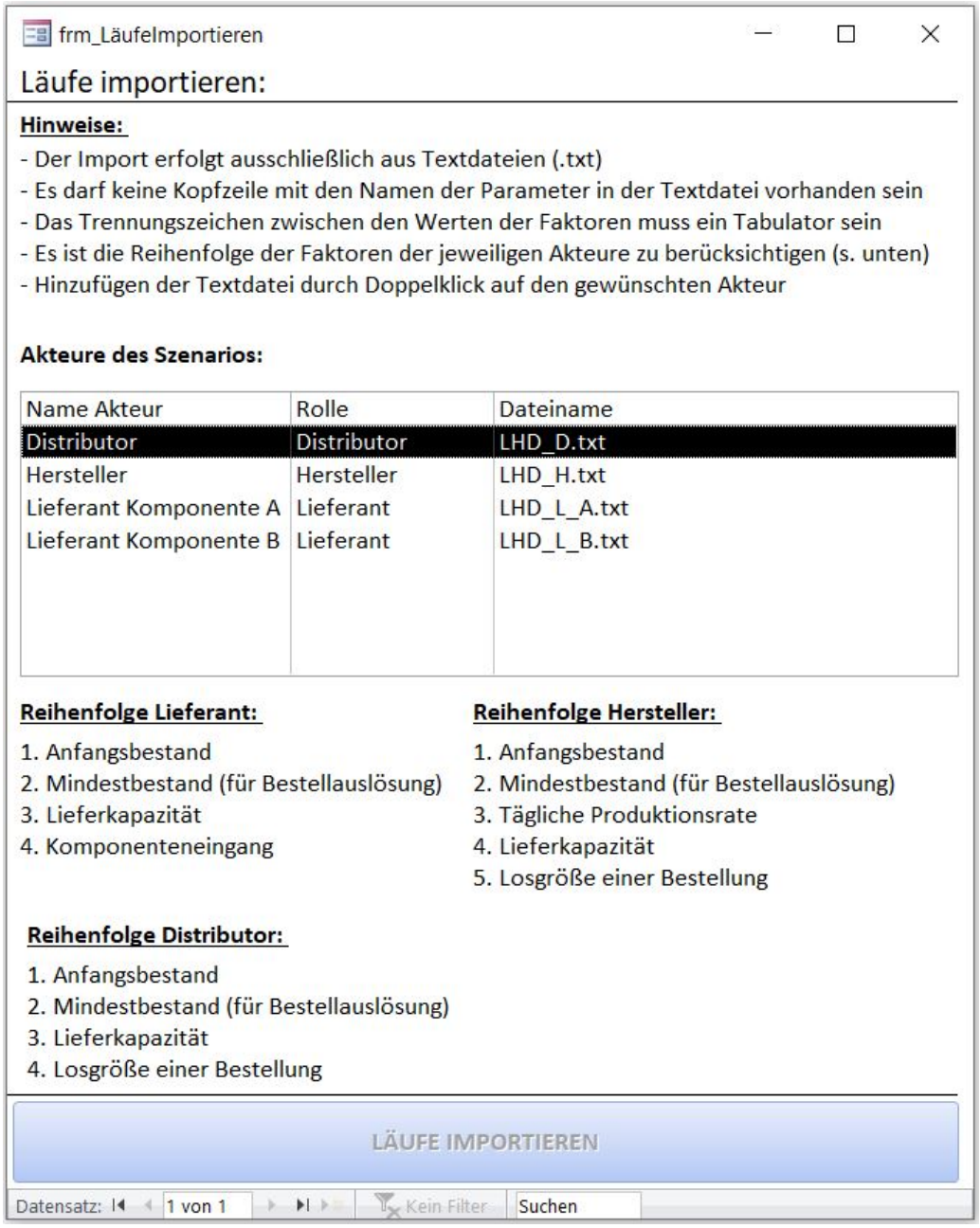

Abb. B.3: Formular zum Importieren der Versuchsläufe

C Backend

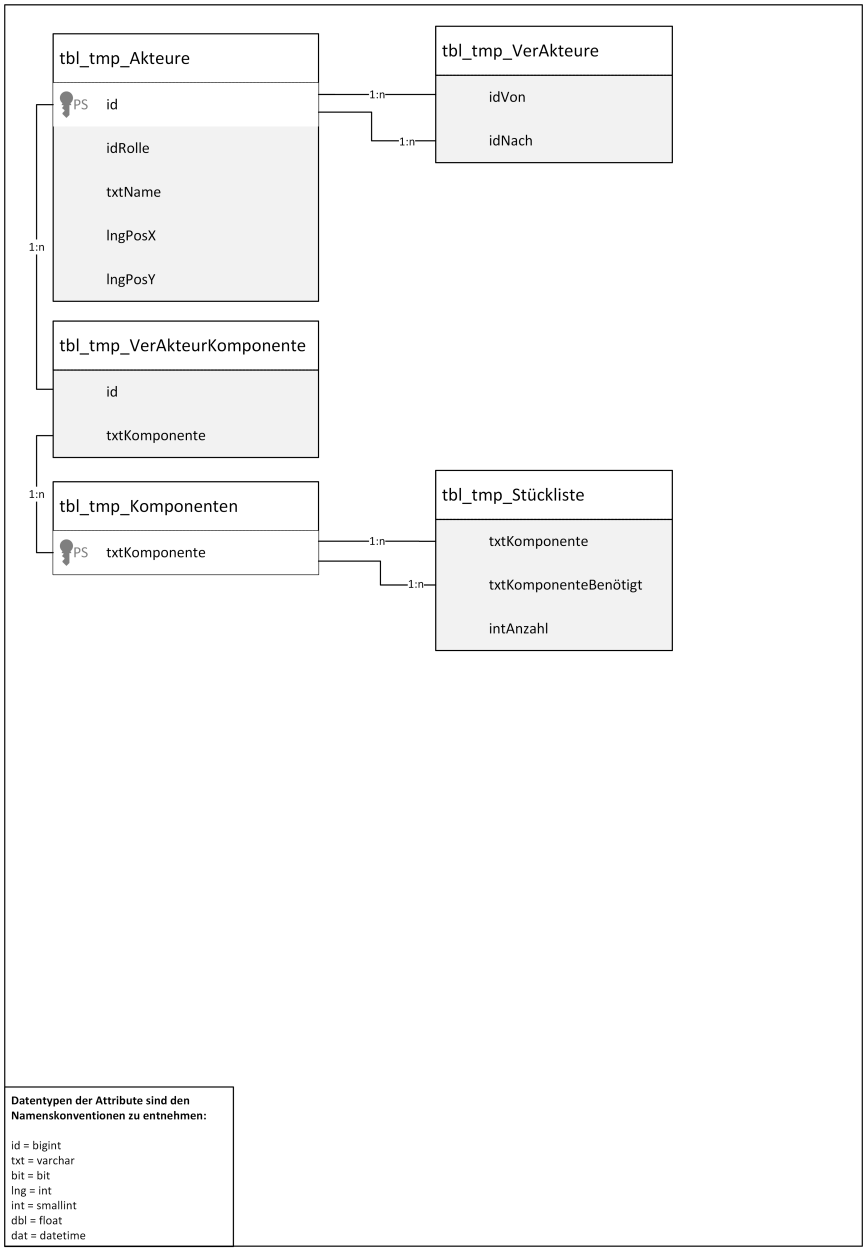

Abb. C.1: Backend des DFSSs (1/4) - Transfertabellen für den XML-Import des Szenarios

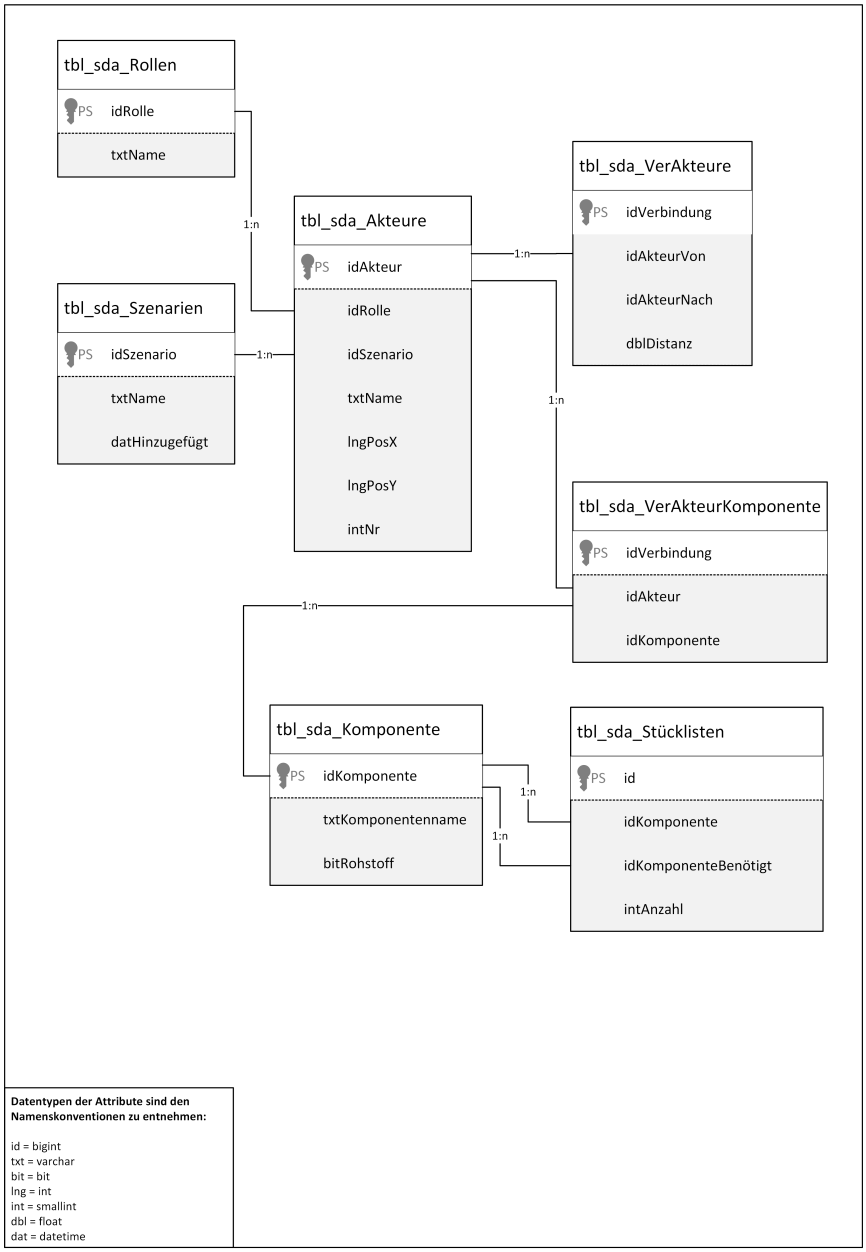

Abb. C.2: Backend des DFSSs (2/4) - Stammdatentabellen

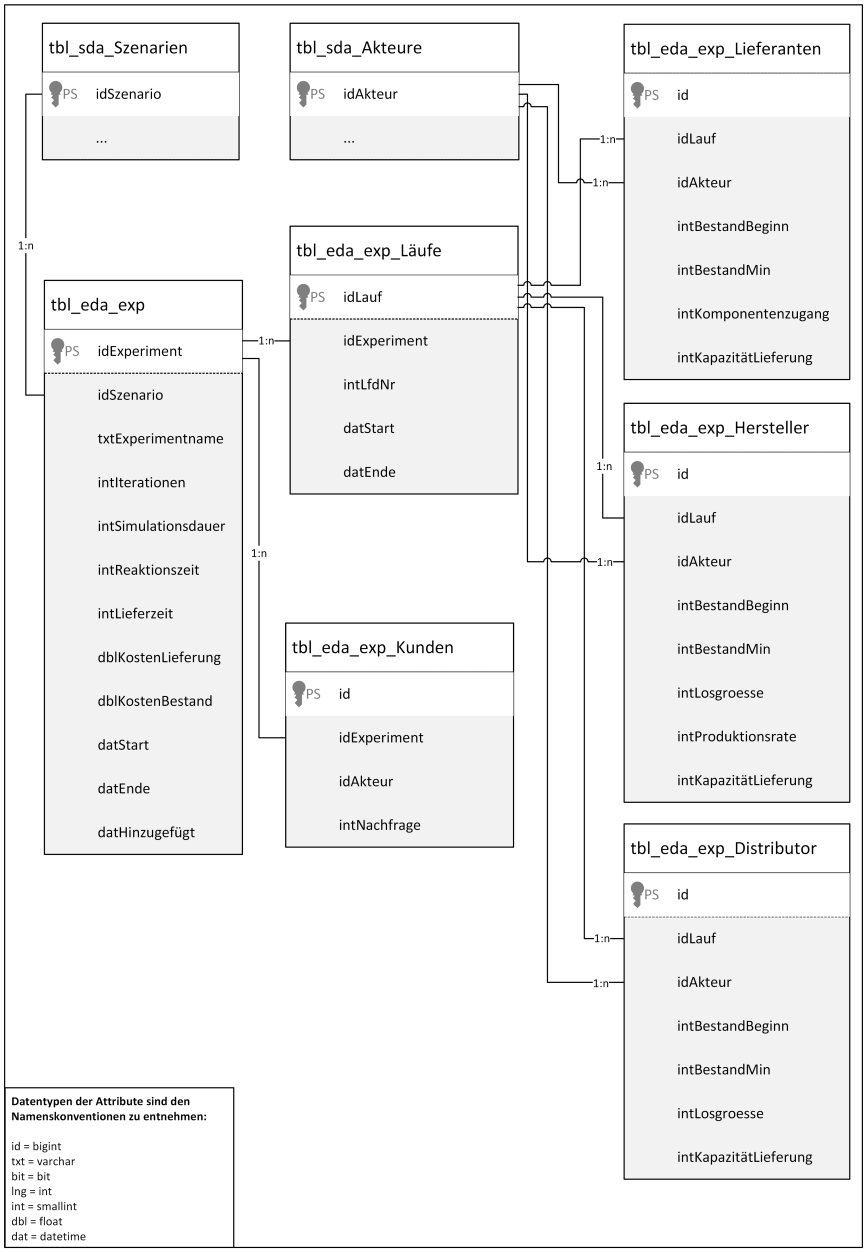

Abb. C.3: Backend des DFSSs (3/4) - Tabellen für die Versuchsplanung

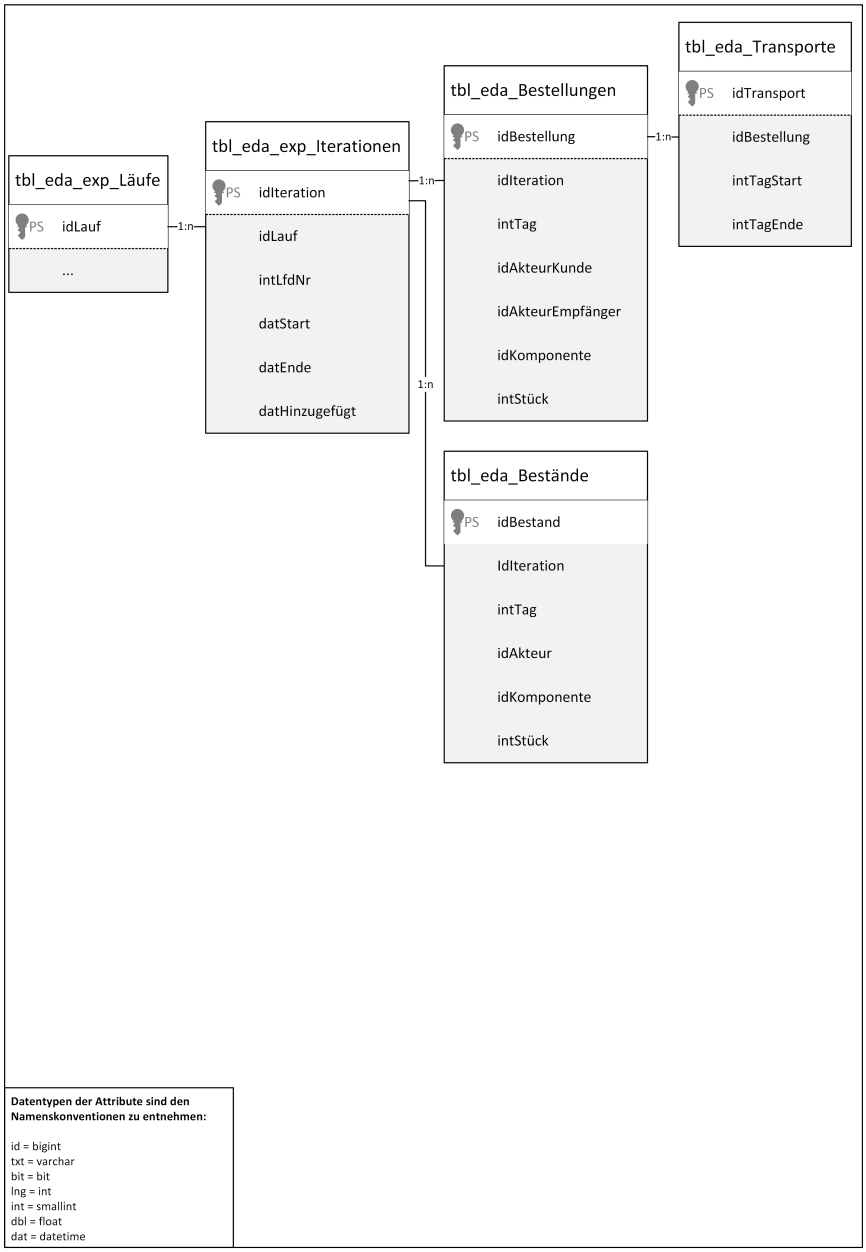

Abb. C.4: Backend des DFSSs (4/4) - Ereignisdatentabellen/Simulationsergebnisse

D SQL-Abfragen

```
DECLARE @XML AS XML
SELECT @XML = BulkColumn
FROM OPENROWSET(BULK 'Pfad der XML-Datei', SINGLE_BLOB) AS qxml

INSERT INTO dbo.tbl_tmp_Akteure(id,txtName,idRolle,lngPosX,lngPosY)
SELECT
id = sc.value('@id','int'),
txtName = sc.value('@name','varchar(255)'),
idRolle = sc.value('@idRolle','int'),
lngPosX = sc.value('@lngPosX','int'),
lngPosY = sc.value('@lngPosY','int')
FROM @XML.nodes('data/akteure/akteur') AS XTbl(sc)
DECLARE @XML AS XML
SELECT @XML = BulkColumn
FROM OPENROWSET(BULK 'Pfad der XML-Datei', SINGLE_BLOB) AS qxml
```

Quellcode D.1: XML-Import (1/5): SC-Akteure

```
DECLARE @XML AS XML
SELECT @XML = BulkColumn
FROM OPENROWSET(BULK 'Pfad der XML-Datei', SINGLE_BLOB) AS qxml

INSERT INTO dbo.tbl_tmp_VerAkteure(idVon,idNach)
SELECT
idVon = sc.value('@id','int'),
idNach = sc.value('@idnach','int')
FROM @XML.nodes('data/akteure/akteur/verbindungen') AS XTbl(sc)
```

Quellcode D.2: XML-Import (2/5): Verbindungen zwischen den SC-Akteuren

```
DECLARE @XML AS XML
SELECT @XML = BulkColumn
FROM OPENROWSET(BULK 'Pfad der XML-Datei', SINGLE_BLOB) AS qxml

INSERT INTO dbo.tbl_tmp_Komponenten(txtKomponente)
SELECT
txtKomponente = sc.value('@komponentenname','varchar(255)')
FROM @XML.nodes('data/komponenten/komponente') AS XTbl(sc)
```

Quellcode D.3: XML-Import (3/5): Import der Komponenten/Produkte

```
DECLARE @XML AS XML
SELECT @XML = BulkColumn
FROM OPENROWSET(BULK 'Pfad der XML-Datei', SINGLE_BLOB) AS qxml

INSERT INTO dbo.tbl_tmp_VerAkteurKomponente(id,txtKomponente)
SELECT
id = sc.value('@id','int'),
txtKomponente = sc.value('@komponentenname','varchar(255)')
FROM @XML.nodes('data/akteure/akteur/komponenten') AS XTbl(sc)
```

Quellcode D.4: XML-Import (4/5): Information welcher Akteure welche Komponente/welches Produkt liefert oder herstellt

```
DECLARE @XML AS XML
SELECT @XML = BulkColumn
FROM OPENROWSET(BULK 'Pfad der XML-Datei', SINGLE_BLOB) AS qxml

INSERT INTO dbo.tbl_tmp_Stückliste(txtKomponente,txtKomponenteBenötigt,intAnzahl)
SELECT
txtKomponente = sc.value('@komponente','varchar(255)'),
txtKomponenteBenötigt = sc.value('@komponenteB','varchar(255)'),
intAnzahl = sc.value('@anzahl','int')
FROM @XML.nodes('data/komponenten/komponente/stueckliste') AS XTbl(sc)
```

Quellcode D.5: XML-Import (5/5): Stücklisten der Produkte

```
1  WITH qry_lf AS
2  (SELECT lf.idLauf
3  FROM dbo.tbl_eda_exp experiment
4  INNER JOIN dbo.tbl_eda_exp_Läufe lf
5  ON experiment.idExperiment = lf.idExperiment
6  WHERE experiment.idExperiment = 55),
7
8  qry_tr AS
9  (SELECT lf.idLauf, it.idIteration, bs.idBestellung,
10 bs.idAkteurKunde AS Kunde, bs.idAkteurEmpfänger AS Lieferant,
11 bs.intStück AS StkBestellt,
12 SUM(IIF(tr.idTransport IS NOT NULL, 1, 0)) AS AnzTransporte,
13  ISNULL(SUM(tr.intStück), 0) AS StkTransportiert,
14 SUM(ver.dblDistanz) AS Kilometer,
15 AVG(tr.intTagEnde − bs.intTag) AS [MittlereDauer]
16 FROM qry_lf lf
17 INNER JOIN dbo.tbl_eda_exp_Iterationen it
18 ON lf.idLauf = it.idLauf
19 INNER JOIN dbo.tbl_eda_Bestellungen bs
20 ON it.idIteration = bs.idIteration
21 LEFT JOIN dbo.tbl_eda_Transporte tr
22 ON bs.idBestellung = tr.idBestellung AND tr.intTagEnde IS NOT NULL
23 INNER JOIN dbo.tbl_sda_VerAkteure ver
24 ON bs.idAkteurKunde = ver.idAkteurNach AND bs.idAkteurEmpfänger = ver.idAkteurVon
25 GROUP BY lf.idLauf, bs.intStück, it.idIteration,
26 bs.idBestellung, bs.idAkteurKunde, bs.idAkteurEmpfänger),
27
28 qry_stk AS
29 (SELECT idLauf, AVG(StkBestellt) AS StkBestellt,
30 AVG(StkTransportiert) AS StkGeliefert
31 FROM
32     (SELECT qry_tr.idLauf, qry_tr.IdIteration,
33     SUM(qry_tr.StkBestellt) AS StkBestellt,
34     SUM(qry_tr.StkTransportiert) AS StkTransportiert
35     FROM qry_tr
36     INNER JOIN dbo.tbl_sda_Akteure akt
37     ON qry_tr.Kunde = akt.idAkteur
38     WHERE akt.idRolle = 1
39     GROUP BY qry_tr.idLauf, qry_tr.IdIteration) AS qry
40 GROUP BY idLauf),
```

Quellcode D.6: Datentransformation für die Experimentenauswertung (1/3)

```
qry_bs AS
(SELECT idLauf, AVG(BestandSumme) As SummeBestand
    FROM
    (SELECT lf.idLauf, it.idIteration,
    SUM(bs.intStück) AS [BestandSumme]
    FROM qry_lf lf
    INNER JOIN dbo.tbl_eda_exp_Iterationen it
    ON lf.idLauf = it.idLauf
    INNER JOIN dbo.tbl_eda_Bestände bs
    ON it.idIteration = bs.idIteration
    GROUP BY lf.idlauf, it.idIteration)
    AS qry
GROUP BY idLauf)

SELECT DISTINCT lf.idLauf,
--Eingabeparameter
lief1.intBestandBeginn AS L_A_Banf,
lief1.intBestandMin AS L_A_Bmin,
lief1.intKapazitätLieferung AS L_A_Lkapa,
lief1.intKomponentenzugang As L_A_Zugang,
lief2.intBestandBeginn AS L_B_Banf,
lief2.intBestandMin AS L_B_Bmin,
lief2.intKapazitätLieferung AS L_B_Lkapa,
lief2.intKomponentenzugang As L_B_Zugang,
her.intBestandBeginn AS H_Banf,
her.intBestandMin AS H_Bmin,
her.intProduktionsrate AS H_Pr,
her.intKapazitätLieferung AS H_Lkapa,
her.intLosgroesse As H_Los,
dis.intBestandBeginn AS D_Banf,
dis.intBestandMin AS D_Bmin,
dis.intKapazitätLieferung AS D_Lkapa,
dis.intLosgroesse As D_Los,
(lief1.intBestandBeginn + lief2.intBestandBeginn +
her.intBestandBeginn + dis.intBestandBeginn) AS SummeAnfangsbestand,
(her.intLosgroesse + dis.intLosgroesse) As SummeLosgroesse,
(lief1.intKapazitätLieferung + lief2.intKapazitätLieferung +
her.intKapazitätLieferung + dis.intKapazitätLieferung) As SummeKapazität,
```

Quellcode D.7: Datentransformation für die Experimentenauswertung (2/3)

```
--Ergebnisse
bs.SummeBestand,
ROUND(CAST((bs.Summebestand*experimente.dblKostenBestand) AS float), 2)
AS Bestandskosten,
tr.AnzahlTransporte, tr.Kilometer, ROUND(tr.Kilometer*experimente.dblKostenLieferung, 2) AS
    Transportkosten,
ROUND((tr.Kilometer*experimente.dblKostenLieferung) +
(bs.SummeBestand*experimente.dblKostenBestand), 2) As SCKosten, stk.StkGeliefert, stk.StkBestellt,
ROUND((CAST(StkGeliefert AS float)/CAST(StkBestellt AS float)), 2)
As [% Geliefert],
IIF(stk.StkGeliefert <> 0,
ROUND(((tr.Kilometer*experimente.dblKostenLieferung) + (bs.SummeBestand*experimente.
    dblKostenBestand))/stk.StkGeliefert, 2)
, NULL) AS [Kosten/Stk]
FROM qry_lf lf
INNER JOIN dbo.tbl_eda_exp_Läufe lfsda
ON lf.idLauf = lfsda.idLauf
INNER JOIN dbo.tbl_eda_exp experimente
ON lfsda.idExperiment = experimente.idExperiment
LEFT JOIN dbo.tbl_eda_exp_Lieferanten lief1
ON lf.idLauf = lief1.idLauf AND lief1.idAkteur = 114
LEFT JOIN dbo.tbl_eda_exp_Lieferanten lief2
ON lf.idLauf = lief2.idLauf AND lief2.idAkteur = 115
LEFT JOIN dbo.tbl_eda_exp_Hersteller her
ON lf.idLauf = her.idLauf
LEFT JOIN dbo.tbl_eda_exp_Distributor dis
ON lf.idLauf = dis.idLauf
LEFT JOIN qry_bs bs ON lf.idLauf = bs.idLauf
LEFT JOIN qry_tr2 tr ON lf.idLauf = tr.idLauf
LEFT JOIN qry_stk stk ON lf.idLauf = stk.idLauf
ORDER BY
ROUND((CAST(StkGeliefert AS float)/CAST(StkBestellt AS float)), 2) DESC
```

Quellcode D.8: Datentransformation für die Experimentenauswertung (3/3)

E Versuchspläne

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17
Min	**0**	**20**	**20**	**20**	**0**	**20**	**10**	**20**	**20**	**0**	**20**	**20**	**10**	**0**	**20**	**20**	**10**
Max	**100**	**70**	**50**	**50**	**100**	**70**	**50**	**50**	**50**	**100**	**70**	**50**	**50**	**100**	**70**	**50**	**50**
Lauf																	
1	0	20	20	50	0	20	10	20	50	100	20	50	10	100	20	20	10
2	100	70	20	50	0	70	10	20	20	100	70	20	50	100	20	20	50
3	0	70	50	50	0	70	10	50	20	0	70	50	50	100	20	50	50
4	100	70	20	20	0	20	50	20	20	0	20	50	50	0	70	20	50
5	0	20	50	50	0	70	10	50	20	0	20	50	50	0	70	50	10
6	0	20	50	20	0	20	10	50	50	0	70	20	50	0	20	20	50
7	100	70	50	50	0	70	50	50	50	100	20	20	10	0	70	20	10
8	100	70	20	50	100	20	10	50	20	0	70	50	50	0	70	20	50
9	100	20	50	20	0	70	50	50	50	0	70	50	50	100	70	20	10
10	0	70	50	50	100	20	50	50	50	100	70	20	10	0	20	50	10
11	100	70	50	50	0	20	10	20	50	0	20	20	10	100	70	20	50
12	100	20	50	50	100	70	50	20	20	0	20	50	10	0	20	20	50
13	0	20	50	20	0	70	50	50	20	100	20	50	10	0	70	50	50
14	100	20	20	20	0	70	10	20	20	0	70	50	10	100	20	50	10
15	100	70	50	20	0	20	10	50	20	0	20	20	50	100	20	50	10
16	0	20	20	20	0	20	10	20	20	0	20	20	10	0	20	20	10
17	100	20	20	50	100	70	50	20	50	100	70	50	50	0	20	20	10
18	100	20	50	50	0	20	50	20	20	100	70	50	10	100	20	50	10
19	0	70	20	50	0	20	10	50	50	0	70	50	10	0	70	20	10
20	100	70	50	20	100	20	50	20	20	100	70	50	50	0	70	50	50
21	100	70	20	20	100	20	10	50	50	100	20	50	10	100	20	20	50
22	100	20	50	20	100	20	10	20	50	100	20	50	50	0	20	50	10
23	0	20	50	50	0	70	50	20	20	100	20	20	50	100	70	20	50
24	0	70	50	50	100	70	10	20	20	0	70	20	10	0	20	50	50
25	0	20	50	50	100	20	50	20	50	0	20	50	50	100	70	20	50
26	0	70	20	20	0	20	50	50	20	100	20	50	10	0	20	50	50
27	100	70	20	50	0	70	10	20	50	100	70	50	10	100	70	50	50
28	0	20	20	20	100	20	10	20	20	100	70	20	50	0	70	20	10
29	100	70	50	50	100	20	10	20	20	100	20	20	10	0	70	50	10
30	0	70	50	20	100	70	10	20	50	0	20	50	50	100	20	50	10
31	0	70	50	20	100	20	50	20	20	0	70	50	10	100	70	20	10
32	0	20	20	50	100	20	50	20	50	0	20	20	50	100	20	50	50
33	0	70	20	20	100	70	50	20	50	0	70	20	10	100	70	50	50
34	100	20	20	50	100	70	10	50	20	100	20	20	50	100	70	50	10
35	0	20	20	20	100	70	10	50	20	100	20	20	10	100	70	20	50
36	0	70	20	50	0	20	50	50	50	100	20	50	50	100	70	50	10
37	0	20	50	50	100	70	10	50	50	100	70	50	10	0	20	20	50
38	0	70	50	20	0	70	10	20	50	100	70	20	50	0	70	20	10
39	100	70	20	50	100	70	50	50	20	0	20	20	50	0	20	20	10
40	100	70	50	20	100	70	50	50	50	0	20	20	10	100	20	20	10
41	100	20	20	20	0	70	50	20	50	0	70	20	10	0	70	50	10
42	100	20	50	20	100	20	10	50	50	100	70	20	50	100	70	50	50
43	0	20	20	50	100	20	50	50	20	0	70	20	10	100	70	50	10
44	100	20	20	20	100	70	10	50	50	0	20	50	10	0	70	50	50
45	0	70	20	20	100	70	50	50	20	100	70	50	50	100	20	20	10
46	100	20	50	20	0	20	50	50	20	100	70	20	10	100	20	20	50
47	100	20	20	50	0	20	50	50	50	0	70	20	50	0	20	50	50
48	0	70	20	20	0	70	50	20	50	100	20	20	50	0	20	50	50

F1	D_Banf
F2	D_Bmin
F3	D_Lkapa
F4	D_Los
F5	H_Banf
F6	H_Bmin
F7	H_Pr
F8	H_Lkapa
F9	H_Los
F10	L_A_Banf
F11	L_A_Bmin
F12	L_A_Lkapa
F13	L_A_Zugang
F14	L_B_Banf
F15	L_B_Bmin
F16	L_B_Lkapa
F17	L_B_Zugang

Abb. E.1: Plackett-Burman Versuchsplan [k=17/n=48]

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17
Min	**0**	**20**	**20**	**20**	**0**	**20**	**10**	**20**	**20**	**0**	**20**	**20**	**10**	**0**	**20**	**20**	**10**
Max	**100**	**70**	**50**	**50**	**100**	**70**	**50**	**50**	**50**	**100**	**70**	**50**	**50**	**100**	**70**	**50**	**50**
Lauf																	
1	83	20	49	20	33	28	20	25	45	35	48	41	42	41	66	37	15
2	28	69	27	32	50	40	44	34	25	0	70	38	10	4	52	43	29
3	2	68	20	45	17	42	47	40	38	50	25	32	36	37	59	49	33
4	17	29	23	38	54	25	40	38	21	80	62	43	46	46	61	25	13
5	54	60	34	29	30	43	33	28	45	63	24	43	44	43	65	35	45
6	37	66	49	25	9	21	34	31	21	72	37	44	25	17	44	34	10
7	85	27	21	36	74	37	26	21	49	39	38	36	20	33	41	32	16
8	93	55	38	35	89	57	43	48	32	57	68	30	43	13	29	47	11
9	72	56	37	30	100	65	47	47	37	46	35	46	22	52	25	24	12
10	78	46	28	27	15	66	30	36	22	17	27	23	41	76	30	30	24
11	59	57	29	21	52	70	38	22	50	89	36	40	26	93	37	44	27
12	20	38	36	33	59	44	20	44	34	43	34	25	30	2	57	21	43
13	4	50	26	41	37	61	33	47	48	76	61	45	18	54	33	27	31
14	46	54	32	43	26	38	28	50	42	28	42	35	17	61	63	31	23
15	96	25	39	38	72	49	49	35	23	20	29	37	47	11	55	30	40
16	74	49	47	42	46	58	32	27	35	52	56	38	13	65	58	39	50
17	52	61	40	34	43	54	12	39	38	65	22	36	40	24	22	47	42
18	61	67	43	39	57	52	39	34	44	11	43	28	45	59	69	42	25
19	65	21	25	42	4	48	22	43	39	33	33	33	38	30	24	42	47
20	41	35	47	32	7	63	50	45	43	24	30	30	15	98	70	22	28
21	24	23	27	40	87	32	29	42	40	70	50	26	33	72	46	43	38
22	35	70	30	49	35	24	25	24	30	30	40	32	37	96	31	45	21
23	48	31	36	48	0	60	19	21	30	15	60	47	16	35	36	29	20
24	39	33	24	27	39	41	27	46	42	7	69	27	34	63	42	33	17
25	63	52	22	28	13	45	15	20	47	54	67	23	39	39	60	26	19
26	87	48	44	43	2	20	18	49	40	22	55	49	27	89	40	38	48
27	9	58	42	31	80	30	10	41	47	37	54	29	48	0	43	23	46
28	50	62	23	23	83	56	11	37	27	4	28	50	47	87	54	34	27
29	26	45	30	23	96	23	36	33	24	78	31	34	33	80	48	25	40
30	13	44	50	47	65	31	46	28	41	67	44	21	32	70	35	21	34
31	100	36	21	37	67	27	40	32	43	26	23	25	12	48	21	23	30
32	91	40	45	50	63	53	37	45	46	83	58	47	50	20	38	40	20
33	11	34	43	46	20	34	35	25	28	48	20	27	31	22	20	41	13
34	80	53	34	45	85	46	42	26	32	9	59	31	23	28	53	38	49
35	30	42	38	21	70	29	45	29	49	91	47	34	21	50	32	49	35
36	67	59	32	49	76	36	21	27	33	93	63	48	35	100	34	20	33
37	57	47	42	28	48	35	27	36	36	74	32	49	14	15	47	28	32
38	70	63	35	22	22	62	13	42	26	98	57	22	13	7	27	27	37
39	98	32	25	30	41	22	31	49	20	87	41	42	20	78	56	45	41
40	22	41	48	26	98	33	16	43	36	13	53	28	11	85	49	50	14
41	0	37	40	24	24	67	48	30	28	41	66	39	49	74	28	28	44
42	43	22	28	25	28	50	41	23	29	61	65	40	29	26	50	48	47
43	15	30	33	47	78	68	13	32	31	85	21	45	19	9	67	36	18
44	89	65	41	40	91	55	24	23	23	59	49	21	24	57	62	32	36
45	76	43	45	36	11	47	23	40	25	100	52	20	40	91	45	36	22
46	33	28	31	44	61	69	17	38	34	96	46	24	27	67	68	46	26
47	7	24	46	34	93	59	14	30	27	2	45	42	28	83	23	40	39

F1	D_Banf
F2	D_Bmin
F3	D_Lkapa
F4	D_Los
F5	H_Banf
F6	H_Bmin
F7	H_Pr
F8	H_Lkapa
F9	H_Los
F10	L_A_Banf
F11	L_A_Bmin
F12	L_A_Lkapa
F13	L_A_Zugang
F14	L_B_Banf
F15	L_B_Bmin
F16	L_B_Lkapa
F17	L_B_Zugang

Abb. E.2: Latin-Hypercube Design [k=17/n=47]