technische universität
dortmund

Fachgebiet IT in Produktion und Logistik

Masterarbeit

Implementierung einer auf Graph Mining basierenden Open-Source-Applikation zur Risikoidentifizierung von Supply Chains

Felix Steinfurth

Matrikelnummer: 177508
Studiengang: Maschinenbau M.Sc.
Fakultät Maschinenbau

Erstgutachter:
Dr.-Ing. Dipl.-Inform. Anne Antonia Scheidler
Zweitgutacher:
M. Sc. Joachim Hunker

Eingereicht am:
25. Oktober 2021

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AUC	Area Under the Curve
DBS	Datenbanksystem
ERM	Entity-Relationship-Modell
ERD	Entity-Relationship-Diagramm
KDD	Knowledge Discovery in Databases
LBC	Link-Based Object Classification
LBR	Link-Based Object Ranking
MDS	Multidimensional Scaling
MSAGL	Microsoft Automatic Graph Layout
OEM	Original Equipment Manufacturer
OOA	Objektorientierte Analyse
OSS	Open-Source-Software
ROC	Receiver Operating Characteristic
SC	Supply Chain
SCM	Supply Chain Management
SCRM	Supply-Chain-Risikomanagement
SN	Supply Network
SNA	Social Network Analysis
SNLP	Supply Network Link Predictor

UML Unified Modeling Language

URL Uniform Resource Locator

WPF Windows Presentation Foundation

WWW World Wide Web

Kapitel 1

Einleitung

Betriebsstörungen und Unterbrechungen in heutigen SCs können sich fatal auf Unternehmen auswirken. Lieferengpässe, Umsatzeinbußen und Verluste im Marktanteil können die Folgen einer Störung bei einem Knoten des Logistiknetzwerks sein [GMR12, S. 24]. Nach einer Studie des Fraunhofer IPA sehen sich mehr als ein Drittel der befragten Unternehmen stark durch unvorhergesehene Ereignisse im Beschaffungsbereich gefährdet [SMH10, S. 31]. Daher liegt es im Interesse der Unternehmen, die Risiken von SC-Unterbrechungen zu identifizieren und zu minimieren. Durch die zunehmende Komplexität der SCs aufgrund von logistischen Trends und Entwicklungen finden diese Risiken eine größer werdende Beachtung durch Unternehmen [GMR12, S. 1]. Gleichzeitig wird es schwieriger, SC-Risiken zu bewältigen [Kho17, S. 1]. Zur Identifizierung etwaiger SC-Risiken finden sich in der Literatur eine Reihe von analytischen oder kreativen Verfahren und Hilfsmitteln, die beispielsweise auf Brainstorming, Expertenbefragungen oder Checklisten basieren [Wer17, S. 219-220]. Forschungen und Entwicklungen im Bereich des Graph Mining bieten weitere Ansätze, um Gefahrenquellen und Störpotenziale in komplexen Supply Chains zu identifizieren. Dabei liegt der Fokus auf der Graphen- bzw. Netzwerkstruktur der SC, insbesondere in den Beziehungen zwischen den Knoten dieses Netzwerks. In vielen Fällen sind die Ansätze jedoch nur im Rahmen einer Forschungsarbeit entwickelt und zur Evaluierung implementiert. Für eine Anwendung in der Praxis können ein hoher Aufwand und umfassende Kenntnisse der Programmierung, der Data Science und der Logistik nötig sein, um die entwickelten Verfahren umzusetzen und anzuwenden. Dies stellt für Unternehmen eine Hürde dar, diese Verfahren zu nutzen. So ist ein strategischer Umgang mit Risiken in Unternehmen nur rudimentär implementiert. Bei einer Befragung von Unternehmen hatten ein Viertel der befragten Unternehmen keine Software-unterstütze Risikoidentifizierung, bei über einem Drittel kamen selbst erstellte Tools z.B. auf Basis

von MS Excel zum Einsatz [HS11, S. 304]. Eine auf Graph Mining basierende Open-Source-Applikation zur Risikoidentifizierung von Supply Chains kann den Aufwand einer Eigenentwicklung reduzieren und dabei die Hürde für Unternehmen senken, Graph-Mining-Verfahren im Kontext des Supply-Chain-Risiko-Managements anzuwenden.

Ziel dieser Arbeit ist die Implementierung einer Open-Source-Applikation zur Identifizierung von Risiko-Objekten in Supply Chains, basierend auf Verfahren des Graph Mining. Dazu sollen als Teilziel Anforderungen an die Anwendung in Hinsicht auf die Risikoidentifizierung sowie auf sonstige Aspekte aufgestellt werden. Als weiteres Teilziel ist eine Softwarearchitektur zu planen, die eine Benutzeroberfläche, eine Datenbankanbindung und die Anwendung von Graph-Mining-Verfahren umfasst und eine Erweiterbarkeit für weitere Verfahren bietet. Ein entwickelter Prototyp soll das Endergebnis dieser Arbeit darstellen, der die aufgestellten Anforderungen erfüllt.

Die Arbeit beginnt mit einer Einführung in die technischen Grundlagen des Graph Minings und deren Einordnung in aktuelle Forschungsbereiche. Die Basis des Graph Minings stellt der Graph dar - der Datenstruktur, auf der alle Graph-Mining-Verfahren operieren. Dazu werden in Abschnitt 2.1 Definitionen und Konstrukte der Graphentheorie erläutert. Eine Überleitung zum Graph Mining bietet der darauffolgende Abschnitt 2.2, in dem das Forschungsgebiet des Data Mining veranschaulicht wird. Anschließend wird das Thema Graph Mining als spezieller Fall des Data Mining in Abschnitt 2.3 erklärt. Die in Kapitel 2 ausgeführten Definitionen und Konzepte sind grundlegend für ein Verständnis der im späteren Verlauf der Arbeit vorgestellten graphenbasierten Methoden der Risikoidentifizierung in SCs. Hierzu werden zunächst in Abschnitt 3.1 Grundlagen über SCs und Supply Chain Management (SCM) vermittelt. Darauf aufbauend findet in Abschnitt 3.2 eine Einführung in das Supply-Chain-Risikomanagement (SCRM) und eine Einordnung der Risikoidentifizierung statt. In Abschnitt 3.3 werden Methoden der Risikoidentifizierung angeführt. Neben in der Praxis etablierten Verfahren werden hier graphenbasierte Methoden vorgestellt, die im darauf folgenden Kapitel Anwendung finden. Das Kapitel 4 stellt den inhaltlichen Schwerpunkt dieser Arbeit dar. Die in den ersten beiden Kapiteln vorgestellten Konzepte werden in Abschnitt 4.1 genutzt, um Anforderungen an die Applikation aufzustellen. Dabei fließen auch weitere Aspekte neben Methoden der Risikoidentifizierung und Verfahren des Graph Minings ein, die Anforderungen an die Applikation definieren. In Abschnitt 4.2 wird die Applikation anhand der aufgestellten Anforderungen konzeptioniert und weiterhin in 4.3 eine geeignete Architektur geplant. Die Architektur umfasst Komponenten der Datenhaltung bzw. Datenbankanbindung, Anwendung von Graph-Mining-Verfahren und zur Interaktion und Visualisierung eine Benutzeroberfläche.

Die prototypische Umsetzung wird in Abschnitt 5.1 präsentiert und erläutert. Im folgenden Kapitel findet eine Evaluierung des Prototypen anhand der aufgestellten Anforderungen aus Kapitel 4.1 und mittels ausgewählter Datensätze statt. Die Arbeit schließt mit einer Zusammenfassung und einem Ausblick auf weitere Erweiterbarkeit und Anwendungspotenziale in Kapitel 6 ab.

Kapitel 2

Technische Grundlagen und Einordnung des Graph Minings

Um ein umfassendes Verständnis über Graph Mining zu erhalten, müssen zunächst die Grundlagen erläutert, sowie Begrifflichkeiten definiert und Themengebiete voneinander abgegrenzt werden. Die grundlegende Struktur, auf die das Graph Mining aufbaut und die in den folgenden Kapiteln thematisiert wird, ist der Graph. In Abschnitt 2.1 werden im Kontext der Graphentheorie Begriffe eingeführt und definiert, sowie Eigenschaften erläutert. Der Abschnitt 2.2 bietet eine Einführung in das Data Mining im Allgemeinen, während der Abschnitt 2.3 das Graph Mining als speziellen Fall des Data Minings behandelt und dabei Graphentheorie und Data Mining thematisch vereint.

2.1 Graphentheorie

Ein Graph $G = (V, E)$ besteht aus einer Menge von Kanten E und einer Menge von Knoten V. Für den Ausdruck $G = (V, E)$ existieren auch weitere Schreibweisen wie $G = (V(G), E(G))$. Zwei Endknoten $v, w \in V$ definieren eine Kante $e = (v, w) \in E$ [KVR08]. Bei Graphen wird zwischen *gerichteten* und *ungerichteten Graphen* unterschieden. Im Gegensatz zu ungerichteten Graphen hat eine Kante eines gerichteten Graphen einen Start- und einen Endknoten [KVR08]. Ein Graph lässt sich darstellen, indem Knoten als Punkte und Kanten als gerade Linien oder Kurven zwischen den Knoten gezeichnet werden [Aig15, S. 5]. Für die Darstellung existiert kein einheitlich festgelegtes Regelwerk, z.B. dürfen Kanten disjunkt abgebildet werden oder sich schneiden. Über die Darstellung von Graphen entscheidet viel mehr die Zweckmäßigkeit und Übersichtlichkeit. Gleichwohl kann eine übersichtliche

Darstellung von großen und komplexen Graphen ein schwieriges Unterfangen werden, sodass die Visualisierung von Graphen ein eigenes Forschungsthema bildet [van12]. Ein Beispiel einer Darstellung findet sich in Abbildung 2.1. Bei gerichteten Graphen können Kanten als Pfeile vom Start- zum Endknoten dargestellt werden.

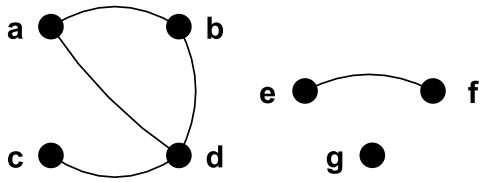

Abbildung 2.1: Ein Graph mit der Knotenmenge $V = \{a, b, c, d, e, f, g\}$ und der Kantenmenge $E = \{(a, b), (a, d), (b, d), (c, d), (e, f)\}$

Die Anzahl der Knoten $|V|$ wird auch Ordnung eines Graphen genannt. Ist v ein Endknoten von einer Kante e, so sind e und ein Knoten v *inzident*. Zwei Knoten v und w heißen *adjazent* oder *benachbart*, wenn eine Kante $e = (v, w)$ existiert, die die beiden Knoten verbindet. Eine *Clique* sind eine Menge von Knoten, die paarweise benachbart sind. Zwei verschiedene Kanten e und f werden ebenfalls benachbart genannt, falls sie sich einen Knoten teilen. Ein Graph heißt *vollständig*, wenn alle Knoten verbunden sind.[Die17]

Weiterhin können Kanten *gewichtet* bzw. Gewichte zugeordnet werden. Besitzt ein Graph gewichtete Kanten, nennt man diesen einen gewichteten Graphen [RKF12].

Der Graph $H = (V(H), E(H))$ ist ein *Teilgraph* eines Graphen $G = (V(G), E(G))$, wenn $V(H) \subset V(G)$ und $E(H) \subset E(G)$ (siehe Abbildung 2.2). Zwei strukturell gleiche Graphen G und H heißen *isomorph* (siehe Abbildung 2.3). Isomorphe Graphen werden in der Regel nicht voneinander unterschieden.[KVR08]

Ein *Weg* ist ein nicht leerer Graph $P = (V, E)$ mit $V = \{x_0, x_1, ..., x_k\}$ und $E = \{(x_0, x_1), (x_1, x_2)..., (x_{k-1}, x_k)\}$. Dabei gilt, dass x_i paarweise verschieden sind. Die Knoten x_0 und x_k werden Endknoten des Wegs genannt. Die *Länge* des Wegs ist die Anzahl der Kanten $|V(P)|$ (siehe Abbildung 2.4).[Die17] Ein *zusammenhängender* Graph ist ein Graph, in dem Wege von jedem Knoten zu jedem anderen Knoten existieren. Ein *Kreis* ist ein Weg, dessen Endknoten gleich sind. Ein ungerichteter Graph, der keine Kreise aufweist, also kreisfrei ist, wird *Wald* genannt. Ein *Baum* ist ein zusammenhängender Wald, also ein zusammenhängender, kreisfreier Graph.

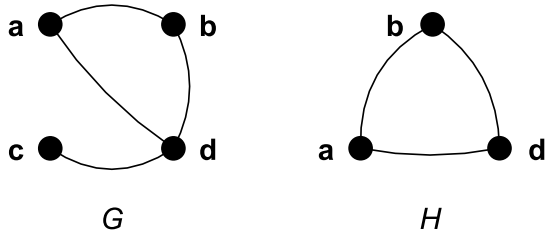

Abbildung 2.2: Zwei Graphen G (links) und H (rechts). Der Graph H ist ein Teilgraph von G.

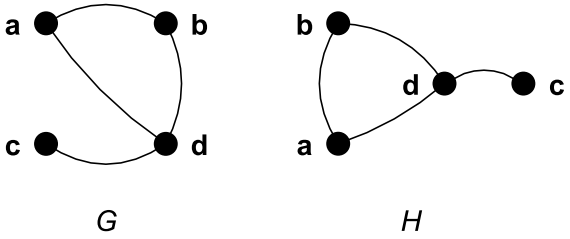

Abbildung 2.3: Zwei Graphen G (links) und H (rechts). G und H sind isomorph.

2.2 Data Mining

Unter Data Mining wird die Lehre vom Sammeln, Bereinigen, Verarbeiten, Analysieren und Gewinnen von nützlichen Erkenntnissen aus Daten verstanden [Agg15, S. 1]. Weitere Definitionen heben den Prozess- oder Verfahrenscharakter von Data Mining hervor. Kantardzic definiert Data Mining als Prozess der Entdeckung verschiedener Modelle, Zusammenfassungen und abgeleiteter Werte aus einer gegebenen Datensammlung [Kan20, S. 6]. Nach Fayyad et al. ist Data Mining die Anwendung von bestimmten Algorithmen mit dem Ziel nützliche Muster in Daten zu finden [FPS96, S. 39]. Etabliert wurde der Begriff bereits in den 1980er-Jahren [Lov83]. Das Problem, dem sich das Data Mining widmet, entspringt den wachsenden Datenmengen, die zu groß für manuelle Analysen und Interpretationen sind [Kan20, S. 12]. Der dramatische Zuwachs an Datenvolumen durch die steigende Anzahl an Datensätzen in Datenbanken bei gleichzeitigem Anstieg an Attributen pro Datensatz stellt für eine rein menschliche Analyse eine unmögliche Aufgabe dar [FPS96, S. 38]. Die Anwendung von rechnergestützten Verfahren mittels Data Mining verspricht dabei Abhilfe und ermöglicht das Entdecken von Wissen in großen Datenmengen.

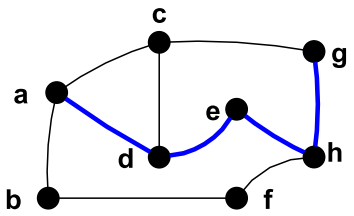

Abbildung 2.4: Ein Weg der Länge 4 (blau) in einem Graphen mit den Endknoten a und g

Bei der Wissensentdeckung stellt Data Mining allerdings ausschließlich einen Kernprozess des sogenannten Knowledge Discovery in Databases (KDD)-Prozesses dar [Sha13, S. 51]. Fayyad et al. beschreiben das Forschungsfeld der KDD als Prozess der Wissensentdeckung von Daten [FPS96, S. 39]. Der Begriff hebt das Wissen als Endprodukt der datengetriebenen Entdeckung hervor. KDD zielt darauf ab, umfangreiche Rohdaten in kompakter (z.B. als Bericht), abstrakter (z.B. als approximatives Modell eines Prozesses) oder nützlicher Form (z.B. als Vorhersagemodell) abzubilden [FPS96, S. 37]. Der KDD-Prozess besteht aus den Phasen Selektion, Vorverarbeitung, Transformation, Data Mining, Interpretation und Evaluierung [Run15, S. 3]. In der Selektionsphase wird ein Zieldatenset zusammengestellt, auf der die Entdeckung durchgeführt werden soll. Es folgen die Vorverarbeitung, bei der die Daten bereinigt werden, und die Transformationsphase, in der die Daten auf die nützlichen Eigenschaften reduziert und projiziert werden. In der darauffolgenden Data-Mining-Phase wird eine ausgewählte Data-Mining-Methode durchgeführt, die Muster finden soll. Zuletzt findet die Interpretation und Evaluierung statt, in der Phase werden entdeckte Muster interpretiert und überlegt, welche Muster als neues Wissen betrachtet werden können [FPS96, S. 42]. Abbildung 2.5 zeigt eine Übersicht über die grundlegenden Phasen und deren Zwischenergebnisse.

In der Praxis werden sowohl bei KDD als auch bei Data Mining zwischen zwei primären Zielen unterschieden: Vorhersage und Beschreibung. Bei der Vorhersage dient die Mustererkennung der Bestimmung von zukünftigen Verhalten oder zukünftiger Werte einer Variable. Die Beschreibung hingegen nutzt die Entdeckung von unbekannten, nicht trivialen Mustern, um die zugrundeliegenden Daten zu beschreiben und diese in ein Format zu überführen, das für Menschen verständlich und interpretierbar ist. [Kan20, S. 2-3]

Die Unterscheidung von Data-Mining-Methoden zur Vorhersage und zur Beschreibung bestimmt ebenfalls die Art, wie Daten im Data-Mining-Prozess analysiert werden. Data Mining bedient sich dabei Methoden des Machine

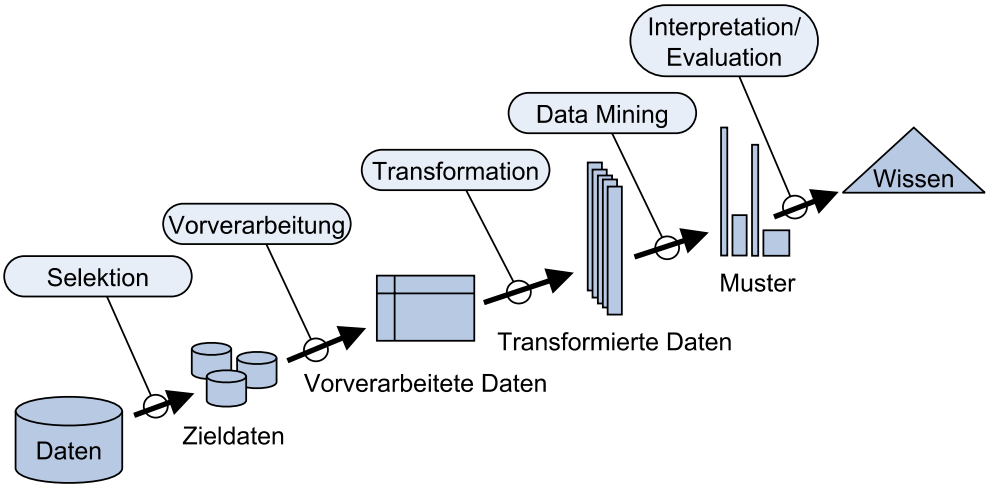

Abbildung 2.5: Übersicht über Phasen des KDD-Prozesses nach [FPS96, S. 41]

Learning, die in der Lage sind, von (historischen) Daten zu lernen [Sch+19, S. 876]. Sogenanntes überwachtes Lernen ist ein Prozess, bei der eine Funktion unter Verwendung eines Trainingsdatensets bestimmt wird. Das Trainingsdatenset kann dabei als „Erfahrungsschatz der Vergangenheit“ des Modells angesehen werden [Gor11, S. 9]. Die in dem Lernverfahren bestimmte Funktion bildet die Eigenschaften der einzelnen Datensätze auf eine Zielgröße (auch Label genannt) ab und kann bei Eingabe eines (bisher unbekannten) Datensatzes mit fehlendem Label dieses vorhersagen [Gor11, S. 9]. Im Gegensatz zu überwachten Lernverfahren dienen unüberwachte Lernverfahren der Analyse und Beschreibung von Daten ohne Label, um Muster in diesen a priori (ohne Erfahrungsschatz) zu finden. Die Zielgröße wird durch die Zielsetzung bestimmt [Gor11, S. 9].

Im Rahmen des KDD ist eine Wissensentdeckung immer mit einer Zielsetzung und einer davon abgeleiteten Aufgabenstellung verbunden [BC06, S. 264]. Neben der Einteilung der Ziele in vorhersagend und beschreibend, existieren in der Literatur eine Vielzahl an Einteilungen von Aufgabenstellungen. Häufig wird zwischen Klassifikation, Regression, Clusteranalyse und Assoziationsanalyse unterschieden [BC06, S. 264]. Im Folgenden werden diese kurz erläutert:

Klassifikation Klassifikation ist ein Data-Mining-Problem, bei dem Datensätze anhand gemeinsamer Eigenschaften in Klassen gruppiert werden bzw. jeweils ein Klassenlabel zugewiesen wird [Dru11]. Innerhalb eines

überwachten Lernverfahrens werden die mit einem Klassenlabel assoziierten Datensätze genutzt, um ein Trainingsmodell zu entwickeln, das anschließend zur Vorhersage eines Klassenlabels von weiteren Datensätzen eingesetzt werden kann [Agg15, S. 18].

Regression Bei einem Regressionsproblem oder Regressionsanalyse ist das Ziel, eine reellwertige Funktion abzuleiten, deren Werte dem Mittelwert einer gegebenen (Ergebnis-)Variable in Abhängigkeit von einer oder mehreren unabhängigen Variablen entsprechen [QB11, S. 838]. Damit lassen sich ein quantitativer Zusammenhang zwischen mehrerer Variablen bestimmen und Werte der Ergebnisvariable anhand von Werten der anderen Variablen vorhersagen [Gor11, S. 26]. Die Regression ähnelt der Klassifikation, unterscheidet sich aber vor allem durch die numerische Zielgröße im Gegensatz zu der kategorialen Zielgröße bei der Klassifikation [Hud20, S. 95].

Clusteranalyse Die Clusteranalyse ist eine Art von unüberwachten Lernens mit dem Ziel, ein Datenset in Gruppen, sogenannte Cluster, zu partitionieren [Sam11]. Die Clusterzugehörigkeiten werden anhand von geeigneten Ähnlichkeitsmaßen zwischen Datensätzen als auch zwischen Clustern bestimmt [BC06, S. 274].

Assoziationsanalyse Mittels der Assoziationsanalyse lassen sich Zusammenhänge und Abhängigkeiten in einer Datenbasis entdecken [BV08, S.261]. Aus einer Datenbasis, in der jeder Datensatz eine Menge an Items darstellt, können sogenannte Assoziationsregeln abgeleitet werden. Diese haben die Form $X \leftarrow Y$, wobei X und Y Itemmengen sind, zwischen denen ein Zusammenhang definiert wird [Toi11, S. 49-50].

2.3 Graph Mining

Data Mining ist mit einer großen Vielfalt von Datenarten möglich. Grundsätzlich lassen sich jedoch zwei Arten von Daten für den Data-Mining-Prozess unterscheiden [Agg15, S. 6]: Nicht-abhängigkeitsorienterte Daten sind am häufigsten anzutreffen und bezeichnen einfache Datenarten wie multidimensionale Daten oder Textdaten, die keine festgelegten Abhängigkeiten zwischen Datensätzen aufweisen. Ein Beispiel hierfür ist ein demographisches Datenset über Alter und Wohnort von Personen. Bei abhängigkeitsorientierten Daten hingegen bestehen implizite oder explizite Beziehungen zwischen den einzelnen Datenobjekten, z.B. bei einem Datenset eines sozialen Netzwerks. Die Komplexität, die durch die gegebenen Beziehungen zwischen

den Datenobjekten geschaffen wird, stellt für einen Data-Mining-Prozess eine große Herausforderung dar, bei der die Abhängigkeiten in den Prozess einbezogen werden müssen [Agg15, S. 6]. Die übliche Annahme von unabhängigen und identisch verteilten Daten (siehe [Kle08]), die die Basis für viele statistische Machine-Learning-Algorithmen bildet, wird durch diese multirelationalen Datensätze mit vielfachen Abhängigkeiten zwischen den Datenobjekten verletzt [Kan20, S. 392]. So korrelieren z.B. die Attributswerte von verknüpften Objekten in vielen Fällen. Um diese Abhängigkeiten zwischen Attributen und Beziehungsstruktur auszuschöpfen, bietet die Graphenstruktur einen Ansatz zur Abbildung der komplexen Beziehungen und der daraus resultierenden Informationen [Kan20, S. 392].

Analog zur Definition von Data Mining gilt Graph Mining als Extraktion von neuem und nützlichem Wissen aus einer graphischen Darstellung von Daten [CH07, S. 2]. Getoor und Diehl benutzen den Begriff Link Mining, um die Verknüpfungen (Links) als Informationsquelle hervorzuheben und das Link Mining vom Data Mining mit nicht-abhängigkeitsorientierten Daten abzugrenzen [GD05, S. 3]. Die Autoren definieren Link Mining als Data-Mining-Techniken, die explizit Verknüpfungen der in Beziehung stehenden Daten beim Erzeugen von vorhersagenden oder beschreibenden Modellen berücksichtigen [GD05, S. 3]. Gleichwohl findet sich Link Mining auch als Begriff der Social Network Analysis im Kontext des Data Minings auf Soziale Netzwerke wieder. Dabei stellt ein soziales Netzwerk ein multirelationales Datenset dar, das als Graph repräsentiert wird und aus dem durch das Link Mining Informationen gewonnen werden [HKP12, S. 556]. In dieser Arbeit wird Link Mining mit Graph Mining gleichgesetzt. Nach Chakrabarti dient Graph Mining der Analyse der Eigenschaften von Graphen, der Vorhersage, wie Struktur und Eigenschaften eines Graphen sich auf eine Anwendung auswirken und der Entwicklung von Modellen, die realistische Graphen anhand von Mustern der realen Welt generieren können [Cha11, S. 469].

Beim Graph Mining lassen sich ebenfalls wie beim Data Mining in der Literatur häufig genannte Aufgabenstellungen aufzählen. Getoor und Diehl unterscheiden diese zunächst in ihrem Fokus auf Objekte, Verknüpfungen und Graphen [GD05, S. 4]. Basierend auf Publikationen aus dem Forschungsfeld um Graph und Link Mining, erstellen die Autoren eine Taxonomie der häufig genannten Aufgabenstellungen des Graph Minings. Im Folgenden werden die in der Taxonomie eingeordneten Aufgabenstellungen erläutert.

Objektbezogene Aufgabenstellungen

Link-Based Object Ranking: Link-Based Object Ranking (LBR) wertet die Kantenstruktur eines Graphen aus, um Objekte (Knoten) in diesem Graphen zu ordnen bzw. ihnen eine Priorität oder Wichtig-

keit zuzuordnen [GD05, S. 4]. Bekannte Algorithmen wie PageRank und HITS finden im Ranking von Webseiten Anwendung [HKP12, S. 567]. Die Datenquelle hierfür ist das World Wide Web (WWW), das als Graph strukturiert ist, in dem Webseiten Knoten und Hyperlinks in Webseiten, die auf andere Webseiten verweisen, Kanten repräsentieren [AW10, S. 45]. Die Kernidee des PageRank-Algorithmus beruht auf dem Zufallssurfer-Modell [Agg15, S. 598]. Ein Internetnutzer (Surfer) besucht Webseiten, in dem er zufällige Hyperlinks auf den Webseiten auswählt. Die Wahrscheinlichkeit, mit der der Surfer eine bestimmte Seite besucht, ist mit der Anzahl an Hyperlinks verbunden, die auf diese Seite verweisen. Ebenso ist die Wahrscheinlichkeit hoch, wenn eine häufig besuchte Seite auf diese Seite verweist. Damit modelliert der PageRank-Algorithmus die Reputation oder Wichtigkeit der Webseiten [Agg15, S. 598]. Weitere Algorithmen finden in der Social Network Analysis Anwendung, in der eine Zentralität ein Maß der Wichtigkeit von Knoten ableitet, anhand derer ein Ranking möglich ist [GD05, S. 5]. Die Analyse der Zentralität in Graphen kann ebenfalls dazu verwendet werden, um kritische Ausfallpunkte in einem Netzwerk zu bestimmten [AW10, S. 51]. Es existieren eine Reihe an Zentralitätsmetriken, die unterschiedliche Eigenschaften messen, z.B. Degree Centrality (Grad), Closeness Centrality (Nähe), Betweenness Centrality, Bonacich (Eigenvektor) Centrality, Eccentricity Centrality [Ste+15, S. 300].

Link-Based Object Classification: Link-Based Object Classification (LBC), auch Label Propagation genannt, dient der Klassifikation von Knoten in einem Graphen [GD05, S. 5]. Anhand einer Menge an Knoten, die ein Klassenlabel aufweisen, wird ein Modell trainiert, das sich zur Vorhersage des Klassenlabels von Knoten ohne Label nutzen lässt [AW10, S. 37]. Der Unterschied zur Klassifikation von nichtabhängigkeitsorientierten Daten im Data Mining besteht in der Annahme von unabhängigen und identisch verteilten Daten, die beim LBC nicht gilt [GD05, S. 5]. Möglich ist eine häufig auftretende Korrelation der Labels von miteinander verknüpften Daten, die berücksichtigt werden muss. So erzielt eine Klassifizierung von Hypertext ein besseres Ergebnis, wenn die Klassenlabels von verknüpften Dokumenten mit in die Klassifizierung einfließt, als eine Klassifizierung anhand der Eigenschaften der verknüpften Dokumente ohne Berücksichtigung deren Labels [CDI98, S. 5]. Rehman et al. nennen D-Walks und ein auf k-NN basierenden Algorithmus als LBC-Verfahren.

Object Clustering (Group Detection): Das Ziel von Object Clustering bzw. Group Detection liegt darin, Knoten eines Graphen

basierend auf gemeinsame Eigenschaften zu gruppieren [GD05, S. 5]. Zur Identifikation von zusammengehörigen Gruppen dient ein definiertes Ähnlichkeitsmaß. Bei einem gewichteten Graphen kann das Gewicht bzw. die Distanz als Ähnlichkeitsmaß herangezogen werden [AW10, S. 32]. Beispielsweise lässt sich mit dem Minimum-Cut der Graph in zwei Cluster einteilen, indem ein Schnitt auf den Kanten mit der minimalen Distanz erfolgt. Dieses Verfahren erzeugt jedoch häufig unbalancierte Partitionen [Wan+07, S. 497]. Weitere Object-Clustering-Verfahren sind der Kernighan-Lin-Algorithmus [AW10, S. 281], k-Means [Gal+12], der Girvan-Newman-Algorithmus [AW10, S. 284] und Spectral Clustering [AW10, S. 287].

Object Identification (Entity Resolution): Die letzte Objektbezogene Aufgabenstellung ist Object Identification bzw. Entity Resolution, das eine Identifizierung von Datenobjekten zum Ziel hat. Dabei werden Referenzen in den Daten zugehörige Entitäten in der realen Welt zugeordnet [GD05, S. 6]. Zum Beispiel finden sich in einem Datenset Datensätze mit den Namen Max M., M. Mustermann und Max Mustermann, die sich auf die selbe Person beziehen können. Dem Ansatz folgend, dass die Referenzen der selben Entität häufig miteinander in Beziehung stehen oder zusammen gruppiert sind, lässt sich daraus ein Graph mit Referenzen als Knoten und Beziehungen zwischen Referenzen als Kanten erzeugen [CH07, S. 311]. Um die Entitäten z.B. für eine Deduplikation aufzulösen, werden nicht nur die Knotenattribute sondern auch Grapheigenschaften als Ähnlichkeitsmaß herangezogen [CH07, S. 312].

Verknüpfungsbezogene Aufgabenstellungen

Link Prediction: Link Prediction bezeichnet die Vorhersage der Existenz einer Kante zwischen zwei Knoten. Dabei nutzt das Verfahren zur Vorhersage ähnliche Knotenattribute und Kantenstrukturen [GD05, S. 6]. Ein Beispiel einer Anwendung von Link Prediction findet sich in sozialen Netzen, bei denen Nutzerprofile als mögliche Freunde vorgeschlagen werden [Agg15, S. 650]. Es existieren unterschiedliche Ähnlichkeitsmaße von Knoten, um Kanten zwischen diesen vorherzusagen. Grundsätzlich wird zwischen strukturbedingten Maßzahlen, bei denen z.B. gemeinsame Nachbarknoten die Wahrscheinlichkeit für eine Existenz einer Kante erhöhen, und inhaltsbezogene Maßzahlen, wobei ähnliche Inhalte von Knoten auf eine vorherzusagende Kante schließen lässt, unterschieden [Agg15, S. 650]. Ähnlichkeitsmaße können auf gemeinsamen direkten Nachbarknoten basieren (Common Neighbor, Ja-

card, Adamic-Adar), auf indirekten Verbindungen (Katz) oder auf einem Random-Walk-Modell [Agg15, S. 650-653]. Ebenfalls kann Link Prediction als binäres Klassifikationsproblem behandelt werden, indem für ein Knotenpaar eine 1 für die Existenz einer Kante oder eine 0 für die Nicht-Existenz einer Kante vorhergesagt wird [GD05, S. 6].

Graphbezogene Aufgabenstellungen

Subgraph Discovery: Subgraph Discovery, oder auch Frequent Subgraph Mining genannt, beschäftigt sich mit dem Entdecken von häufig auftretenden bzw. interessanten Teilgraphen [GD05, S.7]. Das Vorgehen besteht in der Regel aus der Generierung von Teilgraph-/Substruktur-Kandidaten und der anschließenden Überprüfung der Häufigkeiten [CH07, S. 100]. Damit ähnelt das Vorgehen dem Apriori-basierten Frequent Itemset Mining bei nicht-abhängigkeitsorientierten Daten. Bei der Überprüfung der Häufigkeiten von Kandidaten muss auf Isomorphie (siehe Abschnitt 2.1) geprüft werden, sodass der Prozess deutlich komplexer als beim letztgenannten Frequent Itemset Mining ist [JCZ10, S. 77-78].

Graph Classification: Klassifizierungsprobleme auf Graphen können nicht nur auf der Ebene der Knoten, sondern ebenfalls auf ganze Graphen angewandt werden. Graph Classification versucht ganze Graphen mit einem Klassenlabel zu versehen, während die bereits beschriebene LBC-Aufgabenstellung ausschließlich Knoten klassifiziert [GD05, S. 7]. Methoden der Graph Classification können ein Klassenlabel unüberwacht anhand von Ähnlichkeitsmaßen oder überwacht durch einen mit Trainingsdaten angelernten Klassifikator bestimmen [TS10, S. 338]. Ein Anwendungsbeispiel findet sich in der Medikamentenforschung bei der Klassifizierung von Wirkstoffen anhand der chemischen Zusammensetzung [TS10, S. 337].

Generative Models for Graphs: Generative Models for Graphs beschäftigt sich mit der Generierung von synthetischen Graphen anhand von Modellen, die bestimmte Eigenschaften oder Verteilungen erzeugen [GD05, S. 7]. Ein Fokus dieses Forschungsbereichs liegt in der Nutzung von Mustern und Gesetzmäßigkeiten aus Graphen der realen Welt, wie bspw. das WWW, kriminelle Netzwerke oder Virusausbreitungen [CH07, S. 65]. Dabei soll möglichst ein Graph generiert werden, der diesen Mustern entspricht [CFM10, S. 70]. Anwendung finden die Verfahren bei der Anomaliedetektion in Graphen, zur Erzeugung von realen Graphen für Simulationsstudien und Graphkompression, in denen Muster Eigenschaften der Daten beschreiben, die wiederum zur

Kompression der Daten eingesetzt werden können [CFM10, S. 70].

Anwendung findet Graph Mining in mehreren und teilweise sehr verschiedenen Domänen, wie z.B. sozialen Netzwerken, Chemie und Pharmazeutik, sowie dem WWW. Eine weitere Domäne, die in dieser Arbeit betrachtet wird, ist das SCRM und insbesondere die Risikoidentifizierung von SC als Teil desselben. Die Grundlagen dazu und die Anwendung des Graph Minings in diesem Kontext vermittelt das folgende Kapitel.

Kapitel 3

Risikoidentifizierung in Supply Chains

Auf dem Gebiet der Logistik existieren zahlreiche Bestrebungen, um datengetriebene und effektive Ansätze zur Handhabung von SC-Risiken zu entwickeln [Hau20, S.16-17]. SCs können dabei in ihrer Grundstruktur als Graphen repräsentiert werden [KK08, S. 202]. Damit ist eine SC als Betrachtungsgegenstand und die Risikoidentifizierung von SCs als Einsatzgebiet für Anwendungen von Graph-Mining-Methoden ein interessantes Forschungsgebiet.

In diesem Kapitel werden die Grundlagen über die Risikoidentifizierung von SCs vermittelt. Zunächst wird der Leser in Abschnitt 3.1 in Begriffe und Konzept von SCs eingeführt, weiterhin werden Aufgaben und Teilgebiete des SCM erläutert. Das Teilgebiet des SCRM, in welches das Thema dieser Arbeit einzuordnen ist, wird in Abschnitt 3.2 behandelt. Dieser Abschnitt vermittelt die Grundlagen von SC-Risiken und ihre Bedeutung für Unternehmen in der Logistik und Produktion. Als Teilaufgabe des SCRM werden Methoden vorgestellt, mit deren Hilfe SC-Risiken identifiziert werden können.

3.1 Supply Chains und Supply Chain Management

Eine SC kann mit Versorgungskette oder Wertschöpfungskette übersetzt werden [KK08, S. 541]. In einigen Werken findet sich auch der Begriff Lieferkette [Cer17] oder logistische Kette [Sys06] als Übersetzung der SC wieder. Im Gabler Lexikon Logistik ist SC wie folgt definiert: „Eine Wertschöpfungskette eines Produktes umfasst sämtliche Fertigungs- und Absatzstufen von der Rohstoffgewinnung über die Produktion bis hin zum Absatz an den Konsumenten“ [KK08, S. 541]. Der Begriff SC ist irreführend, da es sich bei diesem

nicht zwingend um eine Kette handelt. Vielmehr ist darunter ein Netzwerk zu verstehen, sodass auch von SN die Rede ist [GF08, S. 24]. Govil und Proth gehen in ihrer Definition von SC auf diesen Aspekt ein und nennen diese ein Netzwerk von miteinander kooperierenden Organisationen, die das Ziel hat, den Material- und Informationsfluss zwischen Zulieferern und Kunden zu den geringsten Kosten und der größten Schnelligkeit zu verbessern [GP02, S. 7]. Eine SC umfasst typischerweise die Bereitstellung von Rohstoffen durch einen Rohstofflieferanten, die Verarbeitung und Lieferung von Komponenten über mehrere Ebenen durch Lieferanten bis zum Hersteller, weiterhin die Distribution des Endprodukts an den Großhandel und weiter an den Einzelhandel, der schließlich die Endkunden versorgt. Eine SC am Beispiel eines Laptops wird in Abbildung 3.1 dargestellt. Hierbei stellt ein Rohstofflieferant Rohöl bereit, das weiterhin durch einen Lieferanten zu Kunststoffgranulat verarbeitet und an ein weiteres Unternehmen geliefert wird, welches daraus Tastaturen fertigt. Der Laptop-Hersteller bezieht die für die Herstellung von Laptops benötigten Komponenten von den Lieferanten und verteilt die hergestellten Endprodukte an den Elektronikgroßhandel, dieser wiederum beliefert ein Laptopfachgeschäft, in dem der Endkunde den Laptop erwirbt.

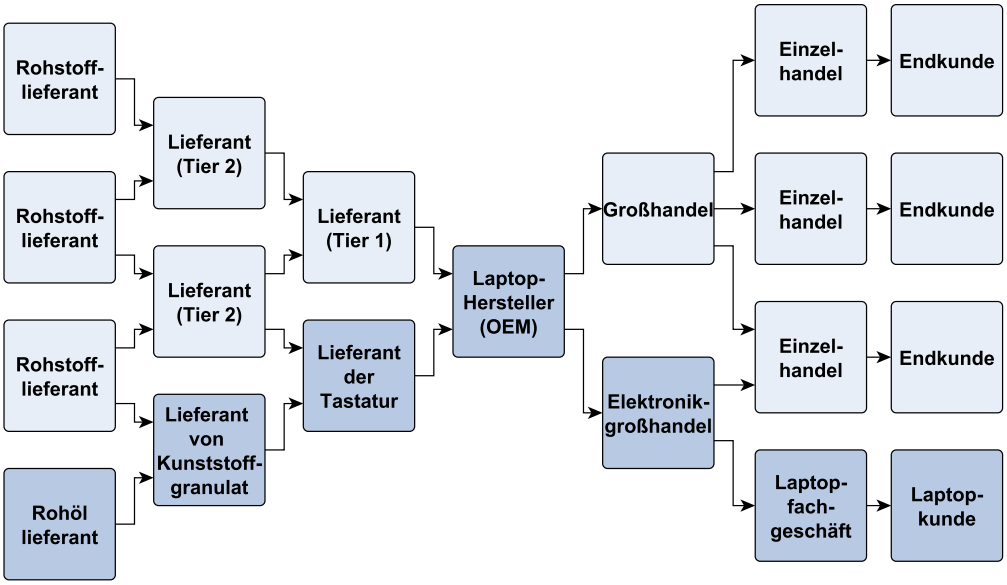

Abbildung 3.1: Eine Supply Chain bzw. ein Supply Network am Beispiel eines Laptops nach [WW11, S. 8]. Die Pfeile stellen den Materialfluss dar.

Anhand des Beispiels in 3.1 lässt sich zudem erkennen, dass der Herstellungsprozess über mehrere Stufen erfolgt, jede Stufe wird hier als vertikal zusammengehörige Gruppe dargestellt. Eine solche Stufe wird auch Tier

genannt, der Hersteller des Endprodukts Original Equipment Manufacturer (OEM) [KK08, S. 179]. So beliefert der Tier-1-Lieferant oder auch First Tier den OEM mit Modulen, der Tier-2- den Tier-1-Lieferanten mit Komponenten für die Module.

Kim et al. beschreiben die Abbildung einer SC bzw. eines SN als graphisches Konstrukt mit Unternehmen als Knoten auf zwei Arten [Kim+11, S. 196]: Zwischen zwei Unternehmen kann eine Kante einen Materialfluss oder eine vertragliche Beziehung repräsentieren. Materialflüsse werden durch einen gerichteten Graphen modelliert, wobei dieser beschreibt, welcher Zulieferer welchen Kunden beliefert. Sofern nicht der ganze Lebenszyklus von Produkten bzw. Materialrückläufer o. ä. abgebildet werden, ist der Graph ein Baum. Ein Graph, basierend auf vertragliche Beziehungen, ist hingegen ungerichtet. Dieser kann sich von einem auf Materialflüssen basierenden Graphen unterscheiden, wenn z.B. ein OEM einen Vertrag mit einem Tier-2-Lieferanten abschließt, welcher den Tier-1-Lieferanten beliefert [Kim+11, S. 196-197]. Bei diesem Beispiel existiert zwischen dem Tier-2- und dem Tier-1-Lieferanten ein Materialfluss, jedoch keine vertragliche Beziehung. Zwischen dem OEM und dem Tier-2-Lieferanten hingegen ist in diesem Fall eine vertragliche Beziehung vorhanden, aber kein direkter Materialfluss. Ein anderes Beispiel sind Kooperationen zwischen Unternehmen, die zueinander kompatible Produkte herstellen ohne einander zu beliefern [Bri+18, S. 4]. Kanten von SC modellierenden Graphen können neben Materialflüssen und vertraglichen Beziehungen auch weitere Beziehungen wie Informationsflüsse darstellen [BB13, S. 240].

Im Kontext der Unternehmenskooperationen in einer SC kann das SCM als Koordination der beteiligten Unternehmen und deren Prozesse sowie der unternehmensübergreifenden Prozessintegration verstanden werden [HHU11, S. 6]. Ziele des SCM sind die Reduzierung von Lieferzeiten, die Verbesserung der Kundenzufriedenheit und die Steigerung der betrieblichen und ressourcenbezogenen Effizienz [Nak20, S. 40]. Werner beschreibt das Hauptanliegen des SCM als gleichzeitige Optimierung der Unternehmenseffektivität („langfristige Erfolgswirksamkeit“) und der Unternehmenseffizienz („Erzielung günstiger Kosten-Nutzen-Relationen“) sowie eine Harmonisierung der Wettbewerbsfaktoren („Kosten, Zeit, Qualität und Flexibilität“) [Wer17, S. 30].

3.2 Supply-Chain-Risikomanagement und -identifizierung

„Jede Planung ist per definitionem zukunftsgerichtet und die Zukunft ist unvermeidlich unsicher“ [SK20, S. 277]. Dieses grundlegende Dilemma jeder Planung führt zu einem Begriff, das in dieser Arbeit im Kontext der SC das Thema bestimmt: Risiko. Romeike definiert Risiko im Kontext des betrieblichen Risikomanagements als „mögliche Abweichung von den geplanten Zielen“ [Rom18, S. 10]. Nach Romeike sind darunter sowohl Gefahren als negative Abweichungen als auch Chancen als positive Abweichungen zu verstehen. Im allgemeinen deutschen Sprachgebrauch bzw. aus anderen betriebswirtschaftlichen Sichtweisen wird Risiko vor allem mit der negativen Abweichung interpretiert und bezeichnet die Möglichkeit eines Eintritts von Ereignissen, die wirtschaftliche Einbußen verursachen [Koc12, S. 2]. Neben den hier aufgeführten Definitionen existieren in der Literatur unzählige weitere. Ritchie und Brindley [RB07, S. 305] führen drei Dimensionen an, die die meisten Definitionen von Risiko gemein haben:

- Eintrittswahrscheinlichkeit eines bestimmten Ereignisses oder eines Resultats
- Folgen des bestimmten eingetretenen Ereignisses oder des Resultats
- Kausaler Verlauf, der zu dem Ereignis führt

Dabei versucht das Risikomanagement alle drei Dimensionen zu behandeln, indem die Quellen analysiert und die Kräfte verstanden werden, die die Folge von Ereignissen auslösen. Ebenso zielt es darauf ab, die Leistungsfähigkeit zu steigern und gleichzeitig negative Konsequenzen zu vermeiden.
Die betriebswirtschaftliche Definition aufgreifend, beschreibt Risikomanagement den Umgang mit Risiken und Chancen, die aus den Prozessen der unternehmerischen Tätigkeit entstehen [Rom18, S. 16]. Nach Ceritoğlu ist Risikomanagement ein Prozess, bestehend aus den Phasen der Erkennung, Analyse, Bewertung, Verwaltung und Kontrolle von Risiken [Cer17, S. 277]. Ziel des Risikomanagements ist nicht nur die Vorbeugung, sondern auch die Verminderung von Effekten von Risiken [Cer17, S. 278].

SCRM wiederum kann als Schnittmenge von SCM und Risikomanagement beschrieben werden [Kho17, S. 7]. Tang definiert das SCRM durch Koordination und Kollaboration unter SC-Partnern, um Rentabilität und Kontinuität zu gewährleisten [Tan06, S. 453]. Um die Risiken zu verstehen, mit dem sich das SCRM befasst, werden diese nach Khojasteh [Kho17, S. 13]

in drei Kategorien eingeteilt: Lieferantenrisiken, Nachfragerisiken und Prozessrisiken. Lieferantenrisiken umfassen alle auf die eingehenden Güter bezogenen Risiken, die auf der Seite des Lieferanten auftreten können. Darunter fallen z.B. Lieferausfälle und -verzögerungen. Nachfragerisiken sind Risiken, die auf der Kundenseite entstehen, z.B. Schwankungen der Nachfrage. Prozessrisiken stellen Risiken dar, die während der Herstellung oder Lagerung auftreten, z.B. Maschinenausfälle. Ceritoğlu nennt noch eine vierte Kategorie: Risiken aus Umwelt und sektoralen Faktoren [Cer17, S. 285]. Darunter zählen bspw. Erdbeben, Epidemien, politische und rechtliche Veränderungen. Umweltbezogene Risiken weisen eine niedrige Eintrittswahrscheinlichkeit auf, können jedoch zu großen Verlusten führen [Cer17, S. 285].
Der SCRM-Prozess lässt sich in Anlehnung an Werner [Wer17, S. 219-224] und Ceritoğlu [Cer17, S. 287] in vier Hauptprozesse unterteilen:

- Identifizierung
- Analyse/Bewertung
- Milderung/Steuerung
- Kontrolle/Überwachung

In der sogenannten Identifikationsphase versuchen Unternehmen, alle Risiken zu identifizieren, die den Materialfluss innerhalb der SC stören können [Cer17, S. 287]. Werner definiert die Risikoidentifizierung als Erfassung von Gefahrenquellen, Störpotenziale und Schadensursachen, die eine negative Auswirkung auf SC-Ziele haben können [Wer17, S. 219]. Die Phase der Identifizierung von SC-Risiken gilt als ausschlaggebend für ein erfolgreiches Risikomanagement [NRC09, S. 154]. In der zweiten Phase des SCRM, der Risikoanalyse und -bewertung, werden die erfassten Risiken mit möglicherweise verletzbaren Bereichen im Unternehmen verknüpft und kategorisiert. Die Kategorisierung wird auch Risiko-Mapping genannt [Wer17, S. 220]. Im Rahmen der Bewertung werden die festgestellten Risiken bewertet und ihnen jeweils ein Bedeutungsgrad zugewiesen. Dies erfolgt anhand der Eintrittswahrscheinlichkeit sowie des Umfangs an Folgen, insbesondere der Schadenshöhe [Cer17, S. 288]. Die Risikomilderung oder Risikosteuerung stellt die dritte Phase des SCRM dar. Hierbei werden Möglichkeiten zur Risikoreduzierung oder -limitierung erörtert und Maßnahmen beschlossen [Wer17, S. 222]. Die letzte Phase der Risikokontrolle oder auch Risikoüberwachung besteht in der Beobachtung des Risikoverlaufs und der Bewertung der Wirksamkeit von erfolgten Maßnahmen [Cer17, S. 291]. Die vier Phasen des SCRM sind als kontinuierlicher Prozess zu verstehen, wie Abbildung 3.2 zeigt.

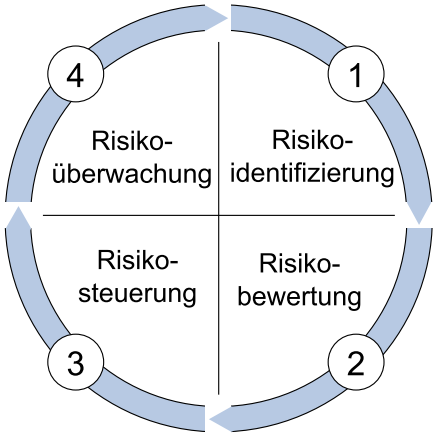

Abbildung 3.2: Die vier Phasen des Risikomanagementprozesses nach [SMH10, S. 20]

Im Folgenden wird sich auf die Risikoidentifizierung als thematischer Schwerpunkt dieser Arbeit konzentriert. Die weiteren beschriebenen Phasen des SCRM dienen der Einordnung in das SCRM und dem Verständnis über Ziele und Grenzen der Risikoidentifizierung.

3.3 Methoden der Risikoidentifizierung in Supply Chains

Hilfsmittel und planmäßige Verfahren, mit denen ein Ziel erreicht werden soll, werden Methoden genannt. Im Kontext der Risikoidentifizierung ist damit ein Erkenntnisweg gemeint [Rom18, S. 55]. In der Literatur existieren zahlreiche Methoden der Risikoidentifizierung. Eine Abgrenzung von Methoden dieser Phase von Methoden anderer Phasen des Risikomanagementprozesses ist nicht immer möglich, denn die Risikoidentifizierung ist eng mit der Risikoanalyse und der -bewertung verbunden [Rom18, S. 55]. Im Folgenden wird eine Auswahl an Methoden, die zur Risikoidentifizierung geeignet sind, aufgezählt und kurz erläutert. Zur Einordnung und für das Verständnis über die thematische Vertiefung in graphenbasierte Methoden werden ebenfalls Verfahren beschrieben, die nicht weiter Anwendung in den folgenden Kapiteln finden.

Grundsätzlich lassen sich Methoden der Risikoidentifizierung in Kollektionsmethoden und Suchmethoden unterteilen, Suchmethoden wiederum in Analytische Methoden und Kreativitätsmethoden [Rom18, S. 56]. Eine Kol-

lektionsmethode eignet sich für Risiken, die bereits bekannt sind. Eine solche Methode stellt die Checkliste dar, die mithilfe von zusammengetragenen Informationen über mögliche SC-Risiken entwickelt wird [Wer17, S. 220]. Damit ist ein Risikomanagement auf Vollständigkeit überprüfbar und kann sicherstellen, dass Risiken auch erfasst werden. Ein Nachteil dieser einfach anzuwendenden Methode ist die Auflistung nur erwarteter Risiken und der damit verbundenen Unvollständigkeit jeder Checkliste [Rom18, S. 62]. Weitere Kollektionsmethoden sind die SWOT-Analyse (Strengths, Weaknesses, Opportunities, Threads), die Risiko-Identifikations-Matrix und das (Experten-)Interview, bei denen ebenfalls Informationen gesammelt werden, um Risiken zu erfassen [Rom18, S. 56]. Kreativitätsmethoden wie z.B. Brainstorming und Mind Mapping sind kreative Denkprozesse, um zu neuartigen Erkenntnissen zu gelangen. Hierbei werden in Gruppenarbeit Ideen gesammelt, diskutiert und dokumentiert [Win04, S. 264]. Analytische Methoden hingegen basieren auf logischen Schlüssen sowie systematischer Zergliederung der Risiken und kommen zum Einsatz, wenn kein oder nur ein geringer Erfahrungsschatz über die Risiken vorhanden ist [ANR09, S. 69]. Im Folgenden wird sich auf die Kategorie der analytischen Methoden konzentriert und Methoden dieser Kategorie zur Risikoidentifizierung vorgestellt.

3.3.1 Risikoidentifizierung mittels Social Network Analysis

Eine analytische Methode zur Risikoidentifizierung ist die Social Network Analysis (SNA) [Rom18, S. 110]. Die SNA ist nach Tang und Liu die Lehre von Beziehungen zwischen Individuen, insbesondere der Analyse von sozialen Strukturen und sozialer Position, sowie der Rollenanalyse [TL10, S. 487]. Die Ausrichtung der Definition an sozialen Eigenschaften begründet sich in der ursprünglichen Anwendung in Sozialwissenschaften. Die Erkenntnisse und Konzepte der SNA fanden auch in anderen Forschungsbereichen Anklang und wurden z.B. auf Krankheitsausbreitungen [Kim+11, S. 195], kriminelle Netzwerke und SCs adaptiert [Rom18, S. 110]. Nach Romeike lässt sich die Methode im SCRM nutzen, um störungsanfällige und kritische Akteure des SC zu identifizieren [Rom18, S. 110]. Bei der SNA werden Akteure als Knoten und Beziehungen zwischen Akteuren als Kanten eines Netzwerks bzw. eines Graphen dargestellt [TL10]. Akteure können z.B. Personen oder Unternehmen sein, Beziehungen zwischen den Akteuren Kommunikation, Warenaustausch oder Geldfluss. Zur Modellierung können den Beziehungen Intensitäten zugewiesen werden, z.B. als Häufigkeit von Interaktionen, sodass stärkere und schwächere Kanten im Netzwerk existieren [Rom18, S. 111].

SNA umfasst eine Vielzahl an Aufgaben. Eine der Hauptaufgaben ist die Zentralitätsanalyse (Centrality Analysis). Ziel ist die Identifizierung des wichtigsten Akteurs im Netzwerk. Dabei stellt die Zentralität ein Maß der Wichtigkeit eines Knotens dar [TL10, S. 488]. Aus den Ergebnissen der Analyse lässt sich die Verwundbarkeit oder Robustheit des Netzwerks ableiten, da ein wichtiger Knoten eine kritische Ausfallstelle darstellt, bei dem ein Ausfall schwerwiegende Auswirkungen auf das Netzwerk hat [Rom18, S. 111]. Für die Zentralitätsanalyse existieren unterschiedliche Zentralitätsmaße, die die Wichtigkeit eines Knotens in verschiedenen Aspekten wiedergeben. Kim et al. fokussieren sich in Hinsicht auf eine Anwendung der Zentralitätsanalyse auf SCs auf die Degree Centrality (Grad), Closeness Centrality (Nähe) und die Betweenness Centrality [Kim+11, S. 196]. Da ein Großteil der SNA-bezogenen Literatur in englischer Sprache vorliegt, werden im Verlauf dieser Arbeit die englischen Begriffe der Maße übernommen.

Degree Centrality: Degree Centrality oder Grad misst die Anzahl an zu einem Knoten inzidenten Kanten. Die Degree Centrality ist damit definiert als

$$C_D(n_i) = \sum_j x_{ij}$$

wobei x_{ij} 1 bei einer vorhandenen Kante zwischen n_i und n_j oder 0 bei keiner vorhandenen Kante ist [Kim+11, S. 197]. Dabei reflektiert die Kennzahl die beziehungstechnische Aktivität eines Knotens [Fre78, S. 221]. Ein Nachteil liegt jedoch in der Abhängigkeit von der Größe des Graphen, der einen Vergleich zweier Graphen unterschiedlicher Größe erschwert, sodass Freeman eine auf die Anzahl an Knoten n im Graphen normierte Degree Centrality vorschlägt [Fre78, S. 221]:

$$C'_D(n_i) = \frac{\sum_j x_{ij}}{n-1}$$

Die Degree Centrality wird auf die zwei Arten von SN (siehe Abschnitt 3.2) unterschiedlich angewandt. Nach Kim et al. charakterisiert der Grad in einem auf vertragliche Beziehungen basierendem SN den Einfluss einer Firma auf andere Firmen in Bezug auf Aktivitäten und Entscheidungen [Kim+11, S. 197]. In einem auf Materialflüssen basierenden, gerichteten SC-Graphen wird zwischen der Anzahl an eingehenden (Indegree) und der Anzahl an ausgehenden Kanten (Outdegree) differenziert. Dabei erfasst ein hoher Grad der eingehenden oder ausgehenden Kanten die Transaktionsintensität oder damit verbundene Risiken für ein Unternehmen [Kim+11; PKS96]. Kim et al. definieren

Indegree und Outdegree als Schwierigkeitsgrad eines Unternehmens, den eingehenden Materialfluss von vorgelagerten bzw. den Bedarf von nachgelagerten Unternehmen zu organisieren [Kim+11, S. 197].

Closeness Centrality: „Die Unabhängigkeit eines Knoten wird durch die Nähe zu allen anderen Knoten des Graphen bestimmt“ [Fre78, S. 224]. Freeman beschreibt ein auf die Nähe bezogenes Zentralitätsmaß als Closeness Centrality, basierend auf der geodätischen Distanz, d.h. dem kürzesten Weg zwischen dem betrachteten Knoten und allen anderen Knoten im Graphen. Die Closeness Centrality wird definiert als

$$C_C(n_i) = \left[\sum_{j=1}^{g} d(n_i, n_j)\right]^{-1}$$

wobei $\sum_{j=1}^{g} d(n_i, n_j)$ der Gesamtdistanz zwischen n_i und allen anderen Knoten entspricht [Kim+11, S. 198]. Ebenso existiert ein normiertes Maß, das von der Größe des Graphen unabhängig ist [Mar02, S. 409]:

$$C'_C(n_i) = \left[\frac{\sum_{j=1}^{g} d(n_i, n_j)}{n-1}\right]^{-1} = \frac{n-1}{\sum_{j=1}^{g} d(n_i, n_j)}$$

Nach Kim et al. kommt eine Anwendung der Closeness Centrality im SC-Kontext nur für vertragliche Beziehungsgraphen in Betracht. In einem gerichteten Materialflussgraphen ergibt die Closeness Centrality keinen Sinn, da sich z.B. die geodätische Distanz zwischen n_i und n_j von der Distanz zwischen n_j und n_i unterscheiden würde [Kim+11, S. 197]. Die Autoren sehen die Closeness Centrality bei vertraglichen Beziehungen als Maß eines freien Handlungsspielraums von Unternehmen, um rechtzeitig auf Ressourcen zuzugreifen [Kim+11, S. 197]. Dabei hat ein Unternehmen mit hoher Closeness Centrality in der Regel kürzere SCs, die sich auf einen besseren Informationsaustausch auswirken.

Betweenness Centrality: Eine weitere Perspektive der Zentralität eines Knotens basiert auf der Häufigkeit eines Knotens, auf einem kürzesten Weg zwischen zwei Knoten zu liegen [Fre78, S. 221]. Die Betweenness Centrality misst diese Häufigkeit und wird definiert als

$$C_B(n_i) = \sum_{j<k} \frac{g_{jk}(n_i)}{g_{jk}}$$

wobei g_{jk} die Gesamtanzahl an kürzesten Wegen zwischen zwei Knoten und $g_{jk}(n_i)$ die Anzahl an kürzesten Wegen, die den Knoten n_i beinhalten, ist [Kim+11, S. 198]. Ein Knoten mit einer hohen Betweenness Centrality kann die Kontrolle über den Material- oder den Kommunikationsfluss ausüben [Fre78, S. 222]. Ein Knoten der nur auf wenigen kürzesten Pfaden liegt, verfügt nur über wenig Kontrolle. Kim et al. unterscheiden bei der Anwendung dieses Zentralitätsmaßes auf SNs erneut zwischen Materialflüssen und vertraglichen Beziehungen. Im Falle eines auf Materialflüssen basierenden Graphen, können Unternehmen mit einem hohen Betweenness-Wert die Rolle eines Hubs oder Umschlagplatzes einnehmen und das Maß als Einfluss über den Arbeitsablauf der nachgelagerten Unternehmen, wie z.B. die Lieferzeit, interpretiert werden. Damit geht von Unternehmen mit einer hohen Betweenness-Zentralität ein großes Risikopotenzial für die gesamte SC aus [Kim+11, S. 198]. In einem auf vertraglichen Beziehungen basierenden Netzwerk bezeichnet die Metrik das Ausmaß an Einfluss über die Interaktionen zwischen anderen Firmen. Ein Unternehmen mit hohem Betweenness-Wert agiert in diesem Fall als Broker, indem es den Handel zwischen anderen vermittelt und dies zu seinem eigenen Vorteil nutzt [Kim+11, S. 197-198].

3.3.2 Risikoidentifizierung mittels Supply Network Link Predictor

Eine Problematik, mit der Unternehmen einer SC konfrontiert sind, ist die fehlende Übersicht über das gesamte SN. Die Sorge vor einem Verlust des Vorteils gegenüber Konkurrenten durch das Teilen von Informationen stellt eine Hürde für die Kollaboration innerhalb der SC dar und führt zu einem Mangel an Daten über das reale Zulieferernetzwerk [ZFW18, S.2]. Die Unwissenheit über Zuliefererbeziehungen birgt ein Risikopotential, während hingegen eine Darstellung des gesamten Netzwerks die Identifizierung von SC-Risiken unterstützt [HBW03, S. 56]. Nach Bellamy und Basole kann eine erhöhte Sichtbarkeit eines Unternehmens über die SC zu einer reaktionsfähigen Risikoidentifizierung führen und Unternehmen mehr Zeit für Maßnahmen der Risikosteuerung einräumen [BB13, S. 244]. Brintrup et al. widmen sich dieser Problematik und stellen einen Ansatz namens Supply Network Link Predictor (SNLP) vor, um mittels Link Prediction (siehe Abschnitt 2.3) nicht sichtbare Kollaborationen zwischen Unternehmen aufzudecken [Bri+18]. Für die Link Prediction werden Informationen über die Topologie des Netzwerks mit aus dem Netzwerk abgeleiteten Informationen kombiniert. Der dem SN zu-

grundeliegende Graph besteht aus Zulieferern, die die Knoten repräsentieren, und Auftragsbeziehungen bzw. Materialflüssen, die die gerichteten Kanten darstellen. Eine Kante verbindet damit einen Verkäufer eines Produktes mit einem Käufer. Für die SNLP-Methode werden unvollständige Daten über ein SN herangezogen, anschließend eine Feature-Extraktion (siehe Abschnitt 2.2) durchgeführt, die relevante Muster für eine Wahrscheinlichkeitsschätzung von existenten Kanten aufzeigt und abschließend ein Verfahren eingesetzt, dass die extrahierten Features mit einer Wahrscheinlichkeitsschätzung verbindet. Die Autoren treffen im Rahmen der Merkmalsextraktion Annahmen über Muster und Eigenschaften von SCs, aus denen sich Wahrscheinlichkeiten von Verbindungen ableiten lassen. Die Merkmale werden anhand der folgenden Muster und Eigenschaften extrahiert:

Outsourcing-Verbindungen Grundlage der Annahme ist das Konzept des Outsourcing, wobei ein Unternehmen entscheidet, welche Produkte es produziert und welche Produktion es an andere Unternehmen auslagert, sodass es die Ware als Käufer einkauft. Sind die angebotenen Produkte der Zulieferer sowie die Produktabhängigkeiten, d.h. welche Komponenten für welche Baugruppe benötigt werden, bekannt, lässt sich daraus eine potentielle Verbindung zwischen Zulieferern ableiten. Bezogen auf eine potenzielle Käufer-Verkäufer-Beziehung wird eine Abhängigkeit des Produktes, das ein Käufer herstellt, von einem Produkt des Verkäufers angenommen. Stellt z.B. ein Verkäufer Lederwaren her und ein Käufer Ledersitze, so ist die Wahrscheinlichkeit einer Verbindung zwischen beiden hoch. Dabei wird eine Adjazenz umso wahrscheinlicher, je mehr Produktabhängigkeiten zwischen Käufer und Verkäufer existieren, d.h. je mehr Produkte des Verkäufers potenzielle Komponenten für die Produkte des Käufers sind [Bri+18, S. 4]. Brintrup et al. erzeugen dazu einen vollständigen, gewichteten Graphen $O^s\{N, L\}$ mit Zulieferern als Knoten und Outsourcing-Verbindungen als Kanten, indem zwischen jedem Zulieferer-Paar deren Produktportfolio auf Abhängigkeiten untersucht wird. Die Summe an Abhängigkeiten ist das Kantengewicht. Existieren keine Outsourcing-Verbindungen zwischen zwei Zulieferern s_i und s_j gilt $w(O_{s_i,s_j}) = 0$.

Käufer-Verbindung Zwischen Produkten, die ein Käufer von Verkäufern bezieht, können Abhängigkeiten existieren. So kann z.B. das Produkt eines Verkäufers kompatibel zu einem Produkt eines anderen Verkäufers sein, sodass ein Käufer der beiden Produkte diese von den beiden Verkäufern beziehen muss. Daraus kann sich über die Zeit eine Käufer-Verbindung zwischen den beiden Verkäufern etablieren. Aufgrund dessen treffen Brintrup et al. die Annahme, dass ein Unternehmen, das

Güter von einem Zulieferer bezieht, auch bei einem anderen (bisher unbekannten) Zulieferer einkauft, mit dem der erstgenannte (bekannte) Zulieferer eine Käufer-Verbindung hat. Für die Feature-Extraktion wird ein vollständiger, gewichteter Graph $B\{N, L\}$ erzeugt, in dem ein Knoten ein Zulieferer und eine Kante eine Käufer-Verbindung repräsentiert. Die Knoten werden aus dem ursprünglichen Graphen über das SN übernommen, vollständig verbunden und ein entsprechendes Kantengewicht $w(B_{s_i,s_j})$ zwischen zwei Knoten für jeden Fall, in denen beide Zulieferer (Knoten) einen gemeinsamen Käufer haben, um 1 inkrementiert. Haben zwei Zulieferer s_i und s_j keinen gemeinsamen Käufer, gilt $w(B_{s_i,s_j}) = 0$.

Konkurrenz-Verbindung Eine weitere Annahme, die die Autoren treffen, ist folgende: Je mehr sich die Produktportfolios zweier Zulieferer ähneln bzw. überdecken, desto höher ist die Wahrscheinlichkeit, dass es sich bei diesen um Konkurrenten handelt und desto geringer ist die Wahrscheinlichkeit einer Abhängigkeit voneinander oder einer gemeinsamen Kollaboration. Um Features dieser Art zu extrahieren, wird ein gewichteter, vollständig verbundener Graph $C\{N, L\}$, bestehend aus Zulieferern als Knoten und Konkurrenz-Verbindungen als Kanten erzeugt, indem ebenfalls die Knoten aus dem gegebenen SN übernommen und miteinander verbunden werden. Anschließend werden die Produktportfolios von jedem Paar Unternehmen verglichen und für jede Übereinstimmung der Produkte das entsprechende Kantengewicht erhöht. Die Gewichtung der Kante $w(C_{s_i,s_j})$ ist gleich der Anzahl der übereinstimmenden Produkte der Zulieferer s_i und s_j.

Grad Die letzte getroffene Annahme über Muster in der SC ist, dass Unternehmen in einer SN existieren, die die Rolle eines Hubs einnehmen und damit über signifikant mehr Verbindungen zu anderen Unternehmen aufweisen als andere Unternehmen des Netzwerks. Darauf aufbauend wird angenommen, dass die Wahrscheinlichkeit eines Zulieferers, weitere Verbindungen mit anderen Unternehmen einzugehen, mit der Anzahl der bereits vorhandenen Verbindungen steigt. Bezogen auf den Graphen bedeutet dies, dass die Wahrscheinlichkeit einer weiteren inzidenten Kante mit dem Grad des Knoten D_{s_i} korreliert.

Das Resultat der Merkmalsextraktion sind die vier, auf jeweils ein Knotenpaar s_i und s_j bezogene, Merkmale $w(O_{s_i,s_j})$, $w(B_{s_i,s_j})$, $w(C_{s_i,s_j})$ und D_{s_i}. Anhand dieser Merkmale eines Knotenpaars soll ein binärer Klassifikator eine abhängige Variable L_{s_i,s_j} bestimmen, die entweder den Wert 1 für eine das

Knotenpaar verbindende Kante oder eine 0 für eine nicht existente Kante annimmt. Für die Klassifikation nutzen Brintrup et al. die Logistische Regression und den naiven Bayes-Klassifikator. Aus dem gegebenen SC-Graphen wird ein Datenset erstellt, in dem zu jedem Knotenpaar ein Datensatz mit den zugehörigen und berechneten Merkmalen besteht. Aus diesem werden Trainingsdaten entnommen, auf denen der Klassifikator trainiert wird. Die Autoren testen die SNLP-Methode auf SNs-Datensets mit über 1000 Knoten und erreichen mit beiden Klassifikationsverfahren gute Ergebnisse bei der Vorhersage von nicht-sichtbaren Verbindungen.

3.3.3 Risikoidentifizierung mittels Netzwerkvisualisierung

Romeike hebt einen weiteren Aspekt der SNA als Methode für die Risikoidentifizierung in SNs hervor: Die Abbildung und grafische Darstellung des Netzwerkes und der Abhängigkeiten [Rom18, S. 114]. Auch Harland et al. stellen ein Software-Tool für ein umfassendes Risikomanagement vor, das als ersten Schritt die Abbildung des SN vorsieht und darauf aufbauend die Identifizierung von Risiken erfolgt [HBW03, S. 56].

Im Rahmen des SCM messen auch Park et al. der Visualisierung von SNs eine große Bedeutung bei, indem sie eine Architektur für ein interaktives, visuell-analytisches SC-Entscheidungsunterstützungssystem vorschlagen [PBB16, S. 100]. Anhand von Literatur, einer umfassenden Umfrage und Experteninterviews leiten die Autoren Anforderungen an das Design eines visuell-analytischen Systems zur Entscheidungsunterstützung im SCM ab. Weiterhin beschreiben sie die Implementierung eines Prototypen und evaluieren den Prototypen und die grafischen Darstellungen und Interaktionen. Die aufgestellten und evaluierten Anforderungen sind wie folgt:

Mehrfache Ansichten in einem integrierten Interface bereitstellen Wechseln Benutzer zwischen Ansichten, kann es dadurch zu einem Kontextverlust kommen. Um dies zu verhindern, schlagen die Autoren mehrere Ansichten der Datenrepräsentation in dem Benutzerinterface einzublenden. Die Ansichten können zwar die selben Daten visualisieren, sollen aber unterschiedliche Repräsentationen der Daten darstellen und damit zusätzliche Erkenntnisse liefern. [PBB16, S. 91-92]

Interaktive Untersuchung des SN erlauben Eine interaktive Visualisierung bietet im Gegensatz zu statischen Bildern die Möglichkeit, detaillierte Informationen nach Bedarf zur Verfügung zu stellen. Dabei wird der Benutzer nicht mit allen Daten und Informationen überladen, sondern erhält stattdessen ein übersichtliches Gesamtbild. [PBB16, S. 92]

Analytische Fähigkeiten einbinden Das Nutzen von Visualisierungen zur Vorhersage zukünftiger Verläufe ist Gegenstand zahlreicher Forschungen. Insbesondere im SCM werden analytische und voraussagende Fähigkeiten benötigt. Park et al. führen zu diesem Punkt das Beispiel der Risikoanalyse an, in dem ein Manager die Auswirkungen einer Störung in der SC nachvollziehen möchte. Analytische Kennzahlen sollen hierbei helfen, um das Netzwerk besser zu untersuchen. [PBB16, S. 92]

Kapitel 4

Implementierung einer auf Graph Mining basierenden Open-Source-Applikation zur Risikoidentifizierung von Supply Chains

Nachdem die Grundlagen über Graph Mining und Methoden der Risikoidentifizierung vermittelt worden sind, wird im Folgenden eine Open-Source-Applikation zur Risikoidentifizierung von SCs, die Methoden des Graph Minings nutzt, entwickelt. Für den gesamten Entwicklungsprozess mit einem implementierten und evaluierten Prototypen als finales Ziel sowie der Anforderungsspezifikation und der Planung einer geeigneten Softwarearchitektur als Teilziele wird ein Vorgehensmodell gewählt. Die Wahl fällt auf das sequentielle Wasserfallmodell ohne Rückführschleifen. Gründe dafür liegen in der überschaubaren Einteilung von Phasen, deren Ergebnisse mit den Teilzielen dieser Arbeit übereinstimmen, sowie in der Einfachheit der Strukturierung. Auch ist im Rahmen der Entwicklung eines Prototypen nicht mit Änderungen der Anforderungen zu rechnen, sodass die Phasen sequentiell durchschritten werden können ohne erneut in eine frühere Phase zurückspringen zu müssen. Die Entwicklungsschritte bzw. Phasen des Wasserfallmodells in seiner ursprünglichen Form sind *Aufstellen der Requirements (Anforderungsspezifikation), Systemanalyse, Systementwurf, Implementierung, Test & Integration und Abnahme* (siehe 4.1). Die Dokumentation der Phasen und der Phasenergebnisse nehmen im Wasserfallmodell einen hohen Stellenwert ein [Gra14, S. 114]. Daher erfolgt die Gliederung dieses Kapitels anhand der Phasen und Anforderungen. Abläufe werden in den einzelnen Abschnitten präzise

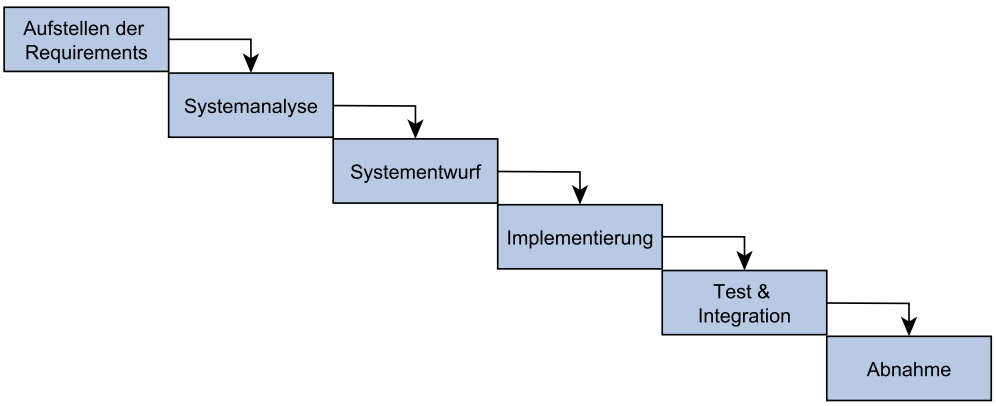

Abbildung 4.1: Phasen eines sequentiellen Wasserfallmodells ohne Rückführschleifen nach [Gol11, S. 84]

und strukturiert beschrieben. Dabei werden in diesem Kapitel *Anforderungsspezifikation*, *Systemanalyse* und *Systementwurf* in jeweils einem Abschnitt betrachtet, die Dokumentation der weiteren Phasen erfolgt in Kapitel 5.3. Zunächst werden in Abschnitt 4.1 Anforderungen an die Applikation aufgestellt. Anhand der Anforderungen kann anschließend in Abschnitt 4.2 ein Applikationsmodell als fachliche Lösung abgeleitet werden. Aus der fachlichen Lösung lässt sich anschließend in Abschnitt 4.3 eine technische Lösung in Form einer Software-Architektur entwerfen.

4.1 Anforderungsspezifikation

„Anforderungen legen fest, welche Eigenschaften ein zu entwickelndes Softwaresystem besitzen soll“ [Bal09, S.437]. Um die Anforderungen an die geplante Applikation aufzustellen, wird sich am Requirements Engineering orientiert. Das Requirements Engineering besteht aus Tätigkeiten mit dem Ziel, Anforderungen zu erheben, analysieren, verstehen und dokumentieren [Her+13, S. 9]. Die Anforderungen werden dabei klar beschrieben, um die Funktionalität, das Aussehen und die Mittel zur Realisierung vorzugeben. Wie das Produkt realisiert wird, legt die Anforderungsbeschreibung hingegen nicht fest [Wit19, S. 61]. In der Regel werden zwischen funktionalen Anforderungen („Was soll ein System tun?“) und nicht-funktionalen Anforderungen im Sinne von Qualitätsanforderungen („Wie gut sollen die Funktionen erledigt werden?“) unterschieden [Bal09, S. 463-465].

Für die Dokumentation der Anforderungen werden der Anforderungstyp (funktional/nicht-funktional), die Beschreibung, die Quelle in Hinblick

auf Ursprung und Begründung der Anforderung und Abnahmekriterien definiert. Die Anforderungen basieren zunächst auf den Methoden zur Risikoidentifizierung (siehe Abschnitte 3.3.1, 3.3.2 und 3.3.3). Die im Rahmen der Aufgabenstellung zu entwickelnde Applikation wird im Folgenden als System bezeichnet.

1 - Das System muss fähig sein, zentrale Knoten in einem SN zu erkennen Diese Anforderung ist funktional. Sie beschreibt dabei eine Kernfunktion der Risikoidentifizierung. Wie in Abschnitt 3.3.1 erläutert, sind die relativen Positionen der Knoten in einem Graphen bzw. in einem SN wichtige Eigenschaften. Die Zentralität stellt dabei ein Maß der Wichtigkeit eines Knotens dar und damit auch ein Maß für damit verbundene SC-Risiken. Zentrale, wichtige Knoten können kritische Ausfallstellen (und damit Risiko-Objekte) sein, bei denen eine potenzielle Störung sich schwerwiegend auf das Netzwerk auswirken kann. Mittels Zentralitätsmaßzahlen, die sich für jeden Knoten berechnen lassen, lässt sich die Zentralität zur quantitativen Risikobewertung nutzen. Die in der Literatur populären und erprobten Zentralitätsmaße *Degree Centrality*, *Closeness Centrality* und *Betweenness Centrality* werden für die Aufgabe der Risikoidentifizierung in SN übernommen. Die Anforderung gilt als erfüllt, sofern das System eine Berechnung der drei Zentralitätsmaße für jeden Knoten des Graphen ermöglicht.

2 - Das System muss fähig sein, nicht-sichtbare Verbindungen zwischen Zulieferern in einem SN aufzudecken Diese Anforderung ist funktional. Hiermit soll eine Methode der Risikoidentifizierung und damit eine Kernfunktionalität des Systems definiert werden. Wie bereits in Abschnitt 3.3.2 beschrieben, sind Daten eines Unternehmens über das eigene SN häufig nicht vollständig. Die Unwissenheit über Kollaborationen zwischen Zulieferern in dem Netzwerk bergen ein unbekanntes Risikopotenzial. Das Sichtbarmachen von bisher unbekannten Verbindungen zwischen Zulieferern kann eine Risikoidentifizierung unterstützen. Die Anforderung basiert auf der Arbeit von Brintrup et al. [Bri+18], die einen Ansatz mittels Link Prediction (siehe Abschnitt 2.3) bieten, um mögliche Verbindungen aufzudecken. Zur Erfüllung dieser Anforderung soll sich an der von Brintrup et al. entwickelten *SNLP*-Methode orientiert werden. Dazu ist eine Methode zu implementieren, die Outsourcing-, Käufer- und Konkurrenzverbindungen sowie Knotengrade aus einem gegebenen Graphen extrahiert und anhand dieser Merkmale eine Klassifikation über das Vorhandensein einer Kante zwischen den Knoten (Zulieferern) durchführt.

3 - Das System muss dem Anwender die Möglichkeit bieten, relevante Informationen in einem SN darzustellen Diese Anforderung ist ein funktionales Kernelement der Risikoidentifizierung. Die Visualisierung relevanter Informationen kann ebenfalls als Methode der Risikoidentifizierung bzw. den gesamten Risikomanagementprozess unterstützend eingesetzt werden (siehe Abschnitt 3.3.3). Zu den relevanten Informationen im Umfeld einer Risikoidentifizierung in SN gehören sowohl die Graphenstruktur als auch die den jeweiligen Knoten und Kanten zugehörigen Eigenschaften. Die Anforderung basiert auf der Studie von Park et al., die bereits in Abschnitt 3.3.3 beschrieben wurde. Die Autoren leiten aus den Ergebnissen der durchgeführten Umfragen und Experteninterviews Designanforderungen an ein SC-Entscheidungsunterstützungssystem ab. Da die zu implementierende Applikation zur Risikoidentifizierung sich nur unwesentlich von einem Entscheidungsunterstützungssystem unterscheidet, können die Anforderungen übernommen werden. Den drei beschriebenen Designanforderungen wird eine weitere hinzugefügt, die die allgemeine Visualisierung der Graphenstruktur einbezieht und sich auf den Erkenntnissen von Romeike sowie Harland et al. (gleicher Abschnitt) begründet. Insgesamt lassen sich hieraus vier Teilanforderungen übertragen:

3a - Das System soll fähig sein, die Graphenstruktur eines SN zu visualisieren
Hierbei handelt es sich um eine funktionale Anforderung. Die Visualisierung eines SN gilt, wie oben beschrieben, als Methode der Risikoidentifizierung. Eine graphische Darstellung von Beziehungen zwischen Objekten wird als intuitiv empfunden, sodass einem Benutzer die Möglichkeit gegeben wird, Zusammenhänge in einem SN zu verstehen. Die Anforderung gilt als erfüllt, wenn das System ein solches Netzwerk als Graph visualisieren kann.

3b - Das System soll fähig sein, mehrfache Ansichten in einem integrierten Interface bereitzustellen
Hierbei handelt es sich um eine nicht-funktionale Anforderung, da sie die Funktionalität nicht beeinflusst und die Ansichten auch erreichbar wären, wenn sie nicht in einem Interface gemeinsam bereitgestellt werden. Ein Benutzer, der bei der Betrachtung der visualisierten Daten zwischen Ansichten wechselt, kann möglicherweise den Kontext verlieren. Um dies zu vermeiden, werden mehrere Ansichten in einem grafischen Benutzerinterface integriert. Verschiedene Darstellungen der Daten sollen so die Möglichkeit schaffen, auf Basis der Daten zu weiteren Erkenntnissen zu gelangen. Die Anforderung basiert auf den oben genannten Studien zu visuell-analytischen Systemen zur Entscheidungsunterstützung im SCM von Park et al. (siehe auch 3.3.3). Bie-

tet das System mehrere verschiedene Repräsentationen der gleichen Daten in einem Interface, gilt diese Anforderung als erfüllt.

3c - Das System soll dem Anwender die Möglichkeit bieten, interaktiv das SN zu untersuchen
Diese Anforderung ist ebenfalls nicht-funktional. Um die Ansichten nicht mit allen verfügbaren Daten für den Benutzer zu überladen, ist eine Detailansicht nach Interaktion sinnvoll. Damit lassen sich detaillierte Informationen zu Objekten interaktiv abrufen, während die Darstellung des gesamten Graphen übersichtlich bleibt. Das Minimum an visualisierten Daten soll die Darstellung des Graphen mit benannten Elementen bieten, alle weiteren Daten werden zunächst ausgeblendet. Die Anforderung gilt als erfüllt, sofern zusätzliche Daten über Elemente des SN durch eine Nutzerinteraktion abgerufen und sichtbar werden.

3d - Das System muss fähig sein, analytische Fähigkeiten einzubinden
Da die Analyse der Daten für eine Risikoidentifizierung zu einer Kernfunktion des Systems zählt, ist diese Anforderung als funktional einzuordnen. Dabei dient die Benutzeroberfläche als Schnittstelle zwischen der Datenanalyse und dem Benutzer sowohl zur Visualisierung der Graphdaten von SN als auch im Sinne einer Bedienmöglichkeit, um eine Datenanalyse anzustoßen. Die Datenanalyse soll dabei mithilfe von Methoden des Graph Minings geschehen, die in den Anforderungen 1 und 2 erläutert sind. Die Anforderung gilt als erfüllt, wenn das System dem Benutzer die Möglichkeit bietet, eine Datenanalyse über die Benutzeroberfläche zu starten und die Ergebnisse der Analyse in einer Ansicht wiedergeben kann.

4 - Das System soll eine Erweiterbarkeit der Funktionalität gewährleisten Obwohl sich mit der Umsetzung dieser Anforderung die Funktionalität erweitern lässt, wird sie dennoch als nicht-funktional betrachtet, da sie sich auf die Qualität der Umsetzung bezieht. Die Erweiterbarkeit der Anwendung ist Teil der Aufgabenstellung dieser Arbeit und bietet eine Möglichkeit, das System um weitere Funktionen zu erweitern oder für andere Anwendungsgebiete anzupassen. Ein modularer Aufbau der Applikation und eine sorgfältige Trennung von Programmteilen der Visualisierung und der Logik sollen diese Erweiterbarkeit gewährleisten. Ist die Anwendung so umgesetzt, gilt die Anforderung als erfüllt.

5 - Das System soll als Open-Source-Software (OSS) veröffentlicht werden Diese Anforderung ist nicht-funktional, bezieht sich jedoch nicht nur auf die Veröffentlichung der Applikation, sondern kann ebenfalls Auswirkungen auf die Implementierung haben. Bei der Einbindung von geeigneten Software-Bibliotheken muss diese Anforderung beachtet werden, wenn eine entsprechende Open-Source-Lizenz ausgewählt werden soll. Die Anforderung hat ihren Ursprung in der Aufgabenstellung dieser Arbeit. Sie gilt als erfüllt, wenn sie den Merkmalen bzw. der Definition von OSS im Sinne der Open Source Initiative entspricht: Dabei muss die Software als von Menschen lesbarer Quelltext vorliegen und die Lizenz, unter die die Software gestellt wird, beliebiges Kopieren, Verbreiten, Nutzen, Verändern und die Weitergabe in veränderter Form erlauben [Ope07].

4.2 Systemanalyse

Bei der Systemanalyse werden die Anforderungen an Leistungsmerkmale, die Funktionen des Systems, das Zusammenspiel der Systemfunktionen untereinander und das Verhalten der Systemfunktionen zu anderen Systemen bestimmt [Wei00, S. 1715]. Die Systemanalyse konzentriert sich auf den sogenannten Problembereich. Hierbei sollen die Anforderungen umgesetzt werden, indem ein logisches Modell des Systems, auch Fachkonzept oder fachliche Lösung genannt, erstellt wird [Gol11, S. 42]. Technische Einschränkungen werden nicht betrachtet. Ergebnisse der Systemanalyse sind ein Systemmodell sowie ggf. ein Konzept der Benutzeroberfläche und ggf. ein Konzept des Benutzerhandbuchs [Bal09, S. 548]. In einiger Fachliteratur wird die Systemanalyse auch als umfassender Prozess verstanden, der auch die Ermittlung von Anforderungen beinhaltet [Rup13]. In dieser Arbeit wird die Systemanalyse als auf die Anforderungsspezifikation aufbauender Schritt des Software-Entwicklungsprozesses verstanden, in der eine fachliche Lösung erarbeitet wird.

Für die Systemanalyse wird sich der Methode Objektorientierte Analyse (OOA) bedient, da sie als etablierte Methode bei der Modellierung eines Systems gilt [Bal09, S. 548]. In Hinsicht auf einen folgenden Systementwurf ist eine Weiterverwendung der Analyseergebnisse möglichst ohne Notationsbrüche hilfreich, sodass ein Modell der OOA direkt in ein Modell des objektorientierten Software-Designs übertragen werden kann [Rup13, S. 4]. Die OOA modelliert daher die fachliche Lösung mit Hilfe der objektorientierten Grundkonzepte (Objekt, Klasse, Attribut, Operation, Botschaft und Vererbung) [Bal09, S. 549-550]. Vorgehensweisen bei der OOA unterscheiden zwischen der Orientierung an den Informationen bzw. Daten des Systems,

aus dem sich ein statisches Modell entwickeln lässt, und der Orientierung an der Funktionalität des Systems mit einem dynamischen Modell als Ergebnis. Ebenfalls möglich ist eine Kombination beider Orientierungen. Übliche Notationen der Modelle basieren auf dem Standard Unified Modeling Language (UML) sowie dem Entity-Relationship-Modell (ERM) [Bal09, S. 559].

Zunächst wird sich an den Daten orientiert, um ein statisches Modell zu erzeugen. Die Anforderungen 1, 2 und 3a aus Abschnitt 4.1 basieren auf Daten in einer Graphenstruktur. Die Modellierung der Graphdaten erfolgt mittels ERM. Um das Modell zu skizzieren, wird ein Graphdatenbankschema verwendet, das auf dem Entity-Relationship-Diagramm (ERD) basiert und aus den Komponenten Knoten, Kanten, Eigenschaften und Kardinalitäten besteht [Roy+17, S. 5]. Die Auswahl dieses Schemas wird dadurch begründet, dass sie direkt im Systementwurf verwendet und mit geringem Aufwand in die Struktur einer gewählten Datenbank umgesetzt werden kann. Im Folgenden wird das Graphdatenbankschema nach Roy-Hubara, Rokach et al. [Roy+17] kurz erläutert: Ein Knoten repräsentiert eine Entität, der ein sogenanntes Label und Eigenschaften besitzt, darunter eine ID zur eindeutigen Identifizierung der Entität. Entsprechend dem ERM wird zwischen einem abstrakten Entitätstypen und der eigentlichen Entität unterschieden. In dem Schema werden sowohl Knotentyp als auch der Knotenobjekt (Entität) als Knoten skizziert. Analog zur Graphentheorie (siehe Abschnitt 2.1) stellt eine Kante eine Beziehung zwischen zwei Knoten dar. Hierbei erfolgt ebenfalls eine Unterscheidung zwischen abstrakten Kantentyp und einer spezifischen Kante. Kanten gelten als gerichtet. Damit folgt das Schema dem Labeled-Property-Graph-Modell. Eigenschaften können sowohl Knoten als auch Kanten zugeordnet werden. Kardinalitäten werden zwischen zwei Knoten einer Kante definiert, als Notation der Kardinalität wird sich am ERD orientiert. Aufbauend auf den Erkenntnissen aus Abschnitt 3.1 wird ein SN als Graph mit Unternehmen als Knoten und Materialflüssen als Kanten repräsentiert. Um Anforderung 2 zu erfüllen, müssen zudem auch Daten über die produzierten Güter (Produkte) der Unternehmen sowie deren Produktabhängigkeiten voneinander (z.B. werden zur Herstellung von Produkt A Produkt B und C benötigt, d.h. Produkt A ist von den Produkten B und C abhängig) gehalten werden (siehe Abschnitt 3.3.2).
Zur Modellierung der Daten wird daher ein Graph gewählt, der die vielfachen Verknüpfungen zwischen Unternehmen und Produkten wiedergeben kann. Sowohl Unternehmen als auch Produkte werden als Knoten und Materialflüsse als Kanten zwischen Unternehmen, Produktionsbeziehungen als Kanten zwischen Unternehmen und Produkten und Produktabhängigkeiten als Kanten zwischen Produkten in einem Graphen modelliert. Aufbauend auf diese Erkenntnisse wird ein Graphdatenbankschema erstellt, das die Kno-

tentypen „Unternehmen“ und „Produkt“ sowie die Kantentypen „beliefert“, „produziert“, „ist abhängig von“ einbezieht. Die Kardinalität des Kantentyps „beliefert“ ist eine m:n-Kardinalität, da ein Unternehmen beliebig viele Unternehmen beliefern kann und ebenfalls durch beliebig viele Unternehmen beliefert werden kann. Ist ein entsprechendes Unternehmen am Anfang oder am Ende der SC einzuordnen, muss das Unternehmen nicht zwingend einen Zulieferer besitzen bzw. nicht an ein weiteres Unternehmen liefern. Ein Produkt wird von mindestens einem Unternehmen produziert, Unternehmen können beliebig viele Produkte herstellen. Die Abhängigkeitsstruktur der Produkte entspricht einer Baumstruktur, in der m:n-Kardinalitäten gelten. Neben der Beziehungsstruktur zwischen Unternehmen und Produkten sind ebenfalls Eigenschaften dem Modell hinzuzufügen. Bei der Modellierung der Datenstruktur als Graph wird zwischen Knotenattributen und Kantenattributen unterschieden. Das Modell wird so generisch wie möglich gehalten, da das Schema auf eine Vielzahl unterschiedlicher SN-Daten anwendbar sein soll. Da die Abbildung der Daten in einem Labeled-Property-Graph eine dynamische Menge an Attributen zulässt, die nach Belieben erweitert werden kann, werden in dem Schema ausschließlich zwingend benötigte Attribute einbezogen. Dabei wird das Schema auf ein für alle SN-Datenmengen gemeinsames Datenmodell reduziert. Benötigte Knotenattribute sind eine eindeutige Identifikation (ID) sowie ein Name für den Knotentyp Unternehmen und eine Bezeichnung für den Knotentyp Produkt. Name und Bezeichnung werden aufgrund der Anforderung 3a (siehe Abschnitt 4.1) benötigt, damit der Benutzer bekannte Knoten des SN-Graphen erkennen und einordnen kann. Ausgehend von den Anforderungen ergeben sich keine Kantenattribute. Das resultierende Graphdatenbankschema in der Chen-Notation ist in Abbildung 4.2 zu sehen.

Für das dynamische Modell werden Szenarien des Benutzers illustriert, die der Bestimmung der Funktionalitäten des Systems dienen. Dazu werden Anforderungen als Anwendungsfälle dokumentiert, die dann im weiteren Verlauf der Systemanalyse und des -entwurfs verwendet werden und ein besseres Verständnis der Funktionalität liefern. Ein Anwendungsfall (engl. Use Case) beschreibt die Wahrnehmung eines Akteurs aus der Außensicht bei der Benutzung des Systems [Bal09, S.251]. Als Akteure werden die Rollen der Menschen oder Systeme bezeichnet, die auf das System zugreifen. In dem UML-Anwendungsfalldiagramm werden Akteure als Strichmännchen und Anwendungsfälle als Oval dargestellt und mit einem Namen beschriftet. Beziehungen zwischen Akteuren und Anwendungsfällen werden durch Pfeile oder Linien gekennzeichnet [DWT15, S. 122]. Als Anwendungsfälle werden die Kernfunktionalitäten der Anforderungen, die nach außen hin sichtbar sind, übernommen und jeweils mit einem kurzen Anwendungsfallnamen zu-

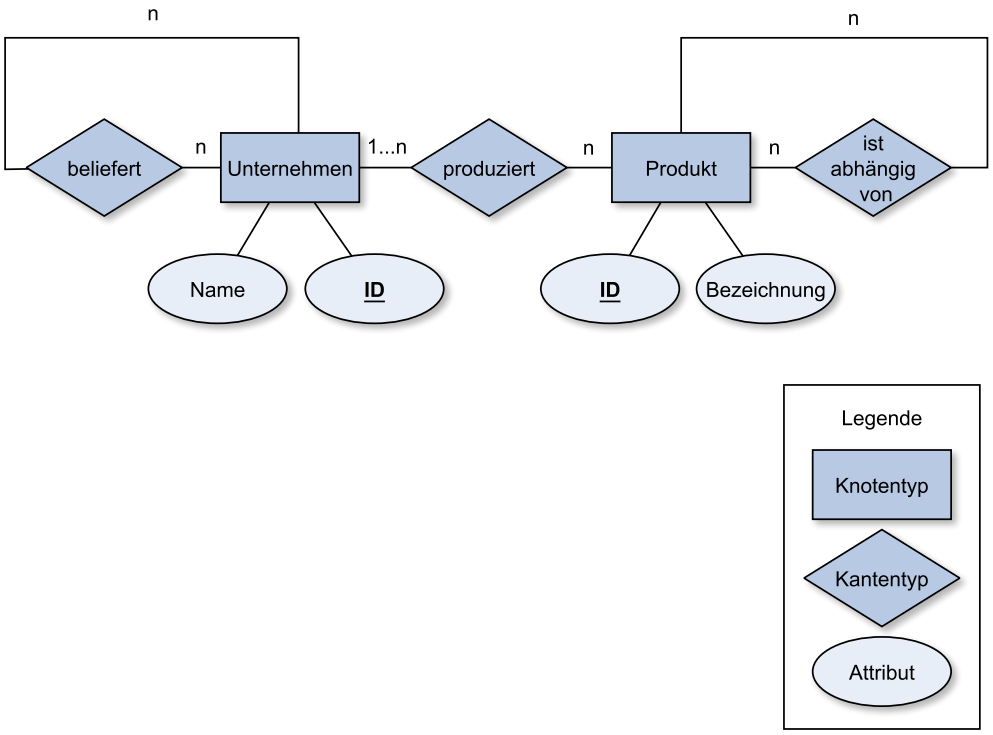

Abbildung 4.2: Graphdatenbankschema in der Chen-Notation mit Knoten- und Kantentypen sowie den berücksichtigten Attributen, das ein SN abbildet.

sammengefasst. Als erster, allgemeingültiger Anwendungsfall ist die Identifizierung von Risiken zu nennen, bei der das System unterstützen soll. Aus den Anforderungen 1, 2, 3a und 3c ergeben sich „Wichtige Zulieferer erkennen“, „nicht-sichtbare Verbindungen aufdecken“, „Struktur des SN einsehen“, „Details zu Unternehmen abrufen“. Anforderungen 3b, 3d, 4 und 5 beschreiben die Umsetzung des Systems und werden daher nicht in das Diagramm aufgenommen. Akteure sind zum einen Anwender aus dem Bereich des SCM sowie der Forschung. Daraus ergibt sich das in Abbildung 4.3 dargestellte Anwendungsfalldiagramm.

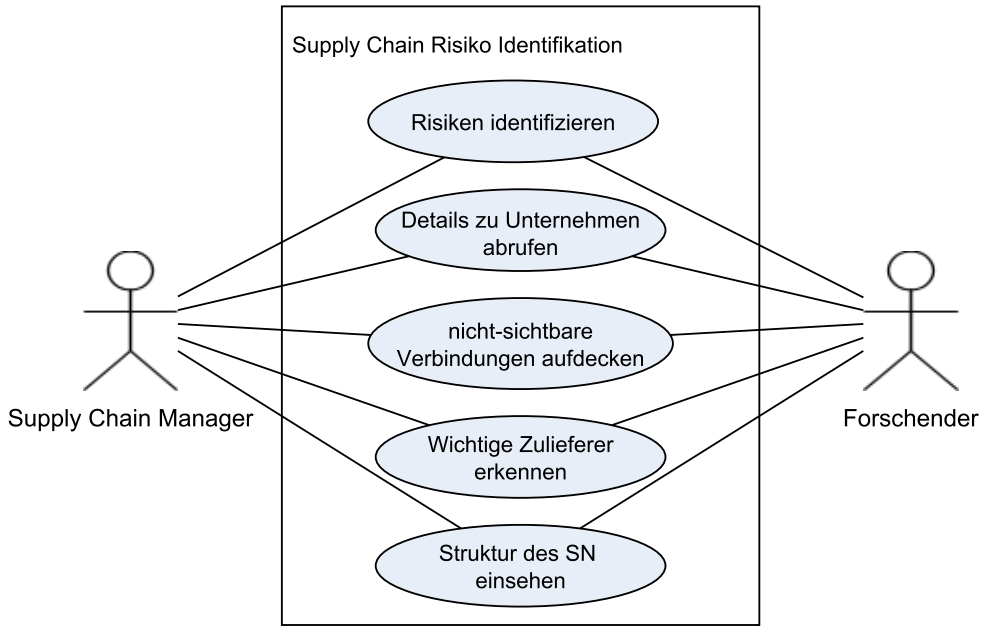

Abbildung 4.3: Anwendungsfalldiagramm des Systems „Supply Chain Risiko Identifikation“

Die im Diagramm als Blackbox skizzierten Anwendungsfälle lassen sich nun weiter konkretisieren, indem das Verhalten des Systems modelliert wird. Hierzu wird der Ablauf eines Anwendungsfalles geplant und mittels Aktivitätsdiagramm dokumentiert. Ein Aktivitätsdiagramm visualisiert die Ablaufreihenfolgen in einem Prozess [Rup13, S. 69]. Zu den Komponenten des Diagramms gehören Aktionen (dargestellt als abgerundete Rechtecke), Kontrollflüsse (Pfeile), initiale Knoten als Startpunkt der Abläufe (dunkel ausgefüllter Kreis), finale Knoten als Endpunkt (dunkel ausgefüllter Kreis in einem umschließenden Kreis), Entscheidungsknoten zur Darstellungen von Bedingungen (Raute mit ausgehenden Pfeilen), zusammenführender Knoten (Raute mit eingehenden Pfeilen) und weiteren Elementen [DWT15, S. 132]. Der Anwendungsfall „Risiken identifizieren“ wird hierbei nicht betrachtet, da die anderen Anwendungsfälle in ihrer Gesamtheit die Mittel der Risikoidentifizierung darstellen. Eine erfolgreiche Risikoidentifizierung wird daher immer durch Interaktion und Interpretation des Anwenders bedingt (siehe 3.3). Zunächst wird der Anwendungsfall „Struktur des SN einsehen“ betrachtet. Auf eine initiale Interaktion des Benutzers folgend, müssen die Daten über das SN geladen werden. Sind neben Zulieferern und deren Lieferbeziehungen weitere Daten wie bspw. Produktdaten vorhanden, entscheidet der Benutzer, ob das abzubildende Netzwerk auch diese Daten enthalten sollen. Andernfalls sind

ausschließlich Daten über das Unternehmensnetzwerk auszuwählen. Weiterhin müssen die Daten in eine Graphstruktur für die Visualisierung überführt werden. Ein gewähltes Graph-Layout legt fest, wie Knoten und Kanten positioniert werden, sodass sie sich möglichst nicht überdecken und den Graphen übersichtlich darstellen. Nach der Berechnung des Layouts werden die Elemente des Graphen gerendert und in der graphischen Benutzeroberfläche abgebildet. Abbildung 4.4 zeigt das entsprechende Aktivitätsdiagramm. Der

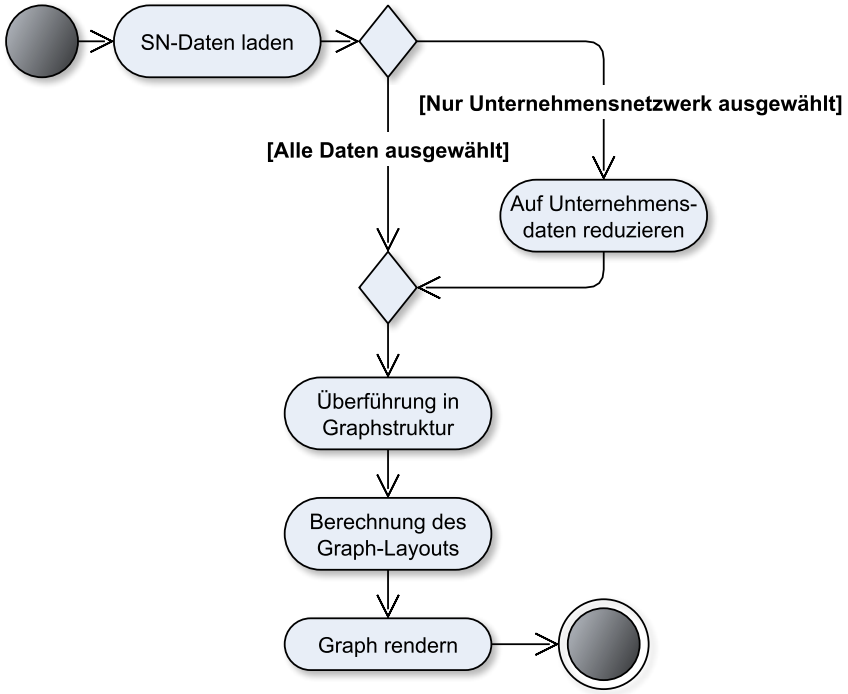

Abbildung 4.4: UML-Aktivitätsdiagramm des Anwendungsfalls „Struktur des SN einsehen“

Anwendungsfall „Details zu Unternehmen abrufen“ erfolgt durch ähnliche Ablaufschritte. Hierzu werden vorab die SN-Daten geladen und der gerenderte Graph in der Benutzeroberfläche dargestellt. Wie in Anforderung 3c beschrieben, sollen detaillierte Informationen zu Objekten interaktiv abgerufen werden (siehe Abschnitt 4.1). Da die Knoten des Graphen ausschließlich Namen und inzidente Kanten in der visualisierten Graphdarstellung aufweisen, jedoch in der Regel mit weiteren Daten über Unternehmen bzw. Zulieferer und Produkte verknüpft sind, soll eine Interaktion mit den Knoten die weiteren Daten laden und anzeigen. Als Interaktion muss der Knoten durch den Benutzer in der Oberfläche ausgewählt werden.

Im Folgenden wird der Anwendungsfall „Wichtige Zulieferer erkennen“, der auf Anforderung 1 basiert, betrachtet. Die Relevanz der Knoten wird durch ihre Zentralität bestimmt (siehe Abschnitt 3.3.1). Daher gilt es, Zentralitätsmaße der einzelnen Knoten zu bestimmen und diese dem Benutzer in der grafischen Oberfläche bereitzustellen. Um als Anwender die zentralen und damit wichtigen Knoten des SN schnell zu erfassen, soll die Größe der Knoten abhängig von der Zentralität festgelegt werden. So erscheinen zentrale und wichtige Knoten groß und bieten eine Überblick auf die zentralen Elemente des SN. Dabei muss gewährleistet werden, dass Knoten mit geringer Zentralität auch dann sichtbar bleiben, wenn die höchste Zentralität um ein Vielfaches größer ist als die der nicht zentralen Knoten. Aus diesem Grund wird eine minimale und eine maximale Knotengröße festgelegt, die der geringsten bzw. größten gegebenen Zentralität zugeordnet wird und alle weiteren Knotengrößen zwischen den festgelegten Größen interpoliert werden. Es soll ebenfalls möglich sein, alle Knoten in der gleichen Größe darzustellen. Ablaufschritte beinhalten die Berechnung der Zentralitätsmaße, sowie der davon abhängigen Knotengrößen und das Rendern des Graphen. Das Aktivitätsdiagramm wird in Abbildung 4.5 veranschaulicht.

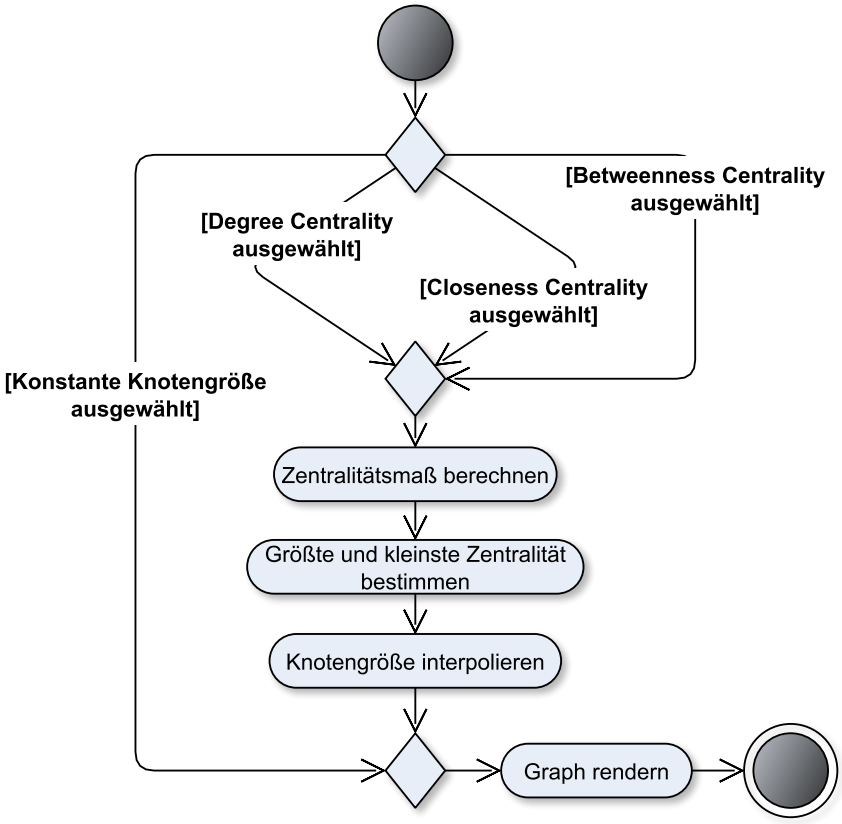

Abbildung 4.5: UML-Aktivitätsdiagramm des Anwendungsfalls „Wichtige Zulieferer erkennen“

Der zuletzt betrachtete Anwendungsfall „nicht-sichtbare Verbindungen aufdecken“ basiert auf Anforderung 2 sowie der in Abschnitt 3.3.2 beschriebenen SNLP-Methode. Anhand dieser wird sich auch beim Modellieren der Ablaufschritte orientiert. Mittels Link Prediction (siehe Abschnitt 2.3) sollen Knotenpaare klassifiziert werden, sodass eine mögliche Existenz einer Kante, die das Knotenpaar verbindet, vorhergesagt wird. Die Klassifikation erfolgt anhand von vier Merkmalen, die von der SNLP-Methode von Brintrup et al. (siehe Abschnitt 3.3.2) übernommen werden. Die vier Merkmale eines Knotenpaars sind die Outsourcing-Verbindungen, Käufer-Verbindungen, Konkurrenz-Verbindungen und der Grad des Startknotens, die aus dem gegebenen Graphen extrahiert werden müssen. Zum Trainieren des Klassifikators werden Datensätze aus dem Knotenpaar, den darauf bezogenen Merkmalen sowie der Adjazenz als Label-Attribut zusammengestellt. Das Label-Attribut ist binär, d.h. es kennt zwei Zustände: Adjazent (Kante existiert) oder nicht

adjazent (Kante existiert nicht). Daher gilt der Klassifikator als Binärklassifikator. Nach dem Trainieren mithilfe der zusammengestellten Trainingsdatensätze wird das Klassifikationsmodell abgespeichert und so für eine spätere Klassifikation von Knotenpaaren eines zu untersuchenden Graphen zur Verfügung gestellt. Der beschriebene Ablauf lässt sich ebenfalls in einem Aktivitätsdiagramm festhalten (siehe Abbildung 4.6).

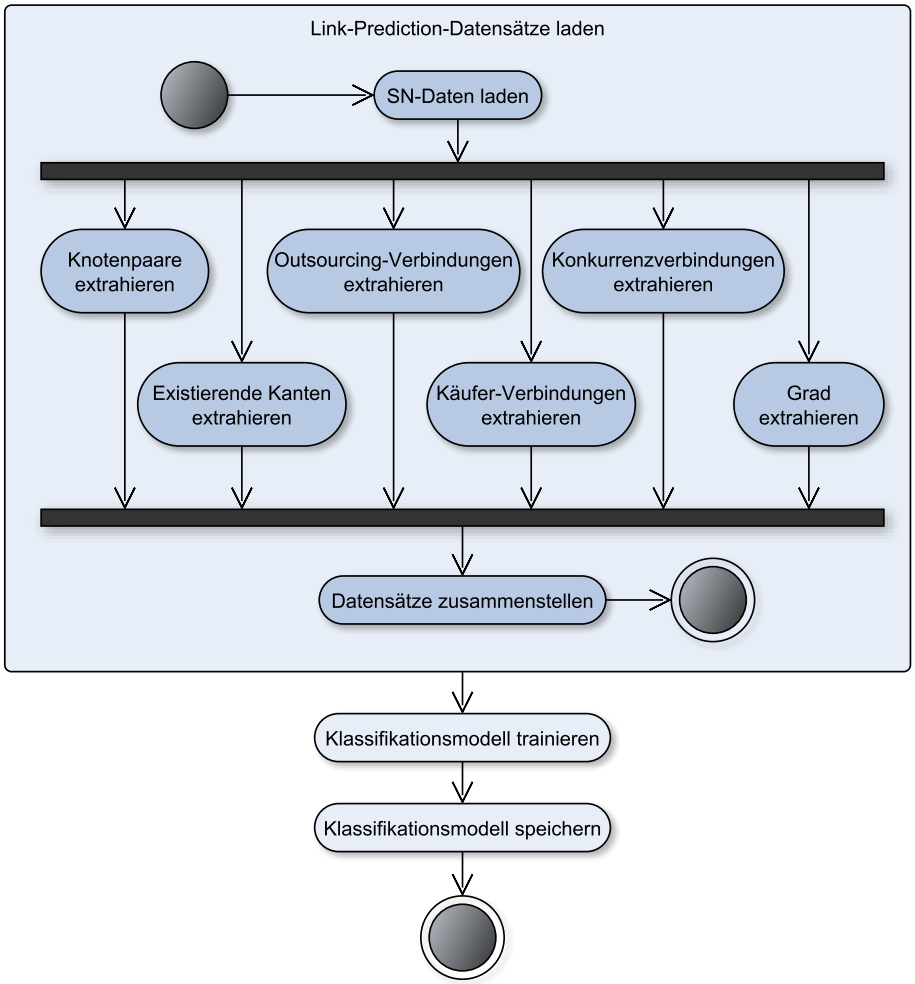

Abbildung 4.6: UML-Aktivitätsdiagramm, das den Ablauf der Erstellung eines Link-Prediction-Klassifikationsmodells beschreibt - als Teil des Anwendungsfalls „nicht-sichtbare Verbindungen aufdecken“

Die Anwendung der Link Prediction auf einen Graphen, der auf nicht-sichtbare Verbindungen zwischen Zulieferern untersucht werden soll, erfolgt in den Schritten der Merkmalsextraktion und des Zusammenstellen der Da-

tensätze analog. Weiterhin wird das im Rahmen des Trainings gespeicherte Modell geladen, um darauffolgend die Klassifikation der Knotenpaare durchzuführen. Als letzter Schritt werden aus den Ergebnissen der Link Prediction die Knotenpaare ausgewählt, die als adjazent vorhergesagt wurden und im gegebenen Graphen keine Adjazenz aufweisen. Die Kanten zwischen den ausgewählten Knotenpaaren stellen die (möglichen) nicht-sichtbaren Verbindungen zwischen Zulieferern dar. Der Ablauf der angewandten Link Prediction wird in Abbildung 4.7 als Aktivitätsdiagramm visualisiert.

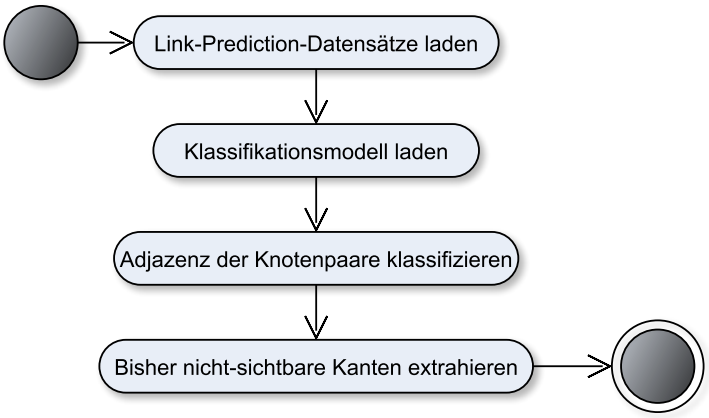

Abbildung 4.7: UML-Aktivitätsdiagramm, das den Ablauf der angewandten Link Prediction beschreibt - als Teil des Anwendungsfalls „nicht-sichtbare Verbindungen aufdecken“

Ein weiterer Schritt der Systemanalyse ist die Konzeptionierung der Benutzeroberfläche. Diese dient als Schnittstelle zwischen dem Benutzer und des Systems, das die Interpretation der Daten im Rahmen der Risikoidentifizierung ermöglicht. Dazu sind sowohl eine geeignete Darstellung als auch Interaktionen zu konzipieren, die Identifizierung der Risiken in SNs unterstützen. Die Anforderungen 3a, 3b, 3c und 3d beziehen sich dabei direkt auf das Design der Benutzeroberfläche. Weitere Designanforderungen ergeben sich aus dem Anwendungsfall „Wichtige Zulieferer erkennen“, in dem eine Visualisierung von variablen Knotengrößen festgelegt wird und dem Anwendungsfall „Struktur des SN einsehen“, der eine Möglichkeit zur Reduzierung der Graphdaten auf bspw. Unternehmensdaten definiert.
Aus der Anforderung 3a geht hervor, dass die Benutzeroberfläche das zu betrachtende SN als Graph visualisieren soll. Da die SNs eine hohe Komplexität und eine Vielzahl an Elementen aufweisen können, muss eine Ansicht, die den Graphen zeigt, den größten Teil der Benutzeroberfläche einnehmen. Der An-

wender soll die Möglichkeit erhalten, das Layout des Graphen auszuwählen, sodass hierfür ein Steuerelement in das Benutzerinterface eingebunden wird. Ebenfalls soll es möglich sein, den Graphen auf bspw. Unternehmensdaten zu reduzieren. Dazu wird ein weiteres Steuerelement für eine optionale Auswahl der Datenreduktion benötigt. Eine weitere Ansicht wird benötigt, um weitere Repräsentationen der Graphdaten darzustellen (Anforderung 3b) sowie um zusätzliche Daten, die nicht in der Graph-Ansicht eingeblendet werden, aufzulisten (Anforderung 3c). Weitere Steuerungselemente müssen in das Benutzerinterface eingebunden werden, sodass der Anwender die Möglichkeit erhält, die in Anforderung 3d genannte Datenanalyse zu starten. Die Ergebnisse der Datenanalyse sollen entweder direkt in der Graph-Ansicht oder über die aufgelisteten zusätzlichen Daten dargestellt werden. Die Ergebnisse der Zentralitätsanalyse verändern qualitativ die Knotengrößen des visualisierten Graphen in der Graph-Ansicht, sollen aber ebenfalls quantitativ als Maßzahlen den zusätzlichen Daten hinzugefügt werden, um eine genauere Interpretation der Zentralitäten durch den Anwender zu ermöglichen. Die Auswahl der zu visualisierenden Zentralitäten wird dem Anwender überlassen, sodass hierzu ebenfalls eine Steuerungskomponente konzipiert wird. Für die Datenanalyse mittels Link Prediction werden zwei Steuerelemente modelliert, jeweils eines für das Erstellen des Klassifikationsmodells und für die eigentliche Anwendung der Link Prediction. Die Ergebnisse der Link Prediction, die nicht-sichtbaren Verbindungen, werden dem Graphen in der Graph-Ansicht hinzugefügt. Um diese hervorzuheben, sollen die Kanten gesondert eingefärbt werden. Für ein besseres Verständnis des dargestellten SN sollen auch die Knoten anhand ihres Knotentyps eingefärbt werden. Dies ermöglicht eine Differenzierung bspw. von Unternehmen und Produkten oder von Zulieferern und OEM. Um dem Anwender zu signalisieren, dass ggf. Berechnungen im Hintergrund durchgeführt werden, auf deren Ergebnisse zu warten sind, sollen außerdem Statusmeldungen in der Benutzeroberfläche sichtbar werden. Im Rahmen einer Risikoidentifizierung durch einen Anwender aus dem Bereich des SCM oder der Forschung existieren möglicherweise mehr als ein SN, das untersucht werden soll. Dazu soll das System einen Wechsel der zugrundeliegenden Graphdaten ermöglichen, sodass der Anwender innerhalb der Benutzeroberfläche das gewünschten SN auswählen kann. Dazu wird eine Auswahlliste der gegebenen Graphen der Datengrundlage in die Benutzeroberfläche integriert.

Insgesamt lassen sich daraus folgende Ansichten und Komponenten des Benutzerinterfaces erarbeiten:

- Ansichten:
  - Visualisierter Graph

  - Zusätzliche Daten und Repräsentationen

- Komponenten:
  - Auswahlliste SN aus Daten
  - Auswahlliste Graph-Layout
  - Auswahlliste Datenreduktion
  - Auswahlliste Visualisierung Zentralität
  - Button Klassifikationsmodell erstellen
  - Button Link Prediction anwenden
  - Textfeld Statusmeldungen

Anhand der erarbeiteten Elemente der Benutzeroberfläche lässt sich weiterhin ein grafisches Konzept erzeugen. Bei der Aufteilung der Ansichten und der Positionierung der Komponenten werden die oben genannten Erkenntnisse berücksichtigt. Die Steuerelemente werden zusammen positioniert, um alle Interaktionen auf eine Schaltfläche zu beschränken. Eine Ausnahme stellt die Auswahl des SN dar, die sich von den anderen Steuerelementen dadurch abhebt, da sie sich auf die gesamte Datengrundlage bezieht und nicht ausschließlich auf den visualisierten Graphen. Die Auswahlliste wird oberhalb der Graph-Ansicht positioniert. Die Statusmeldungen stellen für den Benutzer kein Interaktionselement dar und werden daher am unteren Rand eingeblendet. Daraus ergibt sich das in Abbildung 4.8 gezeigte grafische Konzept.

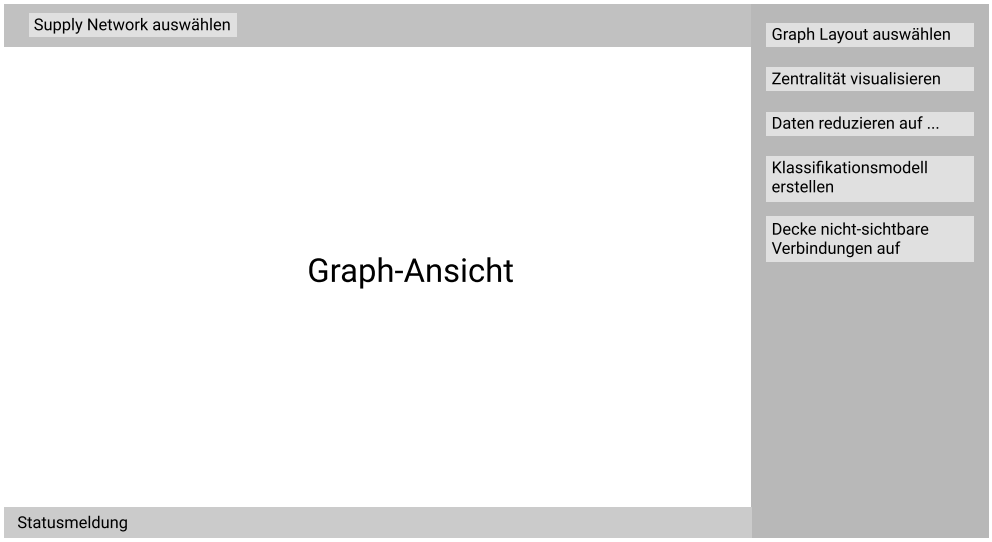

Abbildung 4.8: Grafisches Konzept der Benutzeroberfläche

4.3 Systementwurf

Auf die Systemanalyse folgt der Systementwurf, der sich mit der technischen Lösung beschäftigt. In diesem Entwicklungsschritt werden die Erkenntnisse aus der Systemanalyse in den Lösungsbereich übertragen und innerhalb der technischen Rahmenbedingungen ein lauffähiges System entworfen [Gol11, S.49].

In diesem Kapitel wird das System schrittweise entworfen, indem zunächst die Software-Architektur beschrieben wird und weiterhin anhand der Systemkomponenten und Funktionen Datenbank, Graph Mining und Benutzeroberfläche der Entwurf erläutert wird. Zum Schluss werden weitere Aspekte betrachtet, wie die Erweiterbarkeit und die Veröffentlichung als OSS.

Zu Beginn des Entwurfs ist zunächst eine geeignete Programmiersprache zu wählen, mit der das System umgesetzt werden soll. Aufgrund umfangreicher Bibliotheken und der Erfahrung des Autors fällt die Wahl auf die Programmiersprache C# mit der Software-Plattform .Net. Die Programmiersprache und die Software-Plattform zeichnen sich durch ihre Vielseitigkeit aus, sodass die Wahl die technischen Rahmenbedingungen nur unwesentlich einschränkt. In Hinblick auf eine angestrebte Erweiterbarkeit des Systems ist zu ergänzen, dass die Programmiersprache weit verbreitet ist und Gemeinsamkeiten mit der ebenfalls weit verbreiteten Programmiersprache Java teilt, sodass die Hürde für eine Weiterentwicklung gering ist. Außerdem bietet die Software-Plattform über den Package-Manager NuGet eine einfache Einbindung von externen Bibliotheken.

4.3.1 Software-Architektur

Bei der Planung der Software-Architektur wird zunächst die Form der Anwendung definiert. Die Umsetzung kann als Desktopanwendung, als mobile App oder als Webanwendung erfolgen. Da das Anwendungsszenario innerhalb von Unternehmen oder Forschungseinrichtungen stattfindet und insbesondere das Rendern von großen Graphen und die Graph-Mining-Verfahren ressourcenintensiv sein können, ist die Umsetzung als Desktopanwendung am geeignetsten.

Aus den Anforderungen (siehe Abschnitt 4.1), den beschriebenen Anwendungsfällen und erarbeiteten Ablaufschritten lassen sich die benötigten Komponenten und deren Zusammenhänge herleiten:
Für die persistente Speicherung der SNs wird eine Datenbank benötigt, die die Graphdaten, wie in Abbildung 4.2 skizziert, abbilden kann. Die eigentliche Anwendung zur Risikoidentifizierung muss auf diese Daten zugreifen, um das SN in der Benutzeroberfläche zu visualisieren (siehe 4.4). Dazu wird die

Programmstruktur in ein Service-Layer, das die Daten intern verarbeitet und bereitstellt, und ein Daten-Layer, dass die Daten aus der Datenbank abfragt, aufgeteilt. Innerhalb der Anwendung wird außerdem eine Komponente benötigt, die Machine-Learning-Fähigkeiten vereint und sowohl das Klassifikationsmodell erzeugen als auch die Klassifikation der Knotenpaare im Rahmen der Link Prediction durchführen kann. Die Komponenten werden zunächst als Blackbox in einem UML-Komponentendiagramm (siehe Abbildung 4.9) dokumentiert.

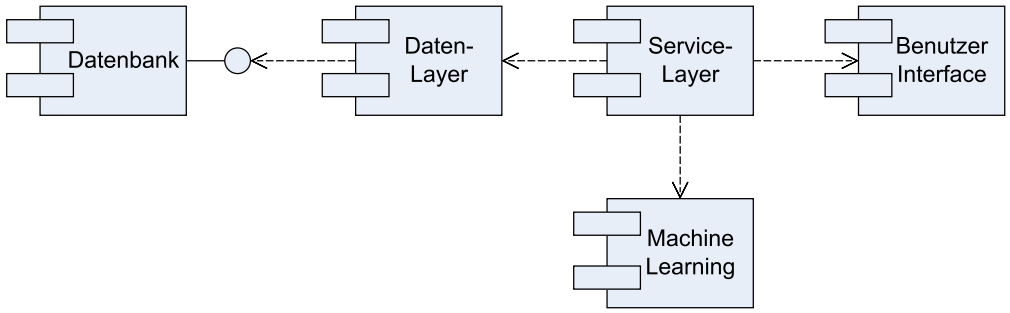

Abbildung 4.9: UML-Komponentendiagramm der Applikation zur Risikoidentifizierung

Im Folgenden werden die einzelnen Komponenten näher betrachtet und technische Lösungen entworfen. Dabei wird diskutiert, welche Techniken zum Einsatz kommen und auf welche bestehenden Lösungen aufgegriffen werden können, um die Komponenten den Anforderungen und bisherigen Erkenntnissen entsprechend umzusetzen. Sowohl das Daten-Layer als auch das Service-Layer werden nicht in eigenen Abschnitten erörtert, da sie ausschließlich die Kommunikation und die Logik der Datenverarbeitung zwischen den Komponenten Datenbank, Benutzerinterface und Machine Learning vereinen. Relevante Entwürfe zu Abläufen finden in folgenden den Abschnitten Betrachtung.

4.3.2 Datenbanksystem und Daten-Layer

Die Datenbank dient der persistenten Speicherung der SN-Graphdaten, auf die von der Applikation zugegriffen werden kann. Das SN soll dabei durch das erarbeitete Graphdatenbankschema in Abschnitt 4.2 abgebildet werden. Daher wird ein Datenbanksystem (DBS) gesucht, das dieses Schema abbilden kann. Eine weitere Anforderung an das DBS resultiert aus der Aktivität „Link-Prediction-Datensätze laden“ (siehe Abbildung 4.6), bei der im Rahmen der Merkmalsextraktion Teilgraphen aus dem SN-Graphen extrahiert

DBS	Abfragesprache	Lizenz	Einschränkung
Neo4j	Cypher	GPL v3	-
Azure Cosmos DB	Gremlin	Kommerziell	Cloud-Service
Virtuoso	Gremlin	GPL v2	-
ArangoDB	Gremlin	Apache 2.0	-
OrientDB	Gremlin	Apache 2.0	-

Tabelle 4.1: Vergleich der fünf populärsten Graphdatenbanksysteme, geordnet nach absteigender Popularität nach [21a]

werden müssen. Das zu wählende DBS soll daher eine geeignete Datenbankabfragesprache unterstützen, um diese Teilgraphen extrahieren zu können. In Hinsicht auf lizenzrechtliche oder technische Bestimmungen sollte das DBS zudem möglichst uneingeschränkt nutzbar sein.

Relationale DBS sind für diesen Anwendungsfall ungeeignet. Es ist zwar möglich, das Graphdatenbankschema abzubilden, jedoch wird keine Abfragesprache unterstützt, die die benötigten Fähigkeiten aufweist. Zur (nativen) Abbildung von Graphen eignen sich Graphdatenbanksysteme, die mächtige Abfragesprachen unterstützen und große Mengen an Daten in komplexen Strukturen speichern können. Für einen Vergleich und die Auswahl für den Anwendungsfall dieser Arbeit werden die fünf populärsten Graphdatenbanksysteme herangezogen. Diese sind in der Reihenfolge ihrer Popularität *Neo4j*, *Azure Cosmos DB*, *Virtuoso*, *ArangoDB* und *OrientDB* [21a]. Für den Vergleich werden die ausgewählten Merkmale der DBS in der Tabelle 4.1 dargestellt.

Sowohl Cypher als auch Gremlin entsprechen den Anforderungen an die Graph-Abfragesprache, wie zuvor beschrieben. GPL v3, GPL v2 und Apache 2.0 sind Open-Source-Lizenzen und schränken die Nutzung weder privat noch kommerziell ein. Die *Azure Cosmos DB* ist ausschließlich als kommerzieller Cloud-Service nutzbar und daher für diesen Anwendungsfall ungeeignet. Aus den restlichen zur Auswahl stehenden DBS fällt die Wahl auf Neo4j, da dieses zum einen am weitesten verbreitet ist und zum anderen ausschließlich auf die Darstellung von Graphstrukturen spezialisiert ist im Gegensatz zu den vier anderen Muli-Model-Datenbanksystemen. Weitere Vorteile von Neo4j sind die Bereitstellung eines Datenbank-Konnektors für C#/.Net sowie die Ausführbarkeit von Graphalgorithmen direkt auf dem DBS. Wie in der Systemanalyse (siehe Abschnitt 4.2) gefordert, ist das Graphmodell hierbei ein Labeled-Property-Graph. Neo4j bildet nur gerichtete Graphen ab, die Richtung der Kanten kann jedoch beim Abfragen ignoriert werden, sodass ein ungerichteter Graph extrahiert werden kann.

Das Daten-Layer ist die Datenzugriffsschicht und kapselt den Zugriff auf die Daten, sodass dieser von der Logik der internen Abläufe (Service-Layer) getrennt ist. Die Verbindung eines Clients mit Neo4j wird über die Netzwerkprotokolle HTTP oder Bolt hergestellt. Den Aufbau und die Aufrechterhaltung der Verbindung vereinfacht der bereits erwähnte Konnektor „Neo4jDotNetDriver“, der von Neo4j offiziell unterstützt wird. Das Daten-Layer bindet den Konnektor ein, um Abfragen zu tätigen. Ebenfalls werden innerhalb des Daten-Layers die Voraussetzungen für die Datenbankabfragen geprüft. Bspw. wird überprüft, ob benötigte Plugins in Neo4j installiert sind. Da Daten fast ausschließlich abgefragt werden, werden die Datenbankzugriffe lesend durchgeführt. Eine Ausnahme stellen die Zentralitätsmaße dar, die direkt als Attribute den Knoten hinzugefügt werden, um diese, wie in der Systemanalyse modelliert, auch quantitativ in der Benutzeroberfläche anzuzeigen. Dieses wird in den folgenden Abschnitten genauer beschrieben.

4.3.3 Graph Mining

Dieser Abschnitt beschreibt den technischen Entwurf zu den Anwendungsfällen „Wichtige Zulieferer erkennen“ und „nicht-sichtbare Verbindungen aufdecken“. Ersterer nutzt zur Identifizierung von zentralen Zulieferern die SNA mittels Zentralitätsanalyse, letzterer zum Aufdecken von Verbindungen zwischen Zulieferern die Link Prediction. Zunächst wird die Zentralitätsanalyse betrachtet. Zur Berechnung der Degree, Closeness und Betweenness Centrality eignet sich die Graph Data Science Library [21b], die Graphalgorithmen bereitstellt und frei zugänglich unter einer Open-Source-Lizenz veröffentlicht ist. Die Library wird als Plugin auf der Neo4j-Instanz installiert. Die Anweisung zur Berechnung der Zentralitätsmaße und der anschließenden Ausgabe der Ergebnisse wird in einem Cypher-Befehl im Daten-Layer festgehalten. Das Plugin verwendet zur Anwendung von Graphalgorithmen eine benutzerdefinierte Projektion des Graphen, die entweder alle oder nur ausgewählte Knoten und Kanten enthält. Soll die Berechnung der Zentralitäten ausschließlich auf Knoten der Zulieferer erfolgen, kann eine Projektion gewählt werden, die nur Zulieferer-Knoten und Kanten zwischen diesen enthält. Die Cypher-Anweisung lässt sich bei Nutzerinteraktion über den Neo4j-Konnektor an das DBS senden. Die Ausgabe der Ergebnisse erfolgt zunächst als schreibende Datenbankoperation, sodass das jeweilige Zentralitätsmaß auf ein Knotenattribut „degree“ bzw. „closeness“ oder „betweenness“ geschrieben wird. Um bspw. die Degree Centrality der Knoten des gesamten Graphen zu berechnen und den Grad anschließend auf das entsprechende Attribut zu schreiben, lautet die Cypher-Anweisung:

```
CALL gds.degree.write({
        nodeProjection: '*',
        relationshipProjection: '*',
        writeProperty: 'degree'})
```

Für den Entwurf der Link Prediction wird sich am SNLP-Verfahren sowie den hergeleiteten Abläufen aus der Systemanalyse (siehe Abbildung 4.6) orientiert. Im Rahmen der Merkmalsextraktion müssen zu jedem Knotenpaar die vier Merkmale Outsourcing-Score, Käufer-Score, Konkurrenz-Score, Grad und das Label Adjazenz bestimmt werden, um die Datensätze für die Klassifikation zu erstellen. Eine Liste aller möglichen Knotenpaare aus einem gegebenen Graphen lässt sich extrahieren, indem das kartesische Produkt aus der Menge der vorhandenen Knoten mit derselben gebildet wird. Hierbei werden Paare ausgenommen, deren Elemente beide auf den gleichen Knoten referenzieren, da keine Verbindungen von einem Zulieferer zu sich selbst gesucht sind und ein solcher Datensatz möglicherweise das Ergebnis der Link Prediction verfälscht. Eine Liste der adjazenten Knotenpaare lässt sich aus der Datenbank abfragen, indem Start- und Endknoten der vorhandenen Kanten ausgegeben werden. Die entsprechende Cypher-Anweisung lautet:

```
MATCH (a)-[]->(b) Return a,b
```

Die Merkmalsextraktion der Outsourcing-, Käufer- und Konkurrenzverbindungen ist deutlich komplexer. Dazu müssen entsprechende Teilgraphen mittels eines beschriebenen Musters (Pattern Matching) gefunden werden. Cypher eignet sich für diesen Anwendungsfall ebenfalls, indem deklarativ das zu suchende Muster und die Ausgabevariablen beschrieben werden. Dabei werden die zu extrahierenden Muster anhand der Beschreibungen der SNLP-Methode (siehe Abschnitt 3.3.2) hergeleitet. Für eine Outsourcing-Verbindung zwischen Zulieferer s1 und Zulieferer s2 wird angenommen, dass ein von s1 hergestelltes Produkt von einem von s2 hergestellten Produkt abhängig ist. Daraus ergibt sich die folgenden Cypher-Anweisung, wobei „Supplier“ und „Product“ den Labels der Zulieferer-Knoten bzw. der Produkt-Knoten entsprechen:

```
MATCH (s1:Supplier)-[]->(p1:Product)
-[]->(p2:Product)-[]-(s2:Supplier)
RETURN s1, s2
```

Das Muster der Käufer-Verbindung beschreibt die Konstellation eines Käufers, der sowohl Waren eines Verkäufers s1 als auch die (angenommen kompatiblen) Waren eines Verkäufers s2 bezieht. In diesem Fall würde eine Käufer-Verbindung im Sinne einer Absprache über Kompatibilitätseigenschaften zwischen den Verkäufern s1 und s2 existieren. Die das Muster beschreibende

Datenbankabfrage lautet:

```
MATCH (s1:Supplier)-[]->(buyer:Supplier)<-[]-(s2:Supplier)
RETURN s1, s2
```

Eine Konkurrenz-Verbindung zwischen einem Zulieferer s1 und einem Zulieferer s2 existiert, wenn ein Produkt von beiden hergestellt wird. Analog dazu ist die Cypher-Abfrage wie folgt:

```
MATCH (s1:Supplier)-[]->(product:Product)<-[]-(s2:Supplier)
RETURN s1, s2
```

Die Datenbankabfragen liefern dabei die betreffenden Knotenpaare zu jeder Übereinstimmung mit den beschriebenen Mustern. Als letztes Merkmal ist der Grad der Knoten auszulesen. Dies erfolgt analog und wie zu Beginn dieses Abschnitts beschrieben. Anschließend gilt es, aus den extrahierten Merkmalen die Datensätze zu erzeugen. Hierzu wird pro Knotenpaar ein Datensatz generiert und zu jedem Datensatz der Grad als Fließkommazahl und die Adjazenz als Wahrheitswert hinzugefügt, sowie für jede Übereinstimmung mit einem extrahierten Muster der entsprechende ganzzahlige Outsourcing-, Käufer- oder Konkurrenz-Score um 1 erhöht. Das Resultat sind Datensätze in einer Tabellenstruktur, die ein Klassifikationsmodell zum Lernen oder Vorhersagen nutzen kann.

Für die eigentliche Klassifikation wird ML.NET als Machine-Learning-Bibliothek ausgewählt. Diese verfügt über eine Reihe an Algorithmen zur binären Klassifikation, darunter das Perzeptron, logistische Regression, Entscheidungsbäume und Support Vector Machine. ML.NET bindet sich als .Net-Bibliothek optimal in das System ein und ist des Weiteren unter der MIT-Lizenz als OSS verfügbar. Möglich ist zudem das Importieren und Exportieren eines Klassifikationsmodells bspw. von oder für TensorFlow. Um das Klassifikationsmodell auszuwählen, das die besten Ergebnisse erzielt, wird das zusammengestellte Datenset in einen Test- und einen Trainingsdatensatz eingeteilt. Die verfügbaren Algorithmen trainieren ein Klassifikationsmodell auf den Trainingsdaten und testen dieses anschließend auf den Validierungsdaten. Zum Schluss werden die Ergebnisse der Modelle verglichen und das Modell mit den besten Ergebnissen ausgewählt. Der Vergleich und die Auswahl des besten Modells geschieht anhand einer ausgewählten Metrik. Das Klassifikationsmodell wird anschließend auf dem Dateisystem gespeichert und ist zum Abruf für eine anzuwendende Klassifikation verfügbar. Möchte der Nutzer nicht-sichtbare Verbindungen eines vorliegenden SN aufdecken, so werden, wie in Abbildung 4.7 beschrieben, die Merkmale von dem gegebenen Graphen extrahiert, die Datensätze zusammengestellt und das Klassifikationsmodell geladen. Der Klassifikator klassifiziert anhand

der Merkmale die Adjazenz mit einer Wahrscheinlichkeit und gibt die Ergebnisse an das Service-Layer zurück. Die Link Prediction kann jedoch nur durchgeführt werden, wenn das SN Graphdaten über Zulieferer und Produkte enthalten und diese mit den entsprechenden Knotentypen versehen sind.

Um ein Klassifikationsmodell zum Aufdecken von nicht-sichtbaren Verbindungen zwischen Zulieferern in SNs zu trainieren, ist eine geeigneter Datenbasis erforderlich. Die Daten sollten nach Möglichkeit ein vollständiges, reales SN abbilden. Daten über SNs der realen Welt finden sich in [CH02]. Die Autoren Choi und Hong skizzieren mehrere SNs aus der Automobilbranche und untersuchen anhand von Fallstudien bei Honda, Acura und DaimlerChrysler die Struktur der Netzwerke. Das abgebildete SN von Honda beschreibt die Zulieferer und ihre Produkte, die zur Herstellung der Mittelkonsole eines Honda Accords benötigt werden [CH02, S.476]. Anhand der Skizze werden Knoten und Kanten eines Graphen, der dieses Netzwerk repräsentiert, in einer Datenbank in Neo4j händisch eingepflegt. In Hinblick auf eine Evaluierung der Link Prediction und der damit verbundenen Merkmalsextraktion werden Zwischenhändler als Zuliefererknoten ohne adjazente Produktknoten abgebildet, da sonst die Gefahr besteht, dass Verbindungen zwischen Zulieferern und Zwischenhändlern fälschlicherweise als Konkurrenzverbindungen extrahiert werden, wenn beide das gleiche Produkt vertreiben. Zwischenhändler sind ausschließlich mit weiteren Zulieferer-Knoten adjazent. Dies entspricht dem in der Systemanalyse beschriebenen Graphdatenbankschema. Zur Anwendung als Trainingsdaten zum Lernen eines Klassifikators hat die Datenbasis einen sehr geringen Umfang. Zum Vergleich nutzen die Autoren der SNLP-Methode drei verschiedene SNs, deren Graphen jeweils mehr als 1000 Knoten aufweist. Aufgrund der eingeschränkten Verfügbarkeit von Daten über größere SNs wird diese Datenbasis verwendet.

Aus dem Graphen werden die Merkmale anhand der zuvor beschriebenen Methode extrahiert und eine Trainingsdatenbasis zusammengestellt. Die Datenbasis besteht aus 756 Datensätzen mit jeweils verschiedenen Knotenpaaren. Darunter weisen 66 Datensätze ein positives Label auf, d.h. die Knotenpaare sind adjazent. Für die Binärklassifikation bedeutet das Verhältnis zwischen der Anzahl an Datensätzen mit positivem und negativem Label ein starkes Ungleichgewicht der Klassen. Der größte Teil der Datensätze gehört der „Nicht-Adjazenz“-Klasse an. Im Rahmen der Link Prediction liegt das Interesse jedoch in der Minderheitenklasse mit einem positiven Adjazenz-Label. Damit wird eine Binärklassifikation problematisch. Ein Klassifikationsmodell, das Knotenpaare ausschließlich mit einem negativen Adjazenz-Label bewertet, kann eine hohe Korrektklassifizierungsrate (engl. Accuracy) erzielen, die Korrektklassifizierungsrate spiegelt jedoch nur das zugrundeliegende unausgewogene Verhältnis der Klassen wider. Eine Möglichkeit, dieser

Problematik entgegenzuwirken, ist das Sammeln von weiteren Daten. Aufgrund der eingeschränkten Verfügbarkeit von Daten über reale SNs ist diese Lösung innerhalb dieser Arbeit nicht möglich. Aus diesem Grund wird eine geeignete Metrik gewählt, mit Hilfe derer das optimale Modell ausgewählt wird. Eine bei Datensätzen mit unbalancierten Klassen geeignete Metrik ist die Receiver Operating Characteristic (ROC)-Kurve bzw. die Fläche unter der ROC-Kurve [Agg15, S. 259]. Diese wird auch zur Bewertung der Ergebnisse bei der SNLP-Methode herangezogen [Bri+18, S. 8]. Die ROC-Kurve bildet das Verhältnis zwischen der Effizienz und der Fehlerrate in einem Diagramm ab. Sie wird berechnet, indem für jeden Wert eines Modellparameters ein Punkt im Diagramm mit der Sensitivität als Ordinate und der Falsch-Positiv-Rate als Abszisse eingetragen wird. Ein optimales Modell liefert eine Kurve, die entlang der Ordinate verläuft (mit einer Falsch-Positiv-Rate von 0) und erst anschließend mit steigender Falsch-Positiv-Rate weiter verläuft. Daher gilt die Fläche unter der ROC-Kurve, Area Under the Curve (AUC), als aggregierte Metrik, um ein Modell zu bewerten. Angestrebt ist eine AUC von 1.
Eine weitere Metrik, die sich zur Bewertung von Binärklassifikationsmodellen bei unausgewogenen Daten eignet, ist das F_1-Maß [Agg15, S. 340]: Das F_1-Maß bemisst das harmonische Mittel zwischen der Genauigkeit P (engl. precision) und der Trefferquote Q (engl. recall):
$F_1 = 2 * \frac{P*R}{P+R}$
Beide Metriken sollen verwendet werden, um das optimale Klassifikationsmodell zu finden.

Dazu wird die Datenbasis mit Hilfe der ML.Net-Bibliothek in Trainings- und Validierungsdatensätze eingeteilt. Der Anteil der Validierungsdatensätze an der Gesamtheit der Datensätze beträgt 10%. Das beste Klassifikationsmodell mit einer AUC von 0,835 konnte mit Hilfe des *LightGbmBinary*-Algorithmus erzeugt werden, der einen Gradient-Boosting-Entscheidungsbaum aufbaut. Jedoch wird eine Richtig-Positiv-Rate der Validierungsdatensätze von 0 erzielt, d.h. alle Knotenpaare, die adjazent sind, wurden fälschlicherweise als nicht-adjazent vorhergesagt. Das zugehörige F_1-Maß beträgt dabei 0.201 und fällt damit deutlich niedriger aus als die F_1-Maße anderer Modelle. Aus diesem Grund werden die Modelle primär am F_1-Maß bewertet und erst sekundär nach der AUC. Das Modell mit dem besten F_1-Maß konnte durch den *FastTreeBinary*-Algorithmus erzeugt werden. Der Algorithmus basiert auf einem Gradient-Boosting-Regressionsbaum. Das Modell erzielt ein F_1-Maß von 0,643 und eine AUC von 0,824, welche nur unwesentlich geringer ist als beim Modell mit dem höchsten AUC. Dieses Klassifikationsmodell wird für den zu implementierenden Prototypen ausgewählt.

4.3.4 Benutzeroberfläche

In diesem Abschnitt wird der Entwurf der Benutzeroberfläche beschrieben. Dazu ist zunächst die Wahl eines geeigneten Toolkits zum Erstellen einer grafischen Benutzeroberfläche erforderlich. Weiterhin wird auf die Anforderungen eingegangen, die sich auf die Benutzeroberfläche und die Visualisierung des Graphen beziehen (siehe Abschnitt 4.1).

Wahl des GUI-Toolkits

Die Wahl von C# als Programmiersprache und .Net als Software-Plattform schränkt die Auswahl der zur Verfügung stehenden Benutzeroberflächen-Frameworks auf *Windows Forms*, *Windows Presentation Foundation (WPF)* und *Universal Windows Platform (UWP)* ein. Alle Frameworks sind nicht Cross-Plattform fähig, sie funktionieren ausschließlich unter dem Betriebssystem Windows. Es ist jedoch möglich über weitere Frameworks die damit erstellten Desktop-Anwendungen auf andere Betriebssysteme wie Linux oder MacOS zu portieren. Die Einschränkung auf Windows wird jedoch in Kauf genommen, da es als führendes Betriebssystem in seiner Verbreitung unter Desktop-Computern gilt. Unter den genannten Toolkits gilt WPF als am ausführlichsten dokumentiert und mit vielen verfügbaren Bibliotheken unterstützt, sodass die Benutzeroberfläche der Anwendung damit erstellt wird. Zur Gestaltung der grafischen Oberfläche kann die Beschreibungssprache Extensible Application Markup Language (XAML) benutzt werden.

Visualisierung des Graphen

Zur Visualisierung des Graphen existieren eine Vielzahl an externen Tools, die den Graph rendern und sich als Ansicht in die mittels WPF erzeugte Benutzeroberfläche integrieren lassen. Diese bestehen in der Regel aus einer Bibliothek, die die Graphdatenstruktur abbildet (z.B. Quickgraph oder Quikgraph, einem Fork von Quickgraph) und einer Bibliothek, die den Graphen visualisiert und dabei das Layout sowie das Rendering übernimmt. Für einen Vergleich werden die verfügbaren Bibliotheken gegenübergestellt und geprüft, ob sie sich für die zu entwerfende Applikation in dieser Arbeit eignen. Für die Auswahl sind folgende Faktoren relevant: Laufende Weiterentwicklung/ zuletzt aktualisiert, Lizenz, Dokumentation und ggf. Performance auf großen Graphen. Sofern die Bibliothek innerhalb der Applikation getestet wurde, wird die Performance gemessen, indem die umfassende Beispieldatenbank „Northwind“ als Graph in Neo4j geladen wird und die Renderzeit betrachtet wird. Die Ergebnisse werden in der Tabelle 4.2 festgehalten.

Name	Stand	Lizenz	Dokumentation	Performance
GraphSharp	2019	Apache 2.0	Nicht vorhanden	Nicht getestet
MSAGL	2021	MIT	Vorhanden	Schnell
GraphX for .NET	2020	Apache 2.0	Umfassend	Sehr langsam
GraphViz for .NET	2016	MIT	Vorhanden	Nicht getestet

Tabelle 4.2: Vergleich von Open-Source-Bibliotheken zur Berechnung des Layouts und Rendern von Graphen

Der Vergleich fällt zugunsten des Microsoft Automatic Graph Layout (MSAGL) aus, da die Bibliothek aktuell (Stand Oktober 2021) weiterentwickelt wird, die Lizenz keine Einschränkungen bewirkt und insbesondere eine gute Performance aufweist. Eine Dokumentation ist zwar vorhanden, diese ist jedoch nicht umfangreich. Die Bibliothek benutzt intern Quikgraph als Bibliothek zur Verwaltung der Graphdatenstruktur. MSAGL schließt ein WPF-Steuerelement ein, das in eine Ansicht in die Benutzeroberfläche integriert werden kann. Unterstützte Steuerungen sind das Orientieren im Graph mittels Scrollen und Ansicht verschieben, sowie das Verschieben von einzelnen Knoten. In dem Entwurf wird das Steuerelement in die Graph-Ansicht eingebunden. Die Software-Komponente wird jedoch durch den Autor dieser Arbeit modifiziert, um die Möglichkeit einer benutzerdefinierten Knotengröße hinzuzufügen (zur Visualisierung der Zentralitätsmaße, siehe Abschnitt 4.2). Zur Berechnung des Graph-Layouts stehen vier Layout-Algorithmen zur Verfügung:

Fast Incremental: Ein einfacher Layout-Algorithmus, der adjazente Knoten möglichst nebeneinander positioniert und ggf. Überlappungen auflöst.

Multidimensional Scaling (MDS): MDS bezeichnet Methoden der Dimensionsreduktion, um gegebene Unterschieden von Elementen als wenigdimensionale Distanzen darzustellen. Das Erzeugen des Graph-Layouts mittels MDS zur Positionierung der Knoten nutzt die graphentheoretische Distanz der Knoten [KB13, S.55].

Sugiyama: Die Sugiyama-Methode zeichnet einen Graphen hierarchisch, in dem jeder Knoten einer Schicht zugeordnet ist. Dabei soll jede Kante von einer Schicht in die nächst höhere (oder niedrigere) reichen. [STT81]

Ranking: Ein Layout-Algorithmus, der die Knoten anhand seines Page-Ranks (siehe 2.3) positioniert.

Die Auswahl der Layout-Algorithmen kann über die Benutzeroberfläche mittels einer Combobox getätigt werden. Die Eingabe der Daten, die zur Verbindung mit der Datenbank erforderlich sind, wird in ein weiteres Fenster ausgelagert. So muss die Benutzeroberfläche nicht mit mehr Steuerelementen ausgestattet werden und die Übersicht wird beibehalten. Beim Verbinden mit der Datenbank werden im Fehlerfall aussagekräftige Fehlermeldungen angezeigt.

4.3.5 Erweiterbarkeit und Open Source

In diesem Abschnitt wird die Erweiterbarkeit der Funktionen in der Applikation und die Veröffentlichung als OSS im Rahmen des Systementwurfs beschrieben. Der Entwurf sieht gekapselte, modulare Programmteile vor, die eine Orientierung im Programmcode ermöglichen. Je nach Anwendungszweck lassen sich die Komponenten modifizieren, um die Applikation um die gewünschte Funktionalität zu erweitern. Eine Anpassbarkeit der Programmteile auf ein gegebenes Anwendungsszenario wird auch dadurch gewährleistet, dass eine Konfigurationsdatei mit einzustellenden Parametern beim Start der Applikation eingelesen wird. Diese sind die Namen der Zulieferer- bzw. Produkt-Knotentypen sowie das Zeitintervall zum Lernen des Klassifikationsmodells. Bei der Programmierung werden möglichst beschreibende Namen für Variablen, Methoden und Klassen benutzt und der Quelltext mit Kommentaren versehen, sodass sich die Einarbeitung in den Programmcode für eine etwaige Weiterentwicklung einfach gestaltet. Eine umfassende Dokumentation wird zusätzlich zu dem bereitgestellten Programmcode publiziert.

Die Applikation zur Risikoidentifizierung soll weiterhin als OSS veröffentlicht werden. Hierzu ist eine geeignete OSS-Lizenz zu wählen unter der die Anwendung veröffentlicht wird. Dabei sollte die Lizenz die Nutzung und Weiterentwicklung nach Möglichkeit nicht einschränken und ebenfalls geringe Hürden zur Anwendung in kommerziellen Umfeldern aufweisen. Eine Copyleft-Klausel wäre dem hinderlich, wenn jegliche Veränderung ebenfalls unter die gleiche Lizenz gestellt werden muss. Aus diesem Grund werden sowohl ausschließlich Software-Pakete und -Bibliotheken eingebunden, die unter freizügigen Lizenzen stehen, als auch die zu implementierende Applikation dieser Arbeit mit einer Lizenz ohne Copyleft versehen. Als Lizenz wird die MIT-Lizenz gewählt.

Kapitel 5

Evaluierung und Fazit

In diesem Kapitel wird die prototypische Umsetzung vorgestellt und der Prototyp anhand der in Abschnitt 4.1 spezifizierten Anforderungen evaluiert. Das Kapitel endet mit einem Fazit, dass die Ergebnisse der Evaluierung zusammenfasst.

5.1 Prototypische Umsetzung

Anhand der Erkenntnisse aus der Systemanalyse (Abschnitt 4.2) und dem Systementwurf (Abschnitt 4.3) wird ein Prototyp der auf Graph Mining basierenden Applikation zur Identifikation von SCs implementiert. Der Prototyp startet mit einem Fenster, das die Verbindung mit einer Graphdatenbankinstanz aufbaut. Dazu sind die erforderlichen Daten durch den Anwender einzugeben. Das Fenster ist in Abbildung 5.1 dargestellt. Ein Betätigen des Buttons *Connect* startet den Verbindungsversuch unter der angegebenen Uniform Resource Locator (URL) und mit den eingegebenen Anmeldedaten. Das darunterliegende Textfeld zeigt den Status der Verbindung oder Fehlermeldungen bei aufgetretenen Fehlern an. Ist der Verbindungsversuch erfolgreich, öffnet sich das Hauptfenster der Applikation (siehe Abbildung 5.2). Hier wird zunächst die Datenbank aus Neo4j geladen, die als Standard eingestellt ist, und der Graph initial visualisiert. Auf der rechten Seite findet sich die Ansicht der Steuerelemente und der zusätzlichen Daten zu Elementen des Graphen. Als Graph-Layout ist MDS ausgewählt. In der Abbildung 5.2 ist ein Produkt in der Benutzeroberfläche ausgewählt, sodass dessen Daten aufgelistet werden. Die Zentralitätsmaßzahlen „degree“, „closeness“ und „betweenness“ finden sich ebenfalls darunter. Die zwei Buttons *Start Link Prediction Training* und *Predict not-visible Links* bieten dem Anwender die Möglichkeit ein Klassifikationsmodell zur Link Prediction anhand des ge-

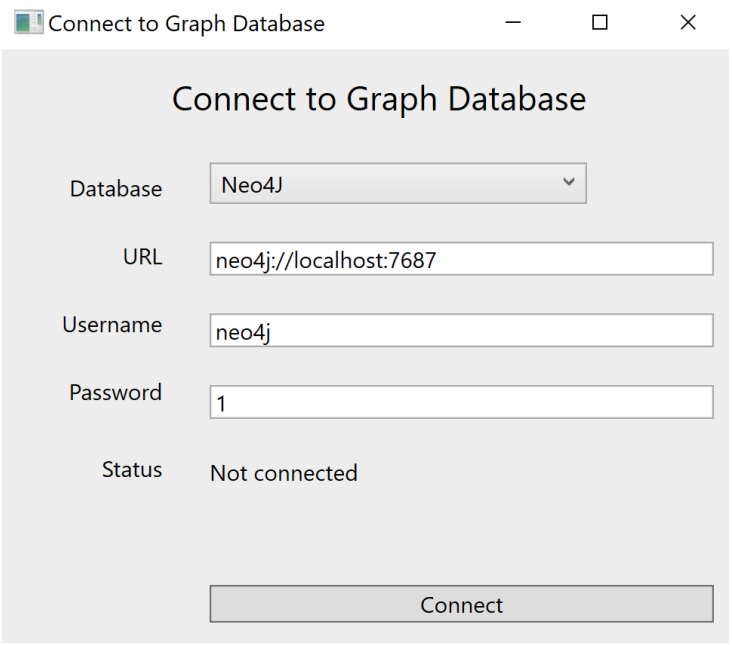

Abbildung 5.1: Screenshot des Startfensters zum Verbinden mit der Datenbankinstanz

ladenen Graphen zu erstellen sowie nicht-sichtbare Verbindung durch Link Prediction mit einem zuvor gespeicherten Klassifikationsmodell aufzudecken. Ein Textfeld in der Statusleiste in der unteren linken Ecke der Benutzeroberfläche zeigt Status- und Fehlermeldungen der Applikation an.

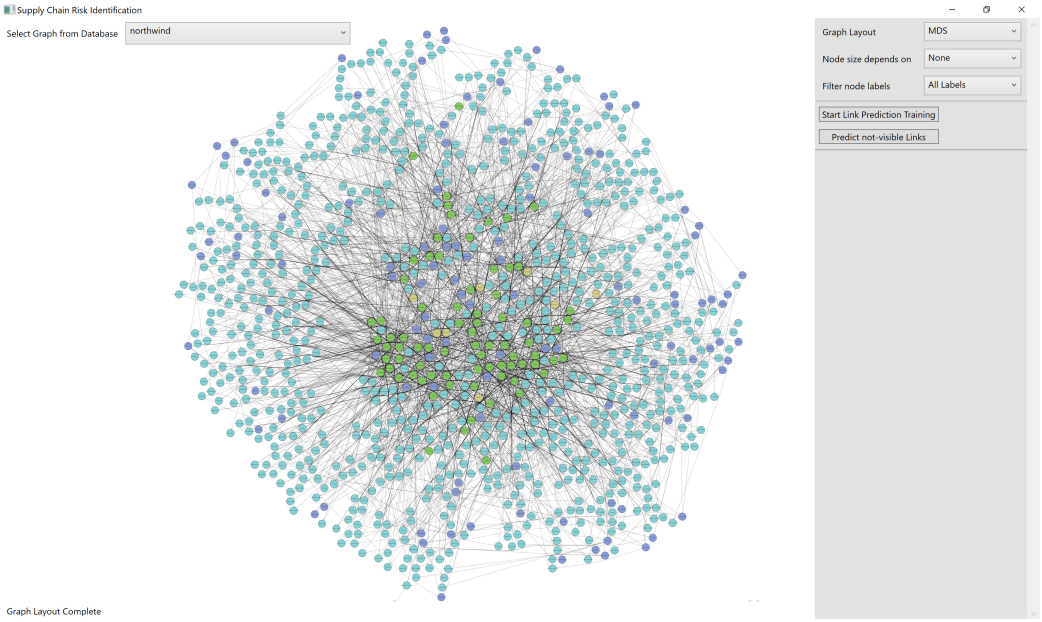

Abbildung 5.2: Screenshot des Hauptfensters der Applikation zur Risikoidentifizierung. In der Graphansicht wird der Graph der Northwind-Beispieldatenbank visualisiert.

5.2 Evaluierung des Prototypen anhand der Anforderungen

In diesem Abschnitt wird der Prototyp anhand der in Abschnitt 4.1 aufgestellten Anforderungen evaluiert. Dabei kommen die definierten Akzeptanzkriterien zum Einsatz.

Um die umgesetzten Methoden der Risikoidentifizierung auf Funktionalität und Anwendbarkeit zu prüfen, werden öffentliche Daten zu realen SNs aus der Literatur verwendet und in der Graphdatenbank gespeichert. Eine öffentlich verfügbare Datenbasis zu dem SN von Honda wurde bereits in Abschnitt 4.3.3 vorgestellt. Eine weitere Datenbasis, die im Rahmen der Evaluierung verwendet wird, wurde von Willems veröffentlicht und beschreibt 38 mehrstufige SCs aus der realen Welt [Wil08]. Die anonymisierten Daten stammen von Unternehmen, die dem Autor im Rahmen seiner Studien zugesendet wurden. Sie sind frei verfügbar und eine Nutzung ist vom Autor gestattet. Aus dieser Datenbasis werden beispielhaft die Daten der SC 7, die eine Lieferkette eines Maschinenbauunternehmens beschreiben, extrahiert und in Neo4j gespeichert. Die Datensätze entsprechen nicht dem definierten Graphdatenbankschema, sodass eine Link Prediction damit nicht möglich ist.

Alle weiteren Funktionen der Applikation lassen sich mit diesen Daten testen.

Im Folgenden wird untersucht, ob der implementierte Prototyp die Akzeptanzkriterien der Anforderungen erfüllt. Die geforderte Funktionalität wird ggf. mit Screenshots nachgewiesen.

1 - Das System muss fähig sein, zentrale Knoten in einem SN zu erkennen Der Prototyp ermöglicht die Berechnung der Zentralitätsmaße *Degree Centrality*, *Closeness Centrality* und *Betweenness Centrality* direkt auf der Graphdatenbank. Dabei ist es möglich, die Zentralitätsanalyse auf dem gegebenen Graphen oder auf einem Teilgraphen, der ausschließlich einen Knotentyp enthält, durchzuführen. Die Zentralitätsmaße werden für jeden Knoten berechnet, als Attribute des Knoten abgespeichert und zur Visualisierung in der Benutzeroberfläche in die Applikation geladen. Eine beispielhafte Visualisierung ist in Abbildung 5.3 dargestellt.

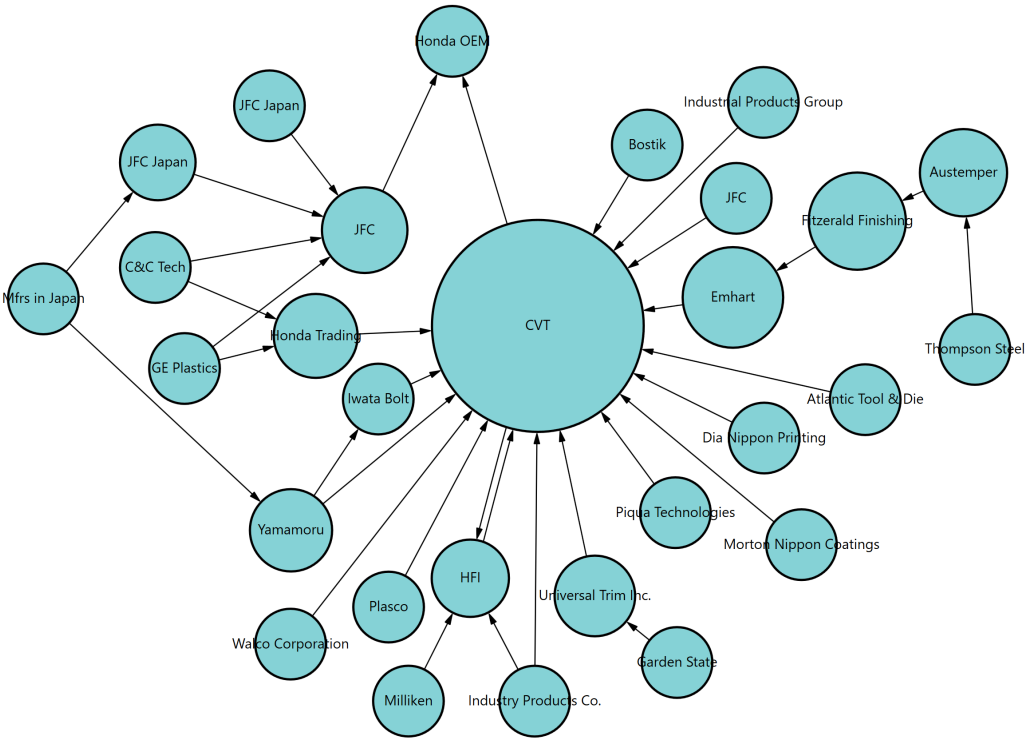

Abbildung 5.3: Visualisierung des Graphen, der die Zulieferer des Honda-SN zeigt, durch den Prototypen. Die Knotengrößen werden in Abhängigkeit von der berechneten Betweenness-Centrality visualisiert.

Ein Anwender kann zentrale Knoten im Sinne der jeweiligen Zentralitäten aus der Graphansicht erfassen, wenn die Darstellung der Knotengröße

in Abhängigkeit der Zentralität ausgewählt ist. Am Beispiel der Abbildung 5.3 weist der Zuliefererknoten *CVT* die höchste *Betweenness Centrality* der Knoten des Graphen auf. Durch seine Knotengröße kann ein Anwender diesen Knoten als zentralen Zulieferer identifizieren. Die Anforderung gilt hiermit als erfüllt.

2 - Das System muss fähig sein, nicht-sichtbare Verbindungen zwischen Zulieferern in einem SN aufzudecken Um zu überprüfen, ob das Akzeptanzkriterium dieser Anforderung erfüllt wird, wird die Link Prediction des Prototyps an einem unvollständigen SN getestet. Als Validierungsdatenbasis für die Link Prediction dient der Honda-SN-Graph, aus dem fünf gerichtete Kanten zwischen Zulieferern entfernt werden. Dies entspricht ca. 5% aller Kanten bzw. ca. 15% der Kanten zwischen Zulieferer-Knoten. Die probeweise Durchführung der Link Prediction soll die fehlenden Verbindungen zwischen den Zulieferern aufdecken. Die Merkmale werden, wie in Abschnitt 4.3.3 beschrieben, erfolgreich aus dem Graphen extrahiert und die Datensätze für den Binärklassifikator erzeugt. Nach Betätigen des Buttons „Predict not-visible Links“ ist die Vorhersage der Verbindungen zwischen den Zuliefererknoten in unter einer Sekunde beendet und der visualisierte Graph mit den zusätzlich Verbindungen aktualisiert. Die aufgedeckten Verbindungen sind ungerichtete Kanten und werden in der Graphansicht als grüne Doppelpfeile dargestellt und mit einer Beschriftung versehen (siehe Abbildung 5.4). Die Auswertung der neu hinzugefügten Kanten in den Graphen zeigt: Die fehlenden Verbindungen werden korrekt vorhergesagt, dazu weitere drei Verbindungen, die bisher nicht bekannt waren. Eine Bewertung der drei weiteren vorhergesagten Verbindungen ist aufgrund des fehlenden Wissens über die realen Ausmaße des SN nur bedingt möglich, da sie nicht-sichtbare Verbindungen aufzeigen könnten, die den Autoren Choi und Hong möglicherweise nicht bekannt waren. Es wird jedoch die Annahme getroffen, dass die Datenbasis ein vollständiges Bild des SN liefert, sodass die drei zusätzlichen Verbindungen als falsch klassifiziert gelten. Da mit der Implementierung das Akzeptanzkriterium eingehalten wurde, gilt die Anforderung als erfüllt, auch wenn die Ergebnisse der Link Prediction nicht vollständig korrekt sind.

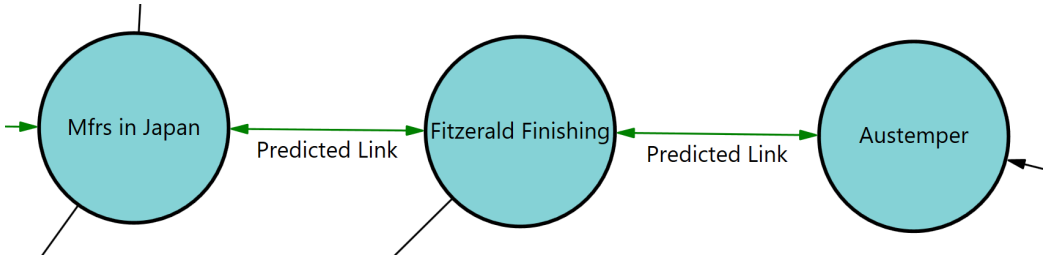

Abbildung 5.4: Ausschnitt des visualisierten Graphen mit vorhergesagten Verbindungen zwischen Zulieferern des Honda-SN. Die vorhergesagten Verbindungen werden durch grüne Doppelpfeile mit der Beschriftung „Predicted Link“ dargestellt.

3a - Das System soll fähig sein, die Graphenstruktur eines SN zu visualisieren Der Prototyp ist fähig, Daten eines SN als Graph zu visualisieren und mittels verschiedenen Graph-Layouts die Struktur des SN abzubilden. Für eine Validierung des Akzeptanzkriteriums werden die Daten von Willems zu der mehrstufigen SC getestet. Dabei sollen die jeweiligen Stufen der SC in der visualisierten Struktur sichtbar werden. Dies lässt sich mit der Auswahl des Sugiyama-Layouts (siehe Abschnitt 4.3.4) realisieren, sodass ein Benutzer ohne weiteres Eingreifen in das Layout eine Übersicht über die Struktur erhält. Die Struktur wird weiterhin mit Informationen über die Knotentypen angereichert, indem verschiedene Knotentypen unterschiedlich eingefärbt werden. Die Visualisierung in der Graphansicht ist in Abbildung A.1 dargestellt. Damit wird die Anforderung erfüllt.

3b - Das System soll fähig sein, mehrfache Ansichten in einem integrierten Interface bereitzustellen Diese Anforderung wird durch den Prototypen nur teilweise erfüllt. Die Applikation ermöglicht die Darstellung mehrfacher Ansichten in dem Benutzerinterface z.B. über die Auswahl des Graph-Layouts. Um auch große Graphen zu visualisieren, wird die Graphansicht jedoch nicht aufgeteilt und damit nicht in einem integrierten Interface bereitgestellt. Bezogen auf verschiedene Repräsentationen der Zentralitätsmaße kann dennoch das Akzeptanzkriterium erfüllt werden. Hierbei werden die Zentralitätsmaße sowohl qualitativ in Form von variablen Knotengrößen als auch quantitativ in der Ansicht der zusätzlichen Daten dargestellt.

3c - Das System soll dem Anwender die Möglichkeit bieten, interaktiv das SN zu untersuchen Innerhalb der Benutzeroberfläche des Prototypen ist es möglich, mit dem Cursor auf einen Knoten zu zeigen. Dabei werden zu dem Knoten zugehörigen Daten in der rechten Ansichten aufgelis-

tet. Abbildung A.2 zeigt ein Beispiel der zugehörigen Daten. Die Anforderung wird hiermit erfüllt.

3d - Das System muss fähig sein, analytische Fähigkeiten einzubinden Der Prototyp bietet dem Anwender über die Benutzeroberfläche die Möglichkeit, eine Datenanalyse sowohl in Form einer Zentralitätsanalyse als auch in Form der Link Prediction zu starten. Die Zentralitätsanalyse wird implizit bei der Visualisierung des ausgewählten Graphen gestartet. Die Einbindung dieser analytischen Fähigkeiten entspricht dem Akzeptanzkriterium der Anforderung.

4 - Das System soll eine Erweiterbarkeit der Funktionalität gewährleisten Die Erweiterbarkeit der Funktionalität des Prototypen wird auf mehreren Ebenen realisiert. Aus der Perspektive der Softwareentwicklung ist die Architektur der Anwendung nach Komponenten getrennt, sodass eine Modifizierung ausschließlich Änderungen an der betreffenden Komponente erfordert. Der zugrundeliegende Quellcode ist weiterhin mit Kommentaren versehen, um die Einarbeitung in das Programm zu vereinfachen. Im Daten-Layer sind zusätzliche Cypher-Abfragen implementiert, die für zukünftige Funktionen genutzt werden können. Mittels Dependency Injection lassen sich mit wenig Aufwand Komponenten austauschen, ohne tiefgreifende Änderungen durchzuführen. Aus der Perspektive eines Anwenders ohne Programmiererfahrung bietet der Prototyp die Möglichkeit, Funktionalitäten anzupassen. Das Klassifikationsmodell der Link Prediction lässt sich über den Button auf anderen, möglicherweise umfassenderen, SN-Graphen trainieren, um bessere Ergebnisse zu erzielen. Dabei werden die voreingestellten Parameter der Klassifizierung verwendet. Außerdem wird die Konfigurationsdatei appsettings.json bereitgestellt, die von der Applikation eingelesen wird. Hier lasen sich die Namen der für die Link Prediction erforderlichen Knotentypen ändern und eine maximale Dauer des Trainingsprozesses des Klassifikationsmodells einstellen. Damit ist Anforderung 4 erfüllt.

5 - Das System soll als OSS veröffentlicht werden Der Quelltext des Prototypen wird auf der Plattform GitHub veröffentlicht und mit der freizügigen OSS-Lizenz MIT lizenziert. Damit ist jegliche Nutzung, Vervielfältigung und Änderung an dem Quelltext und dem Programm gestattet. Da die Lizenz keine Copyleft-Klausel enthält, können Teile des Quelltextes auch in weiteren Programmen genutzt werden, ohne dass dessen Lizenzierung in irgendeiner Form eingeschränkt wird. Dies entspricht dem Akzeptanzkriterium der Anforderung.

5.3 Fazit

Dieser Abschnitt führt die Ergebnisse der Arbeit und deren Implikationen zusammen. Hierzu wird geprüft, ob die Ziele dieser Arbeit im Rahmen der Aufgabenstellung erfüllt wurden. Das primäre Ziel ist die Implementierung einer auf Graph Mining basierenden Open-Source-Applikation zur Identifizierung von Risiko-Objekten in SCs. Teilziele bestehen in der Aufstellung von Anforderungen an die zu entwickelnde Anwendung, dem Planen einer geeigneten Software-Architektur und die Implementierung eines Prototypen, der die aufgestellten Anforderungen erfüllt. Das primäre Ziel gilt als erfüllt, sofern die Teilziele entsprechend umgesetzt werden.

Zur Erfüllung des ersten Teilziels wurden fünf Anforderungen an die zu entwickelnde Applikation aufgestellt, darunter eine Anforderung, die noch in vier Teilanforderungen aufgeteilt wurde. Dabei wurden sowohl Aspekte der Funktionalität der Anwendung als auch nicht-funktionale Aspekte, die die Qualität der Umsetzung betreffen, berücksichtigt. Durch die Dokumentation nach Regeln des Requirements Engineering konnten Ursprung, Beschreibung, Typ und Akzeptanzkriterien nachvollzogen werden. Die auf die Anforderungsspezifikation folgende Systemanalyse und der Systementwurf haben gezeigt, dass die Anforderungen bei der fachlichen und technischen Lösung umgesetzt werden können.

Das zweite Teilziel, die Planung einer geeigneten Software-Architektur, umfasst die Konzeptionierung von Komponenten und deren Zusammenwirken. Im Rahmen dieser Arbeit konnte eine Architektur erarbeitet werden, die modular aufgebaut ist und deren Komponenten nach Funktion getrennt sind. Dabei wurden die Komponenten zunächst als Blackbox modelliert und weiterhin spezifiziert, sodass eine technische Lösung die Komponenten und deren Zusammenwirken konkretisiert. Die erfolgreiche Implementierung des Prototypen nach den Erkenntnissen des Systementwurfs hat gezeigt, dass die Architektur umsetzbar ist.

Das dritte Teilziel ist die Implementierung eines Prototypen und dessen Evaluierung anhand der aufgestellten Anforderungen. Dazu wurde der Prototyp, dem Systementwurf entsprechend, als Windows-Desktopapplikation in der Programmiersprache C# implementiert. Die Evaluierung zeigt, dass der Prototyp alle funktionalen Anforderungen an die Anwendung erfüllt. Darunter wird Anforderung 2 „Das System muss fähig sein, nicht-sichtbare Verbindungen zwischen Zulieferern in einem SN aufzudecken“ zwar erfüllt, dennoch ist das Ergebnis der testweisen Durchführung der Link Prediction fehlerhaft. Hierbei werden neben den richtig vorhergesagten Verbindungen auch Verbindungen aufgedeckt, die nicht der Datenbasis entsprechen. Die fehlerbehafteten Ergebnisse lassen sich durch den geringen Umfang der Trainingsda-

tenbasis erklären. Das Klassifikationsmodell wurde mit 756 Datensätzen aus einem Graphen mit 67 Knoten trainiert, wobei ausschließlich 66 Datensätze ein positives Label aufweisen. Sowohl die unausgewogene Klassenverteilung als auch die geringe Anzahl an Datensätzen erschwerten das Lernen eines binären Klassifikationsmodells. In Ermangelung an weiteren Datensätzen über reale SNs konnte die Link Prediction zudem ausschließlich an einem SN validiert werden, sodass die Qualität der Ergebnisse nur bedingt auf andere Graphen übertragbar ist.
Die nicht-funktionalen Anforderungen 3c, 4 und 5 werden ebenfalls erfüllt. Anforderung 3b „Das System soll fähig sein, mehrfache Ansichten in einem integrierten Interface bereitzustellen“ kann nur teilweise erfüllt werden. Zwar können die Ergebnisse der Zentralitätsanalyse in mehreren Ansichten innerhalb eines Benutzerinterfaces auf unterschiedliche Weise repräsentiert werden. Die verschiedene Repräsentation der Graphdaten in einem Interface ließ sich jedoch nicht umsetzen, ohne die Qualität der Darstellung zu reduzieren. Aus diesem Grund wurde die Anforderung nicht gänzlich erfüllt. Da der Prototyp auf den Erkenntnissen aus Kapitel 4 basiert und anhand derer umgesetzt wurde validiert die Evaluierung des Prototyps zudem die Modellierung im Rahmen der Systemanalyse und die getroffenen Entscheidungen beim Systementwurf. Die umgesetzten Anforderungen 1, 2 und 3 basieren direkt auf den in Abschnitt 3.3 beschriebenen Methoden zur Risikoidentifizierung in SCs. Der Prototyp setzt die Risikoidentifizierungsmethoden Netzwerkvisualisierung, SNLP und Zentralitätsanalyse um, sodass die Evaluierung ebenfalls die Eignung der Applikation zur Risikoidentifizierung und damit die Erfüllung der Hauptaufgabenstellung belegt. Auch die weiteren Anforderungen 4 und 5, die sich aus der Aufgabenstellung dieser Arbeit ergeben, werden umgesetzt. Damit wurde die Zielsetzung dieser Arbeit erreicht.

Kapitel 6

Zusammenfassung und Ausblick

In dieser Masterarbeit wurde eine Open-Source-Applikation zur Risikoidentifizierung von Supply Chains, die auf Verfahren des Graph Minings basiert, implementiert. Die Applikation soll in Unternehmen und in der Forschung Anwendung finden, um die Identifizierung von Störpotenzialen und Risiko-Objekten in komplexen Liefernetzwerken zu unterstützen. Dazu wurden Graph-Mining-Methoden eingesetzt, die auf der Basis der Graphen- bzw. Netzwerkstruktur einer SC operieren.

Die Entwicklung der Applikation erfolgte nach dem Wasserfallmodell. Hierzu wurden zunächst Anforderungen an die Applikation aufgestellt. Die im Vorfeld der Grundlagenforschung ermittelten Methoden der Risikoidentifizierung wurden hierzu aufgegriffen und deren Nutzung in der Anwendung als funktionale Anforderungen spezifiziert. Anforderung 1 beschreibt die Identifizierung von zentralen Knoten als Risiko-Objekte mittels der Zentralitätsanalyse. Das Aufdecken von nicht-sichtbaren Verbindungen zwischen Zulieferern in einem Supply Network, von denen ebenfalls ein Risikopotenzial ausgeht, wurde in Anforderung 2 dokumentiert. Anforderung 3 definiert die Umsetzung der Risikoidentifizierung mittels Netzwerkvisualisierung und die Benutzerschnittstelle. Die Anforderung wurde in vier Teilanforderungen aufgeteilt, die die Umsetzung der Aspekte Visualisierung, Datenrepräsentationen, Nutzerinteraktion und Analysefähigkeiten bestimmen. Aus der Aufgabenstellung wurden zwei weitere, nicht-funktionale Anforderungen an die Applikation übernommen. Darunter ist die Erweiterbarkeit der Funktionalität in Anforderung 4 sowie die Veröffentlichung der Applikation als OSS in Anforderung 5 festgehalten. Die Dokumentation der Anforderungen erfolgte nach den Konventionen des Requirements Engineering, wobei für jede Anforderung Name, Typ, Beschreibung, Begründung und Akzeptanzkriterium definiert wurden.

Aufbauend auf den spezifizierten Anforderungen wurde im Rahmen der

Systemanalyse ein Fachkonzept erarbeitet. Dabei wurde sich der Objektorientierten Analyse bedient, um ein Graphdatenbankschema als statisches Modell auszuarbeiten und um ein dynamisches Modell zu erstellen, das das Verhalten der Applikation definiert. Aus der Anforderungsanalyse konnten Anwendungsfälle erarbeitet werden, deren Verhalten weiterhin konkretisiert und mittels Aktivitätsdiagrammen festgehalten wurde. Darüber hinaus wurde eine Benutzeroberfläche konzeptioniert, die eine Graphansicht, eine Ansicht für zusätzliche Daten und Steuerelemente einschließt.

Die technische Lösung des Fachkonzepts wurde im Systementwurf erarbeitet. Dabei wurde zunächst eine Software-Architektur einer Desktopapplikation entworfen, die aus mehreren Komponenten besteht, darunter eine Benutzeroberfläche, eine Machine-Learning-Komponente und ein Daten-Layer, das auf eine Datenbank zugreift. Der technische Entwurf der Benutzeroberfläche nutzt das UI-Framework Windows Presentation Foundation. Die Graphvisualisierung innerhalb der Benutzeroberfläche konnte mit der Software-Bibliothek Microsoft Automatic Graph Layout realisiert werden, die in einem Vergleich von Graph-Bibliotheken die besten Ergebnisse erzielte. Für die Wahl des Datenbanksystems, das das Graphdatenbankschema eines Supply Networks abbilden kann und eine geeignete Abfragesprache besitzt, wurden die fünf verbreitetsten Graphdatenbanksysteme verglichen. Ausgewählt wurde Neo4j. Die Zentralitätsanalyse wird direkt auf Neo4j mit Hilfe des Graph-Data-Science-Plugins durchgeführt und die Ergebnisse als Knotenattribute in der Datenbank gespeichert. Über das Data Layer werden zudem Cypher-Abfragen an die Datenbank gesendet, um die für die Link Prediction benötigten Merkmale zu extrahieren. Die Machine-Learning-Komponente basiert auf der Bibliothek ML.Net, mit der ein binäres Klassifikationsmodell für die Link Prediction erzeugt wird. Die Trainingsdaten wurden aus einem Graphen extrahiert und zusammengestellt, der ein reales Supply Network abbildet. Für die Wahl des optimalen Klassifikationsmodells werden mehrere Klassifikatoren trainiert und validiert. Als Metriken zur Bewertung der Klassifikationsmodelle wurden das F_1-Maß und die AUC verwendet. Das beste Ergebnis erzielt der FastTreeBinary-Algorithmus, der einen Gradient-Boosting-Regressionsbaum aufbaut.

Auf den Entwurf und das erstellte Klassifikationsmodell aufbauend, wurde ein Prototyp entwickelt und unter der MIT-Lizenz veröffentlicht. Anschließend wurde der Prototyp anhand der aufgestellten Anforderungen evaluiert, wobei die Akzeptanzkriterien der Anforderungen berücksichtigt wurden. Die Evaluierung zeigt, dass alle Anforderungen, bis auf eine nicht-funktionale Anforderung, die das Benutzerinterface betrifft, gänzlich erfüllt wurden. Aus der Evaluierung der Link Prediction ging hervor, dass die Ergebnisse zum Teil fehlerhaft sind, die auf eine unausgewogene und zu geringe Trainingsda-

tenbasis zurückzuführen ist. Außerdem konnte die Link Prediction aufgrund Datenmangels nicht an weiteren Datensätzen getestet werden, sodass die Ergebnisse nur bedingt übertragbar sind.

Insgesamt wurde die Zielsetzung dieser Arbeit erreicht. Dabei kombiniert die implementierte Applikation mehrere, auf Graph-Mining-basierende Methoden der Risikoidentifizierung und setzt diese in einer Open-Source-Software um. Damit steht die Applikation Anwendern aus Unternehmen und der Forschung zur Verfügung und ermöglicht durch ihre MIT-Lizenz jegliche Nutzung, Änderung und Verbreitung. In Hinblick auf die geringe Trainingsdatenbasis der Link Prediction kann es sinnvoll sein, ein Klassifikationsmodell anhand eines größeren Liefernetzwerk zu erzeugen. Zukünftige Forschung im Rahmen des Graph Minings könnte an die eingeschränkte Verfügbarkeit von Daten über reale Supply Chains anknüpfen, indem mittels generativen Graphmodellen synthetische Netzwerke erzeugt werden, die wiederum zum Lernen einer optimalen Link Prediction verwendet werden können. Auch die Zentralitätsanalyse bietet weiteren Forschungsmöglichkeiten, indem weitere oder ähnliche Maße der Zentralität wie der PageRank zur Identifizierung potenzieller Ausfallstellen untersucht werden. Eine weiterführende Evaluierung des Prototypen in der Unternehmenspraxis könnte außerdem zu neuen Erkenntnissen zu der entwickelten Anwendung und den eingesetzten Methoden führen. Möglicherweise ist die Anwendung der umgesetzten Methoden der Risikoidentifizierung mit nicht-analytischen Methoden kombinierbar, um eine Vielzahl an Informationen für eine effektive Risikoidentifizierung zu gewinnen.

Literaturverzeichnis

[21a] *DB-Engines Ranking von Graph DBMS*. solid IT gmbh. Oktober 2021. URL: https://db-engines.com/de/ranking/graph+dbms (besucht am 14. 10. 2021).

[21b] *The Neo4j Graph Data Science Library Manual*. Neo4j Inc. 2021. URL: https://neo4j.com/docs/graph-data-science/current/ (besucht am 15. 10. 2021).

[Agg15] Charu C. Aggarwal. *Data Mining. The textbook*. eng. Aggarwal, Charu C. (VerfasserIn). Cham: Springer International Publishing, 2015. 734 S. ISBN: 978-3-319-14141-1. DOI: 10.1007/978-3-319-14142-8.

[Aig15] Martin Aigner. *Graphentheorie. Eine Einführung aus dem 4-Farben Problem*. ger. 2., überarbeitete Auflage. Lehrbuch. Aigner, Martin (VerfasserIn). Wiesbaden: Springer Spektrum, 2015. 196 S. ISBN: 978-3-658-10322-4. DOI: 10.1007/978-3-658-10323-1.

[ANR09] Volker Altenähr, Tristan Nguyen und Frank Romeike. *Risikomanagement kompakt*. ger. Bd. 5. Veröffentlichungen an den Berufsakademien in Baden-Württemberg Studiengang Versicherung. Karlsruhe: Verl. Versicherungswirtschaft, 2009. 151 S. ISBN: 9783899524116.

[AW10] Charu C. Aggarwal und Haixun Wang, Hrsg. *Managing and Mining Graph Data*. eng. Bd. 40. Advances in Database Systems. 1386-2944. Boston, MA: Springer Science+Business Media, LLC, 2010. 1 online resource. ISBN: 978-1-4419-6045-0. DOI: 10.1007/978-1-4419-6045-0.

[Bal09] Helmut Balzert. *Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering. Basiskonzepte und requirements engineering*. Heidelberg: Spektrum Akademischer Verlag, 2009. ISBN: 978-3-8274-1705-3. DOI: 10.1007/978-3-8274-2247-7.

[BB13] Marcus A. Bellamy und Rahul C. Basole. „Network analysis of supply chain systems: A systematic review and future research“. In: *Systems Engineering* 16.2 (2013), S. 235–249. ISSN: 10981241. DOI: 10.1002/sys.21238.

[BC06] Frank Beekmann und Peter Chamoni. „Verfahren des Data Mining“. In: *Analytische Informationssysteme. Business-Intelligence-Technologien und -Anwendungen ; mit 13 Tabellen*. Hrsg. von Peter Chamoni. 3., vollst. überarb. Aufl. Berlin: Springer, 2006, S. 263–282. ISBN: 978-3-540-29286-9. DOI: 10.1007/3-540-33752-0_13.

[Bri+18] A. Brintrup u. a. „Predicting Hidden Links in Supply Networks“. In: *Complexity* 2018 (2018). PII: 9104387, S. 1–12. ISSN: 1076-2787. DOI: 10.1155/2018/9104387.

[BV08] Udo Bankhofer und Jürgen Vogel. *Datenanalyse und Statistik. Eine Einführung für Ökonomen im Bachelor ; [Bachelor geeignet*. ger. 1. Aufl. Lehrbuch. Wiesbaden: Gabler, 2008. 328 S. ISBN: 978-3-8349-0434-8. DOI: 10.1007/978-3-8349-9654-1.

[CDI98] Soumen Chakrabarti, Byron Dom und Piotr Indyk. „Enhanced hypertext categorization using hyperlinks“. In: *ACM SIGMOD Record* 27.2 (1998), S. 307–318. ISSN: 0163-5808. DOI: 10.1145/276305.276332.

[Cer17] A. Bahar Ceritoğlu. „Risikomanagement in Lieferketten – Supply-Chain-Risikomanagement“. In: *Risikomanagement in Unternehmen. Interkulturelle Betrachtungen zwischen Deutschland, Österreich und der Türkei*. Hrsg. von Stephan Schöning, Handan Sümer Gögüs und Helmut Pernsteiner. Wiesbaden: Springer Gabler, 2017, S. 269–295. ISBN: 978-3-658-07072-4. DOI: 10.1007/978-3-658-07073-1_11.

[CFM10] Deepayan Chakrabarti, Christos Faloutsos und Mary McGlohon. „Graph Mining: Laws and Generators“. In: *Managing and Mining Graph Data*. Hrsg. von Charu C. Aggarwal und Haixun Wang. Advances in Database Systems. 1386-2944 40. Boston, MA: Springer Science+Business Media, LLC, 2010, S. 69–123. ISBN: 978-1-4419-6045-0.

[CH02] Thomas Y. Choi und Yunsook Hong. „Unveiling the structure of supply networks: case studies in Honda, Acura, and Daimler-Chrysler“. In: *Journal of Operations Management* 20.5 (2002), S. 469–493. ISSN: 02726963. DOI: 10.1016/S0272-6963(02)00025-6.

[CH07] Diane J. Cook und Lawrence B. (. Holder, Hrsg. *Mining graph data*. eng. Cook, Diane J., (ed. lit.) Holder, Lawrence B. ((ed. lit.) Hoboken (New Jersey): Wiley-Interscience, 2007. 479 S. ISBN: 978-0-471-73190-0.

[Cha11] Deepayan Chakrabarti. „Graph Mining“. In: *Encyclopedia of machine learning. With 78 tables*. Hrsg. von Claude Sammut. Springer Reference. New York, NY: Springer, 2011, S. 469–471. ISBN: 978-0-387-30768-8.

[Die17] Reinhard Diestel. *Graphentheorie*. ger. 3., neu bearb. und erw. Aufl. Berlin: Spektrum Akademischer Verlag Gmbh, 2017. pages. ISBN: 978-3-662-53633-9.

[Dru11] Chris Drummond. „Classification“. In: *Encyclopedia of machine learning. With 78 tables*. Hrsg. von Claude Sammut. Springer Reference. New York, NY: Springer, 2011, S. 168. ISBN: 978-0-387-30768-8.

[DWT15] Alan Dennis, Barbara Haley Wixom und David Tegarden. *System analysis & design. An object-oriented approach with UML*. eng. Fifth edition. Dennis, Alan (VerfasserIn) Wixom, Barbara Haley (VerfasserIn) Tegarden, David (VerfasserIn) Seeman, Elaine (Sonstige Person, Familie und Körperschaft). Hoboken, NJ: Wiley, 2015. 525 S. ISBN: 9781118804674.

[FPS96] Usama Fayyad, Gregory Piatetsky-Shapiro und Padhraic Smyth. „From Data Mining to Knowledge Discovery in Databases“. In: *AI Magazine Volume 17* 3 (1996), S. 37–54. DOI: 10.1609/aimag.v17i3.1230.

[Fre78] Linton C. Freeman. „Centrality in social networks conceptual clarification“. In: *Social Networks* 1.3 (1978). PII: 0378873378900217, S. 215–239. ISSN: 03788733. DOI: 10.1016/0378-8733(78)90021-7.

[Gal+12] Laurent Galluccio u. a. „Graph based k-means clustering“. In: *Signal Processing* 92.9 (2012). PII: S0165168411004427, S. 1970–1984. ISSN: 01651684. DOI: 10.1016/j.sigpro.2011.12.009.

[GD05] Lise Getoor und Christopher P. Diehl. „Link mining: A Survey“. In: *ACM SIGKDD Explorations Newsletter* 7.2 (2005), S. 3–12. ISSN: 1931-0145.

[GF08] Harald Gleißner und Christian Femerling. *Logistik. Grundlagen - Übungen - Fallbeispiele ; Bachelor geeignet*. ger. 1. Aufl. Lehrbuch. Wiesbaden: Gabler, 2008. 306 S. ISBN: 978-3-8349-0296-2. DOI: 10.1007/978-3-8349-9547-6.

[GMR12] Haresh Gurnani, Anuj Mehrotra und Saibal Ray. *Supply Chain Disruptions*. London: Springer London, 2012. 347 S. ISBN: 978-0-85729-777-8. DOI: 10.1007/978-0-85729-778-5.

[Gol11] Goll. *Methoden und Architekturen der Softwaretechnik*. Wiesbaden: Vieweg+Teubner Verlag, 2011. ISBN: 978-3-8348-1578-1. DOI: 10.1007/978-3-8348-8164-9.

[Gor11] Florin Gorunescu. *Data mining. Concepts, models and techniques*. eng. Bd. 12. Intelligent systems reference library. Berlin: Springer, 2011. 357 S. ISBN: 978-3-642-19720-8. DOI: 10.1007/978-3-642-19721-5.

[GP02] Manish Govil und Jean-Marie Proth. *Supply chain design and management. Strategic and tactical perspectives*. eng. Academic Press series in engineering. San Diego, Calif.: Acad. Press, 2002. 187 S. ISBN: 9780122941511. DOI: 10.1016/B978-0-12-294151-1.X5000-0.

[Gra14] Marcus Grande. „Vorgehensmodelle“. In: *100 Minuten für Anforderungsmanagement*. Hrsg. von Marcus Grande. Wiesbaden: Springer Fachmedien Wiesbaden, 2014, S. 111–114. ISBN: 978-3-658-06434-1. DOI: 10.1007/978-3-658-06435-8_15.

[Hau20] Iris Hausladen. *IT-gestützte Logistik. Systeme - Prozesse - Anwendungen*. 4., aktualisierte und erweiterte Auflage. Lehrbuch. Wiesbaden: Springer Fachmedien Wiesbaden, 2020. 388eiten. ISBN: 978-3-658-31259-6. DOI: 10.1007/978-3-658-31260-2.

[HBW03] Christine Harland, Richard Brenchley und Helen Walker. „Risk in supply networks“. In: *Journal of Purchasing and Supply Management* 9.2 (2003). PII: S1478409203000049, S. 51–62. ISSN: 14784092. DOI: 10.1016/S1478-4092(03)00004-9.

[Her+13] Andrea Herrmann u.a. *Requirements Engineering und Projektmanagement*. Xpert.press. Berlin, Heidelberg: Springer Berlin Heidelberg und Imprint: Springer, 2013. 188 S. ISBN: 978-3-642-29431-0. DOI: 10.1007/978-3-642-29432-7.

[HHU11] Otto-Ernst Heiserich, Klaus Helbig und Werner Ullmann. *Logistik. Eine praxisorientierte Einführung.* ger. 4., vollst. überarb. und erw. Aufl. Lehrbuch. Wiesbaden: Gabler, 2011. 395 S. ISBN: 978-3-8349-1852-9. DOI: 10.1007/978-3-8349-6451-9.

[HKP12] Jiawei Han, Micheline Kamber und Jian Pei. *Data mining. Concepts and techniques (3rd ed.)* eng. 3rd ed. Morgan Kaufmann series in data management systems. Amsterdam und Boston: Elsevier/Morgan Kaufmann, 2012. 703 S. ISBN: 978-0-12-381479-1. URL: http://search.ebscohost.com/login.aspx?direct=true&scope=site&db=nlebk&db=nlabk&AN=193647.

[HS11] Marco Hermann und Anja Schatz. „Supply Chain Risk Management – Relevanz und Handlungsbedarf". In: *Zeitschrift für wirtschaftlichen Fabrikbetrieb* 106.5 (2011), S. 301–305. ISSN: 0947-0085. DOI: 10.3139/104.110558.

[Hud20] Marlis von der Hude, Hrsg. *Predictive Analytics und Data Mining. Eine Einführung mit R.* Lehrbuch. Wiesbaden: Springer Vieweg, 2020. 224 S. ISBN: 978-3-658-30152-1. DOI: 10.1007/978-3-658-30153-8.

[JCZ10] Chuntao Jiang, Frans Coenen und Michele Zito. „Frequent Subgraph Mining on Edge Weighted Graphs". In: 6263 (2010), S. 77–88. DOI: 10.1007/978-3-642-15105-7_7.

[Kan20] Mehmed Kantardzic. *Data mining. Concepts, models, methods, and algorithms.* Third edition. IEEE Press. Kantardzic, Mehmed, (VerfasserIn.) Hoboken, NJ und Piscataway, NJ: Wiley und IEEE Press, 2020. 1 Online-Ressource. ISBN: 9781119516057.

[KB13] Mirza Klimenta und Ulrik Brandes. „Graph Drawing by Classical Multidimensional Scaling: New Perspectives". In: *Graph Drawing.* Hrsg. von David Hutchison u. a. Bd. 7704. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, S. 55–66. ISBN: 978-3-642-36762-5. DOI: 10.1007/978-3-642-36763-2_6.

[Kho17] Yacob Khojasteh. *Supply Chain Risk Management. Advanced Tools, Models, and Developments.* eng. Singapore: Springer Singapore, 2017. 335 S. ISBN: 978-981-10-4105-1. DOI: 10.1007/978-981-10-4106-8.

[Kim+11] Yusoon Kim u. a. „Structural investigation of supply networks: A social network analysis approach“. In: *Journal of Operations Management* 29.3 (2011), S. 194–211. ISSN: 02726963. DOI: 10.1016/j.jom.2010.11.001.

[KK08] Peter Klaus und Winfried Krieger, Hrsg. *Gabler Lexikon Logistik. Management logistischer Netzwerke und Flüsse ; [A - Z.* ger. 4., komplett durchges. und aktualisierte Aufl. Gabler Lexikon. Wiesbaden: Gabler, 2008. 637 S. ISBN: 978-3-8349-0149-1. DOI: 10.1007/978-3-8349-8772-3.

[Kle08] Achim Klenke. *Wahrscheinlichkeitstheorie.* ger. 2., korrigierte Aufl. Berlin: Springer, 2008. 624 S. ISBN: 978-3-540-76317-8. DOI: 10.1007/978-3-540-77571-3.

[Koc12] Peter Koch. *Geschichte der Versicherungswirtschaft in Deutschland.* ger. Karlsruhe: Verlag Versicherungswirtschaft, 2012. 554 S. ISBN: 978-3-89952-371-3. URL: http://sub-hh.ciando.com/book/?bok_id=322097.

[KVR08] Bernhard Korte, Jens Vygen und Rabe von Randow. *Kombinatorische Optimierung. Theorie und Algorithmen.* ger. Berlin: Springer, 2008. 675 S. ISBN: 978-3-540-76918-7. DOI: 10.1007/978-3-540-76919-4.

[Lov83] Michael C. Lovell. „Data Mining“. In: *The Review of Economics and Statistics* 65.1 (1983), S. 1. ISSN: 00346535. DOI: 10.2307/1924403.

[Mar02] Peter V. Marsden. „Egocentric and sociocentric measures of network centrality“. In: *Social Networks* 24.4 (2002). PII: S0378873302000163, S. 407–422. ISSN: 03788733. DOI: 10.1016/S0378-8733(02)00016-3.

[Nak20] Mikihisa Nakano. *Supply Chain Management. Strategy and Organization.* 1st ed. Singapore: Springer Singapore, 2020. ISBN: 978-981-13-8478-3. DOI: 10.1007/978-981-13-8479-0.

[NRC09] Dina Neiger, Kristian Rotaru und Leonid Churilov. „Supply chain risk identification with value-focused process engineering“. In: *Journal of Operations Management* 27.2 (2009), S. 154–168. ISSN: 02726963. DOI: 10.1016/j.jom.2007.11.003.

[Ope07] Open Source Initiative, Hrsg. *The Open Source Definition.* Version 1.9. Open Source Initiative. 22.03.2007. URL: https://opensource.org/docs/osd (besucht am 30. 08. 2021).

[PBB16] Hyunwoo Park, Marcus A. Bellamy und Rahul C. Basole. „Visual analytics for supply network management: System design and evaluation“. In: *Decision Support Systems* 91 (2016). PII: S0167923616301397, S. 89–102. ISSN: 01679236. DOI: 10.1016/j.dss.2016.08.003.

[PKS96] Walter W. Powell, Kenneth W. Koput und Laurel Smith-Doerr. „Interorganizational Collaboration and the Locus of Innovation: Networks of Learning in Biotechnology“. In: *Administrative Science Quarterly* 41.1 (1996), S. 116. ISSN: 00018392. DOI: 10.2307/2393988.

[QB11] Novi Quadrianto und Wray L. Buntine. „Regression“. In: *Encyclopedia of machine learning. With 78 tables.* Hrsg. von Claude Sammut. Springer Reference. New York, NY: Springer, 2011, S. 838–842. ISBN: 978-0-387-30768-8.

[RB07] Bob Ritchie und Clare Brindley. „Supply chain risk management and performance“. In: *International Journal of Operations & Production Management* 27.3 (2007), S. 303–322. ISSN: 0144-3577. DOI: 10.1108/01443570710725563.

[RKF12] Saif Ur Rehman, Asmat Ullah Khan und Simon Fong. „Graph mining: A survey of graph mining techniques“. In: *2012 Seventh International Conference on Digital Information Management (ICDIM).* 2012 Seventh International Conference on Digital Information Management (ICDIM) (Macau, Macao). Hrsg. von IEEE. IEEE / Institute of Electrical and Electronics Engineers Incorporated, 2012, S. 88–92. ISBN: 978-1-4673-2428-1. DOI: 10.1109/ICDIM.2012.6360146.

[Rom18] Frank Romeike. *Risikomanagement.* ger. Lehrbuch. Romeike, Frank (VerfasserIn). Wiesbaden, Germany: Springer Gabler, 2018. 247 S. ISBN: 978-3-658-13951-3. DOI: 10.1007/978-3-658-13952-0.

[Roy+17] Noa Roy-Hubara u. a. „Modeling Graph Database Schema“. In: *IT Professional* 19.6 (2017), S. 34–43. ISSN: 1520-9202. DOI: 10.1109/MITP.2017.4241458.

[Run15] Thomas A. Runkler. *Data mining. Modelle und Algorithmen intelligenter Datenanalyse.* ger. 2., aktualisierte Auflage. Lehrbuch. Runkler, Thomas A. (VerfasserIn). Wiesbaden: Springer Vieweg, 2015. 145 S. ISBN: 978-3-8348-1694-8. DOI: 10.1007/978-3-8348-2171-3.

[Rup13] Chris Rupp. *Systemanalyse kompakt*. ger. Unter Mitarb. von Chris Rupp. 3. Auflage. IT kompakt. Rupp, Chris (MitwirkendeR). Berlin und Heidelberg: Springer Vieweg, 2013. 161 S. ISBN: 978-3-642-35445-8. DOI: 10.1007/978-3-642-35446-5.

[Sam11] Claude Sammut, Hrsg. *Encyclopedia of machine learning. With 78 tables*. eng. Springer Reference. New York, NY: Springer, 2011. 1030 S. ISBN: 978-0-387-30768-8. DOI: 10.1007/978-0-387-30164-8.

[Sch+19] Günther Schuh u. a. „Data Mining Definitions and Applications for the Management of Production Complexity". In: *Procedia CIRP* 81 (2019). PII: S2212827119305220, S. 874–879. ISSN: 22128271. DOI: 10.1016/j.procir.2019.03.217.

[Sha13] Armin Sharafi. *Knowledge Discovery in Databases*. Wiesbaden: Springer Fachmedien Wiesbaden, 2013. 326 S. ISBN: 978-3-658-02001-9. DOI: 10.1007/978-3-658-02002-6.

[SK20] Georg Schreyögg und Jochen Koch. *Management. Grundlagen der Unternehmensführung*. 8., vollständig überarbeitete Auflage. Lehrbuch. Wiesbaden: Springer Gabler, 2020. 772 S. ISBN: 978-3-658-26513-7. DOI: 10.1007/978-3-658-26514-4.

[SMH10] Anja Schatz, Jörg Mandel und Marco Hermann. *Studie Risikomanagement in der Beschaffung 2010. Eingesetzte Strategie und Methoden, organisatorische Verankerung, Bedeutung und Reifegrad des Risikomanagements in der Beschaffung in der Industrie*. Stuttgart: Fraunhofer IPA, 2010.

[Ste+15] George Stergiopoulos u. a. „Using Centrality Measures in Dependency Risk Graphs for Efficient Risk Mitigation". In: *Critical Infrastructure Protection IX. 9th IFIP 11.10 International Conference, ICCIP 2015, Arlington, VA, USA, March 16-18, 2015, Revised Selected Papers*. Hrsg. von Mason Rice und Sujeet Shenoi. 1st ed. 2015. Bd. 466. IFIP Advances in Information and Communication Technology 466. Cham und s.l.: Springer International Publishing, 2015, S. 299–314. ISBN: 978-3-319-26566-7. DOI: 10.1007/978-3-319-26567-4_18.

[STT81] Kozo Sugiyama, Shojiro Tagawa und Mitsuhiko Toda. „Methods for Visual Understanding of Hierarchical System Structures". In: *IEEE Transactions on Systems, Man, and Cybernetics* 11.2 (1981), S. 109–125. ISSN: 0018-9472. DOI: 10.1109/TSMC.1981.4308636.

[Sys06] Andreas Syska, Hrsg. *Produktionsmanagement. Das A - Z wichtiger Methoden und Konzepte für die Produktion von heute*. ger. 1. Aufl. Wiesbaden: Gabler, 2006. 184 S. ISBN: 978-3-8349-0235-1. DOI: 10.1007/978-3-8349-9091-4.

[Tan06] Christopher S. Tang. „Perspectives in supply chain risk management“. In: *International Journal of Production Economics* 103.2 (2006). PII: S0925527306000405, S. 451–488. ISSN: 09255273. DOI: 10.1016/j.ijpe.2005.12.006.

[TL10] Lei Tang und Huan Liu. „Graph Mining Applications to Social Network Analysis“. In: *Managing and Mining Graph Data*. Hrsg. von Charu C. Aggarwal und Haixun Wang. Bd. 40. Advances in Database Systems. 1386-2944 40. Boston, MA: Springer Science+Business Media, LLC, 2010, S. 487–513. ISBN: 978-1-4419-6045-0. DOI: 10.1007/978-1-4419-6045-0_16.

[Toi11] Hannu Toivonen. „Association Rule“. In: *Encyclopedia of machine learning. With 78 tables*. Hrsg. von Claude Sammut. Springer Reference. New York, NY: Springer, 2011, S. 49–50. ISBN: 978-0-387-30768-8.

[TS10] Koji Tsuda und Hiroto Saigo. „Graph Classification“. In: *Managing and Mining Graph Data*. Hrsg. von Charu C. Aggarwal und Haixun Wang. Advances in Database Systems. 1386-2944 40. Boston, MA: Springer Science+Business Media, LLC, 2010, S. 337–364. ISBN: 978-1-4419-6045-0.

[van12] Jarke J. van Wijk. „Graph Visualization“. In: *Graph drawing. 19th International Symposium, GD 2011, Eindhoven, The Netherlands, September 21 - 23, 2011 ; revised selected papers*. Hrsg. von Marc van Kreveld. Bd. 7034. Lecture Notes in Computer Science 7034. Berlin und Heidelberg: Springer, 2012, S. 86. ISBN: 978-3-642-25877-0. DOI: 10.1007/978-3-642-25878-7_9.

[Wan+07] Jiabing Wang u. a. „A Graph Clustering Algorithm Based on Minimum and Normalized Cut“. In: *Computational science - ICCS 2007. 7th international conference, Beijing China, May 27 - 30, 2007; proceedings*. Hrsg. von Yong Shi. Bd. 4487. Lecture Notes in Computer Science 4487. Berlin: Springer, 2007, S. 497–504. ISBN: 978-3-540-72583-1. DOI: 10.1007/978-3-540-72584-8_66.

[Wei00] Martin H. Weik. „system analysis“. In: *Computer science and communications dictionary*. Hrsg. von Martin H. Weik. Boston, Mass.: Kluwer, 2000, S. 1715–1716. ISBN: 978-0-7923-8425-0. DOI: 10.1007/1-4020-0613-6_18845.

[Wer17] Hartmut Werner. *Supply Chain Management. Grundlagen, Strategien, Instrumente und Controlling*. ger. 6., aktualisierte und überarbeitete Auflage. Lehrbuch. Werner, Hartmut (VerfasserIn). Wiesbaden: Springer Gabler, 2017. 548 S. ISBN: 978-3-658-18383-7. DOI: 10.1007/978-3-658-18384-4.

[Wil08] Sean P. Willems. „Data Set —Real-World Multiechelon Supply Chains Used for Inventory Optimization“. In: *Manufacturing & Service Operations Management* 10.1 (2008), S. 19–23. ISSN: 1523-4614. DOI: 10.1287/msom.1070.0176.

[Win04] Georg A. Winkelhofer. *Management- und Projekt-Methoden. Ein Leitfaden für IT, Organisation und Unternehmensentwicklung ; mit 64 Tabellen*. ger. 3., vollst. überarb. Aufl. Berlin: Springer, 2004. 322 S. ISBN: 3-540-22912-4. DOI: 10.1007/b138122.

[Wit19] Frank Witte. „Requirements Engineering“. In: *Testmanagement und Softwaretest*. Hrsg. von Frank Witte. Wiesbaden: Springer Fachmedien Wiesbaden, 2019, S. 61–73. ISBN: 978-3-658-25086-7. DOI: 10.1007/978-3-658-25087-4_8.

[WW11] Andreas Wieland und Carl Marcus Wallenburg. *Supply-Chain-Management in stürmischen Zeiten*. ger. Berlin: Univ.-Verl. der TU, 2011. 19 S. ISBN: 9783798323049.

[ZFW18] Tirazheh Zare-Garizy, Gilbert Fridgen und Lars Wederhake. „A Privacy Preserving Approach to Collaborative Systemic Risk Identification: The Use-Case of Supply Chain Networks“. In: *Security and Communication Networks* 2018 (2018). PII: 3858592, S. 1–18. ISSN: 1939-0114. DOI: 10.1155/2018/3858592.

Anhang

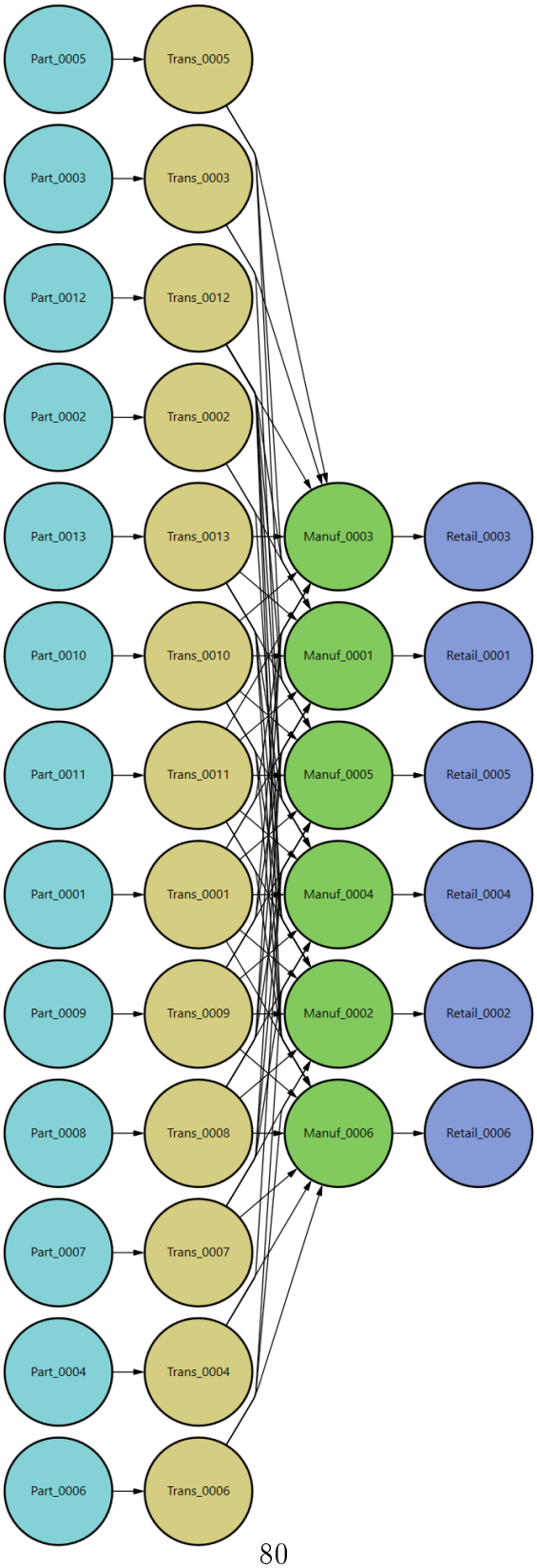

Abbildung A.1: Visualisierung der von Willems [Wil08] bereitgestellten Daten einer SC als Graph im Sugiyama-Layout

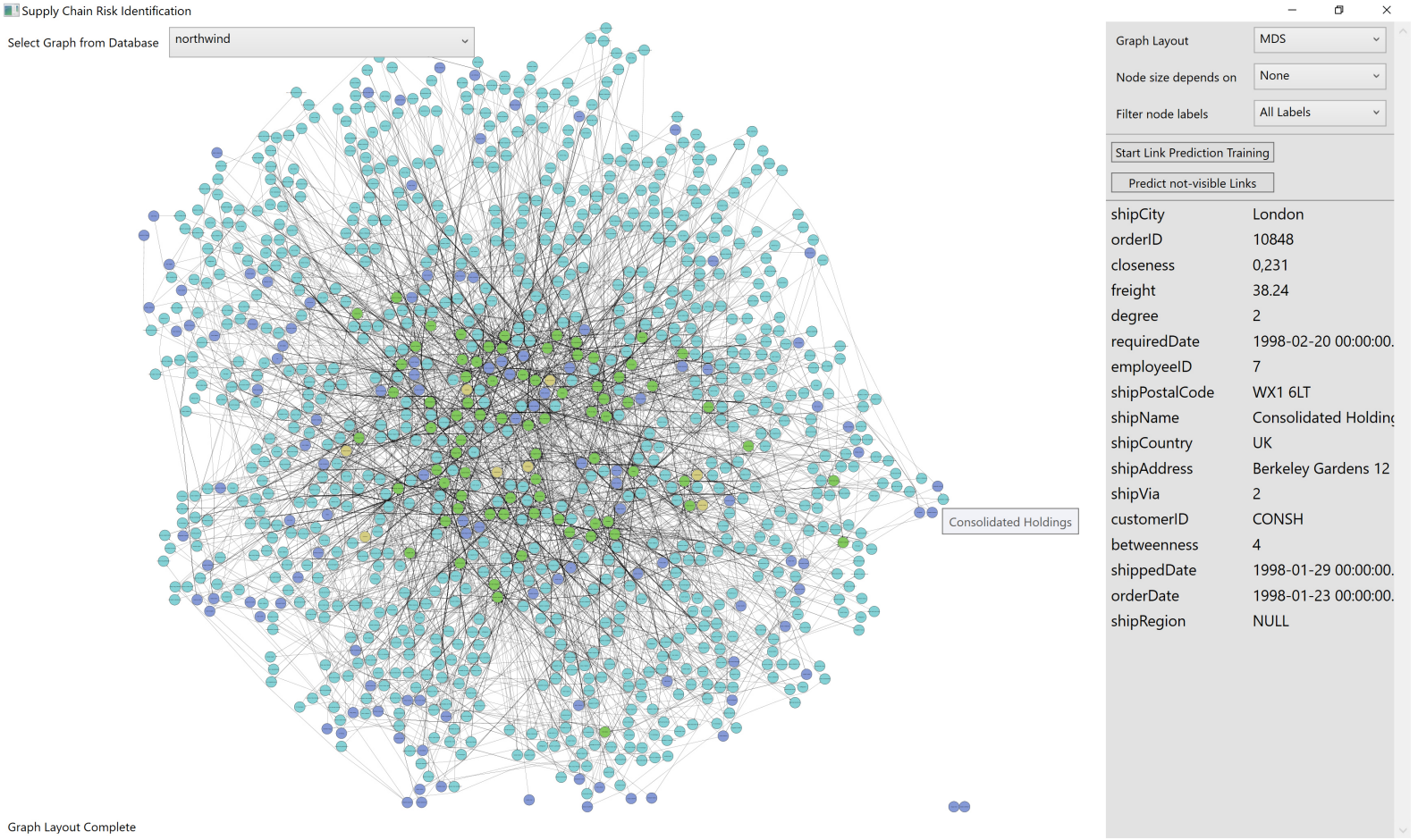

Abbildung A.2: Screenshot der Benutzeroberfläche des Prototypen. Die Interaktion des Cursors mit einem Knoten listet zugehörige Daten in der Ansicht auf.