

# **CausalML – Konzeptbeschreibung**

Dipl.-Ing. Oliver Lemke

Institut für Eisenbahnwesen und Verkehrssicherung

TU Braunschweig

Version 0.1.1, 18.08.2005

---

# Inhaltsverzeichnis

<b>1</b>	<b>Document control .....</b>	<b>1</b>
<b>2</b>	<b>Hintergrund und Einleitung .....</b>	<b>1</b>
<b>3</b>	<b>Randbedingungen und Anforderungen.....</b>	<b>2</b>
3.1	Allgemeine Anforderungen .....	2
3.2	Spezifische Anforderungen .....	3
<b>4</b>	<b>Die Architektur von CausalML .....</b>	<b>4</b>
4.1	Wahl von XML als Grundlage .....	4
4.2	Grundlegender struktureller Aufbau.....	4
4.2.1	Struktur der Meta-Informationen .....	5
4.2.2	Struktur der Nutzlast .....	6

---

## 1 Document control

ID: IfEV-CML-Le-001

Autor und ©: Oliver Lemke

Version: 0.1.1

Datum: 18.08.2005

Lizenz: (noch nicht zugewiesen)

Hinweis: Neue, erweiterte und verbesserte Versionen dieses Dokuments erscheinen ohne Ankündigung.

Änderungen: In 4.2.2.1 die Erklärung der impliziten Strukturdefinition erweitert

## 2 Hintergrund und Einleitung

CausalML ist ein XML-basiertes Datenaustauschformat für kausale Faktoren und deren Abhängigkeitsbeziehungen. Es wird im Allgemeinen zur Speicherung der Ergebnisse einer Why-Because-Analyse (WBA) genutzt. Die WBA ist eine strukturierte Unfall- bzw. Vorfallsanalysemethode, mit der Grundursachen und kausale Abhängigkeiten ermittelt werden, die zum Eintritt eines bestimmten Vorfalles geführt haben [LAD98].

Die Daten, die in einer CausalML-Datei enthalten sind, können zur Erstellung eines Why-Because-Graphen (WB-Graphen) verwendet werden, der eine Form der Ergebnisdarstellung einer WBA ist. Somit kann CausalML auch als Datenaustauschformat für WB-Graphen verstanden werden.

In akademischen Untersuchungen verschiedener Vorfälle aus einer Vielzahl von Fachdomänen (Luftfahrt, Informationstechnik, Eisenbahnwesen, kommerzielle Schifffahrt, etc.) hat die WBA ihre prinzipielle Tauglichkeit als Vorfallsanalysemethode bewiesen. Zunehmend wird sie nun auch im industriellen Umfeld eingesetzt, so z.B. von Siemens Transportation Systems [BRA02]. Zum Zeitpunkt dieser Einführung gab es jedoch kaum leistungsfähige Werkzeuge für die Bearbeitung von WB-Graphen, so dass für diese Aufgabe teilweise einfache Zeichensoftware verwendet wurde. Nachteilig bei diesem Vorgehen ist, dass in den so erstellten Dateien die semantischen Informationen über die Beziehungen zwischen den einzelnen kausalen Faktoren verloren gehen.

Die wenigen existierenden Werkzeuge, die tatsächlich auf Ebene der inhaltlichen Informationen des WB-Graphen arbeiten, verwendeten zum damaligen Zeitpunkt verschiedene text- oder datenbankbasierte Dateiformate, die nicht miteinander kompatibel waren. Im Eindruck dieser unbefriedigenden

Situation wurde dann die Schaffung eines einheitlichen, klar strukturierten und einfach zu handhabenden Datenaustauschformats für kausal strukturierte Vorfallsanalysedaten vorgeschlagen.

## 3 Randbedingungen und Anforderungen

Vor Beginn der eigentlichen Entwicklung von CausalML wurden zunächst Anforderungen an das Datenaustauschformat formuliert. Zum einen waren dies generelle Anforderungen, die für jedes Datenaustauschformat gültig sind, und zum anderen Anforderungen, die sich aus dem vorgesehenen Anwendungszweck ergeben. In den beiden folgenden Unterkapiteln werden die einzelnen Anforderungen beider Kategorien beschrieben.

### 3.1 Allgemeine Anforderungen

Folgende allgemeine Anforderungen wurden für die Entwicklung von CausalML aufgestellt:

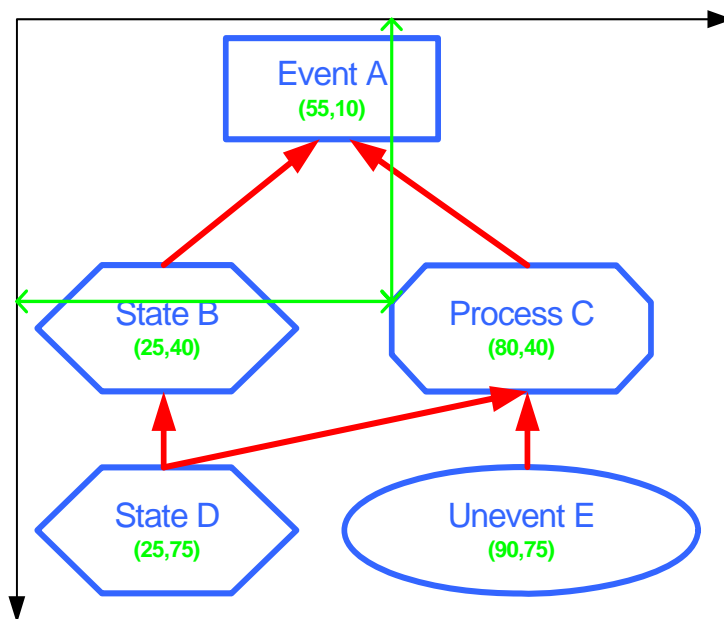
- A. Das Datenaustauschformat soll auf bekannten, gut dokumentierten Standards aufbauen.
- B. Das Datenaustauschformat soll möglichst genau an den geplanten Anwendungszweck angepasst werden.
- C. Lese- und Schreibvorgänge von Daten im Austauschformat sollen durch Toolkits bzw. Softwarebibliotheken gut unterstützt werden.
- D. Die Repräsentation des Datenaustauschformats sollte nach Möglichkeit auch durch den Menschen lesbar sein.

## 3.2 Spezifische Anforderungen

Die spezifischen Anforderungen beziehen sich im Wesentlichen auf die zu speichernden Informationen. CausalML soll demnach die folgenden Informationskategorien speichern können:

- A. **Informationen zu den einzelnen kausalen Faktoren:** Dazu gehören z.B. Informationen wie der den Faktorinhalt beschreibenden Text und ein eindeutiger Bezeichner.
- B. **Informationen zur Struktur des Graphen:** Die Struktur des WB-Graphen wird durch die Angabe der Beziehungen zwischen den einzelnen Faktoren beschrieben.
- C. **Informationen zum grafischen Layout des Graphen:** Neben der logischen Struktur des Graphen sollen zusätzlich auch Angaben zu Position, Größe und Aussehen der einzelnen Knoten<sup>1</sup> auf einer Zeichenfläche abgespeichert werden können.

Die folgende Grafik zeigt einen beispielhaften WB-Graph, in dem diese Informationskategorien durch verschiedene Farben hervorgehoben sind:



Blau dargestellt sind die Knoten des Graphen, rot die Angaben zur logischen Struktur und grün die Angaben zum Layout. Für jede dieser Informationskategorien sollen entsprechende Strukturen im Datenaustauschformat vorgesehen werden.

<sup>1</sup> Als Knoten wird dabei diejenige Entität bezeichnet, die einen kausalen Faktor in einem WB-Graphen repräsentiert.

## 4 Die Architektur von CausalML

Nachdem zuvor die Anforderungen beschrieben worden sind, wird in den folgenden Unterkapiteln die tatsächlich realisierte Struktur von CausalML vorgestellt. Dabei wird zunächst auf die grundlegenden Architekturentscheidungen und anschließend auf die wesentlichen Konstrukte eingegangen.

### 4.1 Wahl von XML als Grundlage

Im Vorfeld der Entwicklung von CausalML zeigte sich, dass die in Abschnitt 3.1 beschriebenen allgemeinen Anforderungen an das Datenaustauschformat am besten erfüllt werden können, wenn die XML (extensible markup language) [W3C02] als Basis für das Datenaustauschformat verwendet werden würde. Die Gründe dafür sind:

- XML ist ein weit verbreiteter Standard für die Erstellung maschinenlesbarer Dokumente, er ist hervorragend beschrieben und bei vielen Anwendern verbreitet, wodurch eine große Wissens- und Erfahrungsbasis zur Verfügung steht.
- XML definiert nur die grundlegenden Regeln für den Aufbau von Dokumenten. Für jeden Anwendungsfall muss die entsprechende Dokumentenstruktur explizit spezifiziert werden – dadurch ergibt sich eine optimale Anpassbarkeit an die jeweilige Aufgabe.
- XML wird durch zahlreiche Werkzeuge, Softwarepakete und Programmierschnittstellen unterstützt. Dadurch ist es sehr einfach, Daten im XML-Format zu lesen oder zu schreiben, Daten zwischen verschiedenen Formaten zu transformieren oder in einer Datenbank abzulegen.
- XML ist bis zu einem gewissen Grad auch ohne zusätzliche Aufbereitung durch Menschen lesbar

CausalML wird somit als eine Anwendung von XML realisiert. Die Struktur von CausalML ist in einer XML-Schema Definition [W3C04\_0], [W3C04\_1], [W3C04\_2] formal beschrieben.

### 4.2 Grundlegender struktureller Aufbau

Jedes CausalML-Dokument repräsentiert den Informationsgehalt genau eines WB-Graphen. In einem CausalML-Dokument müssen somit all diejenigen Informationen zu kausalen Faktoren und deren Verknüpfungen enthalten sein, die notwendig sind, um den Graphen zu rekonstruieren.

Prinzipiell gliedert sich ein CausalML-Dokument in zwei wesentliche Blöcke: Zum einen in einen Block mit Meta-Informationen, zum anderen in die „Nutzlast“-Informationen, aus denen der eigentliche Graph gebildet werden kann. Der Block der Meta-Informationen wird durch das `<info>`-Element

begrenzt, der Block der Nutzlast durch das `<all_factors>`-Element. Root-Element eines CausalML-Dokuments ist das `<graph>`-Element. Folgende Grafik verdeutlicht diese Struktur:

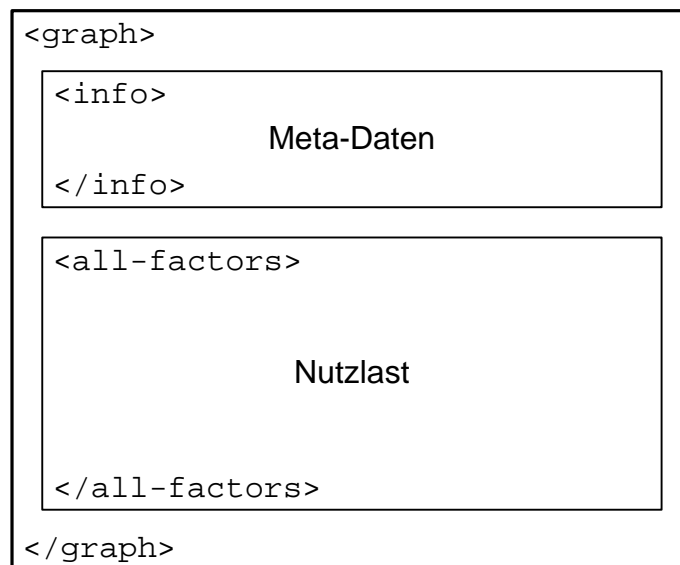


Abbildung 1 - Strukturebene Graph - Meta-Informationen / Nutzlast

#### 4.2.1 Struktur der Meta-Informationen

Wichtigstes (und zwingend vorhandenes) Element im Block der Meta-Informationen ist die Deklaration der möglichen Arten von kausalen Faktoren. Die ursprüngliche Definition der WBA [LAD98] sieht vier verschiedene Arten von kausalen Faktoren vor<sup>2</sup>: Zustände, Ereignisse, Prozesse und Nicht-Ereignisse. Jedoch werden bei manchen Anwendungsfällen der WBA auch andere Arten von kausalen Faktoren verwendet. Siemens Transportation Systems benutzt beispielsweise auch Faktoren der Art „Gegenmaßnahme“. Bei der Entwicklung der CausalML wurde daher vorgesehen, dass in jedem Dokument zunächst die erlaubten Arten für kausale Faktoren definiert werden müssen. Diese Definition erfolgt durch eine Aufzählung aller zulässiger Arten sowie einer optionalen Deklaration der grafischen Form des Knotens, durch die ein kausaler Faktor dieser Art im WB-Graphen dargestellt wird.

Technisch gesehen besteht die Deklaration einer zulässigen Faktoren-Art aus einem `<kind>`-Element unterhalb des `<kind_declaration>`-Elements. Die Deklaration für einen regulären WB-Graphen sieht beispielsweise folgendermaßen aus:

```

<kind_declaration>
  <kind id="event" shape="rectangle"/>
  <kind id="state" shape="hexagon"/>
  <kind id="process" shape="octagon"/>
  
```

<sup>2</sup> Die durch unterschiedliche geformte Knoten im WB-Graphen unterschieden werden können.

```
<kind id="unevent" shape="ellipse" />
</kind_declaration>
```

Hier werden die im Dokument zulässigen Faktoren-Typen „event“, „state“, „process“ und „unevent“ deklariert und diesen jeweils eine grafische Repräsentation zugewiesen. Für das `shape`-Attribut ist dabei eine Liste zulässiger Werte vordefiniert. Details zur diesen hier kurz vorgestellten Konstrukten finden sich in der Referenzdokumentation.

Die übrigen Elemente im Block der Meta-Informationen sind optional. Sie enthalten unter Anderem Informationen zum Ersteller des Dokuments, zur Version und Angaben, die für das Layout des Graphen erforderlich sind. Details dazu können auch hier der Referenzdokumentation entnommen werden.

### 4.2.2 Struktur der Nutzlast

In den folgenden Unterkapiteln werden zunächst einige grundlegende Vorüberlegungen dargestellt, die die Struktur des Nutzlastblocks mitbestimmt haben. Anschließend werden die wesentlichen Merkmale dieser Datenmenge beschrieben.

#### 4.2.2.1 Strukturelle Vorüberlegungen

Allgemein bieten sich zur Speicherung von Graphen zwei verschiedene Strategien an:

- **Explizite Strukturdefinition:** Trennung der Knotendefinition von der Definition ihrer Verbindungen. Dabei werden zunächst alle Knoten definiert und mit einem eindeutigen Bezeichner versehen. In einem davon getrennten Block werden dann die Verknüpfungen dieser Knoten untereinander hergestellt.
- **Implizite Strukturdefinition:** Integration der Graphenstruktur in die Knotendefinition: Dabei wird innerhalb der Definition eines Knoten vermerkt, welche Verbindungen er zu anderen Knoten des Graphen besitzt. Eine Realisierungsmöglichkeit besteht darin, zu jedem Knoten zu vermerken, welche anderen Knoten dessen direkte Nachfolger (Unterknoten) sind. Dieser Speicherungsmechanismus entspricht dem Prinzip einer verketteten Liste in der Informatik.

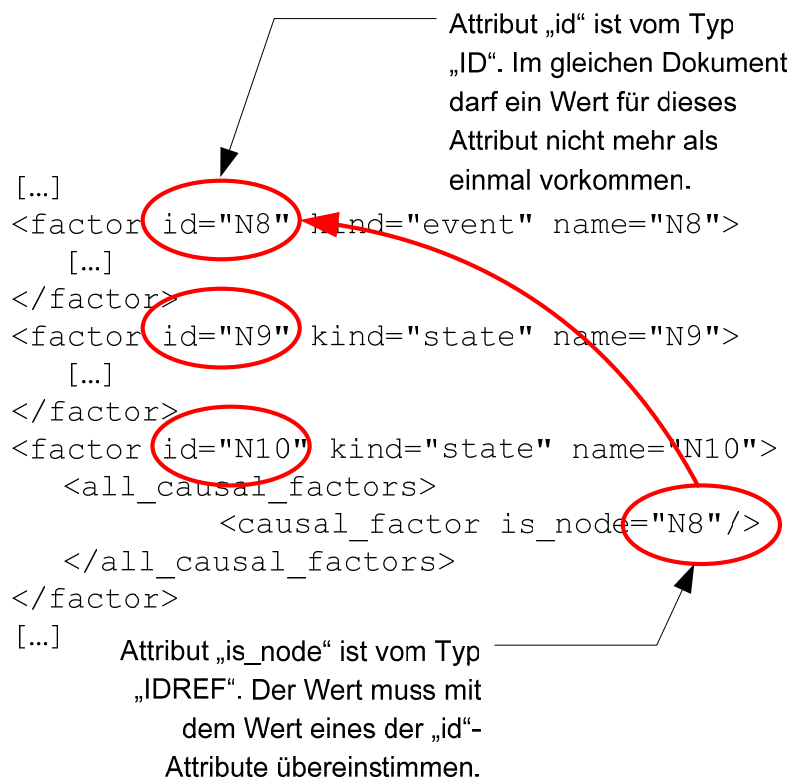
Beide Strategien haben spezifische Vor- und Nachteile. Vorteil der impliziten Strukturdefinition besteht darin, dass man im Fall des WB-Graphen unmittelbar erkennen kann, ob ein Faktor eine Grundursache darstellt oder nicht: Ein Faktor repräsentiert genau dann eine Grundursache, wenn er keine weiteren Unterknoten besitzt. Die einfache Abbildbarkeit dieser Struktureigenschaft war unter anderem ausschlaggebend dafür, dass für CausalML die implizite Strukturbeschreibung gewählt wurde. Ein weiterer Grund war die bessere Kapselung aller zu einem Knoten gehörenden Attribute. Dies erleichtert das Einlesen der Daten aus einem CausalML-Dokument in ein objektorientiertes Modell.

#### 4.2.2.2 Sicherstellung eindeutiger Bezeichner und der Integrität von Referenzen

Zur Sicherstellung der Integrität des in einem CausalML-Dokument beschriebenen WB-Graphen müssen drei wesentliche Randbedingungen erfüllt sein:

1. Jeder Faktor innerhalb eines Dokuments muss eindeutig identifiziert werden können
2. Bei Verweisen auf andere Faktioren muss sichergestellt werden, dass das Verweisziel tatsächlich existiert
3. Es muss sichergestellt werden, dass jedem kausalen Faktor eine gültige Faktoren-Art zugewiesen wird.

Zur Erfüllung dieser Anforderungen bieten sich die ID/IDREF-Konstrukte an. Ein Attribut oder Element vom Typ ID wird vom Parser als eindeutige Referenz angesehen. Der einem Attribut vom Typ ID zugewiesene Wert darf daher innerhalb eines Dokuments nur genau einmal vorkommen, wobei die Einhaltung dieser Randbedingung bereits auf Parser-Ebene überwacht werden kann. Ein Dokument, in dem verschiedene ID-Attribute den gleichen Wert besitzen, wird somit als „nicht gültig“ (non-valid) zurückgewiesen. Ähnliches gilt für die Referenzierung über das IDREF-Attribut. Dabei muss der Wert dieses Attributs mit dem Wert eines der ID-Attribute innerhalb des Dokuments übereinstimmen. Folgende Grafik zeigt die Funktionsweise dieser beiden Konstrukte:



**Abbildung 2 - Veranschaulichung der ID und IDREF Attribute**

Innerhalb von CausalML kommt dieses Konstrukt an zwei Stellen zum Einsatz. Zum einen werden die verschiedenen Arten von kausalen Faktoren (➔ 4.2.1) über ID-Attribute definiert. Dadurch wird sichergestellt, dass nicht zwei identische Faktor-Arten in einem Dokument definiert werden. Zweiter Anwendungsfall ist die Sicherstellung von eindeutigen Bezeichnern für die einzelnen Faktoren, weswegen diese als Attribut vom Typ ID deklariert werden. Gleichzeitig kann dann über den IDREF-Mechanismus geprüft werden, dass innerhalb eines Faktors nur auf einen tatsächlich existierenden, anderen kausalen Faktor als Unterknoten referenziert werden kann.

Anzumerken ist jedoch, dass die Werte aller im Dokument vorkommenden Attribute vom Typ ID eine gemeinsame Menge bilden. Daher kann durch das IDREF-Konstrukt nicht überprüft werden, ob die Referenzierung im jeweiligen Kontext auf einen sinnvollen ID-Wert verweist, wenn an mehreren Stellen innerhalb des Dokuments ID-Attribute verwendet werden. Konkret bedeutet dies für CausalML, dass ein Verweis im „is\_node“-Attribut auf eine der Attribut-Arten gültig wäre. Eine Prüfung auf derartige Fehler kann nicht direkt vom Parser erledigt werden und muss in „höheren“ Verarbeitungsschichten erfolgen. Dennoch ist die in CausalML realisierte Prüfung über ID/IDREF sicher besser als nichts... ☺

### 4.2.2.3 Abbildung der kausalen Faktoren

In diesem Abschnitt wird die realisierte Struktur der Nutzdatenspeicherung beschrieben. Der Block der Nutzdaten als Ganzes wird durch das <all\_factors>-Element begrenzt (➔ Abbildung 1 - Strukturebene Graph - Meta-Informationen / Nutzlast). Unterhalb dieses Elements werden die einzelnen im Graph abgebildeten kausalen Faktoren durch einzelne <factor>-Elemente repräsentiert, wobei ein Faktor durch genau ein solches Element dargestellt wird. Dieses kapselt alle Informationen zum jeweiligen kausalen Faktor. Die folgende Skizze zeigt dies schematisch:

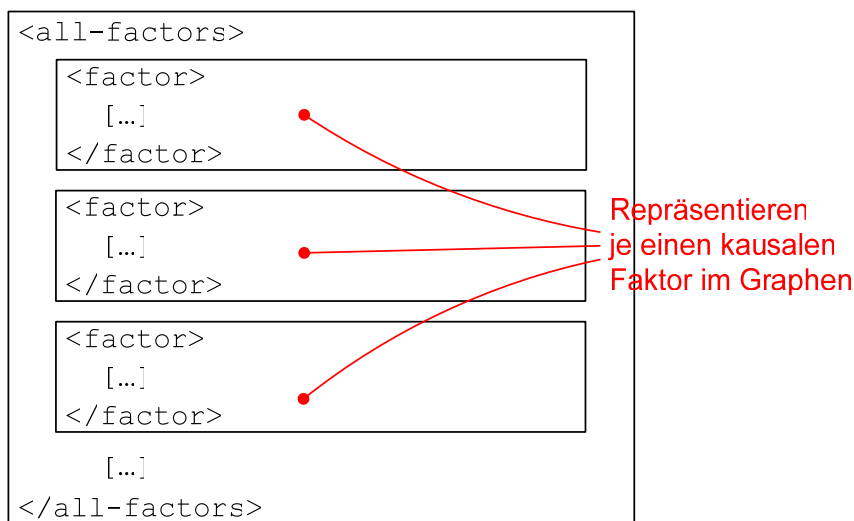


Abbildung 3 - Struktur der kausalen Faktoren in CausalML

Jedem `<faktor>`-Element sind weitere Unterelemente und Attribute zugewiesen, die den Inhalt des kausalen Faktors (Text, Art des Faktors (→ 4.2.1), Kommentar, etc.) näher spezifizieren. Besondere Bedeutung kommt dabei dem `<all_causal_factors>`-Element zu. Über dieses Element (bzw. dessen Unterelemente) wird gemäß der Entscheidung für die implizite Abbildung der Graphen-Struktur (→ 4.2.2.1) die Verknüpfung zu untergeordneten kausalen Faktoren gesteuert.

Konkret wird dabei jede Referenz auf einen anderen kausalen Faktor durch ein `<causal_factor>`-Element repräsentiert. Das Verweisziel wird dabei durch das Attribut „is\_node“ spezifiziert. Folgendes Codebeispiel zeigt einen solchen Verweis:

```
<faktor id="N3" kind="state" name="N3">
  <text>S5210 hat hohe Fahrgeschwindigkeit erreicht.</text>
  <all_causal_factors>
    <causal_factor is_node="N22"/>
    <causal_factor is_node="N7"/>
  </all_causal_factors>
</faktor>
```

Hierbei wird zunächst der Faktor N3 definiert, für den die beiden anderen Faktoren „N22“ und „N7“ kausale Faktoren darstellen. Dies wird in CausalML durch die beiden `<causal_factor>`-Elemente repräsentiert, die auf diese beiden Faktoren verweisen. Das gesamte Konstrukt unterhalb des `<all_causal_factors>`-Elements kann somit gelesen werden als: „kausale Faktoren für diesen Knoten sind die Knoten N22 und N7“.

Die folgende Skizze soll zeigen, wie die kausalen Beziehungen in einem WB-Graph durch CausalML abgebildet werden. Jeder Pfeil zeigt dabei auf diejenige kausale Beziehung, die durch den rot eingefärbten Textblock im zugehörigen CausalML-Dokument beschrieben wird.

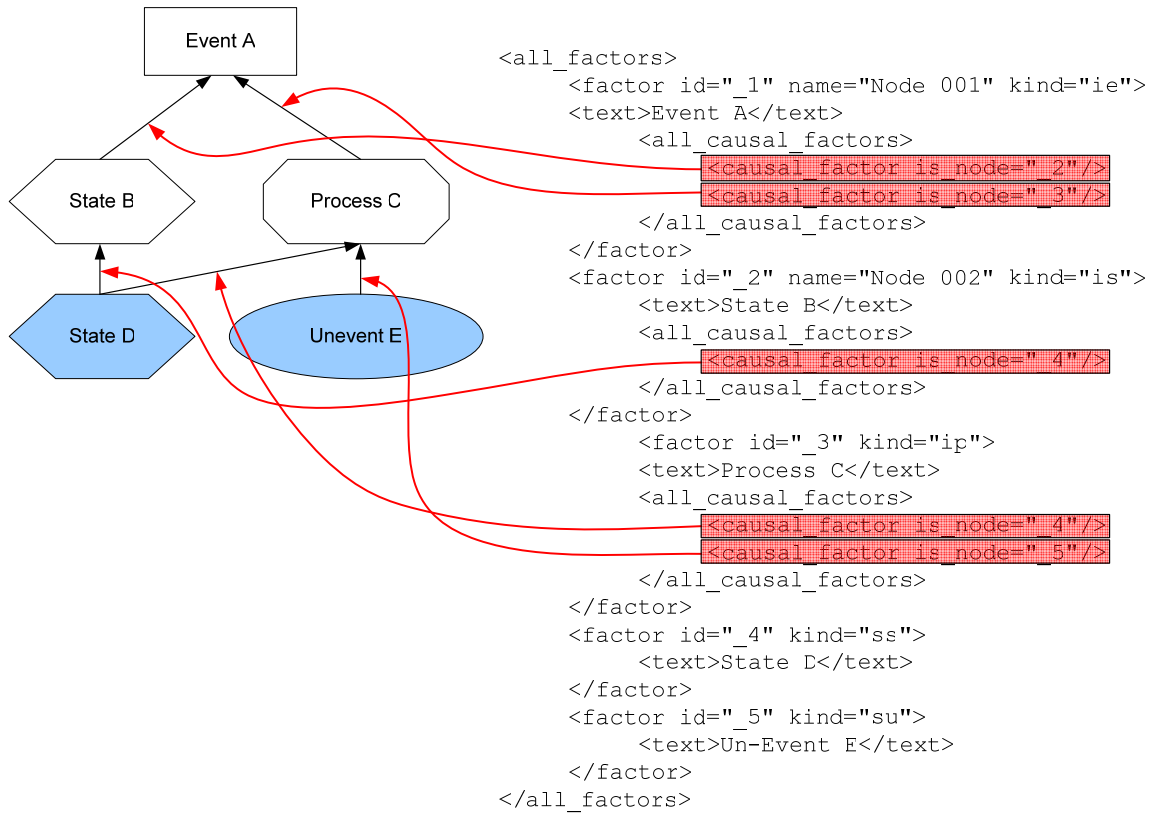


Abbildung 4 - Abbildung der kausalen Beziehungen in CausalML

---

## Literatur

- [LAD98] Ladkin, Peter B.; Loer, Karsten: „Why-Because Analysis: Formal Reasoning About Incidents”, 1998
- [BRA02] Braband, Jens: “Application of Why-Because Graphs to Railway Near-Misses”, in: Johnson, Chris (Hrsg.): “Investigation and Reporting of Incidents and Accidents (IRIA 2002)”, S. 26 – 31.
- [W3C02] World Wide Web Consortium (Hrsg.): „Extensible Markup Language (XML) 1.0 (Zweite Auflage)”, 2002, deutsche Übersetzung unter <http://www.edition-w3c.de/TR/2000/REC-xml-20001006/>
- [W3C04\_0] World Wide Web Consortium (Hrsg.): „XML Schema Part 0: Primer Second, Edition”, 2004, <http://www.w3.org/TR/xmlschema-0/>
- [W3C04\_1] World Wide Web Consortium (Hrsg.): „XML Schema Part 1: Structures, Second Edition”, 2004, <http://www.w3.org/TR/xmlschema-1/>
- [W3C04\_2] World Wide Web Consortium (Hrsg.): „XML Schema Part 2: Datatypes, Second Edition”, 2004, <http://www.w3.org/TR/xmlschema-2/>

## Abbildungen

Abbildung 1 - Strukturebene Graph - Meta-Informationen / Nutzlast .....	5
Abbildung 2 - Veranschaulichung der ID und IDREF Attribute.....	7
Abbildung 3 - Struktur der kausalen Faktoren in CausalML .....	8
Abbildung 4 - Abbildung der kausalen Beziehungen in CausalML.....	10

Ich danke Luke Emmet, Mirco Hilbert, Peter B. Ladkin, Bernd Sieker, und Fergus Toolan für die gute Zusammenarbeit und ihre Ratschläge und Ideen, ohne die CausalML nicht hätte verwirklicht werden können.