Ein einfaches Verfahren zur Erkennung häufiger Fehler in EPKs

Volker Gruhn, Ralf Laue
{gruhn,laue}@ebus.informatik.uni-leipzig.de
Lehrstuhl für Angewandte Telematik und E-Business*
Universität Leipzig, Fakultät für Informatik

Abstract: In diesem Beitrag nutzen wir den in [GL06] eingeführten Ansatz, die in einer ereignisgesteuerten Prozesskette (EPK) enthaltenen Informationen in eine Prolog-Faktenbasis zu übersetzen und durch Abfragen an den Prolog-Interpreter mögliche Modellverbesserungen zu lokalisieren.

Dabei werden, anders als in früheren Veröffentlichungen beschrieben, die Beschriftungen von Ereignissen und Funktionen in die Analyse der Modelle einbezogen.

Durch Erzeugen einer textuellen Normalform und die Beachtung von Synonymen und Antonymen wird versucht, Beschriftungen mit gleicher Bedeutung und Beschriftungen, die sich widersprechen, zu finden. Auf diese Weise lassen sich bestimmte Klassen häufig anzutreffender Modellierungsfehler in EPKs automatisch erkennen.

Das Verfahren wurde an 1253 deutschsprachigen Modellen getestet. Dabei wurden mehrere Fehler identifiziert, die mit Hilfe herkömmlicher Validierungstechniken unentdeckt bleiben.

1 Einführung

Die Semantik eines Modells lässt sich nach Esswein et al. [EGS04] stets in zwei Teile gliedern: in die (in der Regel formal wohldefinierte) Semantik der Modellierungssprache sowie die „konkrete Semantik“ jedes einzelnen Modellelements. Bei EPKs wird diese „konkrete Semantik“ durch die Beschriftung von Ereignissen und Funktionen in natürlicher Sprache angegeben. Ähnlich unterteilen Pfeifer und Niehaves [PN05] die Bedeutung graphischer Modelle in die Anordnung der formalen Modellierungselemente („model element structure“) sowie die Bedeutung der Modellelemente in der Sprache der Anwendungsdomäne („terminological structure“). Pfeifer und Niehaves unterstreichen, dass eine Validierung eines Modells immer beide genannten Aspekte berücksichtigen muss.

Bisher vorgeschlagene Verfahren zur Validierung von Geschäftsprozessmodellen beziehen jedoch (von wenigen Ausnahmen wie [ADW08] abgesehen) kaum die Beschriftung von Modellelementen in die Analyse ein. Generell spielen Funktionen und Ereignisse bei den gängigen Validierungsverfahren (siehe etwa [van97, Rum99, Men07]) keine Rolle. Bei

*Der Lehrstuhl für Angewandte Telematik und E-Business ist ein Stiftungslehrstuhl der Deutschen Telekom AG

diesen Verfahren werden lediglich Kontrollflussfehler (z. B. Deadlocks) betrachtet, die sich aus der Anordnung und den Typen der Konnektoren ergeben.

Beim Auswerten publizierter Modelle fielen uns jedoch mehrfach Fehlermuster auf, die nur bei einer Betrachtung der Beschriftung von Modellelementen - insbesondere Ereignissen - erkannt werden können. Ziel unserer Arbeit war es, auch solche Fehler automatisch zu lokalisieren. Durch eine automatische Erkennung soll insbesondere dem ungeübten Modellierer ein Werkzeug zur Hand gegeben werden, um die Qualität seiner Modelle bereits während des Modellierungsprozesses zu prüfen und zu verbessern.

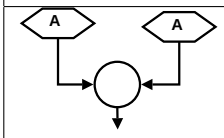
Zur technischen Validierung unseres Verfahrens wurde unser Algorithmus in das quelloffene Modellierungswerkzeug *bflow* Toolbox* integriert. Dieses bietet somit neben der Analyse von Syntax- und Kontrollflussfehlern [GL06, GLKK08] auch eine Modellüberprüfung auf die in diesem Beitrag beschriebenen inhaltlichen Fehler.

Die Fehlermuster, die untersucht werden, sind in Abschnitt 2 beschrieben. Anschließend zeigt Abschnitt 3, wie diese Fehler gefunden werden. Das Ergebnis einer ersten Validierung unserer Verfahren enthält Abschnitt 4. Schließlich werden in Abschnitt 5 die Ergebnisse diskutiert und mit anderen Arbeiten verglichen.

2 Betrachtete Fehlermuster

Bei der manuellen Analyse von Modellen aus der Praxis identifizierten wir einige häufige Modellierungsfehler, die durch gängige Verfahren zur Validierung des Kontrollflusses nicht entdeckt werden können, die jedoch durch einfache Analyse der Beschriftungen von Modellelementen zu identifizieren sind. Diese Muster werden im Folgenden vorgestellt.

2.1 Logisch identische Ereignisse vor/nach einem Konnektor

Muster A	
	Ein Konnektor hat zwei logisch identische Ereignisse als Vorgänger oder Nachfolger. Folgen die logisch identischen Ereignisse auf einen XOR-Split, ist dies ein inhaltlicher Fehler. Andernfalls ist dieses Muster ein Hinweis auf die Möglichkeit, das Modell durch Weglassen redundanter Elemente zu verkleinern.

Stehen die logisch identischen Ereignisse nach einem XOR-Split, ist dies offenbar stets ein inhaltlicher Fehler, da der XOR-Split ja gerade aussagt, dass nur eines der Ereignisse eintreten kann (und das andere nicht).

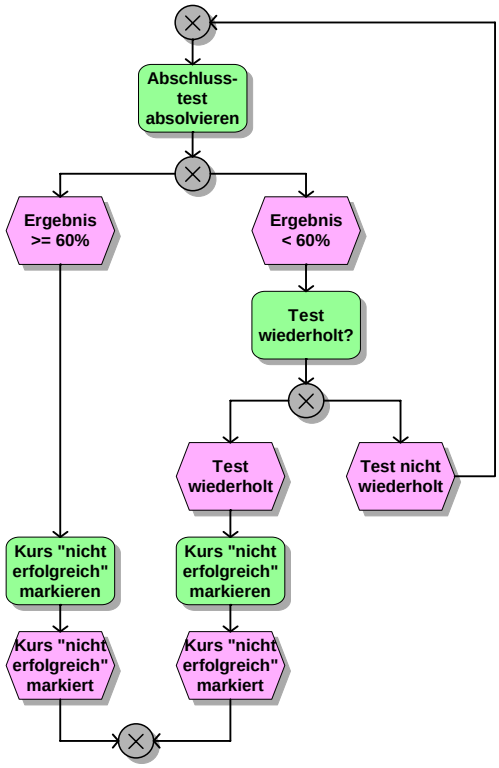


Abbildung 1: inhaltlich fehlerhaftes Modell

In allen anderen Fällen sind verschiedene Situationen möglich: Abb. 1 ist ein Modellausschnitt[1] aus [GKMZ07]. Hier weist das doppelte Vorkommen des Ereignisses “Kurs ’nicht erfolgreich’ markiert” auf einen Modellfehler hin: Vermutlich hätte im linken Zweig die Situation „Kurs erfolgreich“ modelliert werden sollen.

In den meisten Fällen jedoch dürfte das doppelte Vorkommen eines Ereignisses lediglich ein Indiz dafür sein, dass die EPK vereinfacht werden kann. Ein Beispiel zeigt Abb. 2, entnommen einer Diplomarbeit. Eine Modellierung wie im linken Modell von Abb. 2 widerspricht dem Grundsatz, die Zahl der Modellelemente auf das erforderliche Maß zu beschränken. Diese Forderung ist etwa unter der Bezeichnung „Minimalität“ in [Ron97] zu finden. Becker, Rosemann and Schütte fordern in den „Grundsätzen ordnungsgemäßer Modellierung“, dass „das Modell nicht mehr Elemente beinhalten soll, als zum Verständnis und zur Wiedergabe der Intention notwendig sind“(zitiert aus [BRS95]).

Gesondert zu betrachten sind Ereignissen, die mit trivialen Standardtexten wie „erfolgreich durchgeführt“ oder „OK“ für Ereignisbeschriftungen beschriftet sind. Da hier der Bezug darauf fehlt, *was* erfolgreich durchgeführt wurde, können Ereignisse trotz gleichlautender

[1] Die in diesem Beitrag gezeigten Modellbeispiele stellen keine kompletten EPK-Modelle dar, sondern zeigen lediglich unvollständige Modellfragmente.

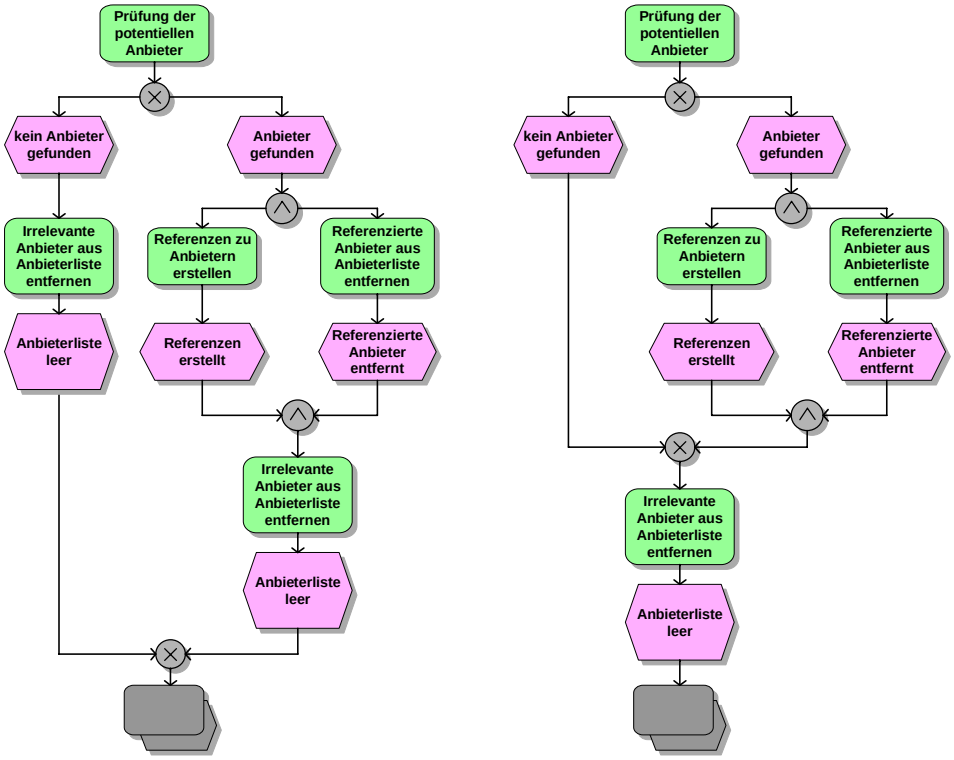


Abbildung 2: links: Originalmodell, rechts: vereinfachtes Modell

Beschriftung verschiedene betriebliche Situationen darstellen. Diese Fälle werden durch unseren Algorithmus, der solche “Trivialereignisse erkennt”, ausgeschlossen. Trotzdem sind gelegentliche “Fehlalarme” wie in dem in Abb. 3 (entnommen aus [KÖ6]) gezeigten Modellfragment möglich.

Weiter ist zu beachten, dass zwei Ereignisse trotz gleichlautender Beschriftung erkennbar verschieden sind, wenn ihnen in einer erweiterten EPK verschiedene Organisationseinheiten zugeordnet sind. Ein Ereignis „Prüfung erfolgreich“, ausgeführt von der Buchhaltung, ist ein anderes Ereignis als „Prüfung erfolgreich“, ausgeführt von der Rechtsabteilung.

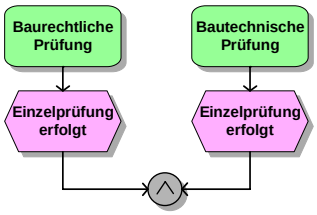
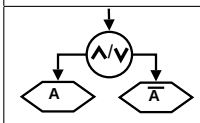


Abbildung 3: Fehlalarm bei Muster A

2.2 Zwei sich ausschließende Ereignisse durch AND-bzw. OR-Konnektor vereint

Muster B	
	Ein Konnektor vom Typ OR oder AND hat zwei Ereignisse, die sich logisch widersprechen, als Vorgänger oder Nachfolger. Dies ist ein logischer Fehler im Modell. Möglicherweise sollte der Konnektor durch einen XOR-Konnektor ersetzt werden.

In dem Modellfragment aus Abb. 4 (das einer Diplomarbeit entnommen wurde), wurde statt des die Situation korrekt beschreibenden XOR-Splits ein OR-Split verwendet. Bei unerfahrenen Modellierern ist eine solche Verwechslung zwischen OR und XOR ein häufiger Fehler.

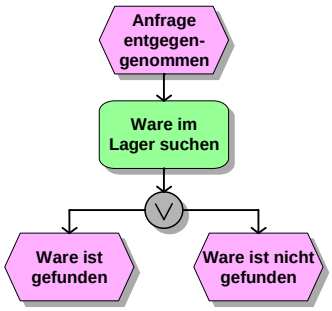


Abbildung 4:

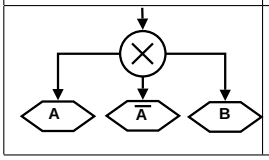
Leitet ein OR- oder AND-Split mehrere Kontrollflusszweige ein, so können diese im Modell parallel durchlaufen werden. Insbesondere heißt das, dass mehrere Ereignisse, die auf einen AND- oder OR-Join folgen, gemeinsam eintreten dürfen.

Schließen sich nun zwei auf einen OR- oder AND-Split folgende Ereignisse logisch aus (z.B. „Genehmigung erteilt“ / „Genehmigung verwehrt“), so ist davon auszugehen, dass ein Fehler im Modell vorliegt. Dieser besteht meist darin, dass statt eines XOR-Splits fälschlicherweise ein OR- oder AND-Split verwendet wurde.

Eine analoge Aussage gilt für den Fall, dass einander widersprechende Ereignisse direkt vor einem OR- bzw. AND-Join stehen.

Es ist anzumerken, dass sich die beiden Ereignisse auch logisch ausschließen können, wenn nicht eines die Negation des anderen ist. Ein Vorliegen des Musters soll auch erkannt werden, wenn z.B. eine Kombination von Ereignissen wie „Der Wert x hat zugenommen“ / „Der Wert x hat abgenommen“ gefunden wird, auch wenn außerdem noch der dritte Fall „Der Wert x ist konstant geblieben“ möglich wäre.

2.3 Ereignis, dessen Negation und weiteres Ereignis am XOR-Konnektor vereint

Muster C	
	Ein Konnektor vom Typ XOR hat die Ereignisse A, $\neg A$ sowie ein weiteres Ereignis B als Vorgänger oder Nachfolger. Da stets entweder A oder $\neg A$ eintritt, ist die Berechtigung des dritten Ereignisses B fraglich (*Tertium non datur*).

In diesem Muster folgen auf einen XOR-Split ein Ereignis A, dessen Negation $\neg A$ und mindestens ein weiteres Ereignis B. Da stets eines der Komplementärereignisse A und $\neg A$ eintreten muss, sollte es eigentlich keine Berechtigung für ein drittes Ereignis B geben.

Das Modellfragment in Abb. 5 (entnommen dem Zeitschriftenartikel [GK00]) zeigt ein Beispiel für das Auftreten des Musters. In einem solchen Modell leidet zumindest die Verständlichkeit[2]: Da offenbar stets genau eines der beiden linken Ereignisse eintritt, ist die Berechtigung der beiden anderen Ereignisse unklar.

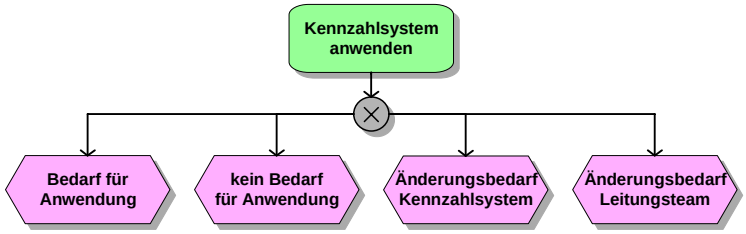


Abbildung 5: Es tritt stets eines der beiden linken Ereignisse ein.

Einen möglichen durch dieses Muster erzeugten „Fehlalarm“ zeigt Abb. 6. Hier entsteht der Eindruck eines Fehlers dadurch, dass die Aussage „CR ist vorhanden und (un)vollständig“ verkürzt wurden auf „CR ist (un)vollständig“. Für den Leser des Modells dürfte diese verkürzte Schreibweise sogar leichter verständlich sein.

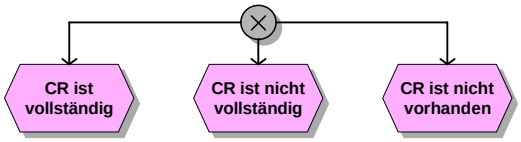


Abbildung 6: Fehlalarm bei Muster C

In Fällen wie in Abb. 6 ist es kaum möglich, Fehlalarme auszuschließen. Dies gelingt jedoch in einem häufigen Spezialfall, nämlich dann, wenn die Ereignisbeschriftungen A, $\neg A$ und B Entscheidungen der Art „ja“ / „nein“ / „unter Umständen“ beschreiben. Um auch solche Situationen richtig zu behandeln, meldet unsere Mustersuche in den Fällen,

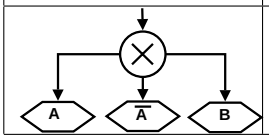
[2] Den fachlichen Inhalt des Modellausschnitts erlauben wir uns nicht zu beurteilen.

in denen B einschränkende Begriffe wie „zum Teil“, „eventuell“ oder „unter Vorbehalt“ enthält, kein Auftreten des.

In den im Rahmen der Validierung untersuchten Praxis-Modellen (vgl. Abschnitt 4) fanden sich die folgenden Kombinationen von Ereignissen nach einem XOR-Split, für die *kein* Auftreten des Musters gemeldet wird:

- „Berichte sind in Ordnung“ / „Berichte sind nicht in Ordnung“ / Berichte sind nur teilweise in Ordnung“
- „Bestellstatus zurückgewiesen“ / „Bestellstatus zugestimmt“ / „Bestellstatus teilweise zugestimmt“

2.4 Vergessen des Falles „Gleichheit“ beim Vergleich von Werten

Muster D	Ereignisse nach einem Konnektor beschreiben Größenvergleiche der Form “$a < b$” / “$a > b$” oder Größenveränderungen der Form “a ist gestiegen” / “a ist gesunken”
	Es besteht die Möglichkeit, dass bei der Modellierung der Fall der Gleichheit (“$a < b$”) bzw. des Gleichbleibens (“a blieb unverändert”) vergessen wurde.

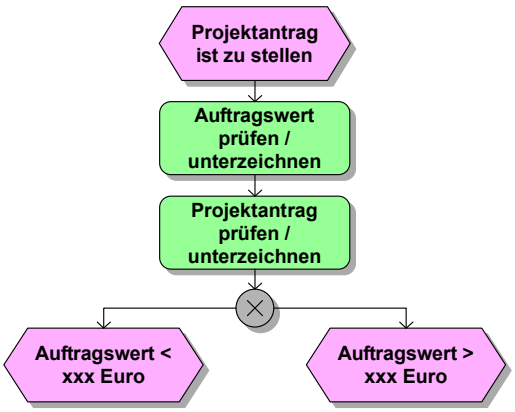


Abbildung 7: Vergessen des Falles „Gleichheit“ beim Vergleich von Werten

Als Ereignisse, deren Eintreten über die weitere Abarbeitung eines EPK-Modells nach einem Konnektor entscheidet, dienen in manchen Fällen Vergleiche von Werten.

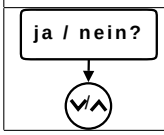
Im Beispiel von Abb. 7 (entnommen dem Buch [BKR08, Seite 634]) wird der Projektantrag sofort eingereicht, wenn der Auftragswert kleiner als ein bestimmter Betrag ist. Ist er größer als der genannte Betrag, muss eine zusätzliche Prüfung erfolgen. Im beschriebenen

Falle ist zu vermuten, dass der Modellierer den Fall „Auftragswert ist genau xxx Euro“ vergessen hat (vgl. [Amb03], Regel 233).

Unser Werkzeug identifiziert solche Situationen in einem Modell und weist auf ein mögliches Problem hin.

Analog untersuchen wir Aussagen zur Veränderung von Größen. Folgen etwa auf einen Split die beiden Ereignisse „Bedarf ist angestiegen“ und „Bedarf ist gesunken“ erfolgt ein Hinweis, dass der Fall „Bedarf ist gleich geblieben“ eventuell bei der Modellierung vergessen wurde.

2.5 AND- oder OR-Split nach einer Entscheidungsfrage

Muster E	
	Auf eine Entscheidungsfrage folgt ein AND- oder OR-Split, so dass laut Modell mehrere Ereignisse zugleich auftreten können.

In der Regel sollte nach einer Entscheidungsfrage (also einer Frage, auf die entweder mit “ja” oder mit “nein” geantwortet werden kann), ein XOR-Split folgen. Gleiches gilt für Fragen der Art “Prüfe, ob x oder y”. Unser Algorithmus identifiziert Fragen der genannten Art, denen ein AND- oder OR-Split folgt und gibt einen entsprechenden Hinweis aus.

Unser Algorithmus findet z.B. den Modellabschnitt von Abb. 8, bei dem nach der Überprüfung korrekterweise ein OR-Split stehen sollte[3].

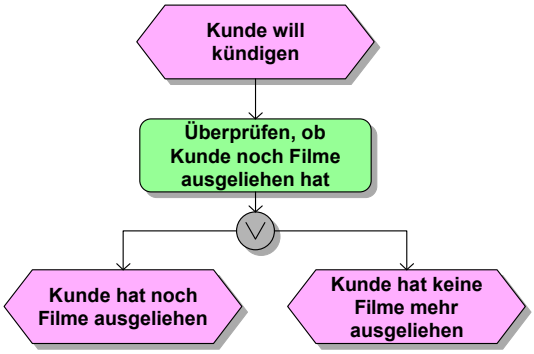


Abbildung 8: Nach einer solchen Entscheidung sollte immer ein XOR-Split stehen

[3] Im gezeigten Modell liegt außerdem Muster B vor; dies muss jedoch nicht immer der Fall sein.

3 Algorithmus

Wir benutze den in [GL06] eingeführten Ansatz, die in einer ereignisgesteuerten Prozesskette (EPK) enthaltenen Informationen in eine Prolog-Faktenbasis zu übersetzen und durch Abfragen an den Prolog-Interpreter Fehler im Modell zu lokalisieren.

In bisherigen Arbeiten wurde auf diese Weise die syntaktische Korrektheit [GL06], Freiheit von Deadlocks und Kontrollflussfehlern [GLKK08] sowie mögliche Modellvereinfachungen, die zu einer leichteren Erfassbarkeit einer EPK führen [GL09], untersucht.

Die grundsätzlichen Schritte bei einer logikbasierten Analyse sind:

1. Übersetzung der im Modell enthaltenen Informationen in eine Prolog-Faktenbasis. Diese enthält dann Aussagen wie `event(i_3)` (es gibt ein Ereignis mit der ID i_3) oder `elementname(i_3,"Fahrzeug betanken")` (das Modellelement mit der ID i_3 ist mit dem Text "Fahrzeug betanken" beschriftet).
2. Überführen der Beschriftungen der Modellelemente in eine Normalform, so dass Beschriftungen gleicher Bedeutung in dieselbe Normalform überführt werden
3. Suche nach den beschriebenen Mustern mittels Abfragen an den Prolog-Interpreter

3.1 Schritt 1: Überführen des Modells in eine Prolog-Faktenbasis

Die Überführung von EPK-Modellen in eine Prolog-Faktenbasis erfolgt mittels eines XSLT-Skripts, dass eine EPML-Datei in Prolog-Regeln überführt. Details sind in [GL06] zu finden.

3.2 Schritt 2: Erzeugen einer Normalform der Modellbeschriftungen

In Schritt 2 werden die Beschriftungen der Modellelemente in eine Normalform gebracht.

Zweck dieser Normalform ist es, dass Beschriftungen gleicher Bedeutung in die gleiche Normalform überführt werden. Zu diesem Zwecke werden Stoppwörter, Synonyme und Antonyme beachtet. In unserem prototypischen Werkzeug haben wir 210 Synonyme und 70 Antonyme zu Begriffen aufgenommen, die sich sehr häufig in Modellen betrieblicher Abläufe finden.

Um die Population des Synonym/Antonym-Katalogs aufzubauen, wurde zunächst untersucht, welche Begriffe in Modellen eines uns vorliegenden EPK-Katalogs am häufigsten vorkommen. Hierfür wurden insgesamt 1154 deutschsprachige EPKs (darunter die 749 EPKs des deutschsprachigen SAP R/3-Referenzmodells) ausgewertet. Deren Beschriftungen von Ereignissen und Funktionen enthielten insgesamt 66088 Wörter, darunter 7599 verschiedene. Unter den am häufigsten gefundenen Begriffen wurden, sofern sinnvoll, Wörter zu Gruppen von Synonymen zusammengefasst. Die in einer solchen Gruppe enthaltenen Wörter werden dann bei der Normalformbildung alle durch die selbe Zeichenkette dargestellt.

Um die Normalform zu erhalten, bildet unser Algorithmus durch wiederholte Ersetzung von Zeichenketten eine Normalform einer Modellbeschriftung, indem Begriffe mit gleicher Bedeutung in die selbe Zeichenkette überführt werden. In den beiden Ereignisbeschriftungen „Der Antrag ist genehmigt“ und „Dem Antrag wurde zugestimmt“ werden zunächst Stoppwörter wie *der*, *die*, *das*, *ein*, *eine* u.ä. durch die leere Zeichenkette ersetzt, da sie für die Erfassung der Bedeutung der Zeichenkette verzichtbar sind. Da die Wörter “genehmigt” und “zugestimmt” beide im Synonymkatalog enthalten sind, werden weiterhin beide Zeichenketten in die selbe Normalform “Antrag nf_genehmigt” überführt.

Auf die gleiche Weise werden Zeichenketten ersetzt, die zum Antonym-Katalog enthalten sind. In diesem Falle wird außerdem ein Flag gesetzt, das ausdrückt, dass die Normalform das Gegenteil der ursprünglichen Zeichenkette darstellt. Auf diese Weise wird etwa die Zeichenkette “Der Antrag wurde abgelehnt” ebenfalls in die Normalform “Antrag nf_genehmigt” überführt. Am gesetzten Flag ist jedoch erkennbar, dass sie das Gegenteil aussagt.

Tabelle 1 zeigt die in unserem Korpus von EPK-Modellen am häufigsten gefundenen Wörter sowie (falls zutreffend) ihre Ersetzung im Zuge der Normalform-Bildung:

Vorkommen	Wort	Behandlung
5719	ist	ersetzen durch leere Zeichenkette (Stoppwort)
1774	sind	ersetzen durch leere Zeichenkette (Stoppwort)
1400	zu	
821	für	
718	der	ersetzen durch leere Zeichenkette (Stoppwort)
696	und	ersetzen durch nf_und
645	vor	in Kombinationen wie “liegt vor” ersetzen durch nf_vorhanden
536	wurde	ersetzen durch leere Zeichenkette (Stoppwort)
527	nicht	ersetzen durch leere Zeichenkette und Flag “Aussage ist negiert” setzen
494	liegt	in Kombinationen wie “liegt vor” ersetzen durch nf_vorhanden
477	an	
455	durchgeführt	ersetzen durch nf_beendet
404	werden	ersetzen durch leere Zeichenkette (Stoppwort)

Tabelle 1: Behandlung der häufigsten Begriffe bei der Normalformbildung

Eine besondere Behandlung erfahren Ereignisbeschriftungen der Form $x \diamond y$ mit $\diamond \in \{<, >, =, \leq, \geq, \}$ sowie Formen wie “x hat sich erhöht”, “x verringerte sich”, “x ist gefallen”, “x ist gestiegen”, “x blieb konstant” etc. Auf eine Beschreibung der Details soll an dieser Stelle verzichtet werden. Die folgenden Beispiele illustrieren jedoch, dass als Resultat der Normalformbildung festgestellt werden kann, dass

- “$x > 1000$” ebenso wie “$1000 > x$” den Aussagen “$x < 1000$” oder “x ist gleich1000” widerspricht.

- “x ist gestiegen” und “x hat sich erhöht” die gleiche Aussage darstellen, jedoch im Widerspruch zu “x blieb konstant” und “x verringerte sich” stehen.
- die Aussagen “x ist kleiner als y” und “x ist größer als y” zwei der drei möglichen Falle “größer/kleiner/gleich” beschreiben.

3.2.1 Schritt 3: Abfragen an die Faktenbasis

Aus der Beschreibung der in Abschnitt 2 betrachteten Muster wird deutlich, dass zur Identifikation der beschriebenen Muster die folgenden logischen Abfragen ausreichend sind:

1. **Vorgänger- und Nachfolgerbeziehung zwischen Ereignis und Konnektor:**
   Diese wird einfach durch das Vorhandensein eines Kontrollflusspfeiles ausgedrückt, was sich in der Prolog-Repräsentation des Modells durch ein Prädikat `arc(x,y)` widerspiegelt.
2. **Zwei Ereignisse beschreiben den gleichen Sachverhalt:**
   Dies führt, wie in Schritt 2 dargestellt, dazu, dass beide Beschriftungen in die selbe Normalform überführt wurden. Eine Meldung zu Problem A wird nur ausgegeben, wenn die Ereignisse nicht eine triviale Beschriftung wie “erledigt” oder “fertig” haben.
3. **Zwei Ereignisbeschriftungen A und B widersprechen einander:**
   Dies ist der Fall, wenn A und B in die selbe Normalform überführt wurden, jedoch bei der Ersetzung das Flag gesetzt wurde, das auf eine Ersetzung aus dem Antonymkatalog hinweist.
   Ebenso ist dies der Fall, wenn A aus B hervorgeht, indem eine der negierenden Zeichenketten “un”, “nicht” bzw. “nicht-” eingefügt wurde.
   Schließlich widersprechen sich Ereignisbeschriftungen, wenn sie genau zwei der drei Fälle “kleiner/größer/gleich” oder “verringert/vergrößert/unverändert” darstellen.
4. **Eine Beschriftung weist darauf hin, dass ein Ereignis nur teilweise eintritt:**
   Dies wird angenommen, wenn die Ereignisbeschriftung bestimmte einschränkende Zeichenketten wie z.B. “möglicherweise”, “eventuell”, “vielleicht”, “möglichenfalls”, “unter Umständen”, “u.U.”, “unter Vorbehalt(en)”, “zum Teil”, “z.T.”, “teilweise”, “partiell” oder “unvollständig” enthält.
5. **Eine Beschriftung stellt eine Entscheidungsfrage (ja/nein-Frage) dar:**
   Dies wird angenommen, wenn die Beschriftung einer Funktion die Zeichenkette “, ob” enthält (“Prüfe, ob der Auftrag ausgeführt wurde”) sowie wenn die Beschriftung mit einem Fragezeichen endet, jedoch nicht mit einem Fragewort (“wer”, “womit”, etc.) beginnt. Als Entscheidungsfrage wird somit z.B. “Erfolgte der Widerspruch rechtzeitig?” eingestuft, jedoch nicht “Welche Zusatzoptionen werden gewünscht?”.

Mit diesen genannten Prädikaten sowie weiteren einfachen Prädikaten, die z.B. bestimmen, ob ein Konnektor Split oder Join ist (siehe auch [GL06]) lässt sich z.B. eine Abfrage nach Muster A für zwei Ereignisse nach einem XOR-Split (was auf einen ernsten Modellierungsfehler hinweist) in Prolog wie folgt formulieren:

```
findemuster(E1,E2) :-
split(C),type(C,xor),          \% C ist ein XOR-Split...
arc(C,E1),arc(C,E2),           \% von dem aus es Pfeile zu E1 und E2 gibt
event(E1),event(E2),           \% E1 und E2 sind Ereignisse
E1 @< E2,                      \% bedeutet insbesondere: E1 ungleich E2
elementname(E1,NameE1),        \% E1 hat die Beschriftung NameE1
elementname(E2,NameE2),        \% E2 hat die Beschriftung NameE2
equivalent(NameE1,NameE2),     \% NameE1 und NameE2 sind logisch äquivalent
not(trivialereignis(NameE1)), \% NameE1 ist kein Trivialereignis
not(trivialereignis(NameE2)). \% NameE2 ist kein Trivialereignis
```

4 Validierung

Wir überprüften unser Verfahren an 1253 EPK-Modellen in deutscher Sprache, die wir aus verschiedenen Quellen zusammengetragen haben.

Dabei stammten:

- 591 EPKs aus dem deutschsprachigen SAP R/3 Referenzmodell
- 127 EPKs aus Büchern (vornehmlich Lehrbücher zur EPK-Methode oder zu SAP R/3)
- 48 EPKs aus Dissertationsschriften
- 70 EPKs aus wissenschaftlichen Veröffentlichungen in Zeitschriften und Konferenzbänden
- 252 aus Bachelor-, Diplom- und Seminararbeiten
- 84 EPKs aus Praxisprojekten
- 22 EPKs aus Vorlesungsskripts
- 13 EPKs aus Software-Handbüchern

Bei den verbleibenden 37 Modelle handelt es sich um im Internet veröffentlichte EPKs, deren Einordnung in die o.g. Kategorien nicht möglich ist. Durch die Einbeziehung der verschiedenen Quellen ist zu erwarten, dass Modelle von Autoren mit höchst unterschiedlichen Erfahrungen mit der EPK-Modellierungsmethode berücksichtigt wurden.

Alle vom Werkzeug gemeldeten Problemmeldungen wurden durch manuelle Prüfung der betroffenen EPK untersucht, um zu entscheiden, ob tatsächlich ein Modellierungsproblem vorliegt. Insbesondere bei Muster C war dies nicht immer eindeutig zu entscheiden; in Zweifelsfällen wurde die Fehlermeldung als “unberechtigt” eingestuft. Es ergab sich die folgende Verteilung von gefundenen Fehlern bzw. “Fehlalarmen”:

Insgesamt wurden 114 tatsächliche sinnvolle Hinweise in 84 EPKs erkannt. Dem stehen 14 “Fehlalarme” in 13 EPKs entgegen. Die zahlreichen Fehlalarme bei Muster C legen nahe, dass eine Untersuchung dieses Musters in der Praxis nicht unbedingt sinnvoll ist. Bei allen anderen Mustern zeigen die gemeldeten Musterinstanzen fast ausnahmslos tatsächliche

Muster	gemeldete Vorkommen des Musters	davon unberechtigte Fehlermeldungen
Muster A	71 in 55 EPKs	1 in 1 EPK
Muster B	31 in 19 EPKs	1 in 1 EPK
Muster C	21 in 18 EPKs	12 in 11 EPKs
Muster D	3 in 3 EPKs	keine
Muster E	2 in 2 EPKs	keine

Tabelle 2: Gefundene Fehler in den einzelnen Fehlerklassen

Fehler bzw. Modellierungsprobleme. Dem Modellierer dürfte somit geholfen sein, wenn er zur Modellierungszeit eine Rückmeldung über Auftreten der genannten Muster und ggf. möglichen Modellverbesserungen erhält.

Bemerkenswert ist der Zusammenhang zwischen den gefundenen Fehlern und der Herkunft der Modelle. So stellt Muster B einen typischen Anfängerfehler dar - statt eines XOR-Splits wird das umgangssprachlich naheliegende OR verwendet. Während dieser Fehler in den studentischen Arbeiten (wie auch in einem Praxisprojekt aus dem Bereich Medien) recht häufig auftritt, wurde kein einziges Vorkommen dieses Musters im SAP R/3-Referenzmodell gefunden. Diese Beobachtung zeigt, dass von einem automatischen Erkennen der beschriebenen Fehlermuster hauptsächlich Anfänger profitieren dürften.

5 Diskussion und Vergleich mit verwandten Arbeiten

Das im vorangehenden Abschnitt beschriebene Ergebnis belegt, dass sich durch eine Analyse der Beschriftung von Modellelementen eine nennenswerte Zahl von Modellierungsfehlern aufspüren lässt. Diese Modellierungsfehler bleiben bei bloßer Betrachtung des Kontrollflusses unentdeckt.

Unser Algorithmus setzt keine Beschränkung der im Modell zu verwendenden Elemente der natürlichen Sprache voraus. Ereignisse und Funktionen können durch beliebigen Freitext beschrieben werden, was der heute meist gängigen EPK-Modellierungsmethode entspricht. Weit bessere Ergebnisse dürften zu erwarten sein, wenn die natürliche Sprache, die zur Beschreibung von Ereignissen und Funktionen verwendet wird, beschränkt wird.

Eine Vereinheitlichung der Beschriftungen von Modellierungselementen leistet etwa das Werkzeug Semtalk [FW05], das Ontologien verwendet, um eine einheitliche Verwendung von Substantiven und Verben über ein oder mehrere Modelle hinweg zu gewährleisten. Bei Verwendung eines solchen ontologiebasierten Konzeptes ist es auch möglich, echte inhaltliche Prüfungen von Geschäftsregeln automatisiert vorzunehmen. Fillies und Weichhardt führen in [FW03] ein Beispiel an, in dem zwei Geschäftsprozessmodelle „Bestellungseingang“ und „Bestellungsbearbeitung“ betrachtet werden. Sie nennen die Geschäftsregel „Nur bestätigte Bestellungen dürfen ausgeführt werden“ als Beispiel einer Regel, die man mit Hilfe solcher ontologiebasierter Modellierung modellübergreifend prüfen kann. Ein ähnliches Beispiel nennen Thomas und Fellmann. Sie beschreiben in in [TF07] einen Ansatz, Modellierung betrieblicher Abläufe mit EPKs mit Ontologien zu verknüpfen.

Wir sind davon überzeugt, dass durch die Einbindung von Ontologien in Geschäftsprozessmodellierungsmethoden mächtige Werkzeuge zur Konsistenzprüfung und Validierung von Geschäftsprozessmodellen sowie für Abfragen in Modellkatalogen geschaffen werden können. Vorhandene Ansätze werden beispielsweise in [WHM08] oder [GHSW08] beschrieben.

Wir sind jedoch ebenso davon überzeugt, dass sich in der betrieblichen Praxis solche ontologiebasierte Verfahren schwer durchsetzen werden, da dem Ziel (Verbesserung der Modellqualität) ein zumindest in der Einführungsphase erhöhter Aufwand im Modellierungsprozess entgegensteht. Letzlich sehen die beschriebenen Verfahren vor, eine Domänenontologie zusätzlich zum Modell zu erstellen, was in der Regel per Hand erfolgt (vgl. etwa die Beschreibung zur Erstellung von sog. semantischen EPKs in [FKS08]).

Unser Verfahren verzichtet auf eine aufwendige Erstellung einer Domänenontologie. Lediglich die im Standard-Synonym/Antonym-Katalog enthaltenen Begriffe stellen eine einfache Art einer (allerdings unvollständigen) Ontologiebeschreibung dar. In der aktuellen Form hat unser Synonym/Antonymkatalog noch einen recht geringen Umfang, so dass wir keinen Anspruch auf vollständige Erkennung der untersuchten Problemmuster erheben können. Ebenso gibt es sicher neben den betrachteten häufigen Problemmustern noch weitere.

Dem Nachteil dieser Unvollständigkeit steht der Vorzug der für den Modellierer einfachen Verfügbarkeit gegenüber. In der Validierung wurde gezeigt, dass sich bereits mit einem kleinen Katalog von Synonymen und Antomymen eine bemerkenswerte Zahl von Modellfehlern finden lässt. Im Unterschied zu Verfahren, die eine ontologiebasierte Modellierung verlangen, erhält der Modellierer Informationen zu diesen Fehlern, ohne dass die Modellierungsmethode komplexer wird. Im Werkzeug *bflow* Toolbox* ist die Mustersuche fest eingebaut und kann "auf Knopfdruck" gestartet werden (siehe Bildschirmfoto in Abb. 9). Dies erweitert die in [GLKK08] beschriebenen Möglichkeiten der *bflow* Toolbox*, dem Modellierer zur Modellierungszeit Rückmeldungen über mögliche Modellverbesserungen zu geben.

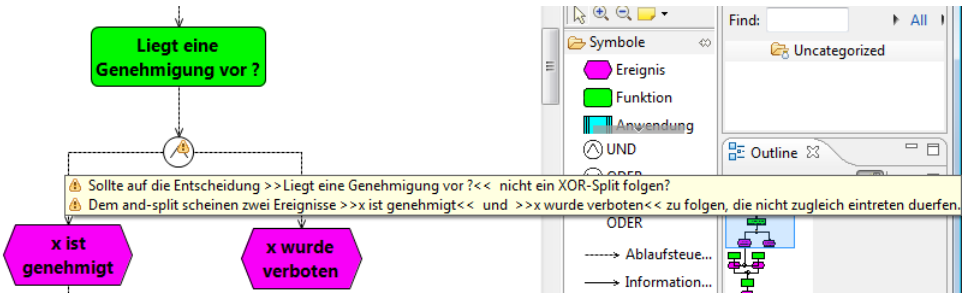


Abbildung 9: Integration in die bflow Toolbox, gemeldet werden hier Muster B und Muster E

In [ADW08] verwenden Awad et al. Verfahren aus dem Gebiet des Information Retrieval, um ein Ähnlichkeitsmaß zwischen Namen von Aktivitäten innerhalb eines BPMN-Diagramms zu definieren. Dieser Ansatz, der auf die englische Wortschatz-Datenbank WordNet [Fel98] zurückgreift, kommt ohne Beschränkung des zur Beschreibung von Aktivitäten verwendbaren Wortschatzes aus. Er erzielt gute Ergebnisse beim Erkennen gleicher oder ähnlicher Aktivitäten, auch wenn diese nicht identisch bezeichnet sind. Eine Kombination des in [ADW08] beschriebenen Verfahrens mit dem von der gleichen Forschungsgruppe beschriebenen Validierungsansatz [ADW08] erlaubt auch eine Überprüfung inhaltlicher Aussagen wie „Es darf kein Konto eröffnet werden, bevor die Identität des Inhabers überprüft wurde“.

Einen zu [ADW08] ähnlichen Ansatz beschreibt [KO07], hier steht jedoch nicht die Validierung von Modellen sondern das Erkennen von Modellvarianten im Vordergrund.

Ein wesentlicher Unterschied zwischen unserem Ansatz und [ADW08] sowie [KO07] besteht darin, dass wir bereits mit einem sehr kleinen Katalog von Synonymen und Antonymen beachtliche Ergebnisse erzielen können. Generell ist jedoch eine Verbesserung der Fehlererkennung zu erhoffen, wenn statt unseres eher einfachen Ansatzes zur Erkennung von identischen bzw. negierten Aussagen mächtigere Verfahren (wie in [ADW08], [KO07] oder [GZ05] beschrieben) verwendet werden.

Ein weiteres Feld für künftige Erweiterungen ist die Überprüfung der Einhaltung von Modellierungskonventionen für die Beschriftung von Ereignissen und Funktionen. So sollten Ereignisse etwa durch ein adjektivisch verwendetes Partizip („Der Antrag wurde genehmigt“) und Funktionen durch den Infinitiv eines Verbs („Antrag genehmigen“) dargestellt werden. Abweichende Modellierungskonventionen („Der Antrag ist zu genehmigen“ / „Der Antrag wird genehmigt“) sind möglich. Bögl et al. zeigen in [BSPW08], wie mit Hilfe von Wortdatenbanken und semantischen Mustern für Modellbeschriftungen die Einhaltung von Modellierungskonventionen wirkungsvoll überprüft werden kann.

Wir planen, unseren Ansatz in Zukunft um weitere Muster, die mögliche Modellverbesserungen beschreiben, zu erweitern. Forscher und Praktiker sind eingeladen, die im Werkzeug *bflow* Toolbox* bereits integrierten Tests zu nutzen und zu erweitern. Rückmeldungen und Erweiterungsvorschläge sind herzlich willkommen. Der jeweils aktuelle Entwicklungsstand kann von der Website `www.bflow.org`[4] heruntergeladen werden.

[4] Das Prolog-Programm befindet sich im Plugin org.bflow.toolbox.prolog, dort findet der interessierte Leser auch die Liste der Synonyme und Antonyme.

Literatur

[ADW08] Ahmed Awad, Gero Decker und Mathias Weske. Efficient Compliance Checking Using BPMN-Q and Temporal Logic. In *BPM '08: Proceedings of the 6th International Conference on Business Process Management*, Seiten 326–341, Berlin, Heidelberg, 2008. Springer-Verlag.

[Amb03] Scott W. Ambler. *The Elements of UML Style*. Cambridge University Press, 2003.

[BKR08] Jörg Becker, Martin Kugeler und Michael Rosemann. *Prozessmanagement. Ein Leitfaden zur prozessorientierten Organisationsgestaltung*. Springer-Verlag, 6. Auflage, 2008.

[BRS95] Jörg Becker, Michael Rosemann und Reinhard Schütte. Grundsätze ordnungsgemäßer Modellierung. *Wirtschaftsinformatik*, 37(5):435–445, 1995.

[BSPW08] Andreas Bögl, Michael Schrefl, Gustav Pomberger und Norbert Weber. Semantic Annotation of EPC Models in Engineering Domains by Employing Semantic Patterns. In *ICEIS 2008 - Proceedings of the Tenth International Conference on Enterprise Information Systems, Volume AIDSS, Barcelona, Spain*, Seiten 106–115, 2008.

[DBL07] *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management held in conjunction with the 3rd European Semantic Web Conference (ESWC 2007), Innsbruck, Austria, June 7, 2007*, Jgg. 251 of *CEUR Workshop Proceedings*, 2007.

[EGS04] Werner Esswein, Andreas Gehlert und Grit Seiffert. Towards a Framework for Model Migration. In *Advanced Information Systems Engineering, 16th International Conference, CAiSE 2004, Riga, Latvia, June 7-11, 2004, Proceedings*, Jgg. 3084 of *Lecture Notes in Computer Science*, Seiten 463–476. Springer, 2004.

[Fel98] Christiane Fellbaum, Hrsg. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, May 1998.

[FKS08] Agata Filipowska, Monika Kaczmarek und Sebastian Stein. Semantically Annotated EPC within Semantic Business Process Management. In Danilo Ardagna, Massimo Mecella und Jian Yang, Hrsg., *Business Process Management Workshops*, Jgg. 17 of *Lecture Notes in Business Information Processing*, Seiten 486–497. Springer, 2008.

[FW03] Christian Fillies und Frauke Weichhardt. Towards the Corporate Semantic Process Web. In *Berliner XML Tage*, Seiten 78–90, 2003.

[FW05] Christian Fillies und Frauke Weichhardt. On Ontology-based Event-driven Process Chains. In *EPK 2005, Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten*, 2005.

[GHSW08] Guido Governatori, Jörg Hoffmann, Shazia Sadiq und Ingo Weber. Detecting Regulatory Compliance for Business Process Models through Semantic Annotations. In *BPD-08: 4th International Workshop on Business Process Design*, September 2008.

[GK00] Frank Giesa und Herbert Kopfer. Management logistischer Dienstleistungen der Kontraktlogistik. *Logistik Management*, 2(1):43–53, 2000.

[GKMZ07] Guido Grohmann, Wolfgang Kraemer, Frank Milius und Volker Zimmermann. Modellbasiertes Curriculum-Design für Learning Management Systeme: Ein Integrationsansatz auf Basis von ARIS und IMS Learning Design. In *Wirtschaftsinformatik*, Seiten 795–812. Universitätsverlag Karlsruhe, 2007.

[GL06] Volker Gruhn und Ralf Laue. Validierung syntaktischer und anderer EPK-Eigenschaften mit PROLOG. In *EPK 2006, Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, 5. Workshop der Gesellschaft für Informatik e.V.*, Seiten 69–84, 2006.

[GL09] Volker Grund und Ralf Laue. Reducing the Cognitive Complexity of Business Process Models. In *IEEE International Conference on Cognitive Informatics, Hong Kong 2009*, 2009.

[GLKK08] Volker Gruhn, Ralf Laue, Heiko Kern und Stefan Kühne. EPK-Validierung zur Modellierungszeit in der bflow* Toolbox. In Peter Loos, Markus Nüttgens, Klaus Turowski und Dirk Werth, Hrsg., *MobIS*, Jgg. 141 of *LNI*, Seiten 181–194. GI, 2008.

[GZ05] Vincenzo Gervasi und Didar Zowghi. Reasoning about inconsistencies in natural language requirements. *ACM Trans. Softw. Eng. Methodol.*, 14(3):277–330, 2005.

[KÖ6] Markus König. Workflow-Management in der Baupraxis. In *4. Tag des Baubetriebs 2004 - Tagungsbeiträge Nachtragsmanagement in Praxis und Forschung, Schriften der Professur Baubetrieb und Bauverfahren*. Bauhaus-Universität Weimar, 2006.

[KO07] Agnes Koschmider und Andreas Oberweis. How to detect semantic business process model variants? In *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*, Seiten 1263–1264, New York, USA, 2007. ACM.

[Men07] Jan Mendling. *Detection and Prediction of Errors in EPC Business Process Models*. Dissertation, Wirtschaftsuniversität Wien, 2007.

[PN05] Daniel Pfeiffer und Björn Niehaves. Evaluation of Conceptual Models - A Structuralist Approach. In *Proceedings of the 13th European Conference on Information Systems, Information Systems in a Rapidly Changing Economy, ECIS 2005, Regensburg, Germany, May 26-28, 2005*, 2005.

[Ron97] Ron Weber. Ontological Foundations of Information Systems. Bericht 4, Coopers and Lybrand Accounting Research Methodology monograph, 1997.

[Rum99] Frank J. Rump. *Geschäftsprozeßmanagement auf der Basis ereignisgesteuerter Prozeßketten*. B. G. Teubner Verlag Stuttgart Leipzig, 1999.

[TF07] Oliver Thomas und Michael Fellmann. Semantic EPC: Enhancing Process Modeling Using Ontology Languages. In *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management held in conjunction with the 3rd European Semantic Web Conference (ESWC 2007), Innsbruck, Austria, June 7, 2007* [DBL07].

[van97] Wil M. P. van der Aalst. Verification of Workflow Nets. In *Application and Theory of Petri Nets 1997, 18th International Conference, ICATPN '97, Toulouse, France, June 23-27, 1997, Proceedings*, Seiten 407–426, 1997.

[WHM08] Ingo Weber, Jörg Hoffmann und Jan Mendling. Semantic Business Process Validation. In *SBPM-08: 3rd international workshop on Semantic Business Process Management at ESWC-08*, Juni 2008.