Markus Nüttgens, Frank J. Rump (Hrsg.)

EPK 2003

Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten

2. Workshop der Gesellschaft für Informatik e.V. (GI)
und Treffen ihres Arbeitkreises „Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (WI-EPK)“

08. Oktober 2003 in Bamberg

Proceedings

Veranstalter

veranstaltet vom GI-Arbeitskreis "Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (WI-EPK)" der GI-Fachgruppe WI-MobIS (FB-WI) in Kopperation mit der GI-Fachgruppe EMISA (FB-DBIS) und der GI-Fachgruppe Petrinetze (FB-GInf).

Dr. Markus Nüttgens (Sprecher)
Email: markus@nuettgens.de

Prof. Dr. Frank J. Rump (stellv. Sprecher)
Email: rump@informatik-emden.de

EPK 2003 / Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten. Hrsg.: Markus Nüttgens, Frank J. Rump. – Bamberg 2003

Vorwort

Ereignisgesteuerte Prozessketten (EPK) haben sich in der Praxis als Beschreibungsmittel für betriebliche Abläufe etabliert. Mit dem Aufbau der Arbeitsgruppe "Formalisierung und Analyse Ereignisgesteuerter Prozessketten (EPK)" im Jahre 1997 wurde ein erster Schritt unternommen, einen organisatorischen Rahmen für Interessenten und Autoren wesentlicher Forschungsarbeiten zu schaffen und regelmäßige Arbeitstreffen durchzuführen (Organisatoren: Markus Nüttgens, Andreas Oberweis, Frank J. Rump). Im Jahr 2002 wurden die Arbeiten der "informellen“ Arbeitsgruppe in den GI-Arbeitskreis "Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (WI-EPK)" der GI-Fachgruppe WI-MobIS (FB-WI) überführt und inhaltlich erweitert. Der 1. Workshop „EPK 2002 - Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten“ fand im November 2002 in Trier statt.

Der Arbeitskreis soll Praktikern und Wissenschaftlern als Forum zur Kontaktaufnahme, zur Diskussion und zum Informationsaustausch dienen. Die Aktivitäten des Arbeitskreises werden unter der Internetadresse http://www.epk-community.de dokumentiert (aktuell: 120 Mitglieder).

Der vorliegende Tagungsband enthält sieben vom Programmkomitee ausgewählte und auf dem Workshop präsentierte Beiträge. Jeder Beitrag wurde innerhalb der Kategorien Fachbeiträge und Diskussionsbeiträge zweifach begutachtet.

Die Beiträge decken ein breites Spektrum zur Spezifikation und Anwendung Ereignisgesteuerter Prozessketten (EPK) ab:

- Rahmenkonzept für eine EPK-Semantik,
- EPK-Syntax Validierung mit XML Schemasprachen,
- Prozessmodellierung in eGovernment-Projekten mit der eEPK,
- Multiperspektivische EPK,
- Ontologische Evaluierung von Ereignisgesteuerten Prozessketten,
- Dokumentenmanagement-Prozesse mit EPKs,
- Kundenorientierte Dienstleistungsmodellierung mit EPKs.

Der Tagungsband wurde ausschließlich in digitaler Form publiziert und ist frei verfügbar unter: http://www.epk-community.de/epk2003/epk2003-proceedings.pdf.

Wir danken den AutorInnen, den Mitgliedern des Programmkomitees und dem lokalen Organisationsteam der MobIS 2003 für die Beträge zur Realisierung des Workshops.

Saarbrücken und Emden, im Oktober 2003

Markus Nüttgens
Frank J. Rump

Programmkomitee

Jörg Becker, Universität Münster
Jörg Desel, Katholische Universität Eichstätt
Gerhard Keller, St. Leon-Rot
Ekkart Kindler, Universität Paderborn
Peter Loos, Universität Mainz
Markus Nüttgens, Universität des Saarlandes (Vorsitz)
Andreas Oberweis, Universität Frankfurt a.M.
Peter Rittgen, University College of Borås, Schweden
Frank Rump, Hochschule Weser-Ems
Reinhard Schütte, Universität Essen
Oliver Thomas, Universität des Saarlandes

Organisation

Markus Nüttgens, Universität des Saarlandes
Frank J. Rump, Hochschule Weser-Ems

Inhaltsverzeichnis

Fachbeiträge

Diskussionsbeiträge

On the semantics of EPCs: A framework for resolving the vicious circle Extended abstract*

Ekkart Kindler
Computer Science Department, University of Paderborn, Germany
kindler@upb.de

Abstract: One of the most debatable features of *Event driven Process Chains* (EPCs) is their non-local semantics. Most non-local semantics for EPCs either have a formal flaw or are given informally only. In fact, it can be shown that there is no formal semantics that precisely captures the informal idea of the non-local semantics of EPCs.

In this paper, we formally define a non-local semantics for EPCs in the best way possible. To this end, we use standard techniques from fixed point theory. Actually, there are several choices when defining non-local semantics for EPCs. These choices, however, do not compromise the application of the underlying fixed point theory. The mathematics used in this paper, can be considered as a semantical framework for formally defining non-local semantics for EPCs. This framework can be used for the discussion and, eventually, for settling the discussion on the semantics of EPCs.

1 Introduction

Ever since the definition of *Event driven Process Chains* (*EPCs*) in the early 90ties [KNS92], there has been a debate on their precise semantics. One feature recurrently provoking a debate is the *non-locality* of the semantics of the OR-join and the XOR-join connectors. On the one hand, this non-local semantics helps to simplify many models. On the other hand, there is no satisfactory formalization of this semantics yet. Many formalizations simply ignore the non-local semantics of the OR-join and XOR-join connectors; others provide ad-hoc solutions. Rittgen [Rit00] discusses some aspects of this problem and some approaches towards defining more satisfactory semantics for EPCs, which help resolving the problems. One concept proposed by Langner, Schneider and Wehler [LSW98], for example, is some additional synchronization, which is similar to *dead path elimination* in IBM's MQ Series process model [LA94]. Rittgen himself introduces some new syntax for EPCs in order to (partially) cure the problem [Rit00].

One reason for the ongoing debate on the semantics of EPCs is inherent to the non-locality of the informal semantics for EPCs: In essence, a non-local semantics refers to itself in its

*The full version of this paper, which includes all proofs can be obtained from http://www.upb.de/cs/kindler/publ.html

own definition (see Sect. 2 for more details). Even worse, this self-reference occurs under a negation, which easily results in a mathematical, conceptual, and technical short-circuit. In [ADK02], we pinpoint these arguments and prove that there is no formal semantics that fully complies with the informal semantics of EPCs – the *vicious circle*.

From a theoretical point of view, the result of [ADK02] should have settled the debate – against non-local semantics for EPCs. In practice, however, there are many EPC models that benfit from the non-locality. Therefore, we set out to define a sound semantics for EPCs that comes as close as possible to the informal semantics[1], which will be presented in this paper. In fact, we provide a framework for easily defining non-local semantics for EPCs. Technically, this framework resolves the vicious circle by distinguishing two related transition relations for EPCs and by using fixed point theory for capturing self-references.

What is more, this framework comes with a characterization of *unclean* EPCs for each concrete definition of a semantics. These unclean EPCs are those that do not exactly capture the informal semantics and, therefore, are ambiguous. Maybe, this framework helps to, eventually, settle the debate about 'the semantics' of EPCs – this way, resolving the vicious circle of never-ending discussions.

2 The problem

In this section, we present the syntax and the non-local semantics of EPCs and discuss the problem with the informal semantics of EPCs. Here, we can give a rough outline only. For a more detailed motivation of EPCs, their syntax, and their semantics, we refer to [KNS92, NR02]. For a more detailed exposition of the problems with the non-local semantics of EPCs, we refer to [ADK02].

Figure 1 shows an example of an EPC. It consists of three kinds of nodes: *events* graphically represented as hexagons, *functions* represented as rounded boxes, and *connectors* graphically represented as circles. The dashed arcs between the different nodes represent the *control flow*. The two black circles do not belong to the EPC itself; they represent a *state* of an EPC. A state, basically, assigns a number of *process folders* to each arc of the EPC. Each black circle represents a process folder at the corresponding arc.

The semantics of an EPC defines how process folders are propagated through an EPC. Clearly, this depends on the involved node. For events and functions, a process folder is simply propagated from the incoming arc to the outgoing arc. The *transition relation* for events and functions is graphically represented in the top row of Fig. 2 (a. and b.). For connectors, the propagation of folders depends on the type of the connector (*AND*, *OR*, resp. *XOR*) and whether it is a *join* or a *split connector*. Figure 2 shows the transition relation for the connectors. For example, the AND-split connector (c.) propagates a folder from its incoming arc to all outgoing arcs. The AND-join connector (d.) needs one folder on each incoming arc, which are propagated to a single folder on the outgoing arc.

[1] Actually, there is not a single well-accepted informal semantics for EPCs. We refer to the informal semantics presented by Nüttgens and Rump [NR02].

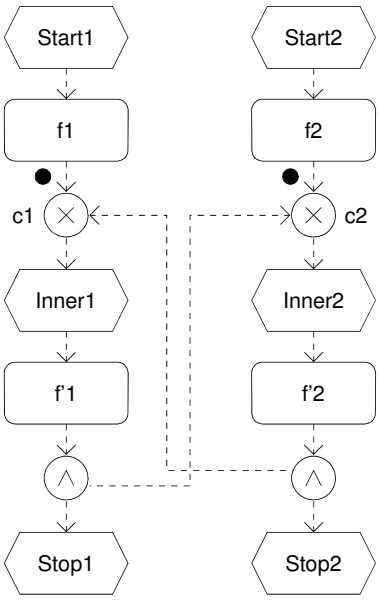

Figure 1: An EPC

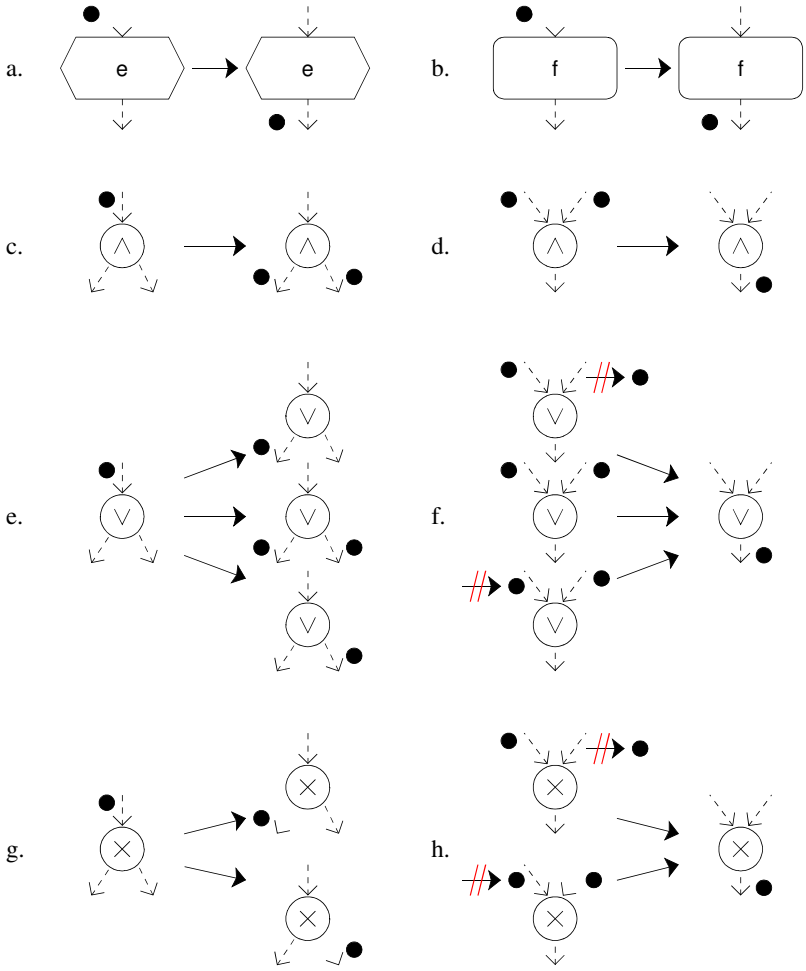

Figure 2: The transition relation for the different nodes

The more interesting connectors are the OR-join and the XOR-join. Here, we focus on the XOR-join. An XOR-join connector (h.) waits for a folder on one incoming arc, which is then propagated to the outgoing arc. But, there is one additional condition: The XOR-join must not propagate the folder, if there is or there could arrive a folder on the other incoming arc. In Fig. 2.h, this is represented by a label $\not\rightarrow\bullet$ at the other arc. Note that this condition cannot be checked locally for the XOR-join connector, since whether a folder could arrive at the other arc or not depends on the overall behaviour of the EPC. Therefore, we call the semantics of the XOR-join *non-local*. Likewise, the OR-join connector (f.) has a non-local semantics. Note that the additional condition refers to the transition relation itself and that the transition relation occurs under a negation in this condition.

Basically, the intended non-local behaviour has two problems, a technical one and a conceptual one (see [ADK02] for details):
1. In the definition of the transition relation, we refer to the the transition relation itself, which easily results in definitions that are not mathematically sound. In principle, this problem could be avoided by using some kind of fixed point semantics (the standard trick for giving semantics to objects that refer to themselves). The problem with the non-local

semantics of EPCs, however, is that a fixed point does not exist in all cases.
2. The conceptual problem is that, for some EPCs, there is no formal semantics that exactly captures the informal semantics. The reason is that the self-reference occurs under a negation. For example, consider the EPC from Fig. 1 with one process folder on one of the incoming arcs of each XOR-join c_1 and c_2. For symmetry reasons, either both of them should be able to propagate this folder or both should not be able to propagate the folder. But, either way contradicts the definition of the informal semantics of the XOR-join.

One purpose of this paper, is to define a non-local semantics for EPCs that is mathematically sound and comes as close as possible to the informal semantics of EPCs. What is more, this definition allows us identifying problematic EPCs, i. e. EPCs for which the informal semantics is ambiguous. The main purpose of this paper, however, is the presentation of a framework for defining and discussing different kinds of non-local semantics. The concrete semantics defined here, serves as an example for presenting the framework. The definition of 'the semantics' of EPCs is left to the EPC community.

3 The syntax of EPCs

In this section, we formalize the syntax of EPCs. Since the focus of this paper is on a formalization of the semantics of EPCs, we will omit some syntactical restrictions that are not relevant for our semantical considerations; moreover, we consider *flat EPCs* only, i. e. EPCs without subprocesses. Basically, an EPC is a graph, i. e. it consist of nodes and arcs. In order to express some of the syntactical restrictions, we first introduce a simple notation for the ingoing and outgoing arcs of a node:

Notation 1 (Ingoing and outgoing arcs) Let N be a set of *nodes* and let $A \subseteq N \times N$ be a binary relation over N, the *arcs*. For each node $n \in N$, we define the set of its *ingoing arcs* $n_{in} = \{(x, n) \mid (x, n) \in A\}$, and we define the set of its *outgoing arcs* $n_{out} = \{(n, y) \mid (n, y) \in A\}$.

An EPC consists of three different kinds of nodes, *events*, *functions*, and *connectors*, which are connected by *control flow arcs*. A connector can be either an AND-, an OR-, or an XOR-connector, which is indicated by labelling the connector correspondingly. Each function has exactly one ingoing and one outgoing arc, whereas each event has at most one ingoing and at most one outgoing arc. A connector has multiple ingoing arcs and one outgoing arc, or it has one ingoing arc and multiple outgoing arcs:

Definition 2 (EPC) An *EPC* $M = (E, F, C, l, A)$ consists of three pairwise disjoint sets E, F, and C, a mapping $l : C \to \{and, or, xor\}$ and a binary relation $A \subseteq (E \cup F \cup C) \times (E \cup F \cup C)$ such that $|e_{in}| \leq 1$ and $|e_{out}| \leq 1$ for each $e \in E$, $|f_{in}| = |f_{out}| = 1$ for each $f \in F$, and $|c_{in}| \geq 1$ and $|c_{out}| \geq 1$ and either $|c_{in}| = 1$ or $|c_{out}| = 1$ for each $c \in C$. An element of E is called an *event*, an element of F is called a *function*, an element of C is called a *connector*, and an element of A is called a *control flow* arc.

Note, that we have omitted some syntactical restrictions for EPCs. These requirements are important from a practical point of view, but they are not relevant when defining the semantics of EPCs. So, we do not formalize these restrictions here. For a complete expo-

sition of the syntax of EPCs, we refer to [NR02] .

In the definition of the semantics, we need to distinguish among different types of connectors: AND-, OR-, and XOR-, each of which can be either a *split* or a *join connector*.

Notation 3 (Nodes and connectors) For the rest of this paper, we fix an EPC $M = (E, F, C, l, A)$. We denote the set of all its nodes by $N = E \cup F \cup C$ and we define the following sets of connectors:
AND-split: $C_{as} = \{c \in C \mid l(c) = and \wedge |c_{in}| = 1\}$
AND-join: $C_{aj} = \{c \in C \mid l(c) = and \wedge |c_{out}| = 1\}$
OR-split: $C_{os} = \{c \in C \mid l(c) = or \wedge |c_{in}| = 1\}$
OR-join: $C_{oj} = \{c \in C \mid l(c) = or \wedge |c_{out}| = 1\}$
XOR-split: $C_{xs} = \{c \in C \mid l(c) = xor \wedge |c_{in}| = 1\}$
XOR-join: $C_{xj} = \{c \in C \mid l(c) = xor \wedge |c_{out}| = 1\}$

At last, we define the *states* of an EPC:

Definition 4 (State of an EPC) For an EPC $M = (E, F, C, l, A)$, we call a mapping $\sigma : A \to \{0, 1\}$ a *state* of M. The set of all states of M is denoted by Σ.

For simplicity, at most one folder at each arc is allowed in a state. But, we will see in Sect. 7 that this restriction can be easily released.

Note that, we assign the process folders to the control flow arcs of the EPC, whereas most other formalizations (e. g. [Rum99, Rit00, NR02]) assign the process folders to the nodes of the EPC. Though this choice is not essential for the definition of the semantics, it allows us a smoother technical presentation of the semantics[2]. In addition, this choice makes the nodes of the EPCs the active parts, whereas the arcs become the passive parts. We feel that this is a more natural conception of EPCs – but this is a very personal view.

4 The transition relation $R(P)$

The semantics of an EPCs will be defined in terms of a *transition relation* between its states. In order to identify the node involved in a transition, the transition relation is a relation $R \subseteq \Sigma \times N \times \Sigma$. A single transition $(\sigma, n, \sigma') \in R$, represents a change from state σ to σ' that corresponds to node n. In order to define and to argue on transition relations, we introduce some notations for restricting it and for its induced reachability relation:

Notation 5 (Restriction and reachability) For some transition relation $R \subseteq \Sigma \times N \times \Sigma$, and some subset $N' \subseteq N$, we define the *restriction of* R to N' as $R|_N = \{(\sigma, n, \sigma') \in R \mid n \in N'\}$. By slight abuse of notation, we define the *reachability relation* R^* of R as the reflexive and transitive closure of the binary relation $\to = \{(\sigma, \sigma') \in \Sigma \times \Sigma \mid \exists n \in N : (\sigma, n, \sigma') \in R\}$.

As discussed in Sect. 2, we need to refer to the transition relation in its own definition

[2]For example, there is no need to extend an EPC with *pre-connectors* as introduced in [NR02].

when defining a non-local semantics for the OR-join and the XOR-join connectors. Such cyclic references, however, are not possible in a sound mathematical definition. In order to resolve this cycle, we assume that some transition relation $P \subseteq \Sigma \times N \times \Sigma$ is given, and we define another transition relation $R \subseteq \Sigma \times N \times \Sigma$ where we refer to P whenever we would like to refer to R itself. We write $R(P)$ in order to make this dependency of R from the transition relation P explicit[3].

For now, we assume that P is some given transition relation. So, there is no cycle in the definition of $R(P)$. Later, in Sect. 5, we will use a standard trick of semantics, fixed points, for establishing a reference of R to itself. But, this need not bother us right now.

In the following definition, we first define one transition relation R_n for each node $n \in N$ of the EPC separately (cf. Fig. 2). Then, the overall transition relation $R(P)$ is the union of the transition relations R_n of all nodes n.

Definition 6 (Transition relation $R(P)$) Let P be a transition relation for an EPC M. For each node $n \in N$, we define the transition relation $R_n \subseteq \Sigma \times N \times \Sigma$ as follows:

a. For $n = e \in E$ with $e_{in} = \{i\}$ and $e_{out} = \{o\}$, we define $R_e \subseteq \Sigma \times N \times \Sigma$ by $(\sigma, e, \sigma') \in R_e$ iff $\sigma(i) = 1$, $\sigma(o) = 0$, $\sigma'(i) = 0$, $\sigma'(o) = 1$, and $\sigma'(a) = \sigma(a)$ for each $a \in A \setminus \{i, o\}$.

a' For $n = e \in E$ with $e_{in} = \emptyset$ or $e_{out} = \emptyset$, we define $R_e = \emptyset$.

b. For $n = f \in F$ with $f_{in} = \{i\}$ and $f_{out} = \{o\}$, we define $R_f \subseteq \Sigma \times N \times \Sigma$ by $(\sigma, f, \sigma') \in R_f$ iff $\sigma(i) = 1$, $\sigma(o) = 0$, $\sigma'(i) = 0$, $\sigma'(o) = 1$, and $\sigma'(a) = \sigma(a)$ for each $a \in A \setminus \{i, o\}$.

c. For $n = c \in C_{as}$ with $c_{in} = \{i\}$, we define $R_c \subseteq \Sigma \times N \times \Sigma$ by $(\sigma, c, \sigma') \in R_c$ iff $\sigma(i) = 1$, $\sigma(o) = 0$ for each $o \in c_{out}$, $\sigma'(i) = 0$, $\sigma'(o) = 1$ for each $o \in c_{out}$, and $\sigma'(a) = \sigma(a)$ for each $a \in A \setminus (\{i\} \cup c_{out})$.

d. For $n = c \in C_{aj}$ with $c_{out} = \{o\}$, we define $R_c \subseteq \Sigma \times N \times \Sigma$ by $(\sigma, c, \sigma') \in R_c$ iff $\sigma(i) = 1$ for each $i \in c_{in}$, $\sigma(o) = 0$, $\sigma'(i) = 0$ for each $i \in c_{in}$, $\sigma'(o) = 1$, and $\sigma'(a) = \sigma(a)$ for each $a \in A \setminus (c_{in} \cup \{o\})$.

e. For $n = c \in C_{os}$ with $c_{in} = \{i\}$, we define $R_c \subseteq \Sigma \times N \times \Sigma$ by $(\sigma, c, \sigma') \in R_c$ iff, for some $S \subseteq c_{out}$ with $|S| \geq 1$, we have $\sigma(i) = 1$, $\sigma(o) = 0$ for each $o \in S$, $\sigma'(i) = 0$, $\sigma'(o) = 1$ for each $o \in S$, and $\sigma'(a) = \sigma(a)$ for each $a \in A \setminus (\{i\} \cup S)$.

f. For $n = c \in C_{oj}$ with $c_{out} = \{o\}$, we define $R_c \subseteq \Sigma \times N \times \Sigma$ by $(\sigma, c, \sigma') \in R_c$ iff, for some $S \subseteq c_{in}$ with $|S| \geq 1$, we have $\sigma(i) = 1$ for each $i \in S$, $\widehat{\sigma}(a) = 0$ for each $\widehat{\sigma}$ with $\sigma(P|_{N \setminus \{c\}})^* \widehat{\sigma}$ and for each $a \in c_{in} \setminus S$, $\sigma(o) = 0$, $\sigma'(i) = 0$ for each $i \in S$, $\sigma'(o) = 1$, and $\sigma'(a) = \sigma(a)$ for each $a \in A \setminus (S \cup \{o\})$.

g. For $n = c \in C_{xs}$ with $c_{in} = \{i\}$, we define $R_c \subseteq \Sigma \times N \times \Sigma$ by $(\sigma, c, \sigma') \in R_c$ iff, for some $o \in c_{out}$, we have $\sigma(i) = 1$, $\sigma(o) = 0$, $\sigma'(i) = 0$, $\sigma'(o) = 1$, and $\sigma'(a) = \sigma(a)$ for each $a \in A \setminus \{i, o\}$.

[3] This reflects the fact, that actually R is a function.

h. For $n = c \in C_{xj}$ with $c_{out} = \{o\}$, we define $R_c \subseteq \Sigma \times N \times \Sigma$ by $(\sigma, c, \sigma') \in R_c$ iff, for some $i \in c_{in}$, we have $\sigma(i) = 1$, $\widehat{\sigma}(a) = 0$ for each $\widehat{\sigma}$ with $\sigma(P|_{N\setminus\{c\}})^*\widehat{\sigma}$ and for each $a \in c_{in} \setminus \{i\}$, $\sigma(o) = 0$, $\sigma'(i) = 0$, $\sigma'(o) = 1$, and $\sigma'(a) = \sigma(a)$ for each $a \in A \setminus \{i, o\}$.

We define the *transition relation* $R(P) = \bigcup_{n \in N} R_n$

Below, we briefly discuss the important cases f. and g. of the above definition, which concerns the non-local semantics of the OR-join and the XOR-join connectors:
When there is a folder on at least one of its incoming arcs $S \subseteq c_{in}$ of an OR-join connector and no folder can arrive (according to P) on the other arcs without the occurrence of c, the folder is propagated to the outgoing arc. In order to formalize that no folder can arrive on the other incoming arcs $a \in c_{in} \setminus S$, the definition refers to the states $\widehat{\sigma}$ that can be reached from σ (with respect to P) without the occurrence of c. This is formalized by $\sigma\ (P|_{N\setminus\{c\}})^*\ \widehat{\sigma}$. The XOR-join is similar to the definition of the OR-join. Instead of selecting some set S of incoming arcs on which a folder must be present, we select exactly one incoming arc i. We require that no folder can arrive on the other incoming arcs (with respect to P) before the occurrence of c. Again, this can be formalized by using the relation $(P|_{N\setminus\{c\}})^*$.

Note that there are several options on how the transition relation for each node type could be defined. One option concerns the question whether a folder on an outgoing arc should block the propagation of a folder from the ingoing arc. Another option would be to define a local semantics for the XOR-join[4]. Actually, the purpose of this paper is not to discuss these question, but to provide a framework for formally defining a semantics for EPCs. The exact definition of the semantics for the individual connectors is left to the a standardization effort within the EPC community. Actually, the precise definition of $R(P)$ is not crucial for making the rest of our theory work (see Sect. 7 for details). There is only one crucial condition: $R(P)$ must be a monotonously decreasing function.

Lemma 7 ($R(P)$ is monotonously decreasing) The operation $R(P)$ is monotonously decreasing; i. e. for two transition relations $P \subseteq P'$ we have $R(P) \supseteq R(P')$.

The proof of this lemma exploits the fact that additional transitions in P mean more states $\widehat{\sigma}$ reachable from σ (f. and h.), which result in a stronger restriction of R_c for the OR-join and the XOR-join connectors.

5 The semantics of EPCs

As mentioned above, the semantics of an EPC should be some transition relation on the states of the EPC. In the previous section, we have not defined a transition relation, but we have defined a transition relation $R(P)$, which depends on some given transition relation P. In this section, we will use $R(P)$ for defining the semantics of EPCs. On a first glance,

[4]There is a debate whether the XOR-join should have a non-local semantic or not. For example, Rittgen [Rit00] proposes a local semantics for the XOR-join connector. Here, we follow Nüttgens and Rump [NR02] in giving it a non-local semantics.

there are two different ways to define this semantics:
1. We could use some transition relation P and then calculate $R(P)$ as the semantics of the EPC. Actually, this idea is used in the semantics of YAWL [AaHo02]. The problem, however, is that we need to define P first. And it is by no means clear how P should be defined, and which definition is the best. YAWL, for example, uses a simple transition relation that ignores all OR-join connectors. Similar ideas came up in private discussions with Nüttgens and Rump during the discussions on [NR02, ADK02]. But each choice appears to be ad hoc in some way.
2. A better solution would be to find some P such that we have $P = R(P)$ – i. e. P is some fixed point of R. In that case, P refers to itself in its own definition $R(P)$ – the fixed point trick. Therefore, a fixed point P of $R(P)$ would exactly meet our initial intension, which justifies to call such a P an *ideal semantics* of the EPC. The problem with this definition, however, is that for some EPCs such a P does not exist; for others there are several different ideal semantics, and it is impossible to characterize one (e. g. the least fixed point with respect to set inclusion) among these ideal semantics to be 'the semantics' of the EPC (see Sect. 6 for a more detailed discussion).

Since both of the above approaches are unsatisfactory, we try a combination of both: We are now looking for a pair of transition relations (P, Q), such that we have $Q = R(P)$ and $P = R(Q)$, i. e. one transition relation is the input for the definition of the other. We will see that such pairs exist for each EPC, and that we can characterize some particular pair that will be used as 'the semantics' of the EPC.

In order to prove the existence of such pairs by standard fixed point theory, we define the domain D of all pairs of transition relations and an order relation $\preceq$, which forms a complete lattice on this domain. Moreover, we define a function φ such that the fixed points of φ are exactly the pairs meeting the above requirement.

Definition 8 For an EPC with nodes N and states Σ, we define the *domain* $D = 2^{\Sigma \times N \times \Sigma} \times 2^{\Sigma \times N \times \Sigma}$, and we define the relation $\preceq$ on D as follows: For two elements $d = (P, Q)$ and $d' = (P', Q')$, we define $d \preceq d'$ if $P \subseteq P'$ and $Q \supseteq Q'$. On D, we define the function $\varphi : D \to D$ by $\varphi((P, Q)) = (R(Q), R(P))$.

Note that $(D, \preceq)$ is a complete lattice on D, because $\preceq$ inherits this structure from $\subseteq$ and $\supseteq$. Moreover, the function φ is monotonic, which follows immediately from the fact that $R(P)$ is monotonously decreasing:

Lemma 9 The function φ on D is monotonic, i. e. for $d \preceq d'$, we have $\varphi(d) \preceq \varphi(d')$.

A fixed point of φ is an element $d \in D$ such that $\varphi(d) = d$. Note that $d = (P, Q)$ is a fixed point of φ if and only if $P = R(Q)$ and $Q = R(P)$, which are exactly those pairs of transition relations we are heading for. What is more, we can show that φ has fixed points by applying the renown Knaster-Tarski-Theorem.

Proposition 10

1. The function φ has fixed points; in particular, it has a least fixed point and it has a greatest fixed point (with respect to $\preceq$).
2. If (P, Q) is a fixed point of φ then also (Q, P) is a fixed point of φ.

3. The pair (P, Q) is the least point of φ, iff (Q, P) is the greatest fixed point of φ.

4. There is a unique fixed point of φ, iff the least fixed point has the form (P, P).

Proposition 10 says, that φ has two distinguished fixed points, the least and the greatest fixed point. Fortunately, if we know the least fixed point (P, Q), we know the greatest fixed point too: the reversed pair (Q, P). In particular, we have $P \subseteq Q$. P is the transition relation with the least transitions in it, and Q is the transition with the most transitions in its. So we can use the least fixed point for defining the semantics of the EPC.

Definition 11 (Semantics of an EPC) Let M be an EPC and let (P, Q) be the least fixed point of φ (wrt. $\preceq$). Then, we call P the *pessimistic transition relation* of the EPC M, and we call Q the *optimistic transition relation* of the EPC M.

Actually, we have defined two semantics. The *pessimistic semantics* P is the one that stops rather than doing something 'awkward'; the *optimistic semantics* Q does something 'awkward' rather than stopping indeliberately. Both semantics are related such that $P = R(Q)$ and $Q = R(P)$ – so one is the input transition relation for the definition of the other.

When the pessimistic and the optimistic semantics coincide, we know that we have an ideal semantics $P = Q = R(P) = R(Q)$. These are the EPCs with a *clean* semantics – without any ambiguity. Unfortunately, there are EPCs for which the pessimistic and the optimistic semantics do not coincide. We call these the EPCs with an *unclean* semantics.

6 The ideal semantics

Now, we have defined two semantics for an EPC, where the pessimistic semantics is the one that was intended in [NR02]. As mentioned in the introduction of Sect. 5, we would like to have an ideal semantics for an EPC, i. e. a transition relation P with $P = R(P)$. Here, we will show that an ideal semantics does not exist for all EPCs. Figure 3 shows an example. We argue indirectly that this EPCs has no ideal semantics: We assume that there is a transition relation P with $P = R(P)$ for this EPC. Now, we consider the state where there is a folder on each outgoing arc of the functions f_1, f_2, and f_3 as shown in Fig. 3. First, let us assume that according to P, connector c_i does not propagate the folder on its incoming arc. Then, according to the definition of $R(P)$, the subsequent connector $c_{(i+1) \bmod 3}$ will propagate the folder in $R(P)$. By $P = R(P)$, we know that, according to P, connector $c_{(i+1) \bmod 3}$ will propagate the folder. Second, let us assume that, according to P, c_i propagates the folder. By the same arguments, we can show that connector $c_{(i+1) \bmod 3}$ will not propagate the folder according to P. Since we have an odd number of XOR-join connectors on the cycle, we can argue that if c_1 propagates the folder according to P, then it does not propagate it and vice versa – a contradiction. So, our assumption that there is an ideal semantics $P = R(P)$ must have been wrong.

For some other examples, there are severals ideal semantics, which are symmetric such that one cannot be preferred to the other. For the example in Fig. 1, we have two completely

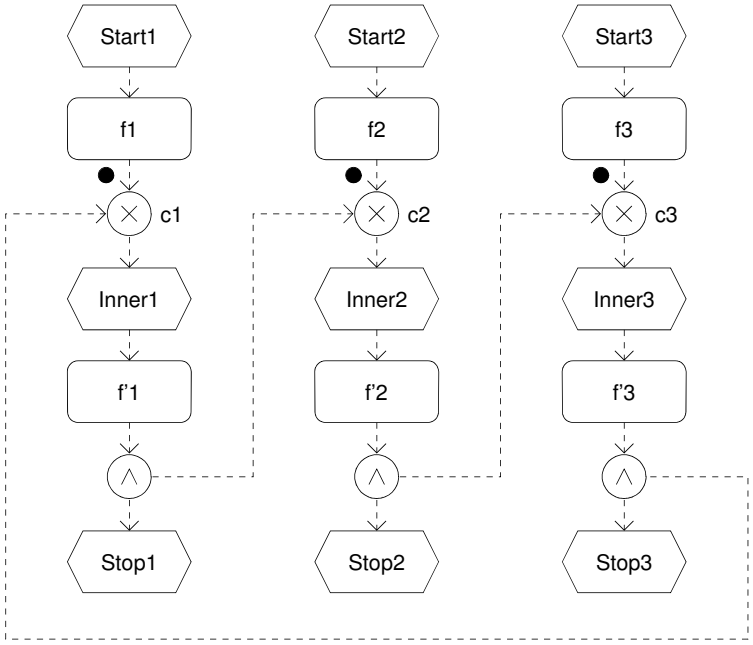

Figure 3: An EPC with no ideal semantics

symmetric ideal semantics[5]. In the first semantics, connector c_1 propagates the folder and connector c_2 does not propagate it. In the second semantics, connector c_2 propagates the folder and c_1 does not propagate it. Since both semantics are completely symmetric, one is as good as the other, there is no argument in favour of one of them.

These examples show that, in order to provide a semantics for all EPCs, we need to consider pairs of transition relations. Our definition gives a semantics to all (syntactically correct) EPCs – what is more, if the pessimistic and the optimistic semantics coincide, we have an ideal semantics. These are the EPCs for which the formal semantics precisely captures the informal semantics, the EPCs with a clean semantics. On the other hand, EPCs for which the the pessimistic and the optimistic semantics do not coincide, are problematic, because their formal semantics does not precisely capture the informal semantics. One benefit of our characterization is that we now have a clear definition of unclean EPCs.

7 The framework

In the previous sections, we presented a semantics for EPCs. Actually, it is not our intension to propose this semantics as 'the semantics' of EPCs. There are still some aspects of this semantics that need to be discussed. The main purpose of this paper is to define a framework for formalizing the semantics of EPCs, which allows us to discuss and to compare different semantics.

[5]The argument is the same as in the previous example. But, in this example we have an ideal semantics because the cycle consists of an even number.

For defining a semantics for EPCs, it is enough to define the function $R(P)$. The semantics of each individual node n of an EPC could be changed by changing the definition of relation R_n in Def. 6. If the resulting function $R(P)$ is monotonously decreasing, the rest of the theory will define a pessimistic and an optimistic semantics in the very same way. Therefore, we can concentrate on the definition of $R(P)$ or even on R_n when discussing semantical issues. The soundness of this framework is captured in the following theorem:

Theorem 12 (Semantical Framework) Let $M = (E, F, C, l, A)$ be an EPC.

1. Let $\mathcal{A}$ be some set. Then, we call $\sigma : A \to \mathcal{A}$ a *state* of M. The set Σ denotes the set of all states. A subset $P \subseteq \Sigma \times N \times \Sigma$ is called a *transition relation* of M with respect to $\mathcal{A}$.

2. Let $R : 2^{\Sigma \times N \times \Sigma} \to 2^{\Sigma \times N \times \Sigma}$ be a monotonously decreasing (with respect to $\subseteq$) function. We define the *domain* $D = 2^{\Sigma \times N \times \Sigma} \times 2^{\Sigma \times N \times \Sigma}$ and $\preceq$ on D as follows: For two elements $d = (P, Q)$ and $d' = (P', Q')$, we define $d \preceq d'$ if $P \subseteq P'$ and $Q \supseteq Q'$. Moreover, we define $\varphi : D \to D$ by $\varphi((P, Q)) = (R(Q), R(P))$.

Then φ has a least fixed point (P, Q) and a greatest fixed point (P, Q) (wrt. $\preceq$).

In essence, for defining a semantics for EPCs, we first define some set $\mathcal{A}$, which represents the folders that are assigned to an arc of the EPC. In our example, $\mathcal{A}$ was the set $\{0, 1\}$; another reasonable choice would be the natural numbers[6], when we would like to have more than one folder on each arc. Second, we need to define a transition relation $R(P)$ (on the states derived from the set $\mathcal{A}$), which in fact defines a function from a transition relation to a transition relation. The only requirement is that this function is monotonously decreasing. Then, $R(P)$ defines the function φ, which according to the above theorem has a least fixed point. The least fixed point (P, Q) of this function defines the the pessimistic semantics of the EPCs P and the optimistic semantics Q.

8 Conclusion

In this paper, we have proposed a semantics for EPCs, which is mathematically sound. We have argued that this semantics is as close to the informal semantics of [NR02] as can be. Nevertheless, we do not claim that this should be 'the semantics' of EPCs[7].

The main contribution of this paper is a sound mathematical theory for defining all kinds of non-local semantics for EPCs. A new semantics can be defined by giving a definition of $R(P)$; when $R(P)$ is monotonously decreasing, the semantics comes for free. For discussing different versions of semantics, we can concentrate on this definition. What is more, for any semantics defined in this framework, the framework clearly identifies clean

[6] We did not chose natural numbers in our paper, because this introduces many choices in the definition of $R(P)$, which need careful investigation and is beyond the scope of this paper.

[7] The author is not yet convinced that EPCs should have a non-local semantics because of the ambiguities arising in some EPCs. In order to provide a sound foundation for a discussion of different semantics, however, he saw the need to provide a mathematical theory for defining non-local semantics in a uniform way.

and unclean EPCs, where unclean EPCs are those that do not precisely meet the intended informal semantics. We hope that this framework helps to discuss different semantics and, ultimately, define 'the semantics' of EPCs.

Once 'the semantics' has been defined, the framework can be used for proving necessary and sufficient syntactical conditions characterizing unclean EPCs and to develop efficient algorithms for calculating the pessimistic and optimistic semantic for EPCs – whichever seems to be the more appropriate one. Of course, there are some obviously sufficient syntactical conditions for clean EPCs. For example, if no cycle of control flow of an EPCs has an XOR-join and OR-join connectors on it, the EPC has a clean semantics. But, stronger conditions are more difficult to find and strongly depend on the exact definition of $R(P)$. Likewise, fixed point approximation immediately gives us an algorithm for calculating the semantics of an EPC. Efficient algorithms, however, strongly depend on the exact definition of $R(P)$. Therefore, we need to define the 'the semantics' of EPCs first. Then we can start working on efficient simulation algorithms and efficient syntactical conditions for EPCs with a clean semantics.

Acknowledgements I would like to thank Markus Nüttgens and Frank Rump for many dicussions on the semantics of EPCs, which inspired me to provide a sound definition of a non-local semantics for EPCs. In particular, the arguments with Markus Nüttgens encouraged me not to stop with an impossibility result. Moreover, I would like to thank Wil van der Aalst and Björn Axenath for comments on earlier versions of this paper, which helped to improve the presentation of the ideas.

References

[ADK02] Wil van der Aalst, Jörg Desel, and Ekkart Kindler. On the semantics of EPCs: A vicious circle. In M. Nüttgens and F. J. Rump, editors, *EPK 2002, Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten*, 71–79, November 2002.

[AaHo02] W.M.P. van der Aalst and A.H.M. ter Hofstede. YAWL: Yet Another Workflow Language. Technical Report QUT Technical report, FIT-TR-2002-06, Queensland University of Technology, Brisbane, 2002.

[KNS92] G. Keller, M. Nüttgens, and A.-W. Scheer. Semantische Prozessmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK). Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi), Heft 89, Universität des Saarlandes, January 1992.

[LA94] F. Leymann and W. Altenhuber. Managing business processes as an information resource. *IBM Systems Journal*, 33(2):326–348, 1994.

[LSW98] P. Langner, C. Schneider, and J. Wehler. Petri Net Based Certification of Event driven Process Chains. In J. Desel and M. Silva, editors, *Application and Theory of Petri Nets 1998*, *LNCS* 1420, 286–305. Springer, 1998.

[NR02] Markus Nüttgens and Frank J. Rump. Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK). In *PROMISE 2002, Prozessorientierte Methoden und Werkzeuge fürr die Entwicklung von Informationssystemen, GI Lecture Notes in Informatics* P-21, 64–77. Gesellschaft für Informatik, 2002.

[Rit00] Peter Rittgen. Quo vadis EPK in ARIS? *Wirtschaftsinformatik*, 42:27–35, 2000.

[Rum99] Frank J. Rump. *Geschäftsprozeßmanagement auf der Basis ereignisgesteuerter Prozeßketten*. Teubner-Reihe Wirtschaftsinformatik. B.G.Teubner, 1999.

EPC Syntax Validation with XML Schema Languages

Jan Mendling

Universität Trier
E-Mail: mendling@web.de

Markus Nüttgens

Universität des Saarlandes
E-Mail: markus@nuettgens.de

Abstract: This paper addresses syntactical validation of Event-Driven Process Chains (EPC), in particular the syntactical validation of consecutive connectors. Building on the distinction between implicit and explicit element and arc types a definition of EPC syntax properties is given. Different XML Schema languages are examined in how far they are able to express this definition. The contribution of this paper is twofold: it is shown how EPCs can be represented in an XML syntax, called EPC Markup Language (EPML), and it is demonstrated how Schematron can be used to implement the EPC syntax definition as an EPML Schema validator.

1. Introduction

Event Driven Process Chains (EPC) are a popular technique for model business processes on a conceptual level [KNS92]. In order to leverage the benefits of business process modeling, process documentations should be reused in workflow applications or exchanged between different business process modeling tools. This avoids time-consuming re-input of data, which is less error prone and which grants consistency between models in different applications. Even for a small number of tools involved, it becomes efficient to define an intermediary or exchange format.

The approach towards an EPC Markup Language (EPML) [MN02] follows the guiding principles of readability, extensibility, tool orientation, and syntactical correctness [MN03b]. The first three features mainly address the presence of certain markup elements. The fourth aspect deals with correctness rules concerning the logical sequence of different EPC elements.

There have been a lot of the contributions on EPCs focusing on semantics, especially on the semantics of OR connectors. The translation of EPC process models to Petri Nets plays an important role in this context. Examples of this research can be found in Chen/Scheer [CS94], Rodenhagen [Ro97], Langner/Schneider/Wehler [LSW98], van der Aalst [Aa99], Rittgen [Ri00], and Dehnert [De02]. A major point of discussion is the "non-locality" of join-connectors [ADK02]. This paper will present a syntax related work based on the formal syntax definition of EPCs in [NR02].

Type consistency of consecutive connectors poses a major problem for these definitions. Fig. 1 illustrated this problem: when there is a chain of consecutive connectors, the syntactical correctness of them depends on the types of transitive non-connector ancestors and descendants. The connectors in the example are both correct because their transitive ancestors are all Events and the transitive descendants are all Functions. Mendling/Nüttgens [MN03a] address this problem by introducing implicit types for elements and arcs. Such a concept allows validation of type consistency of connector chains without taking transitive ancestors and descendants into consideration. Section two will present a respective definition of flat EPCs including EPC syntax rules that allow one to determine type consistency by checking only direct ancestors and descendants. Properties of hierarchical EPCs can be found in [NR02]. Section three will examine in how far different XML Schema Languages are capable to express these flat EPC syntax rules. Finally, we will show how Schematron, a rule-based XML Schema language, can be used to perform EPC syntax checks for process models expressed in EPC Markup Language (EPML), an XML syntax for EPCs.

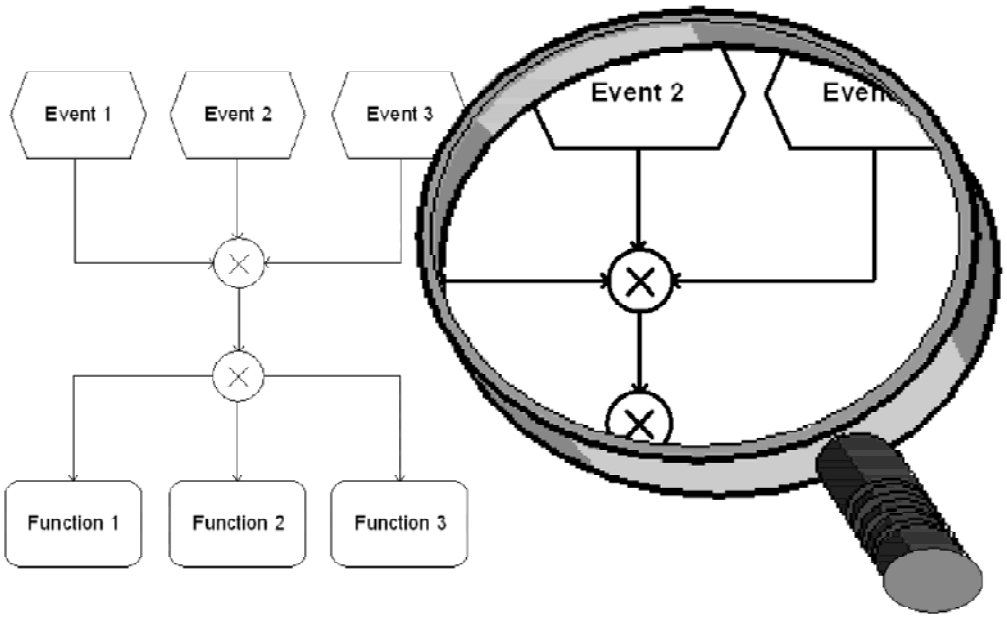

Fig. 1. Looking through the magnifier illustrates that syntactical correctness of the first connector cannot be determined by checking only direct ancestors and descendants.

2. EPC Syntax Properties and Implicit Types

2.1. Explicit and Implicit Element Types

The distinction of explicit and implicit element types is rooted in two perspectives on business process modelling, the perspective of the modeller and of verification. The task of the modeller is to compose a structure from a given set of symbols that is able to represent concepts of the domain in a pragmatic and abbreviated way [Si97]. This set of symbols is provided by a business process modelling tool. EPC symbols usually refer to the original definition of Keller/Nüttgens/Scheer [KNS92] distinguishing event type E, function type F, process interface P, connector AND, connector OR, and connector XOR. We refer to them as explicit element types.

Implicit element types relate to the perspective of syntactical verification. Each implicit element type captures a specific constellation in which explicit element types may occur implying different restrictions on the set of allowed ancestors and descendants, and their cardinality [MN03a]. Events E may be Start Events E_S, Inner Event E_{Int} or End Event E_E. Process Interfaces P can be used as start and end symbols. Connectors can be either joins or splits. When they have (transitive) event ancestors, their (transitive) descendants have to be functions. When they have (transitive) function ancestors, their (transitive) descendants have to be events. For OR- and XOR-connectors Event-Function-Splits are forbidden [KNS92].

2.2. Explicit and Implicit Arc Types

Analogously to the distinction between explicit and implicit element types, implicit arc types are defined as a partition of the control flow arc (explicit arc type) [Me03]. Implicit arc types are subsets of the product of implicit element types. Fig. 2 presents which arcs are allowed from and to implicit element types. The 16x16 matrix shows 100 implicit arc types which are permitted. The distribution of them in this matrix suggests the distinction between two different groups of implicit arcs. We will refer to them as Function-Event-Arcs FEA and Event-Function-Arcs EFA. In order to define these two kinds of arcs, a grouping of implicit element types into Event Types (from), Function Types (from), Event Types (to), and Function Types (to) is needed.

$$\text{Event Types (from) } ET_{from} := E_S \cup E_{Int} \cup AND_{EFS} \cup AND_{EFJ} \cup OR_{EFJ} \cup XOR_{EFJ} \quad (1)$$

$$\text{Function Types (from) } FT_{from} := F \cup P_S \cup AND_{FES} \cup AND_{FEJ} \cup OR_{FES} \cup OR_{FEJ} \cup XOR_{FES} \cup XOR_{FEJ}\,, \quad (2)$$

$$\text{Event Types (to) } ET_{to} := E_{Int} \cup E_E \cup AND_{EFS} \cup AND_{EFJ} \cup OR_{EFJ} \cup XOR_{EFJ}\,, \quad (3)$$

$$\text{Function Types (to) } FT_{to} := F \cup P_E \cup AND_{EFS} \cup AND_{EFJ} \cup OR_{EFJ} \cup XOR_{EFJ}\,. \quad (4)$$

TO (above) / (left) FROM	E_S	E_{Int}	E_E	F	P_S	P_E	AND_{EFS}	AND_{EFJ}	AND_{FES}	AND_{FEJ}	OR_{EFJ}	OR_{FES}	OR_{FEJ}	XOR_{EFJ}	XOR_{FES}	XOR_{FEJ}
E_S				→		→	→	→			→			→		
E_{Int}				→		→	→	→			→			→		
E_E																
F		→	→						→	→		→	→		→	→
P_S		→	→						→	→		→	→		→	→
P_E																
AND_{EFS}				→		→	→	→			→			→		
AND_{EFJ}				→		→	→	→			→			→		
AND_{FES}		→	→						→	→		→	→		→	→
AND_{FEJ}		→	→						→	→		→	→		→	→
OR_{EFJ}				→		→	→	→			→			→		
OR_{FES}		→	→						→	→		→	→		→	→
OR_{FEJ}		→	→						→	→		→	→		→	→
XOR_{EFJ}				→		→	→	→			→			→		
XOR_{FES}		→	→						→	→		→	→		→	→
XOR_{FEJ}		→	→						→	→		→	→		→	→

Fig. 2. Implicit arc types are a subset of the product of implicit element types. The arcs in the grey cells are Function-Event-Arcs; those arcs in the white cells are Event-Function-Arcs.

We can then define these two different implicit arc type groups as

$$FEA \subseteq (FT_{from} \times ET_{to}) , \quad (5)$$

$$EFA \subseteq (ET_{from} \times FT_{to}) , \quad (6)$$

In the following, the definition of implicit arc type groups will be used in a redefinition of EPCs. The advantages of such a definition are presented in section four.

2.3. Syntactical Constraints of Flat EPCs

Before presenting the syntactical properties of flat EPCs we still need some more definitions. We restrict our discussion to flat EPC properties as presented in [MN03a], which are based on the definitions in [NR02]. Let E_S, E_{Int}, E_E, F, P_S, P_E, AND_{EFS}, AND_{EFJ}, AND_{FES}, AND_{FEJ}, OR_{EFJ}, OR_{FES}, OR_{FEJ}, XOR_{EFJ}, XOR_{FES}, XOR_{FEJ} be sets of elements of the respective element types. Then a *set of vertices V* is

$$V := E_S \cup E_{Int} \cup E_E \cup F \cup P_S \cup P_E \cup AND_{EFS} \cup AND_{EFJ} \cup AND_{FES} \cup AND_{FEJ} \cup OR_{EFJ} \cup OR_{FES} \cup OR_{FEJ} \cup XOR_{EFJ} \cup XOR_{FES} \cup XOR_{FEJ} \quad (7)$$

with all the elements of the union being mutually disjoint. Referring to the definitions (5) and (6) a *set of arcs A* is defined as

$$A := FEA \cup EFA. \quad (8)$$

The *precondition set* of a vertex v is made up by the set of ancestor arcs written as

$$\rightarrow v := \{a \in A \mid a = (x,v) \wedge x,v \in V\} . \quad (9)$$

The *postcondition set* of a vertex v is defined as the set of descending arcs:

$$v\rightarrow := \{a \in A \mid a = (v,y) \wedge v,y \in V\} . \quad (10)$$

A *cycle set C* is a set of vertices building a cycle:

$$C \subseteq V = \{v_1,v_2,v_3,..,v_n\} \text{ with } v_1\rightarrow = \rightarrow v_2,\ v_2\rightarrow = \rightarrow v_3,\ \ldots\ ,\ v_n\rightarrow = \rightarrow v_1 \quad (11)$$

Then, a flat EPC Schema EPC_{flat} has the following *flat EPC* properties:

1. EPC_{flat} is a directed graph.
2. EPC_{flat} is a simple graph forbidding reflexive arcs or multiple arcs between two vertices.
3. EPC_{flat} is a coherent graph.
4. EPC_{flat} is an antisymmetric graph.
5. Cycles made up only of connectors are forbidden:
   $\forall C \subseteq V: C \cap (E_{Int} \cup F) \neq \emptyset$.
6. The set of Events $E = E_S \cup E_{Int} \cup E_E \neq \emptyset$.
7. The set of Functions $F \neq \emptyset$.

Concerning vertices there are the following *cardinality* constraints:

1. Start vertices: $\forall v \in E_S \cup P_S: \rightarrow v = \emptyset$ and $|v\rightarrow| = 1$.
2. End vertices: $\forall v \in E_S \cup P_S: |\rightarrow v| = 1$ and $v\rightarrow = \emptyset$.
3. Inner Events: $\forall v \in E_{Int}: |\rightarrow v| = 1$ and $|v\rightarrow| = 1$.
4. Functions: $\forall v \in F: |\rightarrow v| = 1$ and $|v\rightarrow| = 1$.
5. Splits: $\forall v \in AND_{EFS} \cup AND_{FES} \cup OR_{FES} \cup XOR_{FES}: |\rightarrow v| = 1$ and $|v\rightarrow| > 1$.
6. Joins: $\forall v \in AND_{EFJ} \cup AND_{FEJ} \cup OR_{EFJ} \cup OR_{FEJ} \cup XOR_{EFJ} \cup XOR_{FEJ}$: $|\rightarrow v| > 1$ and $|v\rightarrow| = 1$.

Concerning vertex types the following *type consistency* constraints apply:

1. Start Events: $\forall v \in E_S: v\rightarrow \subseteq EFA$.
2. Inner Events: $\forall v \in E_{Int}: \rightarrow v \subseteq FEA$ and $v\rightarrow \subseteq EFA$.
3. End Events: $\forall v \in E_E: \rightarrow v \subseteq FEA$.
4. Start ProcessInterface: $\forall v \in P_S: v\rightarrow \subseteq FEA$.
5. End ProcessInterface: $\forall v \in P_E: \rightarrow v \subseteq EFA$.
6. Function: $\forall v \in F: \rightarrow v \subseteq EFA$ and $v\rightarrow \subseteq FEA$.
7. Event-Function-Connects: $\forall v \in AND_{EFS} \cup AND_{EFJ} \cup OR_{EFJ} \cup XOR_{EFJ}$: $\rightarrow v \in EFA$ and $v\rightarrow \in EFA$.
8. Function-Event-Connects: $\forall v \in AND_{FES} \cup AND_{FEJ} \cup OR_{FES} \cup OR_{FEJ} \cup XOR_{FES} \cup XOR_{FEJ}$: $\rightarrow v \in FEA$ and $v\rightarrow \in FEA$.

In section four EPC syntax checks are discussed. We will refer to the EPC syntactical constraints as *Flat 1-7* for flat EPC Schema properties, *Card 1-6* for cardinality constraints, and *Type 1-8* for type consistency constraints. Here, we still need to mention constraints on arcs. Relating to our definition (1) - (4) of the different arc types, this seems redundant. But when it comes to model checking, implicit arc type groups will be a label on the respective arc which does not have to be consistent or correct. In that context there is a need to check if the types of the referenced vertices match the implicit arc type. Thus, we add *Arc 1-2* to check *arc consistency*:

1. $\forall (x,y) \in FEA: x \in FT_{from}$ and $y \in ET_{to}$.
2. $\forall (x,y) \in EFA: x \in ET_{from}$ and $y \in FT_{to}$.

3. EPC Markup Language (EPML) and XML Schema Languages

3.1. EPML by Example

Fig. 3 shows two EPC processes that both conform to the EPC syntax requirements. The Start Event is followed by a Function "List Requirement". Via a XOR connector two Events may occur. The "Design Process" is linked to "Can be fulfilled" by a ProcessInterface. The second process starts with the same ProcessInterface and the same Event "Can be fulfilled" follows. Afterwards we present EPML code for these two processes. It is simplified in that sense that it only describes the control flow, tool-orientation and extensibility issues are left out.

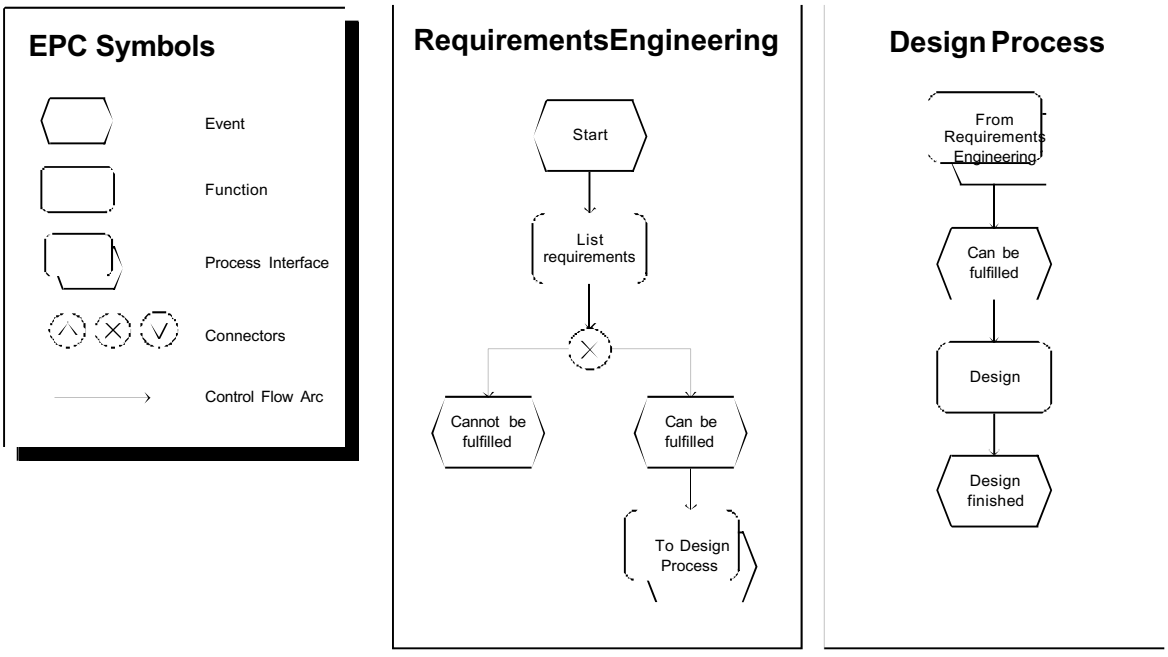

Fig. 3. Example of two flat EPC processes.

The code snippet in fig. 4 shows the corresponding representation in EPML according to [MN03b]. In the <Definitions> part objects are defined in order to allow their reusability via multiple references in one or more <EPC> processes. Connectors are excluded from this. The DefId attribute is set to a number unique in the context of <Definitions>, and later referenced as DefRef in process objects. The name attribute allows a description of the object. The <EPC> part is identified by a unique EpcId attribute, a name shall also be allowed. The explicit type is taken for markup, the implicit type is featured in the type attribute. <Arc> tags are identified by a unique combination of FromId and ToId attributes. They also have a type attribute indicating their implicit arc type.

```
<EPML>
    <Definitions>
       <EventDef DefId="1" name="Start"/>
       <FunctionDef DefId="2" name="List Requirements"/>
       <EventDef DefId="3" name="Cannot be fulfilled"/>
       <EventDef DefId="4" name="Can be fulfilled"/>
       <ProcessInterfaceDef DefId="5" name="Design Process"/>
       <FunctionDef DefId="6" name="Design"/>
       <EventDef DefId="7" name="Design finished"/>
    </Definitions>
    <EPC EpcId="1" name="Requirements Engineering">
       <Event DefRef="1" Id="11" type="EventStart"/>
       <Function DefRef="2" Id="12" type="Function">
       <Arc FromId="11" ToId="12" type="EventFunctionArc"/>
       <XOR Id="13" type="XORFunctionEventSplit"/>
       <Arc FromId="12" ToId=13" type="FunctionEventArc"/>
       <Event DefRef="3" Id="14" type="EventEnd"/>
       <Arc FromId="13" ToId="14" type="FunctionEventArc"/>
       <Event DefRef="4" Id="15" type="EventInternal"/>
       <Arc FromId="13" ToId="15" type="FunctionEventArc"/>
       <ProcessInterface DefRef="5" Id="16" type="ProcessIntEnd">
              <ToProcess EpcLinkId="2"/>
       </ProcessInterface>
       <Arc FromId="15" ToId="16" type="EventFunctionArc"/>
    </EPC>
    <EPC EpcId="2" name="Design Process">
       <ProcessInterface DefRef="5" Id="21" type="ProcessIntStart"/>
       <Event DefRef="4" Id="22" type="EventInternal"/>
       <Arc FromId="21" ToId="22" type="FunctionEventArc"/>
       <Function DefRef="6" Id="23" type="Function"/>
       <Arc FromId="22" ToId="23" type="EventFunctionArc"/>
       <Event DefRef="7" Id="24" type="EventEnd"/>
       <Arc FromId="23" ToId="24" type="FunctionEventArc"/>
    </EPC>
</EPML>
```

Fig. 4. Example of an EPML representation of the two flat EPC processes given in Fig. 3.

3.2. XML Schema Languages

In the following we will examine different XML schema languages. They are able to express the structure given by the example above, but they differ in their ability to express EPC syntax properties which are presented in section two. XML schema languages play an important role when it comes to exactly describing a class of XML documents. Document Type Definition (DTD) [Br00] has been the first schema language for XML, granting continuity from SGML where DTDs have also been used.

But with the spread of XML, new schema languages have been developed. Mainly, these efforts have been motivated by shortcomings of DTDs: lack of namespace support, non-XML syntax, and a lack of a mechanism to express constraints [Og00]. We will follow van der Vlist [Vl02] to distinguish three major groups: object-oriented XML schema languages, grammar-based XML schema languages, and rule-based XML schema languages. W3C XML Schema will be discussed as an object-oriented language, RELAX NG as a grammar-based approach, and Schematron as a rule-based schema language.

3.3. W3C XML Schema

W3C XML Schema [BM01], [Be01] is a schema language introduced by the World Wide Web Consortium to overcome shortcomings of DTDs. A major advantage is the much higher flexibility in defining uniqueness constraints. The old ID and IDREF data types of DTDs can still be used, but <xs:unique>, <xs:key>, and <xs:keyref> enable uniqueness constraints on arbitrary data types, in definable contexts, and for both attribute and element nodes. Nevertheless, W3C XML Schema has three grave disadvantages. Firstly, constraints can only be expressed in terms of restrictive content models or by using the above described uniqueness constructs. Take the four constraint packages defined in section 2. Flat 1 of EPCs being a directed graph can be handled by modelling arcs to have a FromId and a ToId. Flat 2 demanding for a simple graph can be controlled by forbidding multiple arcs between two vertices with a uniqueness constraint, but reflexive arcs cannot be prohibited by a uniqueness constraint. Concerning Flat 3-5 coherence, antisymmetry, and cycles also cannot be expressed with the help of uniqueness constraints.

For the objects within <EPC> a <xs:choice maxOccurs="unbounded"> is the appropriate definition to allow arbitrary sequences of any kind of object. To meet the existence-constraints of a minimum of one Event and one Function in an EPC of *Flat 6-7* there should be one `<Function>` and one `<Event>` defined before that choice. This is still okay with W3C XML Schema, but we have fixed the place of one mandatory Event and Function. An idea might be to add another leading choice block similar to the other. But this is forbidden according to the Unique Particle Attribution Rule, already a part of DTDs as nondeterministic content models. This is the second major disadvantage of W3C XML Schema. Consider the schema validator to find an `<Event>`. It would not know whether this element belongs to the choice definition or the element declaration. In formal languages this is not a problem, but in W3C XML Schema it is forbidden.

In general, the cardinality constraints *Card 1-6* could be expressed by a uniqueness constraint as a cardinality of one ancestor or one descendant respectively is demanded. For example, the `FromId` of a ProcessInterface-Event-Arc would have to be unique because there is only one arc from a ProcessInterface allowed. Unfortunately, this constraint would have to be declared in the `<arc>` container fixing it for all arc types. Taking an XML structure as presented above will not allow one to control cardinality via W3C XML Schema uniqueness checks.

The *Type 1-8* constraints pose a problem due to the third shortcoming of W3C XML Schema, the so called Consistent Declaration Rule. The `Ids` of objects could be modelled as belonging to a specific simple type, i.e. by defining a prefix using a regular expression ("Event[0-9]*"). This would cause problems with the type attribute in `<Arc>`, because the Consistent Declaration Rule forbids the choice between elements of the same name and different types. The `Ids` would have to be of different types which is not allowed. Finally, *Arc 1-4* pose the same problems as *Type 1-8*.

3.4. RELAX NG

Let's now take a look at RELAX NG [CM01]. This XML schema language is based on the formal concept of regular tree grammar. Its expressive power goes beyond W3C XML Schema [MLM00]. It does not have something like a Unique Particle Attribution Rule, and is therefore able to handle nondeterministic content models. But when it comes to constraints, RELAX NG depends on external data types. This means that only DTD classic ID, IDREF and IDREFS can be used. For a readable EPML structure as presented above, this is a major shortcoming. Take the cardinality constraints *Card 1-6*. We need to define the Id's of the object types as data type ID, FromId and ToId of the arc elements as IDREF type. This forbids the control of uniqueness on the FromId or the ToId attribute, because they need to be referenced and therefore have to have the IDREF data type, which cannot be additionally declared to be a unique ID as well. This shortcoming impedes the detection of multiple arcs between two vertices via a uniqueness check, because attributes in arcs need to be of IDREF type. Thus, concerning *Flat 1-5* RELAX NG shows this disadvantage in contrast to W3C XML Schema. But thanks to its ability to express nondeterministic content models, we can use the technique described in W3C XML Schema section to grant the existence of at least one Event and one Function element. If we consider the type constraints of ancestor and descendant vertices *Type 1-8* and also *Arc 1-2*, we face the same problems as already with W3C XML Schema. Even though RELAX NG is more powerful to describe structure than W3C XML Schema, it is less flexible to check uniqueness, which is a disadvantage in the context of our document structure. We will now have a look at Schematron belonging to the family of rule-based schema languages.

3.5. Schematron

Schematron [Je02] follows a different paradigm than W3C XML Schema and RELAX NG. Instead of declaring permitted structures Schematron checks for structures that are not permitted. For this purpose Schematron allows assertions to be declared which state positive properties of an instance document formulated as an XPath [CD99] expression. This allows a great flexibility due to XPath built-in functions like element count or string manipulation. A wide range of constraints can be expressed which is not possible using W3C XML Schema or RELAX NG.

As *Flat 1* is already granted by the structure of the arc elements, we continue with *Flat 2* demanding the EPC graph to be simple. Consider the following code which declares a rule in the context of <Arc> elements:

```
<sch:rule context="Arc">
    <sch:assert test="./@FromId!=./@ToId">An arc is not reflexive
    </sch:assert>
</sch:rule>
```

This check for reflexivity is easy to implement with Schematron, just like the check for multiple arcs and coherence. The constraint for cycles demand closure calculation, they cannot be formulated as XPath expressions. *Flat 6-7* concerning the existence of Events and Functions can easily be asserted with Schematron.

```
<sch:rule context="EPC">
    <sch:assert test="count(./Event)>1"/>
</sch:rule>
```

The cardinality constraints *Card 1-6* are also easy to specify with Schematron, for example take the Function element:

```
<sch:rule context="Function">
    <sch:assert test="count(./@Id=../Arc/@FromId)=1"/>
    <sch:assert test="count(./@Id=../Arc/@ToId)=1"/>
</sch:rule>
```

The type consistency constraints *Type 1-8* can also be expressed using XPath. E.g. *Type 7* imposes type restrictions for Event-Function-Connects:

```
<sch:rule context="AND|OR|XOR">
    <sch:assert test="
       (type='ANDEventFunctionSplit' or type='ANDEventFunctionJoin' or
        type='OREventFunctionJoin' or type='XOREventFuntionJoin')
        and @Id[@Id=@FromId[../@type='EventFunctionArc']]"/>
</sch:rule>
```

Similar to *Type 1-8* also *Arc 1-2* can be checked using Schematron.

4. Conclusion on the Usefulness of Implicit Element and Arc Types

Table 3 summarizes the results. Schematron provides the best mechanism to express syntactical constraints demanded by the formal definition of EPCs. W3C XML Schema shows some advantages over RELAX NG when it comes to uniqueness assertions. But both of them do not permit the expression of constraints as it is possible in Schematron. Only coherence and cycles (*Flat 3*, *Flat 5*) cannot be checked even with Schematron, because the EPC graph has to be traversed for these properties.

The good performance of Schematron is mainly a result of the EPC definition using implicit element and arc types. Implicit types avoid graph expansion and closure calculation for type consistency checks of connectors. Therefore, this EPC syntax definition has proved valuable in conjunction with the definition of an EPC Markup Language (EPML). It allows a straight forward EPC syntax validation (except *Flat 3* and *Flat 5*) given an EPML file as input, a Schematron definition of EPML, and a Schematron schema validator. This represents a major simplification of EPC syntax validation.

	W3C XML Schema	RELAX NG	Schematron
Flat 1: Directed	via structure	via structure	via structure
Flat 2: Simple	partly via uniqueness	no	via rule
Flat 3: Coherent	no	no	no
Flat 4: Antisymmetric	no	no	via rule
Flat 5: Cycles	no	no	no
Flat 6: ∃ Event	partly via structure	via structure	via rule
Flat 7: ∃ Function	partly via structure	via structure	via rule
Card 1: Start	no	no	via rule
Card 2: End	no	no	via rule
Card 3: Int. Event	no	no	via rule
Card 4: Functions	no	no	via rule
Card 5: Splits	no	no	via rule
Card 6: Joins	no	no	via rule
Type 1: Start Event	no	no	via rule
Type 2: Int. Event	no	no	via rule
Type 3: End Event	no	no	via rule
Type 4: Start Proc.-I.	no	no	via rule
Type 5: End Proc.-I.	no	no	via rule
Type 6: Function	no	no	via rule
Type 7: EF-connects	no	no	via rule
Type 8: FE-connects	no	no	via rule
Arc 1: FEA	no	no	via rule
Arc 2: EFA	no	no	via rule

Table 1. The summary of the findings: XML schema languages and their ability to express the constraints necessary to decide syntactical correctness of EPCs represented in EPML.

References

[Aa99] v.d. Aalst, W.M.P.: Formalization and Verification of Event-driven Process Chains, in: Information and Software Technology 41(1999)10, pp. 639-650.

[ADK02] v.d. Aalst, W.M.P.; Desel, J.; Kindler, E.: On the semantics of EPCs: A vicious circle, in: Nüttgens, M.; Rump, F.J. (eds.): Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten - EPK 2002, Proceedings of the GI-Workshop EPK 2002, Trier, 2002, pp. 71-79

[Be01] Beech, D.; Lawrence, S.; Moloney, M.; Mendelsohn, N.; Thompson, H.S. (eds.): XML Schema Part 1: Structures. World Wide Web Consortium, USA, 2001.

[BM01] Biron, P.V.; Malhotra, A. (eds.): XML Schema Part 2: Datatypes. World Wide Web Consortium, USA, 2001.

[Br00] Bray, T. et al. (eds.): Extensible Markup Language (XML) 1.0 (Second Edition). World Wide Web Consortium, USA, 2000.

[CD99] Clark, J.; DeRose, S.: XML Path Language (XPath) Version 1.0, World Wide Web Consortium, Boston, USA, 1999.

[CM01] Clark, J.; Murata, M.: RELAX NG Specification, 3 December 2001. URL: http://www.relaxng.org/spec-20011203.html.

[CS94] Chen, R.; Scheer, A.-W.: Modellierung von Prozessketten mittels Petri-Netz-Theorie, in: Scheer, A.-W. (ed.): Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 107, Saarbrücken 1994.

[De02] Dehnert, J.: Making EPC´s fit for Workflow Management, in: Nüttgens, M.; Rump, F.J. (eds.): Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten - EPK 2002, Proceedings of the GI-Workshop EPK 2002, Trier, 2002, pp. 51-69.

[Je02] Jelliffe, R.: The Schematron Assertion Language 1.5, 2002-10-01. URL: http://www.ascc.net/xml/resource/schematron/Schematron2000.html.

[KNS92] Keller, G.; Nüttgens, M.; Scheer, A.-W.: Semantische Prozeßmodellierung auf der Grundlage „Ereignisgesteuerter Prozeßketten (EPK)". In: Scheer, A.-W. (ed.): Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 89, Saarbrücken, 1992.

[LSW98] Langner, P.; Schneider, C.; Wehler, J.: Petri Net Based Certification of Event driven Process Chains, in: Desel, J.; Silva, M. (eds.): Application and Theory of Petri Nets 1998, LNCS Vol. 1420, Springer, Berlin et. al. 1998, pp. 286 305.

[Me03] Mendling, J.: Event-Driven-Process-Chain-Markup-Language (EPML): Anforderungen, Konzeption und Anwendung eines XML-Schemas für Ereignisgesteuerte Prozessketten (EPK), in: Höpfner, H.; Saake, G. (eds.): Proceedings of the Student Program of the 10th Conference "Database Systems for Business, Technology and Web", GI Section Databases and Information Systems, Leipzig, 25.02.2003, Madgeburg, 2003, pp. 48-50.

[MLM00] Murata, M.; Lee, D.; Mani, M.: Taxonomy of XML Schema Languages using Formal Language Theory. In: Extreme Markup Languages, Montreal 2001. URL: http://www.cobase.cs.ucla.edu/tech-docs/dongwon/mura0619.pdf.

[MN02] Mendling, J.; Nüttgens, N.: Event-Driven-Process-Chain-Markup-Language (EPML): Anforderungen zur Definition eines XML-Schemas für Ereignisgesteuerte Prozessketten (EPK), in: Nüttgens, M.; Rump, F.J. (eds.): Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten - EPK 2002, Proceedings of the GI-Workshop EPK 2002, Trier, 2002, pp. 87-93.

[MN03a] Mendling, J.; Nüttgens, M.: EPC Modelling Based on Implicit Arc Types. In: Godlevsky, M.; Liddle, S.W.; Mayr, H.C. (eds.): Information Systems Technology and its Applications, International Conference ISTA'2003, June 19-21, 2003, Kharkiv, Ukraine, Proceedings. LNI 30, pp. 131-142.

[MN03b] Mendling, J.; Nüttgens, M.: XML-basierte Geschäftsprozessmodellierung. In: Uhr, W., Esswein, W.; Schoop, E. (eds): Wirtschaftsinformatik 2003 / Band II, Heidelberg, 2003, pp. 161 -180.

[NR02] Nüttgens, M.; Rump, F.: Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK), in: Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen (Promise'2002), Hasso-Plattner-Institut für Softwaresystemtechnik an der Universität Potsdam, 9.-11. Oktober 2002, Potsdam 2002.

[Og00] Ogbuji, C.: Validating XML with Schematron, O'Reilly XML.com, Sebastopol et. al. 2000. URL: http://www.xml.com/pub/a/2000/11/22/schematron.html.

[Ri00] Rittgen, P.: Paving the Road to Business Process Automation, European Conference on Information Systems (ECIS) 2000, Vienna, Austria, July 3 - 5, 2000, pp. 313-319.

[Ro97] Rodenhagen, J.: Darstellung ereignisgesteuerter Prozeßketten (EPK) mit Hilfe von Petrinetzen, Diplomarbeit Universität Hamburg Fachbereich Informatik (Prof. Valk), Hamburg 1997.

[Sc00] Scheer, A.-W.: ARIS – Business Process Modeling, 3rd edition, Berlin et. al. 2000.

[St73] Sinz, E.: Modell. In: Mertens, P. et al. (eds.): Lexikon der Wirtschaftsinformatik, Berlin et al., 1997, S. 270.

[Vl02] van der Vlist, E.: XML Schema, O'Reilly, Sebastopol et. al. 2002.

Prozessmodellierung in eGovernment-Projekten mit der eEPK

Jörg Becker,
Lars Algermissen,
Björn Niehaves

Lehrstuhl für Wirtschaftsinformatik und Informationsmanagement,
Universität Münster
Leonardo-Campus 3
48149 Münster
{becker | islaal | bjni}@wi.uni-muenster.de

Abstract: Die prozessorientierte Analyse und Optimierung von Verwaltungsabläufen ist eine zentrale Voraussetzung für die erfolgreiche organisatorische und informationstechnische Umgestaltung der Kommunalverwaltung im Zuge des eGovernment. Die Komplexität der Prozessmodellierung ist wegen der Vielzahl von Modellierungszwecken, -gegenständen, -methoden und Modellnutzern sehr hoch und erfordert daher eine systematische Vorbereitung. Besondere Wichtigkeit kommt hier der Auswahl der zu verwendenden Modellierungssprache zu. In der Domäne eGovernment hat diese Methode diverse, vor allem domänenspezifische Anforderungen zu erfüllen. Die Untersuchung der Eigenschaften von Verwaltungsprozessen, die Bestimmung von Anforderungen an Modellierungsmethoden bzw. -sprachen im eGovernment und die vergleichende Untersuchung verschiedener Alternativen, darunter die eEPK, wird in den Rahmen eines mehrphasigen Vorgehensmodells zur Durchführung von prozessorientierten eGovernment-Projekten eingebettet. Dabei wird auf Erfahrungen der Prozessmodellierung mit der eEPK in der Stadtverwaltung Emsdetten und auf die aus den Anforderungen der Praxis erwachsenen Erweiterungen dieser Modellierungssprache eingegangen.

1 Prozesse im Fokus des eGovernment

Prozessorientierung ist seit Beginn der 90er Jahre als eine Maxime der Unternehmensgestaltung akzeptiert. In den letzten Jahren wurde im Zuge der Verbreitung von Electronic Government (eGovernment) auch in der öffentlichen Verwaltung begonnen, die Organisationsstrukturen an den Geschäftsprozessen auszurichten.

In diesem Kontext wird eGovernment verstanden als die Vereinfachung und Abwicklung von Informations-, Kommunikations- und Transaktionsprozessen zur Erbringung einer Verwaltungsdienstleistung durch den Einsatz von Informations- und Kommunikationstechnologien innerhalb und zwischen Behörden sowie zwischen Behörden und Privatpersonen, Unternehmen sowie politischen Entscheidungsträgern [BAN03a].

Die Komplexität der Prozessmodellierung ist wegen der Vielzahl von Modellierungszwecken, -gegenständen, -methoden und Modellnutzern sehr hoch und erfordert daher eine systematische Vorbereitung und ein methodisches Vorgehen bei der Durchführung von prozessorientierten eGovernment-Projekten. Von besonderer Wichtigkeit ist hier die Auswahl der Prozessmodellierungsmethode. Da jede Organisationsstruktur ihren eigenen Wirkungszusammenhang besitzt, müssen bei der Modellierung von Verwaltungsprozessen die Eigenschaften der Domäne detailliert untersucht und die sich daraus ergebenden Anforderungen im Rahmen der Methodenwahl besondere Berücksichtigung finden.

Zielsetzung des vorliegenden Artikels ist, die Erfahrungen der Prozessmodellierung im eGovernment mit der eEPK zu schildern und Vorschläge für Erweiterungen der Methode vorzustellen, die sich aus der Praxis der Domäne ergeben haben. Die Erfahrungen werden eingebettet in ein mehrphasiges Vorgehensmodell zur Durchführung von prozessorientierten Reorganisationsprojekten in der öffentlichen Verwaltung. In der Vorbereitungsphase werden im Anschluss an die Festlegung des Modellierungszwecks und des Modellierungsgegenstands (Kapitel 2) die Eigenschaften von Verwaltungsprozessen analysiert und die sich daraus ableitenden Anforderungen an eine Prozessmodellierungsmethode im eGovernment dargestellt. Die Vorbereitungsphase schließt mit der kriterienbasierten Selektion der Modellierungsmethode (Kapitel 3). Die zweite Phase umfasst die Entwicklung eines Ordnungsrahmens (Kapitel 4) und dient der Identifizierung und Priorisierung von Handlungsfeldern für die Prozessmodellierung. In der dritten Phase sind Ist-Abläufe zu modellieren, welche die Basis der sich anschließenden Phase der Soll-Modellierung bilden (Kapitel 5). Die aus den Praxis-Anforderungen der Domäne des eGovernment erwachsenden Erweiterungen der Prozessmodellierungsmethode eEPK werden in Kapitel 6 vorgestellt. Die Ausführungen werden mit praktischen Erfahrungen aus einem Prozessmodellierungsprojekt untermauert.

2 Festlegung des Modellierungszwecks und -gegenstands

In der Regel ist eine umfassende Vorbereitung der Prozessmodellierung erforderlich, da sich die Modellerstellung durch eine hohe prozessuale und die entstehenden Informationsmodelle durch eine hohe gegenständliche Komplexität auszeichnen. Dabei sind die ersten beiden Vorbereitungsschritte die Festlegung des Modellierungszwecks (‚wofür' soll modelliert werden, beispielsweise Zertifizierung, Softwareauswahl, Organisationsgestaltung) und die Festlegung des Modellierungsgegenstands (‚was' soll modelliert werden, beispielsweise Unternehmenstotalmodell vs. -partialmodell).

Festlegung des Modellierungszwecks

Die Hauptzwecke der Prozessmodellierung sind die Organisations- und Anwendungssystemgestaltung [RSD03, S. 58]. Modelle für die Organisationsgestaltung erfordern eine hohe Anschaulichkeit, wogegen Modelle für die Anwendungssystemgestaltung aufgrund ihrer Implementierungsnähe eine hohe technische Präzision erfordern. Der Hauptzweck bei der Stadtverwaltung Emsdetten war die Organisationsgestaltung, insbesondere die prozessorientierte Reorganisation von Verwaltungsprozessen.

Die Gestaltung der Anwendungssysteme wurde erst auf Basis von entsprechend verbesserten Prozessen als sinnvoll erachtet. Neben der Festlegung des Modellierungszwecks ist es grundsätzlich sinnvoll, im Rahmen der Modellerstellung die unterschiedlichen Anforderungen der Modellnutzer zu berücksichtigen (multiperspektivische Informationsmodellierung). Im Idealfall wird ein integriertes Prozessmodell erstellt, das abhängig vom Verwendungszweck und den Anforderungen der Modellnutzer unterschiedliche Perspektiven auf das Prozessmodell bietet [Be+02, S. 25 ff.].

Festlegung des Modellierungsgegenstands

Bei der Festlegung des Modellierungsgegenstands ist vor allem die Frage zu beantworten, ob ein Totalmodell oder ein Partialmodell eines Objektsystems erstellt werden soll. Da das Dienstleistungsportfolio einer mittelgroßen Kommune durchaus mehr als tausend einzelne Produkte umfassen kann, ist in diesem Kontext die klare Fokussierung auf ein Partialmodell in Form ausgewählter Kernprozesse zu fordern. Dabei hilft ein organisationsübergreifender Ordnungsrahmen, den Modellierungsgegenstand klar abzugrenzen (vgl. Kapitel 4).

3 Auswahl einer Modellierungsmethode

Der dritte und letzte Schritt im Rahmen der Vorbereitung der Prozessmodellierung besteht in der Auswahl einer geeigneten Modellierungsmethode. Es existieren diverse Methoden zur Modellierung von (Geschäfts-)Prozessen. Welche Modelltypen im Rahmen der Prozessmodellierung zum Einsatz kommen wird vor allem durch den Modellierungszweck und die Anforderungen der Modellnutzer beeinflusst (siehe Kapitel 2). Darüber hinaus sind die Spezifika der Prozesse innerhalb des adressierten Anwendungsbereichs detailliert zu untersuchen. Auf Basis dieser Größen sind Anforderungen an eine Modellierungsmethode zu definieren, die im Anschluss Bewertungskriterien zur Gegenüberstellung alternativer Modellierungsmethoden darstellen.

3.1 Eigenschaften von Verwaltungsprozessen

Eine Methode zur Modellierung von Verwaltungsprozessen muss deren Besonderheiten berücksichtigen und abbilden können. Zu den wichtigsten Eigenschaften von Verwaltungsprozessen gehören:

- **Repetitivität**. Abläufe in Verwaltungen weisen hohe Fallzahlen auf. Vor allem bürgerbezogene Dienstleistungen (bspw. Passverlängerung, Ahndung von Verkehrsordnungswidrigkeiten) und unternehmensbezogene Dienstleistungen (bspw. Ausstellung von Sondernutzungserlaubnissen) stellen zu einem Großteil massenhafte Routinearbeit dar.
- **Linearität**. Für Verwaltungsprozesse existieren verschiedene Ablaufmöglichkeiten, die durch zahlreiche Entscheidungssituationen bedingt sind. Betrachtet man die Entscheidungssituationen jedoch in Relation zur Länge der Prozesse, die aus der Fülle

verschiedener Bearbeitungsstufen und Kenntnisnahmeinstanzen (Mitzeichnungsverfahren) resultieren, so lassen sich Verwaltungsprozesse im Vergleich zu anderen betrieblichen Domänen als linear charakterisieren.

- **Strukturiertheit, Stabilität, Konsistenz**. Das Verwaltungshandeln ist entweder gesetzlich geregelt oder zumindest von gesetzlichen Regelungen abhängig. Diese Rechtsbindung gewährleistet einerseits eine hohe Verfahrenssicherheit/-konsistenz, Stabilität und Vergleichbarkeit, andererseits führt sie häufig zu Inflexibilität und stellt zumeist die entscheidende Barriere bei Veränderungsbestrebungen dar. Auch die Existenz von offenen Entscheidungsfeldern und Ermessensspielräumen weicht das Charakteristikum der Strukturiertheit von Verwaltungsprozessen nicht auf. Offenen Entscheidungsfeldern kann (im Rahmen sonst strukturierter Abläufe) durch entsprechend explizite Modellierung Rechnung getragen werden.

- **Bilateralität**. Eine Vielzahl von Verwaltungsprozessen kennzeichnet sich durch eine starke Einbindung des Kunden (Bürger/Unternehmen). Entsprechende Interaktionspunkte sind als Schnittstelle zwischen Kunde und Verwaltung zentral für die Qualitätswahrnehmung der Verwaltungsdienstleistung durch den Kunden [MB00, S. 42 ff.].

- **Dezentralität**. Viele Verwaltungsprozesse sind charakterisiert durch stark fragmentiertes, verteiltes Arbeiten [Le95], das aus der Beteiligung diverser Organisationseinheiten innerhalb und außerhalb der einzelnen Verwaltungen resultiert. Umfassende Prozesse werden in kleinen Ausschnitten durch viele verschiedene Mitarbeiter quasi „bürokratisch“ bearbeitet. Dabei besteht zwischen den einzelnen Teilschritten ein enormer Dokumentenfluss.

Die Besonderheiten der Verwaltungsprozesse leiten sich auf der einen Seite aus ihrem Dienstleistungscharakter und aus den zumeist immateriellen prozessprägenden Objekten ab. Auf der anderen Seite prägen rechtliche Rahmenbedingungen und die vor dem Hintergrund *Weber'scher* Bürokratieprinzipien [We22] historisch gewachsenen Arbeits- und Ablaufgrundsätze die Prozesse in Kommunalverwaltungen.

3.2 Anforderungen an Modellierungsmethoden im eGovernment

Aus den domänenspezifischen Besonderheiten und den Forderungen der Modelladressaten lassen sich Anforderungen an Modellierungsmethoden im eGovernment, vor allem im kommunalen Umfeld, ableiten. Dies sind zunächst Anforderungen, die auch außerhalb der Domäne eGovernment Gültigkeit entfalten. Hierzu gehören:

- **Einfachheit**. Ein einfaches Prinzip der Darstellung und die Übersichtlichkeit der dargestellten Sachverhalte mittels leicht zugänglicher Modellelemente erhöht die Verständlichkeit der beschriebenen Sachverhalte.

- **Hohe Abbildungsgenauigkeit**. Da die Prozessbeschreibung auf Fachkonzeptebene den ersten Schritt zur Überführung betrieblicher Sachverhalte in Informationstechnik darstellt, hat die Modellierungssprache entsprechende formale Mechanismen zur Gewährleistung einer hohen Abbildungsgenauigkeit aufzuweisen. Auf diese

Weise wird die Erweiterbarkeit im Sinne der Anwendungssystemgestaltung (z. B. Workflow-Management-Systeme) gewährleistet [Ga97].

- **Standardisierung des Modellinhalts**. Durch die Definition von Fachbegriffen wird die terminologische Konsistenz gewährleistet, während Ontologien die semantische Konsistenz sicherstellen. Diese Arten der Standardisierung des Modellinhalts müssen durch die Modellierungsmethode unterstützt werden, um so die Verständlichkeit der Inhalte zu erhöhen.

- **Ganzheitliche Perspektive**. Eine ganzheitliche Modellierung muss sowohl die Darstellung von Anwendungssystemen, von Organisationseinheiten und Stellen sowie des Dokumentenflusses vornehmen. Sie muss den geschilderten Eigenschaften der Verwaltungsprozesse gerecht werden.

- **Prozess-Controlling**. Durch die Abbildung qualitativer und quantitativer Informationen über Prozessausführungen mittels Kennzahlen ist die fortlaufende Erfolgsüberprüfung des Verwaltungshandelns gewährleistet [Sc^{+}96]. Im Rahmen von Prozessanpassungen ist so die Wirkung von Veränderungen messbar zu machen.

- **Werkzeugunterstützung**. Eine Anforderung, die sich vor allem aus der faktischen Komplexität von Geschäftsprozessoptimierungsprojekten ergibt, ist die Unterstützung der Modellierungsmethode durch ein Modellierungswerkzeug.

- **Unterstützung von Prozesssimulationen**. Häufig zentrales Ziel der prozessorientierten Reorganisation ist die Verbesserung der Kosteneffizienz. Dabei ermöglicht die Prozesssimulation eine Vorausschau auf die zu verändernde betriebliche Situation und kann auf Basis entsprechend attributierter Prozessmodelle (Eintrittswahrscheinlichkeiten, Kosten etc.) Aussagen über den Erfolg potenzieller Restrukturierungsmaßnahmen treffen.

Neben generellen Anforderungen ergeben sich zum Zwecke der adäquaten Abbildung der domänenspezifischen Problemstruktur spezielle Anforderungen:

- **Abbildung von Interaktionspunkten.** Die Explikation des bilateralen Prozesses zwischen dem Kunden (Bürger/Unternehmen) und der Verwaltung sowie die Darstellung von Interaktionspunkten (in puncto Häufigkeit, Dauer, Medium etc.) zwischen den Prozessbeteiligten dient der Analyse der Qualitätswahrnehmung der Verwaltungsdienstleistung durch den Kunden. Handlungsempfehlungen in Bezug auf die Interaktionsgestaltung − sowohl im Front Office als auch im Back Office − lassen sich auf diese Weise einfacher ableiten. Wegen der hohen Anzahl unterschiedlicher Verwaltungsprozesse und der jeweils starken Einbindung der Bürger kommt der Abbildung von Interaktionspunkten in der Domäne Verwaltung besonders hohe Bedeutung zu.

- **Explikation der fragmentierten, dezentralen Prozessbearbeitung.** Durch die Darstellung der Dezentralität der Arbeitsabläufe können im Rahmen eines Schnittstellenmanagements Prozessübergaben (in puncto Häufigkeit, Inhalt, Übergabeart etc.) gezielt analysiert und Optimierungspotenziale erschlossen werden.

- **Multiperspektivische Darstellung.** Es existiert eine Vielfalt von Modellnutzern (bspw. Bürgermeister, Bürger, Organisationsabteilung, IT-Abteilung, Sachbe-

arbeiter etc.), welche die Prozessmodelle zu unterschiedlichen Zwecken (bspw. Controlling, Organisations- oder Anwendungssystemgestaltung etc.) und Befugnissen (bspw. Bürgermeister oder Bürger) verwenden. Zur Reduzierung der Komplexität des dargestellten Sachverhalts sind durch die Modellierungsmethode Konfigurationsmechanismen zu unterstützen, die eine multiperspektivische Modelldarstellung ermöglichen. So ist der spezifische Informationsbedarf einzelner Nutzer gezielt zu erfüllen. Die Perspektivierbarkeit der Prozessmodelle stellt eine wichtige Voraussetzung zur Kommunikation und Anwendung der Modellierungsergebnisse dar.

- **Einfache Vergleichbarkeit verschiedener Modelle.** Kommunalverwaltungen haben zum großen Teil gleiche Aufgaben und erbringen ähnliche Dienstleistungen (Produkte). Daher sind Prozessmodelle einer einzelnen Verwaltung vom Grundsatz her in hohem Maße verallgemeinerbar. Konfigurierbare Referenzmodelle können in diesem Kontext als Anhaltspunkt zur Prozessverbesserung genutzt werden und die gemachten Erfahrungen anderer Kommunalverwaltungen integrieren. Die Vergleichbarkeit verschiedener Modelle, welche nicht zuletzt maßgeblich durch die Modellierungsmethode beeinflusst wird, ist hierzu eine unabdingbare Voraussetzung.

Die erhobenen domänenübergreifenden und domänenspezifischen Anforderungen an eine Modellierungsmethode im eGovernment erheben keinen Anspruch auf Vollständigkeit. Weiterhin sind sie in Hinblick auf den Modellierungszweck (siehe Kapitel 2) und in Hinblick auf die Rahmenbedingungen der zu untersuchenden Organisation (bspw. vorhandenes methodisches Know-How bei beteiligten Mitarbeitern) entsprechend zu gewichten.

3.3 Vergleich von Modellierungsmethoden

Im Zentrum des Projekts mit der Stadtverwaltung Emsdetten stand die Organisationsgestaltung, die in Hinblick auf die Verbesserung des Einsatzes von Anwendungssystemen erfolgen sollte. Umfassendes methodisches Kow-How zur Prozessmodellierung konnte bei den Mitarbeitern der Stadtverwaltung nicht vorausgesetzt werden.

Drei verschiedene Methoden zur Prozessmodellierung wurden im Rahmen dieses Projekts vergleichend untersucht und mit den Mitarbeitern diskutiert. Dazu gehören die erweiterte Ereignisgesteuerte Prozesskette (eEPK) [vgl. KNS92], Petri-Netze [vgl. Ba90] und Wertschöpfungskettendiagramme (WSK-Diagramme). Die Darstellung der Ergebnisse findet sich in Tabelle 1.

Anforderungen an Modellierungsmethoden	eEPK	Petri-Netze	WSK-Diagr.
Einfachheit der Darstellung	+	-	++
Formalisierungsgrad	+	++	-
Verständlichkeit des Modellinhalts	+	-	++
Ganzheitliche Perspektive	++	+	+
Prozess-Controlling	+	+	+
Werkzeugunterstützung	++	++	++
Unterstützung von **Prozesssimulationen**	+	++	-
Abbildung von **Interaktionspunkten**	-	-	-
Modellierung **verteilter Abläufe**	-	+	-
Multiperspektivische Darstellung	-	-	-
Vergleichbarkeit verschiedener Modelle	+	-	+

Tab. 1: Gegenüberstellung von eEPKs, Petri-Netzen und WSK-Diagrammen

Auf Basis dieser Kriterien wurde die eEPK als Modellierungsmethode ausgewählt, da sie erstens eine ganzheitliche und integrierende Sicht der Betriebssituation ermöglicht und zweitens trotz der Einfachheit und Klarheit der Darstellung einen entsprechenden Formalisierungsgrad aufweist, welcher eine sich anschließende Anwendungssystemgestaltung zulässt. Petri-Netze wurden vor allem aufgrund ihres hohen Grades inhärenter Komplexität nicht zur Prozessmodellierung verwendet, da auch die Darstellung einfacher Fakten häufig schwierig zu verstehen ist. Der von vielen Autoren gelobten syntaktischen Klarheit stand die schwere Nachvollziehbarkeit von Seiten der Verwaltungsmitarbeiter gegenüber. Die Verwendung von WSK-Diagrammen wurde wegen ihrer mangelnden Ausdrucksstärke abgelehnt, welche die Modellierung komplexer Betriebssituationen erheblich erschwert.

Den sich vor allem aus den domänenspezifischen Anforderungen ergebenden Defiziten der eEPK sollte im Ablauf des Modellierungsprojekts konstruktiv begegnet werden.

4 Ordnungsrahmen und Identifikation von Handlungsfeldern

Vor der Ist-Modellierung müssen relevante Problembereiche identifiziert und angesichts knapper finanzieller und personeller Ressourcen zunächst klassifiziert und dann priorisiert werden. Auf Basis eines im Rahmen von diversen Interviews mit den Mitarbeitern der Verwaltung erarbeiteten strukturierenden Ordnungsrahmens (vgl. Abb. 2) wurde ein mehrstufiges Verfahren zur Klassifikation und Priorisierung möglicher Handlungsfelder für Prozessverbesserungen angewendet (vgl. [BAN03b]).

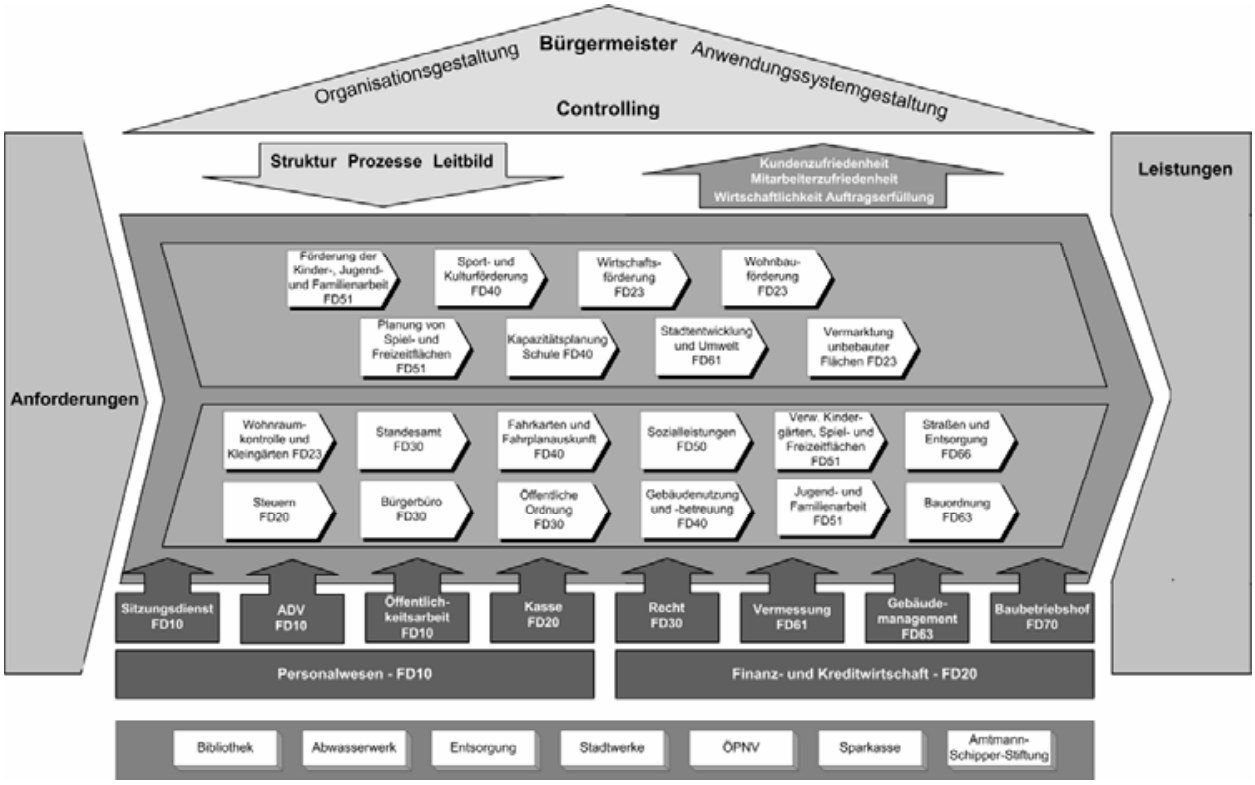

Abb. 2: Ordnungsrahmen für die Stadtverwaltung Emsdetten

Entwicklung des Ordnungsrahmens

Neben dem Umfeld der Verwaltung (Bürger, Unternehmen und andere Verwaltungen), aus welchem Anforderungen (z. B. in Form von Anträgen) artikuliert und an welches Leistungen abgegeben werden, lassen sich vier Bereiche identifizieren. Von oben nach unten sind dies Führungsprozesse, strategische Kernprozesse, operative Kernprozesse und Supportprozesse. Im Bereich der *Führungsprozesse* werden die technischen und organisatorischen Rahmenbedingungen durch Vorgabe von Leitbildern, Strukturen und Prozessen festgelegt. Der Grad der Zielerreichung der Verwaltung wird durch Kennzahlen über Kundenzufriedenheit, Mitarbeiterzufriedenheit, Auftragserfüllung und Wirtschaftlichkeit gemessen. *Strategische Kernprozesse* wie Stadtplanung, Stadtentwicklung, Wirtschaftsförderung etc. dienen der Schaffung geeigneter Rahmenbedingungen für die Erreichung der vorgegebenen Ziele.

Die Schaffung von Rahmenbedingungen steht hier im Vordergrund. *Operative Kernprozesse* dienen der Erbringung von Verwaltungsdienstleistungen mit direktem Bürger- und Unternehmenskontakt. Hierzu zählen vor allem Dienstleistungen, wie sie in den letzten Jahren zunehmend in so genannten Bürgerbüros zentralisiert wurden (z. B. die Bestellung von Ausweisdokumenten, Lohnsteuerkarten, Melderegisterauskünften etc.). Hier liegt der Fokus auf effizienter Leistungserbringung. *Supportprozesse* dienen der Unterstützung von Kernprozessen. Beispiele aus dem Praxisbeispiel sind z. B. das Rechtsamt oder die EDV-Abteilung, welche Kernkompetenzen vorhalten, die bei Bedarf abgerufen werden können. Ebenso zu diesem Bereich zählen Querschnittsfunktionen wie das Personalwesen und die Finanz- und Kreditwirtschaft.

Da in Emsdetten eine stärkere Bürgerorientierung und gleichzeitig Effizienzgewinne angestrebt wurden, konnten mit Hilfe des Ordnungsrahmens die möglichen Gestaltungsfelder auf den Bereich der operativen Kernprozesse als Grundlage einer weiteren Klassifikation und Priorisierung eingeschränkt werden.

Klassifikation und Selektion von Handlungsfeldern

Im ersten Schritt wurde ein Klassifikationsschema auf potenziell zu optimierende Kernprozesse angewendet. Die Strukturierung der Dienstleistungen erfolgte entlang einer internen und einer externen Perspektive, um sowohl Verbesserungspotenziale für die Verwaltung (Effizienzsteigerung) als auch für den Bürger als Kunden (Steigerung der Leistungsqualität) darstellen zu können. Ein Maß für die Qualität einer Dienstleistung kann der Interaktionsgrad zwischen Bürger und Kommune sein. Weit verbreitet sind die drei Stufen Information, Kommunikation und Transaktion [BS99, S. 155 f.; BB01, S. 56]. Ein Maß für die Effizienz einer Dienstleistung ist ihr Integrationsgrad. Eine Abstufung kann sich beispielsweise daran orientieren, ob eine Dienstleistung mit Medienbrüchen, medienbruchfrei oder komplett automatisiert erbracht wird. Kombiniert man den Interaktionsgrad mit dem Integrationsgrad, so entsteht eine Matrix, mit dessen Hilfe sich die einzelnen schon vorhandenen Dienstleistungen einer Kommune einordnen und klassifizieren lassen. Zur Selektion geeigneter Dienstleistungen auf Basis des Klassifikationsschemas wurde ein zweistufiges Vorgehen, bestehend aus der sukzessiven Anwendung erstens der Portfoliomethode und zweitens der Profilmethode (vgl. exemplarisch [Th89]), vorgeschlagen. Dabei nimmt von Stufe zu Stufe die Anzahl der betrachteten Dienstleistungen ab- und der Präzisionsgrad der Untersuchung durch eine steigende Anzahl von Unterscheidungskriterien zu (zur Ausführlichen Anwendung des mehrstufigen Verfahrens zur Prozessklassifizierung und -priorisierung vgl. [BAN03b]). Im Fallbeispiel wurde letztendlich das Baugenehmigungsverfahren zur Reorganisation ausgewählt.

5 Schwachstellenanalyse und Prozessverbesserung

Schwachstellenanalyse auf Basis der Ist-Modellierung

Im Zuge der Ist-Modellierung und -Analyse wird der aktuelle Stand der Abläufe erfasst und in Hinblick auf den Erreichungsgrad der verfolgten Ziele analysiert und bewertet. Damit dient die Ist-Modellierung der Schaffung von Transparenz in der Verwaltung, welche das Verständnis für fachliche Zusammenhänge und Probleme fördert. Sie bildet die Grundlage zur Identifikation von Schwachstellen und der sich anschließenden Beschreibung von Verbesserungspotenzialen. Im Rahmen des Praxisbeispiels wurde der Prozess des Baugenehmigungsverfahrens in einer mehrwöchigen Interviewphase analysiert und dokumentiert. Die Modellierung erfolgte mit erweiterten Ereignisgesteuerten Prozessketten.

Dabei konnten vor allem folgende Merkmale bzw. Schwachstellen festgestellt werden:

- Häufige Bearbeiterwechsel
- Viele beteiligte Ämter
- Zahlreiche Medienbrüche

Zusätzlich wurde ein Informationsdefizit sowohl bei den beteiligten Fachdiensten als auch bei den Antragstellern festgestellt (unklare Rechtslage, schwer verständliche Formulare etc.). Dies resultiert einerseits in einer Vielzahl von Rückfragen und andererseits in einer hohen Zahl unvollständiger oder nicht genehmigungsfähiger Anträge.

Prozessumgestaltung und Soll-Modellierung

Zur Verbesserung der identifizierten Schwachstellen wurde auf Basis der Ist-Prozesse und Handlungsempfehlungen der Kommunalen Gemeinschaftsstelle für Verwaltungsvereinfachung (KGSt) [Kg01, S. 39 ff.] eine Reihe von Maßnahmen konzipiert:

- Einführung von Baukonferenzen
- Einführung von Prozessmanager
- Nutzung der Workflow-Komponente der Fachanwendung Bauen
- Einführung eines Geoinformationssystems

Das vorgestellte Vorgehensmodell und die vorgeschlagene Modellierungssprache haben sich in dem Praxisbeispiel als zielführend erwiesen. Die in der eEPK-Notation erstellten Prozessmodelle sind durch die Mitarbeiter der Verwaltung positiv aufgenommen worden. Die darin enthaltenen Vorschläge zur Prozessverbesserung sind entweder bereits umgesetzt oder stehen kurz vor der Einführung.

6 Erweiterungen der eEPK im eGovernment

Das Praxisbeispiel zeigt aber auch, dass die aus den Besonderheiten der Domäne eGovernment erwachsenden Anforderungen an Modellierungssprachen durch existierende Ansätze nicht vollständig erfüllt werden und weiterer Forschungsbedarf besteht. Nicht alle Schwachstellen lassen sich direkt aus den Prozessmodellen erkennen und nicht alle Verbesserungsmaßnahmen spiegeln sich in veränderten Soll-Prozessmodellen wider. Weiterhin kann beispielsweise die Bilateralität (Kunde, Verwaltung) und die Dezentralität des Verwaltungshandelns nicht hinreichend abgebildet werden.

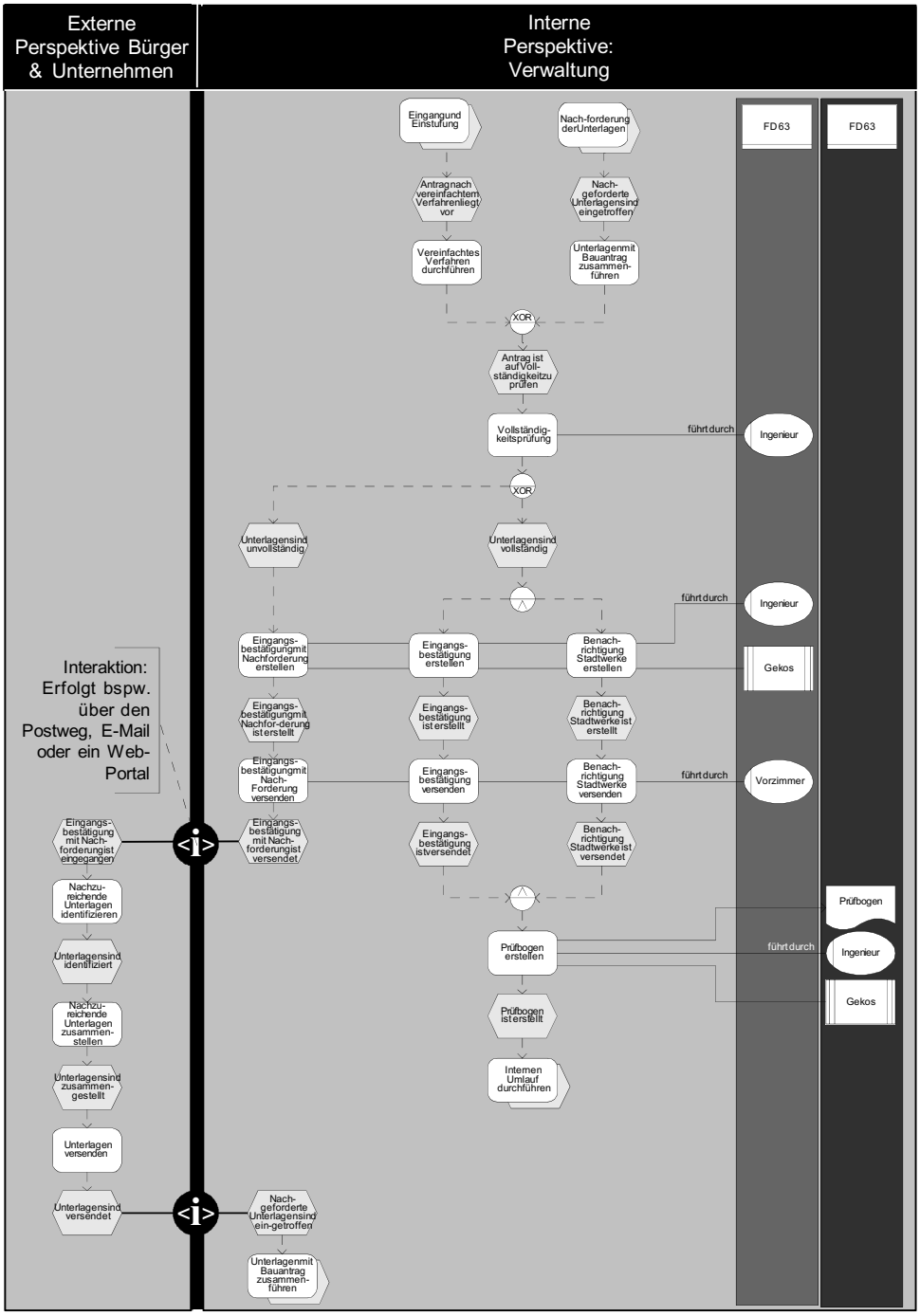

Abb. 2: Modifizierte eEPK am Beispiel der Vorprüfung im Baugenehmigungsverfahren

Am Beispiel der Vorprüfung im Kontext des kommunalen Baugenehmigungsverfahrens werden hier vor allem die Erweiterungen um

- die Spaltendarstellung und
- die Darstellung der Interaktionspunkte

ersichtlich (vgl. Abb.2).

Dabei orientiert sich die Einteilung der Spalten an Organisationseinheiten respektive Fachdiensten. Diesen Einheiten können nun dezidiert auf Prozessebene Wissensressourcen (Dokumente), Mitarbeiterressourcen (Stellen) und Technologieressourcen (Anwendungssysteme) zugewiesen werden. Eine solche auch grafisch explizierte Ressourcenzuordnung erlangt vor allem durch die sich im Zuge der Neuen Kommunalen Finanzmanagements auch in der öffentlichen Verwaltung etablierenden erfolgsorientierten Steuerung besondere Bedeutung. Ferner sind sowohl der Grad der Dezentralität des Arbeitens als auch die Prozessschnittstellen expliziert.

Die dargestellten Interaktionspunkte können durch eine Interaktionssicht zusammengeführt und somit zur im eGovernment zentralen Interaktionsoptimierung verwendet werden. In Kombination mit der Explikation des bilateralen Prozesses zwischen dem Bürger/Unternehmen und der Verwaltung dienen diese der Analyse der Qualitätswahrnehmung der Verwaltungsleistung durch den Kunden. Handlungsempfehlungen in Bezug auf die Interaktionsgestaltung lassen sich auf diese Weise einfacher ableiten. Besondere Berücksichtigung findet hier die sonst häufig vernachlässigte Sicht auf die kundenseitigen Prozesse, deren explizite Modellierung ebenfalls zum Teil erhebliche Prozessverbesserungspotenziale aufzeigt.

7 Fazit und weiterer Forschungsbedarf

Das vorgestellte Vorgehensmodell und die Verwendung der erweiterten Ereignisgesteuerte Prozesskette haben sich im vorgestellten Beispiel bewährt. Es hat sich jedoch ebenfalls gezeigt, dass die bestehenden Modellierungssprachen zum Teil wesentliche, vor allem domänenspezifische Anforderungen nicht erfüllen. Mit den vorgenommenen Erweiterungen der eEPK soll diesem Problem konstruktiv begegnet werden.

Darüber hinaus ist festzustellen, dass erhebliche Strukturanalogien zwischen verschiedenen Verwaltungsprozessen innerhalb einer Behörde und in noch stärkerem Maße bei gleichartigen Prozessen zwischen Behörden bestehen. So ist bspw. die Struktur des Ordnungsrahmens wegen des weitgehend einheitlichen Aufgabenspektrums der deutschen Kommunen mit geringen Anpassungen übertragbar.

Im Zuge einer weitreichenden prozessorientierten und IT-unterstützten Verwaltungsmodernisierung bietet die Entwicklung eines Referenzprozessmodells als Speicher von Domänenwissen Potenzial, die Komplexität von eGovernment-Projekten erheblich zu reduzieren und deren Umsetzung durch die Orientierung an Referenzprozessen zu vereinfachen.

Literatur

[Ba90] Baumgarten, B.: Petri-Netze. Grundlagen und Anwendungen. Mannheim, 1990.

[BAN03a] Becker, J.; Algermissen, L.; Niehaves, B.: E-Government – State-of-the-Art und Entwicklungsperspektiven. In: Becker, J. et al. (Hrsg.): Arbeitsberichte des Instituts für Wirtschaftsinformatik, Nr. 94. Münster, 2003.

[BAN03b] Becker, J.; Algermissen, L.; Niehaves, B.: Prozessmodellierung als Grundlage des E-Government – Ein Vorgehensmodell zur prozessorientierten Organisationsgestaltung am Beispiel des kommunalen Baugenehmigungsverfahrens. Erscheint in: Tagungsband Wirtschaftsinformatik 2003.

[Be+02] Becker, J. et al.: Konfigurative Referenzmodellierung. In: Becker, J.; Knackstedt, R. (Hrsg.): Wissensmanagement mit Referenzmodellen. Heidelberg, 2002, S. 25-144.

[BB01] Boller, R.; Beuchat, A.: Vertrauen und Sicherheit im Netz. In: Gisler, M.; Spahni, D. (Hrsg.): eGovernment. 2. Aufl., Bern u. a., 2001, S. 53-74.

[BS99] Budäus, D.; Schwiering, K.: Die Rolle der Informations- und Kommunikationstechnologien im Modernisierungsprozeß öffentlicher Verwaltungen. In: Scheer, A.-W. (Hrsg.): Electronic Business und Knowlege Management. Heidelberg, 1999, S. 143-165.

[Ga97] Galler, J.: Vom Geschäftsprozessmodell zum Workflow-Modell. Wiesbaden. Zugl. Dissertation. Universität Saarbrücken 1997.

[Kg01] KGSt: Management der Bauordnung. Ziele, Produkte und Organisationsgestaltung. Köln, 2001.

[KNS92] Keller, G.; Nüttgens, M.; Scheer, A.-W.: Semantische Prozeßmodellierung auf Grundlage „Ereignisgesteuerter Prozeßketten“. In: Scheer, A.-W. (Hrsg.): Veröffentlichungen des Instituts für Wirtschaftsinformatik. Heft 89. Saarbrücken, 1992.

[Le95] Lenk, K.: Business Process Reengineering. Sind die Ansätze der Privatwirtschaft auf die öffentliche Verwaltung übertragbar?. In: Traunmüller, R. (Hrsg.): Geschäftsprozesse in öffentlichen Verwaltungen. Neugestaltung mit Informationstechnik. Heidelberg, 1995. S. 27-43.

[MB00] Meffert, H.; Bruhn, M.: Dienstleistungsmarketing. Grundlagen – Konzepte – Methoden. Wiesbaden, 2000. 3. Auflage.

[RSD03] Rosemann, M.; Schwegmann, A., Delfmann, P.: Vorbereitung der Prozessmodellierung. In: Becker, J.; Kugeler, M.; Rosemann, M. (Hrsg.): Prozessmanagement. 4. Aufl., Berlin u. a., 2003, S. 47-105.

[Sc+96] Scheer, A.-W.; Bold, M.; Heib, R.: Geschäftsprozessmodellierung als Instrument zur Gestaltung von Controlling-Systemen in der öffentlichen Verwaltung. In: Scheer, A.-W.; Friederichs, J. (Hrsg.): Innovative Verwaltungen 2000. 1996, S. 119-130.

[Th89] Thoma, W.: Erfolgsorientierte Beurteilung von F&E-Projekten. Darmstadt, 1989.

[We22] Weber, M.: Wesen, Voraussetzungen und Entfaltung der bürokratischen Herrschaft. In: Weber, M. (Hrsg.): Wirtschaft und Gesellschaft. Tübingen, 1922. S. 551-579.

Multiperspektivische ereignisgesteuerte Prozessketten

Jörg Becker, Patrick Delfmann, Thorsten Falk, Ralf Knackstedt

Westfälische Wilhelms-Universität Münster
Institut für Wirtschaftsinformatik
Leonardo-Campus 3, D-48149 Münster
E-Mail: {becker|ispade|isthfa|israkn}@wi.uni-muenster.de

Abstract: Im Rahmen der Informationssystemgestaltung unterscheiden sich die Projektbeteiligten insbesondere hinsichtlich ihrer Anforderungen an die Inhalte und Repräsentationsformen der für die Anwendungssystem- und Organisationsgestaltung verwendeten Modellen. Die multiperspektivische Modellierung stellt einen viel versprechenden Ansatz dar, um diesem Umstand durch die Bereitstellung perspektivenspezifischer Sichten auf Informationsmodelle gerecht zu werden. Der Beitrag stellt eine Erweiterung ereignisgesteuerter Prozessketten durch Konfigurationsmechanismen vor, welche die Erstellung und Nutzung von Perspektiven auf Prozessmodelle unterstützt.

1 Perspektiven auf Prozessmodelle

Die wissenschaftliche Diskussion über die Qualität von Prozessmodellen wurde in den letzten Jahren wesentlich durch die Entwicklung von allgemeinen Modellierungsempfehlungen, welche auf die syntaktische Korrektheit der Modelle und der Einhaltung der Grundsätze ordnungsmäßiger Modellierung bei der Erstellung von Prozessmodellen abzielen [BRS95; Ro96; BRU00], geprägt. Ein dominierendes Kriterium, das die Qualität von Prozessmodellen determiniert, ist dabei die Entsprechung der Modellkonzeption und -repräsentation mit den Anforderungen der jeweiligen Modellnutzergruppe [RS99, S. 25f.; Be02, S. 28ff.; RSD03, S. 49]. So werden z. B. innerhalb eines prozessbasierten Reorganisationsprojektes diese Modellnutzergruppen durch die verschiedenen organisatorischen Rollen der Projektteilnehmer bestimmt. Ein Sachbearbeiter benötigt hierbei lediglich die für ihn relevanten Prozessmodelle und deren Schnittstellen. Aus der Sicht des Managers der zu reorganisierenden Unternehmung sollten die Modelle in aggregierter Form vorliegen und durch erweiterte visuelle Merkmale, wie z. B. ansprechender Farb- oder Symboldarstellung, gekennzeichnet sein. Einem Anwendungsentwickler sollten sehr detaillierte Modelle vorgelegt werden, damit sich die aus der Reorganisation ergebenden Konsequenzen im Anwendungssystem berücksichtigt werden können. Die verschiedenen Anforderungen der Nutzergruppen resultieren aus den Differenzen ihrer subjektabhängigen Vorstellungswelten und den daraus resultierenden Problemlösungsansätzen [Be01a; Be02, S. 30ff.]. Ein Ansatz zur expliziten Vermittlung zwischen diesen Vorstellungswelten findet sich in der Bereitstellung von anforderungsgerechten Perspektiven wieder [Fr94, S. 36f.]. Perspektiven werden dadurch determiniert, welcher Modellierungszweck verfolgt wird, welche organisatorische Rolle die Nutzer einnehmen und welche persönlichen Präferenzen bzgl. der Modellkonzeption und -repräsentation bestehen [Be02, S. 38ff., RSD03, S. 49]. Die Entwicklung und Bereitstellung perspektiven-

spezifischer Modelle wird unter dem Begriff der *multiperspektivischen Informationsmodellierung* diskutiert [Fi92; RS99, S. 25f.; RSD03, S. 52].

Soll den Anforderungen mehrerer Nutzergruppen entsprochen werden, d. h. sollen mehrere Perspektiven berücksichtigt werden, ergeben sich für den Modellersteller zwei Alternativen. Modelle könnten einerseits für unterschiedliche Perspektiven redundant vorgehalten werden. Nachteile dieser Vorgehensweise sind die üblichen, mit Redundanzen verbundenen Zusatzaufwände wie erhöhter Pflegeaufwand und Gefahr von Inkonsistenzen. Andererseits kann ein Gesamtmodell derartig erstellt werden, dass es die für sämtliche Perspektiven relevanten Elemente redundanzfrei enthält. Den einzelnen Vertretern der Perspektiven werden die für sie relevanten Modelle bereitgestellt, indem ihnen eine View auf das Gesamtmodell zur Verfügung gestellt wird, die das Gesamtmodell um die nicht relevanten Inhalte reduziert. Die Maßnahmen, welche vom Modellierer zur Erstellung dieser Views durchgeführt werden müssen, können hierbei durchaus zu einer komplexen Aufgabe werden. Im Folgenden wird anhand ereignisgesteuerter Prozessketten[1] (EPKs) [KNS92] die Umsetzung der zweiten Variante diskutiert.

Dabei wird in Abschnitt 2 zunächst eine metamodellbasierte Formalisierung der EPK vorgenommen, um eine konzeptionelle Basis für die multiperspektivische Erweiterung der Modellierungstechnik durch Konfigurationsmechanismen zu schaffen. Darauf aufbauend werden in Abschnitt 3 die notwendigen Erweiterungen der Modellierungstechnik vorgenommen. Abschnitt 4 behandelt die Problematik und Notwendigkeit von Konsistenzsicherungsmechanismen, die bei der Verwendung der Konfigurationsmechanismen auftreten können. Nach einem Praxisbeispiel in Abschnitt 5 schließt der Beitrag mit einem Ausblick in Abschnitt 6.

2 Metamodellbasierte Formalisierung der EPK

Das hier vorgestellte Verfahren zur Erstellung multiperspektivischer ereignisgesteuerter Prozessketten basiert auf der Manipulation von Modellstrukturen durch die Anwendung von Konfigurationsmechanismen. Da hierbei die strukturellen Veränderungen auf Ebene der Sprachspezifikation sowie deren Auswirkungen auf die Struktur der Modelle im Zentrum der Betrachtung stehen, empfiehlt es sich, die Sprache der EPK formalisiert darzustellen. Dies kann wiederum durch Informationsmodelle – sogenannte sprachorientierte Metamodelle – erfolgen [St96, S. 24ff.]. Als Notationsform eignen sich insbesondere Strukturspachen wie das Entity-Relationship-Modell [Ch76]. Ein erweiterter Dialekt dieser Modellierungssprache wird im vorliegenden Beitrag zur Metamodellierung verwendet.[2]

Das zentrale Element der EPK Sprache ist das *Prozesselement* (vgl. Abbildung 1). Alle am Prozessgraphen beteiligten Knoten werden als Spezialisierung dieses Elements mo-

[1] Der Begriff EPK wird im Folgenden synonym zur erweiterten ereignisgesteuerten Prozesskette (eEPK) verwendet, die neben der Ablauflogik die Annotation von benötigten Ressourcen erlaubt.
[2] Es handelt sich hierbei um einen Dialekt, der die perspektivenabhängige Modifikation von Sprachen erlaubt (vgl. ausführlich [Be02, S. 77ff.]).

delliert. Hierzu gehören die Elemente *Prozessfunktion*, *Prozessereignis* und *Operator* (vgl. Spezialisierung des Entitytyps *Prozesselement*). Eine Prozessfunktion beschreibt eine Aktivität innerhalb eines Prozesses. Diese Aktivitäten werden beim Eintreten eines oder einer Kombination von Ereignissen ausgeführt und führen nach ihrer Beendigung selbst wieder zu dem Eintreten von Ereignissen. Somit kann ein Ereignis als Zustandswechsel interpretiert werden. Die notwendige Kombination der Ereignisse für die Ausführung einer Aktivität sowie die Definition der Menge der eintretenden Ereignisse nach erfolgreicher Ausführung werden über die Operatoren definiert. Hierdurch lassen sich parallele Prozessstränge und deren Wiederzusammenführung im Prozessmodell abbilden.

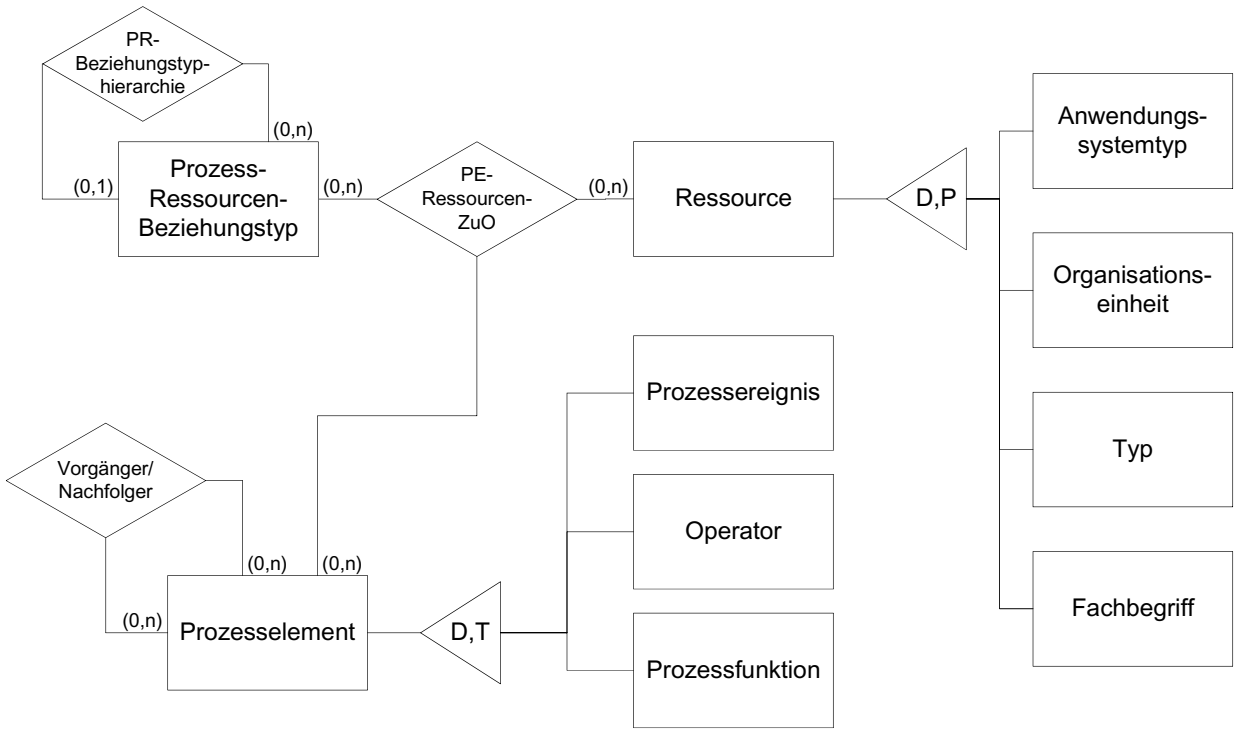

Abbildung 1: Metamodell für ereignisgesteuerte Prozessketten[3]

Die Prozesselemente können zueinander in Beziehung gestellt werden und bilden somit einen Graphen. Diese Beziehung wird durch den Relationshiptyp *Vorgänger/Nachfolger* dargestellt. Da die EPK allerdings als bipartiter Graph definiert ist, müssen Einschränkungen bzgl. dieser Beziehungen getroffen werden. So dürfen, abgesehen von Operatoren, nur jeweils verschiedene Prozesselemente aufeinander folgen. Weiterhin sind Regeln zu definieren, welche die Abfolge der Prozesselemente weiter einschränken. So muss z. B. eine Prozessfunktion mit einem oder mehreren Ereignissen beginnen und enden. Zur Veranschaulichung findet sich eine Auswahl dieser Regeln in Tabelle 1 wieder.[4] Neben dieser semi-formalen Darstellungsweise durch explizite Regeleinträge in einer Tabelle existieren formale Darstellungsformen der Syntax einer EPK [NR02].

[3] Das hier vorgestellte Metamodell beschreibt die Modellierungssprache der erweiterten EPK-Notation. Diese wurde jedoch aus Gründen der vereinfachten Darstellung des vorgestellten Verfahrens reduziert. Für eine vollständige Angabe des hier verwendeten Metamodells vgl. [Be02, S. 87].
[4] Eine vollständige Definition der Regeln findet sich in [Be02].

Vorgänger (VG)	Nachfolger (NF)	Kardinalität VG	Kardinalität NF
Ereignis	Funktion	(0,1)	(0,1)
Ereignis	Ereignis	(0,0)	(0,0)
Funktion	Ereignis	(0,1)	(0,1)
…	…	…	…
XOR-Operator	Funktion	(0,n)	(0,1)
…	…	…	…

Tabelle 1: Beschränkung der Vorgänger/Nachfolger-Typen der EPK (Auszug)

Zur Unterstützung des Prozessablaufes werden zum Teil Ressourcen benötigt, welche durch den Entitytyp *Ressource* dargestellt werden. Die verschiedenartigen Spezialisierungen von Ressource wie *Anwendungssystemtyp*, *Organisationseinheit*, *(Entity-* bzw. *Relationship-)Typ* und *Fachbegriff* unterstützen den Prozessablauf und fokussieren dabei jeweils eine andere Unterstützungsfunktion. Klassifiziert wird die Art der Unterstützung durch den *Prozess-Ressourcen-Beziehungstyp*. Als Beispiel können hier die Beziehungstypen *„führt aus“* oder *„ist Input für“* genannt werden. Die Beziehungstypen selbst stehen in einer hierarchischen Beziehung zueinander, welche durch den Relationshiptyp *PR-Beziehungstyphierarchie* ausgedrückt wird. Auf einer untergeordneten Hierarchieebene sind dabei Verfeinerungen des Beziehungstypen der übergeordneten Ebene wieder zu finden. Die Annotation der Ressource an das Prozesselement über einen Beziehungstyp wird durch den Relationshiptyp *PE-Ressourcen-ZuO* festgelegt. Da nicht alle ternären Beziehungen dieser Art zugelassen sind, muss festgelegt werden, welche Beziehungstypen für welche Kombinationen von Prozesselementen und Ressourcen erlaubt sind.

3 Perspektivenspezifische Konfiguration der EPK

Die Gestaltung von multiperspektivischen ereignisgesteuerten Prozessketten wird durch die Bereitstellung und Verwendung von ausgewählten Konfigurationsmechanismen vorgenommen, die in die Modellierungstechnik selbst durch Erweiterung des Metamodells und der Metamodellsprache integriert werden. Die Anwendung dieser Mechanismen führt zu perspektivenspezifisch konfigurierten Modellen. Für die Konstruktion konfigurierbarer Modelle erscheint es vorteilhaft, Konfigurationsmechanismen mit unterschiedlichem Wirkungsgrad bereitzustellen. Dieser Wirkungsgrad determiniert sich durch die Modellebene, auf welcher der Konfigurationsmechanismus angewendet wird. Konfigurative Veränderungen auf der Metamodellebene wirken sich auf alle Modelle aus, welche durch die entsprechende Modellierungssprache instanziiert worden sind, während Anpassungen auf Modellebene einen geringeren Wirkungsgrad und damit keinerlei weitere Auswirkungen außerhalb der Modellinstanz besitzen.

Zur konsistenten Definition der zulässigen konfigurativen Veränderungen im Metamodell der EPK werden diese auf einer weiteren Modellebene beschrieben. Diese Meta-Metamodellebene ist als Sprachdefinition der Metamodellebene aufzufassen. Die fol-

gende Vorstellung der Konfigurationsmechanismen[5] erstreckt sich daher jeweils über alle notwendigen Modellebenen.

Elementtypselektion: Das Metamodell für ereignisgesteuerte Prozessketten charakterisiert deren Elementtypen und deren Beziehungen zueinander. Der Konfigurationsmechanismus der *Elementtypselektion* erlaubt eine Variantenbildung dieses Modelltyps, indem Elementtypen innerhalb einer Perspektive selektiert und damit ausgeblendet werden. Die hierdurch entstandene *Modelltypvariante* (vgl. Abbildung 2) beinhaltet nun nicht mehr alle Elementtypen, die innerhalb des *Metamodells* der EPK maximal möglich sind (vgl. Relationshiptyp *EMM enthält MME*). Der Konfigurationsmechanismus kann z. B. angewendet werden, um den Elementtyp *Anwendungssystemtyp* innerhalb der Organisationsgestaltungsperspektive auszublenden.

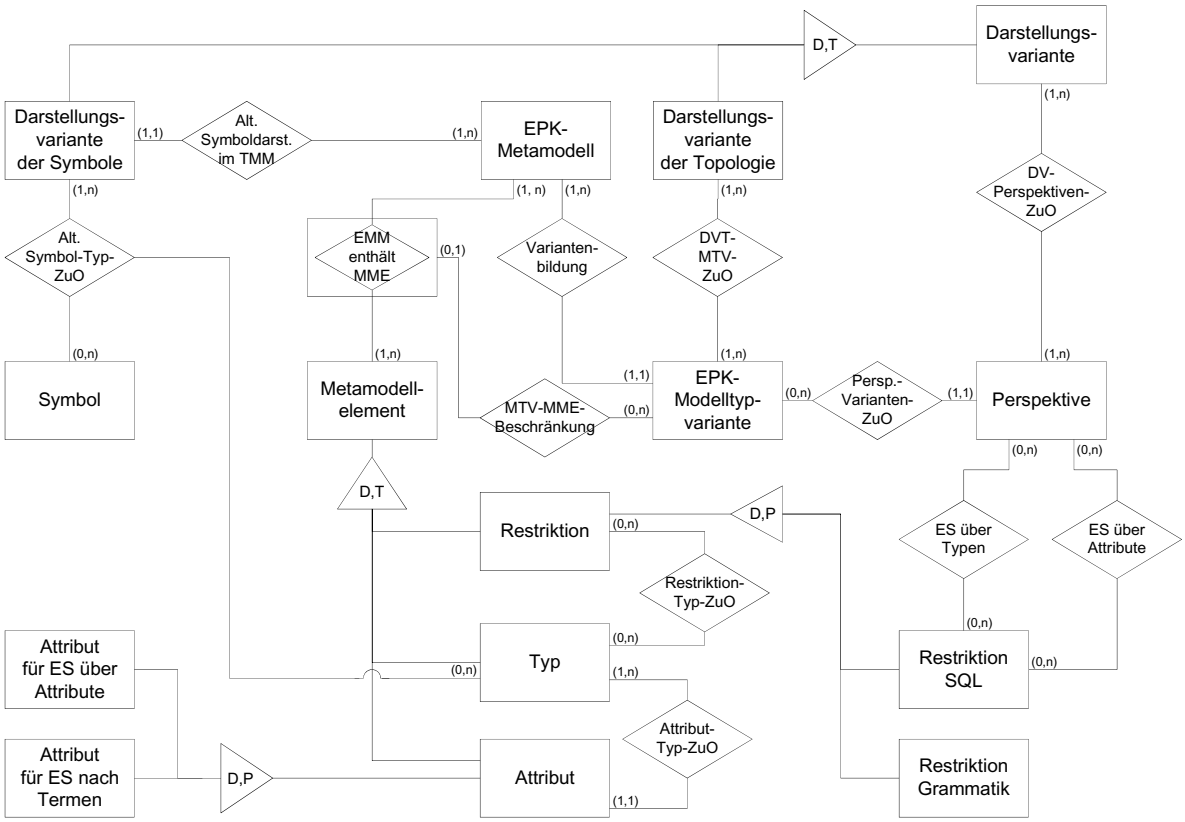

Abbildung 2: Auszug aus dem Meta-Metamodell für multiperspektivische ereignisgesteuerte Prozessketten

Der vorgestellte Konfigurationsmechanismus arbeitet auf Ebene der Sprachdefinition der Modellierungstechnik und wirkt sich somit auf alle Modellinstanzen der EPK-Technik gleichermaßen aus. Die nun folgenden Mechanismen der *Elementselektion* werden ebenfalls auf der Meta-Metamodellebene spezifiziert und erweitern somit die Sprache der Metamodellebene, selektieren die Modellelemente allerdings durch Informationen auf der Modellebene selbst und ziehen sich somit durch alle Ebenen hindurch.

[5] Bei den hier vorgestellten Konfigurationsmechanismen handelt es sich um eine Auswahl, die sich nach der Eignung der Mechanismen für eine Anwendung auf die EPK richtet. Eine ausführlichere Unterscheidung von Konfigurationsmechanismen findet sich bei [Be02].

Elementselektion über Typen: Die *Elementselektion über Typen* führt eine Selektion durch die Auswertung von Beziehungen auf Modellinstanzebene durch. Ein Entitytyp kann durch eine Beziehung zu einem anderen Entitytyp typisiert werden. Beispielhaft typisiert der Entitytyp *Prozess-Ressourcen-Beziehungstyp* im EPK-Metamodell die Beziehung zwischen einem Prozesselement und einer prozessunterstützenden Ressource. Mit Hilfe dieses Mechanismus können bspw. Beziehungen auf den Typ „*führt aus*" eingeschränkt werden. Da die Typisierung über Instanzbeziehungen auf der Modellebene vorgenommen wird, können dort ohne Anpassungen am Metamodell neue Typen eingeführt werden. Zur Repräsentation dieses Mechanismus definiert das Meta-Metamodell eine perspektivenabhängige Existenz von Instanzbeschränkungen für Typen auf Metamodellebene (vgl. Entitytyp *Restriktion SQL* in Abbildung 2). Die Sprache der Restriktion wird als SQL-ähnlich definiert, da sie eine Sicht auf die für den betroffenen Typ definierten Instanzen erzeugt.[6] Auf Metamodellebene können somit SQL-Restriktionen an den typisierenden Entitytyp annotiert werden. Sollen bspw. nur Prozess-Ressourcen-Beziehungen vom Typ *„führt aus"* eingeblendet werden, so wird an den Entitytyp *Prozess-Ressourcen-Beziehungstyp* im Metamodell die *SQL-Restriktion „WHERE Name='führt aus'"* annotiert. Die Ausführung dieser Restriktion auf Modellebene bewirkt die Ausblendung aller Instanzen, die über die Typisierung nicht erfasst worden sind.

Elementselektion über Attribute: Die *Elementselektion über Attribute* ähnelt der Selektion über Typen. Auch hier werden unter Rückgriff auf Restriktionen, die an Entitytypen des Metamodells annotiert werden, Instanzen auf der Modellebene ausgewählt. Allerdings basiert die Selektion hier auf Attributwertausprägungen der entsprechenden Entitytypen. Die Attributgestaltung wird vom Modellersteller auf Metamodellebene vorgenommen. Die perspektivenabhängige Definition des Mechanismus geschieht durch den Relationshiptyp *Elementselektion über Attribute* im Meta-Metamodell (vgl. Abbildung 2). Als Anwendungsbeispiel können die Prozessfunktionen einer EPK im Metamodell das Attribut *Automatisierungsgrad* erhalten. Mit Hilfe der Elementselektion über Attribute lassen sich dann auf Modellebene alle Instanzen von Prozessfunktionen ausblenden, die *vollständig manuell* ausgeführt werden, um so der Perspektive der Anwendungssystemgestaltung besser gerecht zu werden.

Elementselektion nach Termen: Die Abgrenzung von Elementen auf Modellebene über die perspektivenspezifische Selektion von Attributausprägungen stößt an ihre Grenzen, wenn für detaillierte Konfigurationsmaßnahmen jeweils neue Attribute eingeführt werden müssen und damit ihre Menge unübersichtlich zu werden droht. Daher wird der Konfigurationsmechanismus der *Elementselektion nach Termen* eingeführt, der die direkte Zuweisung auf Modellebene von Modellelementen zu Perspektiven über einen booleschen Term ermöglicht. Die Relevanz eines Modellelements für eine ausgewählte Perspektive hängt bei diesem Mechanismus von einem dem Element zugeordneten Term ab. Terme sind hierbei Attribute in Textform, deren Ausprägungen beschreiben, in welchen Perspektiven das jeweilige Modellelement zur Verfügung steht. Zu Auswertungs-

[6] SQL ist eine Abkürzung für Structured Query Language und ist eine Sprache zur Administration, Manipulation und Abfrage von relationalen Datenbanken [DD97].

zwecken haben diese einer vordefinierten Grammatik (vgl. Abbildung 3) zu gehorchen [Sc99, S. 144ff.].

```
<Term>                                  ::= <Ausdruck> {<Operator> <Ausdruck>}
<Ausdruck>                              ::= <Präfix> "Perspektive"
                                            <Perspektiven-Ausprägungsliste>
<Perspektiven-Ausprägungsliste>         ::= "(" <Präfix> <Perspektiven-Ausprägungsliste>
                                            { <Operator> <Präfix> <Perspektiven-Ausprägungsliste> } ")"
<Perspektiven-Ausprägungsliste>         ::= <Perspektive>
<Operator>                              ::= "|" | "+"
<Präfix>                                ::= "NOT" | <leer>
```

Abbildung 3: Auszug aus der Grammatik zur Elementselektion nach Termen.[7]

Realisiert wird die Elementselektion nach Termen, indem im Meta-Metamodell ein spezieller Attributtyp *Attribut für Elementselektion nach Termen* eingeführt wird (vgl. Abbildung 2). Um die formale Korrektheit der Terme sicherzustellen, muss die Existenz einer *Restriktion Grammatik* definiert werden, die die Formulierung von Termen in der vorgestellten Grammatik sichert.

Die im Meta-Metamodell definierte Sprachkomponente macht sich auch auf Metamodellebene bemerkbar. Hier werden alle Metamodellelemente der EPK, für die eine Termzuweisung möglich ist, zu einem Metamodellelement generalisiert (vgl. Abbildung 4). Ein *für eine Termzuweisung geeigneter Elementtyp* steht mit dem *booleschen Konfigurationsparameterterm* in Beziehung. Dieser hat wiederum ein Attribut, welches den *Termtext* enthält und das eben dem auf Meta-Metamodellebene definierten Typ (*Attribut für Elementselektion nach Termen*) angehört. Die Restriktion, die an den Term annotiert ist, ist vom Typ *Restriktion Grammatik* und sichert die Formulierung des Termtextes in der bereits vorgestellten konkreten Grammatik (vgl. Abbildung 3).

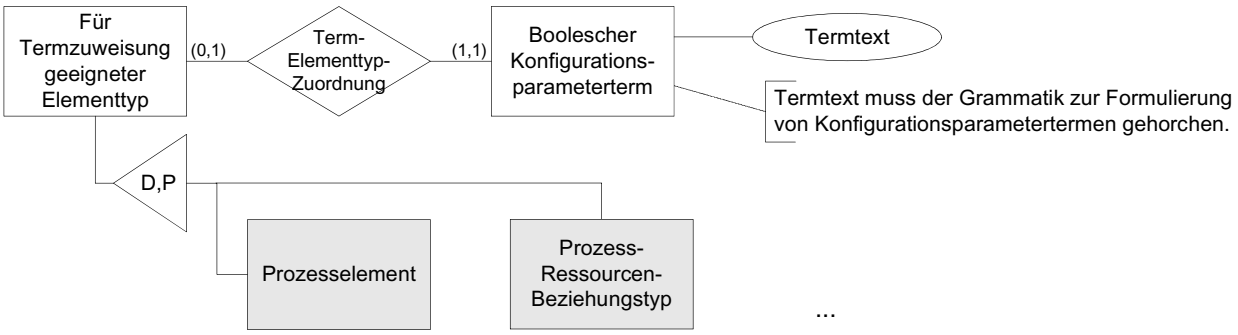

Abbildung 4: Metamodellkonstrukte zur Elementselektion nach Termen

Das Ergebnis für die Modellebene ist, dass jedem Modellelement, das dem Typ *für Termzuweisung geeigneter Elementtyp* angehört, textuell Terme hinterlegt werden können, die von einem entsprechenden Interpreter ausgewertet werden können. Anschließend kann eine Ausblendung gemäß der im Term definierten Konfigurationsparameterausprägungen durchgeführt werden.

[7] Auf die Darstellung atomarer Elemente der Grammatik (wie z. B. die Definition des gültigen Alphabetes und die Einschränkung von <Perspektive> auf gültige Elemente) ist hier verzichtet worden.

Darstellungsvariation: Durch den Mechanismus der *Darstellungsvariation* ergibt sich die Möglichkeit der perspektivenabhängigen Ausgestaltung von repräsentationellen Aspekten der Modelle [Be01b]. Zum einen kann über die *Darstellungsvariation der Symbole* die Darstellung der Modellelemente flexibel gestaltet werden. Die Zuordnung von perspektivenabhängigen Symbolen zu einem Elementtyp geschieht auf der Metamodellebene, deren Sprachdefinition auf Meta-Metamodellebene durch die Konstrukte *Symbol* und *Alternative-Symbol-Typ-ZuO* vorgenommen wird. Neben der reinen Repräsentation der Modellelemente kann ebenfalls die Anordnung der Symbole durch den Mechanismus der *Darstellungsvariation der Topologie* verändert werden. Eine adäquate Topologie sollte in jedem Fall den Grundsätzen ordnungsmäßiger Modellierung [BRS95; Ro96; BRU00] genügen. Im Rahmen deren Anwendung ergeben sich im Zuge der unterschiedlichen Priorisierung der zum Teil konfliktären Grundsätze Entscheidungsspielräume, die eine perspektivenspezifische Gestaltung von Topologien nahelegen. Eine beispielhafte abweichende Darstellung der EPK ist die Spaltendarstellung, bei der die Annotation von Ressourcen an Prozessfunktionen in eigens dafür vorgesehenen Spalten erfolgt [RSD03, S. 71f.].

4 Modellierungsbegleitende Konsistenzsicherungsmechanismen

Der Modellierer stellt sich bei der Konstruktion komplexer konfigurierbarer Prozessmodelle einer nicht trivialen Aufgabe. Je nach Anzahl und Verschiedenartigkeit der zu berücksichtigen Perspektiven und dem Verwendungsgrad der bereitgestellten unterschiedlichen Konfigurationsmechanismen verändert sich die zu handhabende Komplexität. Eine besondere hieraus resultierende Problematik ist die Notwendigkeit der Bereitstellung von Mechanismen, die während der Modellierungsphase den Modellierer mit einer Unterstützungsfunktion für die Einhaltung von Modellkonsistenzen versorgen (vgl. auch [Ha94; PF96]). Diese werden im Folgenden als Konsistenzsicherungsmechanismen bezeichnet.

Die Notwendigkeit für solche Mechanismen ergibt sich aus der Möglichkeit des Auftretens von Inkonsistenzen auf Modellebene nach perspektivenabhängiger Auswertung der in konfigurierbaren Prozessmodellen definierten Konfigurationsmechanismen. Die Auswertung dieser Mechanismen wird als Konfiguration des Prozessmodells bezeichnet. Eine Übersicht des Vorgehensmodells und die Einordnung der Konsistenzsicherungsmechanismen finden sich in Abbildung 5.

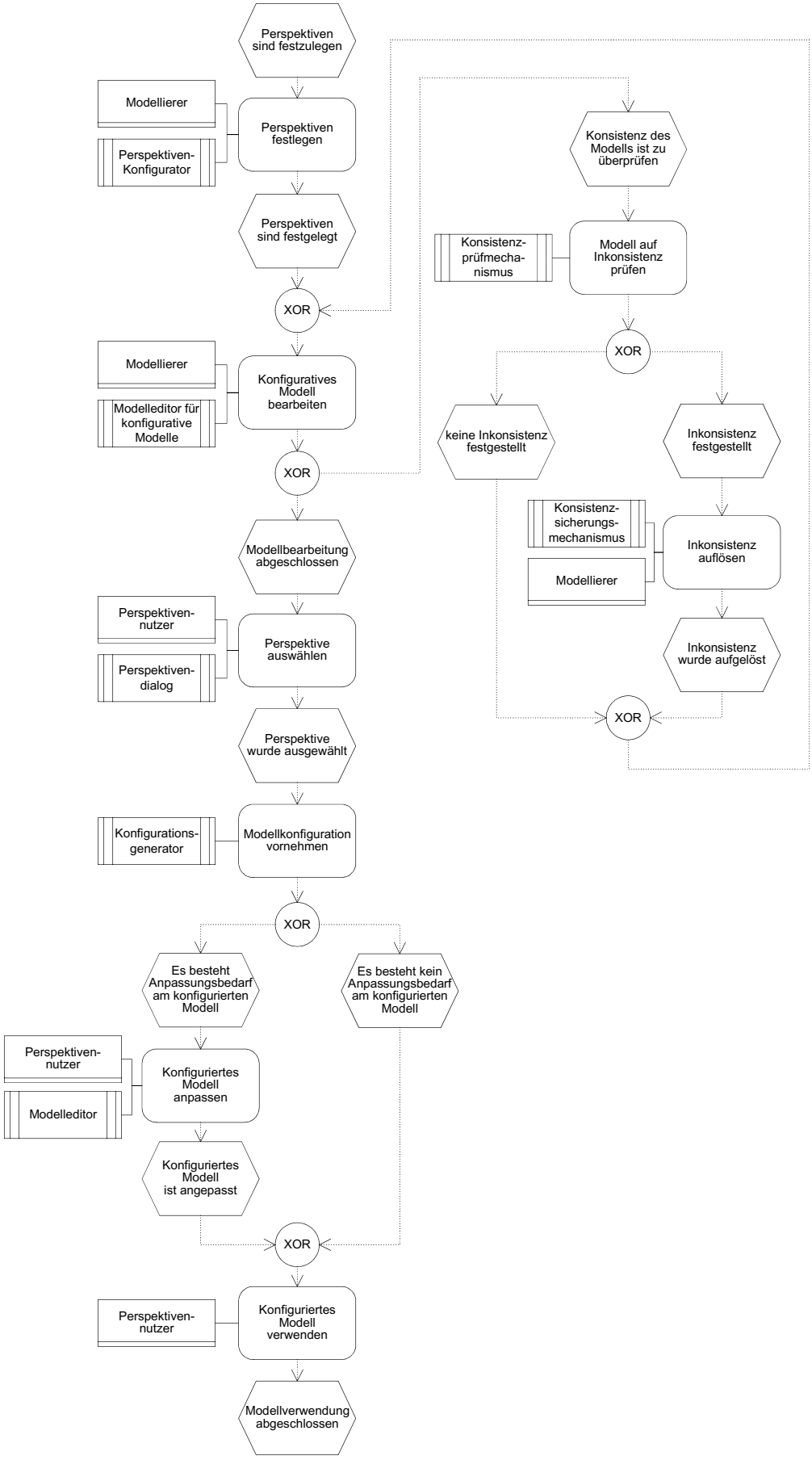

Abbildung 5: Vorgehensmodell einer werkzeuggestützten multiperspektivischen Modellierung

Um das Auftreten möglicher Inkonsistenzen durch uneingeschränktes Ausblenden bzw. Modifizieren konkreter Modellelemente durch den Vorgang der Konfiguration vermeiden zu können, greift der Unterstützungsmechanismus schon in der Phase der Modellerstellung. Während dieser Phase können unterschiedliche Kategorien von möglichen Inkonsistenzen erkannt und behoben werden, von denen im Folgenden eine Auswahl vorgestellt wird.

- Durch Löschung, Änderung oder Hinzufügen von Modellbestandteilen können in einem ersten Schritt ungültige Instanzen des Metamodells entstehen. Solche Inkonsistenzen lassen sich relativ einfach erkennen und teilweise automatisiert beheben, indem ein Abgleich mit dem Metamodell der verwendeten Modellierungssprache durchgeführt wird und das Modell entsprechend angepasst wird (vgl. Abbildung 6). Im Beispiel kann die Konsistenz des Modells automatisiert gesichert werden, da durch Ausblendung des Entitytyps *Organisationseinheit* sämtliche referenzierenden Instanzen des Relationshiptyps *PE-Ressourcen-ZuO*, welche Kanten zu Ressourcenobjekten repräsentierten, ebenfalls ausgeblendet werden.

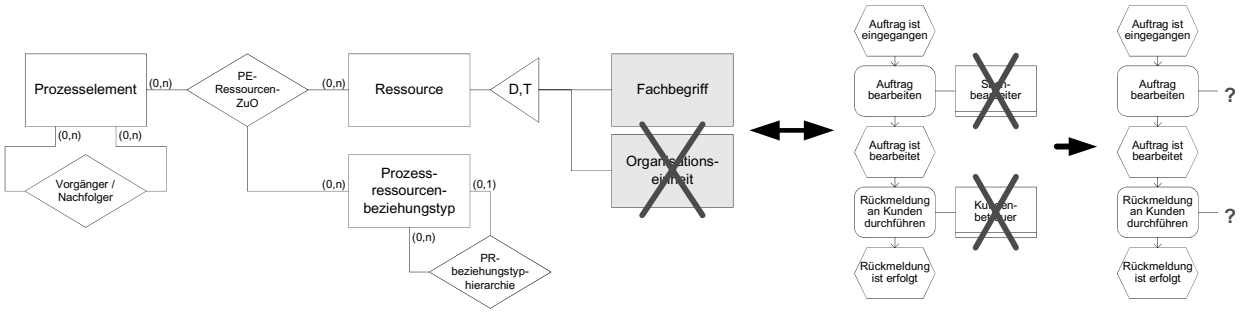

Abbildung 6: Rein syntaktische Inkonsistenzen

- Ein besonderer, EPK-spezifischer Fall ergibt sich, wenn zu Präsentationszwecken so genannte Trivialereignis [Be02, S. 109ff.] entfernt werden sollen (vgl. Abbildung 7).[8] Einerseits ist das Modell nun syntaktisch nicht mehr korrekt, da Teilprozesse existieren, die nicht mit einem Ereignis starten und enden [KNS92], andererseits ist der Kontrollfluss des Prozesses unterbrochen, wodurch die dem Gesamtmodell entsprechende Ablauflogik nicht mehr korrekt dargestellt wird. Zur Wiederherstellung könnte zum einen das Metamodell des Prozesses angepasst werden, um direkt aufeinander folgende Funktionen zuzulassen, und zum anderen der Kontrollfluss durch Verbindung der beiden Prozessstränge wieder vervollständigt werden. Derartige mögliche Inkonsistenzen können automatisiert bei der Modellierung erkannt werden, da bei Konfiguration des Modells eine Verletzung der Syntax vorliegen würde. Offensichtlich muss die Syntax der EPK zur Behebung der Inkonsistenz angepasst werden, was im Vorfeld, abhängig von der jeweiligen Perspektive zu spezifizieren ist (bspw. durch entsprechende Einträge in Tabelle 1). Die Behebung der möglichen Inkonsistenz ist auch in diesem Falle automatisiert möglich. Bei komplexeren Beschneidungen desselben Typs ist ggf. eine Anwenderinteraktion notwendig.

[8] In der Praxis werden Trivialereignisse, d. h. rein transitorische Ereignisse, häufig nicht dargestellt wodurch die Forderung eines bipartiten Graphen aufgeweicht wird.

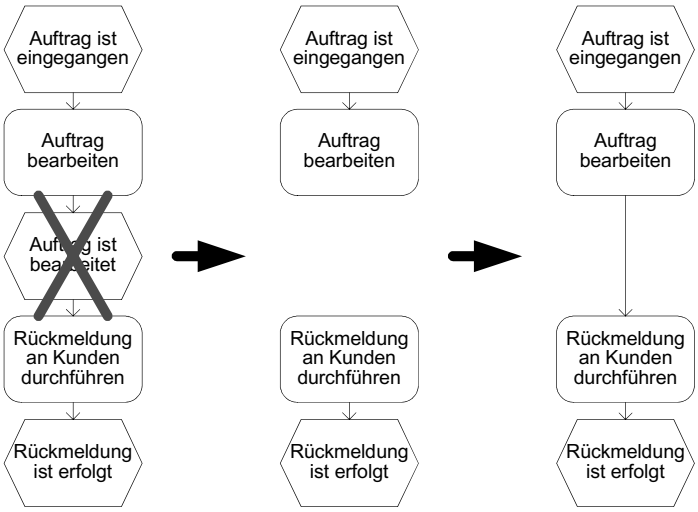

Abbildung 7: Syntaktisch-semantische Inkonsistenzen

- In anderen Fällen kann das entstehende Modell oder die entstehenden Teilmodelle durch Löschung, Änderung oder Hinzufügen von Modellbestandteilen syntaktisch korrekt bleiben, ohne dass eine vollständige Automatisierung der Konfiguration durchgeführt werden kann. Dieser Fall liegt dann vor, wenn das syntaktisch korrekte (Zwischen-)ergebnis die Ableitung weiterer Modellvarianten nahe liegt, zwischen denen eine inhaltlich begründete Entscheidung zu treffen ist. In der Abbildung 8 ist ein Prozessmodell gezeigt, aus dem eine für eine Perspektive nicht mehr relevante Funktion entfernt wurde. Es sind zwei syntaktisch korrekte Teilprozesse entstanden, da diese konsistent mit ihrem Metamodell sind. Da der Kontrollfluss aber nicht mehr vollständig ist, geht ein Teil der Modellsemantik verloren. Im Beispiel ist nicht eindeutig, auf welche Weise der Kontrollfluss wiederherzustellen ist. Eine vollautomatisierte Anpassung des Modells zur Einhaltung der Konsistenz der konfigurierten Prozessmodelle ist an dieser Stelle ohne Zusatzinformationen nicht möglich. Der Unterstützungsmechanismus könnte allerdings zum Zeitpunkt der Modellierung eine mögliche semantische Inkonsistenz erkennen und den Modellierer auffordern, das Gesamtmodell mit Zusatzinformationen zu versorgen, welche während des Vorganges der Konfiguration ausgewertet werden und somit zu einem semantisch korrekten konfigurierten Prozessmodell führen können. Für die in Abbildung 8 dargestellten Varianten 2 und 3 ist eine Zusatzinformation notwendig, die angibt, welches weitere Ereignis zu streichen ist und wie der Kontrollfluss wiederhergestellt wird. Für die Auflösung der Inkonsistenz mit Hilfe der Variante 4 muss der Modellierer das neue zusammengesetzte Ereignis selbst benennen und ihm somit eine Semantik zuweisen.

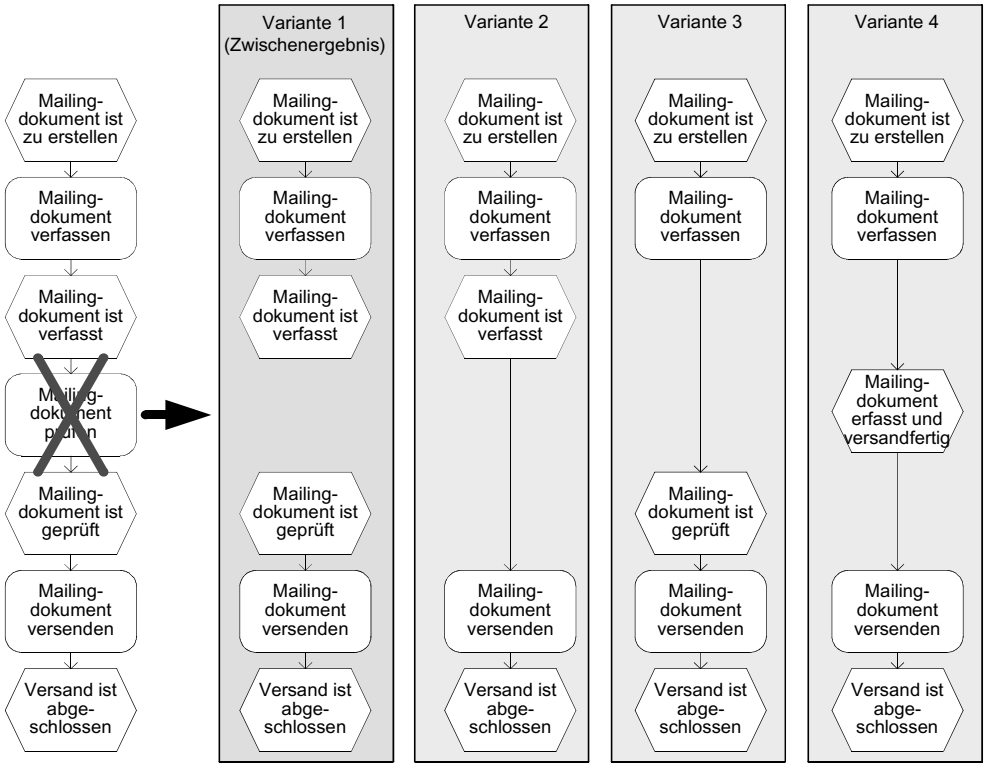

Abbildung 8: Auswahl inhaltlicher Varianten nach Anwendung von Konfigurationsmechanismen

Bei der Implementierung von Konsistenzsicherungsmechanismen ist insbesondere im Falle der rein semantischen Inkonsistenzen darauf zu achten, dass nicht jeglicher Konfigurationsschritt durch Konsistenzchecks unterbrochen wird, um den Modellierungsprozess nicht unnötig zu verzögern. Hier ist eine geeignete Kategorisierung der Fälle vorzunehmen.

5 Beispiel für eine multiperspektivische Konfiguration einer EPK

Ein vereinfachtes, multiperspektivisches Prozessmodell der Rechnungsprüfung, welches sowohl Informationen für Anwendungssystem- als auch Organisationsgestalter enthält, soll den jeweiligen Perspektivenvertretern anforderungsgerecht bereitgestellt werden [Be00]. Einige Prozessfunktionen der Rechnungsprüfung werden manuell ausgeführt, einige laufen vollautomatisiert ab, wiederum andere befinden sich an der Schnittstelle Mensch-Maschine und repräsentieren Benutzereingaben. Es wird davon ausgegangen, dass vollautomatische Abläufe lediglich für Anwendungssystemgestalter, rein manuelle ausschließlich für Organisationsgestalter relevant sind. Durch eine *Elementselektion über Attribute*, die auf dem Attribut „Automatisierungsgrad“ basiert, können jeweils perspektivenspezifische Modelle generiert werden (vgl. Abbildung 9).

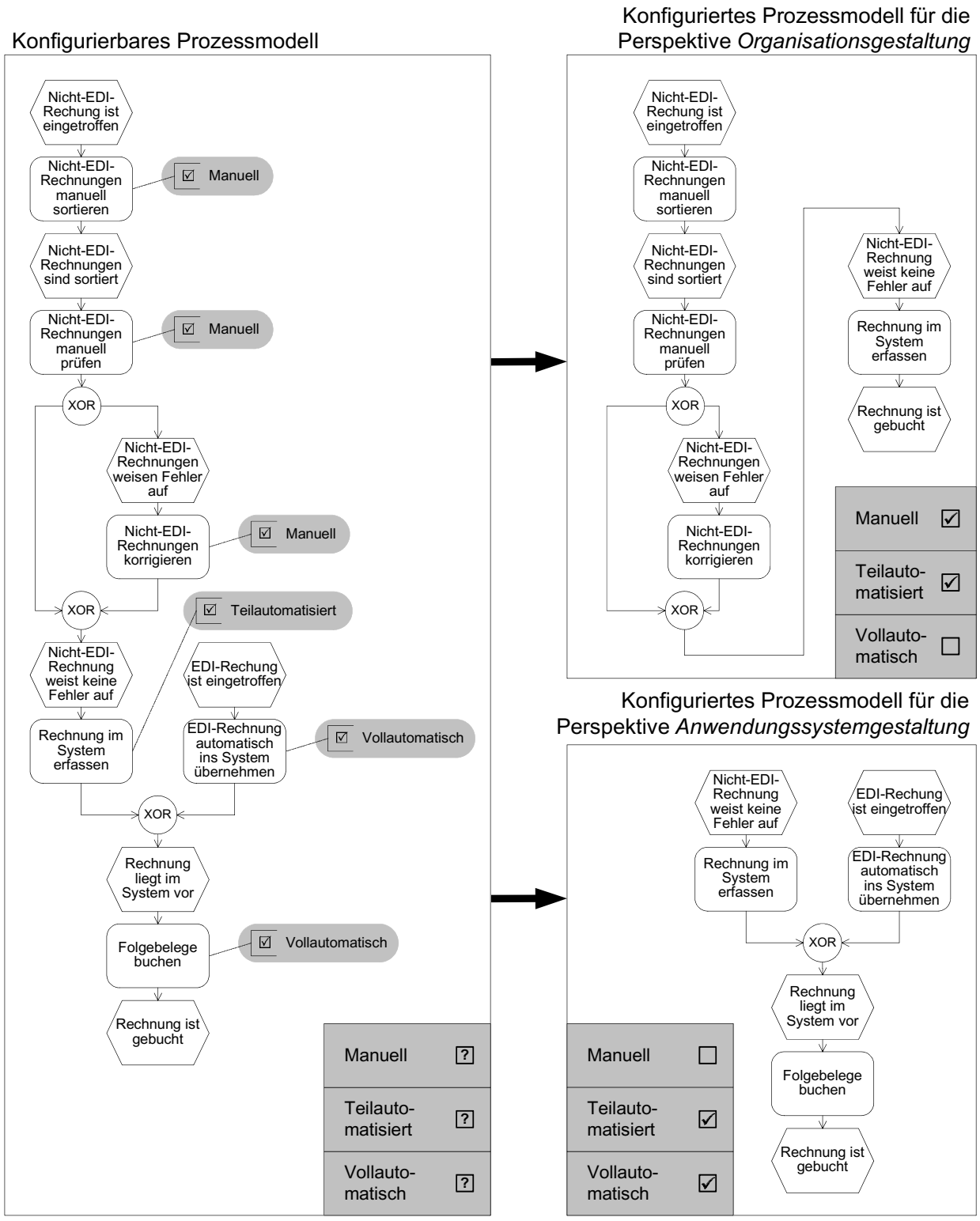

Abbildung 9: Beispiel für die Elementselektion über Attribute

Die notwendigen Veränderungen am Metamodell der EPK und die Generierung des Modells für die Perspektive der Anwendungssystemgestaltung soll am vorliegenden Beispiel detailliert dargestellt werden, um so die Prinzipien der Elementselektion und Konsistenzsicherung aufzuzeigen. Das Prozesselement *Prozessfunktion* erhält im Metamodell der EPK ein Attribut *Automatisierungsgrad* vom Entitytyp *Attribut für Element-*

selektion über Attribute. Weiterhin wird im Metamodell eine Instanz des Entitytyps *Restriktion SQL* mit dem Wert *„WHERE Automatisierungsgrad <> ´manuell´“* angelegt und der Perspektive *Anwendungssystemgestaltung* zugeordnet.

Auf Modellebene werden hierdurch alle Instanzen von Prozessfunktionen ausgeblendet, die dort als *manuell* gekennzeichnet worden sind. Im Beispiel betrifft dies drei Prozessfunktionen. Durch diese Selektion werden weiterhin automatisch alle Instanzen des Relationshiptyps *Vorgänger/Nachfolger* ausgeblendet, in welchen die nicht selektierten Prozessfunktionen die Nachfolger- oder Vorgänger-Rolle einnehmen. Die jeweils vor- oder nachgelagerten Ereignisse und Operatoren der ausgeblendeten Prozessfunktionen stehen nun „alleine“ im Prozessmodell, wodurch die syntaktische Korrektheit nicht mehr gewährleistet ist. Eine Anpassung der Syntax – analog zum zweiten Aufzählungspunkt des Abschnitts 4 – für diese Perspektive, in der direkt aufeinander folgende Ereignisse zugelassen werden und Operatoren direkt auf Ereignisse folgen dürfen, ist hier nicht vorgesehen. Daher sind die entsprechenden Ereignisse und Operatoren ebenfalls aus dem Modell auszublenden. In diesem Fall könnte die Konsistenzsicherung automatisiert durchgeführt werden. Das entsprechend konfigurierte Prozessmodell für die Perspektive der Anwendungssystemgestaltung beinhaltet nunmehr nur noch teilautomatisierte und automatisierte Prozessfunktionen, deren relevante Ereignisse und die notwendige Ablaufsteuerung.

6 Zusammenfassung und Ausblick

Prozessmodellierungsaufgaben können durch die Nutzung von multiperspektivischen EPKs in zweierlei Hinsicht erleichtert werden: Zum Einen werden dem Modellierer Doppelarbeiten durch redundanzfreie Modellierung erspart, von denen auch der Anwender in Form von weitergegebenen Kosten- und Zeitersparnissen profitieren kann. Zum Anderen können den Modellanwendern auf Vorrat Modelle zur Verfügung gestellt werden, die seinen spezifischen Anforderungen in besonderem Maße genügen.

Eine Ausweitung des Perspektivenkonzeptes – nämlich die Konfiguration von EPKs in Abhängigkeit von weiteren ausgewählten Kriterien – ist insbesondere für die Referenzmodellierung interessant. Referenzprozessmodelle besitzen einen Empfehlungscharakter für die Ausgestaltung von Prozessen innerhalb einer Domäne (z. B. Branche oder Wirtschaftszweig). Zwischen den Unternehmen einer Branche können allerdings, in Abhängigkeit von der Ausprägung spezifischer Unternehmensmerkmale der entsprechenden Branche, durchaus Differenzen in den Referenzprozessmodellen identifiziert werden. Mit Hilfe des vorgestellten Konzepts können somit auch unternehmensmerkmalsabhängige Referenzprozessmodelle konfiguriert werden [Be02].

Da das Konzept der Konfiguration durch seine Spezifikation auf Meta-Metamodellebene eine partielle Allgemeingültigkeit erhält, lässt es sich zudem gut auf verschiedene Modellierungstechniken übertragen. Dies gilt insbesondere für solche Techniken, die sich auf Fachkonzeptebene in die einzelnen Sichten der ARIS-Architektur einordnen lassen.

Weiterer Forschungsbedarf besteht in der Übertragung der multiperspektivischen Informationsmodellierung auf ARIS-fremde Modellierungstechniken, insbesondere stark formalisierte, dynamische Techniken wie die der Petri-Netze (vgl. z. B. [Re85; Ba90]) sowie objektorientierten Methoden [BJR98]. Hierdurch kann ein stärkerer Bezug zur Anwendungssystementwicklung hergestellt werden, wodurch das Konzept besonders für Softwarehäuser im Rahmen des Customizing an Bedeutung gewinnt.

Literaturverzeichnis

[Ba90] Baumgarten, B.: Petri-Netze. Grundlagen und Anwendungen. Mannheim 1990.

[Be00] Becker, J.; Holten, R.; Knackstedt, R.; Schütte, R.: Referenz-Informationsmodellierung. In: F. Bodendorf, M. Grauer (Hrsg.): Verbundtagung Wirtschaftsinformatik 2000. Aachen 2000, S. 86-109.

[Be01a] Becker, J.; Knackstedt, R.; Kuropka, D.; Delfmann, P.: Subjektivitätsmanagement für die Referenzmodellierung. Vorgehensmodell und Werkzeugkonzept. In: Proceedings zur Tagung IFM, COMTEC, KnowTech. Dresden, 1.-3. November 2001.

[Be01b] Becker, J.; Knackstedt, R.; Holten, R.; Hansmann, H.; Neumann, S.: Konstruktion von Methodiken: Vorschläge für eine begriffliche Grundlegung und domänenspezifische Anwendungsbeispiele. In: J. Becker, H. L. Grob, S. Klein, H. Kuchen, U. Müller-Funk, G. Vossen (Hrsg.): Arbeitsberichte des Instituts für Wirtschaftsinformatik, Nr. 77. Münster 2001.

[Be02] Becker, J.; Delfmann, P.; Knackstedt, R.; Kuropka, D.: Konfigurative Referenzmodellierung. In: J. Becker, R. Knackstedt (Hrsg.): Wissensmanagement mit Referenzmodellen. Heidelberg 2002, S. 25-144.

[BJR98] Booch, G.; Jacobson, I.; Rumbough, J.: The Unified Modeling Language User Guide. Biston 1998.

[BRS95] Becker, J.; Rosemann, M.; Schütte, R.: Grundsätze ordnungsmäßiger Modellierung. Wirtschaftsinformatik. 37 (1995) 5, S. 435-445.

[BS96] Becker, J; Schütte, R.: Handelsinformationssysteme. Landsberg am Lech 1996.

[Ch76] Chen, P. P.: The Entity-Relationship Model. Toward a Unified View of Data. ACM Transactions on Database-Systems. 1 (1976) 1, S. 9-36.

[DD97] Date, C.; Darwen, H.: A Guide to the SQL Standard. 4. Auflage, Boston 1997.

[Fi92] Finkelstein, A.; Kramer, J.; Nuseibeh, B.; Finkelstein, L.; Goedicke, M.: Viewpoints: a framework for integrating multiple perspectives in system development. International Journal of Software Engineering and Knowledge Engineering. 2 (1992) 1, S. 31-57, 1992.

[Fr94] Frank, U.: Multiperspektivische Unternehmensmodellierung. Theoretischer Hintergrund und Entwurf einer objektorientierten Entwicklungsumgebung. München, Wien 1994.

[Ha94] Hars, A.: Referenzdatenmodelle. Grundlagen effizienter Datenmodellierung. Wiesbaden 1994.

[KNS92] Keller, G.; Nüttgens, M.: Scheer, A.-W.: Semantische Prozeßmodellierung auf der Grundlage „Ereignisgesteuerter Prozeßketten“. In: A.-W. Scheer (Hrsg.): Veröffentlichungen des Instituts für Wirtschaftsinformatik. Heft 89. Saarbrücken 1992.

[NR02] Nüttgens, M.; Rump, F.J.: Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK), in: Desel, J.; Weske, M. (Hrsg.): Promise 2002 - Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen, Proceedings des GI-

Workshops und Fachgruppentreffens (Potsdam, Oktober 2002), LNI Vol. P-21, Bonn 2002, S. 64-77.

[PF96] Poon, W. L.; Finkelstein, A.: Consistency Management for Multiple Perspective Software Development. In: Joint proceedings of the second international software architecture workshop (ISAW-2) and international workshop on multiple perspectives in software development (Viewpoints '96) on SIGSOFT '96. San Francisco 1996, S. 192-196.

[Re85] Reisig, W.: Systementwurf mit Netzen. Berlin et al. 1985.

[Ro96] Rosemann, M.: Komplexitätsmanagement in Prozeßmodellen. Methodenspezifische Gestaltungsempfehlungen für die Informationsmodellierung. Wiesbaden 1996.

[RS99] Rosemann, M.; Schütte, R.: Multiperspektivische Referenzmodellierung. In: J. Becker, M. Rosemann, R. Schütte (Hrsg.): Referenzmodellierung. State-of-the-Art und Entwicklungsperspektiven. Heidelberg 1999, S. 22-44.

[RSD03] Rosemann, M.; Schwegmann, A.; Delfmann, P.: Vorbereitung der Prozessmodellierung. In: J. Becker, M. Kugeler, M. Rosemann (Hrsg.): Prozessmanagement. Ein Leitfaden zur Prozessorientierten Organisationsgestaltung. 4. Auflage, Berlin et al. 2003, S. 47-105.

[Sc99] Schwegmann, A.: Objektorientierte Referenzmodellierung. Theoretische Grundlagen und praktische Anwendung. Wiesbaden 1999.

Ontologische Evaluierung von Ereignisgesteuerten Prozessketten

Peter Fettke, Peter Loos

Johannes Gutenberg-Universität Mainz
Information Systems & Management
Lehrstuhl Wirtschaftsinformatik und Betriebswirtschaftslehre
D-55099 Mainz, Germany
E-Mail: {fettke|loos}@isym.bwl.uni-mainz.de

Abstract: Die ontologische Evaluierung wurde von Wand/Weber Anfang der 1990'er Jahre eingeführt, um Eigenschaften von Modellierungssprachen zu untersuchen. In diesem Beitrag wird der Evaluierungsansatz auf Ereignisgesteuerte Prozessketten (EPK) angewendet. Im Unterschied zu der bereits durchgeführten Untersuchung von Green/Rosemann, die primär die Praktikabilität der ontologischen Evaluierung von Sprachen zur Prozessmodellierung im Allgemeinen verdeutlicht, werden im vorliegenden Beitrag die Konstrukte der EPK aus ontologischer Sicht detailliert analysiert. Ergebnis der Untersuchung sind verschiedene ontologische Defizite von EPK. Darüber hinaus wird gezeigt, dass die von Rosemann bzw. Schütte vorgeschlagenen Erweiterungen der EPK um Prozessobjekttypen bzw. Ereignis-Zustands-Konstrukte aus ontologischer Sicht positiv zu bewerten sind.

1 Ausgangssituation und Problemstellung

Ereignisgesteuerte Prozessketten (EPK) haben als Mittel zur Repräsentation von Prozessmodellen sowohl in der Theorie als auch in der Praxis eine gewisse Verbreitung gefunden. Im Allgemeinen kann davon ausgegangen werden, dass die Qualität einer Modellierungssprache die Qualität der in ihr repräsentierten Sachverhalte beeinflusst. Vor diesem Hintergrund stellt sich die Frage, wie die Qualität von EPK zu beurteilen ist.

Die Qualität einer Modellierungssprache äußert sich in unterschiedlichen Aspekten. Nach Frank [Fr98, S. 13-17] können je nach Aufgabe im Anwendungsgebiet und Rolle im Modellierungsprozess unterschiedliche Qualitätsauffassungen vertreten werden: Exemplarisch mag aus einer Endanwenderrolle die Natürlichkeit der Modellierung dominieren, wohingegen aus Sicht der Systementwicklung die Präzision, Exaktheit und Konsistenz der Modellierungssprache von besonderer Bedeutung sind. Somit erscheint die von Frank geforderte multiperspektivische Evaluierung generell sinnvoll.

Als eine mögliche Perspektive wird in der Literatur die von Wand/Weber entwickelte ontologische Evaluierung vorgeschlagen (siehe Abschnitt 2.2). Dieser Ansatz kann nicht sämtliche Qualitätsaspekte einer Modellierungssprache überprüfen, sondern ausschließlich ausgewählte Aspekte, die aus einer bestimmten Sichtweise wünschenswert erscheinen. Hierbei handelt es sich um die noch näher zu definierenden Kriterien der Vollstän-

digkeit, Redundanz, Übermächtigkeit und Überladung. Diese Kriterien werden inhaltlich durch eine Fundierung in der philosophischen Disziplin der Ontologie entfaltet.

Die Ontologie als eine Teildisziplin der Philosophie beschäftigt sich mit der Ganzheit der Wirklichkeit [Bu77, 5f.]. Sie untersucht "the most pervasive features of reality, such as real existence, change, time, causation, chance, life, mind, and society"[Bu03, S. 201]. So wie bspw. mithilfe der Automaten- oder Netzwerktheorie die Gegenstände „Automat" und „Netzwerk" untersucht werden können, behandelt die Ontologie Gegenstände der Realität und deren Verhalten in einem weitest möglichen Sinne. Ontologie behandelt die allgemeinen Strukturen der Wirklichkeit, die allen Gegenständen der Welt gemeinsam sind. Dagegen ist die Analyse von Begriffen nicht Gegenstand dieser Disziplin, obgleich der Zugang zur Welt sowie ihrer Strukturen sprachlich erfolgt und die Ontologie sprachlich formuliert wird. Folglich wird in diesem Beitrag das Wort Ontologie weder im Sinne eines Begriffsverzeichnisses [FS01, S. 722f.] noch als eine formalisierte Spezifikation von Konzeptualisierungen [Gr95; Ze01, S. 186] verstanden. Das dieser Arbeit zugrundeliegende Verständnis ermöglicht es, die Ontologie als eine theoretische Grundlage der Informationsmodellierung zu verstehen, wenn von der Annahme ausgegangen wird, dass Informationsmodelle die Wirklichkeit repräsentieren [WW95, S. 208; Wa95, S. 287; WW02, S. 2]. Ebenso wird in diesem Beitrag den von Bunge postulierten ontologischen und epistemologischen Prinzipien gefolgt, die einen wissenschaftlichen Realismus konstituieren [Bu77, S. 16f.; Bu83, S. 264-271]: 1. Die Welt existiert unabhängig von der Erfahrung eines Beobachters (ontologische Prämisse). 2. Die Wahrnehmung der Welt ist einerseits beschränkt und trügerisch. Andererseits kann keine endgültige Wahrheit über die Welt gewonnen werden. Vielmehr ist menschliche Erkenntnis grundsätzlich fehlbar (epistemologische Prämisse).

Das Ziel dieser Untersuchung ist die ontologische Evaluierung von EPK. Der Beitrag ist wie folgt aufgebaut: Nach diesem einleitenden Abschnitt wird im nächsten Abschnitt der theoretische Hintergrund der Untersuchung dargelegt. Abschnitt drei beschreibt die Ergebnisse der ontologischen Evaluierung. Der Beitrag schließt im vierten Abschnitt mit Schlussfolgerungen und Grenzen der Untersuchung sowie einem Ausblick auf weitere Fragestellungen.

2 Theoretischer Hintergrund

2.1 Ereignisgesteuerte Prozessketten

EPK wurden Anfang der 1990'er Jahre am Institut für Wirtschaftsinformatik der Universität des Saarlandes entwickelt [Ke92] und haben sowohl national als auch international eine gewisse Verbreitung gewonnen [Da00]. Zweck der EPK ist die Repräsentation von Prozessmodellen. In ihrer Grundform besteht eine EPK aus Ereignistypen und Funktionstypen, die mit Hilfe von Kontrollflusskanten und logischen Verknüpfungsoperatoren – sogenannten Konnektoren – in eine zeitlich-sachlogische Abfolge angeordnet werden. Für eine detaillierte Darstellung siehe [Ke92; Sc97]. Zum EPK-Konzept existieren verschiedene Erweiterungen, von denen für die weitere Untersuchung zwei Ansätze von Relevanz sind: erstens die Erweiterung um Prozessobjekttypen von Rosemann [Ro95b,

76-84] und zweitens die Erweiterung um Ereignis-Zustand-Konstrukte von Schütte [Sc98, 103-105]. Für eine detailliertere Darstellung der EPK-Konstrukte sei auf die angegebene Literatur verwiesen.

2.2 Ontologische Evaluierung von Modellierungssprachen

Bisher hat sich keine universelle und allgemein akzeptierte Ontologie in der Philosophie herausgebildet. Vielmehr existieren unterschiedliche Positionen [RK98]. Die vorliegende Untersuchung greift auf die von Bunge eingeführte Ontologie zurück [Bu77; Bu79], die von Wand und Weber für den Kontext der Modellierung adaptiert worden ist (Bunge-Wand-Weber-Modell (BWW-Modell) [WW89a; WW95; We97a]). Zur Verbesserung der Klarheit und Verbreitung des BWW-Modells haben Rosemann/Green ein Meta-Modell entwickelt [RG02]. Das BWW-Modell kann an dieser Stelle nicht rekapituliert werden, hierfür sei auf die genannte Literatur verwiesen. Eine zusammenfassende Erläuterung der wesentlichen Konstrukte des BWW-Modells findet sich im Anhang des Beitrages.

Zielstellung der ontologischen Evaluierung ist es, zwischen den Konstrukten des BWW-Modells und den Konstrukten der untersuchten Modellierungssprache Zusammenhänge herzustellen, um beurteilen zu können, ob mit Hilfe der Modellierungssprache sämtliche Konstrukte des BWW-Modells eindeutig repräsentiert werden können. Der Kern des Verfahrens ist die Erstellung einer Transformationsvorschrift. Die Transformationsvorschrift besteht aus zwei mathematischen Funktionen, die folgendermaßen zu konstruieren sind: Einerseits werden die Konstrukte der Ontologie auf die Konstrukte der Modellierungssprache abgebildet. Diese Abbildung wird Repräsentationsabbildung genannt. Andererseits werden mit der sogenannten Interpretationsabbildung die Konstrukte der Modellierungssprache auf die Konstrukte der Ontologie abgebildet. Grundsätzlich lassen sich auf Basis beider Abbildungen vier ontologische Defizite unterscheiden:

- Unvollständigkeit: Kann jedes Konstrukt der Ontologie durch ein Konstrukt der Modellierungssprache abgebildet werden?
- Redundanz: Wird ein Konstrukt der Ontologie durch genau ein oder durch mehrere Konstrukte der Modellierungssprache abgebildet?
- Übermächtigkeit: Gibt es Konstrukte der Modellierungssprache, die nicht durch ein Konstrukt der Ontologie abgebildet werden können?
- Überladung: Kann ein Konstrukt der Modellierungssprache mehrere Konstrukte der Ontologie repräsentieren?

2.3 Stand der Forschung

Bisher wurden verschiedene Modellierungssprachen einer ontologischen Evaluierung unterzogen [WW89a; WW90; WZ91; WW93; WW95; Ro95a; WZ96; We96; Gr96; Wa99; GR00a; GR00b; OH01; EW01b; EW01a; GR01; Bo01; OH02]. Die vorliegenden Arbeiten verdeutlichen einerseits, dass der Ansatz in der Literatur eine gewisse Verbreitung gefunden hat. Andererseits können die vorliegenden Untersuchungen als ein Indiz gewertet werden, dass der Ansatz eine bestimmte Allgemeingültigkeit besitzt.

Im Folgenden wird dargelegt wie die hier vorgenommene Untersuchung auf den Untersuchungen von Green/Rosemann [GR00b; GR00a] aufbaut:

- Green/Rosemann bezwecken mit ihrer Untersuchung primär nachzuweisen, dass das BWW-Modell zur Untersuchung von Prozessmodellen prinzipiell geeignet ist („The intention of this research is to demonstrate the potential usefulness of the BWW theory to evaluate and improve integrated process modeling grammars"[GR00b, S. 83]). Dahingegen ist das erklärte Ziel der vorliegenden Arbeit die ontologische Analyse von EPK.
- Während Green/Rosemann primär die ontologische Vollständigkeit mit Hilfe der Repräsentationsabbildung untersuchen („the principal focus for this work is the situation of ontological incompleteness"[GR00b, S. 82]), ist der Fokus der vorliegenden Arbeit die Analyse der Interpretationsabbildung.
- In der hier vorgelegten Untersuchung werden die einzelnen Konstrukte detaillierter diskutiert. Ebenso werden nicht nur die „üblichen" Konstrukte von EPK untersucht, sondern darüber hinaus auch spezielle Erweiterungen wie Prozessobjekttypen und Ereignis-Zustand-Konstrukte.

Ferner soll die Intention der vorliegenden Arbeit von Ansätzen abgegrenzt werden, die auf eine Formalisierung des EPK-Konzepts abzielen [Aa99; NR02; Ru98; Ro97; Ut97]. Diese Arbeiten verfolgen das Ziel, Syntax und Semantik von EPK-Konstrukten wohldefiniert zu (re-)konstruieren. Hierzu werden mathematische Konstrukte wie Mengen, Graphen, Zustandsübergangsrelation etc. verwendet. Indes beantwortet selbst eine vollständige Formalisierung der Syntax und Semantik von EPK nicht die Frage, wie EPK-Konstrukte ontologisch zu interpretieren sind. Vielmehr beschäftigen sich die Formalisierungsansätze ausschließlich mit formalen Konstrukten und deren Beziehungen. Dahingegen untersucht die vorliegende Arbeit, wie die EPK-Konstrukte ontologisch, also im Hinblick auf empirisch beobachtbare Phänomene der Welt, interpretiert werden können.

3 Ergebnisse

In den folgenden Unterabschnitten werden ontologische Interpretationsmöglichkeiten jeweils für ein einzelnes Konstrukt der EPK diskutiert.

3.1 Ereignistyp

Ein Ereignistyp repräsentiert eine Klasse von Ereignissen. Ereignisse wiederum werden als passive Komponenten einer EPK verstanden und beschreiben eingetretene Zustände, die in Beziehung zu bestimmten Attributsausprägungen in Informationsobjekten in betrieblichen Systemen stehen können.

Zum Verständnis der ontologischen Interpretation des Konstrukts werden zunächst einige Vorbemerkungen angebracht. Hierbei wird insbesondere die Unterscheidung von Ereignissen und Ereignistypen thematisiert.

Keller/Nüttgens/Scheer führen Ereignistypen folgendermaßen ein: „Durch Abstraktion der realen Ausprägungen erhält man [...] ‚Ereignistypen' [...]. Ein Ereignistyp ist eine eindeutig benannte Sammlung von Ereignissen, die aufgrund des Eingetretenseins von Ausprägungen derselben Attribute einer Klasse zugeordnet werden."[Ke92, S. 11] Die Autoren führen zwei Beispiele für Ereignisse und mögliche Ereignistypen an:

- Das Ereignis et'1 „Bedarf von Material 4711 in einer Menge von 500 Stück ist aufgetreten" ist ein Ereignis des Ereignistyps ET' „Bedarf ist eingetreten".
- Das Ereignis et''2 „Bestellanforderungsposition für Material 4711 erstellt" ist ein Ereignis des Ereignistyps ET'' „Bestellanforderungsposition ist erstellt".

Aus dieser Darstellung kann entnommen werden, dass ein Ereignistyp eine Menge bzw. eine Klasse von Ereignissen darstellt, die hinsichtlich bestimmter Merkmale gebildet werden.

An dieser Stelle soll auf eine Eigenart aufmerksam gemacht werden. Zum oben eingeführten Ereignistyp ET1 sind grundsätzlich beliebig viele weitere Ereignisse denkbar. Weitere Ereignisse können sein:

- Das Material, für das ein Bedarf auftritt, kann bei sonst gleichbleibenden Bedingungen variieren. Beispiel: Ereignis et'2 „Bedarf von Material 4712 in einer Menge von 500 Stück ist aufgetreten" etc.
- Die Bedarfsmenge des Ereignisses kann variieren. Beispiel: Ereignis et'3 „Bedarf von Material 4711 in einer Menge von 100 Stück ist aufgetreten" etc.

Die Beispiele zeigen, dass potenziell unendlich viele konkrete Ereignisse durch den Ereignistyp ET' definiert werden. Gleichsam ist durch ET' keineswegs das Abstraktionsprinzip für konkrete Ereignisse et'1, et'2 etc. näher festgelegt. Vielmehr ist die Art der Abstraktion prinzipiell unbestimmt. Dieser Sachverhalt soll in Analogie zur Datenmodellierung verdeutlicht werden. Innerhalb der Datenmodellierung repräsentieren Entitytypen jeweils eine Klasse von Entities. Hierbei wird über die Menge der Attribute, die einem Entitytyp zugeordnet werden, die Klasse möglicher Ausprägungen bestimmt. Die Abstraktionsbeziehung wird durch das Bilden einer Klasse definiert: Ein Entitytyp beschreibt alle Entities, die durch eine bestimmte Attributmenge definiert werden [Vo94, S. 54]. Dieser konzeptionelle Unterschied kann durch die Begriffe der Extension und Intension eines Prädikators näher umschrieben werden. Während Entitytypen in der Datenmodellierung extensional definiert werden, hat die Definition von Ereignissen im EPK-Konzept einen intensionalen Charakter.

Ferner ist auf eine weitere Eigenart hinzuweisen. Die oben eingeführten Beispiele könnten zu der Vermutung verleiten, dass die Ereignisse et'1, et'2, et'3 etc. in dem Sinne klar definiert sind, dass sie eindeutig ein Ereignis eines realen Sachverhalts repräsentieren (singuläres Ereignis). Diese Vermutung kann intuitiv damit begründet werden, dass es sich um „konkrete" Ereignisse eines Sachverhaltes handelt und keineswegs um Ereignisklassen. Einer solchen Argumentation muss allerdings nicht zwingend gefolgt werden. Vielmehr ist es denkbar, dass die gebildeten Ereignisse erneut als Ereignisklassen verstanden werden. Diese Interpretation kann an dem im Beispiel genannten Ereignis et'1 verdeutlicht werden, das oben als „Bedarf von Material 4711 in einer Menge von 500

Stück ist aufgetreten“ definiert wurde. Dieses Ereignis kann wiederum als Klasse von Ereignissen verstanden werden:

- Konkretere Ereignisse können durch Konkretisierung der Bedarfsorte gebildet werden. Beispiele: „Bedarf von Material 4711 in einer Menge von 500 in Werk A“ oder „... in Werk B“ oder „... bei Kunden A“ etc.
- Konkretere Ereignisse können durch Konkretisierung des Bedarfszeitpunktes gebildet werden. Beispiele: „Bedarf von Material 4711 in einer Menge von 500 in Planungsperiode 1“ oder „... in Planungsperiode 2“ etc.

Diese Vorbemerkungen zeigen zweierlei. Einerseits wird die Art der Abstraktion von Ereignissen zu Ereignistypen innerhalb einer EPK nicht repräsentiert und bleibt damit unbestimmt. Andererseits ist die Angabe eines konkreten Ereignisses zu einem Ereignistyp durchaus interpretationsdürftig. Die Autoren räumen zwar ein, dass die Bildung von Ereignistypen insofern keine Besonderheit darstellt, dass es sich um einen bekannten und üblichen Vorgang der Typenbildung handelt. Allerdings besteht die Besonderheit im EPK-Konzept gerade darin, dass zum einen der Umfang der Grundgesamtheit von Ereignissen nicht definiert ist. Zum anderen ist unklar, wie einzelne Elemente der Grundgesamt der Ereignisse zu interpretieren sind.

Nach diesen Vorbemerkungen, mit denen die Bedeutung von Ereignistypen allgemein aufgezeigt wird, folgt nun eine ontologische Interpretation. Die Beschreibung von dynamischen Aspekten erfolgt im BWW-Modell u. a. über BWW-Zustände und BWW-Ereignisse. Ein BWW-Ereignis besteht aus einem Zustandspaar, das den Vorzustand und den Folgezustand eines BWW-Gegenstandes beschreibt. Gemäß des BWW-Modells ist ein BWW-Zustand ein Vektor aller Eigenschaftsfunktionen eines BWW-Gegenstandes. Über eine BWW-Transformation werden sämtliche BWW-Ereignisse beschrieben.

EPK-Ereignisse können als BWW-Zustände verstanden werden. Allerdings wird ein BWW-Zustand nicht vollständig repräsentiert, sondern es wird jeweils nur ein Teil der gesamten BWW-Eigenschaften des BWW-Gegenstandes abgebildet. Ferner stellt sich bei dieser Interpretation die Frage, welcher BWW-Gegenstand durch ein EPK-Ereignistyp repräsentiert wird. Wie oben ausgeführt wurde repräsentiert ein EPK-Ereignistyp eine Klasse von EPK-Ereignissen, die nicht zwingend eindeutig spezifizierte Ereignisse der Welt repräsentieren, sondern sich erneut auf eine Klasse von Ereignissen beziehen können. In der Literatur wird beschrieben, dass EPK-Ereignisse als bestimmte Attributsausprägungen von Informationsobjekten verstanden werden können. Dieser Zusammenhang kann eine Interpretationshilfe für die ontologische Bedeutung eines Ereignisses geben. Indes wird von dieser Möglichkeit praktisch selten gebraucht gemacht wie bspw. die Referenzmodelle von Scheer [Sc97] und Becker/Schütte [BS96] zeigen. Folglich wird häufig innerhalb der EPK nicht explizit der Gegenstand repräsentiert, auf den sich das Ereignis bezieht. Für die Interpretation einer EPK ist es notwendig zu klären, welcher BWW-Gegenstand durch ein Ereignis beschrieben wird. Beispielsweise kann das Ereignis et’1 in einer bestimmten Organisation zu einem bestimmten Zeitpunkt auftreten. Diese ontologische Übermächtigkeit kann als ein Abstraktionsprinzip verstanden werden: Ereignisse können unabhängig von einem bestimmten Gegenstand beschrieben werden. Indes ist bei der ontologischen Interpretation des Ereignisses

zu beachten, dass seitens des Modellverwenders eine Konkretisierung der Bedeutung vorgenommen werden muss.

Andererseits können EPK-Ereignisse als BWW-Ereignisse verstanden werden, wobei sich allerdings gewisse Bedeutungsunterschiede ergeben. Ein BWW-Ereignis beschreibt zwei Zustände, wobei der erste Zustand den Systemzustand vor dem Eintritt des Ereignisses und das zweite den Zustand nach dem Eintreten des Ereignisses festhält. Demnach beschreibt ein BWW-Ereignis gerade den Wechsel von einem Zustand in einen anderen. Dahingegen beschreiben EPK-Ereignisse nicht Zustandswechsel eines Systems, sondern das Vorliegen eines ganz bestimmten Zustandes. Andererseits kann argumentiert werden, dass ein EPK-Ereignis gerade von dem Vor- und Folgezustand eines BWW-Ereignisses abstrahiert und nur einen Namen bzw. eine Bezeichnung für ein bestimmtes BWW-Ereignis darstellt. Ferner können bei dieser Interpretation einem EPK-Ereignis keine bestimmten Ausprägungen von Eigenschaften eines BWW-Gegenstandes zugeordnet werden. Vielmehr wäre es dann notwendig, zu jedem Ereignis den Vor- und den Nachzustand separat zu definieren.

3.2 Funktionstyp

Analog zu Ereignistypen repräsentieren Funktionstypen eine Klasse von Funktionen, wobei eine Funktion verstanden wird als die „Durchführung eines betrieblichen Vorgangs, der zur Erfüllung eines Unternehmensziels beiträgt“[Ke92, S. 10]. Funktionen werden als aktive Komponenten umschrieben. Während in der betriebswirtschaftlichen, insbesondere organisationstheoretischen Literatur unter einem betrieblichen Vorgang i. d. R. eine menschlich durchgeführte Handlung verstanden wird, abstrahiert die Wirtschaftsinformatik meist von dem Automatisierungsgrad und subsumiert unter Funktionen sowohl manuelle auch (voll-)automatisierte Funktionen [Es93, S. 552]. Dieser Auffassung wird hier gefolgt.

In Anlehnung an Keller/Nüttgens/Scheer sei exemplarisch der Funktionstyp FT’ „Bestellanforderung erstellen“ und die Funktion ft’1 „Bestellanforderung für Material 4711 erstellen“ als zugehörige konkrete Funktion genannt [Ke92, S. 12]. Analog zur Diskussion von Ereignistypen ist erneut festzuhalten, dass einerseits die Abstraktionsbeziehung zwischen Funktion und Funktionstyp unbestimmt ist. Andererseits ist unklar, ob sich die genannte Funktion ft’1 auf eine konkrete, einzigartige Handlung in einer spezifischen Unternehmenssituation bezieht. Repräsentiert diese Funktion bspw. die Handlung der Erstellung einer Bestellanforderung für Material 4711 von Sachbearbeiter Herrn Müller am 2003-06-17 um 17.15? Oder repräsentiert die Funktion eine weitere Klasse von Handlungen, die so gebildet wird, dass alle Erstellungen für Bestellanforderungen für Material 4711 in einer Klasse zusammengefasst werden?

Zur ontologischen Interpretation wird zunächst davon ausgegangen, dass der Funktionstyp voll-automatisiert ausgeführt werden kann. In diesem Falle führt eine Funktion dazu, dass ein Informationssystem von einem Startzustand in einen Folgezustand überführt wird, wobei die Art der Überführung ausschließlich von Systemzuständen und nicht etwa von Benutzeraktionen determiniert wird. Folglich kann ein Funktionstyp als eine BWW-Transformation beschrieben werden, welche eine Abbildung von Systemzustän-

den auf weitere Systemzustände definiert. Es gilt zu beachten, dass die BWW-Transformation nur dann vollständig definiert ist, wenn aus einem Funktionstyp hervorgeht, auf welche Art und Weise die Transformation durchzuführen ist. Derartige Festlegungen sind bei EPK durch natürlichsprachliche Ergänzungen vorzunehmen.

Ebenso ist es denkbar, einen Funktionstyp als ein BWW-Gegenstand zu verstehen. Diese Interpretation betrachtet den Funktionstyp als ein Subsystem. Der Funktionstyp wird hierbei als eine Blackbox interpretiert, von der ausschließlich bestimmte Eingabe- und Ausgabe-Systemzustände bekannt sind, ohne den genauen Aufbau des Subsystems zu kennen. Diese auf den ersten Blick abwegige Interpretation wird deutlich, wenn berücksichtigt wird, dass Funktionstypen durch weitere, sogenannte Funktionsverfeinerungen präzisiert werden können. Derartige verfeinerte EPK repräsentieren den inneren Aufbau des Subsystems, das durch eine EPK-Funktion repräsentiert wird. In dieser Interpretation wird zwischen Funktionstypen und Funktionsverfeinerungen kein prinzipieller Bedeutungsunterschied gesehen. Vielmehr betont das Konstrukt Funktionsverfeinerung, dass eine Funktion nicht ausschließlich als Blackbox, sondern an anderer Stelle als Whitebox betrachtet wird.

Die obige Darstellung hat vollständig automatisierte EPK-Funktionen unterstellt. Es stellt sich die Frage, welche Interpretationsänderungen vorzunehmen sind, wenn von dieser Annahme abgesehen wird. Die Aufhebung dieser Annahme hat zur Konsequenz, dass EPK-Funktionen nicht nur von Systemzuständen abhängig sind, sondern zusätzlich ebenso vom menschlichen Verhalten. Grundsätzlich kann hierbei die oben vorgestellte Interpretation ebenso angewendet werden. Allerdings ist festzustellen, dass die Abhängigkeiten zwischen dem menschlichen Akteur als ein BWW-Gegenstand und der EPK-Funktion nicht explizit in einer EPK repräsentiert werden.

Abschließend sei auf die ontologische Interpretation von Funktionstypen von Green/Rosemann eingegangen, demnach Funktionstypen BWW-Eigenschaften von BWW-Gegenständen repräsentieren [GR00b, S. 82]. Diese Beziehung wird von den Autoren mit dem Argument begründet, dass Funktionen als Transformationen von betriebswirtschaftlich relevanten Objekten verstanden werden können und somit Eigenschaften von Gegenständen darstellen. Diese Erläuterung erinnert an den informatischen Objektbegriff, demnach Funktionen im Sinne von Methoden neben Attributen als Eigenschaften von Objekten verstanden werden. Hier kann dieser informatisch geprägten Interpretation nicht gefolgt werden: Die Eigenschaften eines BWW-Gegenstandes werden von Funktionen vorausgesetzt bzw. verändert. Funktionen repräsentieren allerdings keine Eigenschaften eines Gegenstands, sie haben eine andere Intension. Vielmehr beschreiben sie den Übergang von einem Startzustand in einen Folgezustand.

3.3 Kontrollflusskanten und Konnektoren

Mit Hilfe von Kontrollflusskanten wird beschrieben, welche Ereignistypen und Funktionstypen in welcher Reihenfolge durchzuführen sind. Ergänzend dazu können Konnektoren verwendet werden, die es erlauben, Verknüpfungen zwischen Ereignistypen und Funktionstypen zu definieren. Kontrollflusskanten und Konnektoren werden hier ontologisch gemeinsam untersucht, da sich bei der Darstellung herausstellen wird, dass diesen

aus ontologischer Sicht eine ähnliche Bedeutung zukommt. Im Folgenden schließt das Wort Kontrollfluss die Verwendung von Kontrollflusskanten und Konnektoren ein.

Zunächst sei der Fall unterstellt, dass zwischen den Ereignis- und Funktionstypen der EPK kein Kontrollfluss definiert sei, die EPK also ausschließlich aus einer Menge nicht weiter strukturierter Ereignis- und Funktionstypen besteht. Eine solche EPK kann aus verschiedenen Ansätzen hervorgehen:

- Bei der Konstruktion der EPK werden ausschließlich relevante Ereignis- und Funktionstypen spezifiziert, ohne auf einen eventuellen Kontrollfluss einzugehen.
- Aus einer bestehenden EPK wird der vorhandene Kontrollfluss bewusst ausgeblendet.
- In einem Sachverhalt bestehen tatsächlich keine Abhängigkeiten im Ablauf der Ereignisse oder Funktionen.

Aus einer derartig konstruierten EPK-Repräsentation werden folgende Aspekte ersichtlich, die hier von Bedeutung sind:

- Ein betriebliches System verweilt in bestimmten Zuständen – falls EPK-Ereignistypen als BWW-Zustände interpretiert werden.
- In einem betrieblichen System treten bestimmte Ereignisse auf – falls EPK-Ereignistypen als BWW-Ereignisse interpretiert werden.
- Ein betriebliches System kann bestimmte Zustandswechsel erfahren – wenn EPK-Funktionstypen als BWW-Transformation interpretiert werden.

Indes sind in einer EPK, die keine Kontrollflussinformationen enthält, keine Abhängigkeiten zwischen Ereignis- und Funktionstypen ersichtlich. Vielmehr sind nur mögliche BWW-Zustände, BWW-Ereignisse und BWW-Transformationen ersichtlich. Zur Beschreibung, dass das betriebliche System nicht beliebig von einem Zustand in einen anderen Wechseln kann, ist es notwendig, den Kontrollfluss innerhalb der EPK zu definieren. Aus ontologischer Sicht interpretiert, bedeutet dies, dass mit Hilfe des Kontrollflusses die Menge möglicher BWW-Ereignisse eingeschränkt wird. Folglich kommt dem Kontrollfluss die Funktion der Formulierung von Restriktionen bei der Definition von Zustandswechseln eines Systems zu. Demnach definieren die Konstrukte zur Formulierung des Kontrollflusses den BWW-gesetzmäßigen Ereignisraum.

3.4 Erweiterungen

3.4.1 Prozessobjekttyp nach Rosemann

Rosemann unterbreitet den Vorschlag, EPK um ein Konstrukt „Prozessobjekttyp“[1] zu erweitern [Ro95b, 76-84]. Die Motivation hierfür ist, dass nach Rosemann ein Prozess die zeitlich-sachlogische Abfolge von Funktionen an einem prägendem Objekttyp dar-

[1] Im Original spricht Rosemann unscharf von einem Prozessobjekt. Bereits Schütte weist zurecht darauf hin, dass von Prozessobjektypen zu sprechen ist [Sc98, S. 100, Fußnote 271].

stellt, wobei dieser Objekttyp den Prozess konstituiert. Diese definitorische Grundlage sollte folglich auch in der EPK dokumentiert werden.

Ein Ergebnis der obigen Untersuchung ist, dass die ontologische Interpretation einer EPK voraussetzt, dass ihre Funktionen und Ereignisse auf einen BWW-Gegenstand zu beziehen sind. Ein Objekttyp kann als eine BWW-Klasse von BWW-Gegenständen interpretiert werden. Durch die Angabe eines Objekttyps wird somit die mögliche Generizität einer EPK in dem Sinne eingeschränkt, dass die EPK nicht mehr auf beliebige BWW-Gegenstände übertragen werden kann, sondern nur auf die Klasse von BWW-Gegenständen, die durch den Objekttyp repräsentiert wird. Mit anderen Worten: Einerseits wird die Interpretationsmöglichkeit eines Modells eingeschränkt, andererseits wird eine Hilfe für den Interpretationskontext gegeben.

Es stellt sich weiterhin die Frage, ob jeder Prozessobjekttyp grundsätzlich eine Menge von BWW-Gegenständen repräsentiert. Mögliche Prozessobjekttypen sind bspw. „Auftrag“, „Rechnung“, „Produkt“, „Kunde“ etc. Die Frage kann nicht grundsätzlich, sondern muss differenzierter beantwortet werden. Prinzipiell sind BWW-Gegenstände nur materielle Gegenstände. Insofern handelt es sich bei den genannten Prozessobjekttypen um BWW-Gegenstände, wenn diese materiell sind. Diese Annahme ist bspw. dann erfüllt, wenn Aufträge bzw. Rechnungen im Sinne von Papierbelegen verstanden werden.

Anders ist die Situation, falls bspw. unter einem Auftrag nicht der konkrete Papierbeleg, sondern eine abstrakte Informationsstruktur bestehend aus Auftragsdatum, Positionen, Summe etc. verstanden wird. In diesem Fall handelt es sich um BWW-Eigenschaften von BWW-Gegenständen. Beispielsweise kann in dieser Sichtweise ein Auftrag als ein BWW-Ereignis verstanden werden, das genau den Augenblick repräsentiert, an dem ein Kunde dem Unternehmen den Auftrag erteilt hat [EW01a, S. 357].

3.4.2 Ereignis-Zustand-Konstrukt nach Schütte

Schütte unterbreitet den Vorschlag, Ereignisse in der Art zu modifizieren, dass Funktionstypen mittels eines Konstrukts verbunden werden, das sowohl einen Zustandstyp als auch einen Ereignistyp repräsentiert (hier als Ereignis-Zustand-Konstrukt bezeichnet) [Sc98, 103-105]. Als Motivation hierfür wird von Schütte angeführt, dass im EPK-Konzept nicht zwischen Zuständen und Ereignissen unterschieden wird.

Aus ontologischer Sicht ist diese Erweiterung wie folgt zu interpretieren. Ein Ergebnis der oben vorgestellten ontologischen Untersuchung ist, dass EPK-Ereignisse sowohl als BWW-Zustände als auch als BWW-Ereignisse zu interpretieren sind. Die vorgeschlagene Erweiterung kann demnach so interpretiert werden, dass die ontologische Überladung des ursprünglichen EPK-Konstruktes „Ereignis“ vermieden wird. Durch die Trennung des Konstruktes in einen Teil, der einen BWW-Zustand repräsentiert, und einen anderen Teil, der als BWW-Ereignis zu verstehen ist, kann das ontologische Defizit dieses Modellierungskonstruktes reduziert werden. Aus ontologischer Sicht verringert diese Maßnahme den Grad der ontologischen Überladung von EPK.

Indes nutzt die vorgeschlagene Erweiterung die sich aus ihr ergebenen Potenziale nur bedingt aus, solange dem repräsentierten Ereignis und Zustand ausschließlich eine Bezeichnung bzw. ein Namen gegeben wird. Indem die Beziehung eines Zustandes bzw. eines Ereignisses zu den BWW-Eigenschaften geklärt wird, könnte die Bedeutung dieser Konstrukte im vollen Umfang festgelegt werden.

3.5 Zusammenfassung

Tabelle 1 gibt eine zusammenfassende Übersicht über die Untersuchungsergebnisse. Die Vorspalte und Kopfzeile der Tabelle bezeichnen die Konstrukte des BWW-Modells und der EPK. Eine Markierung in einem Tabellenelement beschreibt, dass die Bedeutung des in der Kopfzeile genannten EPK-Konstruktes mit der Bedeutung des in der Vorspalte genannten BWW-Konstruktes verwandt ist.

EPK-Konstrukt / BWW-Konstrukt	Ereignistyp	Funktionstyp	Kontrollfluss	AND-Operator	UND-Operator	XOR-Operator	Prozessobjekt	Ereignis-Zustand-Konstrukt (Ereignis)	Ereignis-Zustand-Konstrukt (Zustand)
Gegenstand		●					●		
Eigenschaft	●						●	●	●
Zustand	●								●
Ereignis	●							●	
Transformation		●							
Subsystem		●							
Gesetzmäßiger Ereignisraum			●	●	●	●			

Legende: ● Konstrukte mit ähnlicher Bedeutung

Tabelle 1: Übersicht über Untersuchungsergebnisse

4 Schlussfolgerungen, Einschränkungen und Ausblick

In der Untersuchung wurden verschiedene Möglichkeiten der ontologischen Interpretation von EPK-Konstrukten aufgezeigt. Es zeigt sich, dass die EPK-Konstrukte ontologisch unklar und mehrdeutig interpretiert werden können. Ebenso wurde gezeigt, dass vorgeschlagene Erweiterungen der EPK dazu beitragen können, ontologische Defizite dieser Sprache zu mildern.

Gleichsam räumen die Autoren ein, dass bei der Interpretation der EPK-Konstrukte Spielräume vorhanden sind, durch welche die Untersuchung subjektiven Einflüssen unterliegt. Allerdings wurde versucht, mögliche Interpretationsspielräume explizit aus-

zuführen, um die Untersuchung nachvollziehbar zu machen. Ebenso ist darauf hinzuweisen, dass derartige Unschärfen nicht nur auf die Evaluierungsmethode, sondern auch auf EPK als Evaluierungsgegenstand zurückzuführen sind. Folglich ist eine gewisse subjektive Färbung auch bei einer Anwendung anderer Evaluierungsmethoden nicht zu vermeiden. Andererseits ist die Herausarbeitung möglicher Interpretationsspielräume u. a. eine Zielstellung der ontologischen Evaluierung: Die Untersuchung bezweckt, derartige Interpretationsspielräume zu verdeutlichen, um Sprachnutzer für Mehrdeutigkeiten der EPK-Konstrukte zu sensibilisieren. Ebenso können Interpretationsspielräume als besondere Mächtigkeit von EPK verstanden werden. Diese Interpretation ergibt sich dann, wenn ontologisch mehrdeutige Konzepte als „generische" Konzepte verstanden werden. Indes sind die Autoren der Ansicht, dass derartige Interpretationsspielräume expliziert werden sollten, um sie einer eindeutigen Beurteilung zugänglich zu machen.

In dieser Untersuchung wurde die ontologische Evaluierung als Methode nicht zur Disposition gestellt, sondern unkritisch übernommen. Gleichwohl können gegenüber diesem Ansatz unterschiedliche Kritikpunkte angeführt werden. Kritiker werfen bspw. diesem Ansatz vor, dass die eingenommenen Basisannahmen nicht vertretbar sind [Ze01, 210-215] bzw. die verwendeten Evaluierungskriterien nicht mit seinen metatheoretischen Basisannahmen harmonisieren [SZ99, S. 7]. Weitere Kritik zielt darauf ab, dass Evaluierungsergebnisse nicht allgemeingültig sind bzw. wenig überraschen [Fr98, S. 13]. Inwieweit diese Kritik berechtigt ist, bedarf einer intensiveren Auseinandersetzung, die im Rahmen dieser Untersuchung nicht geleistet werden konnte. Zur Verteidigung werden hier nur zwei Aspekte angeführt. Erstens hat jede wissenschaftliche Untersuchung u. a. ontologische Basisannahmen zu treffen, die selbstverständlich kritisierbar sind. Allerdings sollten alternative Konzepte für die Wissenschaftspraxis vorgeschlagen werden, die den hier verwendeten Annahmen überlegen sind. Zweitens ist der Vorwurf einer mangelnden Innovationskraft der Evaluierungsergebnisse durchaus differenzierter zu betrachten. Zunächst ist u. a. die tatsächliche Innovationsleistung der vorgestellten Evaluierungsergebnisse umfassender zu beurteilen: Während bspw. einer der anonymen Gutachter der Ansicht ist, dass die vorgenommene Evaluierung keine neuen Erkenntnisse liefert, so beurteilt der andere anonyme Gutachter den Beitrag als „einigermaßen innovativ". Dies belegt, dass die tatsächliche Innovationsleistung diskussionswürdig ist. Schließlich ist darauf hinzuweisen, dass selbst der Fall, dass die vorgebrachten Evaluierungsergebnisse bereits vollständig bekannt sind, aus einem anderen Blickwinkel durchaus positiv zu interpretieren ist: Dieser Fall zeigt gerade, dass eine ontologische Evaluierung in der Lage ist, systematisch Schwachstellen einer Modellierungssprache aufzudecken – schließlich wurden die hier dargestellten Interpretationsspielräume der EPK-Konstrukte nicht ad-hoc, sondern methodisch fundiert identifiziert und beschrieben. Damit hat der Ansatz Potenzial zur systematischen Sprachbeurteilung. Vor diesem Hintergrund erscheint es reizvoll, das Instrument nicht nur zur Beurteilung *bekannter* Sprachen zu verwenden, sondern bereits bei der Gestaltung *neuer* Modellierungssprachen einzusetzen.

Weitere Forschungsbestrebungen können darauf abzielen, die identifizierten ontologischen Defizite als Ausgangspunkt zu verwenden, um EPK-Konstrukte einzuführen, die diese Defizite vermeiden. Hier ist bspw. an Konzepte zur Verknüpfung von Ereignissen mit Prozessobjekttypen zu denken. Ebenso können die hier formulierten Aussagen als

Hypothesen verstanden werden, welche eine Grundlage für empirische Tests bilden. Auf diese Weise können bspw. praktische Konsequenzen ontologischer Defizite untersucht werden. Ein weiterer Ansatzpunkt ergibt sich aus der Entwicklung von ontologisch motivierten Modellierungsregeln [EW01a], mit deren Hilfe ontologische Defizite bei der Modellkonstruktion vermeidbar sind.

Literaturverzeichnis

[Aa99] Aalst van der, W. M. P.: Formalization and verification of event-driven process chains. In: Information and Software Technology 41 (1999), S. 639-650.

[Bo01] Bodart, F.; Patel, A.; Sim, M.; Weber, R.: Should Optional Properties Be Used in Conceptual Modelling? A Theory and Three Empirical Tests. In: Information Systems Research 12 (2001) 4, S. 384-405.

[BS96] Becker, J.; Schütte, R.: Handelsinformationssysteme. Landsberg/Lech 1996.

[Bu03] Bunge, M.: Philosophical Dictionary. 2. Aufl., Amherst, New York 2003.

[Bu77] Bunge, M.: Ontology I: The Furniture of the World. Dordrecht, Holland 1977.

[Bu79] Bunge, M.: Ontology II: A World of Systems. Dordrecht, Holland 1979.

[Bu83] Bunge, M.: Epistemology & Methodology II: Understanding the World. Dordrecht, Holland 1983.

[Da00] Davis, R.: Business Process Modelling With ARIS: A Practical Guide. 2000.

[Es93] Esswein, W.: Das Rollenmodell der Organisation: Die Berücksichtigung aufbauorganisatorischer Regelungen in Unternehmensmodellen. In: Wirtschaftsinformatik 35 (1993) 6, S. 551-561.

[EW01a] Evermann, J.; Wand, Y.: Towards Ontologically Based Semantics for UML Constructs. In: H. S. Kunii; S. Jajodia; A. Sølvberg (Hrsg.): Conceptual Modeling - ER 2001 - 20th International Conference on Conceptual Modeling, Yokohama, Japan, November 27-30, 2001, Proceedings. Berlin, Heidelberg 2001, S. 354-367.

[EW01b] Evermann, J.; Wand, Y.: An Ontological Examination of Object Interaction in Conceptual Modeling. Proceedings of the 11th Workshop on Information Technologies and Systems (WITS 2001). New Orleans, Louisiana 2001

[FL03a] Fettke, P.; Loos, P.: Ontologische Evaluierung von Referenzmodellen - Methode und Anwendungen. In: E. J. Sinz; M. Plaha; P. Neckel (Hrsg.): Modellierung betrieblicher Informationssysteme - MobIS 2003 - Proceedings der Tagung MobIS 2003, 9. bis 10. Oktober 2003 in Bamberg. Bonn 2003, S. 155-173.

[FL03b] Fettke, P.; Loos, P.: Ontologische Evaluierung des Semantischen Objektmodells. In: E. J. Sinz; M. Plaha; P. Neckel (Hrsg.): Modellierung betrieblicher Informationssysteme - MobIS 2003 - Proceedings der Tagung MobIS 2003, 9. bis 10. Oktober 2003 in Bamberg. Bonn 2003, S. 109-129.

[Fr98] Frank, U.: Evaluating Modelling Languages: Relevant Issues, Epistemological Challenges and a Preliminary Research Framework. Institut für Wirtschaftsinformatik der Universität Koblenz Landau, Arbeitsbericht Nr. 15. Koblenz 1998.

[FS01] Frank, U.; Schauer, H.: Software für das Wissensmanagement. In: WISU 30 (2001) 5, S. 718-726.

[GR00a] Green, P.; Rosemann, M.: An Ontological Analysis of Integrated Process Modelling. In: M. Jarke; A. Oberweis (Hrsg.): CAiSE'99. Berlin et al. 2000, S. 225-240.

[GR00b] Green, P.; Rosemann, M.: Integrated Process Modeling: An Ontological Evaluation. In: Information Systems 25 (2000) 2, S. 73-87.

[GR01] Green, P.; Rosemann, M.: Ontological Analysis of Integrated Process Models: Testing Hypotheses. In: Australian Journal on Information Systems 9 (2001) 1, S. 30-38.

[Gr95] Gruber, T. R.: Toward principles for the design of ontologies used for knowledge sharing. In: International Journal of Human-Computer Studies 43 (1995), S. 907-928.

[Gr96] Green, P.: An Ontological Analysis of Information Systems Analysis and Design (ISAD) Grammars in Upper Case Tools. PhD Thesis, University of Queensland., 1996.

[Ke92] Keller, G.; Nüttgens, M.; Scheer, A.-W.: Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)". Institut für Wirtschaftsinformatik, Universität Saarbrücken, 89. Saarbrücken 1992.

[NR02] Nüttgens, M.; Rump, F. J.: Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK). In: J. Desel; M. Weske (Hrsg.): Promise 2002 - Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen - Proceedings des GI-Workshops und Fachgruppentreffens (Potsdam, Oktober 2002). Bonn 2002, S. 64-66.

[OH01] Opdahl, A. L.; Henderson-Sellers, B.: Grounding the OML metamodel in ontology. In: The Journal of Systems and Software 57 (2001) 2, S. 119-143.

[OH02] Opdahl, A. L.; Henderson-Sellers, B.: Ontological Evaluation of the UML Using the Bunge-Wand-Weber Model. In: Software and Systems Modeling 1 (2002) 1, S. 43-67.

[RG02] Rosemann, M.; Green, P.: Developing a meta model for the Bunge-Wand-Weber ontological constructs. In: Information Systems 27 (2002) 2, S. 75-91.

[RK98] Runggaldier, E.; Kanzian, C.: Grundprobleme der Analytischen Ontologie. Paderborn et al. 1998.

[Ro95a] Rohde, F.: An Ontological Evaluation of Jackson's System Development Model. In: Australian Journal of Information Systems 2 (1995) 2, S. 77-87.

[Ro95b] Rosemann, M.: Erstellung und Integration von Prozeßmodellen - Methodenspezifische Gestaltungsempfehlungen für die Informationsmodellierung. Diss., Münster 1995.

[Ro97] Rodenhagen, J.: Darstellung ereignisgesteuerter Prozeßketten (EPK) mit Hilfe von Petrinetzen. Diplomarbeit, Universität Hamburg, Hamburg 1997.

[Ru98] Rump, F.: Durchgängiges Management von Geschäftsprozessen auf Basis ereignisgesteuerter Prozeßketten. Diss., Universität Oldenburg, Oldenburg 1998.

[Sc97] Scheer, A.-W.: Wirtschaftsinformatik - Referenzmodelle für industrielle Geschäftsprozesse. 7. Aufl., Berlin et al. 1997.

[Sc98] Schütte, R.: Grundsätze ordnungsmäßiger Referenzmodellierung - Konstruktion konfigurations- und anpassungsorientierter Modelle. Wiesbaden 1998.

[SZ99] Schütte, R.; Zelewski, S.: Wissenschafts- und erkenntnistheoretische Probleme beim Umgang mit Ontologien. In: W. König; O. Wendt (Hrsg.): Wirtschaftsinformatik und Wissenschaftstheorie '99 - Verteilte Theoriebildung. Frankfurt 1999

[Ut97] Uthmann von, C.: Nutzenpotentiale der Petrinetztheorie für die Erweiterung der Anwendbarkeit Ereignisgesteuerter Prozeßketten. Workshop "Formalisierung und Analyse ereignisgesteuerter Prozessketten (EPK)". Universität Oldenburg 1997

[Vo94] Vossen, G.: Datenmodelle, Datenbanksprachen und Datenbank-Management-Systeme. 2. Aufl., Bonn et al. 1994.

[Wa95] Wand, Y.; Monarchi, D. E.; Parsons, J.; Woo, C. C.: Theoretical foundations for conceptual modelling in information systems development. In: Decision Support Systems 15 (1995), S. 285-304.

[Wa99] Wand, Y.; Storey, V. C.; Weber, R.: An Ontological Analysis of the Relationship Construct in Conceptual Modeling. In: ACM Transactions on Database Systems 24 (1999) 4, S. 494-528.

[We96] Weber, R.: Are Attributes Entities? A Study of Database Designer's Memory Structures. In: Information Systems Research 7 (1996) 2, S. 137-162.

[We97a] Weber, R.: Ontological Foundations of Information Systems. Melbourne 1997.

[WW02] Wand, Y.; Weber, R.: Research Commentary: Information Systems and Conceptual Modelling - A Research Agenda. In: Information Systems Research 13 (2002), S. 363-377.

[WW89a] Wand, Y.; Weber, R.: An Ontological Evaluation of Systems Analysis and Design Methods. In: E. D. Falkenberg; P. Lindgreen (Hrsg.): Information Systems Concepts: An In-Depth Analysis. North-Holland 1989, S. 79-107.

[WW90] Wand, Y.; Weber, R.: Toward a Theory of the Deep Structure of Information Systems. In: J. I. DeGross; M. Alavi; H. Oppelland (Hrsg.): International Conference on Information Systems. Copenhagen, Denmark 1990

[WW93] Wand, Y.; Weber, R.: On the ontological expressiveness of information systems analysis and design grammars. In: Journal of Information Systems 3 (1993) 4, S. 217-237.

[WW95] Wand, Y.; Weber, R.: On the deep structure of information systems. In: Information Systems Journal 5 (1995), S. 203-223.

[WZ91] Weber, R.; Zhang, Y.: An Ontological Evaluation of NIAM's Grammar for Conceptual Schema Diagrams. International Conference on Information Systems. New York, New York 1991

[WZ96] Weber, R.; Zhang, Y.: An analytical evaluation of NIAM's grammar for conceptual schema diagrams. In: Information Systems Journal 6 (1996), S. 147-170.

[Ze01] Zelewski, S.; Schütte, R.; Siedentopf, J.: Ontologien zur Repräsentation von Domänen. In: G. Schreyögg (Hrsg.): Wissen in Unternehmen - Konzepte, Maßnahmen, Methoden. Berlin 2001, S. 183-221.

Anhang: Bunge-Wand-Weber-Modell[2]

In der Tabelle 2 werden die Konstrukte des BWW-Modells zusammenfassend dargestellt. Das elementare Konstrukt der Ontologie ist ein BWW-Gegenstand. Die Welt ist aus BWW-Gegenständen zusammengesetzt. Als BWW-Gegenstand werden grundsätzlich nur materielle Gegenstände (Person, Buch etc.) und keine konzeptionellen Gegenstände (Adresse, Zeit, Bestellung etc.) verstanden. Ein BWW-Gegenstand kann einfach oder zusammengesetzt sein. Ein zusammengesetzter BWW-Gegenstand besteht aus mehreren einfachen oder zusammengesetzten BWW-Gegenständen. BWW-Gegenstände besitzen BWW-Eigenschaften. Eine BWW-Eigenschaft wird beschrieben durch eine Funktion, die einen BWW-Gegenstand auf einen BWW-Eigenschaftswert abbildet (BWW-Eigenschaftsfunktion). Jeder BWW-Gegenstand besitzt BWW-Eigenschaften. Ferner können BWW-Eigenschaften nur BWW-Gegenständen zugesprochen werden. Es wird unterschieden zwischen eigenen BWW-Eigenschaften, die einem spezifischen BWW-Gegenstand zugeschrieben werden, und gemeinsamen BWW-Eigenschaften, die zwei oder mehreren BWW-Gegenständen zugeschrieben werden. Das Fehlen einer BWW-Eigenschaft ist keine BWW-Eigenschaft. BWW-Eigenschaften selber können keine BWW-Eigenschaften tragen. BWW-Gegenstände besitzen BWW-Eigenschaften unabhängig davon, ob diese dem Menschen bekannt oder bewusst sind. Die BWW-Eigenschaften von BWW-Gegenständen werden mit Hilfe von BWW-Attributen vom Menschen wahrgenommen. BWW-Attribute können als Namen für BWW-Eigenschaften verstanden werden. Der BWW-Zustand eines BWW-Gegenstandes wird definiert durch den Vektor der BWW-Eigenschaftswerte, die diesem BWW-Gegenstand durch alle seiner BWW-Eigenschaftsfunktionen zugeordnet werden. Ein BWW-Ereignis ist definiert als ein Wechsel des BWW-Zustandes eines BWW-Gegenstandes.

BWW-Konstrukt	Erläuterung
Gegenstand	Das elementare Konstrukt des BWW-Modells. Die Welt ist aus Gegenständen zusammengesetzt. Ein zusammengesetzter Gegenstand setzt sich zusammen aus weiteren zusammengesetzten oder einfachen Gegenständen.
Eigenschaften	Gegenstände besitzen Eigenschaften. Eine Eigenschaft wird repräsentiert als eine (Eigenschafts-)Funktion, die den Gegenstand auf einen (Eigenschafts-)Wert abbildet.
Zustand	Vektor der Eigenschaftswerte aller Eigenschaftsfunktionen eines Gegenstandes.
Denkbarer Zustandsraum	Menge aller Zustände, die der Gegenstand jemals annehmen kann.
Zustandsgesetz	Ein Zustandsgesetz beschränkt die Werte einer Eigenschaftsfunktion eines Gegenstandes aufgrund von Naturgesetzen oder menschlichen Regeln auf eine gültige Teilmenge.

Tabelle 2: Konstrukte des BWW-Modells (Quelle: [WW93; WZ96], Teil 1/3)

[2] Die Darstellung ist den Beiträgen [FL03a; FL03b] entnommen.

BWW-Konstrukt	Erläuterung
Gesetzmäßiger Zustandsraum	Der gesetzmäßige Zustandsraum eines Gegenstandes ist der Zustandsraum, der den Zustandsgesetzen genügt. Normalerweise ist dieser eine geeignete Teilmenge des denkbaren Zustandsraums.
Ereignis	Ein Ereignis ist ein Wechsel des Zustandes eines Gegenstandes. Es verursacht eine Transformation (siehe unten).
Ereignisraum	Menge aller möglichen Ereignisse, die in einem Gegenstand auftreten können.
Transformation	Eine Transformation ist eine Abbildung von Zuständen auf Zustände.
Gesetzmäßige Transformation	Die gesetzmäßige Transformation beschreibt, welche Ereignisse in einem Gegenstand gesetzmäßig sind.
Gesetzmäßiger Ereignisraum	Der gesetzmäßige Ereignisraum ist die Menge aller Ereignisse in einem Gegenstand, die gesetzmäßig sind.
Historie	Die Historie beschreibt die chronologisch geordnete Folge von Zuständen, die ein Gegenstand durchläuft.
Kopplung	Ein Gegenstand wirkt auf einen anderen Gegenstand ein, wenn die Existenz des ersteren die Historie des letzteren beeinflusst. Dann sind beide Gegenstände gekoppelt.
System	Eine Menge von Gegenständen ist ein System, wenn für jede Zweiteilung der Menge Kopplungen zwischen den Gegenständen beider Teilmengen existieren.
Systemkomposition	Menge der Gegenstände, die ein System bilden.
Systemumgebung	Menge der Gegenstände, die nicht zur Systemkomposition gehören, aber mit Gegenständen des Systems gekoppelt sind.
Systemstruktur	Menge aller Kopplungen, die zwischen den Gegenständen des Systems und den Gegenständen der Umgebung vorhanden sind.
Subsystem	System, dessen Systemkomposition und Systemstruktur Teilmengen der Systemkomposition und Systemstruktur eines anderen Systems sind.
Systemdekomposition	Menge von Subsystemen eines Systems, für die gilt, dass jede Komponente des Systems entweder ein Subsystem der Systemdekomposition ist oder zu einer Systemkomposition eines Subsystems der Systemdekomposition gehört.
Stufenstruktur	Definiert eine partielle Ordnung über die Subsysteme einer Systemdekomposition, aus der hervorgeht, welche Subsysteme Komponenten anderer Subsysteme sind.
Stabiler Zustand	Zustand, aus dem heraus das System keinen anderen Zustand einnimmt, außer wenn ein Gegenstand der Systemumgebung einen Zustandswechsel hervorruft (externes Ereignis)
Instabiler Zustand	Zustand, der nicht stabil ist.

Tabelle 2: Konstrukte des BWW-Modells (Quelle: [WW93; WZ96], Teil 2/3)

BWW-Konstrukt	Erläuterung
Externes Ereignis	Ereignis in einem System, dass durch einen Zustandswechsel eines Gegenstandes außerhalb des Systems hervorgerufen wurde. Der Vorzustand eines externen Ereignisses ist stets stabil, der Nachzustand kann stabil oder instabil sein.
Internes Ereignis	Ereignis in einem System, dass durch einen Zustandswechsel eines Gegenstandes des Systems hervorgerufen wurde. Der Vorzustand eines internen Ereignisses ist stets instabil, der Nachzustand kann stabil oder instabil sein.
Wohl-definiertes Ereignis	Ereignis, bei dem der nächste Zustand des Systems vorhergesagt werden kann, wenn der aktuelle Systemzustand bekannt ist.
Schwach-definiertes Ereignis	Ereignis, bei dem der nächste Zustand des Systems nicht vorhergesagt werden kann.
Klasse	Menge von Gegenständen, die eine gemeinsame Eigenschaft besitzen.
Gattung	Menge von Gegenständen, die zwei oder mehr gemeinsame Eigenschaften besitzen.

Tabelle 2: Konstrukte des BWW-Modells (Quelle: [WW93; WZ96], Teil 3/3)

Dokumentenmanagement-Prozesse

Modellierung der „versteckten Prozesse“ mit ARIS-EPK

Prof. Dr. Heinrich Seidlmeier

Fachbereich Betriebswirtschaft
Fachhochschule Rosenheim (University of Applied Sciences)
Hochschulstr. 1
83024 Rosenheim
seidlmeier@fh-rosenheim.de

Abstract: Dokumente als Informations- und Wissensträger spielen in Unternehmen und damit in betrieblichen Prozessen eine wichtige Rolle. Trotzdem wird die Modellierung von dokumentenbezogenen Prozessen (Erfassen, Indizieren, Ablegen, Recherchieren, Anzeigen), z.B. in Form von ereignisgesteuerten Prozessketten (EPK), i.d.R. vernachlässigt. Aufgrund der fehlenden, aber notwendigen Modellierungstiefe bleiben diese Prozesse „versteckt“. In diesem Papier soll gezeigt werden, wie die Modellierung von Dokumentenmanagement-Prozessen mit ARIS-EPK[1] möglich ist. Dadurch können Rationalisierungspotentiale von elektronischen Dokumentenmanagement-Systemen besser offen gelegt werden.

1 Definition Dokument und Dokumentenmanagement-Prozess

Ein Dokument in einem sehr weiten Sinne ist eine für den menschlichen Gebrauch aufbereitete Einheit an gespeicherter Information [Sp95]. Danach sind Dokumente alle kaufmännischen und technischen Unterlagen (allgemeine Korrespondenz, Rechnungen, Konstruktionszeichnungen u.ä.), aber auch Grafiken, Videos und eMail-Nachrichten usw. Dokumente können in Papierform oder digital (elektronisch) vorliegen. Digitale Dokumente sind zu unterscheiden von formatierten elektronischen Daten, die in Dateien oder Datenbanken abgespeichert sind [Mo97][2].

[1] Unter einer ARIS-EPK wird im folgenden das erweiterte EPK-Modell verstanden, welches das ARIS Toolset zur Prozessmodellierung zur Verfügung stellt.

[2] Formatierte (satzorientierte) Daten sind Datenbestände, die aus Datensätzen (als Menge von Datenelementen) mit einer festen Feldeinteilung bestehen. Elektronische Dokumente sind demnach unformatierte Daten [StHa02].

Dokumentenmanagement-Prozesse, kurz DM-Prozesse, [vgl. zum folgenden Se02] sind Prozesse, die primär zur Durchführung von Unterstützungs-, aber auch von Geschäftsprozessen notwendig sind [vgl. zur Unterscheidung von Unterstützungs- bzw. Support- und Geschäftsprozessen z.B. OF00]. DM-Prozesse als eher kundenferne Prozesse erzeugen in den meisten Fällen keine direkte Wertschöpfung – außer in „dokumentenintensiven" Branchen wie der Versicherungswirtschaft, in der das Leistungsprodukt (z.B. eine Police) als Dokument dargestellt wird.

Ein DM-Prozess kann im Extremfall nur aus einer einzelnen Funktion oder auch aus mehreren Teilprozessen bestehen. Diesem Paper werden die folgenden generischen DM-Prozesse zugrunde gelegt:[3]

- Dokumente erfassen
  = Zur weiteren Verarbeitung vorbereitende Tätigkeiten (z.B. Öffnen von Kuverts, Abstempeln des Eingangs, Scannen)

- Dokumente klassifizieren (indizieren)
  = Vergabe von Dokumentattributen, insbesondere zur späteren Recherche (z.B. Schlüssel oder Dokumententyp)

- Dokumente ablegen (archivieren)
  = Aufbewahrung des Dokuments in einem Speicher (z.B. Papierordner oder elektronisches Archiv)

- Dokumente recherchieren
  = Suchen und (Wieder-) Finden von Dokumenten (mittels definierter Suchattribute)

- Dokumente anzeigen
  = Sichtbarmachen oder auch Ausdrucken von gesuchten Dokumenten

Betrachtet man Prozesse vollständig aus Dokumentensicht, können neben den eben dargestellten DM-Prozessen noch Dokumentenerstellungsprozesse und dokumentenlose Prozesse unterschieden werden. Dokumentenerstellungsprozesse erzeugen als Prozessergebnis Dokumente bzw. genauer dokumentierte Leistungen (z.B. Erstellung eines Gutachtens). Dokumentenlose Prozesse haben keinerlei Bezug zu Dokumenten (z.B. Erstellung einer Berechnung mit einem Tabellenkalkulationsprogramm „aus dem Kopf").

[3] In der Standardliteratur zum Dokumentenmanagement bzw. zu Dokumentenmanagementsystemen [z.B. BL94, Gö01, GSS99] wird i.d.R. von Funktionen, Aktivitäten oder Anforderungen an elektronische Dokumentenmanagementsysteme gesprochen. Diese Begriffe spiegeln aber nicht den ihnen innewohnenden Prozesscharakter wider.

2 Bedeutung von Dokumentenmanagement-Prozessen[4]

Die gewichtige Rolle von Dokumenten im täglichen und beruflichen Leben erschließt sich vermutlich intuitiv sehr schnell, soll aber auch fundierter belegt werden. Dabei wird, über die nahe liegende reine Informationsversorgung durch Dokumente hinaus, die Steuerungsfunktion von Dokumenten hervorgehoben.

„Das Ergebnis einer wertschöpfenden Aktivität besteht v.a. aus auf Dokumenten fixierten Informationen, die wiederum die Ausführung nachfolgender Aktivitäten veranlassen. Es ist somit möglich, Vorgänge durch Dokumente umfassend zu beschreiben" [Ra95, S. 468; die Autoren zitieren auch McDSo91, die von einer XEROX-Studie berichten. Darin zeigte sich, dass 92% aller Kanten eines Kerngeschäftsprozesses durch Dokumente determiniert wurden].

Auf der Basis der Modellierungsmethode „Ereignisgesteuerte Prozessketten (EPK)", entwickelt am Institut für Wirtschaftsinformatik an der Universität des Saarlandes, [KNSch92] lassen sich die folgenden theoretischen Überlegungen ableiten [Se00]:

Die Bedeutung von Dokumenten für die Prozessqualität, -kosten und -zeit wird über die Dokumentenintensität von Prozessen definiert. Dokumente sind dann bedeutsam, d.h. auch relevant für das Prozessergebnis, wenn

- der Anteil der dokumentenabhängigen Prozessschritte (= Funktionen) relativ hoch ist und/oder
- der Anteil der dokumentenbestimmten Prozesszustände (= Ereignisse) relativ hoch ist.[5]

[4] Trotz der in diesem Abschnitt zu zeigenden Wichtigkeit von Dokumenten und Dokumentenmanagement-Prozessen unterbleibt meist eine adäquate Berücksichtigung bei der Prozessmodellierung mit EPK. In diesem kurzen Position Paper kann nicht auf eine fundierte Literaturauswertung oder auf empirische Befunde verwiesen werden. Die folgenden zwei Hinweise sollen aber die Position des Autors stützen.
Das Ziel von [Ru99] ist es, „die methodischen Grundlagen für ein durchgängiges Management von Geschäftsprozessen auf der Basis ereignisgesteuerter Prozessketten zu schaffen". Auf die zentrale Rolle von Dokumenten und deren Management in Geschäftsprozessen wird nicht explizit eingegangen.
Das Internet-Literaturverzeichnis des Arbeitskreises „Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten" in der Gesellschaft für Informatik (http://epk.et-inf.fho-emden.de/literatur.php; Stand 25.09.03) weist 179 EPK-bezogene Quellen für den Zeitraum 1992 bis 2003 aus. Neben einer Quelle des Autors selbst [Se00] enthält die Übersicht nur noch eine weitere Quelle [Gi98], die im Titel einen expliziten Bezug zum Thema „Dokumentenmanagement" aufweist und dadurch eine entsprechende Schwerpunktbehandlung nahelegt – wobei [Gi98] die Thematik „Dokumente in EPK" nicht aufgreift.

[5] „Relativ hoch" soll zumindest bedeuten, dass der Anteil der dokumentenabhängigen Funktionen bzw. der dokumentengesteuerten Ereignisse an den jeweiligen Gesamtmengen größer als 50% ist.

Diese beiden Kriterien beziehen sich unmittelbar auf die beiden zentralen Modellkonstrukte von EPK, auf Funktionen und Ereignisse.[6] Ein Prozessschritt, eine EPK-Funktion, ist dann dokumentenabhängig, wenn ohne die Dokumenteninformation (bzw. auch ohne das dokumentierte Wissen) der Schritt nicht durchgeführt werden kann. Beispielsweise kann eine Lieferantenrechnung nicht ohne diese Rechnung an sich, aber auch nicht ohne die ursprüngliche Bestellung, den Wareneingangsschein u.ä. geprüft werden.

Eine Prinzipdarstellung soll diesen Sachverhalt visualisieren (vgl. dazu die folgende Abbildung 1). Geht man davon aus, dass für jede Funktionsdurchführung ein informatorischer Input notwendig ist und ein informatorischer Output erzeugt wird, stellt sich die Frage, in welcher Form („Dokument“ oder „Datei“) der „Rohstoff“ Daten vorliegt und mit welcher Art von Wissen aus Daten Informationen erzeugt werden.[7]

Ein Prozessschritt bzw. eine Funktion ist, wie bereits erwähnt, dann dokumentenabhängig, wenn zur Durchführung Dokumente bzw. dokumentiertes Wissen unabdingbar sind. Analog gilt, wenn alle durchführungsnotwendigen Daten in Dateien (Datenbanken) vorliegen und die relevanten Informationen durch nicht dokumentiertes (explizites oder implizites) Wissen generiert werden, ist eine Funktion dokumentenunabhängig. Diese Überlegungen spielen insbesondere bei Einführung von elektronischen Dokumentenmanagementsystemen eine Rolle.

Ereignisse steuern in EPK den Prozessablauf, insbesondere den Prozessstart und den weiteren Verlauf an Verzweigungen (Operatoren, Regeln). Von einem dokumentenbestimmten Zustand spricht man, wenn das betreffende EPK-Ereignis direkten Dokumentenbezug hat. Dies wird sich bei der Prozessmodellierung schon meist, aber nicht zwingend, aus der Ereignisformulierung ergeben – Beispiel: „Lieferantenrechnung ist eingetroffen“. In diesem Beispiel startet das Dokument „Rechnung“ den entsprechenden Rechnungsprüfungsprozess.

---

[6] Mit dem Dokument als Informationsobjekt wird die dritte zentrale Modellkomponente [KNSch92] in die EPK-orientierten Überlegungen einbezogen.

[7] [SchKr99] unterscheiden in pragmatischer Weise die Begriffe Daten, Informationen und Wissen: Daten bestehen aus syntaktisch richtigen Zeichenketten und beinhalten keinen Verwendungszweck. Betrachtet man Daten in einem bestimmten Kontext, erhalten sie damit Zweckbezug, spricht man von Information. Wissen kann man als sinnvolle zweckorientierte Informationsvernetzung („Metainformation“) betrachten. In der Abbildung 1 geht man folglich davon aus, dass Wissen notwendig ist, um aus Daten Informationen zu generieren.

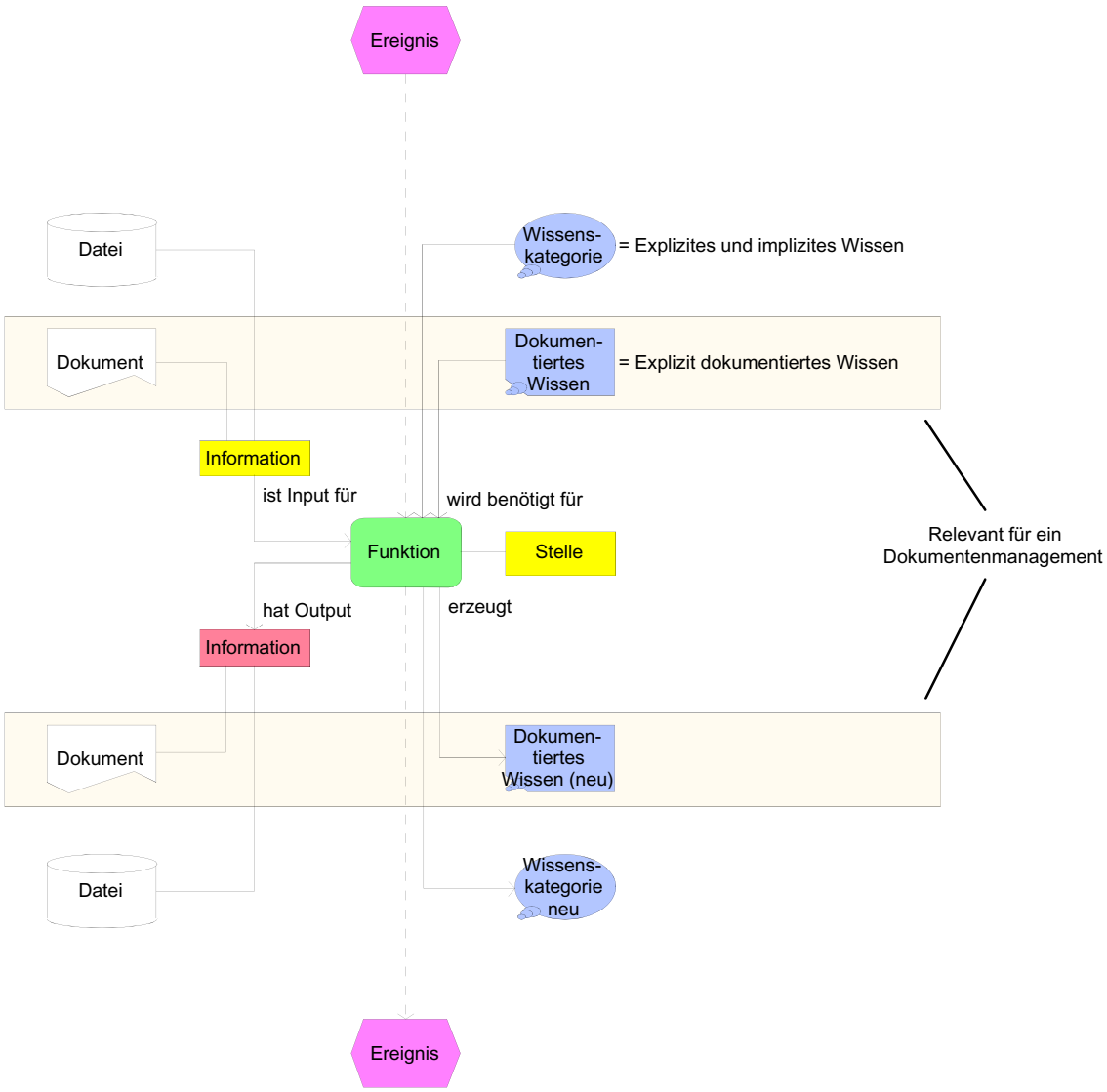

Abbildung 1: Dokumentenabhängigkeit

Neben diesen EPK-bezogenen Kriterien spielen für die Bedeutung von Dokumenten (und deren Management) für Prozesse auch noch andere Faktoren eine Rolle, beispielsweise die Menge und die Verschiedenartigkeit an notwendigen Dokumenten oder an dokumentenbezogenen EDV-Systemen (wie eMail-, Textverarbeitungsprogramme usw.).

3 Modellierung von Dokumentenmanagement-Prozessen in Schichten

In diesem Paper sollen abschließend drei „Modellierungsschichten“ mit zunehmender Prozessdetaillierung definiert und erläutert werden, um die i.d.R. nicht dargestellten DM-Prozesse Schritt für Schritt in Form von ARIS-EPK offen zu legen [vgl. zu weiteren passenden ARIS-Modellen für DM-Prozesse ausführlich Se02].

Als Modellierungsschichten werden vorgeschlagen:[8]

- Schicht 1: Grobe Basis-EPK ohne Dokumentenbezug (d.h. ohne Berücksichtigung der ARIS-Daten- und Organisationssicht, aber mit entsprechenden ARIS-Hinterlegungen zum Verweis auf die Modelle in Schicht 2) – Bsp.: Innerhalb eines Prozesses „Kundenkontakt bearbeiten“ ist die Funktion „Reklamation prüfen“ durchzuführen.

- Schicht 2: DM-Funktions-EPK mit Dokumenten (d.h. die in Kap. 1 eingeführten DM-Prozesse werden nur als ARIS-Funktionen dargestellt, denen die Prozesse in Schicht 3 hinterlegt werden) – Bsp.: Zur Prüfung der Reklamation ist (u.a.) die Funktion „Recherche durchführen“ zu erledigen.

- Schicht 3: DM-Prozesse als EPK (d.h. die DM-Prozesse werden detailliert als Funktionsabfolgen dargestellt) – Bsp.: „Recherche durchführen“ besteht aus einer Reihe von weiteren Unterfunktionen (vgl. dazu Abbildung 2[9]).

Insbesondere die dritte, meist vernachlässigte Prozessschicht legt den Umfang von DM-Prozessen offen (z.B. Anzahl Funktionen und beteiligte Organisationsobjekte, Zeit- und Kostenanfall der Prozessdurchführung). Dieser Prozessumfang (zusammen mit Angaben über die Häufigkeit der Prozessdurchführungen) hilft, die Notwendigkeit und den Nutzen von elektronischen Dokumentenmanagement-Systemen (DMS) abzuschätzen. Dies verdeutlicht Abbildung 2. Ist gemäß dem oben erwähnten Beispiel eine Recherche durchzuführen, sind dazu im ungünstigsten Falle (d.h. das Ereignis „Registratur“ ist eingetreten) insgesamt 11 Unterfunktionen durchzuführen. Der Einsatz eines DMS würde diese 11 Arbeitsschritte auf die zwei arbeitsplatzbezogenen Funktionen „Recherche im elektronischen Gesamtarchiv“ und „Dokumenten bearbeiten“ reduzieren – mit den entsprechenden Zeit- und Kosteneinsparungen und ggf. einem Gewinn an Recherchesicherheit.[10]

Durch die vorgestellte tiefere dokumentenbezogene Sichtweise wird der Rationalisierungsbeitrag von DMS in Geschäftsprozessoptimierungs-Projekten transparenter.

---

[8] Alle im folgenden verwendeten EPK sind als ARIS-EPK zu verstehen.

[9] Abbildung 2 zeigt ein reales Praxisbeispiel; aus Übersichtsgründen teilweise ohne Ereignisse und ohne Organisationsobjekte.

[10] Das Optimierungspotential eines DMS ist in diesem Beispielfall am größten, wenn die meisten Rechercheverfahren in die Registratur führen. Das Potential ist natürlich deutlich geringer, wenn die Mehrzahl an Recherchen am Arbeitsplatz durchgeführt werden kann.

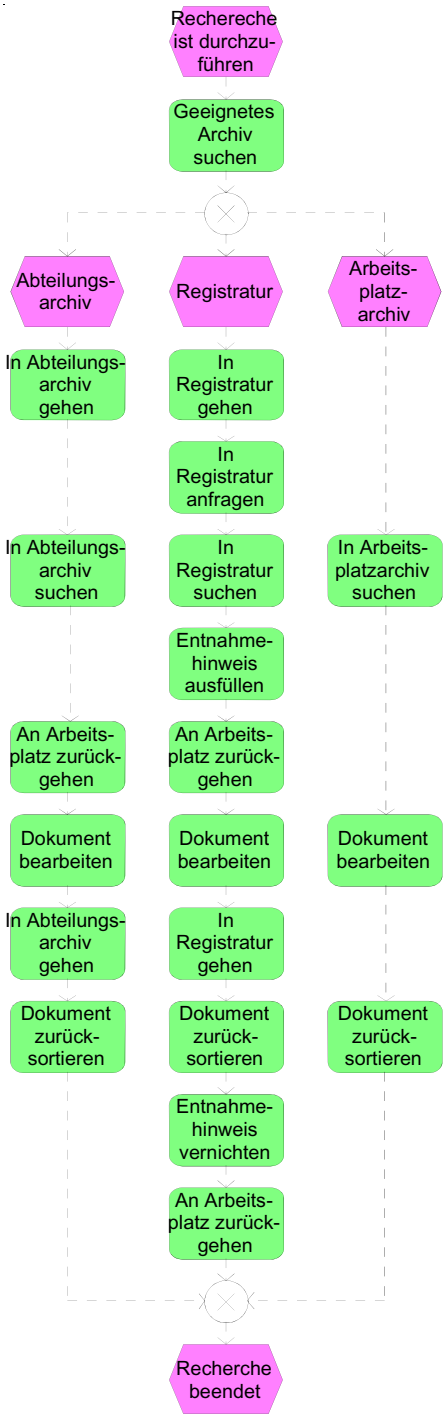

Abbildung 2: Beispiel einer DM-Prozess-EPK (Schicht 3) [in Anlehnung an Ga99]

Literaturverzeichnis

[BL94] Berndt, O., Leger, L., Dokumenten Management Systeme, Luchterhand, Neuwied, usw., 1994.

[Ga99] Gaugler, M., Wirtschaftlichkeitsanalyse eines elektronischen Dokumenten-Management-Systems bei der Avery Dennison Zweckform, unveröffentlichte Diplomarbeit Fachhochschule Rosenheim, 1999.

[Gi98] Gierhake, O, Integriertes Geschäftsprozeßmanagement: Effektive Organisationsgestaltung mit Workflow-, Workgroup- und Dokumentmanagementsystemen, 2. Auflage, Vieweg, Braunschweig und Wiesbaden 1998.

[Gö01] Götzer, K. u.a., Dokumenten-Management, 2. Auflage, dpunkt, Heidelberg, 2001.

[GSS99] Gulbins, J., Seyfried, M., Strack-Zimmermann, H., Dokumenten-Management, Springer, Berlin usw., 1999.

[KNSch92] Keller, G., Nüttgens, M., Scheer, A.-W., Semantische Prozeßmodellierung auf der Grundlage "Ereignisgesteuerter Prozeßketten (EPK)", in: Scheer, A.-W. (Hrsg.): Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 89, Saarbrücken, 1992.

[McDSo91] McDonnel, E. D., Sommerville, G. E., Corporate reengineering that follows the design of document imaging, in: Information Strategy: The Executive's Journal, o.J. (1991) 3, S. 5 ff.

[Mo97] Morschheuser, S., Integriertes Dokumenten- und Workflow-Management, Deutscher Universitäts-Verlag, Wiesbaden,1997.

[OF00] Osterloh, M., Frost, J., Prozessmanagement als Kernkompetenz, 3. Auflage, Gabler, Wiesbaden, 2000.

[Ra95] Raufer, H. u.a., Ein Werkzeug zur Analyse und Modellierung von Geschäftsprozessen als Voraussetzung für effizientes Workflow-Management, in: Wirtschaftsinformatik 37 (1995) 5, S. 467 ff.

[Ru99] Rump, F. J., Geschäftsprozeßmanagement auf der Basis ereignisgesteuerter Prozeßketten, Teubner, Stuttgart und Leipzig, 1999.

[SchKr99] Schwarzer, B., Krcmar, H., Wirtschaftsinformatik, 2. Auflage, Schäffer-Poeschel, Stuttgart, 1999.

[Se00] Seidlmeier, H., Prozessmodellierung zur Gestaltung von Dokumentenmanagementsystemen, in: Schmidt, H. (Hrsg.), Modellierung betrieblicher Informationssysteme - PROCEEDINGS der MobIS-Fachtagung 2000 (Rundbrief der GI-Fachgruppe 5.10, 7. Jahrgang, Heft 1, Oktober 2000), Siegen, S. 205 ff.

[Se02] Seidlmeier, H., Prozessmodellierung mit ARIS, Vieweg, Braunschweig und Wiesbaden, 2002.

[Sp95] Sprague, R. H., Electronic Document Management, in: MIS Quarterly 19 (1995), 1, S. 29 ff.

[StHa02] Stahlknecht, P., Hasenkamp, U., Einführung in die Wirtschaftsinformatik, 10. Auflage, Springer, Berlin usw., 2002.

Kundenorientierte Dienstleistungsmodellierung mit Ereignisgesteuerten Prozessketten

Kristof Schneider, Oliver Thomas

Institut für Wirtschaftsinformatik (IWi)
im Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI)
Stuhlsatzenhausweg 3, Geb. 43.8, 66123 Saarbrücken
[schneider|thomas]@iwi.uni-sb.de

Abstract: Die notwendige Integration des Kunden in Dienstleistungserbringungsprozesse stellt Anforderungen an die Prozessmodellierung. Dieser Beitrag untersucht die Methode der Ereignisgesteuerten Prozesskette hinsichtlich ihrer Eignung zur Modellierung von Dienstleistungserbringungsprozessen und präsentiert eine Erweiterung, welche die organisatorische Zuordnung von Dienstleistungsaktivitäten betont sowie die Kundenintegrationsart visualisiert. Diese Erweiterung ist zugleich Grundlage zur Quantifizierung der Kundenintegration.[1]

1 Dienstleistungsmodellierung

Parallel zum amerikanischen New Service Development wird die systematische Entwicklung von Dienstleistungen seit Mitte der 1990er-Jahre in Deutschland unter dem Begriff „Service Engineering“ diskutiert [Bullinger 1995; Ramaswamy 1996; Cooper, Edgett 1999; Fitzsimmons, Fitzsimmons 2000]. Mit diesem Ansatz ist der Gedanke verbunden, das methodische Vorgehen der industriellen Produktentwicklung auf die Entwicklung von Dienstleistungen zu übertragen [Fähnrich 1998; Deutsches Institut für Normung e.V. 1998; Bullinger, Meiren 2001; Haller 2001; Bullinger, Scheer 2002]. Ziel eines systematischen Dienstleistungsentwicklungsprozesses ist es, die Differenz zwischen der erwarteten und der tatsächlichen Qualität von Dienstleistungen zu verringern, um das Potenzial dieser zur kundenindividuellen Bedürfnisbefriedigung zu steigern [Bruhn 2003, S. 1ff].

In der Literatur besteht Konsens darüber, dass Service Engineering die drei Dienstleistungsdimensionen Potenzial-, Prozess- und Ergebnisdimension fokussieren muss [Fähnrich, Meiren, Barth 1999, S. 14ff]. Diese bilden zugleich die Grundlage für die Erstellung unterschiedlicher Modellkonzepte für Dienstleistungen. Es werden Ressourcenkonzepte (Potenzialdimension), Prozessmodelle (Prozessdimension) und Produktmodelle (Ergebnisdimension) entwickelt [Grieble, Klein, Scheer 2002, S. 16ff; Thomas, Scheer 2002, S. 86ff].

Die Simultanität von Produktion und Absatz (Uno-actu-Prinzip) einerseits sowie die notwendige Integration eines externen Faktors in den Leistungserstellungsprozess ande-

[1] Dieser Beitrag präsentiert Ergebnisse aus dem BMBF-Verbundprojekt „Vom Kunden zur Dienstleistung: Wettbewerbsvorteile durch kundenorientiertes Service Engineering (CoRSE)“, Kennzeichen 01HG0094, URL http://www.corse-projekt.de/.

rerseits [Corsten 2001, S. 21ff] verdeutlichen, dass die Güte einer Dienstleistung nicht ausschließlich von der Beschaffenheit des Entwicklungsprozesses abhängt. Vielmehr spielt insbesondere die Qualität des Dienstleistungserbringungsprozesses eine entscheidende Rolle. Im Kern der Dienstleistungserbringung sollte daher die Frage stehen, auf welche Art die Integration des Kunden erfolgt, um eine optimale Erfüllung seiner Bedürfnisse und Erwartungen zu erreichen. Dieser Zusammenhang wird bis dato nur unzureichend in Modellierungsmethoden berücksichtigt.

In diesem Beitrag wird die Methode der Ereignisgesteuerten Prozesskette (EPK) [Keller, Nüttgens, Scheer 1992] hinsichtlich ihrer Eignung zur Modellierung von Dienstleistungserbringungsprozessen untersucht. Anschließend wird eine Erweiterung der Methode präsentiert, welche die organisatorische Zuordnung der Dienstleistungsaktivitäten betont sowie die Art der Kundenintegration visualisiert. Sie ist zugleich Grundlage zur Integrationsquantifizierung.

2 Existierende Ansätze zur Modellierung der Dienstleistungserbringung

Zur Modellierung und Ausführung von Geschäftsprozessen wurden eine Vielzahl semiformaler Modellierungsmethoden entwickelt. Im Umfeld des Service Engineering wird zur Modellierung der Prozessdimension neben der EPK vor allem die Methode des Service Blueprinting diskutiert. Mit dieser von SHOSTACK [1981; 1982] entwickelten Methode werden Ablaufdiagramme von Dienstleistungen erstellt. Auf diese Weise erfolgt eine detaillierte und transparente Visualisierung der durchzuführenden Arbeitsabläufe. Der Einsatz so genannter Swimlanes ermöglicht die Zuordnung der Tätigkeiten zu Anbieter und Nachfrager. Die externe Kontaktlinie verdeutlicht, welche Arbeitsschritte ausschließlich vom Kunden durchgeführt werden. Die Sichtbarkeitslinie stellt die Grenze des Wahrnehmungsbereichs des Kunden dar. Unterhalb dieser sind die Schritte platziert, die unternehmensintern ohne den Kunden durchgeführt werden. Die interne Interaktionslinie separiert die Unternehmensleistungen von den Leistungen externer Partner.

KINGMAN-BRUNDAGE [1989] unterteilt die Aktivitäten des Dienstleistungsanbieters in verschiedene Ebenen und grenzt diese durch „lines“ ab:

- Die „*line of interaction*“ grenzt die Kunden- von den Anbieteraktivitäten ab.
- Die „*line of visibility*“ separiert die aus Kundenseite wahrnehmbaren („Onstage-Aktivitäten“) von den nicht wahrnehmbaren Aktivitäten („Backstage-Aktivitäten“).
- Die „*line of internal interaction*“ unterscheidet die Aktivitäten des Kundenkontaktpersonals von den Aktivitäten anderer Mitarbeiter, den „Support-Aktivitäten“.
- Die „*line of implementation*“ differenziert die Durchführungsaktivitäten von den Planungs- und Kontrollaktivitäten („Facility-Aktivitäten“).

Die Erweiterung dieses Ansatzes durch die Einführung der „*line of order penetration*“ durch KLEINALTENKAMP [1999; 2000] trägt dem Gedanken Rechnung, dass die Aktivitäten eines Unternehmens im Rahmen der Leistungserstellung zum Teil mit, zum Teil ohne eine Integration des Kunden stattfinden. Daraus folgt eine Unterteilung in kundenin-

duzierte Aktivitäten des Leistungserstellungsprozesses und vom Anbieter autonom disponier- und durchführbaren Potenzialaktivitäten.

Der Vorteil des Service Blueprinting besteht in der Lane-Darstellung, durch die eine eindeutige Zuordnung einzelner Arbeitsschritte zu den ausführenden Handlungsträgern und Organisationseinheiten möglich ist. Dabei ist insbesondere die Kennzeichnung der Kundenaktivitäten hervorzuheben. Mit der Überführung dieser Darstellungsform auf die EPK beschäftigten sich unter anderem KNACKSTEDT, DAHLKE [2001]. Sie stellen heraus, dass durch die Lane-Darstellung die Strukturierung und Betrachtung von Prozessmodellen hinsichtlich der Problematik der Kundenintegrationsgestaltung möglich ist [Knackstedt, Dahlke 2001, S. 427]. Nach wie vor fehlt es jedoch an Überlegungen zur Berücksichtigung der Kundenintegrationsart. Ferner werden die Erfordernisse und Möglichkeiten, die sich aus der zwingenden Einbindung des Kunden bzw. seines Verfügungsobjekts ergeben, vernachlässigt. Nachfolgend wird ein Vorschlag präsentiert, der versucht diesen Mangel zu beheben.

3 Erweiterung der EPK zur Visualisierung der Kundenintegration

Nach ERNENPUTSCH [1986, S. 36ff] lassen sich einem Kunden drei Arten der Integration zurechnen:

– *Bedarfsbedingte Integration:* Die Dienstleistung kann ohne die Anwesenheit des externen Faktors erbracht werden, d. h. ohne deren Inanspruchnahme durch den Kunden und damit ohne wirtschaftliche Verwertung (z. B. Theatervorstellung).
– *Informationsbedingte Integration:* Die Dienstleistungserbringung erfordert eine informatorische Mitwirkung des Nachfragers (z. B. Beratungsleistung).
– *Produktionsbedingte/technische Integration:* Die Dienstleistung kann nur erbracht werden, wenn der Kunde oder sein Verfügungsobjekt physisch präsent sind (z. B. Massage).

In Anlehnung an diese Einteilung werden als Erweiterung für die EPK in Abb. 1 Konstrukte zu Visualisierung der Art der Kundenintegration vorgeschlagen. Diese berücksichtigt drei Darstellungen des Kunden, in denen durch die Stärke der Schattierung des Objekts die Art der Kundenintegration visualisiert wird. Zudem wird eine schattierte Hervorhebung der Funktionen empfohlen, an denen ein Kunde beteiligt ist.

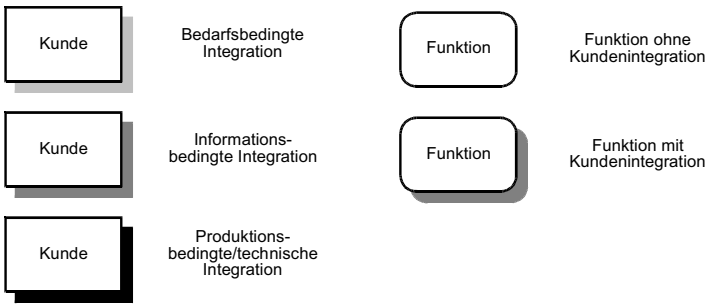

Abb. 1: Konstrukte zur Erweiterung der EPK

Die Verwendung der eingeführten Erweiterung ist in Abb. 2 anhand eines Beispielprozesses veranschaulicht. Die EPK beschreibt einen Dienstleistungserbringungsprozess in einer Autowerkstatt vom Auftreten eines Reparaturbedarfs bis zur Reparatur des Wagens. Der dargestellte Prozess ist weitgehend selbsterklärend.

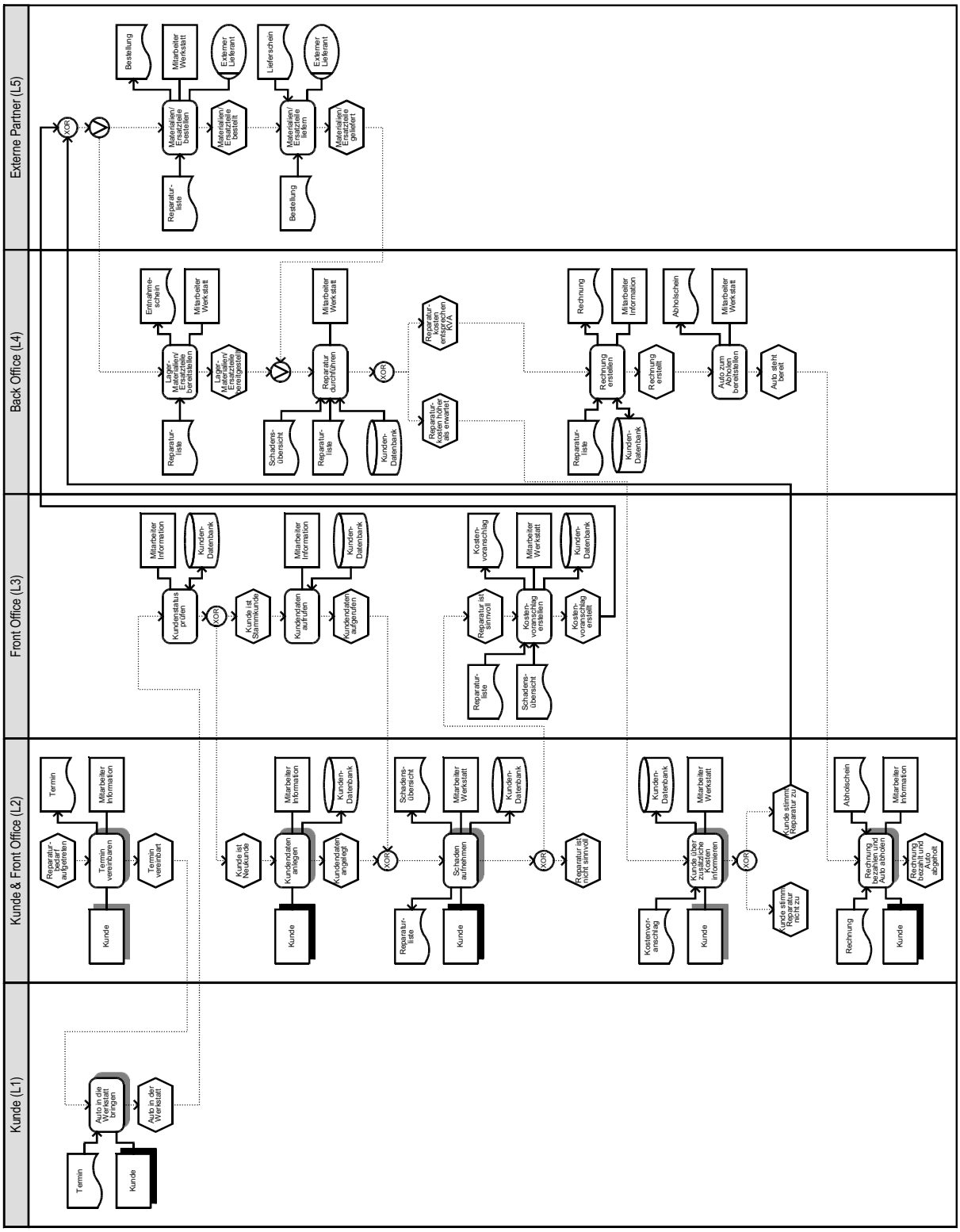

Abb. 2: Prozess der Kundenauftragsbearbeitung in einer Automobilwerkstatt

Der Mehrwert dieses Vorschlags liegt in der Möglichkeit, die Funktionen hinsichtlich der Art der Kundenintegration zu bewerten und daraus Anforderungen zur optimalen Funktionsausführung abzuleiten.

So induziert beispielsweise die *bedarfsbedingte* Integration ein hohes Maß an Standardisierungspotenzial, eine geringe Kontaktrate und marginalen informatorischen Bedarf. Eine Gefahr besteht allerdings in der mangelhaften Wahrnehmung von Kundenbedürfnissen, die sich von den ursprünglichen dienstleistungsinduzierenden Nachfragerbegehren unterscheiden. Die *produktionsbedingte/technische* Integration ist durch den persönlichen Kontakt des Anbieters mit dem Nachfrager gekennzeichnet. Aus diesem Umstand lassen sich z. B. Anforderungen an die soziale Kompetenz des Kundenkontaktpersonals ableiten. Für den Dienstleister birgt dies das Potenzial, individuell auf die Kundenbedürfnisse einzugehen. Die *informationsbedingte* Integration bewegt sich im Spannungsfeld der beiden erstgenannten Formen. Sie impliziert einen geringen Umfang an persönlicher Interaktion und damit Standardisierungspotenzial für die Abläufe. Allerdings sind etwa Forderungen an die Sozialkompetenz des Kundenkontaktpersonals eng mit der Möglichkeit des Leistungsanbieters verbunden, die benötigen Informationen standardisiert zu erfassen und den Kundenkontakt zu minimieren.

Diese Überlegungen berücksichtigen keine Aussagen über die Stärke der Kundenintegration bzgl. des Gesamtprozesses. Für die Prozessoptimierung wird für notwendige Entscheidungen jedoch die exakte Quantifizierung der Entscheidungsprämissen verlangt. Aus diesem Grund wird im Folgenden eine Quantifizierung der Stärke der Kundeneinbindung präsentiert.

4 Quantifizierung der Kundenintegration

Aus der vorgestellten EPK-Erweiterung zur Visualisierung der Kundenintegration in Dienstleistungserbringungsprozessen, insbesondere aus der Zuordnung der EPK-Funktionen zu den Lanes, lässt sich eine einfache Möglichkeit zur Quantifizierung der Kundenintegration ableiten.

Die Menge aller Ereignisgesteuerten Prozessketten sei mit EPK bezeichnet. Für einen Prozess $P \in EPK$ in EPK-Darstellung sei $F(P)$ die Menge aller Funktionen des Prozesses P. Ferner bezeichne

$$Lane := \{L_1, \ldots, L_n\}$$

die Menge der gegebenen Lanes. In diesem Fall gilt $n = 5$, wobei L_1 die Lane der Kundenaktivitäten, L_2 die Lane der Kunden- und Front-Office-Aktivitäten, L_3 die Lane der Front-Office-Aktivitäten, L_4 die Lane der Back-Office-Aktivitäten und L_5 die Lane der Aktivitäten externer Partner darstellt (vgl. Abb. 2). Ergänzend sei

$$L_P : F(P) \to Lane \; ; \; f \mapsto L_P(f)$$

diejenige Abbildung, die jeder Funktion $f \in F(P)$ eines gegebenen Prozesses $P \in EPK$ in EPK-Darstellung eindeutig eine Lane $L_P(f)$ zuordnet. Auf dieser Grundlage sei mit

EPK_{Lane} eine abkürzende Schreibweise für die Menge aller Ereignisgesteuerten Prozessketten in Lane-Darstellung eingeführt. Ist nun $P \in EPK_{Lane}$ gegeben, dann wird der Kundenintegrationsgrad $I(P)$ des Prozesses P definiert als der Quotient aus der Anzahl der Funktionen die den Lanes L_1, L_2 zugeordnet sind und der Anzahl aller Funktionen des Prozesses P, in Zeichen

$$I(P) := \frac{\left|\left\{f \in F(P) \mid L_P(f) \in \{L_1, L_2\}\right\}\right|}{\left|F(P)\right|}.$$

Mit der abkürzenden Schreibweise $n_i(P) := \left|\left\{f \in F(P) \mid L_P(f) = L_i\right\}\right|$ für die Anzahl der Funktionen in der Lane i eines gegebenen Prozesses $P \in EPK$ in EPK-Darstellung, ergibt sich

$$I(P) = \frac{n_1(P) + n_2(P)}{\left|F(P)\right|}.$$

Aus der Definition des Kundenintegrationsgrads folgt unmittelbar, dass der Wertebereich des Kundenintegrationsgrads $I(P)$ für einen Prozess $P \in EPK_{Lane}$ zwischen 0 und 1 liegt,

$$\forall P \in EPK_{Lane}: \; 0 \le I(P) \le 1.$$

Während ein Wert von $I(P) = 0$ einer Dienstleistungserbringung ohne jegliche Kundenintegration entspricht, bedeutet $I(P) = 1$, dass der Kunde in jede Aktivität des gegebenen Prozesses eingebunden ist. Für den Beispielprozess ergibt sich ein Kundenintegrationsgrad von $6/15 = 0,4$.

Ein möglicher Anwendungsbereich besteht neben der Messung des Kundenintegrationsgrads existierender Dienstleistungserbringungsprozesse nach der Einführung der Dienstleistung in der Bewertung des Integrationsniveaus für die Erbringung neu entwickelter Dienstleistungen im Rahmen des Service Engineering oder in der Klassifikation von Dienstleistungsprozessen anhand des Merkmals „Kundenintegrationsgrad“. Damit die Ergebnisse der Methode als Basis zur Verbesserung der Kundenzufriedenheit in Dienstleistungserbringungsprozessen verwendet werden können, bedarf es jedoch eines Regelwerks, aus dem in Abhängigkeit der errechneten Kundenintegrationsgrade konkrete Hinweise zur optimalen Funktionsausführung und Prozessgestaltung abgeleitet werden.

5 Zusammenfassung und zukünftige Forschungsfragen

Sowohl in praktischen als auch theoretischen Überlegungen zum Service Engineering wird die Bedeutung der Kundenintegration bei der Dienstleistungserbringung hervorgehoben. Gleichwohl mangelt es an Gestaltungsvorschlägen zur Visualisierung und Quantifizierung der Kundenintegration. Der vorliegende Beitrag erweitert die Methode der EPK um die genannten Aspekte. Eine Konkretisierung ihrer Zielsetzung sehen die Autoren in der Ableitung von Hinweisen zur Prozessgestaltung, die auf der Berechnung des Kundenintegrationsgrads gegebener Prozesse aufbauen. Eine notwendige Erweiterung des Ansatzes bestünde etwa in der Abschätzung des Integrationsaufwands sowohl aus Sicht des Unternehmens als auch aus Sicht des Kunden.

Literaturverzeichnis

Bruhn, M.: *Qualitätsmanagement für Dienstleistungen.* 4., verb. Aufl. Berlin [u.a.] : Springer, 2003

Bullinger, H.-J.: Dienstleistungsmärkte im Wandel – Herausforderung und Perspektiven. In: Bullinger, H.-J. (Hrsg.): *Dienstleistung der Zukunft.* Wiesbaden : Gabler [u.a.], 1995, S. 45-95

Bullinger, H.-J.; Meiren, T.: Service Engineering – Entwicklung und Gestaltung von Dienstleistungen. In: Bruhn, M.; Meffert, H. (Hrsg.): *Handbuch Dienstleistungsmanagement.* 2. Aufl. Wiesbaden : Gabler, 2001, S. 149-175

Bullinger, H.-J.; Scheer, A.-W.: Service Engineering – Entwicklung und Gestaltung innovativer Dienstleistungen. In: Bullinger, H.-J.; Scheer, A.-W. (Hrsg.): *Service Engineering.* Berlin [u.a.] : Springer, 2002, S. 3-17

Cooper, R. G.; Edgett, S. J.: *Product development for the service sector : Lessons from market leaders.* Cambridge, MA : Perseus, 1999

Corsten, H.: *Dienstleistungsmanagement.* 4. Aufl. München [u.a.] : Oldenbourg, 2001

Deutsches Institut für Normung e.V. (Hrsg.): *Service-Engineering.* Berlin : Beuth, 1998

Ernenputsch, M. A.: *Theoretische und empirische Untersuchungen zum Beschaffungsprozess von konsumtiven Dienstleistungen.* Bochum : Brockmeyer, 1986

Fähnrich, K.-P.: Service Engineering – Perspektiven einer noch jungen Fachdisziplin. In: *IM Information Management & Consulting* 13 (1998), S. 37-39

Fähnrich, K.-P.; Meiren, T.; Barth, T.: *Service engineering : Ergebnisse einer empirischen Studie zum Stand der Dienstleistungsentwicklung in Deutschland.* Stuttgart : Fraunhofer IAO, 1999

Fitzsimmons, J. A.; Fitzsimmons, M. J.: *New service development : Creating memorable experiences.* Thousand Oaks [u.a.] : Sage, 2000

Grieble, O.; Klein, R.; Scheer, A.-W.: Modellbasiertes Dienstleistungsmanagement. In: Scheer, A.-W. (Hrsg.): *Veröffentlichungen des Instituts für Wirtschaftsinformatik*, Nr. 171, Saarbrücken : Universität des Saarlandes, 2002

Haller, S.: *Dienstleistungsmanagement.* Wiesbaden : Gabler, 2001

Keller, G.; Nüttgens, M.; Scheer, A.-W.: Semantische Prozeßmodellierung auf der Grundlage „Ereignisgesteuerter Prozeßketten (EPK)". In: Scheer, A.-W. (Hrsg.): *Veröffentlichungen des Instituts für Wirtschaftsinformatik*, Nr. 89, Saarbrücken : Universität des Saarlandes, 1992

Kingman-Brundage, J.: The ABCs of Service System Blueprinting. In: Bitner, M. J.; Crosby, L. A. (Hrsg.): *Designing a winning service strategy.* Chicago, IL : American Marketing Association, 1989

Kleinaltenkamp, M.: Service-Blueprinting – Nicht ohne einen Kunden: Ein Instrument zur Steigerung der Effektivität und der Effizienz von Dienstleistungsprozessen. In: *Technischer Vertrieb* 1 (1999), Nr. 2, S. 33-39

Kleinaltenkamp, M.: Blueprinting – Grundlage des Managements von Dienstleistungsunternehmen. In: Woratschek, H. (Hrsg.): *Neue Aspekte des Dienstleistungsmarketing.* Wiesbaden : Dt. Univ.-Verl. [u.a.], 2000, S. 3-28

Knackstedt, R.; Dahlke, B.: Kundenintegration als Perspektive der Prozessmodellierung. In: Bauknecht, K.; Brauer, W.; Mück, T. A. (Hrsg.): *Informatik 2001.* Konstanz : UVK-Verlags-Ges., 2001, S. 418-428

Ramaswamy, R.: *Design and management of service processes : Keeping customers for life.* Reading, MA [u.a.] : Addison-Wesley, 1996

Shostack, L. G.: How to design a service. In: Donnelly, J. H.; George, W. R. (Hrsg.): *Marketing of services.* Chicago : American Marketing Association, 1981, S. 221-229

Shostack, L. G.: How to Design a Service. In: *European Journal of Marketing* 16 (1982), Nr. 1, S. 49-63

Thomas, O.; Scheer, A.-W.: Ein modellgestützter Ansatz zum Customizing von Dienstleistungsinformationssystemen. In: Becker, J.; Knackstedt, R. (Hrsg.): *Referenzmodellierung 2002.* Münster : Westfälische Wilhelms-Universität, 2002, S. 81-117