Transformation ereignisgesteuerter Prozeßketten in Workflowbeschreibungen im XPDL-Format [1]

Daniel Beer, Jörg Dümmler[2] und Gudula Rünger
Technische Universität Chemnitz
Fakultät für Informatik
Straße der Nationen 62, 09107 Chemnitz

Zusammenfassung

Aufgrund des stetig zunehmenden Kostendrucks geht ein Trend in Richtung der rechnergestützten Abarbeitung von betrieblichen Geschäftsprozessen und behördlichen Verwaltungsvorgängen in Form von Workflows. Liegen die abzuarbeitenden Vorgänge bereits als Geschäftsprozeßmodell vor, kann durch eine Transformation in ein Workflowmodell eine komplette Neumodellierung vermieden werden. Dieser Bericht beschreibt ein Übersetzungswerkzeug, das eine derartige Transformation halbautomatisch durchführen kann. Als Ausgangspunkt dienen Geschäftsprozeßmodelle basierend auf ereignisgesteuerten Prozeßketten, die über mehrere Transformationsschritte in eine im XPDL Format kodierte Workflowbeschreibung überführt wird. Die einzelnen Transformationsschritte werden an einem konkreten Anwendungsfall aus dem Bereich des E-Governments verdeutlicht und die Bedienung des Werkzeugs, die über eine graphische Benutzeroberfläche erfolgt, erläutert.

[1] Das Projekt RAfEG wurde mit Mitteln des Bundesministeriums für Bildung und Forschung (BMBF) im Rahmen der Forschungsoffensive „Software Engineering 2006“ unter dem Förderkennzeichen 01 IS C07 C gefördert. Das Projekt setzte sich zusammen aus dem Institut für Wirtschaftsinformatik (IWi) im Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI), der NHConsult GmbH, der Professur Praktische Informatik der Technischen Universität Chemnitz und dem Regierungspräsidium Leipzig. Weitere Informationen sind auf der Projekthomepage `http://www.rafeg.de` zu finden.

[2] Ansprechpartner unter Email `joerg.duemmler@informatik.tu-chemnitz.de`

1. Einleitung

Die Modellierung von Arbeitsprozessen in Firmen oder Behörden lassen sich in zwei verschiedene Herangehensweisen mit unterschiedlicher Zielstellung unterteilen. Auf der einen Seite steht die Erstellung von Geschäftsprozeßmodellen, die zur Dokumentation der Abläufe und zur Analyse im Rahmen der Prozeßoptimierung eingesetzt werden. Weitere Vorteile der Geschäftsprozeßmodellierung liegen unter anderem in der Erhöhung der Transparenz für die beteiligten Mitarbeiter, der besseren Kontrolle über Daten und Dokumente und der Verbesserung der Flexibilität, bspw. durch den Austausch von Teilprozessen. Eine rechnergestützte Abarbeitung dieser Modelle ist jedoch meistens nicht vorgesehen. Die Darstellung von Geschäftsprozeßmodellen erfolgt häufig durch ereignisgesteuerte Prozeßketten (EPK), die von einer Reihe von Modellierungswerkzeugen, unter anderem dem ARIS-Toolset[18], Nautilus[16] oder Semtalk[19], unterstützt werden.

Auf der anderen Seite steht die Modellierung von Prozessen in Form von Workflows, die mit Hilfe eines Workflowmanagementsystems(WfMS) rechnergestützt abgearbeitet werden können. Das WfMS verwaltet die anstehenden Aufgaben sowie die Dokumente und Daten des zugrundeliegenden Prozesses und weist den beteiligten Mitarbeitern die zu bearbeitenden Teilaufgaben zu. Einzelne Teilaufgaben können häufig direkt durch das WfMS ohne Nutzerinteraktion ausgeführt werden, wobei auch die Integration von externen Anwendungen möglich ist. Der Einsatz von WfMS kann zu einer Verringerung der Bearbeitungsdauer und damit einer Kostenredukton beitragen. Darüberhinaus wird eine bessere Kontrolle über den Arbeitsablauf erreicht, da jederzeit der Zustand aller Prozesse des Systems abgefragt werden kann. Zur Beschreibung von Workflows wurden viele unterschiedliche Sprachen entwickelt. Besonders hervorzuheben, da sie standardisiert sind und von einer Vielzahl verschiedener WfMS unterstützt werden, sind die Business Process Execution Language (BPEL)[4] und die XML Process Definition Language(XPDL)[5]. BPEL wurde im Jahr 2002 von den Firmen IBM, BEA und Microsoft eingeführt und liegt aktuell in der Version 1.1 vor. Unterstützt wird BPEL bspw. von dem Oracle BPEL Process Manager[17] oder dem quelloffenen ActiveBPEL[1]. XPDL wurde durch die Workflowmanagement-Coalition, einem herstellerunabhängigen Gremium, entwickelt und existiert derzeit in der Version 2.0. Die quelloffenen WfMS Enhydra Shark[8] und WfMOpen[23] basieren in der aktuellen Version noch auf einem Workflowmodell nach XPDL Version 1.0.

Durch die unterschiedlichen Ziele der Geschäftsprozeßmodellierung und der Workflowmodellierung entsteht eine Lücke, die sich auch in den unterschiedlichen verwendeten Beschreibungsmodellen ausdrückt. Einen Versuch, diese Lücke zu überwinden, wurde in [12], wo ein Übersetzungsmodul für ereignisgesteuerte Prozeßketten nach BPEL für den Einsatz in Nautilus entwickelt wurde, beschrieben. In dem vorliegenden Bericht wird das Übersetzungswerkzeug *ConvEX* (Converter from Event-driven process chains to XPDL) zur halbautomatischen Transformation von ereignisgesteuerten Prozeßketten nach XPDL vorgestellt. Die Übersetzung innerhalb von ConvEX wird in mehreren Transformationsphasen durchgeführt. Nutzereingriffe in den Übersetzungsvorgang, die bei fehlenden Informationen oder Inkonsistenzen in den Eingabemodellen eines Geschäftsprozesses notwendig sind, erfolgen über eine graphische Benutzeroberfläche. ConvEX wurde innerhalb des Verbundprojekts „Referenzarchitektur für E-Government“(RAfEG)[3] entwickelt und wird zur Transformation von Geschäftsprozeßmodellen, die von Verwaltungsabläufen innerhalb von Behörden erstellt wurden, in äquivalente Workflowbeschreibungen eingesetzt. Zusätzlich kann ConvEX sogenannte Webformularbeschreibungen erzeugen, die die erforderliche Interaktion der RAfEG-Benutzer mit dem WfMS zum Zeitpunkt der Abarbeitung der übersetzten Prozesse beschreiben. Die Anwendung von ConvEX wird anhand eines Beispielmodells aus dem Projekt RAfEG verdeutlicht.

Dieser Bericht ist wie folgt gegliedert. In Kapitel 2 werden die Grundlagen und der

Aufbau von Workflowbeschreibungen im XPDL-Format erläutert. Das Modell der ereignisgesteuerten Prozeßkette wird in Kapitel 3 beschrieben. Das Hauptaugenmerk dieser beiden Abschnitte liegt auf der Darlegung der zugrundeliegenden Ideen und der von RAfEG genutzten Möglichkeiten. Die technische Realisierung mit Erklärung der einzelnen Transformationsschritte ist in Kapitel 4 zu finden. Kapitel 5 beschreibt die Bedienung der graphischen Benutzeroberfläche von ConvEX während Kapitel 6 mit einer Zusammenfassung abschließt.

2. Aufbau von Workflows in der Workflowbeschreibungssprache XPDL

Dieser Abschnitt gibt einen kurzen Überblick des Workflowmodells nach XPDL Version 1.0. Der Fokus liegt dabei auf den für ConvEX relevanten Modellteilen. Eine umfassendere Beschreibung ist in der Spezifikation[5] durch die Workflowmanagement-Coalition enthalten.

Den Hauptteil einer Workflowdefinition in XPDL bildet die Spezifikation einer Menge von Workflowprozessen, die als eigenständiger Teil eines Arbeitsablaufs betrachtet werden können. Innerhalb eines Prozesses können andere definierte Prozesse als Unterprozeß eingebunden werden, wodurch ein hierarchisches Modell entsteht. Ein Workflowprozeß ist ein gerichteter Graph, dessen Knoten Aufgaben bzw. Aktivitäten entsprechen und dessen Kanten die Richtung der Workflowkontrolle vorgeben. In einer Workflowbeschreibung in XPDL können unterschiedliche Aufgabentypen als Knoten des Workflowgraphen vorkommen, z.B. automatische, manuelle, Unterprozeß- oder Route-Aktivitäten. Automatische Aktivitäten werden für den Anwender unsichtbar innerhalb des WfMS abgearbeitet. Dafür kann in XPDL direkt der Quelltext eines auszuführenden Programmstücks oder eine Schnittstellenbeschreibung zu einer externen Applikation angegeben werden. Diese Definitionen erfolgen in XPDL als Werkzeug (Elementname `tool`) global außerhalb der Workflowprozeßbeschreibungen. Manuelle Aktivitäten werden von den Nutzern des Systems, z.B. den Sachbearbeitern einer Behörde, ausgeführt. Die Beendigung und das Ergebnis dieser Aufgaben müssen dem System durch die Nutzer mitgeteilt werden. Da im Allgemeinen nicht jede Aufgabe durch jeden Bearbeiter ausgeführt werden kann, wird in XPDL jedem Aufgabenknoten ein Benutzer aus einer globalen Benutzerliste zugewiesen. Die Einträge dieser Benutzerliste müssen nicht einzelnen Personen entsprechen, sondern können auch Gruppen oder ganze Institutionen darstellen. Ein konkretes Nutzermanagement, also bspw. die Zuordnung von am System angemeldeten Anwendern zu Einträgen der Benutzerliste, wird von XPDL nicht vorgegeben und muß durch das WfMS erfolgen. Unterprozeßaktivitäten führen einen anderen Workflowprozeß als Kindprozeß aus. XPDL stellt die Möglichkeiten der synchronen Ausführung, d.h. der Elternprozeß wird erst fortgesetzt, wenn der Kindprozeß beendet ist, und der asynchronen Ausführung, d.h. Eltern- und Kindprozeß sind gleichzeitig aktiv, zur Verfügung. Zusätzlich bietet XPDL die Möglichkeit, zu jedem Knoten eine beliebige Menge nutzerdefinierter Annotationen zu speichern.

Die Abarbeitung eines Workflowprozesses beginnt mit der simultanen Übergabe der Kontrolle an alle Knoten ohne eingehende Kanten. Nach Beendigung einer mit einem Knoten assoziierten Aufgabe wird der Kontrollfluß auf die ausgehenden Kanten übertragen. Die Kanten des Workflowgraphen können mit einer Bedingung annotiert sein, d.h. der Kontrollfluß wird nur weitergegeben, wenn die Bedingung wahr ist. Da diese Bedingungen innerhalb des WfMS ausgewertet werden, ist eine spezielle automatisch auswertbare Form erforderlich. Kanten ohne Bedingung geben den Kontrollfluß in jedem Fall weiter. Besitzt ein Knoten des Workflowgraphen mehr als eine eingehende Kante, so muß für den Knoten

eine geeignete Vorbedingung angegeben werden, die festlegt, wie die eingehenden Kanten synchronisiert werden. Zur Auswahl stehen in XPDL die Operationen AND und XOR. Eine AND-Vorbedingung gibt an, daß der Knoten aktiviert wird, wenn der Kontrollfluß alle eingehenden Kanten passiert hat. Bei einer XOR-Vorbedingung ist eine aktive eingehende Kante zur Aktivierung der zugehörigen Aufgabe ausreichend. Analog muß eine Nachbedingung angegeben werden, falls ein Aktivitätsknoten mehr als eine ausgehende Kante besitzt. Diese Nachbedingung legt fest, wie der Kontrollfluß nach Beendigung der Aufgabe auf die ausgehenden Kanten verteilt wird. Im Fall einer AND-Nachbedingung werden die Kantenbedingungen aller ausgehenden Kanten überprüft und alle Kanten, deren Bedingung wahr ergibt (oder die keine Bedingung besitzen), werden aktiviert. Bei einer XOR-Nachbedingung werden die ausgehenden Kanten in der Reihenfolge ihrer Definition überprüft, bis die Bedingung einer Kante wahr ergibt. Zusätzlich kann nach einer derartigen Nachbedingung maximal eine Kante die spezielle Bedingung OTHERWISE haben, die den Standardfall darstellt, falls die Kantenbedingungen aller anderen ausgehenden Kanten falsch sind. In jedem Fall wird von einer XOR-Nachbedingung nur eine der ausgehenden Kanten aktiviert. Route-Aktivitäten sind Aufgabenknoten des Workflowgraphen, die nur eine Vorbedingung oder eine Nachbedingung besitzen, aber nicht mit einer konkreten, auszuführenden Aufgabe assoziiert sind.

Zum Speichern von Informationen über einen Workflowprozeß können in XPDL Prozeßvariablen deklariert werden. Die Deklaration unter Angabe eines Namens und eines Datentyps kann lokal in einer Workflowprozeßbeschreibung oder global erfolgen. Als Datentypen stehen einfache Datentypen, wie z.B. ganzzahlige oder boolesche Werte, sowie komplexe Typen, wie z.B. Felder oder Strukturen, zur Verfügung. Das Verändern der Prozeßvariablen kann intern im WfMS durch automatische Aktivitäten erfolgen. Eine andere Möglichkeit ist das Verändern der Variablen über eine entsprechende Schnittstelle des WfMS, was bspw. bei manuellen Aufgaben genutzt werden kann. Prozeßvariablen können auch als Übergabeparameter oder Rückgabewerte von Unterprozessen eingesetzt werden.

Workflowbeschreibungen im RAfEG-System Die im RAfEG-Projekt entwickelte Referenzsoftwarearchitektur sieht die Austauschbarkeit von verschiedenen Komponenten des Systems, z.B. des WfMS, vor. Für die prototypische Implementierung des RAfEG-Systems wurden die beiden als Open Source Projekte entwickelten WfMS Enhydra Shark[8] und WfMOpen[23] angebunden, die alternativ genutzt werden können. Beide Systeme basieren derzeit auf dem hier beschriebenen Workflowmodell nach XPDL Version 1.0.

Anwender des RAfEG-Systems melden sich über die RAfEG-Oberfläche am System an. Dabei wird ihnen von der RAfEG-Nutzerverwaltung eine bestimmte Rolle, z.B. der Name einer Behörde, zugeordnet. Diese Rolle entspricht einem Eintrag aus der globalen XPDL-Benutzerliste. Der Anwender kann sich von der Oberfläche eine Liste aller anstehenden Aufgaben, die von seiner Rolle ausgeführt werden können, anzeigen lassen. Anstehende Aufgaben sind manuelle Aktivitäten, die der Kontrollfluß eines Workflowprozesses erreicht hat. Nach Ausführung einer Aufgabe teilt der Anwender dem System die Beendigung und ggf. das Ergebnis der Aufgabe mit. Dies erfolgt wiederum über die RAfEG-Oberfläche durch sogenannte Webformulare.

In den vom RAfEG-System verwendeten Workflowbeschreibungen werden in der globalen Nutzerliste die möglichen Rollen für die Anwender des Systems sowie eine zusätzliche System-Rolle angegeben. Der System-Rolle werden alle automatischen Aktivitäten, Unterprozeßaktivitäten und Route-Aktivitäten zugeordnet. Die Ausführung automatischer Aktivitäten erfolgt durch kurze JavaScript-Codestücke, die direkt in die XPDL-Beschreibung eingebettet werden. Die konkrete Syntax hängt dabei vom verwendeten WfMS (Enhydra Shark oder WfMOpen) ab. Sind für eine manuelle Aufgabe mehrere Ergebnisse möglich, so wird nach Beendigung der Aufgabe das Ergebnis in einer Prozeßvariablen gespeichert.

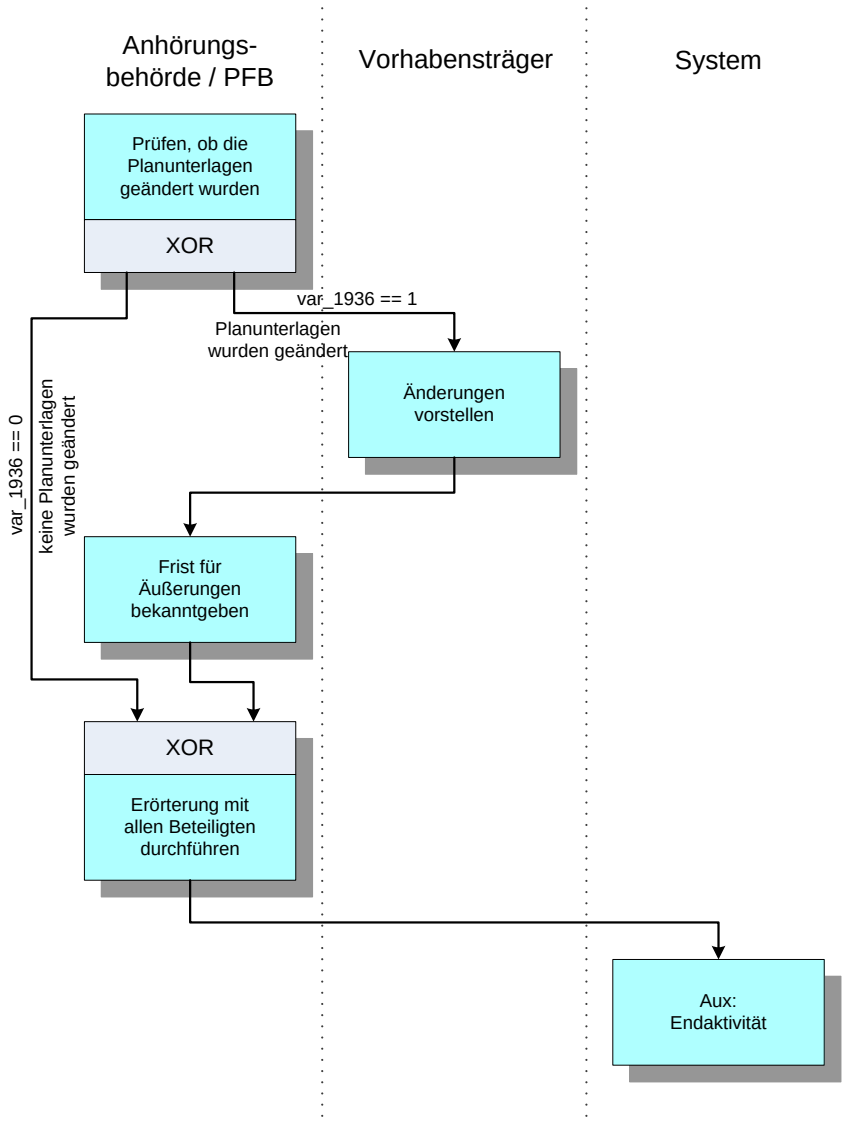

Abbildung 1: Graphische Darstellung des von RAfEG verwendeten Workflowprozesses „Erörterungstermin mit allen Beteiligten durchführen“.

Prozeßvariablen sind von einem Aufzählungstyp, dessen Wertebereich von der Anzahl möglicher Ergebnisse abhängt. Die Prozeßvariablen werden zur Auswertung der Bedingungen an den Kanten des Workflowgraphen genutzt und können somit den weiteren Verlauf des Workflowprozesses beeinflussen.

Für die Abarbeitung der in RAfEG eingesetzten Workflows werden auch Informationen benötigt, für die keine direkte Entsprechung in XPDL existiert. Dies betrifft bspw. die für eine Aufgabe notwendigen Dateien, die vom RAfEG-Dokumentenmanagementsystem[13] verwaltet werden, oder zu beachtende rechtliche Vorschriften. Diese Informationen werden als zusätzliche nutzerdefinierte Annotationen der Aktivitätsknoten der XPDL-Beschreibung abgespeichert.

Abbildung 1 zeigt einen in RAfEG eingesetzten Workflowprozeß. Die graphische Darstellung der Aktivitätsknoten erfolgt sortiert nach der zugewiesenen Bearbeiterrolle. Die manuelle Aufgabe „Prüfen, ob die Planunterlagen geändert wurden“, die von einem Mitarbeiter der Anhörungsbehörde ausgeführt wird, besitzt eine XOR-Nachbedingung mit zwei ausgehenden Kanten. Daraus folgt, daß diese Aufgabe zwei mögliche Ergebnisse haben kann. Das Ergebnis wird dem System vom Bearbeiter mitgeteilt und in einer Prozeßvariablen, hier `var_1936`, gespeichert. Die Endaktivität ist eine durch ConvEX hinzugefügte automatische Aktivität. Sie dient dazu, das Ergebnis des Prozesses zu speichern und an einen übergeordneten Prozeß weiterzugeben, s. auch Abschnitt 4.3.

3. Aufbau von ereignisgesteuerten Prozeßketten

Geschäftsprozesse, die Arbeitsabläufe in Unternehmen oder Verwaltungen beschreiben, können durch ereignisgesteuerte Prozeßketten (EPK)[9] modelliert und analysiert werden. EPKs können als gerichtete, annotierte Graphen aufgefaßt werden, wobei drei unterschiedliche Knotentypen, die *Ereignisknoten*, die *Funktionsknoten* und die *Konnektorknoten*, verwendet werden.

Ereignisse sind die passiven Elemente einer EPK. Sie stellen einen Zustand des modellierten Geschäftsprozesses dar, der unter bestimmten Bedingungen eintreten kann. Durch das Eintreten eines Ereignisses können Funktionen ausgelöst werden, die als Resultat wieder Ereignisse liefern.

Funktionen stellen die aktiven Komponenten einer EPK dar, da nur sie Daten verändern können. Mit einem Funktionsknoten ist das Durchführen einer Aufgabe assoziiert. Eine Funktion kann mit einem kompletten Geschäftsprozeß, der wiederum als EPK modelliert ist, hinterlegt sein. In diesem Fall ist die Aufgabe der Funktion die Abarbeitung des hinterlegten Prozesses. Funktionen werden durch ein oder mehrere Ereignisse aktiviert und erzeugen ein oder mehrere Folgeereignisse.

Ereignisknoten und Funktionsknoten dürfen nur maximal eine eingehende und maximal eine ausgehende Kante besitzen. Zum Aufteilen oder Zusammenführen des Kontrollflusses, der durch die Kanten einer EPK vorgegeben wird, dient der dritte Knotentyp, der Konnektor. Konnektoren können entweder mehrere eingehende und genau eine ausgehende Kante (Join-Konnektor) oder genau eine eingehende und mehrere ausgehende Kanten (Split-Konnektor) besitzen. Jeder Konnektorknoten ist mit einer Verknüpfungsoperation assoziiert, die entscheidet, wie der Kontrollfluß der eingehende(n) Kante(n) auf die ausgehende(n) Kante(n) verteilt wird. Als Verknüpfungsoperationen stehen die Antivalenz (XOR, genau ein Fall tritt ein), die Disjunktion (OR, mindestens ein Fall tritt ein) und die Konjunktion (AND, alle Fälle müssen eintreten) zur Verfügung.

EPKs beginnen jeweils mit mindestens einem Ereignis ohne eingehende Kante (im folgenden Startereignis genannt) und werden mit mindestens einem Ereignis ohne ausgehende Kante (im folgenden Endereignis genannt) beendet. Startereignisse repräsentieren Vorbedingungen der EPK, während durch Endereignisse mögliche Ergebnisse des mit der EPK assoziierten Geschäftsprozesses dargestellt werden.

Werden zusätzlich Informationen über Organisationseinheiten oder Datenobjekte modelliert, spricht man von einer erweiterten ereignisgesteuerten Prozeßkette (eEPK)[21]. Organisationseinheiten und Datenobjekte werden dabei als Annotationen von Funktionsknoten einer EPK dargestellt. Durch die Organisationseinheiten kann der ausführende Anwender einer Funktion ermittelt werden, während Datenobjekte benötigte oder von einer Funktion veränderte Daten, z.B. Dateien, darstellen.

Ereignisgesteuerte Prozeßketten im RAfEG-Projekt Im RAfEG-Projekt werden eEPKs eingesetzt, um Verwaltungsabläufe innerhalb von Behörden zu modellieren. Als Modellierungswerkzeug wird das ARIS-Toolset[18] der IDS Scheer AG verwendet. Als Anwendungsbeispiel wird das Planfeststellungsverfahren betrachtet, das als Referenzmodell und konkret angepaßt an das Regierungspräsidium Leipzig durch das Institut für Wirtschaftsinformatik modelliert wurde[22]. Dabei entstand ein hierarchisches 5-Schichten Modell, das ausgehend vom Planfeststellungsverfahren als eine einzige Aufgabe (modelliert als Schicht 1) bis zu den Einzelaufgaben für die Bearbeiter (modelliert als Schicht 5) dargestellt ist. Funktionen einer EPK in einer Schicht sind mit kompletten EPKs der nächsthöheren Schicht hinterlegt. Der Vorteil einer derartigen Modellierung liegt in der einfachen Austauschbarkeit und der Möglichkeit der externen Abarbeitung von Teilprozessen.

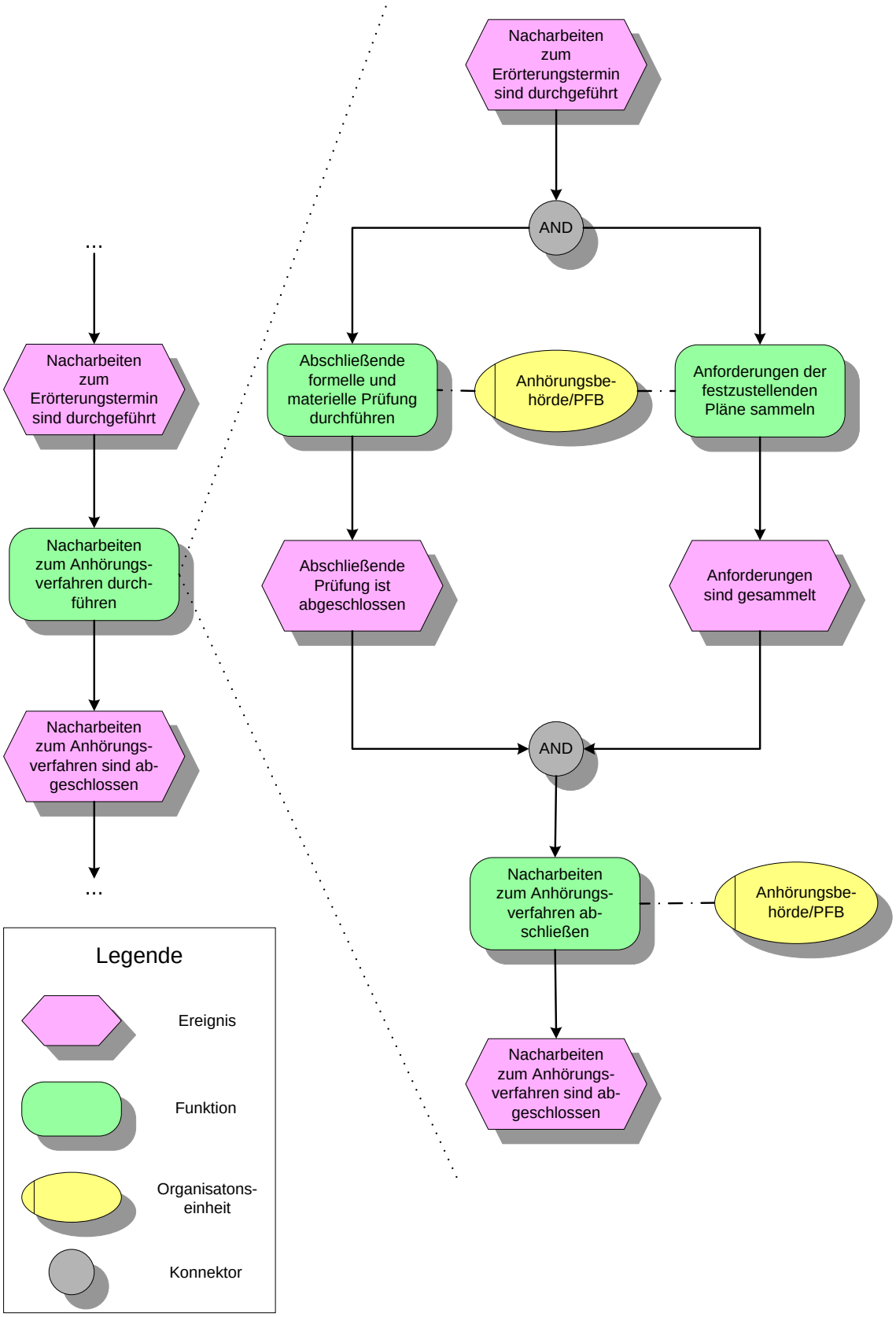

Abbildung 2: Beispiel einer erweiterten ereignisgesteuerten Prozeßkette. Die linke Seite zeigt einen Ausschnitt aus dem Verwaltungsvorgang „Anhörungsverfahren durchführen“. Auf der rechten Seite ist der Unterprozeß „Nacharbeiten zum Anhörungsverfahren durchführen“, der auf der linken Seite als eine Funktion repräsentiert wurde, detailliert durch eine eigene EPK dargestellt.

Abbildung 2 zeigt einen Ausschnitt des für das Regierungspräsidium Leipzig erstellten Modells des Planfeststellungsverfahrens. Auf der linken Seite ist ein kurzer Ausschnitt aus dem Vorgang „Anhörungsverfahren durchführen“, einer EPK der Schicht 3, dargestellt. Die rechte Seite zeigt die detaillierte Aufschlüsselung einer Funktion der Schicht 3 durch eine komplette EPK der Schicht 4, hier der Prozeß „Nacharbeiten zum Anhörungsverfahren durchführen“. Diese EPK hat genau ein Startereignis, „Nacharbeiten zum Erörterungstermin sind durchgeführt“, und ein Endereignis, „Nacharbeiten zum Anhörungsverfahren sind abgeschlossen“. Vor- und Nachbedingung der Funktion „Nacharbeiten zum Anhörungsverfahren durchführen“ in der übergeordneten EPK entsprechen genau dem Start- bzw. Endereignis in der untergeordneten EPK. In der untergeordneten EPK sind direkt ausführende Organisationseinheiten modelliert, so daß hier keine weitere hierarchische Aufgliederung erfolgt.

Als Datenformat liegen die erstellten Modelle zunächst in einem internen Format des ARIS-Toolsets vor, können aber in die ARIS Markup Language (AML)[2] exportiert werden. AML ist eine XML-basierte Sprache, die auch andere Modelltypen außer EPKs darstellen kann. Je nach verwendetem Modell können in AML unterschiedliche Objekttypen auftreten.

Die prototypische Implementierung des RAfEG-Systems wurde an der Professur Praktische Informatik der TU Chemnitz als Open Source Projekt durchgeführt. Einzelne Teilsysteme, z.B. das WfMS, wurden durch die Anpassung freier, quelloffener Software realisiert. Da AML eher als proprietäres Format anzusehen ist, für dessen Bearbeitung keine freie Software zur Verfügung steht, wurde als Ausgangspunkt zur Darstellung der modellierten EPKs die EPC Markup Language (EPML)[15] gewählt. EPML wurde als werkzeugunabhängiges Austauschformat für EPKs konzipiert, das als Zwischenformat für verschiedene Modellierungswerkzeuge dienen kann. Von EPML werden alle EPK-Elemente, die von den für RAfEG erstellten Modelle genutzt werden, unterstützt. Durch die Verwendung von EPML in RAfEG wird also eine weitgehende Unabhängigkeit vom ARIS-Toolset erreicht, und weitere Modellierungswerkzeuge können für die Erstellung der Eingabemodelle verwendet werden. Weitere Vorteile von EPML liegen in der gegenüber AML umfangreicheren Dokumentation und einem einfacheren, strukturierteren Aufbau des Formats, das eine wesentlich kompaktere Speicherung von EPK-Modellen erlaubt. Die Umwandlung der erstellten EPK-Modelle von AML nach EPML kann mit Hilfe eines Transformationsskripts[14] auf Basis von XSLT durchgeführt werden. Dieses Skript ist frei verfügbar und kann mit einem beliebigen XSLT-Transformationswerkzeug ausgeführt werden. Dazu ist jedoch noch die AML-Datentypdefinition, die in einer Datei namens `ARIS-Export.dtd` beschrieben ist und jeder Installation des ARIS-Toolsets beiliegt, erforderlich.

4. Technische Realisierung des Transformationsprozesses

Die Struktur der modellierten EPKs und der Workflowbeschreibungen in XPDL ist ähnlich. Als Grundstruktur der Darstellung wird in beiden Fällen ein hierarchischer, gerichteter Graph genutzt. Unterschiede bestehen vor allem in der semantischen Bedeutung der verschiedenen Knotentypen und den Möglichkeiten, den Kontrollfluß zu steuern. Während in EPKs Entscheidungen über Ereignisse, die unter bestimmten Umständen eintreten können, getroffen werden, ist in den XPDL-Workflowbeschreibungen die Angabe von Kantenbedingungen notwendig, die automatisch von einem WfMS auswertbar sein müssen. Im RAfEG-System werden zu diesem Zweck Prozeßvariablen genutzt, die abhängig vom Zustand der Abarbeitung eines Workflowprozesses unterschiedliche Werte annehmen können. Prozeßvariablen werden auch dafür genutzt, Ergebnisse von manuellen Aktivitäten zu speichern, siehe auch Abschnitt 2. Diese Ergebnisse werden vom Anwender

mit Hilfe der RAfEG-Oberfläche über Webformulare, die die Auswahl zwischen mehreren möglichen Aktivitätsergebnissen gestatten, mitgeteilt. Die Beschreibungen zur automatischen Generierung der Webformulare werden abschließend im Übersetzungsprozeß erzeugt.

Der Transformationsprozeß enthält die folgenden Phasen, die nacheinander ausgeführt werden:

- die EPK-Analysephase, die in Abschnitt 4.1 beschrieben wird,
- die Entfernungsphase der Ereignisknoten aus den EPKs, die in Abschnitt 4.2 vorgestellt wird,
- die Übersetzungsphase der Graphstrukturen, in der EPK-Elemente in entsprechende Äquivalente einer XPDL-Beschreibung übersetzt werden (s. Abschnitt 4.3),
- die Erzeugungsphase der Prozeßvariablen und der Kantenbedingungen, die in Abschnitt 4.4 detailliert dargestellt wird und
- die Webformulargenerierungsphase, in der die Webformularbeschreibungen, wie in Abschnitt 4.5 erläutert, erzeugt werden.

4.1 EPK-Analysephase

Die EPK-Analysephase beinhaltet das Einlesen der EPML-Quelldatei, was durch einen auf der Simple API for XML (SAX)[20] basierenden XML-Parser realisiert wird. In einer entsprechenden Quelldatei können mehrere EPKs enthalten sein, die Teilprozesse einer übergeordneten EPK sind. Das Ergebnis des Parsevorgangs ist eine interne Repräsentation einer hierarchischen EPK in Form von gerichteten Graphen. In den nachfolgenden Transformationsschritten wird für jede eingelesene EPK ein Workflowprozeß konstruiert.

Die Spezifikation der Anwender, die die Funktionen einer EPK ausführen können, erfolgt in EPML lokal, so daß identische Nutzer möglicherweise mehrfach modelliert werden. Ein Beispiel hierfür ist in Abbildung 2 dargestellt, in der der Anwender „Anhörungsbehörde/PFB“ zweifach definiert ist. Workflowbeschreibungen in XPDL verwenden dagegen eine globale Benutzerliste, die jedem Benutzer eine eindeutige Identifikationsnummer zuordnet. Aus diesem Grund wird in der EPK-Analysephase eine Vorstufe dieser Benutzerliste, die *Nutzernamentabelle*, aufgebaut. Die Nutzernamentabelle ordnet jedem Benutzernamen, der beim Einlesen der EPK gefunden wird, eine Identifikationsnummer zu. Dabei werden identische Benutzernamen auf dieselbe Identifikationsnummer abgebildet, unterschiedliche Benutzernamen erhalten verschiedene Identifikationsnummern. Zusätzlich enthält die Nutzernamentabelle einen System-Benutzer, dem später alle vom WfMS durchzuführenden Aufgaben zugewiesen werden.

Die Definition der Konnektorknoten einer EPK in EPML gibt den Typ, also ob der Konnektor den Kontrollfluß aufteilt oder zusammenführt, nicht explizit an. Dieser Typ wird während des Einlesens der Quelldatei implizit über die Anzahl der eingehenden bzw. der ausgehenden Kanten ermittelt und explizit am Konnektorknoten annotiert.

Das Ergebnis der Analysephase ist somit eine Menge gerichteter Graphen, in denen alle EPK-Elemente vorkommen können und die Konnektorknotentypen explizit annotiert sind, sowie die Nutzernamentabelle, die jedem vorkommenden Nutzernamen eine eindeutige Identifikationsnummer zuordnet. Die nachfolgenden Übersetzungsphasen werden für jeden EPK-Graphen getrennt durchlaufen.

4.2 Entfernungsphase der EPK-Ereignisknoten

In einer EPK können drei verschiedene Knotentypen auftreten: Funktions-, Ereignis- und Konnektorknoten. Funktions- und Konnektorknoten können direkt in semantisch äquivalente Aktivitäten in der Workflowbeschreibungssprache XPDL übersetzt werden. Für Funktionsknoten kommen manuelle, automatische und Unterprozeßaktivitäten in Betracht, während Konnektorknoten durch Route-Aktivitäten repräsentiert werden können. Für Ereignisknoten existiert dagegen keine natürliche Entsprechung in XPDL. Abhängig vom Ort des Auftretens kann ein Ereignisknoten jedoch workflowrelevante Informationen, wie z.B. das Ergebnis einer Funktion, enthalten. Ziel der Entfernungsphase des Übersetzungsprozesses ist das Entfernen aller Ereignisknoten aus einer EPK. Noch benötigte Informationen bleiben als Annotationen an den Kanten oder als Hilfsfunktionen, die zusätzlich generiert werden, erhalten.

Das Entfernen der Ereignisknoten wird nach der folgenden Fallunterscheidung durchgeführt.

- Startereignisse, denen ein Konnektorknoten folgt, werden durch eine spezielle Funktion, die als Startfunktion markiert wird, ersetzt. Der symbolische Name des Ereignisses wird an die Kante zum nachfolgenden Konnektorknoten annotiert. Aus diesen Annotationen werden im Abschnitt 4.4 Formeln zur Auswertung des Konnektors konstruiert.

- Endereignisse symbolisieren mögliche Ergebnisse einer EPK und repräsentieren Rückgabewerte von EPKs, die als Unterprozeß ausgeführt werden. Endereignisse werden durch eine spezielle Funktion, die als Endfunktion markiert wird, ersetzt. Der symbolische Name des Ereignisses wird an die eingehende Kante der neuen Endfunktion annotiert. Der Funktionsrumpf der Endfunktion, der für das korrekte Setzen des Rückgabewertes der EPK verantwortlich ist, wird beim Erzeugen der Prozeßvariablen (s. Abschnitt 4.4) hinzugefügt.

- Ereignisknoten, die auf einen OR-Split-Konnektor oder einen XOR-Split-Konnektor folgen, können den Kontrollfluß einer EPK beeinflussen. In diesen Fällen ist das Eintreten des Ereignisses optional und muß überprüft werden. Der Kontrollfluß kann nur über den Ereignisknoten an den Nachfolger weitergegeben werden, wenn dieses tatsächlich eingetreten ist. Diese Ereignisknoten werden zusammen mit ihrer eingehenden und ihrer ausgehenden Kante gelöscht und eine neue Kante vom Vorgänger- zum Nachfolgerknoten des Ereignisknotens eingefügt und mit dem symbolischen Namen des Ereignisses annotiert.

- Alle übrigen Ereignisknoten tragen keine workflowrelevante Informationen, da sie aufgrund ihrer Position in der EPK eintreten müssen, falls der Kontrollfluß ihren Vorgängerknoten passiert. Diese Ereignisknoten werden aus der EPK entfernt und eine Kante ohne Annotationen wird vom Vorgänger- zum Nachfolgerknoten eingefügt.

Das Ergebnis der Entfernungsphase ist ein modifizierter EPK-Graph, der nur noch Funktions- und Konnektorknoten enthält und Kanten, denen Annotationen hinzugefügt wurden. Diese Annotationen werden genutzt, um die Prozeßvariablen und Kantenbedingungen (s. Abschnitt 4.4) und die Webformularbeschreibungen (s. Abschnitt 4.5) zu erzeugen.

Anwendung: Abbildung 3 zeigt einen Ausschnitt aus der EPK „Planfeststellungsverfahren durchführen“ des Prozeßmodells Leipzig[22], das für RAfEG erstellt wurde. Die

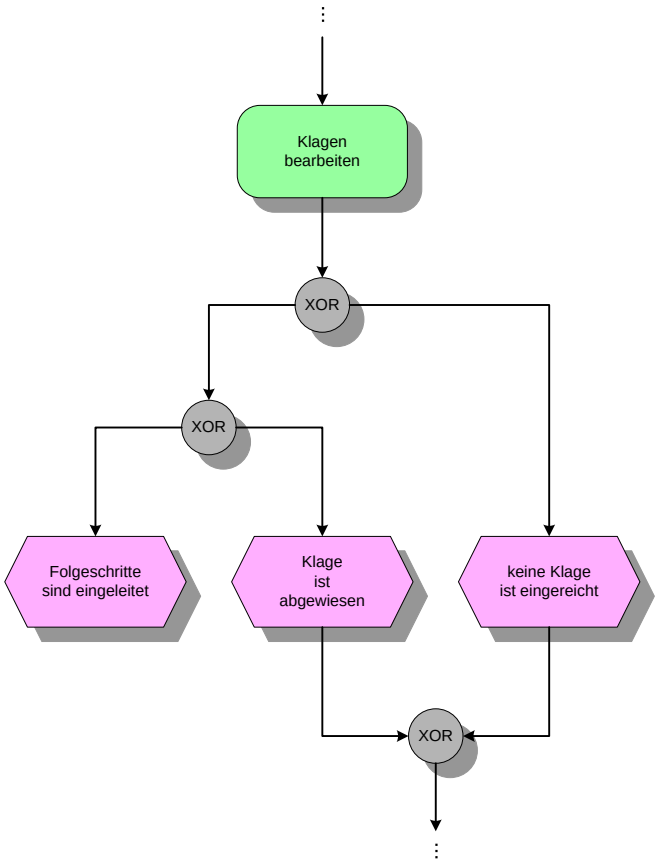

Abbildung 3: Die Ausgangsversion des Transformationsbeispiels, ein Ausschnitt aus dem Modell „Planfeststellungsverfahren durchführen“, beinhaltet noch alle Knotentypen einer EPK.

dargestellte Funktion „Klagen bearbeiten“ ist mit einem Unterprozeß hinterlegt. In Abbildung 4 wird der Prozeßausschnitt nach dem Entfernen der Ereignisknoten dargestellt. Da das Ereignis „Folgeschritte sind eingeleitet“ ein Endereignis der EPK ist, wird es durch eine Endfunktion ersetzt und der symbolische Name an die eingehende Kante annotiert. Die anderen beiden Ereignisse, „Klage ist abgewiesen“ und „keine Klage ist eingereicht“, folgen direkt auf einen XOR-Split-Konnektorknoten. Deshalb wird der symbolische Name nach dem Entfernen an der entsprechenden Kante annotiert.

4.3 Übersetzungsphase der Graphstruktur

In der Übersetzungsphase werden die Elemente des EPK-Graphen in ihre Äquivalente einer XPDL-Beschreibung übersetzt, Annotationen kopiert sowie die Knotenmenge des XPDL-Graphen optimiert. Die notwendigen Teilschritte werden nachfolgend erläutert.

Übersetzung der Knoten des EPK-Graphen Jeder Funktions- und Konnektorknoten eines EPK-Graphen wird in dieser Arbeitsphase durch einen XPDL-Aktivitätsknoten ersetzt. Das natürliche Äquivalent der EPK-Konnektorknoten in XPDL sind die Route-Aktivitäten. Dabei können die folgende Fälle auftreten, wobei der Konnektortyp aus der in Abschnitt 4.1 hinzugefügten Annotation abgeleitet werden kann:

EPK-Konnektortyp	Übersetzung in Route-Aktivität mit
AND-Join	AND-Vorbedingung
OR-Join	XOR-Vorbedingung
XOR-Join	XOR-Vorbedingung
AND-Split	AND-Nachbedingung
OR-Split	AND-Nachbedingung
XOR-Split	XOR-Nachbedingung

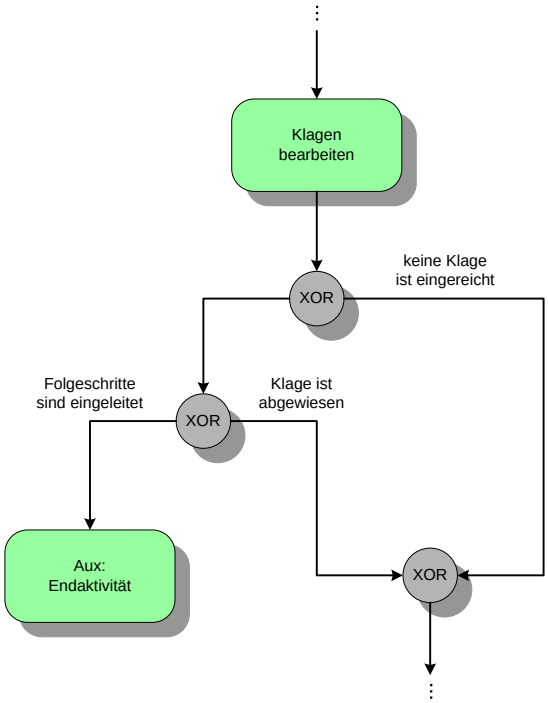

Abbildung 4: Transformationsbeispiel aus Abbildung 3 nach Entfernen der Ereignisknoten.

Eine XOR-Vorbedingung erfüllt die Funktion eines OR-Join-Konnektors, da auch bei mehreren aktiven eingehenden Kanten der Kontrollfluß weitergegeben wird. Die AND-Nachbedingung kann genutzt werden, um den Kontrollfluß auf eine beliebige Teilmenge der ausgehenden Kanten zu verteilen, da die Auswertung der Kantenbedingungen ausschlaggebend ist. Dies entspricht genau der Funktionalität eines OR-Split-Konnektors.

Für EPK-Funktionsknoten sind mehrere Fälle zu unterscheiden. Ist ein Funktionsknoten mit einer anderen EPK hinterlegt, also mit einem Teilprozeß assoziiert, wird eine Unterprozeßaktivität erzeugt. Für Start- und Endfunktionen einer EPK, wie sie beim Entfernen der Ereignisse erzeugt wurden, wird zunächst eine Route-Aktivität ohne Vor- und Nachbedingung erzeugt. Später werden die Route-Aktivitäten in automatische Aktivitäten umgewandelt, falls an dieser Stelle workflowrelevante Informationen zu speichern sind. Die restlichen EPK-Funktionsknoten werden in manuelle Aktivitäten übersetzt, in denen der symbolische Funktionsname gespeichert wird.

Abbildung 5 zeigt den Beispielausschnitt einer EPK nach den bisher beschriebenen Übersetzungssphasen. Die Funktion „Klagen bearbeiten“ wurde in eine Unterprozeßaktivität umgewandelt. Die Konnektorknoten werden durch Route-Aktivitäten mit entsprechender Vor- bzw. Nachbedingung dargestellt. Die Endfunktion wird im aktuellen Transformationszustand ebenfalls durch eine Route-Aktivität dargestellt.

Annotation der ausführenden Anwender Jedem erzeugten Aktivitätsknoten des Workflowgraphen wird ein ausführender Nutzer des Systems zugeordnet. Route-Aktivitäten und Unterprozeßaktivitäten werden durch das WfMS ausgeführt, so daß sie dem System-Nutzer zugeordnet werden. Manuelle Aktivitäten werden durch einen Bearbeiter ausgeführt, der in der Nutzernamentabelle, die in Abschnitt 4.1 aufgebaut wurde, definiert ist. Die Identifikationsnummer des Bearbeiters, die aus der Nutzernamentabelle ermittelt wird, wird an den manuellen Aktivitätsknoten annotiert.

Kopieren weiterer Annotationen Funktionsknoten einer für RAfEG modellierten EPK können zusätzliche Annotationen enthalten, die zur Abarbeitung der Workflowprozesse benötigt werden. Unterstützt werden die folgenden Attribute:

- Zusätzliche textuelle Erläuterungen, die die durchzuführende Funktion näher be-

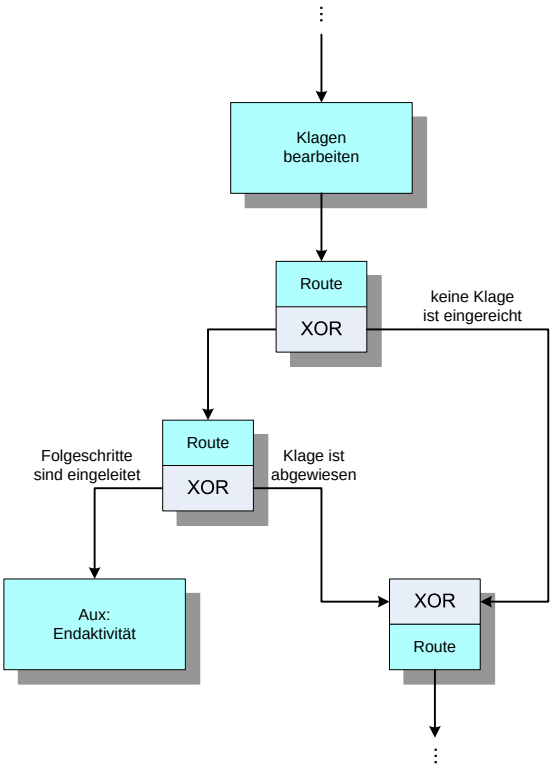

Abbildung 5: Transformationsbeispiel aus Abbildung 4 nach Übersetzen der Knoten des EPK-Graphen in XPDL-Äquivalente.

schreiben. In XPDL kann für jede Aktivität eine Beschreibung abgelegt werden, so daß diese Möglichkeit genutzt wird, um diese Attribute zu übersetzen.

- Paragraphen und direkte Links zu den Gesetzestexten, die bei der Abarbeitung beachtet werden müssen. Hierfür existiert kein Pendant in XPDL, so daß sie als nutzerdefiniertes Attribut abgelegt werden.

- Dateinamen von Formularen, die bei der Bearbeitung der Funktion ausgefüllt werden. Diese werden ebenfalls als nutzerdefiniertes Attribut in der zugehörigen Aktivität gespeichert.

Optimierung der erzeugten Workflows Die Workflowbeschreibungssprache XPDL bietet die Möglichkeit, Aktivitätsknoten, die mit einer bestimmten Aufgabe verbunden sind, zusätzliche Funktionalität zum Aufteilen oder Zusammenführen des Kontrollflusses hinzuzufügen. Im EPK-Modell sind diese beiden Knotentypen streng getrennt, es steht also keine äquivalente Ausdrucksmöglichkeit zur Verfügung. In dieser abschließenden Phase des Transformationsprozesses der EPML-Struktur wird versucht, die Anzahl der Knoten der Workflowprozeßgraphen zu verringern, indem Route-Aktivitätsknoten, also Knoten ohne assoziierte Aufgabe, mit anderen Aktivitätsknoten verschmolzen werden. Dies hat den Vorteil, daß der entstehende Workflowprozeß übersichtlicher wird.

Die Optimierung wird als Iteration über alle Kanten eines Workflowprozesses durchgeführt, wobei für jede Kante überprüft wird, ob Quell- und Zielaktivität der Kante verschmolzen und die betreffende Kante gelöscht werden kann. Ein Verschmelzen ist nur dann möglich, wenn die Quellaktivität einziger Vorgänger der Zielaktivität und die Zielaktivität einziger Nachfolger der Quellaktivität ist, also beide beteiligte Knoten genau eine ausgehende bzw. eingehende Kante besitzen. Eine weitere Voraussetzung ist, daß mindestens eine der betreffenden Aktivitäten eine Route-Aktivität ist. Zwei Aktivitäten, die mit dem Ausführen einer Aufgabe verknüpft sind, können nicht verschmolzen werden. Sind diese Voraussetzungen erfüllt, werden Quell- und Zielaktivität der Kante vereinigt und die entsprechende Kante aus dem Workflowprozeß entfernt.

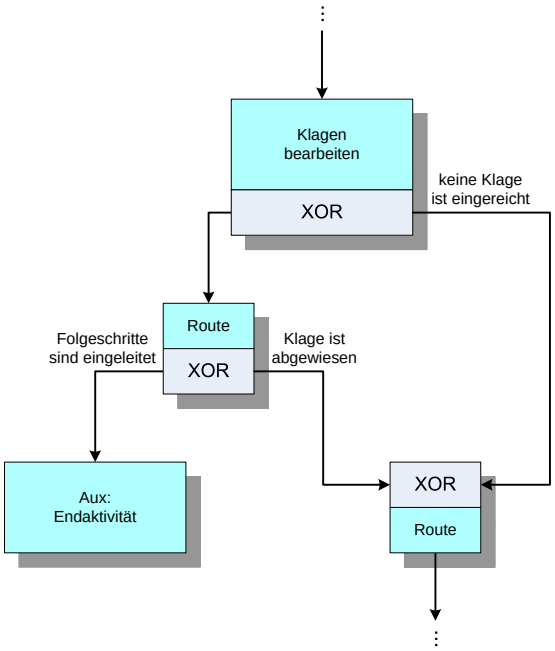

Abbildung 6: Transformationsbeispiel aus Abbildung 5 nach Optimierung des erzeugten Workflows.

Der Beispielausschnitt nach der Optimierungsphase wird in Abbildung 6 dargestellt. Die Unterprozeßaktivität „Klagen bearbeiten“ konnte mit der nachfolgenden Route-Aktivität verschmolzen werden.

4.4 Erzeugungsphase der Prozeßvariablen

In dieser Arbeitsphase werden Deklarationen für alle benötigten Prozeßvariablen ergänzt, entsprechende Kantenbedingungen konstruiert sowie die Parameterübergabe und Ergebnisrückgabe für Unterprozeßaufrufe hinzugefügt. Zunächst werden die Prozeßvariablen, die zur Auswertung der Kantenbedingungen benötigt werden, betrachtet. Zu jeder dieser Prozeßvariablen gibt es eine eindeutige Aktivität, die den Wert der Variablen verändern kann. Eine Prozeßvariable repräsentiert somit das Ergebnis der korrespondierenden Aktivität und eine Veränderung der Variablen an anderer Stelle ist nicht möglich. Die möglichen Funktionsergebnisse von manuellen und Unterprozeß-Aktivitätsknoten werden über eine Tiefensuche ermittelt, die für jeden derartigen Knoten getrennt gestartet wird. Während des Tiefensuchlaufs werden alle annotierten Kanten, die vom Startaktivitätsknoten der Tiefensuche aus erreichbar sind, ohne einen anderen manuellen oder Unterprozeß-Aktivitätsknoten zu passieren, identifiziert, und die zugehörigen Annotationen in einer Liste möglicher Funktionsergebnisse gespeichert. Enthält die erzeugte Liste keinen oder nur einen Eintrag, besitzt der entsprechende Aktivitätsknoten nur ein mögliches Funktionsergebnis und es wird keine Prozeßvariable erzeugt. Andernfalls wird für den Startaktivitätsknoten der Tiefensuche eine neue Prozeßvariable mit einem automatisch generierten Namen angelegt, wobei der Wertebereich den Einträgen der Liste entspricht. Anschließend wird für jede annotierte Kante eine automatisch auswertbare Bedingung unter Verwendung der erzeugten Prozeßvariablen konstruiert.

Symbolisiert eine Prozeßvariable das Ergebnis einer Unterprozeßaktivität muß diese Variable zusätzlich dem Unterprozeß als Parameter übergeben und das korrekte Setzen des Rückgabewertes in den Endaktivitäten des assoziierten Unterprozesses durchgeführt werden. Dazu wird für jedes in der Analyse ermittelte Funktionsergebnis der Unterprozeßaktivität eine Endaktivität mit gleichem symbolischem Namen im Unterprozeß gesucht. Die gefundene Endaktivität wird in eine automatische Aktivität umgewandelt, die die Prozeßvariable auf den entsprechenden Wert setzt. Nach Abschluß der Erzeugungsphase sind alle Informationen, die für eine Ausgabe nach XPDL benötigt werden, vorhanden.

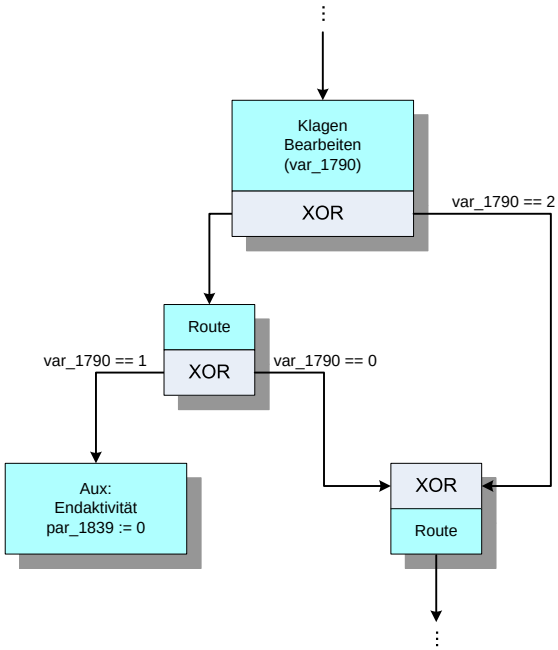

Abbildung 7: Transformationsbeispiel aus Abbildung 6 nach Erzeugung der Prozeßvariablen und Kantenbedingungen.

Anwendung In Abbildung 7 ist der Beispielausschnitt einer Workflowbeschreibung mit den erzeugten Prozeßvariablen und Kantenbedingungen dargestellt. Für die Aktivität „Klagen bearbeiten“ konnten nach Ausführung der Tiefensuche drei mögliche Funktionsergebnisse identifiziert werden. Aus diesem Grund wird für diese Aktivität eine Prozeßvariable, `var_1790`, generiert. Da drei Ergebnisse gefunden wurden, kann die Variable die Werte 0, 1 und 2 annehmen. Die genaue Zuordnung der Werte zu den Ereignissen ergibt sich aus der Reihenfolge der besuchten Kanten während des Tiefensuchlaufs und hat keine weiteren Auswirkungen. Das Belegen der Variablen erfolgt im mit der Aktivität „Klagen bearbeiten“ assoziierten Unterprozeß, dem sie als Parameter übergeben wird. Zusätzlich wird in dem Beispielausschnitt noch der Rückgabewert des aktuellen Prozesses gesetzt. Zu diesem Zweck wird der entsprechende Parameter, hier `par_1839`, in der Endaktivität mit dem zugeordneten Rückgabewert, hier 0, belegt. Da die Endaktivität nun eine Aufgabe erfüllen muß, wird sie von einer Route-Aktivität in eine automatische Aktivität umgewandelt. Das Transformationsbeispiel kann nun als Workflow in XPDL gespeichert werden.

4.5 Webformulargenerierungsphase

Die Webformulargenerierungsphase stellt eine zusätzliche Phase dar, die nach erfolgter Transformation nach XPDL durchlaufen wird. Darin werden Webformularbeschreibungen erzeugt, die alle Informationen über die Interaktion der Anwender, die manuelle Aktivitäten ausführen können, mit dem WfMS enthalten. Mit Hilfe der Webformularbeschreibungen kann eine Benutzeroberfläche speziell auf den entsprechenden Workflow angepaßt werden, falls das zur Abarbeitung der Workflows verwendete WfMS keine Benutzeroberfläche anbietet oder die vorhandene Oberfläche durch eine eigene Lösung ersetzt werden soll. Die Ausgabe der Webformularbeschreibungen erfolgt in Form von XML-Dateien in einem eigens für diesen Zweck entwickelten Dialekt, wobei für jeden Workflowprozeß eine eigene Datei mit dem Namen des entsprechenden Prozesses erzeugt wird.

ConvEX bietet die Möglichkeit, die erzeugten Webformularbeschreibungen direkt im RAfEG-Dokumentenmanagementsystem[13] abzulegen. Die RAfEG-Präsentationskomponente kann in diesem Fall direkt auf die Beschreibungen zugreifen und über mehrere Zwischenschritte, die bspw. die Internationalisierung und die Anpassung auf das RAfEG-Design umfassen, die Ausgabe, die letztendlich dem Anwender angezeigt wird, erzeugen.

Listing 1: Ausschnitt aus einer Webformularbeschreibung eines Workflowprozesses

```
<tool id="tool_205">
  <name>Prüfen, ob nach Planfeststellung Planänderungen notwendig sind</name>
  <description>Der Vorhabensträger prüft, ob der bereits
    festgestellte Plan zu ändern ist, bevor das zugelassene
    Vorhaben fertig gestellt ist.</description>
  <form><layout>
    <select1 ref="var_397">
      <label>Wählen sie aus</label>
        <choices>
          <item>
            <value>0</value>
            <label>Planänderungen nach Planfeststellung sind
              notwendig</label>
          </item>
          <item>
            <value>1</value>
            <label>Planänderungen nach Planfeststellung sind nicht
              notwendig</label>
        </item>
      </choices>
    </select1>
  </layout></form>
</tool>
```

Die Zwischenschritte sind als XSLT-Skripte implementiert und Teil der RAfEG-Präsentationskomponente.

In den Webformularbeschreibungen können für jede manuelle Aufgabe eines Workflowprozesses die folgenden Einträge angegeben werden, falls diese vorher entsprechend modelliert wurden:

- Name der manuellen Aktivität,
- Beschreibung der Aktvität,
- Symbolischer Name und zugeordneter Wert von allen möglichen Funktionsergebnissen, falls beim Erzeugen der Prozeßvariablen (s. Abschnitt 4.4) mehrere Ergebnisse gefunden wurden,
- Fristen, die bei der Ausführung eingehalten werden müssen und
- Paragraphen von Vorschriften und Gesetzen, die bei der Ausführung beachtet werden müssen. Die Angabe kann auch als Link zu einer entsprechende Webseite erfolgen.

Enthält die Webformularbeschreibung für eine Aufgabe keine Informationen über mögliche Funktionsergebnisse, bietet die RAfEG-Oberfläche dem Anwender lediglich an, die Aufgabe als beendet zu markieren. Andernfalls wird eine Liste der Funktionsergebnisse angezeigt, in der der Anwender den entsprechenden Eintrag auswählen kann.

Listing 1 zeigt den Ausschnitt aus einer Webformularbeschreibung eines Workflowprozesses für eine manuelle Aufgabe. Da der bisher verwendete Beispielausschnitt keine manuellen Aktivitäten enthält, also für den dargestellten Teil keine Webformulare generiert werden, wird hier ein anderes Beispiel vorgestellt. Die generierte Identifikation `tool_205` identifiziert die manuelle Aufgabe und wird auch als nutzerdefiniertes Attribut des Aktivitätsknotens in der XPDL-Beschreibung abgelegt. Für die zugehörige Prozeßvariable `var_397` gibt es zwei Auswahlmöglichkeiten, die mögliche Ergebnisse der manuellen Aktivität repräsentieren.

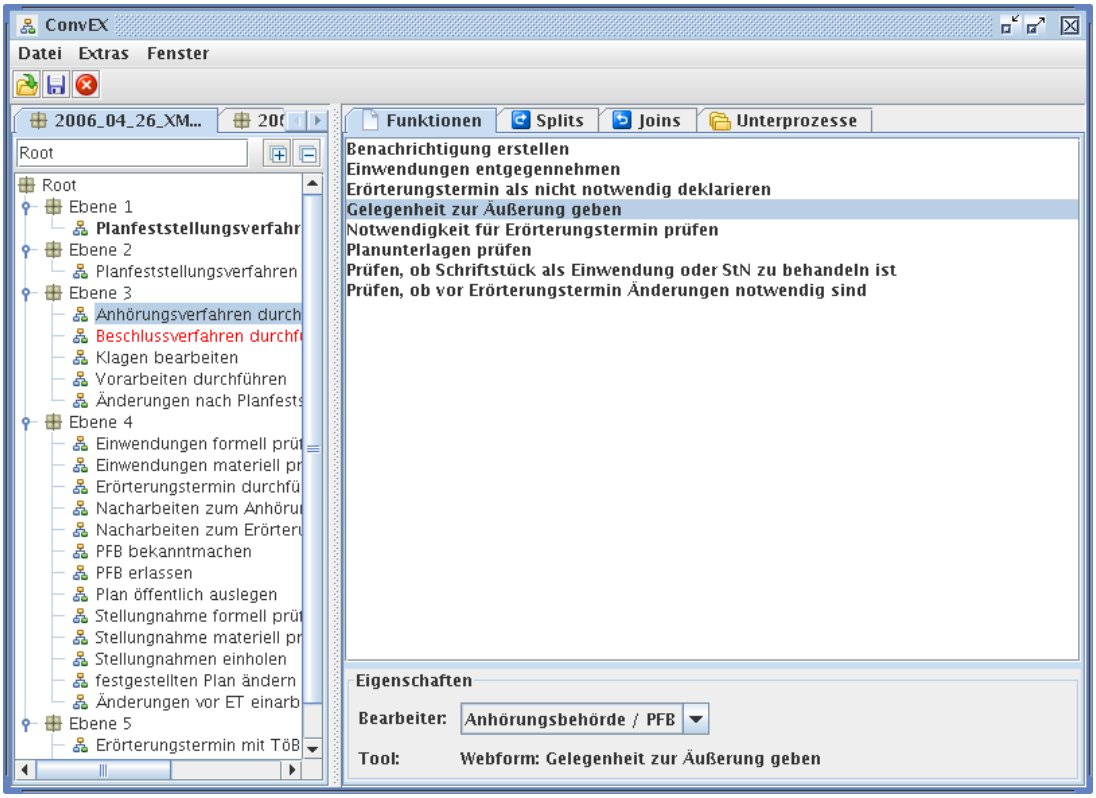

Abbildung 8: Hauptfenster von ConvEX nach Laden zweier EPKs mit aktiviertem Funktionsfenster.

5. Bedienung des Übersetzungsprogramms

ConvEX ist in Java implementiert. Zum Übersetzen und Ausführen wird das Java 2 Standard Edition Development Kit (JDK) in der Version 5.0 oder höher benötigt. Die graphische Benutzeroberfläche von ConvEX, die in diesem Kapitel beschrieben wird, nutzt die plattformunabhängige API Swing[6]. Als externe Komponenten werden die JGraph-Bibliothek[11] zur Darstellung von Graphen und die RAfEG-Filestore-Implementierung[13] zur Kommunikation mit dem Dokumentenmanagementsystem benötigt. Der Anwender kann mit Hilfe der Benutzeroberfläche in den Übersetzungsvorgang von EPK-Beschreibungen im EPML-Format in Workflowbeschreibungen im XPDL-Format eingreifen. Es besteht die Möglichkeit, die Attribute und Annotationen der Knoten und Kanten der eingelesenen EPK zu verändern. Eine Änderung der Struktur des zugrundeliegenden Graphen, z.B. das Einfügen oder Entfernen von Knoten, ist nicht möglich und muß über das Modellierungswerkzeug erfolgen. Während der Modellierung der Verwaltungsabläufe innerhalb des RAfEG-Projekts sind in den erstellten Modellen teilweise Inkonsistenzen aufgetreten, die nicht vom verwendeten Modellierungswerkzeug erkannt wurden. Dies betrifft vor allem fehlende Annotationen, wie z.B. die Bearbeiter für manuelle Aufgaben, die für den Übersetzungsvorgang benötigt werden. Diese Inkonsistenzen werden durch die Benutzeroberfläche erkannt und entsprechend kenntlich gemacht.

Der Übersetzungsvorgang beginnt mit dem Einlesen einer EPK, was über den entsprechenden Menüeintrag oder über die Toolbar erfolgen kann. Beim Laden wird die Eingabedatei eingelesen, die Analysephase (s. Abschnitt 4.1) durchlaufen und die interne hierarchische Struktur aufgebaut. Die hierarchische Struktur beschreibt die eingelesenen EPKs als eine Menge von gerichteten Graphen, wobei die EPKs einer Ebene nur Teilprozesse der nächsthöheren Ebene nutzen können, also EPKs der Ebene 2 Teilprozesse

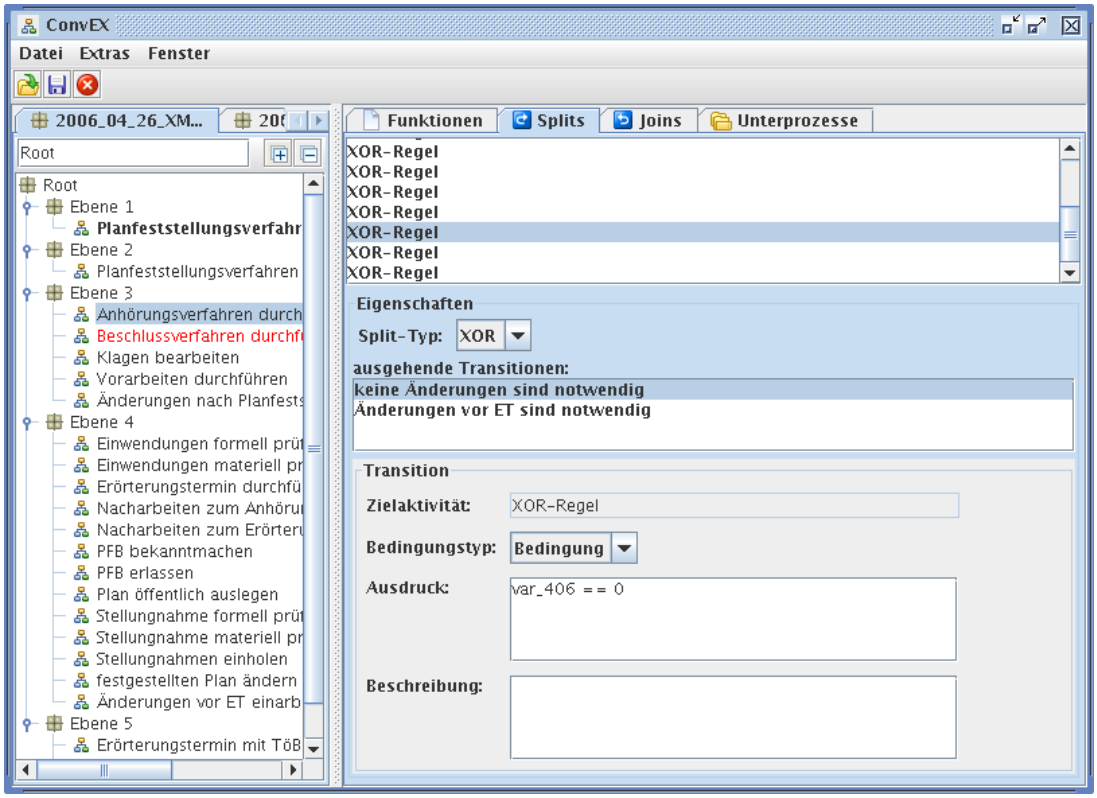

Abbildung 9: Hauptfenster von ConvEX nach Auswahl einer EPK mit aktiviertem Fenster zur Darstellung der Split-Konnektoren.

von EPKs der Ebene 1 sind. Nach dem Einlesen wird diese hierarchische Struktur im linken Teil des zweigeteilten Hauptfensters, nachfolgend als Strukturansicht bezeichnet, angezeigt, s. Abbildung 8. ConvEX erlaubt das Laden von mehreren Eingabedateien, die unabhängig voneinander bearbeitet und gespeichert werden können. Sind wie in Abbildung 8 mehrere Eingaben geladen, so kann mit Hilfe der Tabs, die sich oberhalb der Strukturansicht befinden, das aktuell zu bearbeitende Modell ausgewählt werden. Innerhalb der Strukturansicht kann die aktive EPK gewählt und auf dem rechten Teil des Hauptfensters bearbeitet werden. Der rechte Hauptfensterteil kann zwischen der Anzeige der Eigenschaften der Funktionsknoten, der Split- bzw. Join-Konnektorknoten und der Unterprozeßfunktionsknoten umgeschaltet werden. In der Strukturansicht werden EPKs, die Teilprozesse anderer EPKs repräsentieren, im Standardfont angezeigt. EPKs, die keine Teilprozesse sind, werden im Fettdruck dargestellt. EPKs, in denen Modellierungsinkonsistenzen festgestellt wurden, werden rot markiert. Über das Hauptmenü kann die graphische Darstellung der ausgwählten EPK, die in einem separaten Fenster erfolgt, aktiviert werden.

Nach dem Laden einer Eingabedatei kann diese wieder über die Toolbar oder das Hauptmenü geschlossen werden. Werden keine Inkonsistenzen in der aktuellen Eingabe festgestellt, existieren also keine rot-markierten EPKs, kann diese als XPDL gespeichert werden. Diese Funktion ist über die Toolbar oder das Hauptmenü erreichbar.

Abbildung 8 zeigt das Hauptfenster nach Auswahl einer EPK und aktiviertem Funktionstab auf der rechten Fensterseite. Es wird eine Liste der Funktionen angezeigt, die für manuelle Bearbeitung vorgesehen sind und für die Webformulare erstellt werden. Nach Auswahl einer dieser Funktionen wird am unteren Bildrand der modellierte Bearbeiter und der Name des erzeugten Webformulars angezeigt. Für den Bearbeiter kann aus der Nutzerliste, die alle in der Eingabedatei vorkommenden Nutzer enthält, ein neuer Wert

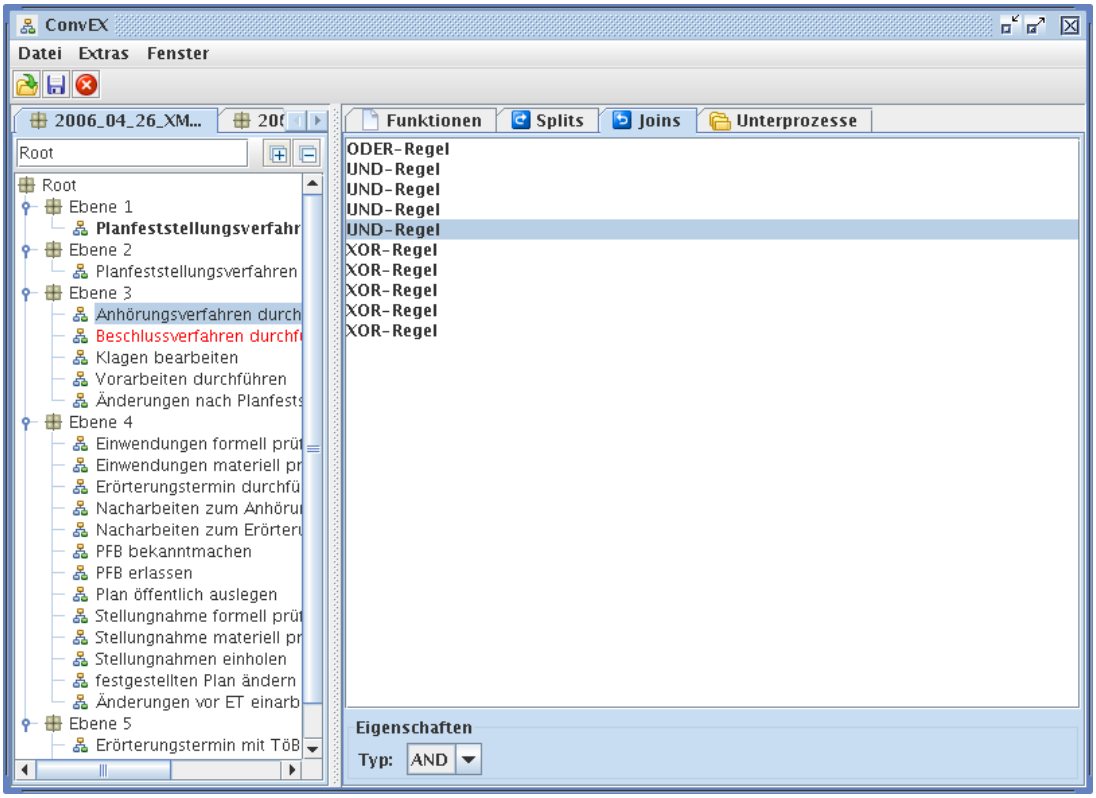

Abbildung 10: Hauptfenster von ConvEX nach Auswahl einer EPK mit aktiviertem Fenster zur Darstellung der Join-Konnektoren.

ausgewählt werden. Das Anlegen neuer Nutzer wird nicht unterstützt. Funktionen, für die kein ausführender Nutzer modelliert ist, werden in der Liste rot angezeigt, bis ein neuer Nutzer spezifiziert wurde.

Durch Auswahl des Tabs *Splits* im rechten Teil des Hauptfensters erscheint wie in Abbildung 9 dargestellt die Liste aller Split-Konnektorknoten, die in der aktuell gewählten EPK definiert wurden. Bei XOR-Split-Konnektoren ist es erforderlich, daß alle (bis auf die letzte) ausgehende Kanten mit einer Bedingung annotiert sind. Diese Bedingungen legen fest, welche Kante nach dem Split-Konnektor aktiv wird, und werden beim Erzeugen der Prozeßvariablen (s. Abschnitt 4.4) gesetzt. Außerdem wird für alle ausgehenden Kanten mit Bedingungen überprüft, ob korrekte Ausdrücke für diese Bedingungen angegeben sind. Wurden bei dieser Überprüfung Modellierungsinkonsistenzen festgestellt, wird der betreffende Knoten in der Liste rot dargestellt. Unterhalb der Funktionsliste kann der Typ der Split-Konnektors verändert werden. Da XPDL nur XOR und AND Splits erlaubt, stehen nur diese beiden Operationen zur Verfügung. Weiterhin wird eine Liste der ausgehenden Kanten angegeben, wobei fehlerhafte Kanten rot markiert sind. Dies betrifft Kanten, die als Bedingungskante definiert sind, aber denen kein Bedingungsausdruck zugeordnet wurde.

Am unteren Bildschirmende lassen sich die Eigenschaften der aktuell selektierten Kante verändern. Das Textfeld *Zielaktivität* bezieht sich auf den Namen des Zielknotens der Kante im Workflowgraphen. Dieses Textfeld dient nur der Information, um die ausgehenden Kanten unterscheiden zu können. Desweiteren kann der Bedingungstyp der Kante, der festlegt, wie sich die Kante bei Erreichen des Kontrollflusses verhält, angegeben werden. Zur Auswahl stehen die von XPDL bereitgestellten Möglichkeiten:

- keine Bedingung, die Kante wird in jedem Fall aktiviert

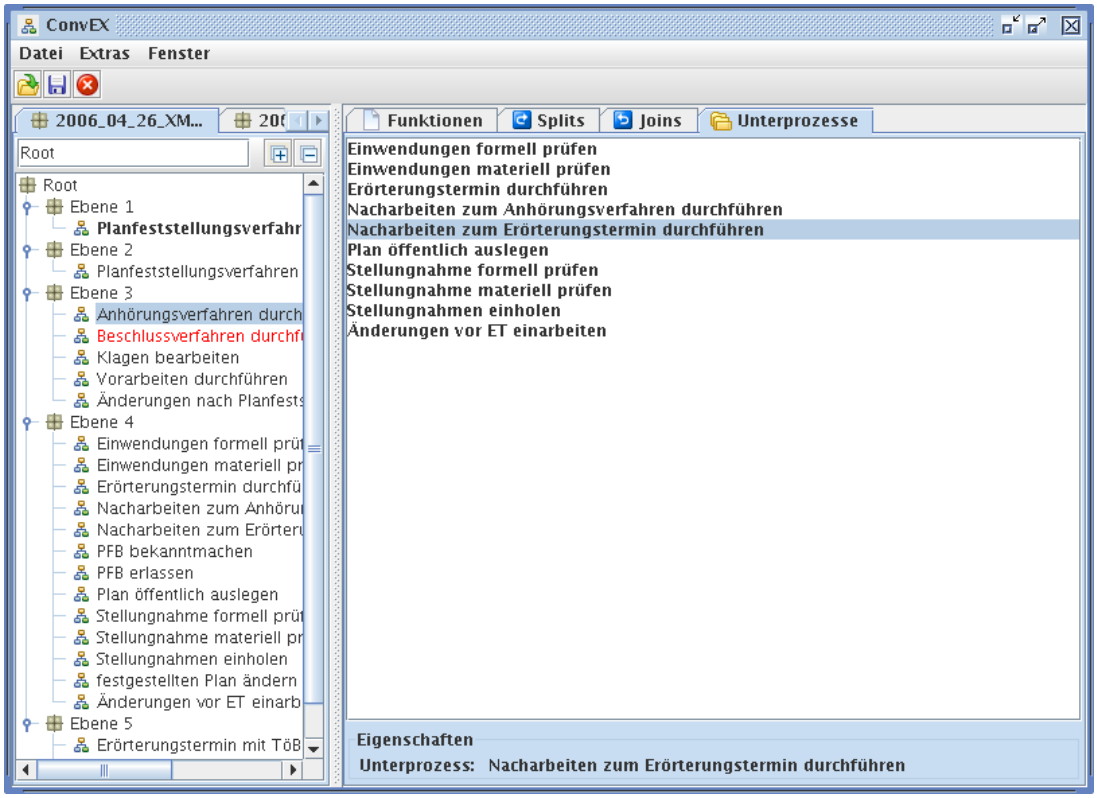

Abbildung 11: Hauptfenster von ConvEX nach Auswahl einer EPK mit aktiviertem Unterprozeßfenster.

- boolesche Bedingung, die Kante wird nur aktiviert, falls die Bedingung wahr ist
- OTHERWISE, gibt die Standardkante an, die nach einem Split-Knoten aktiviert wird, falls die Bedingungen aller anderen Kanten falsch sind. Da in den erzeugten XPDL-Beschreibungen die Kanten vorsortiert werden, ist diese Option gleichbedeutend mit Kanten ohne zugeordnete Bedingung.

Der automatisch erzeugte Ausdruck wird für alle Kanten mit boolescher Bedingung im entsprechenden Fenster angezeigt und kann bearbeitet werden. Im Textfeld am unteren Bildschirmrand kann eine textuelle Beschreibung für die Kante eingegeben werden, die jedoch keinen Einfluß auf die Abarbeitung des Workflowprozesses hat.

Nach Aktivierung des Tabs *Joins* im rechten Hauptfensterteil erscheint eine Liste aller Join-Konnektorknoten der aktuell selektierten EPK, wie etwa in Abbildung 10 dargestellt. Unter Eigenschaften kann der Operator des Join-Konnektors unter den von XPDL unterstützten Operatoren XOR und AND ausgewählt werden.

Nach Auswahl des Tabs *Unterprozesse* im rechten Fensterteil erscheint eine Ansicht analog Abbildung 11. Hier werden alle Funktionen, die mit Teilprozessen hinterlegt sind, aufgelistet. Im unteren Bildschirmteil wird der Name des assoziierten Teilprozesses der aktuell ausgewählten Funktion angezeigt. Eine Veränderung von Attributen ist in diesem Teilfenster nicht möglich. Wurde der assoziierte Teilprozeß einer Funktion in der Eingabedatei nicht gefunden, wird der Name der entsprechenden Funktion rot angezeigt. Ein Korrigieren dieses Fehlers ist in ConvEX nicht möglich und muß über das Modellierungswerkzeug erfolgen.

Der Menüpunkt *"Extras"* erlaubt das Abspeichern der Webformulare und der graphischen Darstellung der EPKs für die aktuell selektierte Eingabedatei. Das Abspeichern ist in das lokale Dateisystem und in das RAfEG-Dokumentenmangementsystem, das über

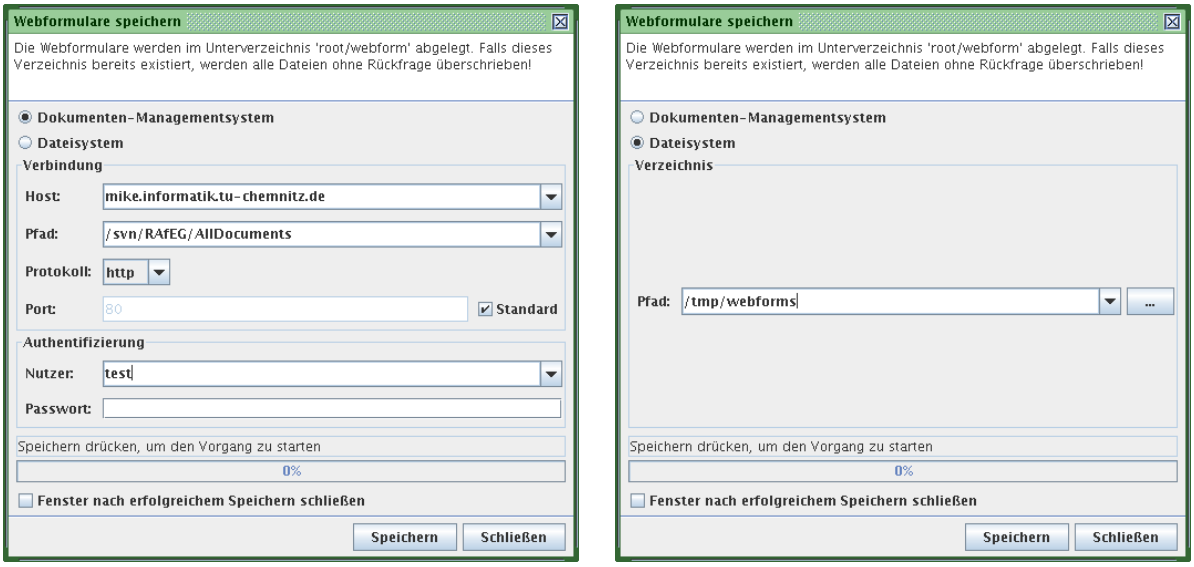

Abbildung 12: Dialog mit Auswahl des Speicherorts für die Webformulare. Die linke Abbildung zeigt die Auswahloptionen bei Verwendung des Dokumentenmangementsystems über WebDAV, die Speicherung in das lokale Dateisystem wird in der rechten Abbildung dargestellt.

WebDAV, einer Erweiterung des HTTP-Protokolls, angesprochen wird, möglich. Abbildung 12 zeigt den Dialog mit den Konfigurationsoptionen für das Speichern der Webformulare. Der Dialog zum Speichern der graphischen Darstellung besitzt einen identischen Aufbau. Beim Speichern in das lokale Dateisystem kann der Pfad, in den die erzeugten Dateien abgelegt werden, konfiguriert werden. Für das Speichern in das Dokumentenmangementsystem können der WebDAV-Server, der Pfad innerhalb des Dokumentenmangementsystems, das Protokoll (HTTP oder HTTPS), der Port sowie Nutzername und Paßwort eingestellt werden.

Sollen die Webformularbeschreibungen abgespeichert werden, wird für jede in der Eingabe definierte EPK eine XML-Datei mit den Informationen zur Darstellung durch die RAfEG-Nutzeroberfläche, wie in Abschnitt 4.5 beschrieben, erzeugt. ConvEX ermöglicht eine graphische Darstellung der bearbeiteten Modelle nur in Form von EPKs, wobei zur Darstellung skalierbare Vektorgraphiken (SVG)[10] eingesetzt werden. Jede EPK wird durch eine eigene Graphikdatei, deren Namen aus der angegebenen Bezeichnung der EPK gebidet wird, dargestellt. Zusätzlich wird eine Datei `ebenen.svg` erzeugt, die die hierarchische Aufrufstruktur graphisch als gerichteten Graphen darstellt. Für jede EPK wird ein Knoten in diesem Graphen erzeugt. Eine gerichtete Kante zwischen zwei EPKs ist vorhanden, falls es in der Quell-EPK eine Funktion gibt, die mit der Ziel-EPK hinterlegt ist. Die Darstellungen der EPKs dienen dem Modellierer als Information. Die abschließende Ausgabe der Workflows erfolgt im XPDL-Format, das mit einem geeigneten Werkzeug, z.B. Enhydra JaWE[7], visualisiert werden kann.

6. Zusammenfassung

In diesem Bericht wurde ein Übersetzungswerkzeug zur Transformation von ereignisgesteuerten Prozeßketten im EPML-Format in Workflowbeschreibungen im XPDL-Format vorgestellt. Der Übersetzungsvorgang gliedert sich dabei in mehrere Arbeitsphasen, in denen ein eingelesenes Modell eines Verwaltungsvorgangs schrittweise transformiert wird. Die Steuerung erfolgt über eine graphische Benutzeroberfläche, in denen die Attribute und Annotationen der Modellelemente verändert bzw. ergänzt werden können. Die An-

wendung des vorgestellten Werkzeugs ist nicht auf das RAfEG-Projekt beschränkt, da keine speziellen Einschränkungen für die Eingabemodelle getroffen wurden.

7. Danksagung

Beim Regierungspräsidium Leipzig bedanken wir uns für die Unterstützung und Einblicke in die Bearbeitung von Verwaltungsvorgängen bei Planfeststellungsverfahren. Dem Institut für Wirtschaftsinformatik (IWi) im Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI) und der NHConsult GmbH danken wir für die Erstellung der Modelle. Für die Bereitstellung einer Version des ARIS-Toolsets danken wir der IDS Scheer AG.

Literatur

[1] ActiveBPEL `http://www.activebpel.org/`.

[2] ARIS Platform: XML-Export und Import (White Paper), July 2005.

[3] D. Beer, S. Höhne, R. Kunis, G. Rünger, and M. Voigt. RAfEG - Eine Open Source basierte Architektur für die Abarbeitung von Verwaltungsprozessen im E-Government. *Chemnitzer Informatik Berichte*, CSR-06-03, 2006.

[4] Business Process Execution Language for Web Services (BPEL), Version 1.1 `http://www.ibm.com/developerworks/library/ws-bpel`, May 2003.

[5] Workflow Management Coalition. Spezifikation von XPDL Version 1.0: `http://www.wfmc.org/standards/docs/TC-1025_10_xpdl_102502.pdf`, October 2002.

[6] B. Cole, R. Eckstein, J. Elliott, M. Loy, and D. Wood. *Java Swing - Developing GUIs in Java*. O'Reilly, 2nd edition, 2002.

[7] Enhydra JaWE `http://forge.objectweb.org/projects/jawe/`.

[8] Enhydra Shark 1.1 Documentation `http://shark.objectweb.org/doc/1.1/`, 2005.

[9] G. Keller, M. Nüttgens and A.-W. Scheer . Semantische Prozeßmodellierung auf der Grundlage „Ereignisgesteuerter Prozeßketten (EPK)". *Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi)*, Heft 89, Januar 1992.

[10] D. Jackson (eds.) J. Ferraiolo, F. Jun. Scalable Vector Graphics (SVG) 1.1 Specification: `http://www.w3.org/TR/SVG11/`, January 2003.

[11] JGraph Users Manual `http://www.jgraph.com`.

[12] O. Kopp. *Abbildung von EPKs nach BPEL anhand des Prozessmodellierungswerkzeugs Nautilus*. Diplomarbeit: Universität Stuttgart, Institut für Architektur von Anwendungssystemen, Oktober 2005.

[13] R. Kunis, G. Rünger, and M. Schwind. Dokumentenmanagement für Verwaltungsvorgänge im E-Government . *Chemnitzer Informatik Berichte*, CSR-06-10, Oktober 2006.

[14] J. Mendling and M. Nüttgens. Transformation of ARIS Markup Language to EPML. In *M. Nüttgens, F.J. Rump, eds.: Proc. of the 3rd GI Workshop on Event-Driven Process Chains (EPK 2004)*, pages 27–38, 2004.

[15] J. Mendling and M. Nüttgens. EPC Markup Language (EPML) - An XML-Based Interchange Format for Event-Driven Process Chains (EPC) . *International Journal Information Systems and e-Business Management (ISeB)*, 4(3):245 – 263, July 2006.

[16] Nautilus `http://www.gedilan.de/`.

[17] Oracle BPEL Process Manager - Administrator's Guide`http://www.oracle.com/technology/products/ias/bpel/index.html`, April 2006.

[18] A.-W. Scheer. *ARIS - Modellierungsmethoden, Metamodelle, Anwendungen*. Springer-Verlag, Berlin; Heidelberg; New York, 1998.

[19] Semtalk `http://www.semtalk.com/`.

[20] Simple API for XML (SAX) `http://www.saxproject.org/`.

[21] J.L. Staud. *Geschäftsprozessanalyse - ereignisgesteuerte Prozessketten und objektorientierte Geschäftsprozessmodellierung für betriebswirtschaftliche Standardsoftware*. Springer Verlag, Berlin ; Heidelberg ; New York, 2001.

[22] O. Thomas, C. Seel, C. Seel, B. Kaffai, and G. Martin. Referenzarchitektur für E-Government(RAfEG): Konstruktion von Verwaltungsverfahrensmodellen am Beispiel der Planfeststellung. *Veröffentlichungen des Instituts für Wirtschaftsinformatik (IWi)*, Heft 179, December 2004.

[23] WfMOpen `http://wfmopen.sourceforge.net/`.

Chemnitzer Informatik-Berichte

In der Reihe der Chemnitzer Informatik-Berichte sind folgende Berichte erschienen:

CSR-02-01 Andrea Sieber, Werner Dilger, Theorie und Praxis des Software Engineering, Oktober 2001, Chemnitz

CSR-02-02 Tomasz Jurdzinski, Miroslaw Kutylowski, Jan Zatopianski, Energy-Efficient Size Approximation of Radio Networks with no Collision Detection, Februar 2002, Chemnitz

CSR-02-03 Tomasz Jurdzinski, Miroslaw Kutylowski, Jan Zatopianski, Efficient Algorithms for Leader Election in Radio Networks, Februar 2002, Chemnitz

CSR-02-04 Tomasz Jurdzinski, Miroslaw Kutylowski, Jan Zatopianski, Weak Communication in Radio Networks, Februar 2002, Chemnitz

CSR-03-01 Amin Coja-Oghlan, Andreas Goerdt, André Lanka, Frank Schädlich, Certifying Unsatisfiability of Random $2k$-SAT Formulas using Approximation Techniques, Februar 2003, Chemnitz

CSR-03-02 M. Randrianarivony, G. Brunnett, Well behaved mesh generation for parameterized surfaces from IGES files, März 2003, Chemnitz

CSR-03-03 Optimizing MPI Collective Communication by Orthogonal Structures, Matthias Kühnemann, Thomas Rauber, Gudula Rünger, September 2003, Chemnitz

CSR-03-04 Daniel Balkanski, Mario Trams, Wolfgang Rehm, Heterogeneous Computing With MPICH/Madeleine and PACX MPI: a Critical Comparison, Dezember 2003, Chemnitz

CSR-03-05 Frank Mietke, Rene Grabner, Torsten Mehlan, Optimization of Message Passing Libraries - Two Examples, Dezember 2003, Chemnitz

CSR-04-01 Karsten Hilbert, Guido Brunnett, A Hybrid LOD Based Rendering Approach for Dynamic Scenes, Januar 2004, Chemnitz

CSR-04-02 Petr Kroha, Ricardo Baeza-Yates, Classification of Stock Exchange News, November 2004, Chemnitz

CSR-04-03 Torsten Hoefler, Torsten Mehlan, Frank Mietke, Wolfgang Rehm, A Survey of Barrier Algorithms for Coarse Grained Supercomputers, Dezember 2004, Chemnitz

CSR-04-04 Torsten Hoefler, Wolfgang Rehm, A Meta Analysis of Gigabit Ethernet over Copper Solutions for Cluster-Networking, Dezember 2004, Chemnitz

CSR-04-05 Christian Siebert, Wolfgang Rehm, One-sided Mutual Exclusion A new Approach to Mutual Exclusion Primitives, Dezember 2004, Chemnitz

Chemnitzer Informatik-Berichte

CSR-05-01 Daniel Beer, Steffen Höhne, Gudula Rünger, Michael Voigt, Software- und Kriterienkatalog zu RAfEG - Referenzarchitektur für E-Government, Januar 2005, Chemnitz

CSR-05-02 David Brunner, Guido Brunnett, An Extended Concept of Voxel Neighborhoods for Correct Thinning in Mesh Segmentation, März 2005, Chemnitz

CSR-05-03 Wolfgang Rehm (Ed.), Kommunikation in Clusterrechnern und Clusterverbundsystemen, Tagungsband zum 1. Workshop, Dezember 2005, Chemnitz

CSR-05-04 Andreas Goerdt, Higher type recursive program schemes and the nested pushdown automaton, Dezember 2005, Chemnitz

CSR-05-05 Amin Coja-Oghlan, Andreas Goerdt, André Lanka, Spectral Partitioning of Random Graphs with Given Expected Degrees, Dezember 2005, Chemnitz

CSR-06-01 Wassil Dimitrow, Mathias Sporer, Wolfram Hardt, UML basierte Zeitmodellierung für eingebettete Echtzeitsysteme, Februar 2006, Chemnitz

CSR-06-02 Mario Lorenz, Guido Brunnett, Optimized Visualization for Tiled Displays, März 2006, Chemnitz

CSR-06-03 D. Beer, S. Höhne, R. Kunis, G. Rünger, M. Voigt, RAfEG - Eine Open Source basierte Architektur für die Abarbeitung von Verwaltungsprozessen im E-Government, April 2006, Chemnitz

CSR-06-04 Michael Kämpf, Probleme der Tourenbildung, Mai 2006, Chemnitz

CSR-06-06 Torsten Hoefler, Mirko Reinhardt, Torsten Mehlan, Frank Mietke, Wolfgang Rehm, Low Overhead Ethernet Communication for Open MPI on Linux Clusters, Juli 2006, Chemnitz

CSR-06-07 Karsten Hilbert, Guido Brunnett, A Texture-Based Appearance Preserving Level of Detail Algorithm for Real-time Rendering of High Quality Images, August 2006, Chemnitz

CSR-06-08 David Brunner, Guido Brunnett, Robin Strand, A High-Perpormance Parallel Thinning Approach Using a Non-Cubic Grid Structure, September 2006, Chemnitz

CSR-06-09 El-Ashry, Peter Köchel, Sebastian Schüler, On Models and Solutions for the Allocation of Transportation Resources in Hub-and-Spoke Systems, September 2006, Chemnitz

CSR-06-10 Raphael Kunis, Gudula Rünger, Michael Schwind, Dokumentenmanagement für Verwaltungsvorgänge im E-Government, Oktober 2006, Chemnitz

CSR-06-11 Daniel Beer, Jörg Dümmler, Gudula Rünger, Transformation ereignisgesteuerter Prozeßketten in Workflowbeschreibungen im XPDL-Format, Oktober 2006, Chemnitz