

# Modellierung von EPKs im Web mit Oryx

Stefan Krumnow, Gero Decker und Mathias Weske  
Hasso-Plattner-Institut an der Universität Potsdam  
stefan.krumnow@student.hpi.uni-potsdam.de  
(gero.decker,mathias.weske)@hpi.uni-potsdam.de

**Abstract:** Ereignisgesteuerte Prozessketten werden häufig zur Prozessmodellierung in betrieblichen Anwendungen eingesetzt. Besonders hilfreich sind Prozessmodelle dann, wenn die Prozessbeteiligten auf die Modelle einfach zugreifen und sie bei Bedarf sogar verändern können. In diesem Beitrag stellen wir Oryx vor, eine webbasierte Plattform zur Erstellung, Speicherung und gemeinsamen Bearbeitung von Prozessmodellen. Oryx ist erweiterbar, so dass projektspezifische Erweiterungen einfach realisierbar sind. Auf diese Weise können auch neue Forschungsergebnisse mit vergleichsweise geringem Aufwand implementiert und evaluiert werden.

## 1 Einleitung

Prozessmodellierung erlaubt es, die Abläufe in einer Organisation zu verstehen, Veränderungsbedarf zu identifizieren, neue Abläufe zu kommunizieren und Anforderungen an Informationssysteme abzuleiten. Prozessmodelle dokumentieren, welche Aktivitäten anfallen, wer daran beteiligt ist, welche Artefakte benötigt und erstellt werden und welche Abhängigkeiten zwischen den einzelnen Aktivitäten bestehen.

Ereignisgesteuerte Prozessketten (EPKs [KNS92]) haben sich als Modellierungstechnik etabliert. Durch ihre Syntax und Semantik ermöglichen sie eine effiziente Kommunikation zwischen den Prozessbeteiligten. Verschiedene Personen müssen dabei Zugriff auf ein Prozessmodell haben, zum einen als Ersteller des Modells als auch als Leser des Modells. Während in manchen Fällen alle Beteiligten der gleichen Organisation angehören, gibt es zahlreiche Szenarien, in denen Beteiligte aus verschiedenen Organisationen zusammenarbeiten müssen. Gerade im Falle von Outsourcing von Prozessen oder Prozessteilen ist es wichtig, ein gemeinsames Prozessverständnis zu erreichen. Auch im öffentlichen Sektor, wo Prozesse verschiedene Behörden überspannen, ist dies ein zentrales Thema.

Die entsprechenden Modellierungswerkzeuge müssen also einen einfachen Zugriff verschiedenster Beteiligter auf die Modelle ermöglichen. Oft wird dies aber durch den Einsatz verschiedener Softwareversionen, fehlender Softwarelizenzen oder inkompatibler Benutzerverwaltungen erschwert.

Außerdem besteht Bedarf nach Modellierungswerkzeugen in der akademischen Community. Solche Werkzeuge sollen es erlauben, mit Spracherweiterungen zu experimentieren, Verifikations- und Validierungstechniken umzusetzen, Prozessmetriken zu testen und verschiedenste andere Algorithmen zu realisieren. Durch eine Plattform, die bereits viele

der üblichen Funktionen bereitstellt, kann unnötige Reimplementierung vermieden werden und die Integration verschiedener Prototypen erreicht werden.

Dieses Papier stellt Oryx vor, eine offene Plattform, die durch seine Erweiterbarkeit die schnelle Entwicklung von akademischen Prototypen ermöglicht. Darüber hinaus erlaubt Oryx einfachen Zugriff auf Modelle und vermeidet Versionskonflikte, da es sich dabei um eine webbasierte Lösung handelt.

Das Papier gliedert sich in folgende Abschnitte: Zunächst wird die Architektur von Oryx erläutert, bevor Abschnitt 3 die EPK-spezifischen Teile von Oryx einführt. Abschnitt 4 gibt einen Überblick über verwandte Arbeiten und Abschnitt 5 schließt das Papier mit Zusammenfassung und Ausblick ab.

## 2 Oryx-Systemarchitektur

Oryx ist ein webbasiertes Prozessmodellierungstool. Abbildung 1 zeigt einen Screenshot des Systems. Zu sehen ist, dass Oryx in einem Standardwebbrowser läuft und das Modell durch eine URL eindeutig identifiziert wird. Dieser Link kann an andere Modellierer verschickt werden, sodass diese einfachen Zugriff auf das Modell haben, ohne Software auf dem eigenen Rechner installieren zu müssen. Voraussetzung ist natürlich, dass entsprechende Zugriffsrechte für das Modell vergeben wurden. Diese sind nötig, da nicht jeder ein Prozessmodell einsehen können soll. Andere Modellierer können, wieder entsprechende

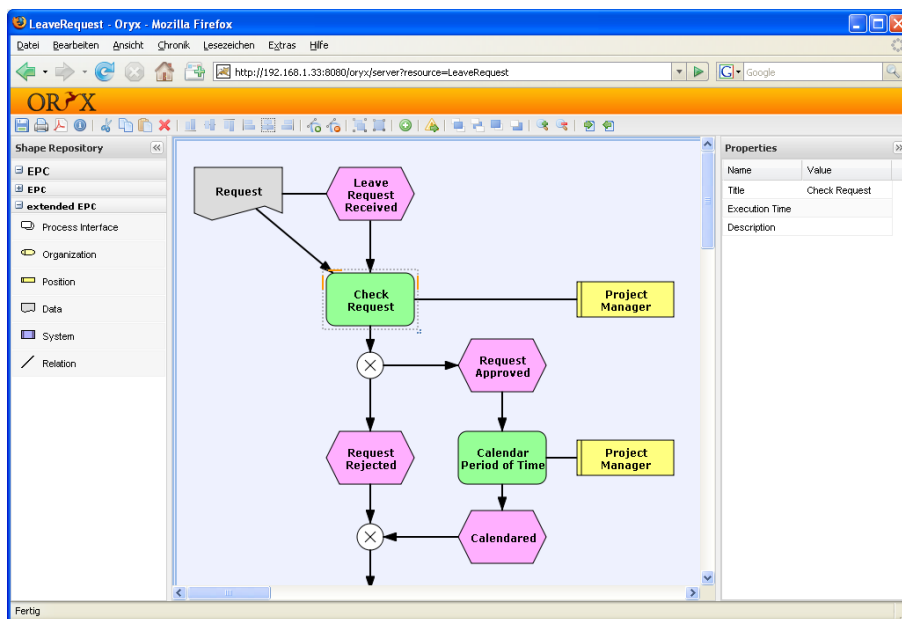


Abbildung 1: Oryx zur Modellierung einer EPK

Zugriffsrechte vorausgesetzt, das Modell modifizieren. Kollaboratives Modellieren über das Web ist somit möglich.

Oryx ist gerade für Szenarien geeignet, in denen Modellierer aus verschiedenen Organisationen auf die gleichen Modelle zugreifen müssen. Zum einen umgeht man mit Oryx das Problem, dass unterschiedliche (und möglicherweise inkompatible) Versionen von Software in den verschiedenen Organisationen installiert sind. Zum anderen besteht beim verteilten Zugriff nicht das übliche Problem, dass verschiedene Organisationen unterschiedliche Benutzerverwaltungen verwenden, oder dass man ein separates Benutzerkonto eigens für das Prozessmodellierungswerkzeug anlegen und pflegen muss. Dies ist möglich, da die Authentifizierung über OpenID [ope07] realisiert ist. Bestehende Benutzerkonten können somit wieder verwendet werden.

Oryx ist darauf ausgelegt, einfach sowohl um neue Notationen als auch um zusätzliche Funktionalität erweitert zu werden. Da er als Open-Source Projekt zur Verfügung steht, kann jede an Prozessmodellierung interessierte Organisation Weiterentwicklungen vornehmen. Hierbei profitieren vor allem Forscher, die für neue Modellierungsverfahren nicht mehr eigene Insel-Prototypen erstellen müssen, sondern sie innerhalb einer existierenden Lösung integrieren und erproben können.

Auf Grund seiner guten Erweiterbarkeit unterstützt Oryx neben EPKs auch weitere Modellierungssprachen wie die *Business Process Modeling Notation* (BPMN) in den Versionen 1.0 [bpm06] und 1.1 [bpm08], sowie Petri- und Workflow-Netze [vdAvH02]. Neben Standardmodellierungsfunktionen wie Copy&Paste, dem Ausrichten von Modellelementen oder dem Layouting von Kanten, werden auch notationsabhängige Funktionen angeboten. So können BPMN-Diagramme in Petri-Netze transformiert oder EPKs im- und exportiert werden. Diagramme einer jeden Notation lassen sich zudem auch als PDF oder PNG exportieren.

## 2.1 Genutzte Technologien

Um Oryx als Webseite in einem Browser anzeigen lassen zu können, ist die Verwendung verschiedener Webtechnologien nötig. Bevor jedoch auf diese eingegangen werden kann, soll zunächst die Architektur von Oryx vorgestellt werden (siehe Abbildung 2). Oryx besteht aus einem in den Browser geladenen Client, in dem das Modell editiert wird, und einem Backend, das eine Menge von Prozessmodellen vorhält.

Zur Anzeige des Editor-Clients werden vor allem JavaScript-Routinen verwendet, die als Teil eines Dokuments in einen Browser geladen werden. Dieses über eine URL adressierbare Dokument enthält ein Prozessmodell, das im Embedded RDF (eRDF [Dav06]) Format vorliegt. eRDF ermöglicht es, Metadaten direkt in ein HTML Dokument zu schreiben und daraus RDF zu extrahieren. Die Benutzung von RDF macht Oryx-Modelle einfach portierbar.

Die mit dem Prozessmodell mitgelieferten JavaScript Routinen sind dafür verantwortlich, die grafische Benutzerschnittstelle zu erzeugen und Modellierungsfunktionalität bereitzustellen. Modellelemente des aktuellen Diagramms finden sich als JavaScript-Objekte im

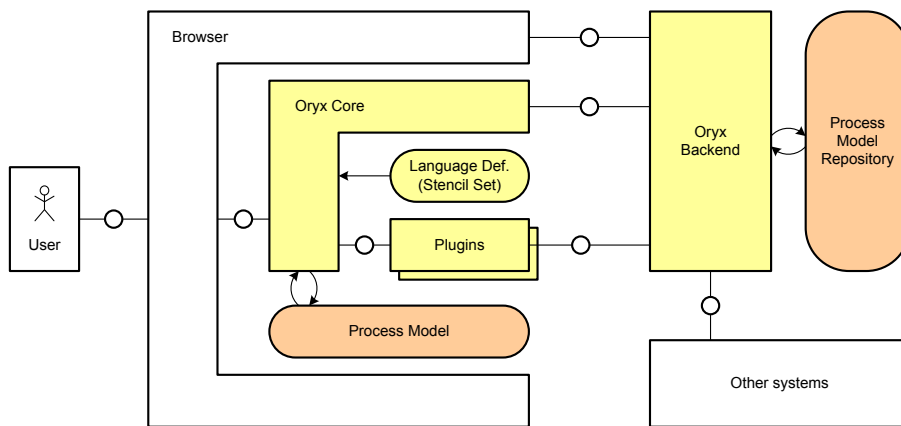


Abbildung 2: Oryx-Systemarchitektur

Browser wieder und können so manipuliert werden. Beim Speichern eines Modells werden die Elemente im eRDF Dokument asynchron an das Backend geschickt und dort in einer Datenbank gesichert.

Die Benutzerschnittstelle, die neben der Zeichenfläche aus Toolbar, Shape Repository und Property Pane besteht, ist mit Hilfe des ExtJS-Frameworks<sup>1</sup> erstellt. ExtJS ist eine JavaScript-UI-Bibliothek, die Widget-Typen für Rich Internet Application bereitstellt. Die Zeichenfläche enthält Modellelemente, die als Scalable Vector Graphics (SVG) in den Browser geladen werden.

Wie bereits beschrieben ist jedes Prozessmodell im Oryx durch eine URL eindeutig adressiert. Dadurch können Modelle durch bloßes Weiterreichen der URL ausgetauscht werden. Darüber hinaus können Modellressourcen unterschiedliche Repräsentationen aufweisen. So kann z.B. neben dem im Oryx editierbaren Dokument auch eine PDF-Repräsentation eines Prozesses angefordert werden.

Durch die OpenID-basierten Authentifizierung können Änderungen an Prozessmodellen – ähnlich wie in Wikis – nachverfolgt werden. Auf kritischen Ressourcen können zudem die Rechte, die ein Nutzer hat, eingeschränkt werden. Das System kennt hierbei die drei Rollen *Owner*, *Contributor* und *Viewer*, wobei ein Contributor ein Modell verändern und ein Viewer das Modell lediglich ansehen darf.

## 2.2 Stencil Set Konzept

Die Erweiterbarkeit ist eine der wichtigsten Eigenschaften von Oryx. Um verschiedene Notationen in Modellierungsprojekten zu unterstützen, wurde Oryx mit einem Mechanismus zum Auflösen beliebiger aus Knoten- und Kantentypen bestehender Modellierungs-

<sup>1</sup>Siehe <http://extjs.com>

sprachen versehen. Dabei wird eine solche Notation im Oryx durch ein so genanntes *Stencil Set* repräsentiert. Stencil Sets werden beim Öffnen eines Prozessmodells generisch in den Editor geladen und erzeugen dort z.B. Einträge im Shape Repository und in der Property Pane.

Kernstück eines Stencil Sets ist eine in der JavaScript Object Notation (JSON) verfassten Datei, die das Metamodell der Notation enthält. In ihr werden die Typen aller Modellelemente definiert. Jeder Elementtyp ist entweder Knoten oder Kante und verfügt über eine ID, einen Namen und eine Beschreibung. Zur grafischen Repräsentation existiert für jeden Typ eine SVG Datei, die als Vorlage für die modellierten Elemente genutzt wird.

Darüber hinaus verfügt ein Typ über eine Menge von Properties, die bei der Modellierung gesetzt werden können. Beispiel hierfür ist der Name eines EPK Ereignisses oder die URL eines durch eine Prozess-Schnittstelle referenzierten Subprozesses. Jede Property hat einen Datentypen und einen Standardwert. Properties können an XML-Knoten innerhalb der SVG Datei des gleichen Elementtyps gebunden werden. Dadurch erhalten die Propertywerte während der Modellierung Einfluss auf die grafische Erscheinung des Elements. Durch diesen Mechanismus wird zum Beispiel der Name einer EPK Funktion innerhalb des sie repräsentierenden abgerundeten Rechtecks angezeigt.

Um die Beziehungen, die zwischen Modellelementen bestehen dürfen, auszudrücken, können im Stencil Set Kompositionsregeln angegeben werden. Dabei gibt es Regeln, die festlegen, ob Elemente zweier Typen durch Knoten oder Kanten eines bestimmten Typs verbunden werden dürfen. So kann beispielsweise ausgedrückt werden, dass ein Ereignis mit einer Funktion, nicht aber mit einem anderen Ereignis verbunden werden darf. Außerdem kann geregelt werden, ob Elemente eines Typs in Elementen eines anderen Typs enthalten sein dürfen. Auf Grundlage dieser Bedingungen wird beim Modellieren geprüft, ob neue Beziehungen korrekt sind. Darüber hinaus werden aus den Verbindungsregeln Vorschläge für nachfolgende Elemente erzeugt, die dann durch Benutzung eines Kontextmenüs einfach zu erstellen sind.

Die Erweiterung von Oryx um eine zusätzliche Notation ist sehr einfach: Es muss nur ein neuer Unterordner im Stencil Set Verzeichnis angelegt werden. In diesem Ordner werden die JSON sowie die Icon- und SVG-Grafikdateien der neuen Notation gespeichert. Anschließend kann sofort damit begonnen werden, Modelle der neuen Notation zu erstellen.

### 2.3 Plugin Konzept

Neben der Erweiterbarkeit durch Stencil Sets kann Oryx auch funktional durch Plugins erweitert werden. Ein Plugin wird beim Laden von Oryx initialisiert und registriert dabei seine Funktionalität im Nutzerinterface. Durch Nutzerinteraktion kann die Funktionalität des Plugins angestoßen werden. Dabei hat es Zugriff auf die JavaScript-Objekte des aktuellen Prozessmodells sowie auf alle anderen geladenen Ressourcen.

Der Großteil der Modellierungsfunktionalität von Oryx ist durch Plugins implementiert: Während der Oryx Core aus Abbildung 2 für den Aufbau des Editor Layouts verantwortlich ist, werden Funktionen wie das Ausrichten von Modellelementen oder auch das Spei-

chern der Modelle durch austauschbare Plugins realisiert. Zudem sind auch das Shape Repository, die Property Pane und selbst die Toolbar, in der die meisten anderen Plugins vertreten sind, als Plugin implementiert.

Plugins werden in JavaScript programmiert. Außerdem muss ein Plugin in der *plugins.xml*-Datei eingetragen werden. Dabei kann auch spezifiziert werden, dass ein Plugin nur für Modelle eines bestimmten Stencil Sets geladen werden soll.

In den vergangenen Monaten wurden schon zahlreiche Plugins entwickelt: So existieren Im- und Exporter für Petri-Netze und ein Transformator, der BPMN in ausführbare Petri-Netze übersetzt. Auch im Bereich der EPKs werden Plugins genutzt: Ein Syntax-Check kann genutzt werden, um die modellierten EPKs zu überprüfen. Außerdem können diese in EPML exportiert und aus EPML importiert werden.

### 3 Modellierung von EPKs mit Oryx

Ereignisgesteuerte Prozessketten sind eine leicht verständliche, semi-formale Notation, die in den frühen 1990er Jahren entwickelt wurde [KNS92]. Sie können gut dazu genutzt werden, Unternehmensabläufe zu modellieren. Dabei nutzen EPKs bipartite Abfolgen von Ereignissen und Funktionen, die durch Konnektoren verzweigt und zusammen geführt werden können.

Im Folgenden wird gezeigt, welche Modellelementtypen existieren und in Oryx unterstützt werden. Außerdem werden spezielle Modellierungsfunktionalitäten für Syntax Checks sowie Im- und Exports vorgestellt.

#### 3.1 EPK Stencil Set

EPKs bestehen in erster Linie aus Ereignissen und Funktionen. Funktionen stellen aktive Elemente dar, die Aufgaben ausführen und dabei Daten erstellen oder verändern. Ereignisse sind passive Komponenten, die den erreichten Zustand innerhalb der aktuellen Prozesskette repräsentieren. Sie werden durch Funktionen erzeugt und verursachen ihrerseits die Ausführung weiterer Funktionen.

Außerdem existieren Konnektoren, die verwendet werden können, um Abläufe zu verzweigen oder wieder zusammenzuführen. Verzweigende Konnektoren haben einen Eingangs- und zwei Ausgangsflüsse, während zusammenführende Konnektoren zwei Eingänge und einen Ausgang haben. Dabei gibt es drei Typen: disjunktive (XOR), adjunktive (OR), und konjunktive (AND) Konnektoren.

Verzweigende ANDs modellieren, dass zur Laufzeit beide ausgehende Pfade abgearbeitet werden. XORs hingegen drücken aus, dass nur einer der beiden ausgehenden Pfade aktiviert wird. Die Entscheidung welcher der beiden Pfade das ist, wird durch das Auftreten eines der beiden nachfolgenden disjunktiven Ereignisse getroffen. Aus diesem Grund darf direkt nach einem verzweigenden XOR keine Funktion folgen. Das gleiche gilt für

verzweigende ORs. Hierbei ist es allerdings möglich, dass beide Ereignisse auftreten und daher auch beide Pfade abgearbeitet werden. Zusammenführende Konnektoren haben eine entsprechende Semantik: ANDs führen zwei ausgeführte Pfade zusammen, XORs vereinen zwei Pfade, von denen genau einer ausgeführt wurde und ORs tun dies mit zwei Pfaden, von denen wenigstens einer ausgeführt wurde.

All diese Standardkonstrukte [KNS92, Wes07] werden von Oryx unterstützt, wie Abbildung 3 zeigt. Dabei ist Oryx darauf ausgelegt, Diagramme zur Dokumentation existierender und neuer Prozesse zu ermöglichen und damit zum besseren Verständnis dieser beizutragen. Automatisch ausführbare Modelle sind hierbei nicht vorgesehen.

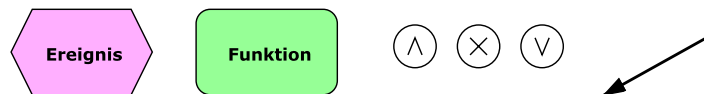


Abbildung 3: Übersicht der EPK Stencils

Ereignisse und Funktionen verfügen jeweils über eine Namen- und eine Beschreibungs-Property, die es dem Modellierer ermöglichen, den Modellelementen eine Bedeutung zuzuordnen. Im Zuge von EPK Modellierungsprojekten, die zusammen mit Industriepartnern durchgeführt wurden, ergab sich der Bedarf nach zusätzlichen Attributen. Diese Properties sind Werte, die vor allem aus bisherigen Ausführungen von Prozessen gewonnen werden konnten. So kann im Oryx angegeben werden, wie häufig ein bestimmtes Ereignis auftritt. Funktionen verfügen über eine Property, die angibt, wie lang die durchschnittliche Ausführungszeit ist und an Kanten können Ausführungswahrscheinlichkeiten annotiert werden. Dies ist vor allem für Kanten, die einer XOR Verzweigung entspringen, interessant und kann in Algorithmen genutzt werden, die die Komplexität von EPK verringern [PSW08b].

Neben den Standard-EPKs existieren noch erweiterte Ereignisgesteuerte Prozessketten (eEPKs). Diese zeichnen sich dadurch aus, dass in die normalen Abläufen auch Zuständigkeiten sowie datenbasierte Informationen integriert werden können. Während über die Menge an EPK Elementen Konsens herrscht, sind die Anzahl und Ausprägung von eEPK Elementen nicht festgeschrieben.



Abbildung 4: Übersicht der eEPK Stencils

Um das Modellieren einfach und übersichtlich zu halten, ist die Menge der unterstützten Elementtypen möglichst klein gehalten, wie Abbildung 4 zeigt. Dabei dient vor allem die *EPC Markup Language* (EPML [MN05]) als Orientierung.

Um verschiedene EPKs miteinander in Verbindung zu setzen, können Prozessschnittstellen genutzt werden. Diese können innerhalb oder am Ende einer EPK mit Hilfe von Sequenzfluss eingebunden werden und verweisen durch eine URL auf eine andere EPK, die

den Sub- oder Nachfolgeprozess modelliert. Zuständigkeiten lassen sich mit Hilfe von Stellen- oder im Falle von Prozessen zwischen verschiedenen Organisationen durch Organisationsknoten modellieren. Diese können mit Hilfe einer Relationsbeziehung an eine Funktion angebunden werden.

Auf die gleiche Art kann auch die Unterstützung oder Ausführung einer Funktion durch ein EDV-System modelliert werden. Letztendlich können Daten modelliert werden. Diese können sowohl mit Ereignissen als auch mit Funktionen in gerichteten oder ungerichteten Relationen auftauchen.

### 3.2 Syntax Checks

Neben der Auswahl von Modellelementen ist vor allem deren korrekte Kombination während der Modellierung von Bedeutung. Hierfür bietet Oryx eine zweistufige Überprüfung der Modelle während der Erstellung und zu einem späteren manuell gewählten Zeitpunkt als Unterstützung an.

Ein Stencil Set enthält, wie bereits erwähnt, neben Element-Definitionen auch Verknüpfungsregeln. So ist im EPK Stencil Set zum Beispiel festgelegt, dass auf ein Ereignis kein Ereignis folgen kann. Diese Regeln werden bei jeder Erstellung einer neuen Beziehung überprüft. Außerdem werden sie immer ausgewertet, wenn potenziell zu verbindende Elementtypen zu bestimmen sind (siehe Abbildung 5). Da diese Prüfung während des Modellierens vorgenommen wird, kann sie aus Performance-Gründen nur die lokale Umgebung einer Verbindung betrachten.

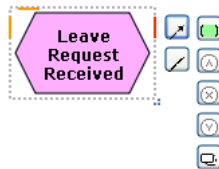


Abbildung 5: Vorschläge für nachfolgende Elemente im Oryx

Wie bereits in [MN03] ausgeführt, reicht eine solche lokale Betrachtung nicht aus um ein syntaktisch korrektes Modell garantieren zu können. Als Beispiel wollen wir die Überprüfung einer Kante von einem XOR Knoten zu einer Funktion anführen:

Ein Konnektor-Knoten kann in Abhängigkeit von der Anzahl ein- und ausgehender Kanten sowohl als zusammenführender als auch als verzweigender Konnektor genutzt werden. Während auf einen zusammenführenden XOR sowohl eine Funktion als auch ein Ereignis folgen darf, müssen auf einen verzweigenden XOR zwei Ereignisse folgen. Soll geprüft werden, ob der Konnektor mit einer nachfolgenden Funktion verknüpft werden darf, ist also das Wissen darüber von Nöten, ob der Konnektor verzweigend oder zusammenführend ist. Da dieses Wissen aber von anderen Verbindungsbeziehungen abhängt, die nicht mehr zur lokalen Umgebung der zu prüfenden Relation gehören, kann die Frage zu diesem Zeitpunkt nicht beantwortet werden.

Vollständige Syntax-Checks können zu einem späteren Zeitpunkt manuell oder zum Beispiel beim Speichern ausgelöst werden. Oryx bietet hierfür ein Framework an, das durch Nutzerinteraktion angestoßen einen Syntax-Check veranlasst und fehlerhafte Modellelemente markiert.

Dabei wird der Plugin-Mechanismus von Oryx genutzt, um den Syntax Check einzubinden. Das Plugin bietet dabei einen Button in der Toolbar an, an den eine Methode gebunden ist, die das in RDF serialisierte Prozessmodell an das Backend weiterreicht. Im Backend ist nun ein Java Servlet dafür verantwortlich, das RDF in eine Java Objektstruktur umzuwandeln. Anschließend kann die Objektstruktur umfassend auf syntaktische Fehler überprüft werden. Anders als in [GL06] passiert dies durch eine Java-Methode, die über alle Kanten und Knoten des Models iteriert und dabei die gesamte Umgebung des Elements für dessen Syntax-Check einbezieht. Neben dem Syntax-Checker für EPKs gibt es auch eine Implementierung BPMN Diagramme.

Die Benutzung einer Umwandlung in Java Objekte hat mehrere Vorteile: Das verwendete Metamodell und der Umwandler stellen wiederverwendbare Komponenten dar. So können sie auch für weitergehende semantische Überprüfungen - wie Soundness-Checks [Men07] - oder für die Implementierung von Transformationen in andere Notationen, z.B. in Petri-Netze, verwendet werden. Darüber hinaus ist die Performance eines deployten Servlets besser als die von lokal im Browser laufenden JavaScript Routinen.

Das Ergebnis der Überprüfung wird zurück an das JavaScript Plugin im Oryx geschickt, welches fehlerhafte Elemente markiert. Dabei wird neben der ID eines Elements auch ein beschreibender Text übermittelt, der als Tooltip mit in die Markierung übernommen wird.

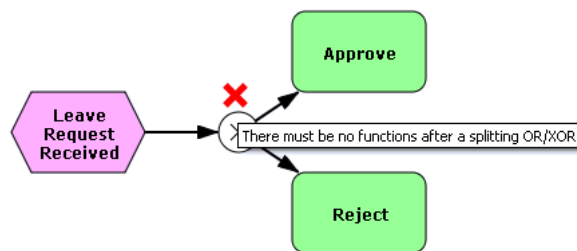


Abbildung 6: Markierung eines fehlerhaften Konnektors in Oryx

Abbildung 6 zeigt, wie der oben beschriebene Fall in einer nicht lokalen Syntax-Prüfung als Fehler erkannt und markiert werden kann. Durch einen weiteren Klick auf den Button in der Toolbar können die Markierungen wieder entfernt werden.

### 3.3 Im- und Exports

Damit mit Oryx modellierte EPKs auch in anderen Umgebungen genutzt werden können und vor allem auch um Ergebnisse bisheriger Arbeiten mit anderen Tools im Oryx fortzuführen, bietet Oryx Im- und Exportmöglichkeiten an.

Mit Hilfe eines weiteren Plugins wurden Funktionen in Oryx hinzugefügt, die einen Export nach EPML, bzw. einen Import aus EPML erlauben. EPML ist ein XML basiertes toolunabhängiges Austauschformat für EPKs [MN05]. Es kann unter Zuhilfenahme von XSL-Transformationen auch in AML, das ARIS-Austauschformat, transformiert werden [MN04].

Der Export eines Prozessmodells nach EPML wird durch den Nutzer per Button-Klick eingeleitet. Anschließend werden zwei Transformationen durchgeführt: Zunächst wird das im eRDF Format gespeicherte Modell in RDF transformiert. Hierfür kann ein gegebenes XSLT Stylesheet<sup>2</sup> verwendet werden. In einem zweiten Schritt muss dann das RDF in EPML umgewandelt werden, was wiederum durch ein XSLT Stylesheet realisiert wird. Die Umwandlung kann direkt Client-seitig im Plugin geschehen, da moderne Browser eigene XSLT-Engines anbieten. Die resultierende EPML Datei wird dem Benutzer in einem neuen Browsertab zur Verfügung gestellt.

Um eine EPK importieren zu können, muss sich der Nutzer ebenfalls im Oryx Editor befinden. Durch einen Button-Klick öffnet er einen Upload-Dialog, mit dem er eine EPML-Datei an ein Servlet im Backend schicken kann. Innerhalb des Servlets wird das EPML wieder mittels XSLT in eRDF umgewandelt. Das Ergebnis kann zurück an den Client gesendet werden, wo das Prozessmodell zur Anzeige kommt und evtl. durch den Nutzer gespeichert werden kann.

## 4 Verwandte Arbeiten

Marktführer auf dem Feld der EPK-Modellierungswerkzeuge ist das *ARIS Toolset* der IDS Scheer AG<sup>3</sup>. Dieses beruht auf dem Konzept der Architektur integrierter Informationssysteme (ARIS [Sch00]), das die Modellierung betriebswirtschaftlicher Abläufe unter den unterschiedlichen Gesichtspunkten der Organisation, der Daten, der Funktionen und der Steuerung vorsieht. Dabei werden diese auch auf verschiedenen Abstraktionsebenen zwischen Konzeptionierung und Implementation betrachtet.

Dementsprechend bietet das ARIS Toolset neben der Modellierung von EPKs auch weitere Notationen von Wertschöpfungsketten bis Organigrammen an. Außerdem existieren verschiedene Module, die auf die unterschiedlicher Benutzer zugeschnitten sind. So können Prozesse auf verschiedenen Abstraktionen modelliert und simuliert, aber auch implementiert und gewartet werden.

Durch eine Client-Server Architektur, können verschiedene Nutzer kollaborativ Modelle verwalten. Hierfür muss jedoch, anders als bei Oryx, die Client-Software auf jedem Rechner installiert werden, von dem ein Nutzer auf die Modelle zugreifen möchte.

Neben ARIS existieren aber auch weitere Tools, die im kommerziellen oder im wissenschaftlichen Bereich eingesetzt werden können, um EPKs zu modellieren. Mit *bflow*<sup>4</sup> entsteht zur Zeit eine Eclipse-basierte Geschäftsprozessmodellierungsumgebung, die, ähnlich

<sup>2</sup>Siehe <http://research.talis.com/2005/erdf/extract-rdf.xsl>

<sup>3</sup>Siehe <http://www.ids-scheer.com>

<sup>4</sup>Siehe <http://www.bflow.org/>

wie Oryx, als Open-Source-Projekt vor allem Forschern die Möglichkeit der Integration eigener Konzepte geben soll.

Ebenfalls auf Eclipse setzt das Werkzeug *EPC Tools*<sup>5</sup> auf. Dabei unterstützt es die Standard-EPK-Modellelemente, bietet aber auch Simulations- und Analysefunktionen an. Anders als Oryx und ARIS sind *bflow\** und *EPC Tools* Standalone Anwendungen, in denen man Modelle nur durch das Verschicken von Dateien oder die Integration von Versionsverwaltungs-Systemen austauschen kann.

Außer frei verfügbaren Shapes für das Modellierungstool *Visio*<sup>6</sup>, die das Erstellen von EPKs und eEPKs unterstützen, existiert mit *SemTalk* ein kommerziell vertriebener Visio-Aufsatz [FW01]<sup>7</sup>. Dieser bietet ähnlich wie Oryx eine Vielzahl von Notationen und dabei auch das Modellieren, den Austausch und die Analyse von EPKs an. *SemTalk* ist als Standalone Anwendung konzipiert, die aber auch Dokumente mit einem Repository abgleichen kann.

Auch wenn bereits Open-Source Modellierungstools in der EPK Community existieren, zeichnet sich Oryx vor allem durch seine Erweiterbarkeit um Notationen und Funktionalitäten sowie seinen kollaborativen Ansatz aus. Dadurch, dass er einfach als Webseite in einem Browser aufgerufen werden kann, ermöglicht Oryx einen im Vergleich zu kommerziellen Lösungen unkomplizierten und direkten Zugriff auf zentral hinterlegte Modelle.

## 5 Zusammenfassung und Ausblick

Dieses Papier hat EPK-Modellierung mit Oryx vorgestellt. Oryx ist eine technische Plattform, in der typische Funktionen eines grafischen Modellierungstools enthalten sind und die Erweiterungen einfach zulässt. Diese Erweiterungen können sowohl Spracherweiterungen als auch funktionale Erweiterungen sein. Syntax-Checks sind vorhanden und die Integration weiterer Verifikationsmechanismen ist geplant.

Als nächster Schritt steht die Integration von EPC-Soundness-Checks [Men07] an. Dies ermöglicht die Identifikation von unerreichbaren Ereignissen und Funktionen, sowie die Prüfung auf Abwesenheit von Deadlocks in den Modellen. Darüber hinaus ist die Integration von Prozessabstraktionsmechanismen geplant, die es ermöglichen, große Prozessmodelle auf Basis von Ausführungswahrscheinlichkeiten und durchschnittliche Ausführungszeiten automatisch zu reduzieren [PSW08a]. Außerdem wird zur Zeit an der Möglichkeit des eigenständigen Im- und Exports von ARIS Modellen, die im proprietären Format der AML ausgetauscht werden, gearbeitet.

Oryx steht allen Forschungsgruppen und Unternehmen offen, eigene Erweiterungen umzusetzen oder Funktionalität zu integrieren. In Kollaborationen mit anderen Universitäten hat sich bereits gezeigt, dass dies tatsächlich mit geringem Aufwand möglich ist und dass so neue Funktionalität schnell einem größeren Publikum zugänglich gemacht werden kann.

<sup>5</sup>Siehe <http://www.cs.uni-paderborn.de/cs/kindler/research/EPCTools/>

<sup>6</sup>Siehe <http://office.microsoft.com/visio/>

<sup>7</sup>Siehe <http://www.semtalk.com/>

**Danksagungen.** Wir möchten uns beim Oryx-Team für die umfangreichen Entwicklungen bedanken.

## Literatur

- [bpm06] Business Process Modeling Notation (BPMN) Specification, Final Adopted Specification. Bericht, Object Management Group (OMG), February 2006.
- [bpm08] Business Process Modeling Notation, V1.1. Bericht, Object Management Group (OMG), Jan 2008. <http://www.omg.org/spec/BPMN/1.1/PDF/>.
- [Dav06] Ian Davis. RDF in HTML. Bericht, Talis Information Limited, 2006. <http://research.talis.com/2005/erdf/wiki/Main/RdfInHtml>.
- [FW01] Christian Fillies und Frauke Weichhardt. SemTalk: A RDFS Editor for Visio 2000. Bericht, SC4, July 2001. <http://www.semtalk.com/pub/swws.htm>.
- [GL06] Volker Gruhn und Ralf Laue. Validierung syntaktischer und anderer EPK-Eigenschaften mit PROLOG. In Markus Nüttgens, Frank J. Rump und Jan Mendling, Hrsg., *Proc. of the 5th GI Workshop on Event-Driven Process Chains (EPK 2006)*, Seiten 69–84. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, 2006.
- [KNS92] Gerhard Keller, Markus Nüttgens und August-Wilhelm Scheer. Semantische Prozessmodellierung auf der Grundlage "Ereignisgesteuerter Prozessketten (EPK)". Bericht Heft 89, Institut für Wirtschaftsinformatik Universität Saarbrücken, 1992.
- [Men07] Jan Mendling. *Detection and Prediction of Errors in EPC Business Process Models*. Dissertation, Vienna University of Economics and Business Administration, 2007.
- [MN03] Jan Mendling und Markus Nüttgens. EPC Syntax Validation with XML Schema Languages. In Markus Nüttgens und Frank J. Rump, Hrsg., *Proc. of the 2nd GI Workshop on Event-Driven Process Chains (EPK 2003)*, Seiten 19–30. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, 2003.
- [MN04] Jan Mendling und Markus Nüttgens. Transformation of ARIS Markup Language to EPML. In M. Nüttgens und F.J. Rump, Hrsg., *Proc. of the 3rd GI Workshop on Event-Driven Process Chains (EPK 2004)*, Seiten 27–38. GI-Arbeitskreis Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, 2004.
- [MN05] Jan Mendling und Markus Nüttgens. EPC Markup Language (EPML): An XML-based Interchange Format for Event-Driven Process Chains (EPC). Bericht, Vienna University of Economics and Business Administration, July 2005.
- [ope07] OpenID Authentication 2.0 - Final. Bericht, OpenID Foundation, December 2007. [http://openid.net/specs/openid-authentication-2\\_0.html](http://openid.net/specs/openid-authentication-2_0.html).
- [PSW08a] Artem Polyvyanyy, Sergey Smirnov und Mathias Weske. Process Model Abstraction: A Slider Approach. In *EDOC 2008: Proceedings of the 12th IEEE International Enterprise Distributed Object Computing Conference*. IEEE Computer Society, 2008.
- [PSW08b] Artem Polyvyanyy, Sergey Smirnov und Mathias Weske. Reducing the Complexity of Large EPCs. Bericht 22, Hasso-Plattner-Institute, 2008.

- [Sch00] August-Wilhelm Scheer. *ARIS Business Process Modeling*. Springer Verlag, 2000.
- [vdAvH02] Wil van der Aalst und Kees van Hee. *Workflow Management: Models, Methods, and Systems (Cooperative Information Systems)*. The MIT Press, January 2002.
- [Wes07] Mathias Weske. *Business Process Management*. Springer, 2007.