

BPMN-Community: Anwendung der Wiki-Prinzipien

Emilia Wittmers

Betreuer:

Prof. Dr. Mathias Weske

MSc. Alexander Großkopf

Lehrstuhl für Business Process Technology

Datum der Abgabe: 30. Juni 2009

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Postdam, den 30. Juni 2009

Die BPMN-Community ist eine Internetplattform zum kollaborativen Modellieren von Prozessmodellen. Sie orientiert sich an bestehenden Wiki-Konzepten und den Prinzipien der Wikinomie. Der Begriff 'Offenheit' spielt in diesem Zusammenhang eine wichtige Rolle: Damit Prozessmodelle gemeinsam entwickelt werden können, muss jeder Nutzer alle Inhalte lesen, kommentieren und bearbeiten dürfen. Die folgende Arbeit beschäftigt sich mit der Anwendung der Wiki- und Wikinomics-Prinzipien auf die Kollaborationsplattform BPMN-Community. Es werden sowohl die Ideen hinter den Konzepten als auch deren technische Realisierung betrachtet.

Inhaltsverzeichnis

1	Einleitung	1
2	Grundlegende Begriffe	3
2.1	Die Wiki-Prinzipien	3
2.2	Wikinomics	4
3	Umsetzung der Wiki- und Wikinomics-Prinzipien	6
3.1	Die Wiki-Prinzipien in der BPMN-Community	6
3.1.1	Offen	6
3.1.2	Schnell und unkompliziert	7
3.1.3	Kommunikativ	7
3.1.4	Sicher	8
3.1.5	Verlinkt	9
3.2	Wikinomics in der BPMN-Community	10
3.2.1	Offenheit, Freiwillige Zusammenarbeit, Teilen	10
3.2.2	Globales Handeln	10
4	Anwendung der Wiki- und Wikinomics-Prinzipien	11
4.1	Anwendungsszenario - Tutorials	11
4.1.1	BPMN Lernen	11
4.1.2	BPMN Lehren	13
4.2	Django App Tutorials	14
4.2.1	Models	15
4.2.2	Views	17
4.2.3	Templates	19
5	Das Wiki in der BPMN-Community	21
5.1	Das Wiki-Objekt und seine Versionierung	21
5.2	YUI Library: Der Rich Text Editor	25
5.2.1	Das Einbinden des Rich Text Editors	25
5.2.2	Die Funktion loadEditor()	26
5.2.3	Die Funktion addProcessModelButton()	29
5.3	Der Wikifilter	33
6	Zusammenfassung und Ausblick	34

Abbildungsverzeichnis

1	Screenshot der Kollaborationsplattform BPMN-Community	1
2	UML Use-Case-Diagramm BPMN Lernen	12
3	UML Use-Case-Diagramm BPMN Lehren	12
4	UML Klassendiagramm Tutorials	15
5	UML Klassendiagramm Wiki	21
6	Screenshot des erweiterten Rich Text Editors	29

1 Einleitung

Die BPMN¹-Community ist eine Kollaborationsplattform zum gemeinsamen Erarbeiten von Prozessmodellen.

Sie dient dem Austausch von Erfahrungen und soll sowohl BPMN-Anfängern als auch BPMN-Experten die Möglichkeit bieten, sich als Mitglied einer Community zusammen mit anderen Interessierten mit dem Thema der Prozessmodellierung auseinanderzusetzen.

Die Plattform besteht aus den vier Bereichen: Tutorials (BPMN lernen), Best Practices² (Best Practices diskutieren), Gruppen (Kollaborativ modellieren) und Forum (Wissen austauschen).



Abbildung 1: Screenshot der Kollaborationsplattform BPMN-Community

¹Business Process Modeling Notation: eine Modellierungsnotation für Geschäftsprozesse

²Ein 'Best Practice' beschreibt die sinnvolle Verwendung bestimmter BPMN-Konstrukte/ Patterns in gewissen Szenarien. Die technische Umsetzung der Best Practices in der BPMN-Community ist in der Bachelorarbeit von Markus Günter beschrieben [Günter, 2009].

Die BPMN-Community lebt von den Inhalten, die ihre Nutzer selbst erzeugen. Wie wertvoll diese Inhalte sind, entscheidet die Gemeinschaft. Jeder Nutzer hat die Möglichkeit eigene Texte und Modelle zu erstellen. Er kann aber auch Beiträge anderer Nutzer kommentieren und Wertungen dazu abgeben oder sie einfach sofort selbst bearbeiten, um so mitzuhelfen, diese zu verbessern. Der Ansatz „Jeder kann alles bearbeiten.“ gehört zu den wichtigsten Grundgedanken des Wiki-Konzepts. Jeder Leser kann gleichzeitig Autor sein. Um erfolgreich in einer großen Community arbeiten zu können, müssen bestimmte Faktoren gegeben sein: 'Being Open', 'Peering', 'Sharing' und 'Acting Globally', die Prinzipien der Wikinomie, sind wichtige Voraussetzungen für das Gelingen einer Kollaborationsplattform.

Das zentrale Thema dieser Bachelorarbeit ist die Anwendung der Wiki- und Wikinomics-Prinzipien auf die BPMN-Community. Die Arbeit soll die Ideen hinter den Konzepten, ihre Umsetzung und die technische Realisierung dokumentieren, um bei der stetigen Weiterentwicklung und Verbesserung der Plattform beizutragen und helfend zu unterstützen.

Im folgenden Kapitel, Kapitel 2, wird zunächst erklärt, was die 'Wiki-Prinzipien' sind und was sich hinter dem Begriff 'Wikinomics' verbirgt. Anschließend geht es in Kapitel 3 darum, wie die zuvor erläuterten Konzepte in der BPMN-Community umgesetzt wurden. Es wird beschrieben, an welchen Stellen der Plattform die Wiki- und Wikinomics-Prinzipien wiederzufinden sind.

Anhand der Beispielszenarien 'BPMN Lernen' und 'BPMN Lehren' wird in Kapitel 4 die Anwendung der Wiki-Ideen im Bereich 'Tutorials' dargestellt. Im Anschluss an diese Betrachtung folgt ein Überblick über die Implementierung der Tutorial-Applikation.

Kapitel 5 beschäftigt sich mit der technischen Realisierung der Wiki-Texte in der BPMN-Community, die ebenfalls den Wiki- und Wikinomics-Prinzipien folgen. Es wird dokumentiert, wie Texte auf der Plattform gespeichert und versioniert werden und wie der Rich Text Editor der Yahoo! User Interface Library [Yahoo!, 2009], mit dem der Nutzer Texte auf der Plattform editieren kann, eingebunden, angepasst und um einen Button zum Einfügen von Prozessmodellen erweitert worden ist.

2 Grundlegende Begriffe

In den ersten zwei Monaten der Ideenfindung ergaben sich einige Diskussionspunkte über die Konzepte der Rechteverwaltung und das Modellieren in geschlossenen Gruppen. Durch Befragungen potenzieller Nutzer stellte sich jedoch bald heraus, dass vielmehr eine offene als eine geschlossene Plattform gewünscht wird.

Eine Art 'Wikipedia für Prozessmodellierung' lautete der Wunsch von Kunden.

Die gemeinsame Entwicklung von Best Practices und die Möglichkeit des Erlernens von BPMN-Konstrukten sollte dabei im Vordergrund stehen.

Das Kriterium der Offenheit implizierte die Auseinandersetzung mit bestehenden Wiki-Gedanken.

In diesem Kapitel werden die Wiki- und Wikinomics-Prinzipien erläutert, an denen sich die zu entwickelnde Kollaborationsplattform BPMN-Community orientieren sollte.

2.1 Die Wiki-Prinzipien

Wie schafften es Wikis so erfolgreich zu werden und sich in der gesamten Online-Community durchzusetzen?

Fünf einfache Prinzipien garantierten ihnen den großen Durchbruch:

Offen, schnell und unkompliziert, kommunikativ, sicher und verlinkt lauten die erfolgreichsten Adjektive.

Offen Alle Inhalte dürfen von allen Nutzern gelesen und von angemeldeten Nutzern sogar editiert werden. Es gilt: Jeder kann alles bearbeiten; die Plattform ist für jedermann offen.

Schnell und unkompliziert Inhalte sollen mühelos und durch einen minimalen Aufwand des Nutzers erzeugt werden können. Der Nutzer muss schnell verstehen können, wie er Inhalte erschaffen kann. Die Seiten der Plattform sollten nicht mit 'Spielereien' überladen sein, sondern nur die nötigsten Funktionen bieten.

Kommunikativ Es muss die Möglichkeit zur Diskussion geboten werden, die den Austausch und das Entstehen unterschiedlicher Meinungsbilder fördert.

Sicher Da jeder alles bearbeiten kann, muss gewährleistet sein, dass geänderte Daten notfalls wiederhergestellt werden können. Die Beiträge sollen den jeweiligen Verfassern zuzuordnen sein. Gleichzeitiges Bearbeiten von Inhalten sollte verhindert werden, um eventuelle Datenverluste zu vermeiden.

Verlinkt Das Anlegen neuer Seiten auf der Plattform soll problemlos funktionieren. Die Seiten sollen untereinander verlinkt und Links automatisch generiert werden können.

2.2 Wikinomics

Der Neologismus 'Wikinomics' wurde von dem kanadischen Autor Don Tapscott und seinem Kollegen Anthony D. Williams geprägt. Sie beschreiben in ihrem Buch „Wikinomics: How Mass Collaboration Changes Everything“ eine neu aufkommende Form der wirtschaftlichen Zusammenarbeit. [Tapscott and Williams, 2006]

Selbstorganisiert und über alle Kontinente verteilt arbeiten viele Menschen zusammen an einem gemeinsamen Projekt. Das Wissen und die innovativen Ideen externer Talente und Experten werden auf einer Online-Plattform gesammelt und ausgetauscht.

Es sind die vier Faktoren *Offenheit, Freiwillige Zusammenarbeit, Teilen* und *Globales Handeln*, die den Begriff 'Wikinomics' ausmachen. Durch das Konzept „Gemeinsam mehr wissen“ können Projekte auf diese Art und Weise schnell und kostengünstig vorangetrieben werden.

Offenheit Oftmals investieren Unternehmen sehr viel Kraft in das Internhalten ihres Firmenwissens, wodurch sie jedoch externe Ressourcen, die ihnen einen deutlichen Wettbewerbsvorteil verschaffen könnten, von sich fernhalten.

'Being Open' bedeutet, sich nicht vor den Ideen und innovativen Gedanken anderer zu verschließen, sondern sie zu nutzen, indem man das Unternehmen für die Leute transparent macht und ihnen die Möglichkeit gibt, sie durch ihre Kompetenzen an der Weiterentwicklung mitwirken und teilhaben zu lassen.

Freiwillige Zusammenarbeit 'Peering' bezeichnet das Gegenteil von hierarchisch organisierten Arbeitsstrukturen und bedeutet freiwilliges Zusammenarbeiten in einer horizontalen Organisationsform.

Getrieben von verschiedenen Motivationen, wie Spaß oder das sich selbst darstellen und profilieren wollen, können Menschen ihre Kreativität entfalten. Selbstorganisiert, zwanglos und aus eigenem Antrieb wird effektiv zusammengearbeitet.

Teilen Wird Geistiges Eigentum geteilt, entsteht ein Mehrwert, indem andere auf Basis dieser Gedanken ihre eigenen entwickeln können. 'Sharing' heißt, bestimmtes Wissen öffentlich zu stellen, um ideenreichen Köpfen die Möglichkeit zu geben, den Markt durch ihre eigenen Sichtweisen und ihre produktiven Einfälle zu erweitern.

Globales Handeln Experten sind überall auf der Welt zu finden. Um ihre Talente nutzen zu können, muss man sich jedoch dem globalen Markt öffnen. Nur durch globales Handeln erreicht man all die über alle Kontinente verteilten Ressourcen und Potenziale.

'Acting Globally' heißt, Grenzen aus dem Weg zu schaffen und das gemeinsame Arbeiten unabhängig von Raum und Ort zu ermöglichen.

3 Umsetzung der Wiki- und Wikinomics-Prinzipien

Aus den Prinzipien, die im vorangegangenen Kapitel vorgestellt worden sind, ergeben sich für eine Plattform, die diesen Ansätzen folgen möchte, gewisse technische und konzeptionelle Anforderungen.

Wie die Wiki- und Wikinomics-Prinzipien in der BPMN-Community umgesetzt worden sind, wird in dem folgenden Abschnitt der Arbeit beschrieben.

3.1 Die Wiki-Prinzipien in der BPMN-Community

3.1.1 Offen

In der BPMN-Community kann jeder Nutzer alle Inhalte lesen. Nicht nur die Tutorial-Texte, die modellierten Prozesse der Best Practices und die Forendiskussionen, auch die Gruppenbeiträge, Kommentare und die Profelseiten anderer Nutzer sind öffentlich einsehbar.

Bis auf einige wenige Ausnahmereiche³ kann auch jeder alles bearbeiten.

Um die Inhalte editieren zu können, muss man sich auf der Plattform registrieren. Hierbei genügt es, seine OpenID⁴ anzugeben, beziehungsweise den Vor- und Nachnamen, die E-Mail-Adresse und ein selbstgewähltes Passwort.

Die Registrierung ist vollkommen kostenfrei und dient lediglich der Zuordnung zwischen den Aktivitäten auf der Plattform und den Nutzern. Auch die Authentifizierung gegenüber dem Oryx-Editor⁵, der zum Modellieren der Prozessmodelle verwendet wird, wird durch diese Angaben des Nutzers sichergestellt.

Mit dem Thema 'Benutzerauthentifizierung' beschäftigt sich die Bachelorarbeit 'BPMN-Community: Benutzerverwaltung und Identitätsmanagement' von Tobias Rawald. [Rawald, 2009]

³Profildaten, Kommentare, private Lösungen zu Tutorialübungen und Forendiskussionsbeiträge fremder Nutzer können natürlich nicht bearbeitet werden.

⁴Das einmalige Registrieren bei einem OpenID-Provider mit Nutzernamen und Passwort erlaubt es dem Nutzer, sich auf anderen Plattformen ohne diese Angaben anzumelden.

⁵<http://bpt.hpi.uni-potsdam.de/Oryx>

3.1.2 Schnell und unkompliziert

Inhalte lassen sich in der BPMN-Community schnell und unkompliziert erzeugen und auch bearbeiten.

Zum Anlegen eines neuen Tutorials, eines Best Practices, einer Gruppe etc. muss nur ein Titel angegeben werden und ein anschließender Bestätigungsklick auf den Erstell-Button erfolgen. Das jeweilige Template wird dann automatisch generiert und kann auch sofort editiert werden.

Durch das Konzept des Inplace Editing, mit dem sich unter anderem die Bachelorarbeit 'BPMN-Community: Frontend-Technologien und die Model Viewer API' von Jan-Felix Schwarz [Schwarz, 2009] auseinandersetzt, ist das Bearbeiten von Inhalten an Ort und Stelle möglich, was die Benutzerbedienung deutlich vereinfacht. Durch einen Klick auf das Editier-Icon wird der Rich Text Editor dynamisch geladen und der jeweilige Text kann bearbeitet werden. Der Rich Text Editor ist übersichtlich gestaltet und bietet nur wenige Formatierungsmöglichkeiten an, um auch die Inhalte auf der Plattform übersichtlich zu halten.

Aber nicht nur Texte können schnell und einfach bearbeitet werden. Prozessmodelle lassen sich ebenfalls problemlos editieren. Möchte der Nutzer ein Modell ändern, wird er sofort zum Oryx-Editor weitergeleitet ohne dass er sich dort erneut anmelden muss. Der Oryx-Editor, mit dem der Nutzer die Prozessmodelle bearbeiten kann, wird in einem neuen Tab oder einer neuen Seite des Browsers geöffnet.

3.1.3 Kommunikativ

Als angemeldeter Nutzer kann man zu jedem Bereich in der BPMN-Community Fragen stellen und Anmerkungen schreiben. Dies wird durch die Kommentarfunktion in der Seitenleiste ermöglicht.

Völlig unkompliziert lassen sich dort Kommentare und Antworten auf Kommentare verfassen. Die weiterführende Diskussion zu einem Kommentar wird zur Erhaltung der Seitenübersichtlichkeit ausgeblendet und kann jederzeit eingeblendet werden. Die aktuellste Antwort wird an oberster Stelle angezeigt.

Bei Prozessmodellen können Anmerkungen auch direkt an die Modellelemente angeheftet werden. Diese Funktion wird mit Hilfe der Model Viewer API umgesetzt, die in der oben genannten Arbeit von Jan-Felix Schwarz beschrieben wird.

Kommentare, die zu Prozessmodellelementen gehören, werden sowohl im Modell

selbst dargestellt als auch in der Seitenleiste angezeigt.

Neben dem Weg eine für alle Nutzer nachlesbare offene Diskussion über die Kommentarfunktion zu führen, gibt es auch die Möglichkeit mit anderen angemeldeten Plattformteilnehmern direkt in einem offenen Chatraum seine Meinung auszutauschen. Das Thema der verschiedenen Kommunikationswege in der BPMN-Community wird in der Bachelorarbeit 'BPMN-Community: Konzept und Implementierung der Kommunikation' von Christian Wiggert beschrieben. [Wiggert, 2009]

Allgemeine Fragen, die viele Nutzer interessieren könnten, sollten im Forum gestellt und beantwortet werden. Das Konzept und die Umsetzung des Forums sind in der Arbeit 'BPMN-Community: konzeptionelle und technische Realisierung von Anforderungen' von Markus Güntert nachzulesen. [Güntert, 2009]

Es ist wichtig, dass den Nutzern hinreichende Möglichkeiten zum öffentlichen Diskutieren und Fragenstellen gegeben werden, um die Qualität der Beiträge und der gesamten Plattform zu sichern.

3.1.4 Sicher

Die Qualität der Nutzerbeiträge wird in der BPMN-Community nicht nur durch direkte Kommunikation sichergestellt, sondern auch durch ein Rating-System.

Prozessmodelle können von den Nutzern in den Kategorien 'Verständlich', 'Einfach analysierbar', 'Akurat' und 'Insgesamt' bewertet werden. In diesen vier Kategorien kann jeder angemeldete Nutzer jeweils bis zu fünf Sterne vergeben. Wird das Prozessmodell geändert, sodass sich dessen Qualität eventuell erhöht beziehungsweise verringert, kann die Bewertung des Nutzers dementsprechend auch noch einmal angepasst werden.

Änderungen an Prozessmodellen und Wiki-Texten werden versioniert.

Beim Speichern eines editierten Textes wird nicht nur der gesamte Text in der Datenbank abgelegt, sondern auch ein Changeset, also die Informationen, welche Stellen im Vergleich zur Version davor geändert worden sind. Der Nutzer kann sich die älteren Versionen ansehen und so die Änderungen nachvollziehen. Hinzugefügte Abschnitte werden grün dargestellt und gelöschte Worte rot markiert.

Sind Textabschnitte nicht vernünftig oder sachgemäß editiert worden, lassen sich die vorherigen Versionen wiederherstellen.

Beim Speichern eines editierten Prozessmodells wird ein neues Oryx-Modell mit eigener URL angelegt. Jede Revision des Prozessmodells kennt die Oryx-URL, unter

der das jeweilige BPMN-Modell zu finden ist.

Durch die verschiedenen Versionen des Prozessmodells lässt sich mit Hilfe der 'Time Mashine' scrollen. Zusätzlich können ebenfalls auch zwei Versionen miteinander verglichen werden. Unterschiede zwischen den zu vergleichenden Modellen werden farblich gekennzeichnet.

Wird ein Prozessmodell von zwei Nutzern gleichzeitig editiert, wird als Head-Revision des Prozessmodells das Modell von demjenigen vermerkt, der zuletzt gespeichert hat. Das Modell des anderen ist dann als vorherige Version im System abgelegt. Eine Art Merge-Vorgang, bei dem die verschiedenen Versionen zu einem Modell zusammengeführt werden, gibt es nicht.

Das gleichzeitige Bearbeiten eines Wiki-Textes wird dagegen durch das Setzen eines Locks verhindert. Der Lock wird erst gelöst, wenn der Nutzer, der gerade an dem Text arbeitet, das Editieren abgeschlossen hat oder nachdem seine Sitzung nach einer gewissen Zeit⁶ automatisch beendet worden ist.

3.1.5 Verlinkt

Wie in Abschnitt 3.1.2 bereits beschrieben, ist das Anlegen von neuen Inhalten in der BPMN-Community für den Nutzer leicht verständlich umgesetzt worden. Möchte er beispielsweise zu einem Best Practice einen Vorschlag hinzufügen, muss er lediglich auf einen der beiden Buttons, 'neuer Good Practice Vorschlag' oder 'neuer Bad Practice Vorschlag', klicken. Das jeweils gewählte Element wird dem Best Practice automatisch hinzugefügt und die dazugehörigen Seiten der Plattform werden miteinander verlinkt.

In den Wiki-Texten kann der Nutzer mit Hilfe des Link-Einfüge-Buttons Links setzen. Eine automatische Erkennung von Web-Adressen im Rich Text Editor ist derzeit jedoch nicht möglich.

⁶Der Wert für das zeitliche Ablaufen eines Locks liegt aktuell bei 15 Minuten.

3.2 Wikinomics in der BPMN-Community

3.2.1 Offenheit, Freiwillige Zusammenarbeit, Teilen

In der BPMN-Community kann jeder mitmachen, der mitmachen möchte. Sowohl Anfänger, die auf der Suche nach Antworten auf ihre Fragen sind, als auch Experten, die als Spezialisten in ihren Fachgebieten gelten, sind dazu eingeladen, sich in der Gemeinschaft miteinzubringen.

Lebendigkeit entsteht auf der Plattform nur dann, wenn beide Gruppen zusammen arbeiten.

Hierarchie-Ebenen gibt es in der BPMN-Community nicht. Jeder Teilnehmer hat dieselben Rechte. Lediglich die Administratoren der Website haben zusätzlich noch die Möglichkeit, Inhalte, die unsinnig oder rechtswidrig sind, zu löschen. Die Gemeinschaft organisiert sich selbst. Jeder ist dazu aufgerufen die Inhalte der anderen zu kontrollieren und mitzuhelfen, sie durch sein Wissen und seine Erfahrungen, die er bereit ist, mit der Öffentlichkeit zu teilen, zu verbessern.

3.2.2 Globales Handeln

Die BPMN-Community ist ursprünglich als deutschsprachige Plattform konzipiert worden. Sie sollte das im deutschsprachigen Raum rare Angebot an Tutorials und Best-Practice-Diskussionen im Bereich der BPMN erweitern.

Die Möglichkeit zur späteren Internationalisierung der Kollaborationsplattform ist jedoch von Anfang an mitbedacht worden. Die Verwendung des Django Template Tags 'trans'⁷ ermöglichte schon im frühen Entwicklungsstadium der BPMN-Community die Übersetzung von Beschreibungstexten. Eine Übersetzung der von den Nutzern erzeugten Beiträge wurde jedoch erst umgesetzt, nachdem sich bereits einige Nutzer aus nicht-deutschsprachigen Regionen der Welt, wie zum Beispiel Nordamerika, Schottland und Belgien, auf der Plattform registriert hatten.

Die Internationalisierung der Inhalte erfolgt durch die Verwendung mehrerer Wiki-Objekte, denen die jeweilige Sprache zugeordnet ist. Details zur Umsetzung der Übersetzung sind in der Bachelorarbeit 'BPMN-Community: Architektur und Erweiterungsmöglichkeiten' von Stefan Wehrmeyer beschrieben. [Wehrmeyer, 2009]

⁷Django Dokumentation: <http://docs.djangoproject.com/en/dev/topics/i18n/>

4 Anwendung der Wiki- und Wikinomics-Prinzipien

Tutorials gehören neben den Best Practices zu den wichtigsten Anwendungsszenarien auf der Kollaborationsplattform. BPMN-Anfänger können zu den auf den Tutorial-Seiten erklärten Konstrukten Übungsaufgaben lösen und ihre Lösungen mit BPMN-Experten diskutieren. Im Folgenden der Arbeit werden daher einige konkrete Anwendungsfälle der in der BPMN-Community umgesetzten Wiki- und Wikinomics-Prinzipien am Beispiel der Tutorials gezeigt.

Im Anschluss an diese Betrachtung folgt eine Dokumentation technischer Implementierungsdetails der Tutorial-Applikation.

4.1 Anwendungsszenario - Tutorials

Tutorials, in denen BPMN-Konstrukte erklärt werden, sollen von den Mitgliedern der Plattform gemeinsam entwickelt werden. Hierbei kommt es zu einer Zusammenarbeit zwischen BPMN-lernenden und BPMN-lehrenden Nutzern. Offenheit und die Bereitschaft zum Teilen von Wissen und Erfahrungen sind für das Funktionieren dieser Zusammenarbeit wichtige Grundvoraussetzungen.

Das gemeinsam entwickelte Wissen steht jedem Internetnutzer offen zur Verfügung. Um ein Tutorial lesen zu können und die Übungsaufgaben zu bearbeiten, braucht man kein angemeldetes Community-Mitglied sein. Zum Editieren der Inhalte ist es jedoch notwendig, sich auf der Plattform zu registrieren. (*Siehe Abbildung 2 und Abbildung 3*)

4.1.1 BPMN Lernen

Ein Student, der am Hasso-Plattner-Institut für IT-Systems-Engineering die Vorlesung 'Prozessorientierte Informationssysteme I' hört, sucht zur Klausurvorbereitung im Internet nach Informationen zu einem bestimmten BPMN-Konstrukt und wird auf der Plattform BPMN-Community fündig. Hier wird ein Tutorial speziell zu diesem Thema angeboten.

Der Student liest sich die Erklärung zu den BPMN-Elementen durch (*offen*) und entdeckt, dass es auch eine Übungsaufgabe zu dem erklärten BPMN-Konstrukt gibt.

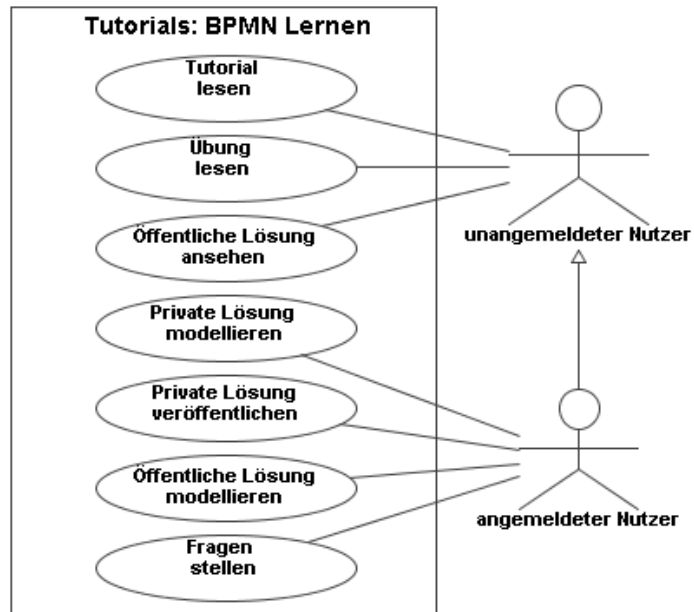


Abbildung 2: UML Use-Case-Diagramm BPMN Lernen

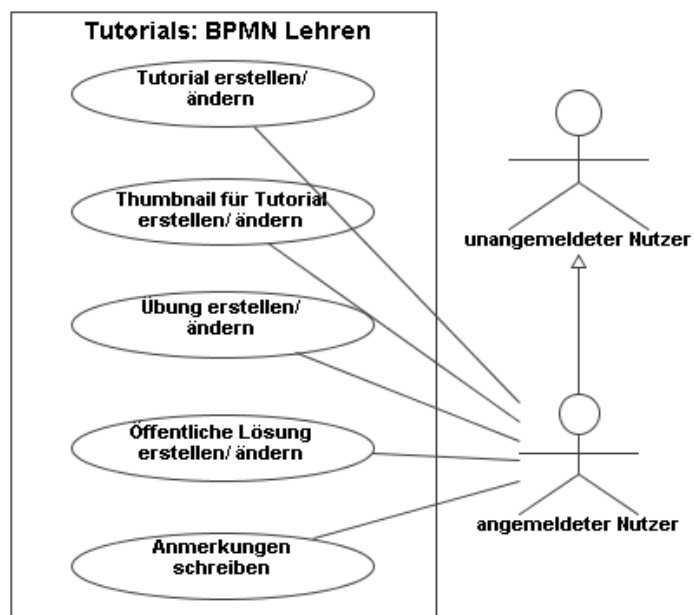


Abbildung 3: UML Use-Case-Diagramm BPMN Lehren

Die Übungsaufgabe könnte er mit dem Demo Oryx-Editor oder mit Bleistift und Papier lösen. Er kann sie aber auch direkt auf der Plattform modellieren und hat so die Möglichkeit seine Lösungsidee öffentlich zur Diskussion freizugeben.

Um das Modell erstellen zu können, muss sich der Student vorerst bei der BPMN-Community registrieren (*sicher*). Er hat die Wahl sich mit einer E-Mail-Adresse und einem Passwort anzumelden oder eine OpenID zu verwenden. Er entscheidet sich dafür seine HPI⁸-OpenID zu benutzen und wählt anhand des HPI-Icons den bereits dafür vordefinierten Prefix aus, so dass er nur noch seinen HPI-Benutzernamen einzutragen braucht.

Um auch einen Namen in der BPMN-Community zu haben und sich beim Oryx-Editor authentifizieren zu können, muss er zusätzlich noch seinen Vor- und Nachnamen angeben. Seine E-Mail-Adresse wird hauptsächlich zu Benachrichtigungszwecken von ihm verlangt.

Nach der erfolgreichen Registrierung kann der Student die Lösung zu der Übungsaufgabe auf der Plattform modellieren. Da er noch etwas unsicher beim Modellieren ist und nicht möchte, dass jeder andere Nutzer ebenfalls seine Gedanken zu der Aufgabe sehen kann, erstellt er unter 'Meine Lösungen' seine Lösungsidee als 'Privaten Lösungsvorschlag'. Er gibt den Titel der Lösung an und es wird automatisch ein leeres Prozessmodell erzeugt (*schnell und unkompliziert*). Der Student wird auf die Prozessmodellseite weitergeleitet (*verlinkt*) und kann sofort mit Hilfe des Oryx-Editors mit dem Modellieren beginnen.

Nachdem er seine Lösung erstellt hat, sieht er sich den 'Öffentlichen Lösungsvorschlag' zu der Übungsaufgabe an, der beim Rating am besten abgeschnitten hat. Beim Vergleich von diesem mit seiner Idee, ist ihm jedoch noch nicht alles ganz klar. Um zu seinem Modell Fragen stellen zu können, veröffentlicht er es nun doch (*Offenheit*). Dazu genügt ein Klick auf den Button 'Lösung als öffentlichen Vorschlag einsenden' (*schnell und unkompliziert*) und schon erscheint es in der Liste der Lösungen, die jeder Nutzer einsehen kann.

Er schreibt in einem Kommentar zu seinem Modell seine Frage (*kommunikativ*) und hofft auf eine schnelle Antwort eines anderen BPMN-Community-Mitgliedes, das ihm weiterhelfen kann (*Freiwillige Zusammenarbeit*).

⁸HPI: Hasso-Plattner-Institut

4.1.2 BPMN Lehren

Ein BPMN-Trainer, der in einem Unternehmen arbeitet, das Seminare rund um das Thema Geschäftsprozessmodellierung anbietet, ist auf der Kollaborationsplattform aktiv, um sich dort profilieren zu können.

Auf seiner Startseite wird ihm angezeigt, dass ein 'Öffentlicher Lösungsvorschlag' zu einer Übungsaufgabe hinzugefügt worden ist. Er sieht sich das Prozessmodell an und liest die Frage, die als Kommentar zu dem Modell verfasst wurde (*Offenheit*).

Anhand der Frage und der Lösungsidee des Nutzers merkt er schnell, dass der Modellierer noch nicht alle Details zu dem in dem Tutorial erklärten BPMN-Konstrukt verstanden hat. Er erkennt jedoch auch, dass dieser Spezialfall nur unzureichend gut in dem Tutorial beschrieben wird. Nachdem der BPMN-Trainer dem Modellierungsfänger auf seine Frage geantwortet hat (*kommunikativ*), editiert er daher sofort den Tutorial-Text und fügt die fehlenden Erläuterungen hinzu (*offen*). Außerdem modelliert er zusätzlich noch einen Beispielprozess, der das erklärte Problem verdeutlichen soll. Für die Modellierung des Beispiels verwendet er den dafür im Rich Text Editor vorgesehenen Button zum Einfügen von neuen oder bereits existierenden Prozessmodellen (*schnell und unkompliziert*).

Da die Kunden des BPMN-Trainers nicht nur aus dem deutschsprachigen Raum kommen, sondern auch überwiegend aus Australien, wo es einen Zweitsitz seines Unternehmens gibt, fügt er dem bisher noch nicht in der englischen Sprache vorhandenen Tutorial-Text auch gleich noch eine Übersetzung hinzu (*Globales Handeln*).

4.2 Django App Tutorials

Der Django Code der Tutorial-Applikation befindet sich in dem Unterverzeichnis 'tutorials' des App-Ordners.⁹ Hier liegen die Python-Dateien `__init__.py`, `admin.py`, `forms.py`, `models.py`, `urls.py` und `views.py`. Die Templates zu der Applikation befinden sich in dem entsprechenden Template-Ordner.

Anhand der Template-Dateien und dem Python Code der `views.py` und `models.py` wird in den folgenden Abschnitten der Aufbau der Tutorial-Applikation beschrieben.

⁹Details zur Architektur eines Django-Projektes sind unter anderem in der Bachelorarbeit von Stefan Wehrmeyer [Wehrmeyer, 2009] zu finden oder in der Django-Dokumentation nachzulesen [Foundation, 2009].

4.2.1 Models

In der Datei `models.py` sind die Klassen `Tutorial`, `Exercise` und `Solution` implementiert.

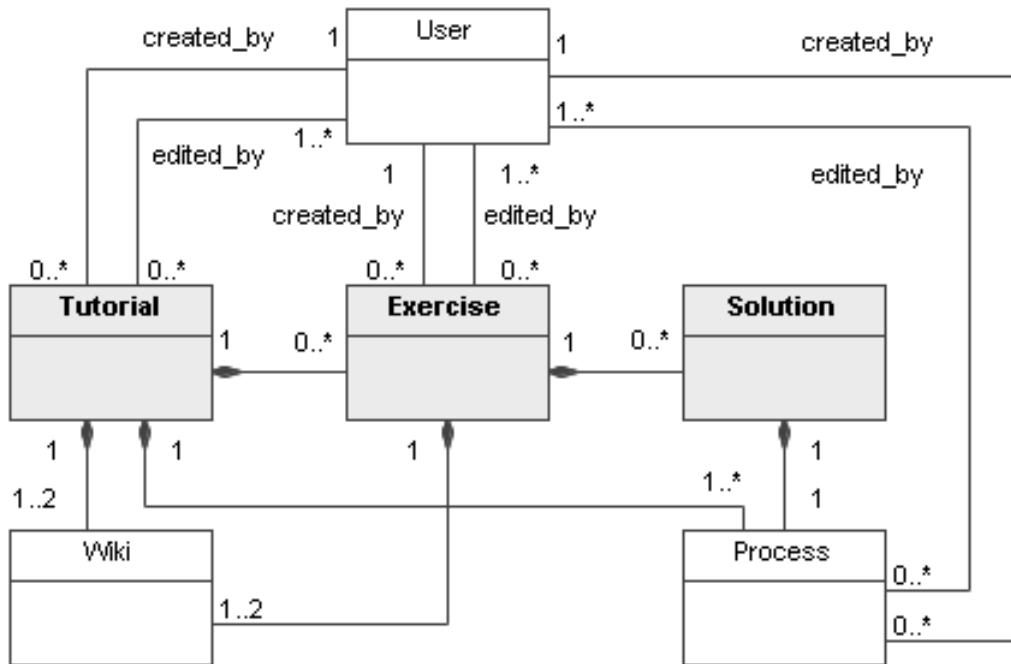


Abbildung 4: UML Klassendiagramm Tutorials

Tutorial Zu einem Tutorial-Objekt gehört immer neben der Angabe des Titels (`title`) mindestens ein Wiki (`_wiki`) und ein Prozessmodell (`_processes`), die Information, wer das Tutorial erstellt hat (`created_by`), wann er es erstellt hat (`created_on`), wer es bearbeitet hat (`edited_by`) und wann es bearbeitet worden ist (`edited_on`).

In dem Wiki steht der Text des Tutorials.

Das gesetzte Attribut `translated_wiki = True` gibt an, dass der Text des Tutorials übersetzbar ist. Übersetzt ein Nutzer den Text, wird für die Übersetzung ein weiteres Wiki-Objekt dem Tutorial hinzugefügt.

Das Prozessmodell, das ein Tutorial-Objekt standardmäßig besitzt, repräsentiert das Thumbnail¹⁰, das auf der Übersichtsseite der Tutorials angezeigt wird.

Die Thumbnails sollen dem Nutzer einen schnell zu erfassenden Überblick über den Inhalt des jeweiligen Tutorials verschaffen.

Aus Nutzersicht betrachtet, ist ein Thumbnail lediglich ein Anzeigebild, das mit dem Oryx-Editor modelliert wird. Hinter dem Bild verbirgt sich keine Prozessmodell-Seite. Klickt der Nutzer auf den Button 'Vorschaubild ändern', gelangt er sofort zu dem Oryx-Modell, das nach dem Anlegen eines Tutorials erst einmal leer ist.

Um das Thumbnail-Prozessmodell von anderen Prozessmodellen unterscheiden zu können, wird bei der Erstellung eines Tutorials das Feld `summary` des Prozess-Objektes mit dem String `'thumbnail'` belegt.

Die Methode `thumbprocess()` gibt das zu dem Tutorial gehörende Thumbnail-Prozessmodell zurück.

```
def thumbprocess(self):  
    return self._processes.get(summary='thumbnail')
```

Weitere Prozessmodelle werden dem Tutorial-Objekt angehängt, wenn ein Nutzer zur Veranschaulichung von Beispielen Modelle in den Wiki-Text des Tutorials einfügt. (*Siehe Abschnitt 5.2.3*)

Die Methode `process()` liefert die zu dem Tutorial gehörenden Prozessmodelle zurück, die kein Thumbnail repräsentieren.

```
def process(self):  
    return self._processes.filter(summary='').all()
```

Exercise Zu jedem Tutorial kann es eine, mehrere oder keine Übung geben.

Zu welchem Tutorial eine Übung gehört, gibt das Fremdschlüssel-Attribut `tutorial` an.

Zusätzlich referenziert das Übungs-Objekt auf die Nutzer, die die Übung erstellt und bearbeitet haben und auf ein bis zwei Wiki-Objekte (je nachdem, ob eine Übersetzung zu der Übung vorliegt oder nicht).

Prozessmodelle kann ein Übungs-Objekt nicht haben, da es bisher dafür keinen Anwendungsfall gegeben hat. Sollte jedoch einmal der Bedarf bestehen, dass einem Übungs-Objekt ebenfalls Prozessmodelle hinzugefügt werden können sollen, braucht

¹⁰Vorschaubild

die Klasse nur um das Attribut `_processes` und die Methode `process()` erweitert werden:

```
_processes = generic.GenericRelation(Process, related_name='
exercise-processes')

def process(self):
    return self._processes.all()
```

Solution Jede Übung kann eine, mehrere oder keine Lösung haben.

Eine Lösung zu einer Übung ist nichts anderes als ein vom Nutzer modelliertes Prozessmodell, das um die beiden Informationen erweitert worden ist, zu welcher Übung es gehört und ob es ein öffentlich sichtbares Prozessmodell ist oder nur von dem Nutzer, der es auch erstellt hat, angesehen und bearbeitet werden kann.

Ein Lösungs-Objekt besitzt die drei Attribute `exercise`, `_process` und `public`.

Der Boolesche Wert, mit dem das Attribut `public` belegt wird, gibt an, ob es sich bei der Lösung um einen öffentlichen oder einen privaten Lösungsvorschlag handelt. Ist `public==False`, liegt eine 'private Lösung' vor. Private Lösungen sollen nur dem Nutzer angezeigt werden, der diese auch erstellt hat.

Das Konzept der Unterscheidung zwischen öffentlichen und privaten Lösungsvorschlägen ist entwickelt worden, um den Nutzern die Möglichkeit zu geben, als BPMN-Lernende ihre Ideen zu den Übungsaufgaben modellieren können, ohne dass diese gleich für alle offen sichtbar sind.

BPMN-lehrende Mitglieder der Plattform können hingegen sofort eine öffentliche Lösung modellieren, wenn diese beispielsweise als Musterlösung für die Übungsaufgabe dienen soll.

Titel, Beschreibungstext und wer zu welchem Zeitpunkt etwas an der Lösung geändert hat, steht direkt in dem zu dem Lösungs-Objekt gehörendem Prozessmodell-Objekt, auf das über die Methode `process()` zugegriffen werden kann.

4.2.2 Views

In der Datei `views.py` sind die Funktionen zur Erstellung, Bearbeitung und dem Anzeigen von Tutorials, Übungen und Lösungen implementiert.

Tutorial Die Funktion `overview()` liefert das Template mit der Übersichtsseite über alle Tutorials zurück.

Klickt der Nutzer auf der Übersichtsseite auf den Button 'Neues Tutorial', muss er einen Titel angeben und den Erstellvorgang anschließend bestätigen. Hat er das getan, wird die Funktion `new_tutorial()` aufgerufen.

Die Funktion legt ein neues Tutorial-Objekt an, fügt diesem ein Wiki- und ein Prozessmodell-Objekt hinzu und ruft zum Schluss die Funktion `show_tutorials(t_id)` auf, die das Template für das im Parameter angegebene Tutorial rendert.

Klickt der Nutzer auf der Übersichtsseite auf ein bereits erstelltes Tutorial, wird die Funktion `show_tutorials(t_id)` ebenfalls ausgeführt.

Möchte der Nutzer den Titel eines Tutorials editieren, wird die Änderung durch den Funktionsaufruf von `edit(t_id)` gespeichert. Die Funktion zum Ändern des Wiki-Textes ist jedoch nicht in der Datei `views.py` der Tutorial-Applikation implementiert, sondern ist in der Datei `views.py` der Wiki-Applikation zu finden.

Exercise Analog zu den Tutorial-Funktionen gibt es für die Übungen die Funktionen `new_exercise(t_id)`, `show_exercise(e_id)` und `edit_exercise(e_id)` zum Anlegen, Ansehen und Bearbeiten.

Eine gesonderte Übersichtsseite über alle existierenden Übungen gibt es nicht. Eine Liste der zu einem Tutorial gehörenden Aufgaben wird dem Nutzer auf der Seite des jeweiligen Tutorials angezeigt. Dafür werden beim Ausführen der Funktion `show_tutorials(t_id)` dem entsprechenden Template die Übungsaufgaben zu dem Tutorial übergeben.

Solution Zum Erstellen einer öffentlichen Lösung wird die Funktion `new_default_solution(e_id)` aufgerufen, die ein Lösungs-Objekt mit dem Attribut `public=True` erzeugt. Eine private Lösung wird durch den Funktionsaufruf von `new_solution(e_id)` erstellt. Das Attribut `public` ist dabei auf `False` gesetzt. Da der Nutzer die Möglichkeit hat, eine private Lösung als öffentlichen Lösungsvorschlag einzusenden, gibt es noch die Funktion `send_solution(s_id)`, die den Wert `public` von `False` in `True` ändert.

Die Funktionen zum Ansehen und Ändern des Prozessmodells einer Lösung sind in der Datei `views.py` der Prozess-Applikation implementiert.

4.2.3 Templates

In der Datei `base.html` sind einige Eigenschaften beschrieben, die für alle Seiten der Tutorial-Applikation gelten. Jede Seite erbt die Eigenschaften, die dort in den Template-Tags stehen und nicht durch einen gleichnamigen Tag überschrieben werden.

Für jede zur Tutorial-Applikation gehörende Seite gilt beispielsweise, dass in der Navigationsleiste der Menüpunkt 'Tutorials' als ausgewählt markiert ist, wenn die Seite angezeigt wird, und die Farbe grün ist.

Tutorial Das Template der Tutorial-Übersichtsseite ist die Datei `list.html`.

In einer Liste von Boxen mit den jeweiligen Titeln und Thumbnails, die die Links zu den einzelnen Seiten repräsentieren, werden alle Tutorials alphabetisch aufgeführt.

Unter den bereits existierenden Tutorials kann man durch einen Klick auf den Button 'Neues Tutorial' weitere Tutorials hinzufügen. Um den Titel angeben zu können, wird die Datei `_tutorialnewform.html` gerendert.

Die Datei `show_tutorials.html` enthält den Template-Code für die Tutorial-Seiten. Hier wird jeweils der Titel und der Wiki-Text des Tutorials angezeigt, die durch das 'Inplace Editing' schnell und einfach editiert werden können.

Auf den Tutorial-Seiten werden außerdem die zu dem erklärten Konstrukt gehörenden Übungsaufgaben aufgelistet. Die Übungsaufgaben stehen nicht nur unter dem jeweiligen Tutorial-Text, sondern auch in der Seitenleiste, in der ebenfalls die Funktionalitäten zum Hinzufügen von Kommentaren und Tags zu finden sind.

Zum Anzeigen der Übungsliste in der Seitenleiste wird die Datei `_exercise_preview.html` in das Template geladen.

Die Übungsaufgaben werden dem Nutzer gleich oben in der Seitenleiste präsentiert, damit er sofort beim Öffnen des Tutorials sehen kann, ob zu dem erklärten Konstrukt Aufgaben vorliegen. Zusätzlich gibt es die Liste mit den Übungsaufgaben direkt unter dem Tutorial-Text, damit der Nutzer nicht wieder hochscrollen muss, nachdem er das Tutorial bis nach unten hin durchgearbeitet hat.

Der Button zum Hinzufügen von weiteren Aufgaben befindet sich entsprechend ebenfalls zweimal auf der Seite (jeweils unter der Auflistung der Übungen). Klickt der Nutzer auf 'Übung hinzufügen', kann er in ein Feld, das daraufhin erscheint, den Namen der neuen Übungsaufgabe angeben und mit einem Klick auf den Ok-Button

den Erstellvorgang bestätigen.

Der Template-Code für das jeweilige Eingabefeld und den Ok-Button steht in den Dateien `_exercisewform.html` und `_exercisewformh.html`.

Die Datei `_tutorialform.html` enthält das Titelfeld, das geladen wird, wenn der Nutzer den Titel des Tutorials ändern möchte.

Neben dem Titel befindet sich auf den Tutorial-Seiten der Button 'Versionen', mit dem die Version des Wiki-Textes zurück gesetzt werden kann.

Mit einem Klick auf den Button 'Vorschaubild ändern', gelangt man zum Oryx-Editor, um das Thumbnail editieren zu können.

Exercise Das Template `show_exercise.html` wird gerendert, wenn sich der Nutzer eine Übungsaufgabe ansieht. Analog zu den Tutorials wird der Titel und der Wiki-Text der Übung angezeigt. Der Titel kann durch Laden der Datei `_exerciseform.html` geändert werden, die den Code für das Titelfeld enthält.

Ebenfalls wie bei den Tutorial-Seiten, gibt es einen Button 'Versionen' zum Zurücksetzen von Wiki-Text-Änderungen.

Unter dem Aufgabentext können private Lösungsvorschläge hinzugefügt werden, die dort nach ihrem Erstellungsdatum absteigend sortiert aufgelistet werden, wenn der Nutzer angemeldet ist. Diese Übersicht wird durch die Datei `_show_mySolutions.html` in das Template geladen.

Unter den privaten Lösungen befindet sich eine Auflistung der öffentlichen Lösungsvorschläge, die zusätzlich auch in der Seitenleiste aufgeführt werden. Der Template-Code für diese Liste ist in der Datei `_publicSolution_preview.html` ausgelagert.

Solution Eine Lösung zu einer Übungsaufgabe wird als Prozessmodell dargestellt. Die Templates zu den Prozessmodell-Seiten liegen in dem Ordner 'process' und gehören zu der gleichnamigen Prozess-Applikation.

5 Das Wiki in der BPMN-Community

Texte, wie die Beschreibungen von Prozessmodellen oder die Inhalte der Tutorials, erstellt und bearbeitet der Nutzer auf der Plattform BPMN-Community mithilfe eines Rich Text Editors.

Diese Texte werden intern als Objekte der Klasse `Wiki` repräsentiert. Ein Wiki-Objekt kennt nicht nur den Textinhalt, sondern besitzt auch Informationen, wie zum Beispiel wer den Text editiert hat oder zu welchem Objekt-Typ (Tutorial, Best Practice, Gruppe, Forum, ...) der Text gehört.

Im Folgenden der Arbeit wird die technische Realisierung der Wiki-Texte beschrieben. Hierbei geht es vor allem um die Implementierung der Wiki-Prinzipien wie die Versionierung der Texte und die Möglichkeit des schnellen und einfachen Bearbeitens der Inhalte durch den Rich Text Editor.

5.1 Das Wiki-Objekt und seine Versionierung

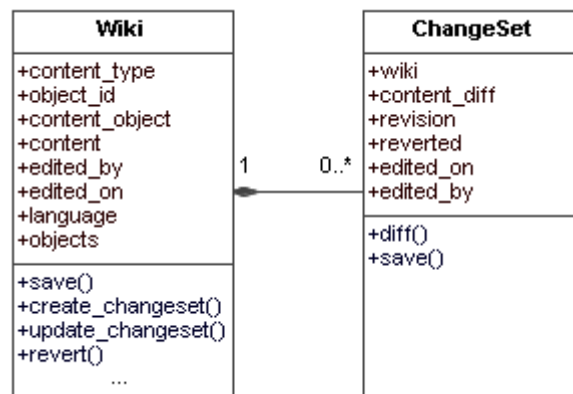


Abbildung 5: UML Klassendiagramm Wiki

Die Klasse `Wiki` ist in der Datei `models.py` der Wiki-Applikation implementiert.

Attribute des Wiki-Objekts Ein Wiki-Objekt gehört immer zu einem anderen Objekt, wie zum Beispiel zu einem Tutorial oder einem Prozessmodell.

Zu welchem Objekt das Wiki gehört, steht in seinem Attribut `content_object`, die

dazugehörige ID in `object_id` und der Typ in `content_type`.

Unter dem Attribut `content` ist der Wiki-Text zu finden. In `edited_by` steht, wer den Text bearbeitet hat und in `edited_on` zu welchem Zeitpunkt die Änderung erfolgte.

Das Attribut `language` gibt die Sprache an, in der der Wiki-Text gespeichert ist. Mithilfe des `WikiManager`, auf den über `objects` zugreiffen werden kann, lässt sich die Objekt-Methode `get_wiki_for_language(self, obj, language, user=None)` aufrufen, die das Wiki für das im Parameter übergebene Objekt und der angegebenen Sprache zurückliefert.

Die Methode `save()` Die Methode `save(self, force_insert=False, force_update=False, force_new_revision=True)` der Klasse `Wiki` überschreibt die Standard-Methode `save(self, force_insert=False, force_update=False)`.

Sie führt zuerst die Funktion `clean_office_xml(value)` auf dem Attribut `content` aus.

Der Befehl `self.content = clean_office_xml(self.content)` bereinigt den Wiki-Text von XML-Tags. Diese Tags stehen in dem Wiki-Text, wenn der Nutzer Inhalte aus einem Office-Dokument in den Rich Text Editor kopiert hat. Sie müssen herausgefiltert werden, damit der Text nach dem Speichern ordentlich dargestellt werden kann.

Nach der Bereinigung durch `clean_office_xml(value)` wird anschließend mithilfe der Abfrage `if self.id != None` überprüft, ob das Wiki-Objekt bereits in der Datenbank existiert oder ob es zum ersten Mal gespeichert werden soll.

Wird das Objekt zum ersten Mal angelegt, wird die Methode `super(Wiki, self).save(force_insert=force_insert, force_update=force_update)` aufgerufen und das neue Objekt in der Datenbank abgelegt.

Existiert das Wiki bereits und es soll nur eine Änderung gespeichert werden, wird zunächst das alte Wiki-Objekt der Variable `oldwiki` übergeben, um es zwischenspeichern. Anschließend wird durch die Abfrage `if force_new_revision` überprüft, ob die Änderungen als neue Revision gespeichert werden sollen oder ob sie in die alte Version zu integrieren sind.

Standardmäßig liegt der Fall vor, dass eine neue Revision angelegt wird. Die Möglichkeit keine neue Revision zu erstellen ist zwar implementiert, dem Nutzer jedoch derzeit nicht gegeben.

Besitzt die Variable `force_new_revision` also den Wert `True`, wird zuerst das geänderte Wiki-Objekt gespeichert. Anschließend wird durch den Funktionsaufruf von `self.create_changeset(oldwiki.edited_by, oldwiki.content)` mit Hilfe der Daten des in `oldwiki` stehendem Wikis ein Objekt der Klasse `ChangeSet` erstellt.

`ChangeSet`-Objekte dienen der Versionierung von Wiki-Texten. Je nachdem, wie oft ein Wiki-Objekt geändert worden ist, besitzt es ein, kein oder mehrere `ChangeSets`. Ein `ChangeSet`-Objekt enthält die Informationen zu welchem Wiki es gehört, welche Änderungen am Wiki-Text vorgenommen worden sind, welche Versionsnummer vorliegt, ob die Version schon einmal zurückgesetzt worden ist, von wem die Version angelegt wurde und wann sie erstellt worden ist. Außerdem besitzt es die Methoden `diff()`, die für das Anzeigen von Versionsunterschieden genutzt wird, und `save()`, die für das Speichern des `ChangeSet`-Objektes zuständig ist.

Die Funktion `create_changeset(self, old_user, old_content)` ruft die Hilfsmethode `diff(txt1, txt2)` mit `self.content` als ersten und `old_content` als zweiten Parameter auf. Diese erzeugt durch Verwendung der Funktionen der Python-Library `diff_match_patch` einen String, der die Unterschiede zwischen der alten und der neuen Version des Wiki-Textes enthält.

Der String, den die Hilfsmethode zurückliefert, wird der Variable `content_diff` übergeben. Die anschließende Zeile `cs = ChangeSet(wiki=self, edited_by=old_user, content_diff=content_diff).save()` kreiert dann daraus das `ChangeSet`-Objekt.

Besitzt die Variable `force_new_revision` nicht den Wert `True`, sondern ist auf `False` gesetzt, wird nicht die Funktion `create_changeset(self, old_user, old_content)` aufgerufen, sondern `update_changeset(self, oldwiki)`.

Die Funktion `update_changeset(self, oldwiki)` holt sich zunächst das letzte `ChangeSet`-Objekt, das noch nicht mithilfe der Funktion `revert(self, revision=None)` - *siehe unten* - zurückgesetzt worden ist. In dem Attribut `content_diff` dieses `ChangeSet`-Objektes steht, welche Änderungen vor den aktuellen Änderungen an dem Wiki-Text vorgenommen worden sind.

Zur Veranschaulichung: Wenn `oldwiki` das Wiki 'B' ist und das neue Wiki-Objekt mit den aktuellen Änderungen als 'C' bezeichnet wird, dann steht in `content_diff` des letzten `ChangeSet`-Objektes von 'B', welche Änderungen an dem Vorgänger-

Wiki 'A' vorgenommen wurden, um auf 'B' zu kommen. Zusammengefasst: In `content_diff` steht der Unterschied zwischen dem Wiki-Text von 'A' und 'B'.

Dieser Unterschied wird einer Variable `patch` zugewiesen. Durch den Vergleich von `patch` mit dem Wiki-Text von `oldwiki`, also 'B', lässt sich auf den Wiki-Text des Vorgänger-Wikis 'A' schließen. Dieser wird der Variable `reverted_content` zugewiesen.

Mithilfe von `reverted_content` kann nun der String erstellt werden, der den Unterschied zwischen dem aktuellen Text des Wikis 'C' und dem alten Text des Vorgänger-Wikis 'A' enthält. Dieser String wird dem Attribut `content_diff` des letzten `ChangeSet`-Objekts zugewiesen, das daraufhin nicht mehr den Unterschied zwischen 'A' und 'B', sondern zwischen 'A' und 'C' kennt.

Das aktuelle Wiki wird anschließend gespeichert und das alte Wiki 'B' somit überschrieben.

Die Methode `revert()` Der Nutzer kann Änderungen, die an einem Wiki-Text vorgenommen worden sind, zurücksetzen.

Klickt er beispielsweise auf den Button 'Versionen' eines Tutorials, werden ihm alle Revisionen aufgelistet, die es zu diesem Tutorial-Text gibt. Wählt er eine Revision aus, wird der Inhalt der von ihm gewählten Version angezeigt, wobei Änderungen zur Vorgängerversion farblich hervorgehoben sind.

Klickt der Nutzer anschließend auf den Button 'Zurücksetzen', wird die Funktion `revert(self, revision=None)` ausgeführt, die den aktuellen Tutorial-Text auf die vom Nutzer ausgewählte Revision zurücksetzt.

Die Funktion `revert(self, revision=None)` holt sich dafür zunächst alle `ChangeSet`-Objekte aus der Datenbank - von dem zuletzt erstellten `ChangeSet` bis zu der Revisionsnummer, die beim Aufruf als Parameter übergeben worden ist (`changesets = self.changesets.filter(reverted=False, revision__gte=revision).order_by('-revision')`).

Anschließend werden die `ChangeSets` der Reihe nach durchgegangen und für jedes einzelne Objekt wird der Wiki-Text mithilfe der Funktionen der Python-Bibliothek `diff_match_patch` um die in dem Attribut `content_diff` gespeicherten Änderungen zurückgesetzt. Außerdem wird zusätzlich das Attribut `reverted` des `ChangeSets` auf `True` gesetzt. Besitzt `reverted` den Wert `True`, bedeutet dies, dass die Version zurückgesetzt worden ist und vom Nutzer nicht noch einmal ausgewählt werden kann.

5.2 YUI Library: Der Rich Text Editor

Die YUI Library [Yahoo!, 2009] ist eine Sammlung von JavaScript-Bausteinen zur Entwicklung von interaktiven Web-Applikationen. Sie steht unter der BSD-Lizenz¹¹ und ist somit für alle Webentwickler frei verfügbar. Die YUI Library enthält unter anderem auch den Rich Text Editor, der auf der Plattform BPMN-Community zur schnellen und einfachen Bearbeitung von Texten verwendet wird.

Im Folgenden Teil der Arbeit wird beschrieben, wie der Rich Text Editor auf den Seiten der Plattform eingebunden wird und welche Anpassungen an dem Editor durchgeführt wurden.

5.2.1 Das Einbinden des Rich Text Editors

Die Integration des Rich Text Editors in die Plattform BPMN-Community erfolgt mit Hilfe der Komponente YUI Loader Utility¹².

Das Script, das zur Ausführung des YUI Loaders benötigt wird, wird durch die Zeile `<script type="text/javascript" src="http://yui.yahooapis.com/2.7.0/build/yuiloader/yuiloader.js"></script>` in den Block `{%block extra_js%}` der jeweiligen Template-Datei geladen, die den Rich Text Editor verwendet.

Zu diesem Script-Tag muss ebenfalls auch der JavaScript-Code der Datei `editorconfig.js` auf der Seite eingebunden werden. Dies geschieht durch die Verwendung des Template-Tags `{% compressed_js 'pcp_wiki' %}`.

In der Datei `editorconfig.js` wird das YUI Loader Objekt `loader` der Klasse `YAHOO.util.YUILoader` erzeugt, das für das Laden und die Initialisierung des Rich Text Editors beim Aufruf der Seite zuständig ist.

Bei der Initialisierung wird die Funktion `loadEditor(textareaId)` ausgeführt, die die Eigenschaften und das Verhalten des für die Plattform angepassten Rich Text Editors definiert.

Soll der erweiterte Rich Text Editor geladen werden, der die Möglichkeit zum Einfügen von Prozessmodellen bietet, wird zusätzlich zu `loadEditor(textareaId)` die Funktion `addProcessModelButton(myEditor, textareaId)` aufgerufen.

Beide Funktionen werden in den folgenden Abschnitten der Arbeit beschrieben.

¹¹<http://developer.yahoo.com/yui/license.html>

¹²<http://developer.yahoo.com/yui/yuiloader/>

5.2.2 Die Funktion loadEditor()

```
function loadEditor(textareaId) {
    var myConfig={
        ...
    }
    var myEditor=new YAHOO.widget.Editor(textareaId;myConfig);
    yuiImgUploader(myEditor ,textareaId , '/wiki/upload_img/' , '
        image ');
    addOkButtons(myEditor ,textareaId);
    myEditor.render();
    var parentform=new YAHOO.util.Element(YAHOO.util.Dom.get(
        textareaId).form);
    YAHOO.util.Event.on(parentform.getElementsByClassName('
        content_submit')[0] , 'click' ,function(){
        myEditor.saveHTML();
    });
    return myEditor;
}
```

Die Funktion `loadEditor(textareaId)` erstellt das Editor-Objekt. Sie erwartet als Parameter die ID der Textarea (`textareaId`), die durch den Rich Text Editor ersetzt werden soll, und gibt ein Objekt der Klasse `Yahoo.widget.Editor`¹³ zurück (`return myEditor;`).

Toolbar-Buttons In der Variable `myConfig` wird festgelegt, welche Buttons die Toolbar des Rich Text Editors nutzen soll, die bereits durch die YUI Library implementiert sind.

Die Toolbar des Editors der BPMN-Community ist recht einfach gehalten:

Der Button `heading` gibt dem Nutzer die Wahlmöglichkeit, einen markierten Textabschnitt in eine Überschrift des Typs `h3` zu transformieren oder diesen in das Standard-Textformat zu bringen. Mit den Buttons `bold` und `italic` kann der Nutzer Worte fett oder kursiv schreiben. Der Button `removeformat` macht diese Style-Änderungen wieder rückgängig. Mit Hilfe der Buttons `insertunorderedlist` und `insertorderedlist` lassen sich Listen mit Aufzählungszeichen oder Nummerierung

¹³<http://developer.yahoo.com/yui/docs/YAHOO.widget.Editor.html>

gen einfügen. Möchte der Nutzer Links setzen, kann er dies durch die Funktion des Buttons `createlink` tun. Der Button `insertimage` bietet die Möglichkeit Bilder in den Text zu integrieren.

Styles In der Variable `myConfig` sind ebenfalls die Cascading Style Sheets für den Rich Text Editor definiert.

Da der Nutzer die Möglichkeit hat, formatierten Text aus anderen Quellen in den Editor einzufügen, muss sichergestellt werden, dass die Textformatierungen den Style-Definitionen der BPMN-Community angepasst werden.

Um die Übersichtlichkeit der Seite zu bewahren wird beispielsweise nur ein Überschriftenformat erlaubt. Jede Überschrift (von `h1` bis `h6`), die der Nutzer einfügt, wird mit den gleichen Attributen belegt, die in den Cascading Style Sheets der BPMN-Community unter dem Überschriften-Typ `h3` definiert sind.

Außerhalb des Rich Text Editors (das heißt, der Editor ist geschlossen und der Wiki-Text wird statisch angezeigt) wird die Umwandlung des Überschriftenformates im Wiki-Text durch die Funktion `headline_filter(value)` des Wikifilters realisiert. (Siehe Abschnitt 5.3)

Das Editor-Objekt Die Zeile `'var myEditor = new YAHOO.widget.Editor(textareaId;myConfig);'` erstellt das Editor-Objekt `myEditor` mit den unter `myConfig` definierten Eigenschaften. Der Parameter `textareaId` gibt die ID der Textarea an, die durch das Objekt ersetzt werden soll. In der Textarea steht der Wiki-Text, der unter dem Attribut `content` des Wiki-Objektes gespeichert ist. Dieser wird beim Ersetzen automatisch in den Rich Text Editor geladen.

Hochladen von Bildern Da der Toolbar-Button `insertimage` standardmäßig nur das Einfügen von Bildern durch Angabe eines Links erlaubt, wird der Editor um die Funktion Bilder auch von der Festplatte hochladen zu können erweitert. Dazu wird die Funktion `yuiImgUploader(myEditor,textareaId,'/wiki/upload_img/','image');` aufgerufen, die ebenfalls in der JavaScript-Datei `editorconfig.js` definiert ist. Die Funktion ist von dem Betreiber der Seite AllMyBrain.com implementiert worden, der den Quellcode zu der Funktion der Öffentlichkeit zur Verfügung gestellt hat. [Dennis, 2009]

Ergänzung von Ok-Buttons Die Menüs der Standard-Toolbar-Buttons `createlink` und `insertimage` besitzen keine Ok-Buttons. Der Nutzer kann das Einfügen eines Links oder eines Bildes dementsprechend nicht explizit bestätigen. Nutzertests, die auf der Plattform BPMN-Community durchgeführt worden sind, ergaben jedoch, dass die Anwender nicht wissen, dass der Link (oder das Bild), den sie dem Text hinzufügen wollen, eingefügt wird, wenn sie auf den Button zum Schließen des Menüs klicken.

Um die Bedienungsfreundlichkeit zu verbessern, ist daher eine Funktion implementiert worden, die vor dem Rendern des Rich Text Editors den beiden Standard-Menüs jeweils ein Ok-Button hinzugefügt. Die Funktion wird durch den Befehl `addOkButtons(myEditor, textareaId);` ausgeführt.

Rendern des Editors Der Rich Text Editor wird durch die Ausführung der YUI-Library-Funktion `render()` auf dem Objekt `myEditor` gerendert (`myEditor.render()`).

Inhalt speichern Zu der Textarea, die durch den Editor ersetzt wird, existiert in der jeweiligen Template-Datei ein Submit-Button mit dem gesetzten Attribut `class="content_submit"`, der den vom Nutzer geschriebenen Text in der Datenbank speichert.

Um diesen Button mit dem Rich Text Editor zu verknüpfen, wird ein Event-Listener erstellt, der den Submit-Button überwacht.

Anhand der Textarea-ID kann in der JavaScript-Datei auf das zu dem Button gehörende Formular in der Template-Datei zugegriffen werden. Das Formular-Element wird der Variable `parentform` zugewiesen, so dass darüber der Submit-Button der Klasse `content_submit` identifiziert werden kann. Das Hinzufügen des Event-Listener erfolgt anschließend durch den Methodenaufruf `YAHOO.util.Event.on(parentform.getElementsByClassName('content_submit')[0], 'click', function() {myEditor.saveHTML();});`. Klickt der Nutzer auf den Button, wird die Funktion `saveHTML()` auf dem Objekt `myEditor` ausgeführt, die den im Editor stehenden Text zurück in die Textarea schreibt, so dass dieser in der Datenbank gespeichert werden kann.

5.2.3 Die Funktion addProcessModelButton()

Die Funktion `addProcessModelButton(myEditor, textareaId)` erweitert die Rich-Text-Editor-Toolbar um den Button `processmodel`. Der Button `processmodel` bietet dem Nutzer die Möglichkeit Prozessmodelle in den Wiki-Text einzufügen. Dies ist vor allem im Bereich der Tutorials wichtig, um Beispielmuster in den Tutorial-Text integrieren zu können. Aber auch im Rich Text Editor des Forums gibt es den zusätzlichen Button, damit der Nutzer, der eine allgemeine Frage zu einem Prozessmodell hat, diese auch an einem konkreten Modell erläutern kann.

Beim Einfügen eines Prozessmodells hat der Nutzer die Wahl ein neues Modell zu erstellen oder ein bereits auf der Plattform existierendes Modell zu verwenden. Durch die Möglichkeit ein Prozessmodell wiederverwenden zu können, spart sich der Nutzer die Arbeit und die Zeit, ein Modell, das es in der BPMN-Community schon gibt, nachmodellieren zu müssen, wenn er sich darauf beziehen möchte.

Durch den Methodenaufruf `addProcessModelButton(myEditor, textareaId)` wird der Button `processmodel` und das dazugehörige Menü erstellt und gerendert.

Das Menü unterteilt sich in die zwei Bereiche 'Insert a New Process Model' (Ein neues Prozessmodell einfügen) und 'Insert a Copy of an Existing Process Model' (Eine Kopie eines existierenden Prozessmodells einfügen).

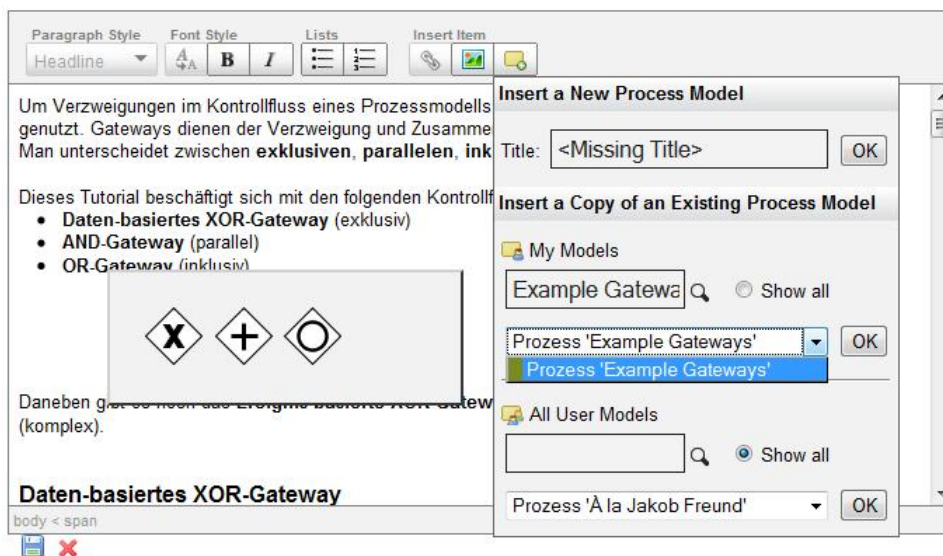


Abbildung 6: Screenshot des erweiterten Rich Text Editors

Insert a New Process Model Um ein neues Prozessmodell einfügen zu können, muss der Nutzer einen Titel angeben und anschließend auf den Ok-Button klicken. Der Event-Listener für den Ok-Button führt dann die Funktion `insertNewProcessModel()` aus.

Die Funktion `insertNewProcessModel()` fragt den eingegebenen Titel ab und leitet diesen per Post-Request an die View-Funktion `insert_pm()` der Wiki-Applikation weiter. Die Funktion `insert_pm()` fügt daraufhin dem jeweiligen Objekt, zu dem das Wiki gehört, ein leeres Prozessmodell hinzu.¹⁴ Anschließend liefert sie ein Dictionary mit der ID und der URL des neu erstellten Prozessmodells zurück.

Mithilfe der Daten aus dem Dictionary erstellt die Funktion `insertNewProcessModel()` den HTML-Code für die Darstellung des Prozessmodells, der in den Text integriert wird.

Das Modell wird in dem Wiki-Text als ein Bild mit einer Verlinkung auf die Prozessmodellseite gespeichert.

Da die Bild-URL abhängig von der jeweiligen Revision des Modells ist, muss diese entsprechend angepasst werden, wenn das Prozessmodell geändert wird. (*Siehe Abschnitt 5.3*)

Das neue Prozessmodell wird außerhalb des Rich Text Editors im Oryx-Editor modelliert, der sich in einem neuen Tab oder einer anderen Seite des Browsers öffnet.

Insert a Copy of an Existing Process Model Möchte der Nutzer ein bereits auf der Plattform existierendes Prozessmodell in den Wiki-Text einfügen, hat er die Wahl dieses aus seinen eigenen Modellen auszuwählen ('My Models') oder aus einer Liste der Modelle aller Nutzer ('All User Models').

Der Ablauf des Einfügens eines bestehenden Prozessmodells wird im Folgenden für den Fall beschrieben, dass der Nutzer ein Modell eines anderen Community-Mitgliedes seinem Text hinzufügen möchte. Die Funktionen für den Fall, dass er ein Modell aus denen auswählt, die er selbst erstellt hat, sind analog dazu implementiert.

Der Nutzer hat die Möglichkeit sich alle Prozessmodelle anzeigen zu lassen, die es auf der Plattform gibt, oder ein bestimmtes Modell anhand seines Titels zu suchen.

¹⁴Bearbeitet der Nutzer beispielsweise gerade den Text eines Tutorials und möchte dort ein neues Modell einfügen, wird dem Tutorial-Objekt das Prozessmodell hinzugefügt.

Möchte sich der Nutzer alle Modelle geben lassen, klickt er dazu den Button 'Show all' an. Der Event-Listener für diesen Button führt daraufhin die Funktion `loadAllAllModels()` aus.

Die Funktion `loadAllAllModels()` sendet einen Get-Request an die View-Funktion `get_allPMs()` der Wiki-Applikation, die sich alle Prozessmodell-Objekte aus der Datenbank holt. Bevor die Daten zu den Prozessmodell-Objekten an die Funktion `loadAllAllModels()` zurück gegeben werden, werden sie alphabetisch nach dem Titel sortiert und die Tutorial-Thumbnail, die keine Repräsentation als Prozessmodell in der BPMN-Community haben (*siehe Abschnitt 4.2.1*), sowie die Lösungen von Tutorial-Übungen, die als 'privat' gekennzeichnet sind (*siehe ebenfalls Abschnitt 4.2.1*), werden vorher herausgefiltert. Anschließend liefert die Funktion `get_allPMs()` ein Dictionary zurück, das für jedes Prozessmodell-Objekt folgende Informationen enthält: Titel, ID, Style-Klasse und URL. Die Style-Klasse gibt an, ob das Prozessmodell zu einem Tutorial, zu einem Best Practice, zu einem Gruppen- oder einem Forenbeitrag gehört.

Anhand der Daten aus dem Dictionary füllt die Funktion `createDropDownBox(diction, boxNumber)`, die innerhalb der Funktion `loadAllAllModels()` aufgerufen wird, eine Auswahl-Box mit den Daten der Prozessmodelle. Klickt der Nutzer auf die Auswahl-Box wird ihm eine Liste mit allen Modell-Titeln angezeigt. Jedes Listenelement ist mit einer der vier BPMN-Community-Farben gekennzeichnet; je nachdem welche Style-Klasse dem Element zugeordnet worden ist (*siehe oben*), ist es mit der Farbe Grün, Blau, Orange oder Pink markiert.

Führt der Nutzer mit der Maus über die Prozessmodell-Titel oder geht die Liste der Modelle mit den Pfeiltasten seiner Tastatur durch, wird durch den Aufruf der Funktion `showImageOverlay(boxnumber)` über Event-Listener ein Vorschaubild zu dem jeweiligen Prozessmodell angezeigt.

Möchte der Nutzer ein bestimmtes Prozessmodell in den Text einfügen, von dem er den Titel beziehungsweise einen Teil des Titels kennt, braucht er sich nicht über 'Show all' alle Modelle geben lassen, sondern kann das gewünschte Modell über die Angabe des Titels suchen.

Hat der Nutzer den Titel in das dafür vorgesehene Feld eingegeben und auf den Such-Button geklickt oder den Beginn des Suchvorgangs mit dem Betätigen der Enter-Taste seiner Tastatur bestätigt, wird über den jeweiligen Event-Listener die Funktion `searchInAllModels()` aufgerufen. Diese schickt einen Post-Request mit dem vom

Nutzer angegebenen Titel als Parameter an die View-Funktion `get_allWishPM()` der Wiki-Applikation, die analog zu der Funktion `get_allPMs()` funktioniert, nur dass sie nicht alle Prozessmodelle aus der Datenbank holt, sondern lediglich die Modelle zurückgibt, die der Titel-Angabe entsprechen.

Anschließend führt die Funktion `searchInAllModels()` ebenfalls die Funktion `createDropDownBox(diction, boxNumber)` aus (*siehe oben*) und die Suchergebnisse werden dem Nutzer in der Auswahl-Box angezeigt.

Wählt der Nutzer ein Prozessmodell aus der Auswahl-Box aus und klickt auf den Ok-Button, wird von dem Event-Listener, der diesen Button überwacht, die Funktion `onClick_insertExPM(myEditor, boxNumber)` ausgeführt.

Die Funktion identifiziert das ausgewählte Prozessmodell und schickt über die Hilfsfunktion `insert_exPM(myEditor, selected_value)` einen Post-Request mit der ID des gewählten Modells an die View-Funktion `insert_exPM()` der Wiki-Applikation. Anhand der übergebenen ID holt sich diese Funktion das dazugehörige Prozessmodell aus der Datenbank.

Das gewünschte Prozessmodell-Objekt wird jedoch nicht direkt an die Funktion `insert_exPM(myEditor, selected_value)` zurückgegeben, sondern es wird eine Kopie des Oryx-Modells erzeugt und mit dieser ein neues Prozessmodell-Objekt erstellt, das dem jeweiligen Objekt, zu dem der Wiki-Text gehört, hinzugefügt wird. Durch das Erstellen einer Kopie des Prozessmodells werden Konflikte vermieden, die auftreten können, wenn das Modell bearbeitet wird. Beispiel: Ist das Prozessmodell ursprünglich als Best Practice modelliert worden und wird anschließend in einem Tutorial-Text weiterverwendet, kann es vorkommen, dass für das Modell im Tutorial Anpassungen vorgenommen werden, die jedoch nicht für das Best-Practice-Modell gelten sollen.

Ist der Kopiervorgang des Prozessmodells erfolgreich verlaufen, werden die Informationen ID und URL zu dem neuen Modell in einem Dictionary an die Funktion `insert_exPM(myEditor, selected_value)` zurückgeliefert. Diese erstellt dann den HTML-Code für die Darstellung des Prozessmodells, der in den Text eingefügt wird.

5.3 Der Wikifilter

Die Python-Datei `wiki_filter.py` liegt in dem Verzeichnis `'templatetags'` des Wiki-App-Ordners. In dem 'Wikifilter' sind die Funktionen `wikiescape(value)` und `wikiformescape(value)` implementiert, deren Hauptaufgabe es ist, den Wiki-Text von HTML-Code zu bereinigen bevor er dem Nutzer angezeigt wird. Mithilfe Regulärer Ausdrücke werden Tags wie `<div>` oder `` gefiltert und durch Escape-Sequenzen ersetzt.

Die Funktion `wikiescape(value)` wird immer dann ausgeführt, wenn der Inhalt des Wiki-Objektes als statischer Text dargestellt wird.¹⁵ Die Funktion `wikiformescape(value)` hingegen wird verwendet, wenn der Wiki-Text zum Bearbeiten in den Rich Text Editor geladen werden soll.¹⁶ Der Unterschied zwischen den beiden Funktionen besteht darin, dass `wikiformescape(value)` eine doppelte Ersetzung durchführen muss, damit der Text im Rich Text Editor ordentlich angezeigt werden kann. Das Zeichen `'>'` wird beispielsweise durch `'>'` substituiert und nicht wie in der Funktion `wikiescape(value)` nur durch `'>'`. Würde nämlich nur eine einfache Ersetzung stattfinden, würde die Textarea, in die der Wiki-Text zuerst geladen wird bevor es zum Aufruf des Rich Text Editors kommt, die Zeichenfolge `'>'` bereits als `'>'` interpretieren und der Rich Text Editor das Zeichen `'>'` anschließend als Teil eines HTML-Tags übersetzen. Durch die doppelte Substitution wird dies jedoch vermieden. Die Textarea übersetzt die Zeichenfolge `'>'` in `'>'` und der Rich Text Editor macht aus `'>'` das Zeichen `'>'`.

Die Funktionen `wikiescape(value)` und `wikiformescape(value)` sind jedoch nicht nur für das Einfügen von Escape-Sequenzen zuständig, sondern auch für das Aktualisieren von im Wiki-Text eingefügten Modellen und das Korrigieren des Überschriftenformates. Die Hilfsfunktion `processmodel_filter(value)` durchsucht den Text nach Prozessmodell-Bildern, holt sich anhand der ID die aktuellste Revision des Modells und ersetzt die alte Modell-URL mit der URL der neusten Revision. Die Hilfsfunktion `headline_filter(value)` ersetzt alle Überschrift-Typen (`h1`, `h2`, `h4`, `h5` und `h6`), die in einem Wiki-Text vorkommen können, durch den `h3`-Typ. Fügt der Nutzer nämlich Inhalte einer anderen Quelle in den Rich Text Editor ein, kann es passieren, dass dort Überschriften definiert sind, die der Nutzer jedoch nicht verwenden soll, um ein einheitliches Layout auf der Plattform zu bewahren.

¹⁵Beispiel: Die Template-Datei `show_wiki.html` nutzt den Filter `wikiescape`.

¹⁶Beispiel: Die Template-Datei `_wikiformedit.html` nutzt den Filter `wikiformescape`.

6 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurde dargestellt, wie die Wiki- und Wikinomics-Prinzipien auf die Kollaborationsplattform BPMN-Community angewendet worden sind: Es wurde erläutert, welche bekannten Wiki-Prinzipien existieren und was hinter den Wikinomics-Konzepten steckt. Im Anschließendenden wurde betrachtet, wie diese Prinzipien in der BPMN-Community umgesetzt worden sind und in welchen Bereichen der Plattform sie überall Anwendung finden. Abschließend folgte eine technische Dokumentation der Wiki-Texte, die Bestandteil der Plattform sind und sich ebenfalls an den Wiki- und Wikinomics-Prinzipien orientieren.

In diesem Kapitel soll ein Ausblick auf weitere Umsetzungsmöglichkeiten der Wiki-Konzepte in der BPMN-Community gegeben werden.

Editieren ohne Registrierung Das oberste Wiki-Gebot lautet: 'Jeder kann alles bearbeiten'. Dies gilt in der BPMN-Community jedoch nur für angemeldete Nutzer. Korrekt hieße das Offenheitsprinzip auf die Kollaborationsplattform angewendet: 'Jedes kann alles bearbeiten, wenn er als BPMN-Community-Mitglied angemeldet ist.' Durch die Tatsache, dass sich ein Nutzer erst auf der Plattform registrieren muss, um Inhalte editieren zu können, entsteht eine Hürde, die ihn davon abhalten könnte, sein Wissen der BPMN-Community beizutragen.

Kleinere Fehlerkorrekturen finden zum Teil nicht statt, weil Nutzer sich nicht die Mühe machen wollen, sich wegen Kleinigkeiten auf der Plattform anzumelden.

Diesen Nutzern käme man entgegen, wenn die Registrierung optional zu der Alternative der IP-Adressen-Speicherung angeboten werden würde.

Die Zuordnung zwischen nicht angemeldeten Nutzern und den Aktivitäten auf der Plattform würde durch das Speichern der IP-Adressen ebenfalls gegeben sein und das Wiki-Prinzip der Sicherheit wäre nicht verletzt.

Das Angebot sich dennoch bei der BPMN-Community mit seinem Namen registrieren zu können, bietet den Nutzern die Möglichkeit, sich über die Plattform im Web zu profilieren.

Vereinfachung der Verlinkung Das Wiki-Prinzip der Verlinkung ist auf der Kollaborationsplattform BPMN-Community nur unzureichend umgesetzt worden. An dieser Stelle besteht ebenso Erweiterungsbedarf.

Im Rich Text Editor können Links nur über den dafür vorgesehenen Link-Einfüge-Button in den Text integriert werden. Die Möglichkeit der automatischen Link-Erkennung besteht noch nicht.

Möchte der Nutzer beispielsweise in seinem Tutorial-Text auf ein anderes Tutorial verweisen, muss er dafür derzeit zuerst die URL des anderen Tutorials herausfinden und kann dann manuell den Link mit Hilfe des Link-Einfüge-Buttons setzen.

Da dies jedoch sehr umständlich ist, sollte in Zukunft eine nutzerfreundlichere Lösung umgesetzt werden.

Es könnte zum Beispiel das Menü des Link-Einfüge-Buttons so angepasst werden, dass dem Nutzer eine Liste von bereits existierenden Seiten auf der Plattform angeboten wird - ähnlich der Liste von bereits existierenden Prozessmodellen in der BPMN-Community im Menü des Prozessmodell-Einfüge-Buttons (*Siehe Abschnitt 5.2.3*). Wählt der Nutzer den Titel einer Seite aus der Liste aus, wird der dazugehörige Link in den Text eingefügt.

Eine andere Möglichkeit wäre, dass der Nutzer Verlinkungen setzen kann, indem er den Titel der zu verlinkenden Seite nach Wiki-Art in CamelCase in seinen Text schreibt und der Link daraufhin automatisch erzeugt wird. Zusätzlich zu dem Titel sollte wahrscheinlich auch der Typ, wie 'Tutorial', 'Best Practice', 'Gruppe' oder 'Forum', angegeben werden, um die Sicherstellung der eindeutigen Titelvergabe zu vereinfachen. - Beispiel für das Tutorial 'Kontrollflussverzweigungen/ Gateways':

„Man unterscheidet zwischen exklusiven, parallelen, inklusiven und komplexen Gateways [Tutorial:KomplexesGateway].“

Quellenverzeichnis

- [Anonymus, 2009] Anonymus (2009). Wiki Principles. <http://c2.com/cgi/wiki?WikiPrinciples>. 6. Juni 2009.
- [Dennis, 2009] Dennis (2009). An Image Upload Extension for YUI Rich Text Editor. <http://allmybrain.com/2007/10/16/an-image-upload-extension-for-yui-rich-text-editor/>. 24. Juni 2009.
- [Foundation, 2009] Foundation, D. S. (2009). Django documentation. <http://docs.djangoproject.com/en/dev/>. 15. Juni 2009.
- [Güntert, 2009] Güntert, M. (2009). BPMN-Community: Konzeptionelle und technische Realisierung von Anforderungen. Technical report, Hasso-Plattner-Institut Potsdam.
- [Klötzke et al., 2009] Klötzke, R., Kirst, K., and Uptojoe (2009). Das Wiki-Prinzip. <http://wiki.zum.de/Wiki-Prinzip>. 6. Juni 2009.
- [Rawald, 2009] Rawald, T. (2009). BPMN-Community: Benutzerverwaltung und Identitätsmanagement. Technical report, Hasso-Plattner-Institut Potsdam.
- [Schwarz, 2009] Schwarz, J.-F. (2009). BPMN-Community: Frontend-Technologien und die Model Viewer API. Technical report, Hasso-Plattner-Institut Potsdam.
- [Tapscott and Williams, 2006] Tapscott, D. and Williams, A. D. (2006). *Wikinomics: How Mass Collaboration Changes Everything*. Portfolio.
- [TWikiGuest, 2002] TWikiGuest (2002). Wiki universal principles. <http://wikix.ilog.fr/wiki/bin/view/Slides/SlideWikiPrinciples1>. 6. Juni 2009.
- [Wehrmeyer, 2009] Wehrmeyer, S. (2009). BPMN-Community: Architektur und Erweiterungsmöglichkeiten. Technical report, Hasso-Plattner-Institut Potsdam.
- [Wiggert, 2009] Wiggert, C. (2009). BPMN-Community: Konzept und Implementierung der Kommunikation. Technical report, Hasso-Plattner-Institut Potsdam.
- [Yahoo!, 2009] Yahoo! (2009). The Yahoo! User Interface Library. <http://developer.yahoo.com/yui/>. 21. Juni 2009.