Ontologien als ein Mittel zur Wiederverwendung von Domänen-spezifischem Wissen in der Software-Entwicklung

Benjamin Horst, Andrej Bachmann, Wolfgang Hesse

Philipps-Universität Marburg,
Hans-Meerwein-Straße
35032 Marburg
{horst, rodionov, hesse}@mathematik.uni-marburg.de

Zusammenfassung: Wiederverwendung hat in der Software-Entwicklung gute Tradition. Sie wird bislang vorwiegend in den späten Phasen erfolgreich eingesetzt, um Software-Produkte effizienter und mit höherer Qualität zu entwickeln. Während der Anforderungsanalyse und der konzeptuellen Modellierung spielt die Wiederverwendung bisher kaum eine Rolle. Mit Hilfe von Ontologien lässt sich jedoch Domänenwissen aus konzeptuellen Modellen einschlägiger Projekte schrittweise aufbauen und kann in späteren Projekten nutzbringend eingesetzt werden. Dies ist der Ansatz der Ontologie-basierten Software-Entwicklung.

In diesem Artikel wird eine Infrastruktur zur Wiederverwendung von konzeptuellen Modellen vorgestellt, welche einen Wissensaustausch zwischen Software-Projekten unter Verwendung von Domänen-Ontologien ermöglicht. Ergänzend wird der zugehörige Prozess dieses Wissensaustauschs skizziert und abschließend der praktische Nutzen des Verfahrens anhand von Ergebnissen einer Fallstudie diskutiert.

Schlüsselworte: Ontologie-basierte Software-Entwicklung, Wissenstransfer, Anforderungsanalyse, Domänen-Ontologie, Konzeptuelle Modellierung

1 Einleitung

Die Wiederverwendung auf der Implementierung- und Entwurf-Ebene wird in der Softwaretechnik seit jeher gefordert und spätestens seit der Einführung der Objekt-orientierung und Komponenten-basierter Software-Entwicklung erfolgreich praktiziert. In den letzten Jahren hat sich der Schwerpunkt der methodischen Unterstützung auf die sogenannten frühen Phasen der Software-Entwicklung – die Anforderungsanalyse und konzeptuelle Modellierung – verlagert. Wiederverwendung spielt jedoch in den frühen Phasen der Software-Entwicklung – der Anforderungsanalyse und konzeptuellen Modellierung – nach wie vor eine eher untergeordnete Rolle.

So werden heute nach wie vor bei den meisten Software-Projekten in den frühen Phasen umfangreiche Studien und Analysen betrieben und viele neue Modelle entwickelt, selbst wenn es im gleichen Anwendungsbereich bereits einschlägige Vor- und Parallel-Projekte gibt. Fertige Modelle und Konzepte wiederzuverwenden, die das Domänenwissen aus solchen Projekten repräsentieren, könnte dabei helfen, wertvolle Entwicklungszeit einzusparen und die Qualität der entstehenden Software zu verbessern.

Für die standardisierte Beschreibung von Wissen aus einem gegebenen Anwendungsbereich ist das Konzept der *Ontologien* entwickelt worden [UG96][He02]. Klassische Methoden zur Verwendung von Ontologien in der Software-Entwicklung umfassen einen Import aus globalen Ontologien oder das Erstellen von Projekt-spezifischen Ontologien. Beide Szenarien bieten jedoch nur sehr eingeschränkte Möglichkeiten zur Wiederverwendung von Wissen, welches innerhalb eines Projekts gewonnen wird.

Die Grundidee des *Ontologie-basierten Software-Entwicklungs (OBSE)*-Projekts besteht darin, Fachwissen aus bereits umgesetzten Projekten in neuen Projekten innerhalb der gleichen Domäne wiederzuverwenden [He05]. Dieser Wissens-Transfer wird auf der konzeptuellen Ebene realisiert.

Für dieses Projekt wurde der *OBSE-Prozess* der kombinierten Projekt- und Ontologie-Entwicklung vorgestellt, der bereits bestehende Vorgehensmodelle ergänzt [Ba10]. Der Prozess dient der Formalisierung von Fachwissen bereits beendeter Projekte, welches mittels Domänen-Ontologien weiteren Projekten zur Verfügung gestellt werden kann. Diese Domänen-Ontologien bilden nicht die gesamte Welt ab, sondern konzentrieren sich auf einen bestimmten Untersuchungsbereich.

Der OBSE-Prozess richtet sich somit in erster Linie an Software-Entwickler und -Häuser, die mehrere Projekte innerhalb einer Domäne realisieren. Den Software-Entwicklern soll es ermöglicht werden, bereits gewonnenes Fachwissen zwischen den häufig voneinander unabhängigen Projekten innerhalb derselben Domäne mit zum Teil überlappenden Untersuchungsbereichen auszutauschen. Durch einen Wissens-Transfer aus bereits realisierten Projekten, sollen Domänen-Ontologien aus dem Wissen entstehen, welches bereits eingesetzt und in der Regel von einem Fachexperten abgenommen wurde.

Für die Auswahl, den Transport und die Eingliederung von Modellen und Fragmenten, die von der Domänen-Ontologie in die laufenden Projekte und (in deren Endphase) zurück zur Domänen-Ontologie zu transportieren sind, wurden *Import- bzw. Export-Brücken* definiert. Diese bilden zusammen mit Werkzeugen zur Transformation von konzeptuellen Modellen in UML und in umgekehrter Richtung den Kern der OBSE-Werkzeugumgebung.

Um die Praktikabilität und die Tragfähigkeit des neuen Verfahrens zu untersuchen, wurde an der Philipps-Universität Marburg ein Evaluierungsprojekt durchgeführt. Dabei wurde aus einem abgeschlossenen Projekt eine Domänen-Ontologie aufgebaut und der OBSE-Ablauf anhand von zwei weiteren, ebenfalls abgeschlossenen Projekten simuliert. Anschließend wurden die Ergebnisse der Simulation mit den tatsächlichen, in den Projekten erzielten Ergebnissen verglichen. Dabei konnte beobachtet werden, dass der

OBSE-Prozess quantitative und qualitative Auswirkungen auf die Modelle eines Projekts hat [Ho10].

In den folgenden Abschnitten gehen wir zunächst auf alternative Ansätze und Arbeiten ein (Abschnitt 2), und geben dann einen Überblick über den OBSE-Prozess und dessen Infrastruktur (Abschnitt 3). Die Evaluierung ist Gegenstand von Abschnitt 4, an den sich eine Zusammenfassung mit Ausblick anschließt.

2 Verwandte Ansätze und Arbeiten

In den aktuellen Publikationen lässt sich ein breites Spektrum an Szenarien für einen Einsatz von Ontologien in der Software-Technik erkennen. Zunächst sind das Ontologien zur Beschreibung der Domäne der Software-Technik selbst (z.B. [SCG05]). Die zweite insgesamt größere Gruppe bilden Einsätze von Ontologien als Artefakte. Analog zu den anderen Artefakten wie Modellen oder Dokumentationen dienen Ontologien zur Herstellung oder Nutzung von Software-Produkten. Das hier vorgestellte OBSE-Verfahren gehört ebenfalls zu dieser Gruppe, d.h. in OBSE wird eine Ontologie als Artefakt während der Software-Entwicklung verwendet.

Knublauch demonstriert [Kn04], wie Ontologien in verschiedenen Phasen der Software-Entwicklung eingesetzt werden können. Verfahren in der Analyse-Phase konzentrieren sich überwiegend auf eine Beschreibung der Domäne in Form einer Ontologie und der Weitergabe dieses Wissens an die Projekte ([deOl06], [De05]). In den späteren Phasen können mit Ontologien zum Beispiel Benutzerschnittstellen konzipiert werden [FG02], oder es kann der Umgang mit dem Code effizienter gestaltet werden [ZR04]. Auch die Übergänge zwischen verschiedenen Phasen (speziell in Bezug auf MDA) können mit Hilfe von Ontologien realisiert werden [CP07].

Schwerpunkt von OBSE ist die Wiederverwendung auf der konzeptuellen Ebene, die hauptsächlich in der Analyse-Phase stattfindet. Der Hauptunterschied von OBSE zu dem Verfahren von Oliveira et al. [deOl06] liegt im Prozess. Während OBSE Wissen über eine Domäne aus den durchgeführten oder laufenden Projekten sammelt, um eine Wiederverwendung zu ermöglichen, wird in [deOl06] eine Ontologie als ein zusätzliches Artefakt speziell für das Projekt erstellt. Das Verfahren von Decker et al. [De05] setzt dagegen ähnliche Mechanismen zur Wiederverwendung wie OBSE ein. Allerdings baut dieses Verfahren auf semantischen Wikis auf, ohne eine direkte Einbindung in die Software-Entwicklungsprozesse zu skizzieren.

3 Der OBSE-Prozess

Der OBSE-Prozess ist als eine Erweiterung zu bestehenden Software-Entwicklungsprozessen konzipiert, für die er eine Möglichkeit bietet, Wissen über eine Domänen-Ontologie auszutauschen. Um diese Einbindung zu ermöglichen, wird eine entsprechende Infrastruktur zur Wiederverwendung und darauf aufbauenden Prozess-Bausteinen benötigt.

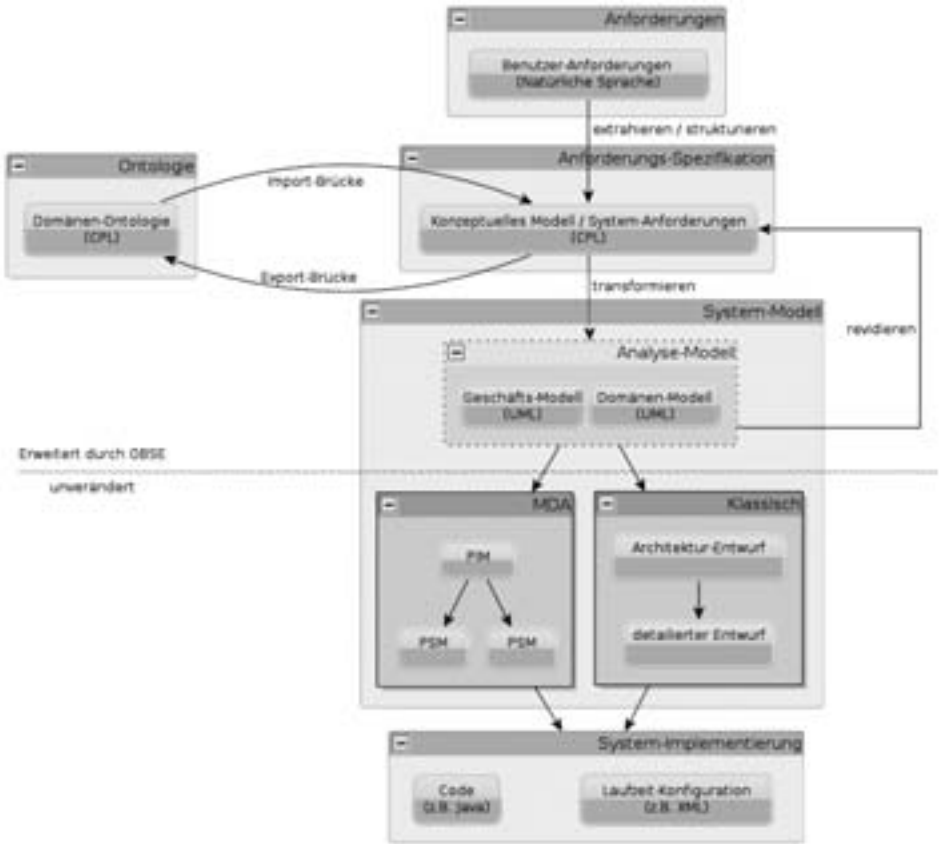

Abbildung 1: Artefakt-basierte Sicht auf den Prozess

Abbildung 1 zeigt eine auf Artefakten und deren Übergängen basierte Übersicht des OBSE-Prozesses. Der Bereich oberhalb der gestrichelten Linie stellt den Anteil des Software-Entwicklungsprozesses dar, der durch OBSE erweitert wird. Die weitere Vorgehensweise, die im unteren Bereich der Abbildung dargestellt ist, folgt bekannten Entwicklungsmodellen. Somit kann OBSE als eine Erweiterung bestehender Vorgehensmodelle angesehen werden, die insbesondere Tätigkeiten im Rahmen der Anforderungsmodellierung eines Vorgehensmodells beeinflusst. Durch eine Anbindung an die Domänen-Ontologie werden in dieser Phase typischerweise zur Verfügung stehende Mittel ausgebaut.

Der Projektablauf, der auf der rechten Seite der Abbildung dargestellt ist, beginnt mit der Analyse der Anforderungen, bei der als Ergebnis Sachverhalte aus Benutzer-Anforderungen extrahiert und formalisiert modelliert werden.

Für diese Anforderungs-Spezifikation baut der OBSE-Prozess auf der konzeptuellen Modellierungssprache *Conceptual Predesign Language (CPL)* auf [Ba07]. Diese Sprache wurde ursprünglich für eine Verbesserung der Kommunikation zwischen den

Fachverantwortlichen auf der Seite des Kunden und von Software-Entwicklern auf der Seite des Projekts konzipierte. Sie basiert auf wenigen Modellierungselementen wie z.B. *Thingtype* zur Beschreibung von Konzepten und *Connectiontype* für die Beschreibung der zugehörigen Beziehungen [MK02]. Entsprechend dieser Zielsetzung existieren für CPL Werkzeuge zur (semi-) automatischen Erzeugung von CPL-Modellen aus natürlicher Sprache [Fl02]. Ein weiterer wesentlicher Grund für die Verwendung von CPL in OBSE war die Eignung als eine Sprache zur Beschreibung von Ontologien. Dies wurde in [KMZ04] gezeigt. Dadurch wurde im OBSE-Prozess die gleiche Sprache sowohl für die konzeptuelle Modellierung der Anforderungen als auch für die Beschreibung der zugehörigen Domäne verwendet, was die Prozessdefinition vereinfacht hat.

Das aus den Anforderungen gewonnene konzeptuelle Modell kann mit Hilfe der Import-Brücke mit dem Wissen aus der Domänen-Ontologie angereichert werden. Welche Elemente dabei in das Projekt importiert werden ist anwendungsabhänging. Diese Entscheidung trifft die für die Brücke zuständige Rolle im Projekt. Gleichzeitig kann an dieser Stelle eine projektinterne Modellierung der Sachverhalte mit bereits erprobter Umsetzung in der Domänen-Ontologie verglichen und ggf. übernommen werden. Dieses Vorgehen soll dazu beitragen, dass die gleichen Sachverhalte in verschiedenen Projekten auf gleiche Weise umgesetzt werden und die Qualität der projektinternen Modelle verbessert wird. Dieser Schritt wird in OBSE durch ein Verfahren zur Ähnlichkeitssuche und Integration auf der konzeptuellen Ebene unterstützt [BV09][Vö10]. Bei diesem Verfahren wird neben der Namensgleichheit auch die semantische Ähnlichkeit mittels *WordNet* überprüft.

Als den nächsten Schritt sieht OBSE eine Transformation des konzeptuellen Modells zu einem Analyse-Modell vor. Dieser Übergang wird durch eine semi-automatische Transformation [Ba10] unterstützt.

Weitere Schritte im Projekt folgen dem entsprechendem Vorgehensmodell und werden von OBSE nicht verändert. Die Abbildung 1 skizziert an dieser Stelle beispielsweise klassische Artefakte oder alternativ MDA-typische Ergebnisse.

Für einen Export des in späteren Phasen des Projekts erworbenen Wissens sollte jedoch ein Vorgehensmodell gewählt werden, welches die Möglichkeit bietet das Analyse-Modell mit den späteren Entwicklungsstufen konsistent zu halten. Mit einer weiteren Transformation von UML zu CPL wird eine Überführung dieser Änderungen in das konzeptuelle Modell unterstützt. Schließlich besteht mit Hilfe der Export-Brücke eine Möglichkeit im Projekt erworbenes Wissen zur Integration in die Domänen-Ontologie zur Verfügung zu stellen.

3.1 OBSE-Infrastruktur

Um den oben beschriebenen Prozess-Ablauf umzusetzen, wurde eine Infrastruktur aufgebaut [Ba10]. Diese baut auf der mehrschichtigen MOF-basierten Modell-Architektur auf. Ausgehend von der Metamodell-Ebene M3, die EMOF/Ecore mit seiner breiten

Werkzeugunterstützung [Wi08] verwendet, werden die einzelnen Infrastruktur-Elemente in der Ebene M1 mit Hilfe der Metamodelle der Ebene M2 definiert.

Entsprechend der Abbildung 1 sind für die OBSE-Infrastruktur folgende Elemente wesentlich: Domänen-Ontologie, konzeptuelles Modell, Analyse-Modell und die beiden Transformationen zwischen UML und CPL. Wie in Abbildung 2 dargestellt, basieren sowohl die konzeptuellen Modelle als auch Domänen-Ontologien auf dem CPL-Meta-Modell. Durch dieses Vorgehen wird auf der konzeptuellen Ebene nur mit einer Sprache gearbeitet, was die Anwendung der Brücken und die Integration der Ontologie-Fragmente vereinfacht.

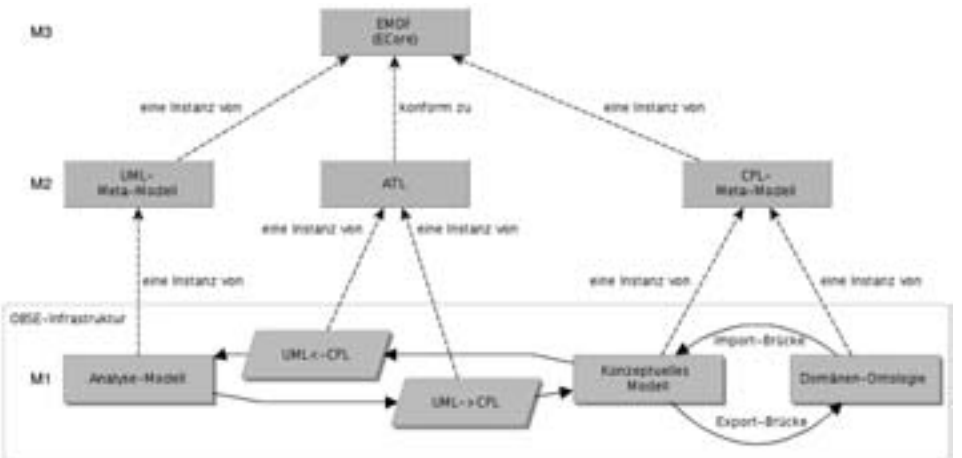

Abbildung 2: Elemente der OBSE-Infrastruktur

Abbildung 2 zeigt die Elemente der Infrastruktur aus der Sicht ihrer Modellierung. Entsprechend der MOF-Architektur existieren drei Schichten, wobei Ecore als Meta-Metamodell die oberste Schicht (M3) bildet. In der zweiten Ebene (M2) befinden sich in Ecore beschriebene Metamodelle für die Sprachen UML und CPL sowie ein Metamodell der *Atlas Transformation Language (ATL)*, womit eine Realisierung der Transformationen zwischen UML und CPL ermöglicht wird. Diese Transformationssprache wurde ausgewählt einerseits wegen ihrer offenen Implementierung und andererseits wegen ihren Reife und praktischer Einsetzbarkeit im Vergleich zu den konkurrierenden Lösungen [FH06], die sich an dem *QVT MOF*-Standard orientieren. Für die Entwicklung der zugehörigen Werkzeuge war weiterhin hilfreich, dass ATL in das Eclipse *Model to Model (M2M)*-Projekt aufgenommen wurde.

Auf der untersten Ebene (M1) befinden sich alle Elemente der OBSE-Infrastruktur als Instanzen der drei Metamodelle. Sie entsprechen den Artefakten bzw. Transformationen den Übergängen zwischen diesen aus Abbildung 1. Der Abstraktionsgrad der Modelle in dieser Ebene nimmt von links nach rechts zu.

3.2 Prozess-Bausteine

Zu einem Software-Entwicklungsprozess gehören neben der Infrastruktur und sich daraus ergebenden Artefakten, auch Rollen, Aktivitäten und Phasen.

Da OBSE vor allem die Analyse-Phase eines Projekts beeinflusst, steht die Rolle eines Systemanalytikers im Mittelpunkt. Die meisten Vorgehensmodelle sehen diese Rolle bereits vor. Aus diesem Grund wird sie in OBSE in der Regel nur erweitert. Ein Systemanalytiker ist in OBSE zusätzlich für das konzeptuelle Modell, das Analyse-Modell und die Transformationen zwischen diesen beiden Modellen verantwortlich. Außerdem ist er für die Anwendung der Import- und Export-Brücken zuständig. Durch die Export-Brücke stellt er in der Form von Ontologie-Fragmenten die Vorschläge für die Anpassungen an der Domänen-Ontologie zusammen. Er hat jedoch keine Rechte, diese selber zu verändern.

Für die Verwaltung der Ontologie werden in OBSE zwei weitere Rollen eingeführt. Zunächst ist dies der Ontologie-Entwickler, der die Domänen-Ontologie pflegt. Unterstützt wird er dabei von einem Domänen-Experten, der mit der Domäne so vertraut ist, dass er verbindliche Aussagen über Konzepte und deren Beziehungen treffen kann. Im Unterschied zu einem Systemanalytiker handelt es sich bei diesen Rollen um projektübergreifende Rollen. Sie werden nicht pro Projekt sondern kontinuierlich für die Pflege und Weiterentwicklung des zentralen Artefakts Domänen-Ontologie benötigt.

3.3 Der System- und Ontologie-Entwicklungsprozess

Die Grundidee von OBSE, die Domänen-Ontologie einerseits auf den in Projekten modellierten Elementen aufzubauen und diese andererseits verzahnt und unterstützend für die Projekt-interne Modellierung zu benutzen, zeigt sich auch in den Phasen des Prozesses.

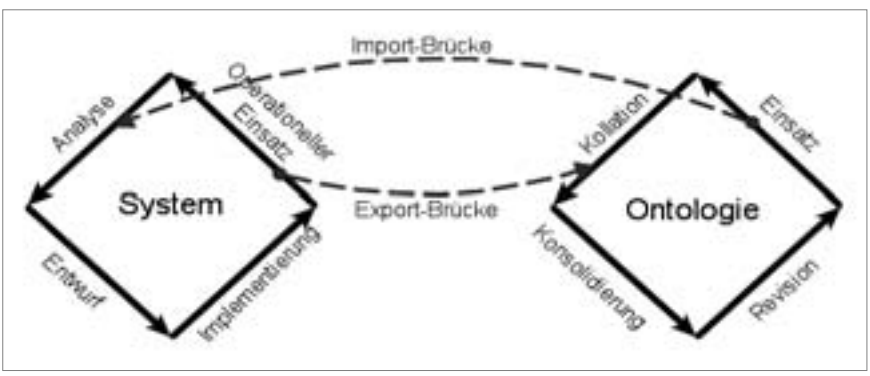

Abbildung 3: Verzahnung von System- und Ontologie-Entwicklungszyklen

Auf der Seite der Systementwicklung werden exemplarisch die Phasen entsprechend des Software-Entwicklungsprozesses *EOS* [He96] definiert. Die vier Phasen eines EOS-Entwicklungszyklus sind *Analyse, Entwurf, Implementierung* und *Operationaler Einsatz*. Sie sind auf der linken Seite in Abbildung 3 dargestellt.

Die Phasen der Ontologie-Entwicklung sind auf der rechten Seite der Grafik zu sehen. Die Entwicklung der Ontologie und jedes neuen Aktualisierungszykluses beginnt mit der *Kollation*. In dieser Phase werden Modell-Fragmente eines Projekts vom Systemanalytiker über die Export-Brücke für eine Integration in die Domänen-Ontologie zur Verfügung gestellt. Während der *Konsolidierunge* werden die in diesen Fragmenten enthaltenen Konzepte und Beziehungen von dem Ontologie-Entwickler daraufhin untersucht, ob und wie sie in die Domänen-Ontologie integriert werden können. Um Konflikte (z.B. unterschiedliche Modellierung gleicher Sachverhalte in den angereihten Fragmenten) aufzulösen, wird der Ontologie-Entwickler von dem Domänen-Experten unterstützt. In der anschließenden *Revision* erfolgt die Integration der ausgewählten Fragmente durch den Ontologie-Entwickler und Freigabe einer neuen Version der Domänen-Ontologie durch den Domänen-Experten. Ab diesem Zeitpunkt steht sie den beteiligten Projekten zur Verfügung und befindet sich somit im *Einsatz*. Der Zugriff auf die Domänen-Ontologie findet dabei über die Import-Brücke statt.

In der Phase *Einsatz* bleibt die Domänen-Ontologie stabil. Neue Entwicklungszyklen führen automatisch zu einer neuen Version der Domänen-Ontologie, die zunächst im Hintergrund die ersten drei Phasen der Ontologie-Entwicklung durchläuft. Erst nach dem Abschluss der *Revision* und einer Freigabe durch den Domänen-Experten ersetzt die neue Version der Domänen-Ontologie die bestehende und wechselt somit in den *Einsatz*.

4 Evaluierung

Für die Erweiterung eines Vorgehensmodells ist es wichtig, deren Praktikabilität möglichst schon im frühen Vorfeld zu erproben. Dabei gewonnene Erkenntnisse und Erfahrungen können noch in die weitere Entwicklung dieses Modells einfließen. Deswegen wurde eine explorative Studie [Ho10] durchgeführt, um erste Erkenntnisse über die Vorteile und Hindernisse zu gewinnen, die sich bei einer praktischen Anwendung des OBSE-Prozesses ergeben. Um dies zu ermöglichen, wurde ein Vergleich von bereits nach anderen Vorgehensmodellen durchgeführten Projekten mit Simulationen dieser Projekte nach OBSE durchgeführt.

Für die Studie wurden zwei bereits abgeschlossene Projekte verwendet, die in der gemeinsamen Domäne *Universität* liegen. Innerhalb dieser Domäne decken die Projekte verschiedene Aspekte des Untersuchungsbereichs *Verwaltung* ab. Sie wurden nach unterschiedlichen Vorgehensmodellen (SCRUM, OpenUP) entwickelt und in unterschiedlichen Programmiersprachen (Java, PHP) implementiert. Alle Projekte wurden am Fachbereich Informatik der Philipps-Universität Marburg von Studenten im Hauptstudium realisiert und die entwickelten Systeme sind in zwei verschiedenen Fachbereichen der Universität im Einsatz. Zusätzlich wurde ein drittes Projekt für die Erzeugung einer initialen Domänen-Ontologie verwendet.

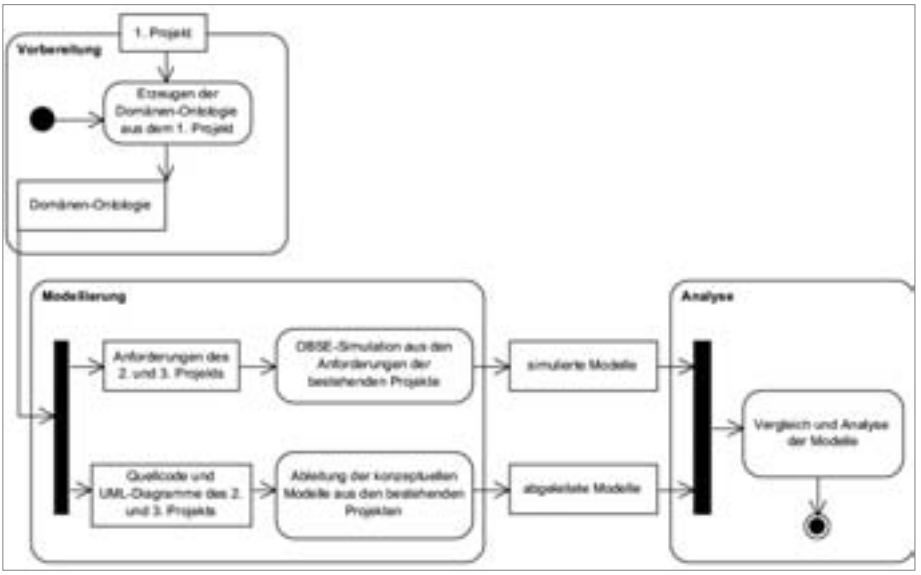

Abbildung 4: Aktivitäts-Diagramm des Ablaufs der Evaluation

Der Ablauf der Studie war im Wesentlichen in drei Phasen aufgeteilt: Vorbereitung, Modellierung und Analyse. In der Vorbereitungsphase wurde aus den Anforderungen und dem Quellcode des ersten Projekts ein CPL-Modell erstellt. Aus diesem Modell wurden Domänen-spezifische Konzepte und Relationen in eine leere Domänen-Ontologie nach dem OBSE-Prozess exportiert. Diese Vorbereitung simulierte den Aufbau der Domänen-Ontologie durch vorhergehende Projekte.

Im ersten Schritt der Modellierungsphase wurde der OBSE-Prozess an den Projekten 2 und 3 (namentlich FoPra-Verwaltung und SpSemOnline) simuliert. Dazu wurde anhand der ursprünglich vorgegebenen Anforderungen für jedes Projekt ein in CPL formuliertes konzeptuelles Modell der dort enthaltenen Konzepte und Relationen erstellt. Diese Modelle wurden dann über die Import-Brücke mit Konzepten und Relationen aus der Domänen-Ontologie ergänzt. Auf diese Weise entstanden nach dem OBSE-Prozess simulierte Modelle von beiden Projekten (im Folgenden: „simulierte Projekte).

Im zweiten Schritt der Modellierungsphase wurden konzeptuelle Modelle aus den entstandenen Artefakten der beiden Projekte abgeleitet. Dies wurde sowohl mittels Reverse-Engineering als auch durch eine Transformation eines UML-Klassendiagramms in ein CPL-Modell durchgeführt. Die aus diesem Schritt entstandenen Modelle repräsentieren die konzeptuelle Sicht auf die tatsächlich entstandenen Artefakte der Projekte (im Folgenden: „abgeleitete Projekte“).

In der Analysephase wurden zunächst die Auswirkungen der Import-Brücke auf ein konzeptuelles Modell untersucht, welches aus den Anforderungen erstellt wurde. Anschließend wurden die Ergebnisse der beiden Schritte der Modellierungsphase mit einander verglichen. In beiden Vergleichen wurden sowohl quantitative als auch qualitative Auswirkungen des OBSE-Prozesses auf die konzeptuellen Modelle untersucht.

4.1 Quantitative Ergebnisse der Untersuchung

Die Tabelle 1 zeigt einen quantitativen Vergleich der Konzepte und Relationen der simulierten Modelle vor und nach der Anwendung der Import-Brücke. In der ersten Spalte stehen aufsummiert die Modellierungselemente aus dem simulierten Modell vor der Anwendung der Import-Brücke. In der zweiten Spalte steht die Anzahl der Elemente des gleichen Modells nach der Anwendung der Import-Brücke.

Projekt	Vor dem Import		Nach dem Import	
	Konzepte	Relationen	Konzepte	Relationen
FoPra-Verwaltung	30	36	41	48
SpSemOnline	34	42	44	53

Tabelle 1: Quantitativer Vergleich der Modelle des Imports

Es ist zu erkennen, dass die Anzahl der Konzepte und Beziehungen nach dem Import um etwa 25% zunahm. Es wurden also in beiden Projekten eine ähnliche Anzahl Konzepte und Relationen importiert.

Abbildung 5 zeigt als Beispiel einen Auszug der quantitativen Ergänzung des Modells des simulierten SpSemOnline-Projekts durch die Anwendung der Import-Brücke. Die eckigen Elemente in der Abbildung stehen für Konzepte, während die runden Elemente und Verbindungen die Relationen darstellen. Weiterhin stehen die Pfeile für eine Generalisierungs-Beziehung und die Kennzeichner unterhalb des Bezeichners der Konzepte für den Ursprung des jeweiligen Konzepts. <<s_a>> steht für das Anforderungsmodell des SpSemOnline-Projekts und <<d>> für die Domänen-Ontologie.

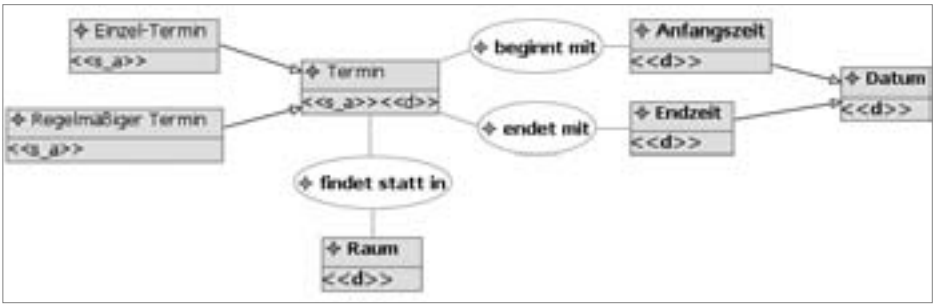

Abbildung 5: Quantitative Import-Brücke – Import von Konzepten und Beziehungen

Die fettgedruckten Konzepte waren nicht in dem Anforderungsdokument des Projekts enthalten. In diesem wurde nur ein Termin beschrieben, nicht jedoch dessen Bestandteile. Diese zusätzlichen Konzepte und ihre Relationen wurden durch die Anwendung der Import-Brücke des OBSE-Prozesses in das Modell integriert.

Ein Vergleich dieses simulierten Modellausschnitts mit der im Projekt entstandenen Umsetzung zeigt, dass die fehlenden Elemente im Projektverlauf ergänzt wurden und somit für die Umsetzung der Anforderungen tatsächlich erforderlich waren. Ein Vorteil von OBSE besteht somit darin, dass diese Elemente bereits in der Analyse-Phase in die Modelle einfließen und nicht in späteren Phasen ergänzt werden müssen, wie es in dem SpSemOnline-Projekt der Fall war.

4.2 Qualitative Ergebnisse der Untersuchung

Für die qualitative Betrachtung der Modelle wurden Modellqualitäts-Richtlinien aus [FHR08] herangezogen. Nach diesen wurden sowohl die Anwendungen der Import-Brücke untersucht, als auch die simulierten Modelle mit den abgeleiteten Modellen verglichen.

Ein Beispiel für die qualitative Verbesserung durch eine Anwendung der Import-Brücke ist in Abbildung 6 dargestellt. Im oberen linken Teil (a) ist ein Ausschnitt des simulierten Modells aus dem SpSemOnline-Projekt vor der Anwendung der Import-Brücke dargestellt. Der rechte Teil (b) zeigt einen Ausschnitt der Domänen-Ontologie. In dem unteren Teil der Abbildung (c) sieht man den Ausschnitt (a) nach der Anwendung der Import-Brücke.

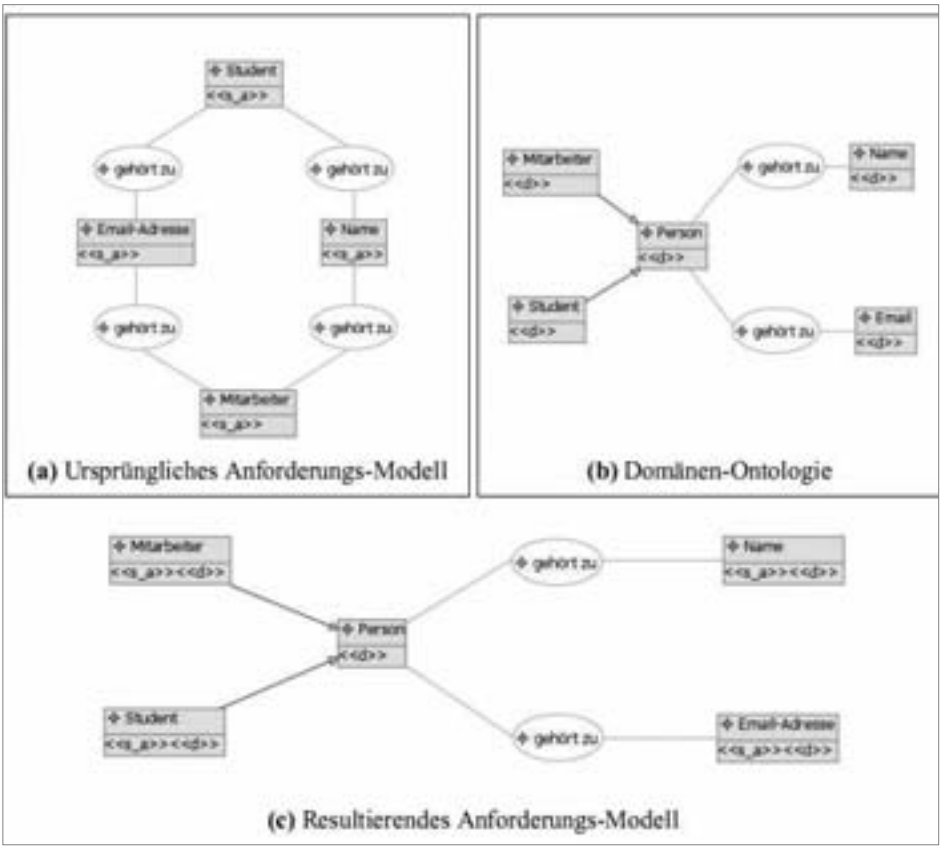

Abbildung 6: Qualitative Import-Brücke – Verbesserung der Änderbarkeit

Aus den Anforderungen des Projekts ging hervor, dass sowohl *Studenten* als auch *Mitarbeiter* über einen *Namen* und eine *Email-Adresse* verfügen, sonst jedoch keine Gemeinsamkeiten besitzen. Dies wurde in (a) gemäß den Anforderungen modelliert. In der Domänen-Ontologie hingegen waren *Mitarbeiter* und *Studenten* eine spezielle Form einer *Person*. Diesem zentralen Konzept *Person* waren *Email* und *Name* zugeordnet.

Durch die Integration des Konzepts *Person* in (c) können zu einem späteren Zeitpunkt andere Personen, wie z.B. ein Berater, über eine Generalisierungs-Beziehung modelliert werden. Weiterhin kann *Person* mit weiteren Relationen zu Konzepten wie z.B. einer Adresse oder einer Telefonnummer verknüpft werden.

In (a) hingegen müssten für ein Konzept *Berater* Relationen zu *Email-Adresse* und *Name* angelegt werden. Wenn dann ein Konzept *Telefonnummer* für alle Personen hinzugefügt werden soll besteht die Möglichkeit, dass eines der drei Konzepte vergessen wird. Durch die Anwendung der Import-Brücke konnte in diesem Beispiel somit die *Änderbarkeit* des simulierten Modells erhöht werden.

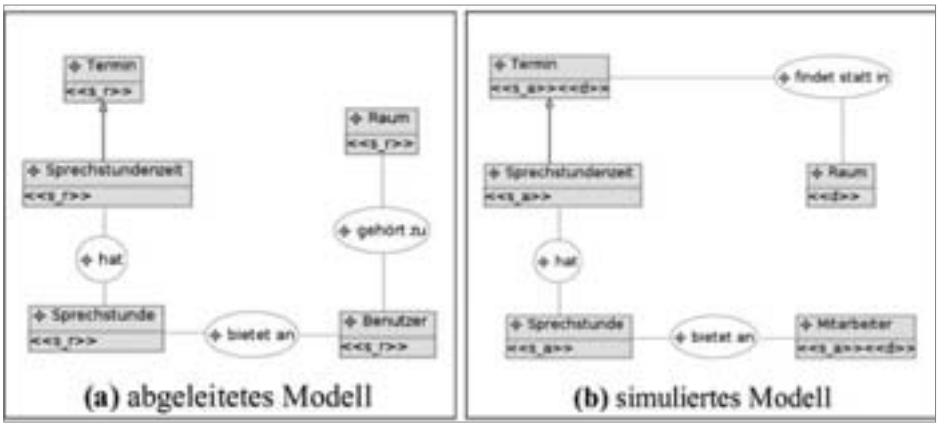

Abbildung 7: Qualitativer Abgleich der Modelle – Verbesserung der Änderbarkeit

Abbildung 7 zeigt ein Beispiel aus dem Abgleich der simulierten Modelle mit den abgeleiteten Modellen, in dem *Raum* das zentrale Konzept darstellt. Ein *Benutzer* (a) bzw. *Mitarbeiter* (b) bietet eine *Sprechstunde* an, die in einem *Raum* stattfindet. In den Anforderungen des Projekts wurde dieser *Raum* nicht mit aufgenommen. In dem beendeten Projekt wurde der Raum nachträglich angelegt und direkt mit dem *Benutzer* in Beziehung gesetzt. In der Domänen-Ontologie hingegen existierte eine Beziehung zwischen *Termin* und *Raum*, wie in Abbildung 5 gezeigt wurde. Diese wurde bei der Verwendung der Import-Brücke als benötigtes Element identifiziert und in das simulierte Modell integriert.

Das simulierte Modell zeigt in diesem Beispiel eine bessere Änderbarkeit als das aus dem beendeten Projekt abgeleitete Modell. Da *Raum* mit *Termin* verknüpft ist, kann zu jedem Termin ein eigener Raum angegeben werden. In dem beendeten Projekt hingegen hat jeder Benutzer einen festen Raum in dem seine Sprechstunden abgehalten werden. Dieser kann nur schwer kurzfristig verändert werden und auch eine Terminplanung in zwei unterschiedlichen Räumen an zwei verschiedenen Tagen ist nicht möglich.

In den beiden simulierten Modellen wurden bei diesem Vergleich jeweils sechs Modellausschnitte identifiziert, die qualitativ besser umgesetzt worden waren als in den realisierten Projekten.

Die Verbesserungen der Modellqualität wurden bei den Qualitäts-Richtlinien *Korrektheit*, *Änderbarkeit*, *Redundanzfreiheit, Präzision* und *Verfolgbarkeit* erzielt. Die Analyse ergab außerdem, dass die *Einfachheit* der OBSE-Modelle nicht qualitativ schlechter war als bei den rekonstruierten Modellen. Das heißt, die aus der Domänen-Ontologie importierten Konstrukte konnten nutzbringend übernommen und eingesetzt werden.

5 Zusammenfassung und Ausblick

Mit dem OBSE-Prozess wurde eine Erweiterung zu bestehenden Software-Entwicklungsprozessen geschaffen, die den Aufbau und die Nutzung von Domänen-Ontologien mit dem "normalen" Projektablauf verbindet und bei dem die Entwickler von dort gesammeltem Domänenwissen profitieren können. Der Prozess gründet sich auf einer bewährten konkreten Modellierungssprache, erprobten Transformationen und Prozessbeschreibungen. Alle Modelle und Transformationen erlauben eine Integration in die MOF-basierte Modell-Architektur. Diese Voraussetzungen ermöglichen es, OBSE in realen Software-Projekten zur Analyse des Gegenstandsbereichs und der konzeptuellen Modellierung der Anforderungen einzusetzen und für die späteren Phasen mit anderen bekannten und bewährten Vorgehensmodellen zu verbinden.

Eine erste Evaluierung des Prozesses ergab eine qualitative Verbesserung von nach OBSE simulierten Modellen im Vergleich zu den ohne OBSE erstellten Modellen. Die in CPL formulierte Domänen-Ontologie zeigte sich dabei als adäquates Mittel, Wissen zwischen Projekten auszutauschen. Damit konnte gezeigt werden, dass eine positive quantitative und qualitative Veränderung von Software-Projekten auf der konzeptuellen Ebene schon bei wenigen teilnehmenden Projekten möglich ist. Das Verfahren richtet sich somit an Entwickler, die mehr als ein Projekt in der gleichen Domäne realisieren wollen sowie an solche, die sich die Ergebnisse früherer oder paralleler Projekt-Entwicklungen zunutze machen wollen. Dabei ist kritisch anzumerken, dass die Domänen-Ontologie bei einer geringen Anzahl an beteiligten Projekten nicht immer optimale Ergebnisse liefert. Weiterhin sollte die Anwendung der Export-Brücke aus Projekten in die Domänen-Ontologie kontrolliert erfolgen.

Aktuell wird noch an der Verbesserung der Integration von CPL-Modellen und deren Einbindung in die Werkzeuge gearbeitet. Zukünftige Projekte in diesem Bereich könnten darauf abzielen, die Möglichkeiten des *Reverse Engineering* mit CPL weiter zu erforschen. Wenn Reverse Engineering-Verfahren für eine breitere Auswahl an Sprachen zur Verfügung stünden, könnte Domänenwissen aus Nicht-OBSE-Projekten ebenfalls der Domänen-Ontologie zur Verfügung gestellt werden.

Weitere künftige Untersuchungs- und Forschungspunkte betreffen die Anknüpfung an weitere Modellierungstechniken und Vorgehensmodelle sowie den Einbau von konstruktiven Qualitätssicherungsmaßnahmen in die Domänen-Ontologien.

Literaturverzeichnis

[Ba10] A. Bachmann: Methoden- und Werkzeugunterstützung für Ontologie-basierte Software-Entwicklung. Dissertation, FB Mathematik und Informatik, Univ. Marburg 2010.

[Ba07] A. Bachmann, W. Hesse, A. Russ, Ch. Kop, H.C. Mayr, J. Vöhringer: OBSE – an Approach to Ontology-based Software Engineering in the Practice. Proc. EMISA conference, St. Goar, pp. 129-142, LNI, Koellen-Verlag, 2007.

[BV09] Peter Bellström und Jürgen Vöhringer: Towards the Automation of Modeling Language Independent Schema Integration. In: Proceedings of the International Conference on Information, Process, and Knowledge Management eKNOW '09, Seiten 110 - 115. Springer, 2009.

[CP07] S. Cranefield und J. Pan: Bridging the gap between the model-driven architecture and ontology engineering. International Journal of Human-Computer Studies, 65(7):595-609, 2007.

[deOl06] K. de Oliveira, K. Vilela and A. R. Rocha und G. H. Travassos: Use Ontologies in Software Development Environments, Kapitel 10. Springer, 2006.

[De05] B. Decker, E. Ras, J. Rech, B. Klein und C. Hoecht: Self-organized Reuse of Software Engineering Knowledge supported by Semantic Wikis. In: Workshop on Semantic Web Enabled Software Engineering (SWESE), held at the 4th International Semantic Web Conference ISWC, 2005.

[FG02] R. De Almeida Falbo und G. Guizzardi: An Ontological Approach to Domain Engineering. In: In Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, Seiten 351-358. ACM Press, 2002.

[FH06] Mario Friske und Konrad Hilse: Evaluation von Transformationsmaschinen in der modellbasierten Qualitiätssicherung. In: Christian Hochberger und Rüdiger Liskowsky (Herausgeber): GI Jahrestagung, Band 94 der Reihe LNI, Seiten 205§209. GI, 2006.

[Fl02] G. Fliedl, C. Kop, W. Mayerthaler, H.C. Mayr, C. Winkler: The NIBA workflow: From textual requirements specifications to UML-schemata. In: ICSSEA2002 – International Conference "Software & Systems Engineering and their Applications", Paris, 2002.

[FHR08] F. Fieber, M. Huhn und B. Rumpe: Modellqualität als Indikator für Softwarequalität: eine Taxonomie. Informatik Spektrum, 31(5): 408-424, 2008.

[GDD06] Dragan Gacevic, Dragan Djuric und Vladan Devedoic: Model Driven Architecture and Ontology Development, Springer, Berlin, Heidelberg, 1. Auflage, 2006.

[He96] W. Hesse: Theory and practice of the software process - a field study and its implications for project management; in: C. Montangero (Ed.): Software Process Technology, 5th European Workshop, EWSPT 96, pp. 241-256. LNCS 1149, Springer 1996.

[He02] W. Hesse: Das aktuelle Schlagwort: Ontologie(n). in: Informatik-Spektrum 25.6, S.477-480, 2002.

[He05] W. Hesse: Ontologies in the Software Engineering process. In: R. Lenz et al. (Eds.): EAI 2005 - Tagungsband Workshop on Enterprise Application Integration, GITO-Verlag Berlin, 2005 and: http://sunsite.informatik.rwth-aachen.de/ Publications/CEUR-WS/Vol-141/.

[Ho10] B. Horst: Evaluierung von Software-Prozessen zur Ontologie-basierten Software-Entwicklung. Diplomarbeit. FB Mathematik und Informatik, Univ. Marburg 2010.

[Kn04] H. Knublauch: Ontology-Driven Software Development in the context oft the Semantic Web: An Example Scenario with Protege/Owl. In: David S. Frankel, Elisa F. Kendall und Deborah McGuiness (Hrsg.): 1st International Workshop on the Model-Driven Semantic Web (MDSW2004), 2004.

[KMZ04] C. Kop, H.C. Mayr und T. Zavinska: Using KCPM for Defining and Integrating Domain Ontologies. In: WISE Workshops, Band 3307, Lecture Notes in Computer Science, S. 190-200. Springer, 2004.

[MK02] H.C. Mayr, C. Kop: A User Centered Approach to Requirements Modeling, Proc. Modellierung 2002, S. 75-86. LNI p-12, Koellen-Verlag, 2002.

[SCG05] Miguel Ángel Sicilia, Juan José Cuadrado, Elena García, Daniel Rodríguez und José R. Hilera: The Evaluation of ontological representations of the SWEBOK as a revision tool, 2005

[UG96] M. Uschold und M.l Grüninger, Ontologies: principles, methods, and applications, S. 93–155, Knowledge Engineering Review, Vol. 11, Nr. 2, 1996.

[Vö10] J. Vöhringer, D. Gälle, G. Fliedl, Ch. Kop, M. Bazhenov (2010): Using linguistic knowledge for fine-tuning ontologies in the context of requirements engineering. In: International Journal of Computational Linguistics and Applications, 2010.

[Wi08] E. D. Willink: Adapting EMOF/EssentialOCL/QVT to Ecore/MDT-OCL/EQVT. In: Eclipse OMG Symposium, 2008.

[ZR04] C. Zimmer und A. Rauschmayer: Tuna: Ontology-Based Source Code Navigation and Annotation. In: Workshop on Ontologies as Software Engineering Artifacts (OOPSLA), 2004.