Zimmermann, Frank

Working Paper

Vergleich von TOPCASED und GMF mit der Atos Origin Methode QSOS im Rahmen von modellgetriebener Softwareentwicklung

Arbeitspapiere der Nordakademie, No. 2007-04

Provided in cooperation with:
Nordakademie - Hochschule der Wirtschaft

ARBEITSPAPIERE DER NORDAKADEMIE

ISSN 1860-0360

Nr. 2007-04

Vergleich von TOPCASED und GMF mit der Atos Origin Methode QSOS im Rahmen von modellgetriebener Softwareentwicklung

Prof. Dr. Frank Zimmermann

Dezember 2007

Eine elektronische Version dieses Arbeitspapiers ist verfügbar unter:
http://www.nordakademie.de/index.php?id=ap

Köllner Chaussee 11
25337 Elmshorn
http://www.nordakademie.de

Vergleich von TOPCASED und GMF mit der Atos Origin Methode QSOS im Rahmen von modellgetriebener Softwareentwicklung *

Frank Zimmermann

20.12.2007

Zusammenfassung

Unter modellgetriebener Software Entwicklung (siehe MDSD (Bettin, 2004), (Stahl u. a., 2006)) werden Vorgehensmodelle verstanden, bei denen Software nicht primär in einer universellen Programmiersprache wie Java, C++, C# oder Smalltalk entwickelt wird, sondern bei der spezielle domänenspezifische „Programmiersprachen“ verwendet werden, die sich besonders eignen, Probleme dieser speziellen Anwendungsdomäne zu lösen. Diese Domain Specific Languages (DSL) können textuell oder visuell sein. Bei den visuellen DSLs war man aufgrund der hohen Komplexität, die die Programmierung von visuellen Umgebungen erfordert, bisher auf die Verwendung von Software großer Hersteller (siehe (metacase.com, 2007), (Tolvanen, 2006) , (Swithinbank u. a., 2005), (Fong, 2006)) angewiesen. Mit dem Eclipse Modeling Project stehen erstmals Open Source Werkzeuge zur Erstellung von visuellen Modellierungssprachen zur Verfügung. Der Anwender hat die Möglichkeit, das Graphical Modelling Framework (Gronback, 2008) oder TOPCASED (Lescot, 2008) zu verwenden. Anhand eines Anwendungsbeispiels sollen die beiden Werkzeugkästen miteinander verglichen werden. Dabei stellt sich heraus, dass TOPCASED zwar einen einfacheren Einstieg erlaubt, GMF aber das ausgereiftere Werkzeug mit umfassenderer Funktionalität ist.

1 Einleitung

1.1 Modellgetriebene Softwareentwicklung

Im November 2000 veröffentlichte die Object Management Group (OMG) das Konzept der Model Driven Architecture (MDA) (Miller und Mukerji, 2003). Die grundlegenden Ideen von MDA umfassen die Konzepte Generative Programming, Domain Specific Languages, Model-integrated computing, Software Factories, etc.. Zentral in dem MDA Konzept ist der Begriff des Modells (Bézivin, 2005) , das zur Beschreibung von Artefakten der Softwareentwicklung auf verschiedenen Abstraktionsebenen verwendet wird.

Während MDA eher als Konzept verstanden werden muss, ist das Eclipse Modeling Project eine Umsetzung dieser Ideen in konkrete Softwareprodukte. MDA verwendet in seiner Definition OMG Standards wie MOF [1], XMI [2] , OCL [3], UML[4] etc. . Das Eclipse Modeling Projekt (EMP) vermeidet aus Gründen der praktischen Umsetzbarkeit die Benutzung der äußerst komplexen OMG Standards und nutzt seine eigenen vereinfachten Standards. Das EMP stellt dabei Werkzeuge zur Modellbeschreibung, -bearbeitung, -speicherung und -transformation zur Verfügung. Obwohl eine visuelle Repräsentation nicht Grundlage einer Anwendung modellgetriebener Softwareentwicklung ist, ist es jedoch eine sehr nützliche Ergänzung (Bézivin, 2005) . Eine sinnvolle Umsetzung einer modellgetriebenen Softwareentwicklung muss daher über Werkzeuge verfügen, in denen die Bearbeitung von Modellen mit visuellen Bearbeitungswerkzeugen möglich ist. Um die Flexibilität nicht von vornherein einzuschränken, ist es

[1] Das Meta Object Facility dient zur Beschreibung von Metamodellen

[2] XML Model Interchange dient zum Austausch von Modellen

[3] Die Object Constraint Language dient zur Beschreibung von Vor- und Nachbedingungen für Modellelemente

[4] Die Unified Modeling Language dient zu Beschreibung von objektorientierten Softwaresystemen

sinnvoll, sich nicht nur auf die Manipulation von UML-Diagrammen zu beschränken, sondern die visuelle Bearbeitung domänenspezifischer Sprachen mit beliebigen Meta Modellen zu ermöglichen. Bei den beiden Projekten TOPCASED und GMF handelt es sich nun um Frameworks, die die Erstellung von graphischen Modelleditoren wesentlich erleichtern. In diesem Arbeitspapier wird die Frage untersucht, wie die beiden Projekte zueinander stehen, welche Unterschiede zu erkennen sind und wann ein Einsatz des jeweiligen Produkts sinnvoll ist.

2 Vergleich von TOPCASED Modeller und GMF

2.1 Vorgehen

Der Aufbau dieses Arbeitspapieres gliedert sich in fünf Schritte. Zunächst erfolgt eine Standortbestimmung. Die Einbettung der Werkzeuge GMF und TOPCASED in das Eclipse Modeling Project wird dargestellt. Danach wird die eine kurze Übersicht über die Vergleichsmethode gegeben. Um den speziellen Anforderungen zur Evaluation von Open Source Projekten Rechnung zu tragen, wird die Methode QSOS (Qualification and Selection of Open Source software (Semeteys, 2006)) herangezogen. Zentraler Bestandteil des Vergleichs bleiben jedoch Anforderungen eines konkreten Beispielprojekts. Dieses Projekt wird vorgestellt und die funktionalen und nichtfunktionalen Anforderungen werden in Form von User Stories aufgelistet. Danach wird die Bewertung der beiden Werkzeuge anhand der QSOS Methode durchgeführt. Schließlich erfolgt eine Zusammenfassung der Ergebnisse.

2.2 Das Eclipse Modelling Project

Modellgetriebene Entwicklung im Sinne der OMG findet große Beachtung in der Entwicklung von Informationssystemen. Getrieben von der hohen technischen Komplexität heutiger Anwendungsarchitekturen suchen viele Unternehmen nach Möglichkeiten, ihre Softwareentwicklungen zu vereinfachen. Microsoft bietet mit dem Konzept der Softwarefactories (Greenfield u. a., 2004) einen herstellerunterstützten Ansatz, der als Gegenpol zur Initiative der OMG gesehen werden kann. Das Eclipse Modeling Project (EMP) der Eclipse Software Foundation ist eine Zusammenstellung vieler Open Source Projekte, die die Konzepte der OMG in die Praxis umsetzen. Einen Vergleich zwischen dem EMP und Microsoft Visual Studio findet man in (Slapeta, 2006).

Kern des EMP ist das Eclipse Modeling Framework, das eine Verwaltung von Meta Modellen mittels des Ecore Standards ermöglicht. Ecore kann als eine Reduzierung des MOF Standards auf die wesentlichen Komponenten EMOF angesehen werden. Ecore ermöglicht die Beschreibung von Metamodellen, und EMF stellt Werkzeuge zur Verfügung, um zu einem gegebenen Metamodell Modelle zu erzeugen (vgl. Abbildung 1). EMF kann zu jedem Ecore Modell ein Java Binding zur Verfügung stellen. Die Java Objekte können in einer XMI Datei persitiert werden. EMF stellt für die gespeicherten Modelle weitere Werkzeuge zur Verfügung. So kann man zur Manipulation von Modellen Editoren generieren, die die textbasierte Bearbeitung von EMF Modellen ermöglichen. Abbildung 1 zeigt die Zusammenhänge des EMP mit EMF als zentralem Bestandteil.

Neben der textbasierten Bearbeitung von Modellen ist es für ein umfassendes MDA Konzept unerlässlich, auch graphische Werkzeuge zur Modellmanipulation anzubieten. Die textbasierte und die graphische Variante ergänzen sich und kommen beide je nach beabsichtigtem Anwendungszweck zum Einsatz.

Zum Erstellen von graphischen Editoren in Eclipse gibt es das Graphical Editing Framework (GEF). Es basiert auf klassischen Entwurfsmustern wie Model View Controller und bietet so eine gute Basis für erweiterbare graphische Editoren. Während View und Controller in GEF durch eigene Komponenten (wie z.B. EditParts, EditPolicies, Commands und Draw2D) vollständig abgedeckt sind, ist Implementierung der Model Komponente offen gelassen. Es gibt zahlreiche Ansätze EMF, in der Modell Komponente von GEF einzusetzen (Moore u. a., 2004). Die Kombination beider Frameworks ist eine Herausforderung für Softwarearchitekten, denn beide Frameworks sind komplex und offen für Erweiterungen.

GMF und TOPCASED können als Brücke zwischen GEF und EMF angesehen werden. Auf Basis einer Definitionsdatei, die zweckmäßigerweise wieder auf EMF basiert, für den graphischen Editor wird eine

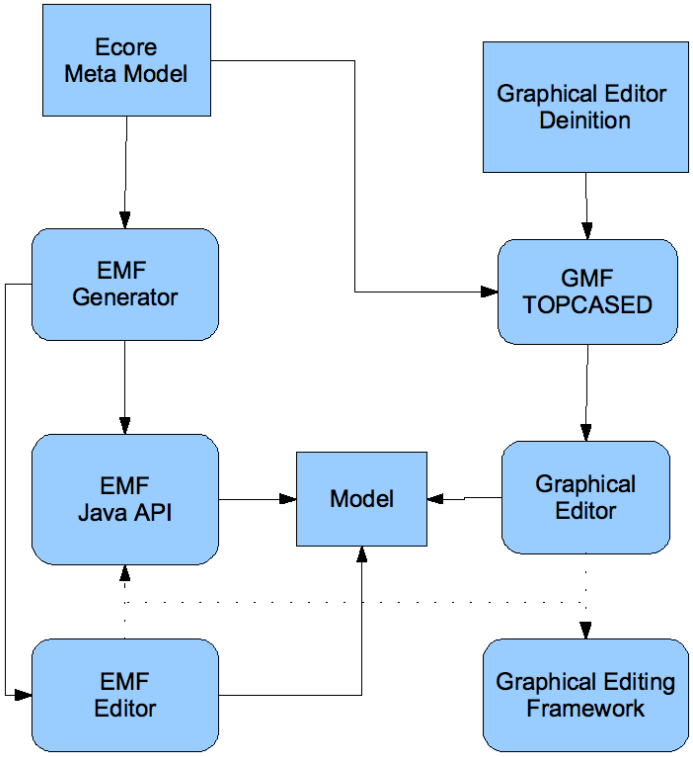

Abbildung 1: Komponenten des Eclipse Modelling Project

GEF Anwendung generiert, deren Backend EMF darstellt. Dabei sind die genutzten Frameworks von GMF und TOPCASED zwar identisch, die Architektur der generierten Anwendungen unterscheidet sich jedoch stark.

2.3 Vergleichskriterien

Ein Vergleich zweier komplexer Frameworks ist besonders schwierig, wenn diese Frameworks Open Source sind. Aus Sicht eines potentiellen Anwenders spielen neben der eigentlichen Funktionalität und Architektur des Produkts auch eine Reihe weiterer Soft-Faktoren eine Rolle, die im Kern das Risikomanagement für den Einsatz eines Open Source Produktes darstellen. Aus Gründen des Investitionsschutzes haben viele Unternehmungen deshalb Hemmungen, sich mit Open Source Software auseinanderzusetzen. Durch die Bewertung der Software nach Reife Gesichtspunkten können diese Bedenken objektiviert werden. Es gibt zahlreiche Alternativen, eine solche Bewertung durchzuführen. Beispiele hierzu findet man unter (Carnegie Mellon West Center for Open Source Investigation u. a., 2005), (Golden, 2004) und (van den Berg, 2005) .

Um die weichen Faktoren ausreichend zu berücksichtigen, wird in dieser Arbeit die von Atos Origin entwickelte Methode Qualification and Selection of Opensource Software (Semeteys, 2006) verwendet. Neben einer Bewertungsmethode stellt QSOS auch einen Fragenkatalog mit Bewertungsrahmen zur Verfügung. Es wird davon ausgegangen, dass der Bewertungsrahmen den Anforderungen eines durchschnittlichen Unternehmens entspricht.

Das QSOS Vorgehensmodell sieht vier Schritte vor

1. Definiton

   Hier wird das Produkt identifiziert und gemäß einer vorgegebenen Schablone typisiert.

2. Evaluation

   Dieser Schritt besteht aus der Beurteilung des Produkts in drei Dimensionen

(a) Funktionale Abdeckung

(b) Risiken des Anwenders

(c) Risiken für den Anbieter

Auf die Bewertung der Risiken für den Anbieter wurde verzichtet.

3. Qualifikation

   Die Gewichtung der evaluierten Kriterien.

4. Selektion

   Anwendung der Gewichtungen aus Schritt 3 auf die in Schritt 2 erhobenen Daten und Auswahl der Software.

2.4 Beispielszenario

Um die Funktionalität des verwendeten Frameworks anhand konkreter Anforderungen prüfen zu können, ist ein Referenzbeispiel erforderlich. Die Aufgabe, die mit dem Modellierungswerkzeug erledigt werden sollte, besteht in der Steuerung von eingebetteten Systemen mittels Zustandsautomaten. Um den Aufwand, der in der Entwicklung einer domänenspezifischen Sprache entsteht, zu rechtfertigen, muss eine „programmierintensive“ Zielumgebung verwendet werden. Lego Mindstorm NXT Roboter ermöglichen eine Vielfalt unterschiedlicher Anwendungsprogramme. Die Entwicklung einer DSL ist hier insbesondere deshalb gerechtfertigt, weil es keine vergleichbaren Anwendungen gibt.

Um sinnvolle NXT Anwendungen programmieren zu können, muss eine DSL folgende User Stories erfüllen:

F1. Erfassung, Darstellung und Bearbeitung von Zuständen als Knoten im visuellen Editor. Die Form der Knoten sollte der Darstellung von Zuständen in der UML ähnlich sein.

F2. Erfassung, Darstellung und Bearbeitung von Zustandsübergängen als Kanten im visuellen Editor. Multiplizität der Kanten von Zustand zu Event M:N und von Event zu Zustand 1:M.

F3. Erfassung, Darstellung und Bearbeitung von zustandsübergangsauslösenden Ereignissen als Knoten im visuellen Editor. Die Form soll Signalen in der UML ähnlich sein.

F4. Erfassung und Bearbeitung von Anweisungen, die eine Verhaltensspezifikation ermöglichen.

F5. Erfassung und Bearbeitung von Bedingungen, die ein auslösendes Ereignis beschreiben.

F6. Erfassung von Zustandsvariablen, um eine komplette Generierung zu ermöglichen.

Darüberhinaus muss die Anwendung noch folgende nichtfunktionalen Anforderungen erfüllen:

NF1. Programmierung des Editors soll so weit wie möglich im Standard durchgeführt werden.

NF2. Erzeugung von lauffähigen Javaprogrammen muss möglich sein.

NF3. Integration in Eclipse und Anschluss an das nachgelagerte Generatorframework oAW muss übergangslos sein.

NF4. Diagramme müssen übersichtlich und leicht zu modifizieren sein.

Zur Bewertung wird für jede Anforderung ein Score vergeben. Dabei bedeutet 0=Anforderung nicht erfüllt, 1=Anforderung ausreichend erfüllt, 2=Anforderung gut erfüllt.

2.5 TOPCASED

2.5.1 Definition des Produktes

Das Toolkit in Open source for Critical Aeronautic SystEms Design(TOPCASED) (Pontisso und Chemouil, 2006) ist ein Open Source Produkt, das als Kooperationsprojekt namhafter Unternehmen (Airbus, EADS Astrium, Siemens VDO, CNES, ...) und akademischer Institutionen (INP Toulouse ENSEEIHT, INSA Toulouse, Universite Paul Sabatier Toulouse III, ENSIETA Brest, ...) entstanden ist. Ziel des TOPCASED-Projekts ist die langfristig angelegte Unterstützung der Softwareentwicklung in kritischen Anwendungsbereichen.

Das TOPCASED Meta Modeling Project ist eines (WP2) von neun Arbeitspaketen des mit 17M€ budgetierten Gesamtprojekts. Mit Hilfe von topcased-MM von Lead Developer Jaques Lescot können visuelle Editoren für die modellgetriebene Entwicklung auf Basis von EMF erstellt werden. Es integriert sich in das Eclipse Modeling Project, insbesondere werden die Generatortechnologien oAW und ATL (Universität Nates) unterstützt. Die Projektseite `http://www.topcased.org` listet das Release 0.11 vom 6. November 2006 als erstes verfügbares Release. Die Software steht auf der Seite `https://gforge.enseeiht.fr/projects/topcased-mm` zum Download bereit. Zum Einsatz in der Evaluation kam das Release 1.0.0 vom 7. Juli 2007. Es steht unter der Eclipse Public Licence. Die im Internet verfügbare Dokumentation beschränkt sich auf einige Tutorials `http://topcased-mm.gforge.enseeiht.fr/website/modeling/documentation.html`. Eine grobe Darstellung der Architektur von topcased-MM findet man in (Farail u. a., 2006). Schulungs- und Beratungsangebote speziell für TOPCASED sind außerhalb des angegebenen Projektumfelds und außerhalb des akademischen Umfelds schwer auffindbar.

2.5.2 Funktionale Abdeckung

Zur Bewertung der funktionalen Abdeckung werden die funktionalen und nichtfunktionalen Anforderungen der Beispielanwendung mit den Abdeckungsgraden erfüllt, teilweise erfüllt und nicht erfüllt bewertet. Einen Eindruck von der implementierten Anwendung soll Abbildung 2 geben. In dem Diagramm erkennt man deutlich das Zusammenspiel der Zustände. Das Verhalten in den Zuständen muss über den Properties View gepflegt werden. Im Standard enthält dieser jedoch zu viel Informationen, so dass eine Anpassung erforderlich wäre. Bewertet man die Abdeckung mit dem vorgeschlagenen Score, so erhält man 14 von 20 möglichen Punkten.

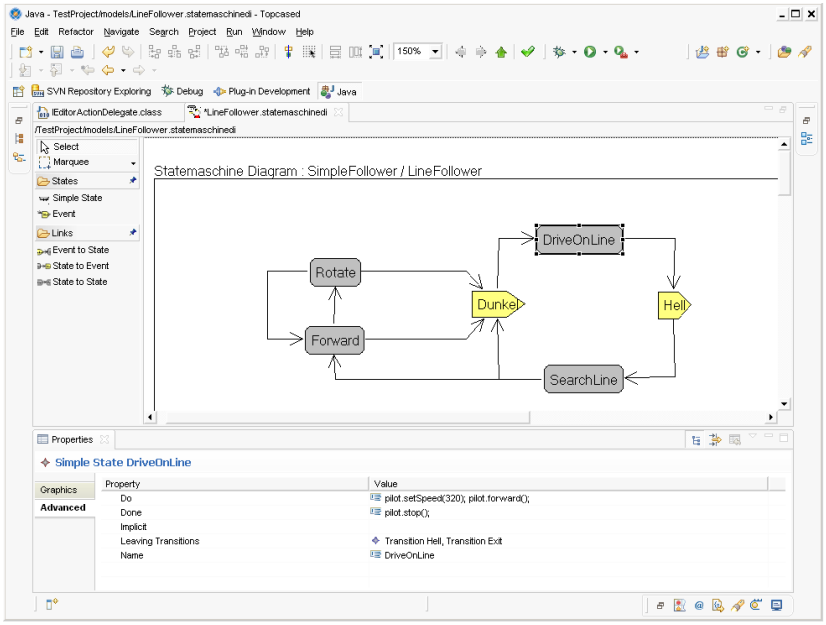

Abbildung 2: TOPCASED Anwendung Line Follower

Tabelle 1: TOPCASED Abdeckung der Anforderungen

Anforderung	Erfüllungs-grad	Begründung
F1	erfüllt	Darstellung der Zustände mit Namen möglich, Name änderbar
F2	erfüllt	Verbindungen zwischen Zuständen und Events darstellbar
F3	erfüllt	Darstellung benannter Events möglich
F4	teilweise erfüllt	Erfassung von Java Code nur im Properties Editor möglich, ansonsten manuelle Programmierung einer angepassten Figure erforderlich.
F5	teilweise erfüllt	Erfassung von Java Code nur im Properties Editor möglich, ansonsten manuelle Programmierung einer angepassten Figure erforderlich.
F6	teilweise erfüllt	Erfassung von Java Code nur im Properties Editor möglich, ansonsten manuelle Programmierung einer angepassten Figure erforderlich. Wegen der Tatsache, dass mehrere Variablen benötigt werden, ist hier besonders viel Code zu schreiben.
NF1	teilweise erfüllt	schon kleine Anforderungen, wie spezielle Figuren erfordern zusätzliche manuelle Erweiterungen
NF2	erfüllt	Aus den Diagrammen konnten vollständige Anwendungen generiert werden.
NF3	teilweise erfüllt	Zur Integration eigener Kontextmenüs für Upload etc. stehen eigene Extension Points zu Verfügung. Es traten keine Probleme auf, die gespeicherten EMF Dateien in oAW zu verarbeiten.
NF4	teilweise erfüllt	Durch den Wechsel zwischen Diagramm und Properties View ist ein Überblick über das erstellte Programm für den Entwickler schwer zu erlangen und noch schwerer mitzuteilen. Positiv ist die Möglichkeit zu bewerten, dass Modelelemente aus dem Diagramm gelöscht werden können, ohne sie aus dem Modell zu löschen. Dadurch wird die Benutzbarkeit deutlich verbessert.

Zu den Punkten F1 und F3 ist hinzuzufügen, dass die vorhandenen Figuren nicht ausreichten. Eine eigene Figur in Form einer Fahne sollte an das UML Symbol für Ereignisse erinnern. Zur Abdeckung dieser Anforderung war die Erzeugung einer eigenen Klasse (vgl. Listing 1) erforderlich, die zusätzlich als Erweiterungspunkt (vgl. Listing 2) definiert werden musste.

2.6 Risiko aus Sicht der TOPCASED Anwender

Die Bewertung erfolgt gemäß QSOS anhand der Kriterienbereiche Investitionsschutz, Softwaremanagement, Technische Anpassbarkeit und Strategie. Für jeden dieser Bereiche legt QSOS Kriterien fest. In jedem Kriterium wird ein Score vergeben. Dabei bedeutet 0=Kriterium nicht erfüllt, 1=Kriterium ausreichend erfüllt, 2=Kriterium gut erfüllt. Bei der Einschätzung des Scores wurden die Vorgaben von QSOS herangezogen.

Auf eine Bewertung der Strategie wurde in diesem Papier bewusst verzichtet. Eine solche Bewertung kann weder den TOPCASED Committern noch den GMF Commitern gerecht werden und bedürfte einer detaillierten Analyse.

In dem Bereich Investitionsschutz soll herausgefunden werden, wie wahrscheinlich eine lange Projektlebenszeit ist. Da modellgetriebene Entwicklung i. A. mit einer Reihe von Werkzeugen betrieben wird, die für jedes neue Projekt aufeinander abgestimmt werden, fällt ein Ersatz eines solchen Produkts nicht besonders schwer.

Aus der Betrachtung des Investitionsschutzes, die in der Tabelle 2 zusammengefasst wird, ergibt sich die Einschätzung eines lebendigen Projekts, dessen Zukunft jedoch nicht wirklich sicher ist. Ein strategisches Projekt mit TOPCASED kann man nur durchführen, wenn man den Support durch den

Hersteller direkt sichert.

Tabelle 2: Bewertung Investitionsschutz TOPCASED

Investitionsschutz			11/26
Reife	Alter	Mit etwa 2,5 Jahren ist das Produkt noch als relativ neu zu bezeichnen.	1
	Stabilität	Das Produkt ist stabil. Es hat einen definierten Releasezyklus, in dem Fehler beseitigt werden. Meist jedoch werden neue Releases zur Umsetzung neuer Features genutzt.	2
	Problem- und Krisenmanagement	Keine bekannten Probleme	1
	Fork Wahrscheinlichkeit	Es ist unwahrscheinlich, dass das Projekt sich in Unterprojekte aufteilt	2
	Unabhängigkeit des Entwicklungsteams	Drei der vier Hauptentwickler arbeiten bei anywhere-tech. Beim vierten war keine Firmenzugehörigkeit ermittelbar.	0
Adoption	Contributing Community	Es gibt eine mailingliste zu dem Projekt.	1
	Popularität	Referenzen im Internet sind schwer auffindbar. Meist aber findet nur der Produktname TOPCASED Erwähnung.	0
	Referenzen	Da es Ziel von Topcased ist, qualitativ hochwertige Anwendungen in der Luft- und Raumfahrtindustie zu entwickeln, ist ein Einsatz in kritischen Anwendungen wahrscheinlich. Auch ist TOPCASED zur Zeit das einzige Open Source UML Modellierungswerkzeug auf Basis von EMF.	1
	Bücher/-Veröffentlichungen	Es gibt zur Zeit keine Bücher und einige wissenschaftliche und populäre Veröffentlichungen. Generell kann man jedoch wegen des universitären Hintergrundes von einer aktiven Veröffentlichungskultur reden.	0
Aktivität	Developers	Das Entwicklungsteam umfasst 4 sehr aktive Entwickler mit mehr als 1000 Commits	1
	Fehlerbeseitigung	Eine Bugliste (bugzilla o. ä.) ist nicht öffentlich zugreifbar. Erst nach Anmeldung und Registrierung erhält man Zugriff.	0
	Neue Features	Eine Möglichkeit neue Features zu beantragen, gibt es zur Zeit nicht. Eine Entwicklungsroadmap ist auf der mailingliste verfügbar.	1
	Releaseaktivitäten	Es sind regelmäßige Releases vorhanden, die sowohl zur Fehlerbeseitigung als auch Weiterentwicklung genutzt werden.	1

Im Bereich Softwaremanagement wird der Industrialisierungsgrad des Softwareentwicklungsprozesses untersucht.

Die Bewertungsergebnisse sind in Tabelle 3 dargestellt. Bei TOPCASED handelt es sich eigentlich um ein internes Projekt, das nach außen hin frei gegeben wird. Da der Entwicklungsfokus eher auf die eigene Verwendung gerichtet ist, kann nicht erwartet werden, dass die Community mit einbezogen wird. Das Ergebnis in diesem Bereich fällt mit 3 von 18 möglichen Punkten deshalb besonders mager aus, weil es aus Sicht der Community und nicht aus Sicht der primären Zielgruppe des Projekts bewertet wurde.

Tabelle 3: Bewertung Softwaremanagement TOPCASED

Softwaremanagement			3/18
Dienstleistungen	Training	Trainings durch die Entwickler werden durchgeführt. Externe Angebote speziell zu dem Thema Topcased MM sind nicht verfügbar.	1
	Support	Nur in geringem Umfang	0
	Consulting	Nur in geringem Umfang	0
Supporting Aktivitäten	Dokumentation	Ist nur in Ansätzen vorhanden. Dabei beziehen sich die verfügbaren Tutorials teils auf ältere Versionen, die man mit Umsicht anpassen muss.	0
	Quality Assurance	Ein Bug Verfolgungstool ist von der Community nicht einsehbar.	0
	Tools	Software wird auf einem GForge Server zur Verfügung gestellt, dessen Funktionalitäten aber nicht systematisch genutzt werden. Das GForge Forum und die mailingliste sind deaktiviert.	0
	Packaging	Ist als Komponente downloadbar oder als Gesamtinstallation. Eine im Eclipse Umfeld übliche Updatesite gibt es nicht.	1
Nutzbarkeit	Benutzerführung	Benutzbarkeit ist intuitiv und selbsterklärend. Trotz der wenigen Dokumentation des Systems ist eine Nutzung möglich.	1
	Administrierbarkeit	Irrelevant	0

Bei der Technischen Anpassbarkeit geht es darum, wie ein Open Source Produkt sich an neue Anforderungen anpassen lässt. In Tabelle 4 spiegelt sich wider, dass Topcased die üblichen Mechanismen nutzt, aber auch nicht darüber hinausgeht.

Tabelle 4: Bewertung Technische Anpassbarkeit TOPCASED

Technische Anpassbarkeit		3/6
Modularität	Mit Topcased MM erstellte graphische Editoren sind modular aufgebaut und gut erweiterbar. Vom Konzept her sind die generierten Editoren weiterzuentwickeln. Die generierte Software stellt nur ein Grundgerüst zu Verfügung, so dass Erweiterungen nicht nur möglich, sondern sogar nötig sind.	1
Code Modification	@generated tags werden mit generiert, so dass standard JMerge Funktionalitäten zu Verfügung stehen.	1
Code Extension	Es gibt zum generierten Code Erweiterungspunkte. Topcased stellt diese im Rahmen der Eclipse Extension/Extension Point Infrastruktur zur Verfügung.	1

2.7 GMF

Im Jahr 2005 fokussierte Borland Software sein Geschäftsfeld auf den Bereich Application Life Cycle Management und wurde Mitglied der Eclipse Foundation. Als erster großer Beitrag wurde das Projekt GMF unter Leitung von Richard Gronback ins Leben gerufen. GMF sollte die beiden Eclipse Projekte GEF und EMF standardisiert miteinander verbinden und damit viele Individualentwicklungen, wie sie zum Beispiel in (Moore u. a., 2004) beschrieben sind, standardisieren und damit stark vereinfachen oder gar überflüssig machen.

GMF ist Bestandteil des Eclipse Modeling Projects. Das eingesetzte Release ist GMF 2.0.1. vom

29.9.2007. Zu Zeit ist das Release 2.1 in Vorbereitung. Die Projektseite ist `http://www.eclipse.org/gmf`. GMF verfügt über zahlreiche Tutorials und FAQs. Ein Buch über GMF ist nicht verfügbar, aber in Vorbereitung und soll im Frühsommer 2008 erscheinen. Schulungsangebote sind von zahlreichen Anbietern verfügbar.

2.7.1 Funktionale Abdeckung

Die Bewertung der Funktionalen Abdeckung erfolgt genauso wie bei TOPCASED. Zur Veranschaulichung der funktionalen Abdeckung dient die Abbildung 3 . Die Abbildung zeigt die vollständige Anwendung. Fast alle Anforderungen sind im Standard von GMF gut abgedeckt, was sich in 19 von 20 Scorepunkten niederschlägt. Durch die Compartments für Zustandsvariablen gewinnt die Anwendung an Übersichtlichkeit. Durch die Zuordnung von Zustandsvariablen zur Statemachine oder zum State können unterschiedliche Sichtbarkeiten realisiert werden.

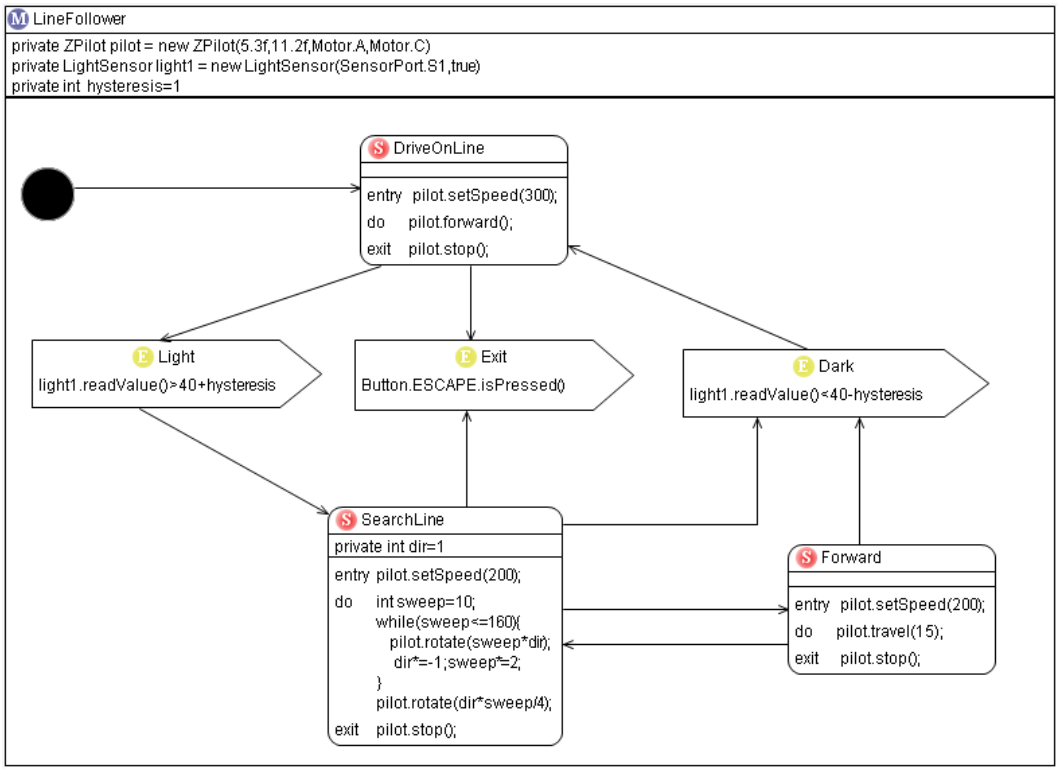

Abbildung 3: GMF Anwendung Line Follower

Es ist zu bemerken, dass für die Verwendung von mehrzeiligem Java Code auch eine Modifikation im generierten Code erforderlich ist.

2.8 Risiko aus Sicht der GMF Anwender

Die Bewertung erfolgt gemäß QSOS nach demselben Schema wie bei TOPCASED.

Aus der Betrachtung des Investitionsschutz, die in Tabelle 6 zusammengefasst wird, ergibt sich die Einschätzung eines lebendigen Projekts mit sicherer Zukunft. GMF ist für den betrieblichen Einsatz uneingeschränkt geeignet.

Tabelle 5: GMF Abdeckung der Anforderungen

Anforderung	Erfüllungs-grad	Begründung
F1	erfüllt	Darstellung der Zustände mit Namen möglich, Name änderbar
F2	erfüllt	Verbindungen zwischen Zuständen und Events darstellbar
F3	erfüllt	Darstellung benannter Events möglich. Darüber hinaus wurden auch Events ohne konkreten Knoten nur als Verbindung dargestellt.
F4	erfüllt	Erfassung von Java Code ist direkt im Diagramm möglich. Allerdings fehlt, wie sonst üblich, die kontextsensitive Hilfe des Java Editors.
F5	erfüllt	Erfassung von Java Code direkt im Diagramm möglich. Allerdings fehlt, wie sonst üblich, die kontextsensitive Hilfe des Java Editors.
F6	erfüllt	Compartments können genutzt werden, um abhängige Objekte mit vielfachen Kardinalitäten darzustellen.
NF1	erfüllt	GMF ist mächtig genug, um benutzerfreundliche und vollständige Anwendungen im Standard zu erzeugen.
NF2	erfüllt	Aus den Diagrammen konnten vollständige Anwendungen generiert werden.
NF3	erfüllt	Zur Integration eigener Kontextmenüs für Upload etc. stehen eigene Extension Points zu Verfügung. Es traten keine Probleme auf, die gespeicherten EMF Dateien in oAW zu verarbeiten.
NF4	teilweise erfüllt	Der Aufwand, ergonomisch gute Anwendungen mit GMF zu erzeugen, ist nicht zu unterschätzen. Die verwendeten Layout Manager sind schwer zu durchschauen, und der Entwickler braucht tiefes Verständnis, um ein gewünschtes Verhalten zu erreichen. Leider ist GMF unter MAC OS X in der ausgelieferten Version nicht stabil. Das Setzen von Eigenschaften im Properties Editor kann zu Abbrüchen führen. Die Darstellung von WrapLabels ist unter MAC OSX fehlerhaft.

Tabelle 6: Bewertung Investitionsschutz GMF

Investitionsschutz			19/26
Reife	Alter	Mit knapp 3 Jahren ist das Produkt noch als relativ neu zu bezeichnen.	1
	Stabilität	Das Produkt ist stabil. Es hat einen definerten Releasezyklus, in dem Fehler beseitigt werden. Meilensteine mit definierten Zielen werden öffentlich bekannt gegeben.	2
	Problem-Krisenmanagement	Keine bekannten Probleme	1
	Fork Wahrscheinlichkeit	Es ist unwahrscheinlich, dass das Projekt sich in Unterprojekte aufteilt	2
	Unabhängigkeit des Entwicklungsteams	Hauptentwickler arbeiten beim Mitglied der Eclipse Foundation Borland und bei IBM	1

Investitionsschutz			Forts.
Adoption	Contributing Community	Es gibt eine mailingliste zu dem Projekt.	1
	Popularität	Referenzen im Internet sind vielfältig vorhanden.	2
	Referenzen	Es gibt kommerzielle Projekte, wie zum Beispiel Gentlewares UML Modellierungstool Appollo, die auf GMF aufsetzen.	1
	Bücher/-Veröffentlichungen	Es gibt zur Zeit keine Bücher und einige wissenschaftliche und viele populäre Veröffentlichungen.	1
Aktivität	Developers	Das Entwicklungsteam umfasst 14 sehr aktive Entwickler mit mehr als 1000 Commits. 5 von IBM und 7 von Borland und einen ohne bekannte Firmenzugehörigkeit.	2
	Fehlerbeseitigung	Eine Bugliste ist öffentlich zugreifbar und recherchierbar. Bugs und werden zeitlich nahe bearbeitet.	2
	Neue Features	Eine Möglichkeit, neue Features zu beantragen. Ansonsten folgt GMF einem definierten Projektplan, der dem Eclipse Modeling Projektplan angegliedert ist.	2
	Release Aktivitäten	Es sind regelmäßige Releases vorhanden, die sowohl zur Fehlerbeseitigung als auch Weiterentwicklung genutzt werden.	1

Tabelle 7: Bewertung Softwaremanagement GMF

Softwaremanagement			10/18
Dienstleistungen	Training	Trainingsangebote zum Thema GMF sind von verschiedenen Anbietern verfügbar. Spezialisten tragen auf allen großen Java Konferenzen die neuesten Entwicklungen vor.	2
	Support	Nur in geringem Umfang	0
	Consulting	Von freien Anbietern verfügbar.	2
Supporting Aktivitäten	Dokumentation	Eine einheitliche Dokumentation ist nur in Ansätzen vorhanden, denn die Eclipse GMF Dokumentation eher eine Sammlung von Tutorials und Spezialartikeln, als eine strukturierte Dokumentation. Allerdings exisitieren viele Tutorials, FAQs und newsgroups, in denen vielfältige Codebeispiele zu finden sind.	1
	Quality Assurance	Ein Bug Verfolgungstool auf eclipse.org ist nutzbar.	1
	Tools	Auf der EMP Homepage steht eine breite Palette an Werkzeugen zur Verfügung, so dass ein Anwender von GMF sich gut informieren kann.	2
	Packaging	Softwareinstallation geschieht über eine Eclipse update Site und ist problemlos. Abhängigkeiten werden (fast immer) automatisch erkannt.	1
Nutzbarkeit	Benutzerführung	Benutzbarkeit ist intuitiv und selbsterklärend. Aufgrund der großen Komplexität und des großen Umfang des GMF Projekts ist der Einstieg für den Anfänger schwierig.	1
	Administrierbarkeit	Irrelevant	0

Im Bereich Softwaremanagement erreicht GMF den Standard, der in der Eclipse Entwicklung üblich ist. Er kann als mustergültig im IT Bereich angesehen werden.

Die Bewertungsergebnisse sind in Tabelle 7 dargestellt.

In Tabelle 8 spiegelt sich wieder, dass GMF die üblichen Entwurfsmuster und Eclipse Mechanismen nutzt.

Tabelle 8: Bewertung Technische Anpassbarkeit GMF

Technische Anpassbarkeit		6/6
Modularität	Die GMF generierten Module haben eine vorbildliche modulare Struktur. Das erschwert zwar den Einstieg, ermöglicht jedoch gute Erweiterbarkeit.	2
Code Modification	@generated tags werden mit generiert, so dass Standardisierte JMerge Funktionalitäten zu Verfügung stehen. Darüberhinaus besteht die Möglichkeit GMF als Runtimekomponente zu verwenden, ohne den Generator zu benutzen.	2
Code Extension	Es gibt zum generierten Code Erweiterungspunkte. GMF stellt diese im Rahmen der Eclipse Extension/Extension Point Infrastruktur zur Verfügung. Diese Extension Points sind gut dokumentiert. Die Integration in andere Frameworks, wie zum Beispiel das Validation Framework von EMF ist gegeben.	2

3 Zusammenfassung

Eine Übersicht über die Ergebnisse der Bewertung gibt Abbildung 4.Da GMF in allen Kategorien besser abschneidet als TOPCASED, ist eine Gewichtung der Bereiche nicht erforderlich.

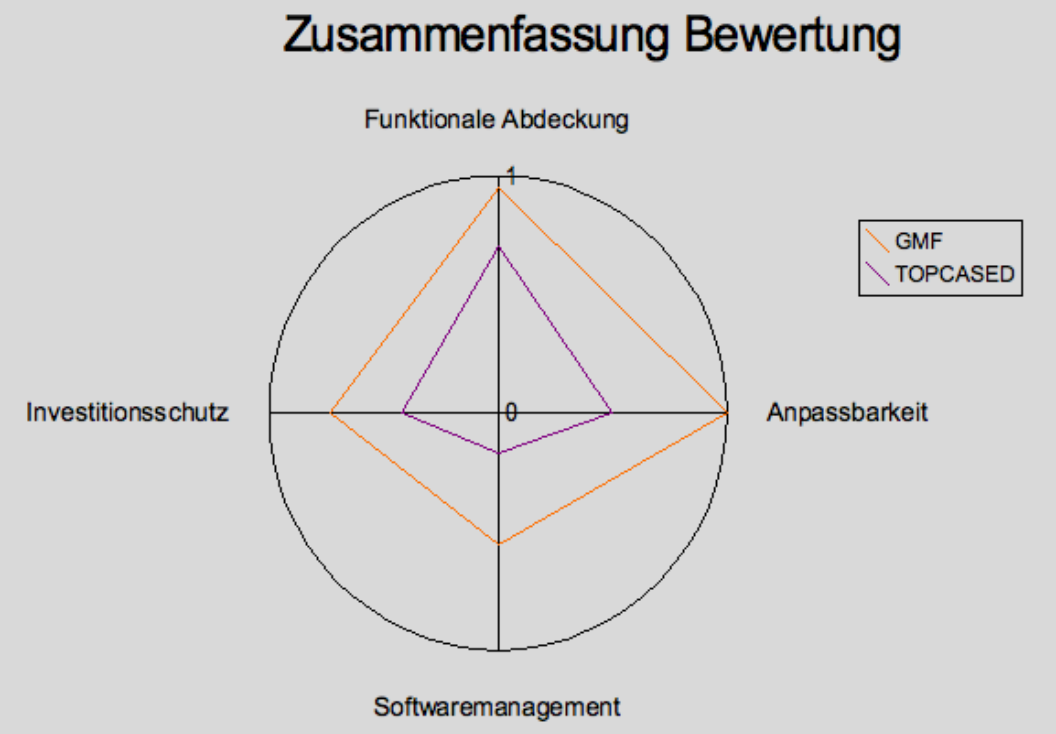

Abbildung 4: Zusammenfassung Vergleich TOPCASED/GMF

Die Funktionale Abdeckung und die Evaluation der Risiken zeigen deutlich, dass ein Einsatz von TOPCASED-MM zu Zeit nur dann in Betracht kommt, wenn die nahtlose Integration in das TOPCASED Gesamtprojekt gewünscht ist. Ansonsten ist GMF das ausgereiftere, zukunftssicherere und mächtigere Werkzeug. Die Funktionalität von TOPCASED steht der von GMF in einigen Punkten deutlich nach.

1. Angepasste Formen sind in TOPCASED nur durch Java Erweiterungen zu realisieren.

2. Unterteilung eines Objekts mit der Darstellung von Eigenschaften und Compartments muss selbst programmiert werden.

Aufgrund der geringeren Funktionalität ist die Beschreibung von graphischen Anwendungen mit TOPCASED einfacher. Beide Frameworks ermöglichen die nachträgliche Änderung des fachlichen Metamodells. Anschließendes Ändern der graphischen Modelle erweißt sich nicht als Problem, sondern kann in kurzer Zeit angepasst werden. In dem Beispielprojekt wurde eine Version mit GMF veröffentlicht.

3.1 Ausblick

Die Methode QSOS ermöglicht den Vergleich von Open Source Produkten für den betrieblichen Einsatz. Insbesondere die Bewertung der vier Bereiche Investitionsschutz, Softwaremanagement, Anpassbarkeit und Strategie sind für einen betrieblichen Einsatz einer Open Source Software aus Gründen des Investitionsschutzes unverzichtbar. Hier entscheidet sich, ob ein Open Source Projekt für den betrieblichen Einsatz geeignet ist. Insbesondere könnte man eine solche Methode sehr gut in Workshops einsetzen, um bei der Bewertung der Kriterien durch Anwendung der Delphi Methode ein präziseres Bild zu bekommen.

A Anhang

Listing 1: TOPCASED Java Code für Flag Figure

```
1  package de.nordakademie.draw2d.figures.impl;
2
3  import org.eclipse.draw2d.Graphics;
4  import org.eclipse.draw2d.geometry.Rectangle;
5
6  public class LabeledFlag extends org.topcased.draw2d.figures.BorderedLabel {
7          private int[] innerPoints = new int[10];
8
9          public LabeledFlag() {
10                 super();
11         }
12
13         protected void fillShape(Graphics graphics) {
14                 graphics.fillPolygon(innerPoints);
15         }
16
17         protected void outlineShape(Graphics graphics) {
18                 Rectangle r = getBounds();
19                 innerPoints[0]=r.x+lineWidth;
20                 innerPoints[1]=r.y+lineWidth;
21                 innerPoints[2]=r.x + 2*r.width/3-lineWidth;
22                 innerPoints[3]=r.y+lineWidth;
23                 innerPoints[4]=r.x + r.width-lineWidth;
24                 innerPoints[5]=r.y + r.height/2+outerPoints[5];
25                 innerPoints[6]=r.x + 2*r.width/3-lineWidth;
26                 innerPoints[7]=r.y + r.height-lineWidth;
27                 innerPoints[8]=r.x;
28                 innerPoints[9]=r.y + r.height-lineWidth;
29                 graphics.drawPolygon(innerPoints);
30         }
31 }
```

Listing 2: Extension Point figureDeclarations

```
1 <extension point="org.topcased.modeler.diagramconfigurator.figureDeclarations">
2       <figureDeclaration
3             class="de.nordakademie.draw2d.figures.impl.LabeledFlag"
4             name="LabeledFlag"/>
5 </extension>
```

Literatur

[van den Berg 2005] Berg, Karin van den: *Finding open options*. Tilburg, Netherlands, Tilburg University, Diplomarbeit, August 2005

[Bettin 2004] Bettin, Jorn: *Model Driven Software Development*. Internet: http://www.softmetaware.com/mdsd-and-isad.pdf. June 2004

[Bézivin 2005] Bézivin, Jean: On the Unification Power of Models. In: *Software and System Modeling (SoSym)* 4 (2005), Nr. 2, S. 171–188. – URL `http://www.sciences.univ-nantes.fr/lina/atl/www/papers/OnTheUnificationPowerOfModels.pdf`

[Carnegie Mellon West Center for Open Source Investigation u. a. 2005] Carnegie Mellon West Center for Open Source Investigation ; O'Reilly CodeZoo ; SpikeSource ; Intel: *Business Readiness Rating*. http://www.openbrr.org. August 2005

[Farail u. a. 2006] Farail, Patrick ; Gaufillet, Pierre ; Canals, Agusti ; Camus, Christophe L. ; Sciamma, David ; Michel, Pierre ; Crégut, Xavier ; Pantel;, Marc: The TOPCASED project: a Toolkit in Open source for Critical Aeronautic SystEms Design / AIRBUS FRANCE, ZAC de la Grande Plaine, Anyware Technologies, FéRIA-ONERA, FéRIA-IRIT-ENSEEIHT. ERTS Toulouse, January 2006. – Forschungsbericht

[Fong 2006] Fong, Choong K.: *Quick Start Guide to MDA; A Primer to Model-Driven Architecture Using Borland Together Technologies*. `http://www.borland.com/resources/en/pdf/products/together/together_mda_quickstart.pdf`. September 2006

[Golden 2004] Golden, Bernard: *Navica Open Source Maturity Model*. http://www.navicasoft.com/pages/about.htm. Oktober 2004

[Greenfield u. a. 2004] Greenfield, Jack ; Short, Keith ; Cook, Steve ; Kent, Stuart: *Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools*. Wiley, August 2004. – URL `http://www.amazon.de/exec/obidos/redirect?tag=citeulike01-21\&path=ASIN/0471202843`. – ISBN 0471202843

[Gronback 2008] Gronback, Richard: *GMF Webpage*. Internet `http://www.eclipse.org/gmf/`. 2008

[Lescot 2008] Lescot, Jaques: *TopCased Web Page*. Internet http://www.topcased.org/. Feb 2008

[metacase.com 2007] metacase.com: *MetaEdit+ Workbench*. 2007. – URL `http://www.metacase.com/mwb/`

[Miller und Mukerji 2003] Miller, J. ; Mukerji, J.: MDA Guide Version 1.0.1 / Object Management Group (OMG). 2003. – Forschungsbericht

[Moore u. a. 2004] Moore, Bill ; Dean, David ; Gerber, Anna ; Wagenknecht, Gunnar ; Vanderheyden, Philippe: *Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework*. IBM ITSO, February 2004 (IBM Redbooks)

[Pontisso und Chemouil 2006] Pontisso, Nadege ; Chemouil, David: TOPCASED Combining Formal Methods with Model-Driven Engineering. In: Society, IEEE C. (Hrsg.): *ASE '06: Proceedings of the 21st IEEE/ACM International Conference on Automated Software Engineering*. Washington, DC, USA : IEEE Computer Society, 2006, S. 359–360. – ISBN 0-7695-2579-2

[Semeteys 2006] Semeteys: *Method for Qualification and Selection of Open Source Software.* http://www.qsos.org/download/qsos-1.6-en.pdf. April 2006

[Slapeta 2006] Slapeta, Stefan: *Ein Vergleich zwischen Visual Studio 2005 und Eclipse Graphical Modelling Framework zur Unterstützung modellgetriebener Softwareentwicklung.* Business Informatics Group der Technischen Universität Wien, TU Wien, Diplomarbeit, March 2006

[Stahl u. a. 2006] Stahl, Thomas ; Voelter, Markus ; Czarnecki, Krzysztof: *Model-Driven Software Development: Technology, Engineering, Management.* John Wiley & Sons, 2006. – ISBN 0470025700

[Swithinbank u. a. 2005] Swithinbank, Peter ; Chessell, Mandy ; Gardner, Tracy ; Griffin, Catherine ; Man, Jessica ; Wylie, Helen ; Yusuf, Larry ; IBM (Hrsg.): *Patterns: Model-Driven Development Using IBM Rational Software Architect.* December 2005 (ITSO Red Book sg247105)

[Tolvanen 2006] Tolvanen, Juha-Pekka: MetaEdit+: integrated modeling and metamodeling environment for domain-specific languages. In: Tarr, Peri L. (Hrsg.) ; Cook, William R. (Hrsg.): *OOPSLA Companion*, ACM, 2006, S. 690–691. – ISBN 1-59593-491-X