Universitatea Babeş-Bolyai
Facultatea de Matematica si Informatica

Generarea codului bazata pe modele

- Rezumat teza de doctorat -

Coordonator stiintific:
Prof. Dr. Bazil Pârv

Autor:
Paul Horaţiu Stan

Februarie 2012

Cuprins

Lista publicatiilor

Articole publicate

[107] Paul Horaţiu Stan. A proposed DSL for data intensive application development. *Studia UBB, Informatica*, LVII(1):accepted, 2012.

[103] Paul Horaţiu Stan. Building blocks dev studio. a tool for component based development. *Studia UBB, Informatica*, LVI(3):9–14, 2011.

[108] Paul Horaţiu Stan and Camelia Şerban. A proposed approach for platform interoperability. *Studia UBB, Informatica*, LV(2):87–98, 2010.

Participari la conferinte

[104] Paul Horaţiu Stan. Building blocks dev studio. a tool for component based development. In *Proc KEPT 2011 (eds: M. Frentiu, HF Pop, S. Motogna)*, pages 247–258, ISSN 2067-1180, Cluj University Press, 2011 (ISI Proc).

[102] Paul Horaţiu Stan. A proposed technique for component-based software development. In *Zilele Academice Clujene*, pages 52–56, 2010.

In curs de publicare

[105] Paul Horaţiu Stan. Case study: An order processing system developed using DevDSL. Submitted to: *16th East European Conference on Advances in Databases and Information Systems*, Poznan, September 18-21, 2012.

[106] Paul Horaţiu Stan. A custom validation mechanism for DevDSL. Submitted to: *Studia UBB, Informatica*, LVII(1):, 2012.

Lista cuvintelor cheie

Model Driven Engineering, Model Driven Architecture, Domain Specific Modeling, Domain Specific Languages, Automatic Code Generation, Component-based development, Platform Independent Model, Platform Specific Model, Model Transformations, General Programming Languages.

Rezumat

Aceasta teza de doctorat este rezultatul cercetarii mele in domeniul Model Driven Engineering (MDE), in special in Model Based Code Generation (MBCG), cercetare ce a inceput in 2008 sub indrumarea domnului profesor Dr. Bazil Pârv.

Motivate

O intrebare simpla este: *De ce titlul aceste teze este Model Based Code Generation sau Generarea codului bazata pe modele?* Raspunsurile ar fi:

- *MBCG* poate fi vazut ca o ramura a MDE orientat pe transformarea modelelor. Metamodelul sursa reprezinta limbajul de modelare folosit la specifiarea modelului independent de platforma (PIM) a unui sistem, in timp ce metamodelul tinta este reprezentat de sintaxa limbajului de programare pentru care se doreste generarea de cod sursa, acesta este de fapt modelul specific platformei (PSM).

- *MBCG* poate imbunatatii procesul de dezvoltare soft deoarece propune automatizarea procesului de generare a codului sursa.

- *MBCG* poate creste calitatea softului pentru ca se elimina erorile de implementare.

- *MBCG* reprezinta un domeniu deschis cercetarilor viitoare, existand numeroase articole, carti si conferinte pe aceasta tema.

- *MBCG* este pasul urmator in instoria limbajelor de programare si a compilatoarelor. La inceputuri programatorii trebuiau sa scrie programele in limbaj masina, apoi a aparut limbajul de asamblare, care practic, asocia un cuvant sugestiv unei instructiuni masina, pasul al treilea a fost introducerea limbajelor de programare ca C++, Pascal etc. Programele scrise in aceste limbaje erau compilate, validate, iar apoi se gerena cod executabil. MBCG poate fi vazut ca urmatorul pas normal pe aceasta evolutie naturala. Practic, MBCG propune specificarea unei aplicatii soft intr-un limbaj specific domeniului problemei (DSL) iar apoi utilizand transformari automate se va genera codul sursa al aplicatiei intr-un anumit limbaj de programare.

- Folisind *MBCG* se reduce distanta dintre persoanele non-tehnice care doresc o aplicatie soft, si dezvoltatorii aplicatiei. Practic, se dezvolta un nou limbaj orientat

cerintelor clientului iar apoi folosind acest limbaj se obtin cerintele clientilor intr-o forma clara si lipsita de ambiguitati.

Obiectivele cercetarii

In contextul problemelor actuale de cercetare in domeniul MDE, obiectivele cercetarii acestei teze au fost orientate pe generarea automata a codului sursa conform modelului aplicatiei. Urmatoarea enumerare prezinta principalele obiective:

- Propunerea unei arhitecturi pentru reutilizarea modelului dependent de platforma a unei aplicatii dintr-un alt model dependent de platforma. Abordarea propusa foloseste un obiect surogat generat in mod automat [108] care actioneaza ca un mediator intre noul model si cel vechi. Fiecare obiect surogat deleaga apelurile metodelor catre vechiul model dependent de platforma.

- Propunerea proces de dezvoltare soft bazat pe componente care sa automatizeze generarea de cod sursa in doua moduri: (1) generarea codului sursa pentru structura de baza a componentelor, si (2) generarea codului sursa pentru conectarea componentelor [102, 103, 104].

- Propunerea unui limbaj specific domeniului (DSL) pentru applicatii care proceseaza date in mod intensiv si care sa permita translatarea automata spre diferite modele dependente de platforma (PSM) web sau non-web [107, 106, 105].

Fiecare obiectiv este descris in detaliu intr-un capitol dedicat, urman acelasi sablo, se incepe cu prezentarea problemei deschise, pasul al doilea prezinta solutia propusa, un studiu de caz sau o lista de studii de caz sunt prezentate in sectiunea a treia, iar la finalul fiecarui capitol, sunt sumarizate concluziile si imbunatatirile ulterioare impreuna.

Capitolul 2 prezinta o trecere in revista Model Driven Architecture (MDA) [4, 116, 8, 87, 59, 114, 90, 22, 12, 91, 27, 33, 70]. Obiectivele MDA sunt [116]: invechirea tehnologiei, portabilitatea, productivitatea, calitatea, integrarea, întretinerea, testarea si simularea si nu in ultimul rand randamentul investitiilor. Acest capitol prezinta sumar principalele concepte MDA [4] precum: Sistemul, Modelul, Orientarea pe model, Arhitectura, Unghiul de vedere, Unghiuri de vedere MDA, Platforma, Independenta de platforma, Modelul Platforma, Transformarea modelului, Implementarea, Modeul independent de calcul, Modeul independent de platforma [26], modelul specific de platforma. Consortiul OMG a propus o specificare generala, astfel ca raspuns, instrumente MDA au fost dezvoltate de echipe de cercetare. Exista numeroase tipuri de instrumente MDA precum [116]: Instrumente de creare, de analiza, de transformare, de compunere, de testare, de simulare, de gestiune a metamodelelor precum si de "Reverse Engineering".

Pasii principali in abordarea MDA sunt definirea modelului independent de plaforma [26] iar apoi transformarea lui automata in un sau mai multe modele dependente de

platforma [28]. Transformarea modelelor consta in doua componente: (1) transformarea model spre text si (2) transformarea text spre model. La finalul acestui capitol este prezentata o lista de standarde folosite de catre abordarea MDA precum: XML, XMI, UML, MOF, MDDA, SCA, BPMN [8, 38, 23, 97, 125].

Capitolul 3 prezinta un set de instrumente si cadre de dezvoltare MDA folosite in cercetarea actuala. Capitolul 3 este structurat in 2 sectiuni: (1) instrumente academice, si (2) instrumente industriale. Sectiunea *Instrumentele academice* prezinta aplicatii dezvoltate de echipe de cercetare in timp ce sectiunea *Instrumente industriale* prezinta aplicatii dezvoltate de organizatii sau firme soft.

Contributiile personale sunt prezentate in capitolele 4, 5 si 6.

- Capitolul 4 *"O abordare pentru interoperabilitatea intre platforme bazata pe obiecte surogat"*: prezinta o tehnica pentru interoperabilitatea bazata pe transformarea modelelor dintre platforme de dezvoltare soft. Abordarea introdusa in acest capitol evita serializarea obiectelor pentru comunicarea intre platforma, si propune generarea automata a unui obiect surogat pentru fiecare obiect la distanta. Codul sursa scris pentru implementarea obiectului la distanta, reprezinta modelul sursa al transformarii si este conform metamodelului sursa (sintaxa limbajul de programare sursa), in timp ce obiectul surogat generat reprezinta modelul destinatie ce este conform metamodelului destinatie (sintaxa limbajului de programare destinatie). In concluzie, tehnica prezentata in acest capitol poate fi considerata contributie originala in domeniul transformatrii modelelor. Solutia originala este descrisa in detaliu in [108].

- Capitolul 5 *"O solutie pentru dezvoltarea soft bazata pe componente folosing generarea automata a codului de conectare a componentelor"* propune o tehnica pentru dezvoltarea soft bazata pe componente. Noutatea solutiei propuse consta in faptul ca fiecare componenta are proprietati publice ce pot avea una din urmatoarele directii: IN, OUT si INOUT, si comunicarea intre componente se realizeaza pe baza acestor proprietati, numite porturi. Ca rezultat al acestei tehnicii propuse, componentele nu sunt dependente unele de altele, fiind dependente doar de tipul de data al porturilor lor. Contributiile originale sunt prezentate in detaliu in urmatoarele lucrari: [104, 102, 103].

- Capitolul 6: *"Un limbaj specific domeniului pentru dezvoltarea de aplicatii ce prelucreaza date"* prezinta o solutie ce presupune: (1) un limbaj pentru definirea modelului independent de platforma a unei aplicatii si (2) un modul de transformare automat care poate transforma modelul independent de platforma in aplicatii web .NET. Ambele pachete sunt incluse intr-un plug-in Eclipse care ofera functionalitati de completare automata a codului si evidentierea cuvintelor rezervate. Contributiile originale din acest capitol sunt descrise in detaliu in urmatoarele lucrari:

[107, 105, 106].

Capitolul 4 prezinta o noua abordare relativ la interoperabilitate. Noutatea solutie propuse consta in folosirea unui obiect surogat pentru fiecare obiect la distanta pentru evitarea serializarii.

O problema actuala in domeniul ingineriei soft este imbunatatirea procesului de dezvoltare soft precum si a calitatii produsului final [43]. Pentru a realiza acest fapt s-a incercat reutilizarea codului sursa [73]. Evident, este o veste buna pentru un programator daca o librarie de rutine dezvoltata intr-un limbaj de programare vechi poate fi utilizata intr-un limbaj de programare nou fara a fi nevoie sa fie rescris codul sursa. In acest caz, se salveaza timpul de re-implementare si se imbunatateste calitatea sistemului soft final pentru ca libraria veche a fost testata si e functionala.

In contextul transformarii modelelor, un obiectiv al acestei teze de doctorat a fost determinarea unui algoritm pentru generarea codului sursa pentru un anumit libaj de programare pe baza unui cod sursa scris in alt limbaj de programare intr-un mod automat. Acest process nu este o translatare a codului sursa dintr-un limbaj de programare in altul, generat actioneaza ca mediator spre codul real, deci apelurile metodelor sunt delegate catre rutinele reale. Aceasta cercetare poate fi inclusa in domeniu "Transformarii Modelelor" deoarece transforma un model executabil, rezultat al compilarii unui cod sursa scris pentru un limbaj de programare intr-un cod sursa sctis pentru alt limbaj de programare care actioneaza ca un mediator spre modelul executabil initial.

Exista numeroase cadre de dezvoltare atat pentru Java and .NET, de exemplu: Hibernate [25] respectiv NHibernate, JUnit [80] respectiv NUnit etc. Aceste cadre de dezvoltare puteau fi scrise intr-un limbaj de programare si refolosite in alt lumbaj de programare, de exemplu, in loc sa fie rescris Hibernate in bersiunea NHibernate pentru .NET se putea reutiliza direct cu solutia propusa. Capitolul 5 foloseste conceptele introduse in capitolul 4 la implementarea unei aplicatii soft. Aplicatia gestioneaza entitati dintr-o biblioteca precum: carti, autori si membri. Modelul conceptual al aplicatiei a fost scris in .NET, dar componenta care gestioneaza baza de date a fost scrisa in Java, asadar, a fost necesar generarea de obiecte surogat scrise in Java care sa acceseze obiectele reale din .NET. Frumusetea solutiei reiese din posibilitatea salvarii si incarcarii obiectelor .NET folosind versiunea Java a cadrului de dezvoltare Hibernate si obiecte surogat la obiectele reala .NET. Nu este necesara folosirea cadrului de dezvoltare NHibernate.

Capitolul 4 este structurad dupa cum urmeaza: prima sectiune prezinta descrierea problemei impreuna cu: (1) motivatia pentru care se doreste rezolvarea ei, (2) abordarile actuale si (3) limitele lor, sectiunea a doua prezinta solutia propusa, iar implementarea practica a notiunilor teoretice este prezentata in sectiunea trei. In sectiunea 4 se aplica doua metrici de evaluare a solutiei de generarea codului propuse. La final sunt prezentate contributiile originale precum si imbunatatirile ulterioare.

Capitolul 5 prezinta o tehnica inspirata din industria dezvoltarii hardware care poate fi aplicata si in industria soft pentru a dezvolta sisteme flexibile si modulare bazat pe componente independente. Pe de alta parte, se prezinta avantajele si constrangerile acestei tehnici si cum poate fi implementata pentru o platforma de dezvoltare actuala precum Java sau .NET.

In procesul de dezvoltare hardware, un inginer poate folosi mai multe circuite integrate pentru a realiza un sistem complex, de exemplu el/ea poate folosi multiplexoare, numaratoare, porti logice precum SI, SAU, memorii, registre [76, 54, 85, 61, 52, 71, 50, 129, 93] etc, aceste componente sunt interconectate pe o placa de baza pentru a forma un sistem complex. Numaratorul, multiplexorul sau alte componente sunt realizate de terte companii, si nu sunt dependente de alte componente de care sunt conectate. Acest principiu poate fi aplicat atat in procesul de dezvoltare hard cat si in procesul de dezvoltare soft, de exemplu sistemele soft pot avea componente independente care pot fi interonecnate, si incluse intr-o componente parinte pentru a forma un sistem complex. Componentele pot fi dezvoltate de terte companii. Momentan exista numeroase sisteme soft care folosesc componente dar conexiunile dintre ele au fost facute de programatori, si faptul ca o resursa umana scrie cod poate introduce erori de implementare, mai mult, implementarea codului de catre un programator reprezinta un proces destul de costisitor, asadar trebuie alocare resurse de timp si buget pentru acest proces.

Aceasta abordare prezinta o tehnica care permite generatea automata a codului de conectare a diferitelor obiecte. Aceasta tehnica inpute constrangeri asupra obiectelor pentru a face posibil procesul de conectare automat. Acest capitol combina conceptele teoretice cu notiunile prezentate in capitolul 4 pentru a exemplifica beneficiile solutiei propuse. Avantajul major al acestei solutii consta in: o tehnica de proiectare a componentelor soft care permite scrierea de componente independente care vor fi conectate automat folosind un generator de cod sursa. Procesul de dezvoltare contine trei pasi: (1) proiectarea sistemului soft intr-un mediu grafic, (2) generarea codului sursa minimal, si (3) implementarea functionalitatilor componentelor.

Capitolul 5 este structurat dupa cum urmeaza: prima sectiune descrie problema de cercetare in trei subsectiuni: (1) motivatia, (2) abordari actuale si (3) limitele lor; sectiunea 2 expune solutia propusa in patru subsectiuni: (1) viziunea conceptuala, (2) glosar de termeni, (3) arhitectura si (4) cum functiuneaza solutia; sectiunea 3 prezinta aplicatia *Building Blocks Dev Studio* in patru subsectiuni: (1) arhitectura aplicatiei, (2) mecanismul de comunicare intre componente, (3) un proces de dezvoltare orientat pe componente rezultat din solutia propusa, (4) un studiu de caz care contine o aplicatie demonstrativa dezvoltata cu instrumentele *Building Blocks Dev Studio* si *Indep* [103, 104]; a patra sectiune prezinta doua metrici de evaluare a solutiei propuse relativ la procentul de cod sursa generat automat, iar in final, sectiunea 5 trece in revista contributiile originale din acest capitol precum si imbunatatirile ulterioare.

Capitolul 6 prezinta cercetarea efectuata in timpul mobilitatii de trei luni de la Univesritatea din Debrecen din Octombrie pana in Decembrie 2010 [107, 105, 106]. Impreuna cu domnul profesor Adamko Attila, am dezvoltat un limbaj specific domeniului (DSL) pentru aplicatii care prelucreaza date in mod intensiv. Cercetarea s-a finalizat cu un plug-in Eclipse care poate fi folosit la specificarea modelului unei aplicatii si transformarea modelului independent intr-o aplicatie web .NET. Beneficiile DSL propus rezulta din posibilitatea implementarii de module de transformare spre alte modele specifice platformei, nu numai spre web. La final este prezentata o comparatie cu WebML [109, 86, 123, 98, 79, 17, 21, 133] si WebDSL [29, 56, 48, 49, 47].

Aceasta cercetare prezinta un DSL pentru specificarea aplicatiilor ce prelucreaza date in mod intensiv. Folosind solutia propusa se poate defini un model independent de platforma (PIM) a unei aplicatii, apoi folosind transformari spre diferite modele specifice platformei (PSM) codul sursa poate fi generat automat. Limbajul a fost implementat folosind Eclipse Xtext, XPand si MWE2, si utilizand documentatiile din [40, 81, 64, 74].

Tremenii PIM si PSM [26] sunt frecvent folositi in contextul abordarii MDA. Conceptul de baza consta in posibilitateafolosirii unui limbaj de transformare a modelelor (MTL) [35, 69, 82, 131] pentru a transforma un model independent de platforma intr-un model specific platformei [112, 31, 115, 82, 131].

Acest capitol propune doua module de transformare: (1) un modul de transformare din PIM in aplicatii web .NET si (2) un modul de transformare din PIM in aplicatii PHP Code Igniter. Aceste module sunt demonstratii practice ale conceptelor prezentate in capitolul 6, evidentiind posibilitatea specificarii modelelor independente de platforma folosind limbajul propus, iar apoi transformarea lor in diferite modele specifice platformei.

Capitolul 6 este structurat dupa cum urmeaza: sectiunea 1 prezinta problema de cercetare in doua susectiuni: (1) motivatia problemei, (2) limitarile limbajelor actuale de specificare a aplicatiilor web; sectiunea 2 expune detaliile tehnice ale solutiei propuse in patru subsectiuni: (1) viziunea conceptuala a solutiei, (2) detalii tehnice, (3) modul de functionare al solutiei si (4) o comparatie cu WebML si WebDSL. Plug-in-ul Eclipse *DevDSL*, care este o demosntratie practica a conceptelor teoretice din acest capitol este prezentat in sectiunea 3 cu urmatoarele subsectiuni: (1) arhitectura si (2) modul de functionare; sectiunea 4 prezinta o lista de studii de caz care contine doua aplicatii demonstrative [105] dezvoltate folosind limbajul propos, un modul de transformare a modelului independent in aplicatii PHP Code Igniter, si un mecanism extins de validare a datelor introduse de utilizator [106].

La final, capitolul 7 prezinta concluziile acestei teze si directiile viitoare de cercetare. Intentiile MDE sunt: (1) imbunatatirea calitatii softului, (2) reducerea timpului de dezvoltare precum si a bugetului alocat proiectelor soft folosing module de generare a rezultatelor proiectelor precum generatoare automate de cod sursa, generatoare automate de cazuri de testare etc; (3) reducerea complexitatii si (4) marirea gradului de reutilizare

a codului ajutand dezvoltatorii sa lucreze la un nivel mai ridicat de abstractizare, astfel reusing sa ignore detaliile irelevante [7]. In acest context, cercetarea actuala a fost orientata spre trei probleme deschise si propune trei solutii conceptuale si tehnice.

Fiecare solutie propusa este insotina de un instrument soft dezoltat cu scopul de a demonstra practic notiunile teoretice introduse. Lista urmatoare prezinta o scurta descriere a acestor instrumente:

- Indep, prezentat in Capitolul 4 este un cadru de dezvoltare care faciliteaza comunicarea dintre obiectele Java si .NET. Acest instrument este o demonstrare practica a conceptelor teoretice despre interoperabilitatea dintre platforme prezentate in [108]. Indep ofera: (1) un generator de cod sursa pentru obiectele surogat si (2) o infrastructura de comunicatie intre Java si .NET.

- Building Blocks Dev Studio, prezentat in Capitolul 5 este o aplicatie dezvoltata in .NET, fiind o demonstratie practica a conceptelor teoretice despre dezvoltarea bazata pe componente prezentate in [102]. Poate fi utilizat impreuna cu *Indep* pentru realizarea de sisteme soft bazate pe componente scrise pentru Java si .NET. Solutia soft ofera: (1) un mediu grafic pentru proiectarea componentelor si legaturilor dintre ele, (2) un generator automat de cod sursa pentru componente exceptand logica de baza a fiecarei componente.

- DevDSL este un plugin Eclipse care implementeaza notiunile teoretice prezentate in lucrarea [107]. Acest plugin ofera: (1) un modul ce poate fi utilizat la scrierea modelului independent de platforma a unei aplicatii si (2) un modul de transformare automata a modelului independent de platforma in aplicatii web .NET.

Doresc sa-mi exprim recunostinta fata de coordonatorul meu, domnul profesor Bazil Pârv, pentru sfaturile, sigestiile si incurajarile sale continue. Mai doresc sa multumesc domnilor profesori Adamko Attila, Simona Motogna, Militon Frentiu, Ioan Laz̆(a)r si tuturor colaboratorilor pentru ajutorul si indrumarea lor. Sunt de asemenea recunoscator tuturor membrilor departamentului de informatica pentru sugestiile lor de mare ajutor.

Doresc sa multumesc pentru sprijinul financiar oferit de la programele co-finantate prin PROGRAMUL OPERATIONAL SECTORIAL DEZVOLTAREA RESURSELOR UMANE, Contract **POSDRU 6/1.5/S/3** – "Studiile doctorale: prin stiinta spre societate".

Bibliography

[1] *Eclipse development using the graphical editing framework and the eclipse modeling framework*. IBM Corp., Riverton, NJ, USA, 2004.

[2] Nour Ali, Rukmani Nellipaiappan, Rajalaxmi Chandran, and Muhammad Ali Babar. Model driven support for the service oriented architecture modeling language. In *Proceedings of the 2nd International Workshop on Principles of Engineering Service-Oriented Systems*, PESOS '10, pages 8–14, New York, NY, USA, 2010. ACM.

[3] Yusuf Altunel and Mehmet R. Tolun. Component-based software development with component variants. In *Proceedings of the 25th conference on IASTED International Multi-Conference: Software Engineering*, pages 235–241, Anaheim, CA, USA, 2007. ACTA Press.

[4] Wim Bast Anneke Kleppe, Jos Warmer. *MDA Explained*. Addison-Wesley Professional, 2003.

[5] Apache. Apache velocity home. http://velocity.apache.org. Last accessed on May 03,2011.

[6] ATL. Atlas transformation language home. http://www.eclipse.org/m2m/atl/. Last accessed on May 03,2011.

[7] P. Tarr B. Hailpern. Model-driven development: The good, the bad, and the ugly. *IBM Systems Journal*, 45(3):451–461, 2006.

[8] L Balmelli B Nolan, B Brown and T Bohn. *Model Driven Systems Development with Rational Products*. IBM, 2008.

[9] A. Basukoski, P. Buhler, V. Getov, S. Isaiadis, and T. Weigold. Methodology for component-based development of grid applications. In *Proceedings of the 2008 compFrame/HPC-GECO workshop on Component based high performance*, CBHPC '08, pages 1:1–1:6, New York, NY, USA, 2008. ACM.

[10] Don Batory, Clay Johnson, Bob MacDonald, and Dale von Heeder. Achieving extensibility through product-lines and domain-specific languages: a case study. *ACM Trans. Softw. Eng. Methodol.*, 11:191–214, April 2002.

[11] Shahid Nazir Bhatti and Asif Muhammad Malik. An xml-based framework for bidirectional transformation in model-driven architecture (MDA). *SIGSOFT Softw. Eng. Notes*, 34:1–5, May 2009.

[12] Stefan Biffl, Richard Mordinyi, and Alexander Schatten. A model-driven architecture approach using explicit stakeholder quality requirement models for building dependable information systems. In *Proceedings of the 5th International Workshop on Software Quality*, WoSQ '07, page 6, Washington, DC, USA, 2007. IEEE Computer Society.

[13] Anna Gerber Bill Moore, David Dean and Gunnar Wagenknecht. *Eclipse Development using the Graphical Editing Framework and the Eclipse Modeling Framework*. IBM, 2004.

[14] G. Blaschek. *Object-Oriented Programming with Prototypes*. Verlag-Springer, 1994. New York.

[15] Grady Booch. The promise the limitsthe beauty of software. http://intranet.cs.man.ac.uk/Events_subweb/special/turing07/PowerPoint _Complete/PPP.pdf, 2007.

[16] Paul Boocock. JaMDA: The java model driven architecture. http://jaMDA.sourceforge.net, 2003. Last accessed on July 13, 2011.

[17] Marco Brambilla. Generation of WebML web application models from business process specifications. In *Proceedings of the 6th international conference on Web engineering*, ICWE '06, pages 85–86, New York, NY, USA, 2006. ACM.

[18] Frank Budinsky, Stephen A. Brodsky, and Ed Merks. *Eclipse Modeling Framework*. Pearson Education, 2003.

[19] Emmanuel Cecchet, Julie Marguerite, and Willy Zwaenepoel. Performance and scalability of ejb applications. In *Proceedings of the 17th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, OOPSLA '02, pages 246–261, New York, NY, USA, 2002. ACM.

[20] Ethan Cerami. *Web Services Essentials: Distributed Applications with XML-RPC, SOAP, UDDI and WSDL*. O'Reilly Media, Inc., 2002.

[21] Stefano Ceri, Marco Brambilla, and Piero Fraternali. Conceptual modeling: Foundations and applications. chapter The History of WebML Lessons Learned from 10

Years of Model-Driven Development of Web Applications, pages 273–292. Springer-Verlag, Berlin, Heidelberg, 2009.

[22] Kenneth Chan and Iman Poernomo. Qos-aware model driven architecture through the uml and cim. *Information Systems Frontiers*, 9:209–224, July 2007.

[23] David Chappell. Introducing sca. http://www.davidchappell.com/articles/ introducing_sca.pdf, 2007. Last accessed on July 14, 2011.

[24] Hyun Cho and Jeff Gray. A domain-specific modeling language for scientific data composition and interoperability. In *Proceedings of the 48th Annual Southeast Regional Conference*, ACM SE '10, pages 107:1–107:4, New York, NY, USA, 2010. ACM.

[25] Gavin King Christian Bauer. *Hibernate in Action: Practical Object/Relational Mapping*. Manning Publications, 2004.

[26] Nuno Amálio Christian Glodt, Pierre Kelsen and Qin Ma. From platform-independent to platform-specific models using democles. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 795–796, 2009.

[27] Stephen Cranefield and Jin Pan. Bridging the gap between the model-driven architecture and ontology engineering. *Int. J. Hum.-Comput. Stud.*, 65:595–609, July 2007.

[28] Krzysztof Czarnecki and Simon Helsen. Classification of model transformation approaches. In *Workshop on Generative Techniques in the Context of Model-Driven Architecture*, 2003.

[29] Eelco Visser Danny M. Groenewegen. Weaving web applications with WebDSL: (demonstration). In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 797–798, 2009.

[30] Andrew J. Kornecki David P. Gluch. Automated code generation for safety related applications: A case study. In *Proceedings of the International Multiconference on Computer Science and Information Technology*, pages 383–391, 2006.

[31] Zekai Demirezen. Semantic framework for dsls. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 833–834, 2009.

[32] Tom Dinkelaker and Mira Mezini. Dynamically linked domain-specific extensions for advice languages. In *Proceedings of the 2008 AOSD workshop on Domain-specific aspect languages*, DSAL '08, pages 3:1–3:7, New York, NY, USA, 2008. ACM.

[33] Mouhamed Diouf, Sofian Maabout, and Kaninda Musumbu. Merging model driven architecture and semantic web for business rules generation. In *Proceedings of the 1st international conference on Web reasoning and rule systems*, RR'07, pages 118–132, Berlin, Heidelberg, 2007. Springer-Verlag.

[34] Bruce Eckel. *Thinking in Java*. MindView, Inc, 2004.

[35] Eclipse. Model to text. http://www.eclipse.org/modeling/m2t, 2010. Last accessed on July 13, 2011.

[36] Elisa & Domain Orientation Case study: An order processing system http://jklunder.home.xs4all.nl/elisa/part04/doc070.html Last accessed on January 24, 2012.

[37] Ralph Johnson Erich Gamma, Richard Helm and John Vlissides. *Design Patterns: 50 specific ways to improve your use of the standard template library*. Pearson Education, Inc, 1995.

[38] Eric Evans. *Domain-Driven Design Quickly*. C4Media Inc, 2006.

[39] Heiko Behrens Michael Clay Sven Efftinge Moritz Eysholdt and contributors. *Xtext User Guide*. XText, 2010.

[40] Moritz Eysholdt and Heiko Behrens. Xtext: implement your language faster than the quick and dirty way. In *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*, SPLASH '10, pages 307–309, New York, NY, USA, 2010. ACM.

[41] Lidia Fuentes, Mónica Pinto, and Pablo Sánchez. Generating CAM aspect-oriented architectures using model-driven development. *Inf. Softw. Technol.*, 50:1248–1265, November 2008.

[42] Dragan Gaaevic, Dragan Djuric, Vladan Devedzic, and Bran Selic. *Model Driven Architecture and Ontology Development*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[43] Mark Gabel, Junfeng Yang, Yuan Yu, Moises Goldszmidt, and Zhendong Su. Scalable and systematic detection of buggy inconsistencies in source code. In *Proceedings of the ACM international conference on Object oriented programming systems languages and applications*, OOPSLA '10, pages 175–190, New York, NY, USA, 2010. ACM.

[44] Nasib S. Gill. Reusability issues in component-based development. *SIGSOFT Softw. Eng. Notes*, 28:4–4, July 2003.

[45] Steffen Goebel and Michael Nestler. Composite component support for ejb. In *Proceedings of the winter international synposium on Information and communication technologies*, WISICT '04, pages 1–6. Trinity College Dublin, 2004.

[46] Adam Griffith. *CodeIgniter 1.7 professional development*. Packt Publishing, 2010.

[47] Danny Groenewegen, Zef Hemel, and Eelco Visser. Separation of concerns and linguistic integration in WebDSL. *IEEE Softw.*, 27:31–37, September 2010.

[48] Danny M. Groenewegen, Zef Hemel, Lennart C.L. Kats, and Eelco Visser. WebDSL: a domain-specific language for dynamic web applications. In *Companion to the 23rd ACM SIGPLAN conference on Object-oriented programming systems languages and applications*, OOPSLA Companion '08, pages 779–780, New York, NY, USA, 2008. ACM.

[49] Danny M. Groenewegen and Eelco Visser. Weaving web applications with WebDSL: (demonstration). In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, OOPSLA '09, pages 797–798, New York, NY, USA, 2009. ACM.

[50] Thomas R. Gross, John L. Hennessy, Steven A. Przybylski, and Christopher Rowen. Measurement and evaluation of the mips architecture and processor. *ACM Trans. Comput. Syst.*, 6:229–257, August 1988.

[51] William Grosso. *Java RMI (Java Series)*. O'Reilly Media, Inc, 2001.

[52] M. Gschwind and D. Maurer. An extendable mips-i processor kernel in vhdl for hardware/software co-design. In *Proceedings of the conference on European design automation*, EURO-DAC '96/EURO-VHDL '96, pages 548–553, Los Alamitos, CA, USA, 1996. IEEE Computer Society Press.

[53] Stuart Hansen and Timothy V. Fossum. Refactoring model-view-controller. *J. Comput. Small Coll.*, 21:120–129, October 2005.

[54] Joseph Heinrich. *MIPS R4000 user's manual*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[55] Zef Hemel. The point of WebDSL. http://zef.me/tag/WebDSL. Last accessed on December 22, 2010.

[56] Zef Hemel, Danny M. Groenewegen, Lennart C. L. Kats, and Eelco Visser. Static consistency checking of web applications with WebDSL. *J. Symb. Comput.*, 46:150–182, February 2011.

[57] Felienne Hermans, Martin Pinzger, and Arie Deursen. Domain-specific languages in practice: A user study on the success factors. In *Proceedings of the 12th International Conference on Model Driven Engineering Languages and Systems*, MODELS '09, pages 423–437, Berlin, Heidelberg, 2009. Springer-Verlag.

[58] Wolfram Hohmann. Supporting modelbased development with unambiguous specifications. In *SAE World Congress, Detroit, MI*, 2004.

[59] Cuiyun Hu, Xinjun Mao, and Hong Ning. Integrating model transformation in agent-oriented software engineering. In *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology - Volume 02*, WI-IAT '09, pages 62–65, Washington, DC, USA, 2009. IEEE Computer Society.

[60] Shan Shan Huang, David Zook, and Yannis Smaragdakis. Domain-specific languages and program generation with meta-aspectj. *ACM Trans. Softw. Eng. Methodol.*, 18:6:1–6:32, November 2008.

[61] Muhammad Jahangir Ikram. A configurable mips simulator for teaching computer architecture. In *Proceedings of the 10th IASTED International Conference on Computers and Advanced Technology in Education*, pages 91–95, Anaheim, CA, USA, 2007. ACTA Press.

[62] Kyungsoo Im, Tacksoo Im, and John D. McGregor. Automating test case definition using a domain specific language. In *Proceedings of the 46th Annual Southeast Regional Conference on XX*, ACM-SE 46, pages 180–185, New York, NY, USA, 2008. ACM.

[63] Mario Rammer Ingo Szpuszta. *Advanced .NET Remoting 2nd Edition.* APRESS, 2005.

[64] Jeronimo Irazabal and Claudia Pons. Supporting modularization in textual dsl development. In *Proceedings of the 2010 XXIX International Conference of the Chilean Computer Science Society*, SCCC '10, pages 124–130, Washington, DC, USA, 2010. IEEE Computer Society.

[65] Hemant Jain, Padmal Vitharana, and Fatemah "Mariam" Zahedi. An assessment model for requirements identification in component-based software development. *SIGMIS Database*, 34:48–63, November 2003.

[66] Pierre-Alain Muller Jean B ezivin, S ebastien G erard and Laurent Rioux. MDA components: Challenges and opportunities. In *Metamodelling for MDA, First International Workshop York, UK, Proceedings*, pages 23–41, 2003.

[67] Jean-Marc Jézéquel, Olivier Barais, and Franck Fleurey. Model driven language engineering with kermeta. In *Proceedings of the 3rd international summer school conference on Generative and transformational techniques in software engineering III*, GTTSE'09, pages 201–221, Berlin, Heidelberg, 2011. Springer-Verlag.

[68] Rod Johnson, Juergen Hoeller, Alef Arendsen, Thomas Risberg, and Dmitriy Kopylenko. *Professional Java Development with the Spring Framework*. Wrox Press Ltd., Birmingham, UK, UK, 2005.

[69] Steven Kelly Juha-Pekka Tolvanen. Metaedit+: defining and using integrated domain-specific modeling languages. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 819–820, 2009.

[70] Salah Kabanda and Mathew Adigun. Extending model driven architecture benefits to requirements engineering. In *Proceedings of the 2006 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries*, SAICSIT '06, pages 22–30, , Republic of South Africa, 2006. South African Institute for Computer Scientists and Information Technologists.

[71] Gerry Kane and Joe Heinrich. *MIPS RISC architectures*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1992.

[72] Lennart C. L. Kats, Karl T. Kalleberg, and Eelco Visser. Domain-specific languages for composable editor plugins. *Electron. Notes Theor. Comput. Sci.*, 253:149–163, September 2010.

[73] Beck Kent. *Test Driven Development: By Example*. Addison-Wesley Professional, 2003.

[74] Christian Köllner, Georg Dummer, Andreas Rentschler, and K. D. Müller-Glaser. Designing a graphical domain-specific modelling language targeting a filter-based data analysis framework. In *Proceedings of the 2010 13th IEEE International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, ISORCW '10, pages 152–157, Washington, DC, USA, 2010. IEEE Computer Society.

[75] Toma Kosar, Pablo E. Martinez Lopez, Pablo A. Barrientos, and Marjan Mernik. A preliminary study on various implementation approaches of domain-specific language. *Inf. Softw. Technol.*, 50:390–405, April 2008.

[76] Stephen Kou and Jens Palsberg. From oo to fpga: fitting round objects into square hardware? In *Proceedings of the ACM international conference on Object oriented*

programming systems languages and applications, OOPSLA '10, pages 109–124, New York, NY, USA, 2010. ACM.

[77] Avraham Leff and James T. Rayfield. Web-application development using the model/view/controller design pattern. In *Proceedings of the 5th IEEE International Conference on Enterprise Distributed Object Computing*, EDOC '01, pages 118, Washington, DC, USA, 2001. IEEE Computer Society.

[78] H. Lieberman. Using prototypical objects to implement shared behavior in object-oriented systems. *Meyrowitz Proceedings*, pages 214–223, 1986.

[79] David Lowe and Rachatrin Tongrungrojana. WebML+ for communication of information flows: an empirical study. In *Proceedings of the 2003 international conference on Web engineering*, ICWE'03, pages 218–221, Berlin, Heidelberg, 2003. Springer-Verlag.

[80] Vincent Massol. *JUnit in Action*. Manning Publications, 2003.

[81] Bernhard Merkle. Textual modeling tools: overview and comparison of language workbenches. In *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*, SPLASH '10, pages 139–148, New York, NY, USA, 2010. ACM.

[82] Marjan Mernik, Jan Heering, and Anthony M. Sloane. When and how to develop domain-specific languages. *ACM Comput. Surv.*, 37:316–344, December 2005.

[83] Microsoft. Introduction to COM interop (visual basic). http://msdn.microsoft.com/en-us/library/kew41ycz.aspx. Last accessed on July 14, 2011.

[84] Mips architecture pictures. http://withfriendship.com/user/servex/mips-architecture.php, 2011. Last accessed on January 25, 2012.

[85] Sunil Mirapuri, Michael Woodacre, and Nader Vasseghi. The mips r4000 processor. *IEEE Micro*, 12:10–22, March 1992.

[86] Nathalie Moreno, Piero Fraternalli, and Antonio Vallecillo. A uml 2.0 profile for WebML modeling. In *Workshop proceedings of the sixth international conference on Web engineering*, ICWE '06, New York, NY, USA, 2006. ACM.

[87] OMG. MOF model to text transformation language, 1.0. http://www.omg.org/spec/MOFM2T/1.0/, 2010. Last accessed on July 13, 2011.

[88] Oracle. Java rmi over iiop. http://download.oracle.com/javase/1.4.2/docs/guide/rmi-iiop. Last accessed on July 14, 2011.

[89] Turhan Ozgur. *Domain-Specific Modeling: A Practical Approach A comparison of Microsoft DSL Tools and Eclipse Modeling Frameworks in the context of Model-Driven Development*. LAP Lambert Academic Publishing, Germany, 2009.

[90] Claus Pahl. Semantic model driven development of web service architectures. *Int. J. Web Eng. Technol.*, 4:386–404, July 2008.

[91] Daniel Perovich, Maria Cecilia Bastarrica, and Cristian Rojas. Model-driven approach to software architecture design. In *Proceedings of the 2009 ICSE Workshop on Sharing and Reusing Architectural Knowledge*, SHARK '09, pages 1–8, Washington, DC, USA, 2009. IEEE Computer Society.

[92] Tracy Gardner Peter Swithinbank, Mandy Chessell and Catherine Griffin. *Patterns: Model-Driven Development Using IBM Rational Software Architect*. IBM, 2005.

[93] Nathaniel Pinckney, Thomas Barr, Michael Dayringer, Matthew McKnett, Nan Jiang, Carl Nygaard, David Money Harris, Joel Stanley, and Braden Phillips. A mips r2000 implementation. In *Proceedings of the 45th annual Design Automation Conference*, DAC '08, pages 102–107, New York, NY, USA, 2008. ACM.

[94] Roman Popp. Defining communication in soa based on discourse models. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 829–830, 2009.

[95] Bartosz Porebski, Karol Przystalski, and Leszek Nowak. *Building PHP Applications with Symfony, CakePHP, and Zend Framework*. Wrox Press Ltd., Birmingham, UK, UK, 1st edition, 2011.

[96] Steven Sanderson. *ASP.NET MVC Framework Preview*. Apress, Berkely, CA, USA, 1 edition, 2008.

[97] Stephen R. Schach. *Object-Oriented Software Engineering*. McGraw-Hill Science/Engineering, 2007.

[98] Andrea Schauerhuber, Manuel Wimmer, and Elisabeth Kapsammer. Bridging existing web modeling languages to model-driven engineering: a metamodel for WebML. In *Workshop proceedings of the sixth international conference on Web engineering*, ICWE '06, New York, NY, USA, 2006. ACM.

[99] Jason H. Sharp and Sherry D. Ryan. A theoretical framework of component-based software development phases. *SIGMIS Database*, 41:56–75, February 2010.

[100] Sylvain Sicard, Noel De Palma, and Daniel Hagimont. J2ee server scalability through ejb replication. In *Proceedings of the 2006 ACM symposium on Applied computing*, SAC '06, pages 778–785, New York, NY, USA, 2006. ACM.

[101] João L. Sobral and Miguel P. Monteiro. A domain-specific language for parallel and grid computing. In *Proceedings of the 2008 AOSD workshop on Domain-specific aspect languages*, DSAL '08, pages 2:1–2:4, New York, NY, USA, 2008. ACM.

[102] Paul Horaţiu Stan. A proposed technique for component based software development. In *Zilele Academice Clujene*, pages 52–56, 2010.

[103] Paul Horaţiu Stan. Building blocks dev studio. a tool for component based development. *Studia UBB, Informatica*, LVI(3):9–14, 2011.

[104] Paul Horaţiu Stan. Building blocks dev studio. a tool for component based development. In *Proc KEPT 2011 (eds: M. Frentiu, HF Pop, S. Motogna)*, pages 247–258, ISSN 2067-1180, Cluj University Press, 2011 (ISI Proc).

[105] Paul Horaţiu Stan. Case study: An order processing system developed using DevDSL. Submitted to: *16th East European Conference on Advances in Databases and Information Systems*, Poznan, September 18-21, 2012.

[106] Paul Horaţiu Stan. A custom validation mechanism for DevDSL. Submitted to: *Studia UBB, Informatica*, LVII(1):, 2012.

[107] Paul Horaţiu Stan. A proposed dsl for data intensive application development. *Studia UBB, Informatica*, LVII(1):accepted, 2012.

[108] Paul Horaţiu Stan and Camelia Şerban. A proposed approach for platform interoperability. *Studia UBB, Informatica*, LV(2):87–98, 2010.

[109] Aldo Bongio Stefano Ceri, Piero Fraternali. *Web Modeling Language (WebML): a modeling language for designing Web sites*. Politecnico di Milano, 2000.

[110] David Steinberg, Frank Budinsky, Marcelo Paternostro, and Ed Merks. *EMF: Eclipse Modeling Framework 2.0*. Addison-Wesley Professional, 2nd edition, 2009.

[111] Susan Stepney. *High Integrity Compilation : a case study*. Prentice-Hall, 1993.

[112] Yu Sun. Model transformation by demonstration. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 831–832, 2009.

[113] Clemens Szyperski. *Beyond Object-Oriented Programming*. ACM Press, 2002. New York.

[114] Mohamed Taleb, Ahmed Seffah, and Alain Abran. Model-driven architecture for web applications. In *Proceedings of the 12th international conference on Human-computer interaction: interaction design and usability*, HCI'07, pages 1198–1205, Berlin, Heidelberg, 2007. Springer-Verlag.

[115] Laurence Tratt Tony Clark. Language factories. In *Proceeding of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications*, pages 949–955, 2009.

[116] Frank Truyen. *The Basics of Model Driven Architecture (MDA)*. Cephas Consulting Corp, 2006.

[117] Pravin V. Tulachan. *Developing EJB 2.0 Components*. Prentice Hall Professional Technical Reference, 2002.

[118] David Upton. *CodeIgniter for Rapid PHP Application Development: Improve your PHP coding productivity with the free compact open-source MVC CodeIgniter framework!* Packt Publishing, 2007.

[119] Matthias Veit and Stephan Herrmann. Model-view-controller and object teams: a perfect match of paradigms. In *Proceedings of the 2nd international conference on Aspect-oriented software development*, AOSD '03, pages 140–149, New York, NY, USA, 2003. ACM.

[120] Thomas Van de Velde, Christian Dupuis, Naveen Balani, and Sing Li. *Beginning Spring Framework 2*. John Wiley & Sons, Inc., New York, NY, USA, 2007.

[121] Eelco Visser. *WebDSL: A Case Study in Domain-Specific Language Engineering*. Delft University of Technology Software Engineering Research Group, 2008.

[122] HaiTao Wang and BaoXian Jia. Research based on web development of Spring integration framework. In *Proceedings of the 2010 International Forum on Information Technology and Applications - Volume 02*, IFITA '10, pages 325–328, Washington, DC, USA, 2010. IEEE Computer Society.

[123] Willian Massami Watanabe, David Fernandes Neto, Thiago Jabur Bittar, and Renata P. M. Fortes. Wcag conformance approach based on model-driven development and WebML. In *Proceedings of the 28th ACM International Conference on Design of Communication*, SIGDOC '10, pages 167–174, New York, NY, USA, 2010. ACM.

[124] WebDSL. A domain-specific language for developing dynamic web applications with a rich data model. http://WebDSL.org. Last accessed on December 12, 2010.

[125] Stephen A. White. Introduction to bpmn. http://www.zurich.ibm.com/~olz/teaching/ETH2011/White-BPMN-Intro.pdf. Last accessed on July 14, 2011.

[126] Wikipedia. Common object request broker architecture. http://en.wikipedia.org/wiki/CORBA. Last accessed on July 14, 2011.

[127] Wikipedia. Component-based software engineering. http://en.wikipedia.org/wiki/Component-based_software_engineering. Last accessed on May 03,2011.

[128] Wikipedia. Spring framework. http://en.wikipedia.org/wiki/Spring_Framework. Last accessed on July 14, 2011.

[129] Daniel T. Wojtaszek and John W. Chinneck. Faster mip solutions via new node selection rules. *Comput. Oper. Res.*, 37:1544–1556, September 2010.

[130] Hui Wu and Jeff Gray. Testing domain-specific languages in eclipse. In *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, OOPSLA '05, pages 173–174, New York, NY, USA, 2005. ACM.

[131] Hui Wu, Jeff Gray, and Marjan Mernik. Unit testing for domain-specific languages. In *Proceedings of the IFIP TC 2 Working Conference on Domain-Specific Languages*, DSL '09, pages 125–147, Berlin, Heidelberg, 2009. Springer-Verlag.

[132] E. Visser Z. Hemel, L.C.L Kats. *Code Generation by Model Transformation. A Case Study in Transformation Modularity.* Delft University of Technology Software Engineering Research Group, 2008.

[133] Guotao Zhuang and Junwei Du. MDA-based modeling and implementation of e-commerce web applications in WebML. In *Proceedings of the 2009 Second International Workshop on Computer Science and Engineering - Volume 02*, IWCSE '09, pages 507–510, Washington, DC, USA, 2009. IEEE Computer Society.