hyväksymispäivä arvosana

arvostelija

AOP, reflektio ja metaohjelmointimekanismit Javassa

Tuomas Nurmela

Helsinki 27. lokakuuta 2002

58302304 Reflective Middleware

Seminaari

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

AOP ja metaohjelmointitekniikat Javassa

Tuomas Nurmela

58302304 Reflective Middleware

Seminaari

Tietojenkäsittelytieteen laitos

Helsingin Yliopisto

27.10.2002, 13 sivua

Tiivistelmä

Aspektilähtöinen ohjelmointi (AOP) ja reflektio, sovelluksen kyky tulkita ja mukautua muutoksiin ennaltamääritettyjen sopimusten mukaisesti, ovat viimeaikoina nousseet vastaukseksi sekä olio-ohjelmoinnin rajoitteita että sovelluspalvelimen adaptaatiota toteuttavina lähestymistapoina. Yhdistettynä nämä kykenevät lisäksi tarjoamaan sovelluspalvelintoimittajille uuden tavan tuottaa sovelluspalvelimen palveluja rajoittumatta komponenttiarkkitehtuurin (esim. J2EE, CORBA) vaatimuksiin, mahdollistaen palveluiden tuoton myös arkkitehtuurista poikkeaville toteutuksille.

Java ja J2EE-sovelluspalvelimet ovat olleet sekä tutkimuksen että avoimen sovelluskoodin yhteisön kehityksessä mukana molemmilla rintamilla. a tästä huolimatta CORBA-projektit sekä näiden saavutukset ovat herättäneet akateemisissa piireissä aina enemmän mielenkiintoa. Paperi tarkastelee Javan AOP:n tilannetta sovellusohjelmoijan kannalta. Johdannon jälkeinen sektio 2 esittelee AOPin uranuurtajan, AspectJ:n sekä tätä seuranneet, lähtökohtaa edelleenkehittäneet staattiset AOP-toteukset. Seuraava sektio metaohjelmointimekanismit eli tekniikat, joilla tuottaa AOP dynaamisesti Javassa. Viimeinen sektio käsittelee AOP:n ja reflektion yhdistämistä sovelluspalvelimissa sovelluksen saatavuuden ja vikasietoisuuden kehittämisen näkökulmasta.

Avainsanat: Aspektilähtöinen ohjelmointi, Java metaohjelmointimekanismit, reflektio, itsensä korjaavat sovellukset

SISÄLTÖ

1 Johdanto

Komponenttilähtöinen sovelluskehitys (component-based software engineering, CBSE) sisältää tällä hetkellä useita haasteita. Näistä keskeisiä ovat olio-ohjelmoinnin (object-oriented programming, OOP) uudelleenkäytettävyyden rajoittaminen, komponenttiarkkitehtuurin rajoitteet sekä sovelluspalvelinten rajoitteet.

OOP:ssa olion on kapseloiva kaikki näkökulmat, joista niitä voidaan lähestyä. Tämä edellyttää mm. ei-toiminnallisten piirteiden (kuten lokiinkirjoituksen tai debuggauksen) koodin sekoittamisen itse ydinongelmaa käsittelevään koodin. *Aspektilähtöinen ohjelmointi* (aspect-orriented programming, AOP) on OOP:n ja CBSE:n rinnalla toimiva lähestymistapa, mahdollistaen monipuolisemman *aihealueiden erottamisen* (seperation of concerns, [HL95]). AOP täydentää sekä OOP:a että CBSE:a, joskin yhtälailla AOP:n täytyy yhtälailla itse tukeutua CBSE:n ajatusmallin mukaisesti aspektien uudelleenkäytettävyyteen.

CBSE:n myötä sovelluspalvelinten sovelluskehykset ovat tarjonneet yleispalveluita tietoturvasta tietokantakäsittelyyn, automatisoiden samalla esim. säikeistyksen. Tämä on parantanut liiketoimintalogiikan uudelleenkäytettävyyttä, vaikkakaan yleispalveluiden käyttöön kirjoitetun koodin osalta uudelleenkäytettävyys ei ole usein toteutunut. AOP:n sisällyttäminen sovelluspalvelimiin mahdollistaa näiden palveluiden tuottamisen joko komponenttiarkkitehtuurin mukaisille sovelluksille tai vaihtoehtoisilla tavoilla tämän rinnalla. Yhtälailla sovelluspalvelimen refaltoroimalla, kykenevät sovelluspalvelintoimittajat itsekin tuottamaan parempaa koodin kapselointia. Hyödyntämällä metaohjelmointimekanismeja, AOP kyetään tuottamaan luokan latausvaiheessa tai ajonaikaisesti, näin ollen välttäen uudelleenkääntämistarve. Yhdistämällä tähän reflektion mukainen kyky tulkita ympäristöä ja mukautua siihen, kyetään sovelluspalvelimissa tuottamaan uudenlaisia palveluita tuntumattomasti sovellukselle.

Tutkielma keskittyy ylläolevaan ongelmakenttään Javan näkökulmasta. Paperin rakenne on seuraava: kohdassa 2 tarkastellaan Javan tunnetuinta AOP-sovelluskehystä, Aspect J:tä, joka on staattinen AOP toteutus. AspectJ toimii vertailukohtana metaohjelmointimekanismeihin pohjautuviin dynaamisiin AOP ratkaisuihin. Kohdassa 3 tarkastellaan erillaisia metaohjelmointimekanismeja Javalla sekä näiden puitteissa toteutettuja AOP-sovelluskehyksiä. Kohdassa 4 tarkastellaan JBoss sovelluspalvelimen AOP-sovelluskehystä sekä sovelluspalvelimelle tyypillisiä tavoitteita ja sovellusalueita. Samassa yhteydessä keskitytään erityisesti itsekorjaaviin ominaisuuksiin. Kohta 5 sisältää liittymät muihin tutkimuksiin kohdan 6 sisältäen päätelmät edellä esitetyn perusteella.

Lähteet

Vviitteet tarkistettiin 3.10.2003

[Asp03] Aspectworkz homepage, http://aspectwerkz.codehaus.org/

[Bac01] J.R.Bachrach: "The Java Syntactic Extender", http://www.ai.mit.edu/~jrb/jse

[BH02] Jason Baker, Wilson Hsieh: "Runtime Aspect Weaving Through Metaprogramming", University of Utah, 2002, http://citeseer.nj.nec.com/baker01runtime.html

[Bha03] Nitin Bharti: "Tech talk with Gregor Kickzales on AOP", TheServerSide, 2003 http://www.theserverside.com/events/videos/GregorKiczalesText/interview.jsp

[BK02] David Bosschaert, Arne Koschiel: "Advanced Class Loading in J2EE", The ServerSide, 2002, http://www.theserverside.com/resources/article.jsp?l=AdvancedClassLoading

[Box97] Don Box, *Essential COM*, Addison-Wessley, 1997

[Box02] Don Box, *Essential .NET: Common Language Runtime*, Addison-Wessley, 2002

[BR01] Eric Bruneton, Michel Riveill: "Experiments with JavaPod, a platform designed for the adaptation of non-functional properties",Reflection 2001, Lecture Notes on Computer Systems 2192, Springer-Verlag, 2001

[Bur03] Bill Burke: "Aspect-Oriented Programming and JBoss", OnJava, http://www.onjava.com/lpt/a/3878

[Bur03b] Bill Burke: "Interview: JBoss 4.0 AOP Framework", TheServerSide, 2003 http://www.theserverside.com/events/videos/BillBurke/dsl/q15.html

[CCF03] G. Candea, J.Cutler, A. Fox: "Improving availability with recursive Micro-boots: a soft-state system case study", to appear, Performance Evaluation Journal, Summer 2003 http://citeseer.nj.nec.com/candea03improving.html

[CKF+02] Mike Chen, Emre Kiciman, Eugene Brewer, Armando Fox, Eric Brewer: "Pinpoint: Problem Determination in Large Dynamic Internet Services", DSN 2002

[CKZ+03] George Candea, Emre Kýcýman, Steve Zhang, Pedram Keyani, Armando Fox: "JAGR: An autonomous Self-Recovering Application Server", Proceedings of the 5th International Workshop on Active Middleware Services, Seattle, WA, June 2003 http://citeseer.nj.nec.com/candea03jagr.html

[Chi03] Shigeru Chiga, Javassist, http://www.csg.is.titech.ac.jp/~chiba/javassist/

[CHV01] Denis Caromel, Fabrice Huet, Julien Vayssiere: "A simple security-aware MOP for Java",Reflection 2001, Lecture Notes on Computer Systems 2192, Springer-Verlag, 2001

[CKF+01] Yvonne Coady et al. "Structuring Operating Systems Aspects", CACM October 2001

[CST03] Shigeru Chiba, Yoshiki Sato, Michiski Tatsubori: "Using HotSwap for Implementing Dynamic AOP Systems", July 13th, 2003

[Dah03] M. Dahm, The Bytecode Engineering Library, http://jakarta.apache.org/bcel/

[Ewa00] Timothy Ewald: “Use AppCenter Server or COM and MTS for Load Balancing your Component Servers”, Microsoft Systems Journal, January 2000

[HL95] Walter L. Hursch, Cristina Videira Lopes: “Seperation of Concerns”, Febtruary 1995

[HP03] Salim Hariri, Manish Parashar, Tutorial on Autonomic Computing, AICCSA’2003, July 2003, http://automate.rutgers.edu/tutorials/AICCSA2003.htm

[HU01] R. Hanenberg, R.Unland: “Using and reusing aspects in AspectJ”, OOPSLA, October 2001

[HS99] Galen C. Hunt, Michael L. Scott: “Intercepting and Instrumenting COM Applications”, COOTS’99, May 3-7, 1999

[Joh+03] Rod Johnson, *Expert One-to-One J2EE Design and Development*, Wiley Publishing, 2003

[KCB+02] Fabio Kon, Fabio Costa, Gordon Blair, Roy H. Cambell: “The Case for Reflective Middleware”, CACM October 2002

[Kic96] Gregor Kiczales: “Beyond the Black Box: Open Implementation”, IEEE Software, January 1996

[KLL+] Gregor Kiczales et al. “Open Implementation Design Guideliness”, ICSE 1997

[Lad02] Ramnivas Laddad: “I want my AOP!”, artikkelisarja, JavaWorld, www.javaworld.com/javaworld/jw-01-2002/jw-0118-aspect_p.html

[Lee03] Bob Lee, Jadvice Home Page, vhttp://crazybob.org/downloads/jadvise-javadoc/

[Lee03] Bob Lee: “Intercepting Method Invocations on POJOs using jAdvice”, Java Technical Insight of the Month, Object Computing Inc, February 2003

[Nan03] Nannings homepage, http://nanning.snipsnap.org/space/start

[NEF01] Paniti Netinant, Tzilla Elrad, Mohammed E. Fayad: “A Layered Approach to Building Open Aspect-Oriented Systems”, CACM October 2001

[NMS99] Priya Narasimhan, Louise E. Moser, P.M. Melliar-Smith: ”Using Interceptors to Enhance CORBA”, IEEE Computer, July 1999

[OL01] D. Orleans, K. Lieberherr: “DJ: Dynamic adaptive programming in Java”, Tech. Rep. College of Computer Science, Northeastern University, Boston, MA, March 2001.

[OMG01] OMG, *Extensible Transport Framework*

[OMG02] OMG, *CORBA 3.0.2 Specification: Chapter 21 – Portable Interceptors*, 02.12.2002 www.omg.org/technology/documents/formal/corba_iiop.htm

[OT01] Harold Ossher, Perri Tarr: “Using Multidimensional Seperation of Concerns to (Re)shape Evolving Software”, CACM October 2001

[PAG01] Andrei Popovici, Gustavo Alonso, Thomas Gross: “Just in Time Aspects: Efficient Dynamic Weaving for Java”,

[Pet02] Brett Peterson: “Understanding J2EE Application Server Class Loading Architecture”, The Server Side, May 2002

[PGA01] Andrei Popovici, Thomas Gross, Gustavo Alonso: “Dynamic Weaving for Aspect-Oriented Programming”, Proceedings of the 1st International Conference on Aspect-Oriented Software Development, 2001
http://citeseer.nj.nec.com/popovici02dynamic.html

[PSD+01] R. Pawlak, L. Seinturier, L. Duchien, G. Florin, “JAC: A Flexible Solution for Aspect-Oriented Programming in Java,” in Metalevel Architectures and Separation of Crosscutting Concerns (Reflection 2001), LNCS 2192, pp. 1–24, Springer, 2001.

[SMS02] Nuno Santos, Paulo Marques, Luis Silva: ”A Framework for Smart Proxies an Interceptors in Java”, University of Coimbra, Portugal, 2002
http://citeseer.nj.nec.com/542994.html

[Sos03] Dennis M. Sosnovski: ”Java-programming dynamics”, artikkelisarja, developerWorks, IBM, http://www-106.ibm.com/developerworks/java/library/j-dyn0429/

[Sul01] Gregory T. Sullivan: ”Aspect Oriented Programming Using Reflection and Metaobject protocols”, CACM October, 2001

[Sun99] Sun: “Java Debugger Interface”, SDK 1.2 dokumentaatio, 1999
http://java.sun.com/products/jdk/1.2/docs/guide/jvmdi/

[Sun02] Sun: ”Java Platform Debugger Architecture”, SDK 1.4 dokumentaatio, 2002
http://java.sun.com/j2se/1.4.1/docs/guide/jpda/index.html

[SVJ+02] Davy Suvée, Wim Vanderperren, Viviaine Jonckers: “JasCo: an Aspect-oriented approach tailored for Component-based software development”, AOSD 2003

[Wha01] John Whaley: “System Checkpointing using Reflection and Program analysis”, Reflection 2001, Springer-Verlag, Lecture Notes on Computer Systems 2192, 2001

[WPS01] Nanbor Wang, Kirthika Parameswaran, Douglas Schmidt: “The Design and Performance of Meta-Programming Mechanisms for Object Request Broker Middleware”, COOTS’01, Jan/Feb 2001