



Departamento de Informática
Universidad de Castilla-La Mancha
Tesis Doctoral

**CARACTERIZACIÓN DEL
CONOCIMIENTO EN DISEÑO DE
MICRO ARQUITECTURAS
ORIENTADAS A OBJETOS**

Ciudad Real, Junio de 2004

Doctorando: D. Javier Garzás Parra

Director: Dr. D. Mario G. Piattini Velthuis

Referencias

Abran A., Bourque P., Dupuis R. y Moore J. (2001) Guide to the Software Engineering Body of Knowledge - SWEBOK, IEEE Press.

Alexander C. (1977). *A pattern language*. New York: Oxford University Press.

Alexander C. (1979). *The timeless way for building*. New York: Oxford University Press.

Alur D. y Malks D. (2001). *Core J2EE Patterns*. Pertice-Hall.

Amaro S., Martínez N. y Cechich A. (2001). Formalising Composite Patterns. XXVII Conferencia Latinoamericana en Informática (CLEI 01).

Ambler S. W. (1998). *Process Patterns Building Large-Scale Systems Using Object Technology*. Cambridge University Press/SIGS Books.

Appleton B (2001). *Patterns and Software: Essential Concepts and Terminology*. En <http://www.enteract.com/~bradapp/docs/patterns-intro.html>

Asencio A., Cardman S. Harris D. y Laderman E. (2002). Relating Expectations to Automatically Recovered design Patterns. Proceedings of the Ninth Working Conference on Reverse Engineering (WCRE 02). Computer Society Press.

Avison D., Lau F., Neilsen P.A. y Myers M. (1999). Action Research. Communications of ACM, Vol.42, No 1, pp. 94-97.

Backlund P. (2003). *Knowledge Transfer in Information Systems Engineering*. Tesis Doctoral, Department of Computer and Systems Sciences, Stockholm University/Royal Institute of Technology.

Baniassad E., Murphy G. C. y Schwanninger C. (2001). Understanding Design Patterns with Design Rationale Graphs. OOPSLA 2001, Workshop Beyond Design: Patterns (mis)used. Bahía Tampa, Florida, EEUU

Baniassad E. L., Murphy G. C. y Schwanninger C. (2003). Design Patterns Rationale Graphs: Linking Design to Source. IEEE 25th International Conference on software Engineering.

Bansiya J. (1998, Junio). Automating Design-Pattern Identification. Dr. Dobb's Journal.

Basili V. y Weiss D. (1984). A Methodology for Collecting Valid Software Engineering Data, IEEE Transactions on Software Engineering, 10, 728-738."

Basili V. y Rombach H. (1988). The TAME Project: Towards Improvement-Oriented Software Environments. IEEE Transactions on Software Engineering 14, 758-773."

Beck K. y Fowler M. (2000). Bad Smells in Code. En Fowler M. (Ed.) Refactoring improving the design of existing code. Addison Wesley

Beck K. (2001). Extreme Programming Explained. Addison-Wesley.

Bell G., Morrey I. y Pugh J. (1987). Software Engineering: a programming approach. New York: Prentice Hall.

Bennet K. H. y Rajlich V. T. (2000). Software Maintenance and Evolution: a Roadmap. En Finkelstein A. (Ed.), The future of Software Engineering, ICSE 2000 (pp 75-87). Limerick, Ireland.

Berczuk S. P. y Appleton B. (2002). Software Configuration Management Patterns: Effective Teamwork, Practical Integration. Addison Wesley Profesional.

Bernárdez B., Durán A., Genero M. y Toro M. (2004). A Controlled Experiment for Evaluating and Metric – Beased Technique for Requirements Inspection. 10th International Software Metrics Symposium. METRICS 2004.

Booch G. (1996). Managing the Object-Oriented project. Addison-Wesley.

Brown W. J., Malveau R. C., Brown W. H., McCormick H. W. y Mowbray T. J. (1998). Antipatterns: Refactoring Software, Architectures, and Projects in Crisis. John Wiley & Sons.

Brooks F. P. (1987). No Silver Bullet: Essence and Accidents of Software Engineering. *Computer*, Vol. 20, No 4, pp. 10-19.

Brachman R. J y Schmolze J. G. (1985). An overview of the KL-ONE knowledge representation system. *Cognitive Science*, Vol. 9, No 2, pp. 171–216.

Briand L., Bunse C. y Daly J. (1999). A Controlled Experiment for evaluating Quality Guidelines on the Maintainability of Object-Oriented Designs. Technical Report IESE 002.99/E, Fraunhofer Institute for Experimental Software Engineering, Kaiserslautern, Alemania.

Bundinsky F. J. y Finnie M. A. (1996). Automatic Code Generation form Design Patterns. *IBM Systems Journal*, Vol. 31, No 2.

Buschmann F., Meunier R., Rohnert H., Sommerlad P. y Stal M. (1996). A System of Patterns: Pattern-Oriented Software Architecture, Addison-Wesley.

Chidamber S. R. y Kemerer C. F. (1994). A metric Suite for Object - Oriented Design. IEEE Transactions on Software Engineering, Vol. 20, No 6, pp. 476 - 493

Cline M. P. (1996). The Pros and Cons of Adopting and Applying Design Patterns in the Real World. Communications ACM, , vol. 39, n 10

Coad P. (1992). Object-Oriented Patterns. Communications ACM, vol. 35, n 9, pp. 152-159.

Conte A., Fredj M., Hassine I., Giraudin J.P. y Rieu D. (2002). A Tool and Formalism to Design and Apply Patterns. En Bellahsène Z., Patel D. y Rolland C. (Eds.), OOIS 2002, 8th International Conference on Object-Oriented Information Systems (pp. 135- 146). Springer.

Coplien J. (1991). Advanced C++ Programming Styles and Idioms. Addison-Wesley.

Coplien J. (1996). SIGS Management Briefings "software Patterns". SIGS Books & Multimedia.

Coplien J. (1998). Software Design Patterns: Common Questions and Answers. En Linda Rising (Ed.), The Patterns Handbook: Techniques, Strategies, and Applications (pp.311-320). Nueva York: Cambridge University Press.

Correa A., Werner C. y Zaverucha G. (2000). Object Oriented Design Expertise Reuse: an Approach Based on Heuristics, Design Patterns and Anti-Patterns. International Conference on Software Reuse (ICSR). pp. 336-352

Cortés M., Fontoura M. y Lucena C. (2003). Using Refactoring and Unification Rules to Assist Framework Evolution. ACM SIGSOFT Software Engineering Notes. Vol. 28, No 6. ACM Press

Cranefield S., Haustein S., y Purvis M. (2001). UML-Based Ontology Modelling for Software Agents. Proceedings of the Workshop on Ontologies in Agent Systems, 5th International Conference on Autonomous Agents, Montreal, pp. 21-28.

Creel C. (1999). Requirements by Pattern. *Software Development Magazine*, en www.sdmagazine.com

Crespo Y. y Marqués J. M. (2001). Definición de un marco de trabajo para el análisis de refactorizaciones de software. JISBD 2001, VI Jornadas de Ingeniería del Software y Bases de Datos (JISBD01). Almagro, Ciudad Real, España.

Davis A. (1995) 201 Principles of Software Development. McGraw-Hill.

DeMarco T. (1979). *Structured Analysis and System Specification*, Yourdon Press Computing Series

Deneckère R. y Souveyet C. (2001). Organising and Selecting Patterns in Patterns Languages with Process Maps. En Wang Y., Patel S. y Johnston R.H. (Eds.), *OOIS 2001, 7th International Conference on Object-Oriented Information Systems* (pp. 15-24). Universidad de Calgary, Calgary, Canadá: Springer.

Deneckère R. (2002). Using Meta - Patterns to Construct Patterns. En Bellahsene Z., Patel D. y Rolland C. (Eds.), *OOIS 2002, 8th International Conference on Object-Oriented Information Systems* (pp. 124- 134). Springer.

Denning P. J. (2003). Great Principles of Computing. *Communications of ACM*, Vol.46, No 11, pp. 15-20

Deursen A. y Moonen L. (2002). The video store revisited - thoughts on refactoring and testing. 3rd Conference eXtreme Programming and Flexible Processes in Software Engineering (XP2002), pp 71-76.

Dodani M. (2003). The Best Practices Promise and Myths. *Journal of Object Technology*, Vol. 2, No. 4, pp. 65-68.

Dong J. Y Yang S. (2003). Extending UML to Visualize Design Patterns in Class Diagrams. 15th International Conference on Software Engineering & Knowledge Engineering.

Emden E. y Moonen L. (2002). Java Quality Assurance by Detecting Code Smells. Proceedings of the Ninth Working Conference on Reverse Engineering (WCRE 02). Computer Society Press.

Fenton N. y Pfleeger S. (1997). *Software Metrics: A Rigorous Approach*. 2a edición. London, Chapman & Hall.

Fernández, M., Gómez-Pérez, A., y Juristo, N. (1997). METHONTOLOGY: From Ontological Art Towards Ontological Engineering. AAAI Spring Symposium. University of Stanford; Palo Alto, California (Estados Unidos), pp. 33-40.

Ferreira M. J. y Loucopoulos P. (2003). Business Analysis Patterns: Methodological and Support Environment Aspects. Ninth Americas Conference on Information Systems.

Florijn G., Meijers M. y van Winsen P. (1997). Tool Support for Object-Oriented Patterns. En Askit M., S. Matsuoka (Eds.), *ECOOP'97. Lecture Notes in Computer Science*, 1241. Berlin: Springer-Verlag.

- Fowler M. (1996). *Analysis Patterns: Reusable Object Models*. Addison-Wesley
- Fowler M. (2000). *Refactoring improving the design of existing code*. Addison Wesley
- Fowler M. (2001a). Reducing Coupling, *IEEE Software*, vol 18, nº 4.
- Fowler M. (2001b). Separating User Interfaz Code, *IEEE Software*, vol 18, nº2.
- France R. B., Kim D.K., Ghosh S., Song E. (2004). UML – Based Patterns Specification Technique. *IEEE Transactions on Software Engineering*. Vol. 30 No 3.
- Gabriel R. P. (1996). *Patterns of Software: Tales from the Software Community*. Oxford University Press.
- Gamma E., Helm R., Johnson R. y Vlissides J. (1995). *Design patterns: Elements of Reusable Object Oriented Software*. Addison-Wesley.
- García F., Ruíz F., Bertoa M. F., Calero C., Genero M., Olsina L., Martín M., Quer C., Tondori N., Abrahao S., Vallecillo A. y Piattini M. (2004). Una Ontología de la Medición del Software. Informe Técnico UCLM DIAB-04-02-2. Departamento de Informática. Universidad de Castilla – La Mancha.
- Gomes P., Pereira F., Paiva P., Seco N., Carreiro P., Ferreira J. L. y Bento C. (2003). Selection and reuse of Software Patterns Using CBR and WordNet. 15th International Conference on Software Engineering & Knowledge Engineering.
- Gómez-Pérez, A. (1998). Knowledge sharing and reuse. *The Handbook of Applied Expert Systems*. Jay Liebowitz (ed) CRC Press, 1998.
- Grand M. (2001). *Java Enterprise Design Patterns. Patterns in Java Volume 3*. Wiley Computer
- Grant M. (1998). *Patterns in Java: a Catalog of reusable design patterns illustrated with UML - Volume 1*. Wiley Computer.
- Gruber, T. (1991). The Role of a Common Ontology in Achieving Sharable, Reusable Knowledge Bases. *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*. Cambridge
- Guizzardi, G., Herre, H. y Wagner, G. (2002). Towards Ontological Foundations for UML Conceptual Models. *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE*. LNCS 2519, 1100-1117.

Harrison W. (2004). Best Practices: who says? IEEE Software, Vol. 21, No 1.

Helm R., Holland I. y Gangopadhyay D. (1990). Contracts: specifying behavioral compositions in object-oriented systems. 5th Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA '90). ACM SIGPLAN Notices, Vol. 25, No. 10, pp. 303-311.

IEEE (1997). STD 1074-1997: IEEE Standard for Developing Software Life Cycle Processes.

ISO 9126 (2001). Software Engineering - Product Quality. ISO/IEC 9126

Janaki D., Anantharaman K.N., Guruprasad K. N., Sreekanth M., Raju S.V.G.K. y Rao A. (2000). An Approach for Pattern Oriented Software Development Based on a Design Handbook. Annals of Software Engineering, Vol. 10, 2000.

Janaki D., Jithendra P. y Rajasree M. S. (2003). An Approach to Estimate Design Attributes of Interacting Patterns. Proceedings of the 7th ECOOP Workshop on Quantitative Approaches of Object Oriented Software Engineering (QAOOSE 03)

Ji K. y Chen S. (2001). DEPA (Design Pattern Application) - a component-based model for applying design patterns in software development. Informatica (Slovenia). Vol 25, No 4.

Johnson R. y Foote B. (1988). Designing Reusable Classes. Journal of Object-Oriented Programming, 1:2, pp 22-35

Johnson, R. (2001). Personal Communication.

Jurisica I., Mylopoulos J. y Yu E. (1999). Using Ontologies for Knowledge Management: An Information Systems Perspective Knowledge: Creation, Organization and Use. Proceedings of the 62nd Annual Meeting of the American Society for Information Science (ASIS'99). Washington, D.C.

Keller R. K. y Schauer R. (1998). Design components: Towards software composition at the design level. Proceedings of the 20th International Conference on Software Engineering, pp. 302-311.

Kerievsky J. (2004). Refactoring To Patterns. Addison Wesley.

Khirs I., Keller R. K. y Hamid I. A. (1999). Supporting Design by Pattern-based Transformations. Proceedings of the Second International Workshop on strategic Knowledge and Concept Formation, pp 157-167.

Kishore R., Zhang H. y Ramesh R. (2004). A Helix-Spindle Model for Ontological Engineering. *Communications of the ACM*, Vol.47, No.2.

Koenig A. (1995, Marzo - Abril). Patterns and Antipatterns. *Journal of Object-Oriented Programming (JOOP)*, Vol. 8, No 1, pp. 46-48

Landay J. A. y Borriello G. (2003). Design Patterns for Ubiquitous Computing. *Computer*, Vol. 36, No 8, pp. 93 – 95.

Larman C. (2001). *Applying UML and Pattern: An Introduction to Object-Oriented Analysis and Design and the Unified Process (2nd Edition)*. Prentice Hall.

Larsen G. (1999). Designing component-based frameworks using patterns in the UML. *Communications of the ACM*, Vol. 42, No 10, pp. 38 – 45.

Lauder A. y Kent S. (1998). Precise Visual Specification of Design Patterns. ECOOP '98 Bruselas, Bélgica. Publicado en *Lecture Notes in Computer Science*, Vol. 1445 / 1998, pp.114 – 134.

Li W. y Henry S. M. (1993). Object-oriented metrics that predict maintainability. *Journal of Systems and Software*, Vol. 23, No 2, pp. 111 - 122.

Li L. (2002). Use Case Patterns. *International Journal of Software Engineering and Knowledge Engineering*. Vol.12, No 1, pp.19-40

Lieberherr K. J. y Holland I. M. (1989). Assuring Good Style For Object-Oriented Programs. *IEEE Software*, pp 38-48

Liskov B. H. y Zilles S. N. (1974). *Programming with Abstract Data Types.*, Computation Structures Group, n 99, MIT, Project MAC, Cambridge Mass.

Love T. (1997). *Object Lessons: Lessons Learned in Object-Oriented Development Projects*. SIGNATURE SOUNDS RECORDING

Mäntylä M. y Vanhanen J. (2003). A Taxonomy and an Initial Empirical Study of Bad Smells in Code. *Proceedings of the IEEE International Conference on Software Maintenance (ICSM'03)*

Marinescu F. (2002). *EJB Design Patterns Advanced Patterns, Processes, and Idioms*. Wiley Computer.

Martin R. C. (1995). Object Oriented Design Quality Metrics: An analysis of dependencies. *ROAD*, Vol. 2, Nº 3.

- Martin R. C. (1996, Enero – Noviembre). Engineering Notebook. C++ Report.
- May D. y Taylor P. (2003). Knowledge Management with Patterns. Communications of the ACM, Vol. 46 No 7
- McConnell S. (2003). Professional Software Development. Addison-Wesley.
- Mens T. y Tourwé T. (2004). A Survey of Software Refactoring. IEEE Transactions on Software Engineering, Vol. 30, No 2, pp. 126 - 139
- Meyer B. (1998). Object Oriented Software Construction. Prentice Hall.
- Nordberg M. E. (2001). Aspect-Oriented Indirection – Beyond OO Design Patterns. OOPSLA 2001, Workshop Beyond Design: Patterns (mis)used. Bahía Tampa, Florida, EEUU
- Novatica. (2002, Marzo - Abril). Extreme Programming. Programación Extrema. Revista de Asociación de Técnicos en Informática.
- Ó Cinnéide, M. (2000). Automated Application of Design Patterns: a Refactoring Approach. Tesis Doctoral. Universidad de Dublin, Trinity College.
- Oivo M., Kuvaja P., Pulli P. y Similä J. (2004). Software Engineering Research Strategy: Combining Experimental and Explorative Research (EER). PROFES 2004, Bomarius F. y Lida H. (Eds.). Lectures Notes in Computer Science, pp. 302 – 317. Springer – Verlag.
- Opdyke W. (1992). Refactoring Object-Oriented Frameworks. Tesis Doctoral, Departamento de Computer Science. University of Illinois at Urbana-Champaign.
- Opdyke W. (2000). Refactoring, reuse and reality. En Fowler M. (Ed.), Refactoring. Improving The Existing Code. Addison – Wesley.
- Page-Jones M. (1988). The Practical Guide to Structured Systems Design, 2a edición. Yourdon Press Computing Series.
- Perry D., Porte A. y Votta L. (2000). Empirical Studies of Software Engineering: A Roadmap. Future of Software Engineering. Ed. Anthony Finkelstein, ACM, 345-355
- Pescio C. (1997). Principles Versus Patterns. IEEE Computer, Vol. 30, No. 9, pp. 130-131
- Pigoski, T. M. (1997). Practical Software Maintenance. Best Practices for Managing your Investments. USA: John Wiley & Sons.

Prechelt L., Unger B., Philippsen M. y Tichy W. (1997). Two controlled Experiments Assessing the Usefulness of Design Patterns Information During Program Maintenance. *Empirical Software Engineering*.

Prechelt L., Unger B., Tichy W. y Bossler P. (2000). A controlled Experiments in Maintenance Comparing Design Patterns to Simpler Solutions. *IEEE Transactions on Software Engineering*.

Priestley M. (2001). *Practical Object Oriented Design with UML*. McGrawHill

Redwine S.T. (1984), "DOD-Related Software Technology Requirements, Practices, and Prospects for the Future," Tech. Report P-1788, Inst. Defense Analyses, Alexandria.

Reibing R. (2001a). Assessing the Quality of OO Designs. *OOPSLA 2001*. Bahía Tampa, Florida, EEUU

Reibing R. (2001b). The impact of Pattern Use on Design Quality. *OOPSLA 2001, Workshop Beyond Design: Patterns (mis)used*. Bahía Tampa, Florida, EEUU

Reifer D. J. (2003). Is the Software Engineering State of the Practice Getting Closer to State of the Art? *IEEE Software*, Vol. 20, No 6

Riehle D. y Zlighthoven H. (1996). Understanding and Using Patterns in Software Development. *Theory and Practice of Object Systems*, 2 (1), pp.3-13.

Riehle D. (1997). Composite design patterns. *Conference on Object Oriented Programming Systems Languages and Applications (OOPSLA)*. Proceedings of the 12th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications. Atlanta, EEUU, pp. 218 – 228.

Riel A. J. (1996). *Object-Oriented Design Heuristics*. Addison-Wesley.

Rising L. (1998). *The Patterns Handbook: Techniques, Strategies, and Applications*, Cambridge University Press.

Roberts D. B. (1999). *Practical Analysis For Refactoring*. PhD Thesis, Department of Computer Sciencie. University of Illinois.

Ruiz F. (2003). *MANTIS: Definición de un Entorno para la Gestión del Mantenimiento de Software*. Tesis Doctoral. Dep. de Informática, Univ. de Castilla-La Mancha.

Schmidt D. C. (1995, Octubre). Experience Using Design Patterns to Develop Reusable Object-Oriented Communication Software. *Communications of the ACM*, 38,10, pp 65-74.

Schulz B., Genssler T., Mohr B. y Zimmer W. (1998). On the Computer Aided Introduction of Design Patterns into Object Oriented Systems. Technology of the Object Oriented Languages and Systems, TOOLS 1998

Schwanninger C. (2001). Patterns as Problems Indicators. OOPSLA 2001, Workshop Beyond Design: Patterns (mis)used. Bahía Tampa, Florida, EEUU.

Seaman C.B. (1999). Qualitative Methods in Empirical Studies of Software Engineering. IEEE Transactions Software Engineering. Vol.25, No 4, pp. 557-572

Seen, M., Taylor, P. y Dick, M. (2000). Applying a Crystal Ball to Design Pattern Adoption. TOOLS Europe, 33, pp. 443-454

Shaw M. (1990). Prospects for an Engineering Discipline of Software. IEEE Software 7(6), pp.15-24.

Shaw M. y Garlan D. (1996). Software Architecture. Perspectives on an Emerging Discipline. Prentice Hall.

Shull F., Melo W. L., y Basili V. R. (1996). An Inductive Method for Discovering Design Patterns from Object-Oriented Software Systems. College Park, University of Maryland

Stroulia E. y Kapoor R. (2001). Metrics for Refactoring-based Development: An Experience Report. En Wang Y., Patel S. y Johnston R.H. (Eds.), OOIS 2001, 7th International Conference on Object-Oriented Information Systems (pp. 15-24). Universidad de Calgary, Calgary, Canadá: Springer.

Tahvildari L. y Kontogiannis K. (2002a). On the Role of Design Patterns in Quality - Driven Re - engineering. Proceedings of the Sixth European Conference on Software Maintenance and Reengineering (CSMR 02).

Tahvildari L. y Kontogiannis K. (2002b). A Software Transformation Framework for Quality-Driven Object-Oriented Re-engineering. Proceedings of the IEEE International Conference on Software Maintenance (ICSM), pp. 596-605.

Tahvildari L. y Kontogiannis K. (2004). Improving Design Quality Using Meta-Pattern Transformations: A Metric-Based Approach. Journal of Software Maintenance and Evolution: Research and Practice (JSME). John Wiley Publishers.

Tautz, C. y Von Wangenheim, C.G. (1998): REFSENO: A Representation Formalism for Software Engineering Ontologies. Fraunhofer IESE-Report No. 015.98/E, version 1.1, October 20.

Tokuda L. (1999). Evolving object-oriented Designs with Refactorings. Ph. D. Thesis, University of Texas in Austin, Department of Computer Sciences.

Tokuda L. y Batory D. (2001). Evolving Object-Oriented Designs with Refactorings. Kluwer Academic Publishers - Automated Software Engineering, vol8, Nº1, pp 89-120

Torchiano M. (2002). Documenting Patterns Use in Java Programs. Proceedings of the IEEE International Conference on Software Maintenance (ICSM'03)

Venners B. (2004). Interface Design Best Practices in Object-Oriented API Design in Java. Retrieved February 2004 from <http://www.artima.com/interfacedesign/contents.html>

Wlissides J. (1997). Patterns: The top ten misconceptions. Object Magazine, Vol. 7, No 11.

Wampler B. E. (2002). The Essence of Object-Oriented Programming with Java and UML. Addison-Wesley.

Wendorff P. (2001). Assessment of Design Patterns during Software Reengineering: Lessons Learned from a Large Commercial Project., CSMR 2001 - European Conference On Software Maintenance And Reengineering, pp 77-84

Wilkinson H. A. y Prieto M. A. (2001). Making Patterns Transparent to You. OOPSLA 2001, Workshop Beyond Design: Patterns (mis)used. Bahía Tampa, Florida, EEUU.

Wohlin C. (2000). Experimentation in Software Engineering – An Introduction published. Kluwer Academic Publishers.

Yacoub S. M. y Ammar H. (2000). Pattern – Oriented Análisis and Design (POAD): A Structural Composition Approach to Glue Design Patterns. En TOOLS 2000

Yap, L. M. y Henderson-Sellers B. (1993). A Semantic Model for Inheritance in Object-Oriented Systems. Procedins. ASWEC'93, IREE, Sydney, pp. 28–35.

Zimmer W. (1995). Relationships between design Patterns. En Coplien J. y Schmidt D. C. (Eds.), PLOP, Pattern Languajes of Program Design. Addison Wesley.

Acrónimos

BACOO	Base de Conocimiento OO
BBDD	Base de Datos
CASE	Computer-Aided Software Engineering
HTML	HyperText Markup Language
IEEE	Institute of Electrical and Electronics Engineers
IGU	Interfaz Gráfica de Usuario
ISO	International Organization for Standardization
OO	Orientado a Objetos / Orientadas a Objetos
OOPSLA	Object Oriented Programming Systems Languages and Applications
UML	Unified Modeling Language

