5.- Calidad de Modelos UML/OCL combinados.

Instructor:
Mg. Luis Reynoso, Universidad del Comahue. l_reynoso@hotmail.com

Destinatarios: Alumnos de grado y graduados.
Requisitos para los destinatarios: Los alumnos deberán contar con conocimientos de UML (especialmente diagrama de clases, diagramas de transición de estados y diagramas de colaboraciones).

Cupo: sin restricción de cupos.

Equipamiento requerido: computadora portátil, cañón, puntero láser, fotocopias del material utilizado por los alumnos que incluye slides utilizadas y fotocopias de un documento sobre los contenidos del curso (aproximadamente 110 hojas).

Idioma: castellano.

Requisitos para la aprobación.
Para la aprobación del curso los asistentes deberán aprobar una ejercitación, la cual es parte de la ejecución de un experimento controlado donde se medirá la entendibilidad y modificabilidad de expresiones OCL en modelos UML/OCL combinados.
La ejercitación que realizan los alumnos ha sido minuciosamente controlada, debido a que la misma es parte de un experimento.

Breve descripción de la importancia de modelos UML/OCL combinados
En los últimos años, los desarrolladores de software han prestado una atención especial a garantizar las características de calidad de los sistemas orientados a objetos (OO) desde las etapas iniciales de su ciclo de vida, centrándose especialmente en la calidad de los modelos conceptuales. A pesar de que la fase de modelado conceptual representa solamente una pequeña porción del esfuerzo global del desarrollo del sistema, el impacto sobre el sistema que finalmente se implementa es probablemente mayor que el de cualquier otra fase.
Recientemente, algunos paradigmas como el Desarrollo Dirigido por Modelos (*Model-Driven Development*) y la Arquitectura Dirigida por Modelos (*Model-Driven Architecture*) consideran que los modelos independientes de la plataforma son la espina dorsal del desarrollo de los sistemas OO, recalcando la importancia de construir "buenos" modelos conceptuales. La utilización de métricas para modelos constituyen los indicadores más tempranos que garantizan que las propiedades estructurales de los modelos UML no impactarán de manera significativa en aquellos atributos de calidad externa que se tratan de preservar en un modelo (comprensibilidad, modificabilidad, etc.)
Por otro lado, si bien la aparición de UML (Unified Modeling Language) como lenguaje de modelamiento ha supuesto un gran avance en pro de construir modelos de mayor calidad, muchas de las restricciones del sistema que se necesitan modelar no pueden expresarse utilizando solamente las notaciones gráficas que proporciona UML. Estas restricciones son escritas comúnmente en el lenguaje OCL (Object Constraint Language) durante las etapas iniciales del desarrollo. Con OCL se especifican expresiones precisas que se asocian a los elementos básicos de los modelos UML. La combinación de los lenguajes UML y OCL nos permite obtener modelos coherentes y completos, independientes de la plataforma y con un alto nivel de madurez. Además, OCL facilita la documentación del sistema, mejora la comunicación entre ingenieros de software (evitando errores producidos por malas interpretaciones) y hace que la expresividad del sistema en etapas iniciales sea mayor.
La utilidad de OCL como elemento esencial de la calidad de modelos es ejemplificada en dos niveles de modelamiento, nivel de modelos (M1) y nivel de metamodelos (M2). Se enseña a los alumnos a definir restricciones semánticas y expresiones de consulta en ambos niveles. Por otro lado, se ejemplifica la utilidad de OCL para definir formalmente métricas para modelos UML.
En este curso, los ejes principales serán:

- **Calidad de Modelos y Métricas:** Se brinda una introducción definiendo los conceptos de Modelos, Niveles de metamodelos de UML (M0, M1, M2 y M3), y Niveles de Madurez de

Modelamiento (MML). Se introduce a los alumnos en características de Calidad de Modelos. Atributos de calidad interna y externa. Definición metodológica de métricas para modelos. Se describen los principales fases de un Marco Metodológico para la definición de métricas: Identificación, Creación, y sus sub-fases: Definición, Validación Teórica y Validación Empírica, Definición formal de Métricas.

- **Métricas para Modelos UML:** Se presenta un estado del arte de las métricas existentes para modelos UML, para diagramas de clase, diagrama de estados, diagramas de casos de uso, y expresiones OCL.
- **El Lenguaje de Restricción de Objetos (OCL) como elemento esencial para complementar Modelos:** Se presenta los conceptos esenciales de OCL, poniendo énfasis en la combinación de propiedades OCL para demostrar la expresividad del lenguaje. Se ejemplifica como OCL permite definir restricciones sobre los elementos del modelo para aumentar la calidad de los mismos.
- **Definiendo restricciones semánticas en modelos M1:** Se muestra como el lenguaje OCL se puede utilizar complementando restricciones semánticas y consultas en nivel M1, para distintos tipos de diagramas UML: diagramas de clase, diagramas de estados, etc. Coherencia y consistencia de modelos UML.
- **Definiendo restricciones semánticas en modelos M2:** Se muestra como el lenguaje OCL se puede utilizar complementando restricciones semánticas y consultas en nivel M2. Se presenta un caso de Estudio, el metamodelo de diagramas de Estados. Se ejemplifican instanciaciones del metamodelo de diagramas de estados. Se muestra la definición formal de métricas para diagramas de Estados utilizando el lenguaje OCL.

Como requisito de aprobación del curso los alumnos participarán en un experimento controlado, el cual consistirá de una serie de ejercicios sobre la entendibilidad y modificabilidad de expresiones OCL en modelos UML/OCL combinados. El experimento que se realizará es parte de familia de experimentos que se han ejecutado previamente en la UCLM (Universidad de Castilla La Mancha, España), UA (Universidad de Alicante, España), UNC (Universidad Nacional del Comahue, Argentina), UACh (Universidad Austral de Chile).

Contenido a desarrollar como parte del curso

Calidad de Modelos.

Definición de Modelos. Niveles de Modelos y de Madurez. Importancia de los modelos en el proceso Model-Driven Architecture (MDA). Calidad de Modelos. Estándares Internacionales. Medidas de Calidad. Marco Metodológico para la definición de métricas: Identificación, Creación, y sus sub-fases: Definición, Validación Teórica y Validación Empírica, Definición formal de Métricas.

Métricas para Modelos UML. Se presenta un estado del arte de las métricas existentes para modelos UML, para diagramas de clase, diagrama de estados, diagramas de casos de uso, y expresiones OCL.

El Lenguaje de Restricción de Objetos (OCL) como elemento esencial para complementar Modelos. Importancia de su utilización. Usos de OCL. OCL en el proceso MDA. Calidad en Modelos Conceptuales Orientados a Objetos. Niveles de madurez de Modelos (MMLs). Diseño por Contrato. Ventajas. Modelos UML/OCL combinados. Usos de OCL. Modelos UML/OCL combinados. Conceptos básicos. Tipos de expresiones en OCL. Propiedades en OCL. Colecciones. La utilización de OCL en diagramas de clases y diagramas de transición de estados. Claves para la implementación de modelos UML/OCL combinados.

Conceptos Básicos de OCL. La instancia contextual. Principales tipos de expresiones. Invariantes. Pre- y Post- Condiciones. Navegaciones Simples y combinadas. Colecciones Predefinidas en el lenguaje OCL. Conformidad de Tipos. Reglas de Precedencia. Valores Indefinidos.

Otros tipos de Expresiones OCL: Atributos/Operaciones adicionales definidos a partir de expresiones <<definition>>. Expresiones “Let”. Expresiones de consulta. Valores iniciales y Derivados. Reglas de derivación. Operaciones Query.

Objetos y Propiedades. Atributos como propiedades. Operaciones como Propiedades. Definiendo Operaciones. Extremos de Asociación como Propiedades. Navegaciones.

Navegación a Clases de Asociación. Navegación de Asociaciones con multiplicidad cero o mas. Combinando propiedades. Propiedades predefinidas en todos los objetos.

Colecciones. Colecciones de colecciones. Jerarquía de Colecciones. Reglas de Conformidad de tipos. Valores previos en Postcondiciones. Forma general de especificar colecciones. Operaciones de Colección: Collect, Forma resumida de Collect, Select, Reject, ForAll, Exists, Iterate. Otras operaciones de Colección.

Completando Modelos UML (nivel M1)

Ciclos en Modelos de Clase. Multiplicidad Dinámica. Multiplicidad Opcional. Restricciones Or. Completando Diagramas de Componentes. Completando diagramas de Interacción. Instancias. Condiciones. Valores de los parámetros actuales. Completando Diagramas de Estados (StateCharts). Ejemplo de un diagrama de Control de Procesos simple. Guardas. Acciones. Valores de Parámetros actuales.

Evento de cambio. Restricciones en Estados. Claves para la especificación de expresiones OCL. Evitar expresiones de navegación complejas. Elegir adecuadamente el contexto. Dividir restricciones and. Nombrar siempre los extremos de asociación

Recomendaciones para la especificación correcta de expresiones OCL. Coherencia y consistencia de modelos UML.

Restricciones y Consultas en Metamodelos (nivel M2)

Niveles de Especificación de Modelos. Metamodelos de UML y OCL. Ejemplos de Instanciacion. Utilizando el metamodelo de Diagramas de Estados, se ejemplifica como puede ser utilizado OCL para especificar la definición de métricas para diagramas de Estados.

Claves para la Implementación de modelos UML/OCL combinados. Implementación de los elementos el modelo UML/OCL: clases, atributos, asociaciones, otros elementos (representación de tipos enumerados, interfaces, estados de diagramas de estados). Implementación de una librería estándar. Implementación de expresiones (acceso a propiedades, reglas de derivación, etc). Ubicando los fragmentos de código de las expresiones en la implementación de los elementos del modelo: expresiones invariantes, pre- y postcondiciones.

Bibliografía:

1. B. Anda, H. Dreiem, D. I. K. Sjoberg, and M. Jorgensen. Estimating Software Development Effort Based on Use Cases-Experiences from Industry. In UML '01: Proceedings of the 4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools, pages 487_502, London, UK, 2001. Springer-Verlag.
2. Alexander. Visualising Requirements in UML. Telelogic Newsbyte. Available in http://easyweb.easynet.co.uk/iany/consultancy/ reqts_in_uml/reqts_in_uml.htm., 2001.
3. E. Arisholm. Dynamic Coupling Measures for Object-Oriented Software. In METRICS '02: Proceedings of the 8th International Symposium on Software Metrics, page 33, Washington, DC, USA, 2002. IEEE Computer Society.
4. L. C. Briand, E. Arisholm, S. Counsell, F. Houdek. Empirical Studies of Object-Oriented Artifacts, Methods, and Processes: State of the Art and Future Directions. Empirical Software Engineering: An International Journal, 4(4):385_402, December 1999.
5. L. C. Briand, C. Bunse, and J. W. Daly. A Controlled Experiment for Evaluating Quality Guidelines on the Maintainability of Object-Oriented Designs. IEEE Transaction on Software Engineering, 27(6):513_530, 2001.
6. L. C. Briand, C. Bunse, and J. W. Daly. A Controlled Experiment for Evaluating Quality Guidelines on the Maintainability of Object-Oriented Designs. Software Engineering, 27(6):513_530, 2001.
7. V. R. Basili, L. C. Briand, and W. L. Melo. Validation of Object-Oriented Design Metrics as Quality Indicators. IEEE Transaction on Software Engineering, 22(10):751_761, 1996.
8. J. Bansiya and C. G. Davis. A Hierarchical Model for Object-Oriented Design Quality Assessment. IEEE Transaction on Software Engineering, 28(1):4_17, 2002.
9. B. Bernandez, A. Duran, and M. Genero. An Empirical Review of Use Cases Metrics for Requirements Veri_cation. Proceedings of the Software Measurement European Forum (SMEF'04)., 2004.
10. L. C. Briand, P. Devanbu, and W. Melo. An Investigation into Coupling Measures for C++. Proceedings of the 19th international conference on Software engineering, pages 412_421, 1997.
11. L. C. Briand, J. W. Daly, and J. W. ust. A Uni_ed Framework for Coupling Measurement in Object-Oriented Systems. Fraunhofer Institute for Experimental Software Engineering, 1996.

12. L. C. Briand, J. W. Daly, and J. W. ust. A Uni_ed Framework for Cohesion Measurement in Object-Oriented Systems. Empirical Software Engineering: An International Journal, 3(1):65_117, 1998.
13. L. Baroni and F. Brito e Abreu. A Formal Library for Aiding Metrics Extraction. International Workshop on Object-Oriented Re-Engineering at ECOOP'2003. Darmstadt, Germany., 2003.
14. J. Bansiya, L. Etzkorn, C. Davis, and W. Li. A Class Cohesion Metric For Object-Oriented Designs. The Journal of Object-Oriented Programming, 11(8):47_52, 1999.
15. L. C. Briand, S. Ikonomovski, H. Lounis, and J. Wust. A Comprehensive Investigation of Quality Factors in Object-Oriented Designs: an Industrial Case Study, 1998.
16. H. Baumeister, A. Knapp, and M. Wirsing. Property-Driven Development. Second International Conference on Software Engineering and Formal Methods, pages 96_102, 2004.
17. L. C. Briand, Y. Labiche, and Y. Miao. Towards the Reverse Engineering of UML Sequence Diagrams. In WCRE '03: Proceedings of the 10th Working Conference on Reverse Engineering, page 57, Washington, DC, USA, 2003. IEEE Computer Society.
18. L. C. Briand, S. Morasca, and V. R. Basili. Property-Based Software Engineering Measurement. IEEE Transaction on Software Engeneering, 22(1):68_86, 1996.
19. L. C. Briand, S. Morasca, and V. R. Basili. Property-Based Software Engineering Measurement. IEEE Transactions on Software Engineering, 22(1):68_86, 1996.
20. L. C. Briand, S. Morasca, and V. R. Basili. Response to: Comments on "Property-Based Software Engineering Measurement: Refining the Additivity Properties". IEEE Transactions on Software Engineering, 23(3):196_197, 1997.
21. B. Binkley and S. R. Schach. Validation of the Coupling Dependency Metric as a Predictor of Run-Time Failures and Maintenance Measures. ICSE '98: Proceedings of the 20th international conference on Software engineering, pages 452_455, 1998.
22. R. K. Bandi, V. K. Vaishnavi, and D. E. Turk. Predicting Maintenance Performance Using Object-Oriented Design Complexity Metrics. IEEE Transaction on Software Engineering, 29(1):77_87, 2003.
23. L. C. Briand and J. W st. Modeling Development E_ort in Object-Oriented Systems Using Design Properties. IEEE Transactions on Software Engineering, 27(11):963_986, 2001.
24. L. C. Briand, J. W. ust, J. W. Daly, and D. V. Porter. Exploring the Relationships between Design Measures and Software Quality in Object-Oriented Systems. The Journal of Systems and Software, 51(3):245_273, 2000.
25. L. C. Briand, J. Wust, and H. Lounis. Using Coupling Measurement for Impact Analysis in Object-Oriented Systems. ICSM '99: Proceedings of the IEEE International Conference on Software Maintenance, pp. 475, 1999.
26. L. C. Briand, J. W st, and H. Lounis:. Replicated Case Studies for Investigating Quality Factors in Object-Oriented Designs. Empirical Software Engineering., 6(1):11_58, 2001.
27. L. C. Briand, J. Wust, and H. Lounis. Replicated Case Studies for Investigating Quality Factors in Object-Oriented Designs. Empirical Softw. Engg., 6(1):11_58, 2001.
28. M. Cartwright. An Empirical View of Inheritance. 1998. S. R. Chidamber, D. P. Darcy, and C. F. Kemerer. Managerial Use of Metrics for Object-Oriented Software: An Exploratory Analysis. IEEE Transactions on Software Engineering, 24(8):629_639, 1998.
29. H. Christian. OCL-Constraints for UMM Business Collaborations.
30. S. R. Chidamber and C. F. Kemerer. Towards a Metrics Suite for Object Oriented Design. Object Oriented Programming: Systems, Languages and Application. OOSPLA '91., pages 197_211, 1991.
31. S. Chidamber and C. Kemerer. A Metrics Suite for Object Oriented Design. IEEE Transactions on Software Engineering, 20(6):476_493, 1994.
32. S. R. Chidamber and C. F. Kemerer. A Metrics Suite for Object Oriented Design. IEEE Transactions on Software Engineering, 20(6):476-493, 1994.
33. S. Cook, A. Kleepe, R. Mitchell, B. Rumpe, J. Warmer, and A. Wills. The Amsterdam Manifiesto on OCL. Tony Clark and Jos Warmer, editors, Advances in Object Modelling with the OCL, pages 115_149, 2001.
34. M. Carbone and G. Santucci. Fast and Serious: a UML Based Metric for Effort Estimation. 6th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2002)., pages 35_44, 2002.
35. L. Correa and C. M. L. Werner. Precise Specification and Validation
36. of Transactional Business Software. Proceedings. 12th IEEE International Requirements Engineering Conference, pages 16_25, 2004.

37. M. Casanova, T. Wallet, and M. D'Hondt. Adaptations to OCL for Ensuring Quality of Geographic Data (Poster Session). OOPSLA '00: Addendum to the 2000 proceedings of the conference on Object-oriented programming, systems, languages, and applications (Addendum), pages 69_70, 2000.
38. J. Daly, A. Brooks, J. Miller, M. Roper, and M. Wood. An Empirical Study Evaluating Depth of Inheritance on the Maintainability of Object-Oriented Software. 1996.
39. K. Derr. Applying OMT. SIGS Books, 1995. M. Dagpinar and J. H. Jahnke. Predicting Maintainability with Object-Oriented Metrics - An Empirical Comparison. 10th Working Conference on Reverse Engineering, pages 155_164, 2003.
40. F. Brito e Abreu and R. Carapu a. Object-Oriented Software Engineering: Measuring and Controlling the Development Process. Proceedings of 4th Int Conference on Software Quality, Mc Lean, VA, USA., pages 3_5, 1994.
41. F. Brito e Abreu, R. Esteves, and M. Goulao. The Design of Eiffel Programs: Quantitative Evaluation Using the MOOD Metrics. Proceedings of TOOLS USA 96 (Technology of Object Oriented Languages and Systems), Santa Barbara, California, USA., 1996.
42. F. Brito e Abreu, M. Goulao, and R. Esteves. Towards the Design Quality Evaluation of Object-Oriented Software System. Proceedings of the 5Th International Conference on Software Quality, Austin, Texas, USA., 1995.
43. F. Brito e Abreu and W. Melo. Evaluating the Impact of Object-Oriented Design on Software Quality. 3rd International Metric Symposium., 1996.
44. K. Emam, S. Benlarbi, N. Goel, and S. Rai. A Validation of Object-Oriented Metrics. 1999.
45. J. Eder, G. Kappel, and M. Schre . Coupling and Cohesion in object-oriented systems. Technical Report, University of Klagenfurt, 1994.
46. K. El Emam. Object-Oriented Metrics: A Review of Theory and Practice. pages 23_50, 2002.
47. Friis-Christensen, N. Tryfona, and C. S. Jensen. Requirements and Research Issues in Geographic Data Modeling. GIS '01: Proceedings of the 9th ACM international symposium on Advances in geographic information systems, pages 2-8, 2001.
48. P. Feldt. Requirements Metrics Based on Use Cases. Master's Thesis. Department of Communication Systems, Lund Institute of Technology, Lund University, Box 118, S-221 00 Lund, Sweden., 2000.
49. S. Flake. Real-time Constraints with the OCL. Proceedings of the Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing. (ISORC 2002), pages 425_426, 2002.
50. S. Flake and W. Mueller. Speci_cation of Real-Time Properties for UML Models. Proceedings of the 35th Annual Hawaii International Conference on System Sciences, pages 3977_3986, 2002.
51. M. Fowler and K. Scott. UML Distilled. Second Edition. Addison-Wesley, 2000.
52. Glasberg, K. El-Emam, W. Melo, and N. Madhavji. Validating Object-Oriented Design Metrics on a Commercial Java Application. Technical Report, National Research Council of Canada, NRC/ERB- 1080, 2000.
53. R. Gronmo and J. Oldevik. An Empirical Study of the UML Model Transformation Tool (UMT). The First International Conference on Interoperability of Enterprise Software and Applications (INTEROP-ESA), 2005.
54. M. Genero, M. Piattini, and C. Calero. Early Measures for UML Class Diagrams. L Objet, 6(4):495_515, 2000.
55. M. Genero, M. Piattini, and M. Calero (Eds.). Metrics For Software Conceptual Models. 2005.
56. Hamie. Enhancing the Object Constraint Language for More Expressive Speci_cations. Software Engineering Conference APSEC 99. Proceedings. Sixth Asia Paci_c, pages 376_383, 1999.
57. Hamie. Translating the Object Constraint Language into the Java Modelling Language. SAC '04: Proceedings of the 2004 ACM symposium on Applied computing, pages 1531_1535, 2004.
58. R. Harrison, S. Counsell, and R. Nithi. Coupling Metrics for Object-Oriented Design. 5th International Software Metrics Symposium Metrics, pages 150_156, 1998.
59. R. Harrison, S. J. Counsell, and R. V. Nithi. An Evaluation of the MOOD Set of Object-Oriented Software Metrics. IEEE Transaction on Software Engineering, 24(6):491_496, 1998.
60. R. Harrison, S. Counsell, and R. Nithi. Experimental Assessment of the Effect of Inheritance on the Maintainability of Object-Oriented Systems. The Journal of Systems and Software, 52(2_3):173_179, 2000.
61. M. Hitz and B. Montazeri. Measuring Coupling and Cohesion in Object-Oriented Systems. Int. Symposium on Applied Corporate Computing, 1995.

62. Henderson-Sellers, D. Zowghi, T. Klemola, and S. Parasuram. Sizing Use Cases: How to Create a Standard Metrical Approach. 8th Object-Oriented Information Systems, Lecture Notes in Computer Science, 2425., pages 409_421, 2002.
63. H. Hussmann and S. Zschaler. The Object Constraint Language for UML 2.0 Overview and ASsessment. Upgrade, Vol. V, issue no. 2, pages 25_28, 2004.
64. P. In, S. Kim, and M. Barry. UML-Based Object-Oriented Metrics for Architecture Complexity Analysis. Proceedings of Ground System Architectures Workshop (GSAW 03), El Segundo, CA, The Aerospace Corporation., 2003.
65. S. R. Judson, D. L. Carver, and R. B. France. A Metamodeling Approach to Model Transformation. OOPSLA '03: Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, pages 326_327, 2003.
66. Jacobson, M. Christerson, P. Jonsson, and G. Overgaard. Object-Oriented Software Engineering: A Use Case Driven Approach. Addison Wesley, 1992.
67. Kang. A Complexity Measure for Ontology Based on UML. FTDCS '04: Proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS'04), pages 222_228, 2004.
68. G. Karner. Metrics for objectory. Master Thesis, Linkuping University, Linkuping, 1993.
69. H. Kim and C. Boldyre_. Developing Software Metrics Applicable to UML Models. Proceedings of the 6th International Workshop on Quantitative Approaches in Object-Oriented Software Engineering. Malaga, Spain, pages 67_76, 2002.
70. Lake and C. Cook. Use a Factor Analysis to Develop OOP Software Complexity Metrics. Sixth Ann. Oregon Workshop Software Metrics, 1994.
71. J. A. C. Lemus, M. Genero, J. A. Olivas, F. P. Romero, and M. Piattini. Predicting UML Statechart Diagrams Understandability Using Fuzzy Logic-Based Techniques. Proceedings of the Sixteenth International Conference on Software Engineering & Knowledge Engineering (SEKE'2004), Alberta, Canada, June 20-24, 2004, pages 238_245, 2004.
72. J. A. C. Lemus, M. Genero, J. A. Olivas, F. P. Romero, and M. Piattini. An Empirical Study of the Nesting Level of Composite States Within UML Statechar Diagrams. Proceedings of First International Workshop on Best Practices of UML, BP-UML 2005, ER (Workshops), pages 12_22, 2005.
73. J. A. C. Lemus, M. Genero, J. A. Olivas, F. P. Romero, and M. Piattini. Evaluating the E_ect of Composite States on the Understandability of UML States on Understandability of UML Statechard Diagrams. Proceedings of the MODELS, pages 113_125, 2005.
74. J. A. C. Lemus, M. Genero, and M. Piattini. Metrics for UML Statechart Diagrams. Metrics for Software Conceptual Models. Genero, Piattini and Calero (eds.), 2005.
75. W. Li and S. Henry. Object-Oriented Metrics that Predict Maintainability. Journal of Systems and Software, 23(2):111_122, 1993.
76. M. Lorenz and J. Kidd. Object-Oriented Software Metrics: A Practical Guide. 1994.
77. Y. S. Lee, B. S. Liang, S. F. Wu, and F. J. Wang. Measuring the Coupling and Cohesion of an Object Oriented Program Based on Information Flow. International Conference Software Quality, 1995.
78. M. Marchesi. OOA Metrics for the Uni_ed Modeling Language. 2nd Euromicro Conference on Software Maintenance and Reengineering, pages 67_73, 1998.
79. G. Miller, A. Evans, I. Jacobson, H. Jondell, A. Kennedy, S. Mellor, and D. Thomas. Model Driven Architecture: How Far Have We Come, How Far Can We Go? OOPSLA '03: Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications, pages 273_274, 2003.
80. Miranda, M. Genero, and M. Piattini. Empirical Validation of Metrics for UML Statechart Diagrams. Fifth International Conference on Enterprise Information Systems (ICEIS 03), pages 87_95, 2003.
81. S. Muthanna, K. Ponnambalam, K. Kontogiannis, and B. Stacey. A Maintainability Model for Industrial Software Systems Using Design Level Metrics. In WCRE '00: Proceedings of the Seventh Working Conference on Reverse Engineering (WCRE'00), page 248, Washington, DC, USA, 2000. IEEE Computer Society.
82. N. Medvidovic, D. S. Rosenblum, D. F. Redmiles, and J. E. Robbins. Modeling Software Architectures in the Uni_ed Modeling Language. ACM Trans. Softw. Eng. Methodol., 11(1):2_57, 2002.
83. Muthiayen. Real Time Reactive System Development - A Formal Approach Based on UML and PVS. Ph.D Thesis, Deparment of Computer Science at Concordia University, Montreal, Canada.,

2000. [MV04] B. A. Malloy and J. M. Voas. Programming with Assertions: a Prospectus. IT Professional, 6(5):53_59, 2004.
84. Nebut, F. Fleurey, Y. Le Traon, and J. Jezequel. Automatic Test Generation: A Use Case Driven Approach. volume 32, pages 140_155, 2006.
85. Object Management Group OMG. UML 2.0 OCL Final Adopted Specification. OMG Document ad/2003-01-07, 2003.
86. Object Management Group OMG. UML Speci_cation. OMG Document formal/03-03-01, 2003.
87. Object Management Group OMG. MOF QVT Final Adopted Specification. OMG Document ptc/05-11-01, 2005.
88. Object Management Group OMG. UML 2.0 OCL Available Specification (FTF Report). OMG Document ptc/2005-06-06, 2005.
89. Software Solutions on Time. A Fresh and Innovative Approach to Systems Development and Software Project Management. Available in http://www.tassc-solutions.com/omx/pages/metric_data.htm#usecase-metrics., 2001.
90. O. Patrascoiu. YATL: Yet Another Transformation Language. First European Workshop on Model Driven Architecture with Emphasis on Industrial Application. University of Twente, Enschede, the Netherlands., 2004.
91. Poels and G. Dedene. Distance: A Framework for Software Measure Construction. Research Report DTEW9937, Dept. Applied Economics, Katholieke Universiteit Leuven, Belgium, 46 p.
92. Poels and G. Dedene. Evaluating the E_ect of Inheritance on the Modi_ability of Object-Oriented Business Domain Models. Fifth European Conference on Software Maintenance and Reengineering, pages 20_29, 2001.
93. Poels. On the Formal Aspects of the Measurement of Object- Oriented Software Speci_cations. Ph.D. Thesis. Faculty of Economics and Business Administration. Katholieke Universiteit Leuven, Belgium., 1999.
94. M. Richters. A Precise Approach to Validating UML Models and OCL Constraints. Monographs of the Bremen Institute of Safe Systems, 2001.
95. Roussev. Generating OCL Speci_cations and Class Diagrams from Use Cases: A Newtonian Approach. In HICSS '03: Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 9, page 321.2, Washington, DC, USA, 2003. IEEE Computer Society.
96. M. Saeki. Embedding Metrics into Information System Development Methods: An Application of Method Engineering Technique. Lecture Notes in Computer Science 2681, pages 374_389, 2003.
97. L. Sourrouille and G. Caplat. Constraint Checking in UML Modeling. SEKE '02: Proceedings of the 14th international conference on Software engineering and knowledge engineering, pages 217_224, 2002.
98. S. Sendall. Supporting Model-to-Model Transformations: The VMT Approach. Workshop on Model Driven Architecture: Foundations and Applications. Proceedings published in Technical Report TR-CTIT-03- 27. University of Twente, The Netherlands., 2003.
99. R. Subramanyam and M. S. Krishnan. Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects. IEEE Transaction Software Engineering, 29(4):297_310, 2003.
100. W. Stevens, G. Myers, and L. Constantine. Structured Design. IBM Systems Journal, 13(2):115_139, 1974.
101. Smith. The Estimation of E_ort based on Use Cases. (Rational Software white paper). Rational Software.Available in http://www.rational.com/media/whitepapers/_nalTP171.PDF., 1999.
102. M. Satpathy, N. T. Siebel, and D. Rodriguez. Assertions in Object Oriented Software Maintenance: Analysis and Case Study. Proceedings. 20th IEEE International Conference on Software Maintenance, pages 124_133, 2004.
103. G. Schneider and J.P. Winters. Applying Use Cases: A Practical Guide. Addison Wesley, 1998.
104. Schleicher and B. Westfechtel. Beyond Stereotyping: Metamodeling Approaches for the UML. 34th Annual Hawaii International Conference on System Sciences, 3:3051_3052, 2001.
105. Tchertchago. Analysis of the Metamodeling Semantics for OCL. Masther Thesis. Department of Computer Science. Dresden University of Technology, 2002.
106. M. Tang, M. Kao, and M. Chen. An Empirical Study on Object-Oriented Metrics. Sixth IEEE International Symposium on Software Metrics, pages 242_249, 1999.
107. Unger, L. Prechelt, and M. Philippsen. The Impact of Inheritance Depth on Maintenance Tasks: Detailed Description and Evaluation of Two Experiment Replications. (Technical Report, Karlsruhe University: Karlsruhe, Germany.), 1998.

108. Verheecke and R. Van Der Straeten. Specifying and Implementing the Operational Use of Constraints in Object-Oriented Applications. CRPITS '02: Proceedings of the Fortieth International Conference on Tools Pacific, pages 23_32, 2002.
109. X. Wang, C. W. Chan, and H. J. Hamilton. Design of Knowledge- based Systems with the Ontology-domain-system Approach. SEKE '02: Proceedings of the 14th international conference on Software engineering and knowledge engineering, pages 233_236, 2002.
110. Warmer and A. Kleppe. The Object Constraint Language. Precise Modeling with UML. Object Technology Series. Addison Wesley, 1999.
111. J. Warmer and A. Kleppe. The Object Constraint Language. Second Edition. Getting Your Models Ready for MDA. 2003.
112. H. Y. Yang, E. Tempero, and R. Berrigan. Detecting Indirect Coupling. Australian Software Engineering Conference ASWEC 2005, pages 212_221, 2005.