MARCO DE REFERENCIA PARA LA GESTIÓN DE LA CALIDAD DE LAS ESPECIFICACIONES DE REQUISITOS

María N. Moreno García*, Francisco J. García Peñalvo, M. José Polo Martín, Vivian López Batista y Angélica González Arrieta
Universidad de Salamanca. Departamento de Informática y Automática.
Teléfono: 34-923-294653, Fax: 34-923-294514, *e-mail: mmg@gugu.usal.es

Resumen

El presente trabajo se enmarca, dentro del campo de la medición del software, en el ámbito de la especificación de requisitos. La mayor parte de la investigación realizada en esta parcela de la calidad se centra en el estudio de atributos estructurales, que casi siempre se evalúan en fases finales del desarrollo, descuidando otras perspectivas del modelado de sistemas. Por otra parte, son muy escasos los estudios existentes sobre la interrelación de diferentes métodos y aspectos de medida y sobre la formalización de dicho proceso cuando se realiza al comienzo del ciclo de vida. En este artículo se analiza el papel que juegan las mediciones iniciales en la determinación y predicción de las características de calidad del software y se propone un marco de referencia que permite formalizar y automatizar el proceso de medición y gestionar la calidad considerando diferentes perspectivas de modelado.

Introducción

Es un hecho ampliamente aceptado que las primeras etapas del desarrollo de software son cruciales en la consecución de productos de calidad dentro de los límites de tiempo y coste establecidos para un proyecto. En este contexto, la medición del software está adquiriendo cada vez mayor importancia, debido a la necesidad de obtener datos objetivos que contribuyan a mejorar la calidad. Muchos investigadores han producido modelos de características de calidad del software que pueden ser útiles para discutir, planificar y obtener índices de calidad de los productos de software. Los modelos de calidad más recientes (CMM [29], SPICE [33] [35]), están orientados a la mejora de procesos mediante la definición de principios y prácticas que conducen a mejores productos de software y la determinación de la madurez de la capacidad en función del grado de cumplimiento de esos principios. Sin embargo, dichos modelos no sirven de referencia para evaluar la calidad del software producido a lo largo del ciclo de vida ya que, desafortunadamente, organizaciones que cumplen los requisitos CMM no están produciendo software de calidad [11]. Otros modelos incluyen métricas para evaluar de forma cuantitativa el grado de calidad de diferentes atributos del producto (FCM, GQM...) [25] [3]. Aunque la medición debería poderse aplicar a productos de cualquier nivel, no siempre se pueden realizar medidas de las características de calidad de las especificaciones debido a la ausencia de métricas específicas o a la indeterminación de las mismas. El desarrollo de métodos de medida en el ámbito de los requisitos del software está centrado fundamentalmente en la medición del tamaño y la funcionalidad del software.

Características de la ERS	Problemas de medición
Abstracción	Dificultad de medición directa de los atributos de calidad
Evolución	Necesidad de reflejar los cambios y asegurar la consistencia
Transformación	Evaluación de la trazabilidad
Diferentes perspectivas de modelado	Gestión conjunta de múltiples notaciones de modelado

Tabla 1. Características de las ERS

La dificultad en la medición de la ERS (Especificación de Requisitos del Software) se debe fundamentalmente a algunas características inherentes a las propias especificaciones (Tabla 1). El problema principal radica en el alto nivel de abstracción en el que se encuentran, lo que hace muy difícil definir y medir de forma directa y objetiva atributos de calidad. A esta dificultad habría que añadir la derivada del hecho de que los requisitos evolucionan a medida que progresa el

desarrollo, esa inestabilidad e incluso volatilidad de los requisitos requiere un proceso sistemático de obtención y almacenamiento de los datos, en el que se asegure la consistencia de los cambios en los resultados obtenidos y que permita realizar estudios comparativos sobre dicha evolución. Otro tipo de evolución sufrida por los requisitos, que repercute igualmente en la dificultad de medición, es la trasformación que sufren los modelos al ir descendiendo en el nivel de abstracción (los modelos de análisis se transforman en modelos de diseño, etc.), esto conlleva la necesidad de examinar los enlaces de trazabilidad, con lo que entrarían en juego, no sólo elementos de modelado del nivel de análisis sino también de nivel de diseño. Por otra parte, el uso de diferentes métodos de especificación con notaciones diferentes, o el uso de métodos y lenguajes actuales que permiten el modelado de los requisitos del sistema desde múltiples perspectivas, exige la definición y aplicación de diferentes métricas específicas para cada notación, si se desea realizar una valoración completa de la calidad de dichas especificaciones. En este trabajo se propone una arquitectura para la gestión de la calidad de la ERS que contribuye a solventar los problemas comentados anteriormente. Dicha arquitectura permite:

- Gestionar conjuntamente la calidad de diferentes perspectivas de modelado
- Formular directamente objetivos de calidad y planes de medida
- Proporcionar una base para la automatización de las medidas
- Mantener registros de información histórica
- Proporcionar soporte para estudios empíricos y construcción de modelos predictivos

Trabajos relacionados

La mayoría de los trabajos relacionados con la medición en el nivel de la especificación de requisitos se han centrado fundamentalmente en el desarrollo de métricas para determinar el tamaño y la funcionalidad del software. Entre las de mayor difusión se encuentran las métricas de puntos de función [1], métricas Bang [13] o los puntos objeto [5].
La medición de atributos de calidad de las especificaciones ha sido objeto de algunos trabajos que van desde la medición de especificaciones formales [34] hasta la aplicación de métricas para evaluar la calidad de especificaciones expresadas informalmente en lenguaje natural. En este último ámbito pueden utilizarse algunas métricas de calidad de la documentación como las métricas de facilidad de comprensión del texto contenido en los documentos [23] o métricas de estructura y organización en documentos convencionales [2] y con hipertexto [15] [32]. Otras técnicas, como las propuestas por Davis y colaboradores [12] o el uso de listas de comprobación [8] [14], realizan la valoración de los atributos de calidad de la especificación mediante métricas que requieren información procedente de revisiones técnicas, inspecciones, *Walkthrough*, o auditorías encaminadas a determinar el cumplimiento de los estándares, directrices, especificaciones y procedimientos. Los resultados obtenidos en este tipo de evaluaciones tienen un alto componente subjetivo y son claramente dependientes de las personas que los realizan, aun fijando previamente criterios objetivos.
La creciente adopción de la tecnología de orientación a objetos en el desarrollo de software ha dado lugar a la aparición de nuevas métricas específicas para este tipo de sistemas. Como más representativas se pueden citar las métricas de diseño [9], métricas orientadas a clases [24], métricas orientadas a operaciones [10] o las métricas para pruebas [4]. Recientemente se han propuesto métricas para la evaluación de la calidad a partir de modelos producidos en etapas iniciales del ciclo de vida, como son las métricas de calidad y complejidad en modelos OMT [16] o las métricas de calidad de los diagramas de clases en UML [17] mediante las cuales se evalúa la complejidad introducida por las jerarquías de generalización en los diagramas de clases obtenidos en las etapas de análisis y diseño. Asimismo han surgido intentos de realizar evaluaciones de la calidad a partir de modelos dinámicos, tal es el caso del trabajo de Poels en el que se realiza la medición de modelos conceptuales basados en eventos [31].
Del análisis de la bibliografía reseñada se puede concluir que las métricas de calidad para sistemas orientados a objetos se caracterizan por estar centradas principalmente en el diseño y más concretamente en el modelado estructural o estático, limitándose únicamente a evaluar la complejidad, reusabilidad, acoplamiento o cohesión, sin tener en cuenta otros atributos de calidad que deben exhibir las especificaciones de requisitos del software.
Por otra parte, la necesidad de medir diferentes aspectos del software y la proliferación de métricas surgidas debido a la creciente importancia que está adquiriendo la medición está contribuyendo a crear confusión sobre las relaciones entre tales medidas, así como sobre su forma y ámbito de aplicación. Este hecho ha conducido la búsqueda de nuevos caminos en la investigación que se orientan hacia

la propuesta de modelos y marcos de referencia ("framewors") que permitan la organización de las medidas y la clasificación de las entidades de software a través de un conjunto de dimensiones que se usan para identificar sistemas, modelos, atributos y objetos susceptibles de medir [30] [7].

Medidas basadas en modelos

La medición efectiva del software y la interpretación significativa de los datos depende del reconocimiento de la dualidad esencial del proceso de medida. La medición implica la definición de dos modelos:

- El contexto empírico del mundo real, en el que tiene lugar la medición Un modelo numérico que incorpora aspectos basados en la medición y bien definidos del modelo empírico [28]. La teoría de la medición implica la definición de interrelaciones formales entre los dos modelos. El proceso completo de medición se muestra en la figura 1.

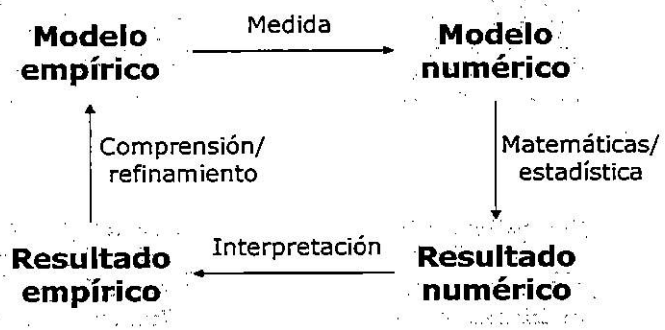


Figura 1. Modelos implicados en el proceso de medida

Las interrelaciones esenciales entre los modelos numérico y empírico subyacentes en el proceso de medida requiere un marco de unificación, o un metamodelo, para razonar sobre dichos modelos y sus interrelaciones. La figura 2 ilustra una jerarquía genérica de modelos. Cualquier modelo empírico de medida relacionado con el software se puede considerar construido a partir de fragmentos de producto y/o de proceso (submodelos) i,j,k. El modelo numérico formal correspondiente se representa por el modelo' y los fragmentos del modelo. Si se tiene que razonar sobre la efectividad y evolución de los modelos de medida, se puede definir un metamodelo para construir de forma razonada y modificar los modelos de medida empíricos y numéricos. Sin embargo, esto requiere un meta-metamodelo para definir la sintaxis y la semántica del metamodelo (por ejemplo, grafo y teoría de conjuntos). Una jerarquía de cuatro niveles similar es la base del formato estándar de intercambio de datos de CASE para el "*interworking*" de herramientas CASE [19].

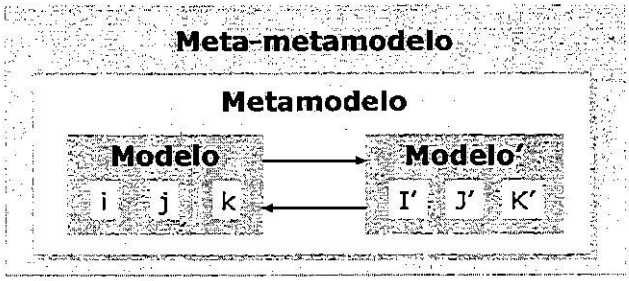


Figura 2. Modelo de jerarquía genérico que recoge los aspectos evolutivos y/o de transformación de dos modelos

Los modelos son necesarios, entre otras cosas, para minimizar la complejidad y relatividad inherentes al concepto "calidad del software", manejar diferentes perspectivas de modelado, gestionar la evolución y asegurar la consistencia de los cambios. Trasladando las ideas anteriores al ámbito de la especificación de requisitos se puede construir una arquitectura de gestión de calidad que sirva de marco para conseguir los objetivos que se han apuntado en la introducción.

Marco de referencia para la gestión de calidad

En este apartado se propone un patrón de arquitectura de gestión de calidad de la especificación de requisitos que separa y enlaza los aspectos referentes a diferentes perspectivas de modelado.

La definición de métricas en el ámbito de la ERS es claramente dependiente del método, lenguaje o notación de modelado que se utilice en la especificación por lo que puede hacerse uso de metamodelos para definir dichas métricas y establecer relaciones entre ellas (por ejemplo establecimiento de conexiones entre la perspectiva estructural, dinámica y funcional en la especificación de requisitos de sistemas orientados a objetos). Además, dichos modelos pueden servir de base para la aplicación automática de dichas métricas, mediante la combinación herramientas automáticas de modelado con un repositorio en el que se almacene y gestione información de

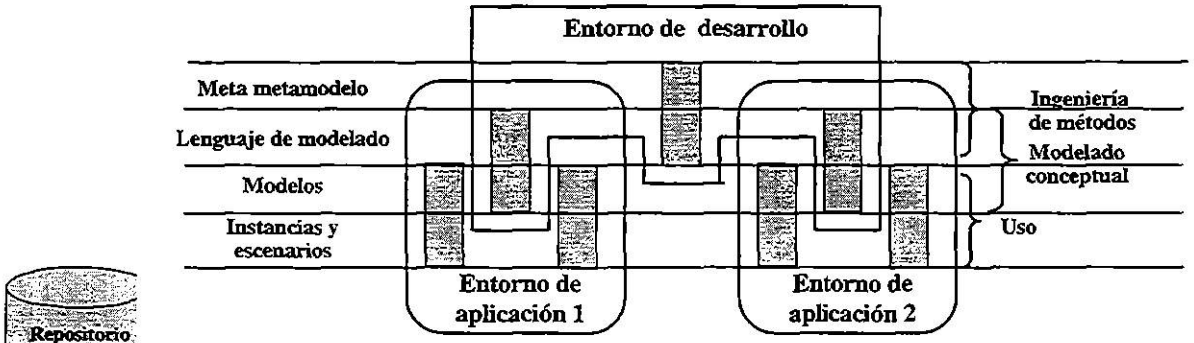
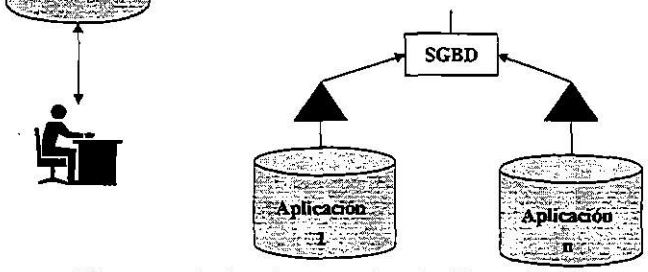


Figura 4. Estructura de referencia ISO IRDS (Information Resource Dictionary System)



diferentes niveles de instanciación (figura 3). Un repositorio de estas características permite a su vez mantener un registro histórico de los resultados obtenidos en el proceso de medida que puede servir de referencia para la realización de estudios comparativos sobre la validez de las métricas o la repercusión de determinados cambios y para la construcción de modelos predictivos.

Figura 3. Arquitectura de gestión de calidad

El primer aspecto que hay que considerar es la transformación del concepto abstracto de calidad en una serie de objetivos concretos de calidad que se corresponderán con las aspiraciones de los diferentes grupos de "*stakeholders*". Para fijar dichos objetivos se puede hacer uso de modelos de calidad descritos en la bibliografía. Podríamos adaptar el enfoque GQM [3] con el fin de enlazar los objetivos conceptuales con técnicas específicas de medición.

Por otra parte, hay que tener en cuenta que la consecución de los objetivos de calidad conlleva la realización de medidas de muy diversa naturaleza, la mayoría de las cuales no pueden realizarse directamente sobre atributos concretos, sino que requieren técnicas complejas de medición. Estos problemas se agravan cuando se intenta evaluar la calidad de las especificaciones de requisitos debido a las características inherentes a la mismas que dificultan la definición y aplicación de métricas, como se comentó en el apartado de introducción. El diseño y aplicación de técnicas de valoración e incluso predicción de la calidad se puede simplificar y sistematizar mediante un repositorio que almacene metadatos sobre las especificaciones y su evolución. El principal objetivo de este enfoque consiste en definir un esquema de metabase de datos que pueda capturar y enlazar todos los aspectos relevantes de la gestión de calidad.

Seguidamente se clasifican algunas de las dimensiones relevantes de calidad de la especificación de requisitos y se dan ejemplos de tipos de medidas que podrían ayudar a establecer la calidad de un componente particular con respecto a una particular dimensión de la calidad [22]. Esta estructura básica puede ser capturada formalmente en una extensión del enfoque GQM e implementada y usada en una metabase de datos.

La satisfacción de los atributos de calidad que se muestran en la tabla 2 conducen a la consecución de las dimensiones de calidad establecidas en la norma ISO 9126 [21] (fiabilidad, funcionalidad, eficiencia, portabilidad, facilidad de uso y facilidad de mantenimiento), la mayoría de las cuales sólo pueden evaluarse en fases próximas a la implementación. Por ejemplo, las dimensiones de corrección y complección conducen a mejorar la funcionalidad del sistema, y dimensiones como trazabilidad y legibilidad repercuten en la facilidad de uso y de mantenimiento.

La sistematización del proceso comienza con una adaptación de la propuesta de formalización de la gestión de calidad para datos de Jarke y colaboradores [22] basada en un análisis cualitativo de las relaciones entre los factores de calidad (ej. Objetivo-subobjetivo, objetivo-significado...). Los stakeholders pueden realizar la evaluación subjetiva de sus propios objetivos y de su importancia relativa. Dichas evaluaciones, junto con las calculadas, se usan como medidas de calidad en el modelo de arquitectura. Esto facilita una simple integración del modelo de calidad y arquitectura. Este enfoque se usa ampliamente en

la ingeniería industrial bajo la etiqueta de "*Quality Function Deployment*"

Dimensión	Medidas
Corrección	Número de conflictos con modelos del mundo real
Complección	Nivel de cobertura, número de reglas de negocios representadas
Minimalidad	Número de representaciones redundantes
Trazabilidad	¿Se registran los requisitos y sus cambios?
Legibilidad	Calidad de la documentación
Evolución	¿Se documenta la evolución del modelo?

Tabla 2. Dimensiones de la calidad de las ERS

La arquitectura propuesta se basa en el estándar ISO Information Resources Dictionary System (IRDS) [20] que propone la organización de la información en cuatro niveles de instanciación (figura 4): ***nivel de instancias y escenarios*** (contiene objetos que no pueden tener instancias: datos, estados, resultados de las mediciones...), ***nivel de modelos*** (clases que definen las propiedades de los objetos del nivel de instancias y las reglas de manipulación de los mismos), ***nivel de lenguajes de modelado*** (metaclases que definen la estructura de las clases del nivel del modelo) y ***nivel de meta metamodelo*** (meta metaclases con instancias en el nivel anterior. Es posible la definición de múltiples lenguajes de modelado mediante la apropiada instanciación). Esas cuatro capas se agrupan en tres pares de niveles entrelazados: ***entorno de uso***, ***entorno de modelado conceptual*** y ***entorno de ingeniería de métodos*** [27].

La gestión conjunta de todos los niveles del repositorio compartido hace posible soportar la coevolución requerida de los modelos, metamodelos e incluso meta metamodelos (modelos M2) que permiten realizar abstracciones de los lenguajes de modelado, y por tanto, definir metainformación sobre métricas adecuadas a las notaciones que soportan dichos lenguajes. Los lenguajes de modelado existentes, nuevos o extendidos, así como la definición de métricas específicas pueden ser instanciados del modelo M2, que puede estar también sujeto a cambios.

La implementación de la arquitectura IRDS mediante un repositorio gestionado por un sistema de gestión de metabase de datos (SGMBD) [26] proporciona soporte para metamodelos orientados a la medición. Combinando dicha arquitectura con modelos como GQM se pueden crear modelos que guían y proporciona soporte automático al usuario en el proceso de gestión de la calidad. Dichos modelos facilitan el establecimiento y gestión de los programas de medida, la elección de métricas y métodos de análisis, la utilización constructiva de experiencias pasadas, la definición de planes de medida y la estructuración de informes. Pueden también proporcionar soporte para definir y gestionar estudios empíricos para construir modelos predictivos (estimación). Debido a que se pueden capturar todos los aspectos del proceso de medida, esta arquitectura facilita el empaquetamiento y utilización de experiencias capturadas durante el uso de la herramienta junto con un conjunto de software convencional. La figura 3 muestra un esquema de la forma de implementación de la arquitectura propuesta.

Conclusiones

Con este trabajo se ha pretendido, en primer lugar, suscitar el interés por la medición en las etapas iniciales del desarrollo de software, ya que dichas etapas son decisivas en la consecución de la calidad de los sistemas. Gran parte de la investigación realizada en este campo se centra en el estudio de atributos estructurales, descuidando los aspectos dinámico y funcional de las especificaciones. Aquí se presenta, además, un marco de referencia que permite gestionar conjuntamente y de forma sistemática diferentes perspectivas de la calidad de los requisitos. Dicho marco se basa en la arquitectura IRDS que contempla la definición de modelos en diferentes niveles de instanciación. El uso de tales modelos proporciona un contexto para la definición de objetivos, reutilización de objetos y experiencias, selección de los procesos de medida más adecuados, evaluación y comparación de resultados y predicción. En nuestro caso, podríamos concluir que el enfoque propuesto proporciona: Un marco para definir modelos de calidad y objetivos específicos del proyecto, un mecanismo para evaluar la calidad en las primeras fases del ciclo de vida, soporte para el registro y uso provechoso de experiencias pasadas y un medio para gestionar la evolución y la consistencia de los cambios

Bibliografía

1. Albrecht, A.J., "Measuring application development", Proc. of IBM Applications DevelopmentJoint SHARE/GUIDE Symposium, Monterey, CA, pp 83-92, 1979.
2. Arthur, J.D. y Stevens, K.T. , "Assessing the adequacy of documentation through document quality indicators", Proceedings of the IEEE Conference of Software Maintenance, pp. 40-49, 1989.
3. Basili, V.R. y Rombach, H.D., "The TAME Project: Towards Improvement-Oriented Software Environments", IEEE Transaction on Software Engineering,14(6), 758-73 1988.
4. Binder, R., "Testing Object-Oriented Systems", American Programmer, 7(4), 22-29, 1994.
5. Boehm, B.W., Kaspar, J.R. y otros "Characteristics of Software Quality", TRW Series of Software Technology, 1978.
6. Boehm, B.W., Clark, B., Horowitz, E. et al., "Cost Models for future life cycle processes: COCOMO 2.0", Annals of Software Engineering 1(1), pp 1-24, 1995.
7. Briand, L.C., Daly, J.W. y Wüst, J.K. "A unified framework for coupling measurement in object-oriented system", IEEE Transaction on Software Engineering, 25 (1), 1999.
8. Brykczynskki, B., "A survey of software inspection checklist", ACM Software Engineering Notes, 24(1), pp 82-89, 1999.
9. Chidamber, S.R. y Kemerer, C.F., "A Metrics Suite for Object-Oriented Design" ,IEEE Transactions of Software Engineering, 20(6), 476-493, 1994.
10. Churcher, N.I. and Shepperd, M.J., "Towards Conceptual Framework for Object-Oriented Metrics", ACM Software Engineering Notes, 20 (2), 67-76, 1995.
11. Cook, A.D., "Confusing Process and Product: Why the Quality is not There Yet", CrossTalk, july, 1999.
12. Davis, A. et al., "Identifying and Measuring Quality in a Software Requirements Specification" Proceedings of the First International Software Metrics Symposium, Baltimore, May 21-22, pp. 141-152, 1993.
13. DeMarco, T., "Controlling Software Projects", Yourdon Press, 1982.
14. Farbey, B., "Software Quality metrics: considerations about requirements and requirements specification", Information and Software Technology, 32 (1), pp 60-64, 1990.
15. French, J.C., Knight, J.C. y Powell, A.L., "Applying hipertext structures to software documentation", Information Processing and Management", 33 (2), pp 219-231, 1997.
16. Genero, M., Manso, M.E., Piattini, M. y García F.J. "Assessing the quality and the Complexity of OMT Models" 2nd European Software Measurements Conference-FESMA 99, Amsterdam, Netherlands, pp 99-109, 1999.
17. Genero, M., Piattini, M. y Calero, C. "Una propuesta para medir la calidad de los diagramas de clases en UML", IDEAS'2000, Cancun, México, pp 373-384, 2000.
18. Gilb, T. "Principles of software engineering management", Addison-Wesley, 1988.
19. Imber, M. "CASE data Interchange format standars", Information and Software Technology, Nov. 1991, pp. 647-655
20. ISO/IEC 10027, "Information Technology - Information Resource Dictionary System (IRDS) - Framework", ISO/IEC international Standard edition, 1990.
21. ISO/IEC 9126, "Software product evaluation - Quality characteristics and guidelines for their use ", 1991.
22. Jarke, M., Jeusfeld, M.A., Quix, C. y Vassiliadis, P. "Architecture and quality in data warehouses: and Extended Repository Approach", Information System, 24(3), pp 229-253, 1999.
23. Lehner, F., "Quality control in software documentation: Measurement of text comprehensibility", Information and Management, 25, pp 133-146, 1993.
24. Lorenz, M. and Kidd, J., "Object_oriented Software Metrics", Prentice Hall 1994.
25. McCall, J.A., Richards, P.K. and Walters, G.F. "Factors in software quality", RADC TR-77-369, US Rome Air Development Center Reports NTIS AD/A-049 014, 015, 055, 1977.
26. Moreno, M.N., Polo, M.J., Miguel, L.A. y García, F.J. "Un sistema de meta-base de datos basado en UML e integrado en un sistema de gestión de bases de datos distribuidas." En Jornadas de Ingeniería del Software y bases de datos, Cáceres, España, 1999.
27. Nissen, H.W., and Jarke, M., "Repository support for multi-perspective requirements engineering". Information System, Vol. 24, No. 2, pp. 131-158, 1999.
28. Offen, R.J. y Jeffery, R., "Establishing software measurements programs", IEEE Software, 14 (2), pp 45-53, 1997.
29. Paulk, M., Curtis, B., Chrissis, M., and Weber, C. "Capability Maturity Model for Software: Version 1.1". Technical Report SEI-93-TR-24, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, 1993.
30. Poels, G., "Towards a size measurement framework for object-oriented especifications". H. Coombes, M. Hooft van Huysduynen, and B. Peeters (eds.), Proceedings of the 1st European Software Measurement Conference (FESMA'98), Antwerp, , pp. 379-388, 1998.
31. Poels, G., "On the measurements of event-based object-oriented conceptual models". 4th International ECOOP Workshop on Quantitative Approaches in Object Oriented Software Engineering, Cannes, France, 2000.
32. Roth, T., Aiken, P. y Hobbs, S., "Hypermedia support for software developmemt: a retrospective assessment", Hypermedia, 6 (3), pp 149-173, 1994.
33. Rout, T.P. "Software process improvement and practice", 1(1), pp 57-66, 1995.
34. Samson, W.B., Nevill, D.G. Y Dugard, P.I., "Predictive software metrics based on a formal

specification", Software Engineering Journal, 5(1), 1990.
35. SPICE, "SPICE Document Suite, Software Process Improvement and Capability determination", http://www.sqi.gu.edu.au/spice/ , 1999.