CACiC 2023
CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACIÓN

Evaluación de requisitos de software utilizando GQM

Gladys Kaplan[1], Gerardo Riera[1], María Inés Bertín[1], Gabriel Blanco[1]

[1]Departamento de Ingeniería e Investigaciones Tecnológicas,
Universidad Nacional de La Matanza. San Justo, Buenos Aires
{gkaplan,griera,mbertin,gblanco}@unlam.edu.ar

Abstract. La Ingeniería de Requisitos ha puesto mucho énfasis en asegurar la calidad de los requisitos. A pesar de estos esfuerzos, estas descripciones siguen presentando serios problemas. La amplia utilización del lenguaje natural puede ser uno de los responsables. Se ha intentado mitigar estos problemas construyendo modelos previos, glosarios, etc., pero aún no se han obtenido los resultados esperados. En este artículo se presenta un método para evaluar las *características* de los requisitos y determinar su grado de comprensión. Para ello se utilizó un método cuantitativo Gold Question Metric que a través de preguntas permite evaluar internamente cada requisito. Para probar el método propuesto se utilizaron especificaciones tomadas directamente de la industria, las cuales eran consideradas correctas por sus autores. En todos los casos, muchos de estos requisitos presentaron serios problemas de descripción. Adicionalmente, la ejecución del método generó un conjunto de dudas que resultaron muy promisorias para corregir cada requisito.

Keywords: requisitos del software, características de los requisitos, verificación, Gold Question Metric (GQM).

1. Introducción

El objetivo de la Ingeniería de Software es construir software de la mayor calidad posible al menor costo [1]. Para ello es indispensable contar con una Especificación de Requisitos de Software (ERS) completa y correcta que permita asegurar el punto de partida de la construcción del software [2]. En 1981, Basili [3] encontró cerca de 88 errores en una ERS de 400 páginas para el proyecto "*A-7E Operational Flight Program*". Esta ERS había sido escrita por un grupo de expertos en especificación de requisitos. En el mismo camino, los estudios de Boehm [4][5] han demostrado que el impacto potencial de los requisitos mal formulados es sustancial. Boehm estudio como los requisitos, la especificación y los errores de diseño son los más numerosos en un sistema, con un promedio del 64 % en comparación con el 36 % de los errores de codificación. Para el presente artículo se analizó, entre otras, una ERS del proyecto "*Seguimiento de Alumnos"* de una Universidad, la que contó con 34 páginas donde se describieron 211 requisitos. En este caso también fueron escritos por expertos en requisitos. Se detectó que más de la mitad de los requisitos presentaban algún tipo de problema. Entre los problemas que pueden acarrear requisitos mal comprendido se encuentra satisfacer parcialmente las necesidades del cliente, multiplicar el costo de construcción, incumplir los tiempos estimados, etc. Se debe considerar que el 71.8% de las ERS existentes utilizan el Lenguaje Natural (LN) [6] para representar los requisitos, siendo esto una gran ventaja para mejorar la comunicación entre los stakeholders y los desarrolladores, pero una desventaja por las dificultades intrínsecas que tiene, como la

ambigüedad y la falta de precisión. Para mitigar estos problemas se han propuesto diferentes mecanismos como puede ser la utilización de glosarios [7]; verificaciones exhaustivas, por ejemplo, con Inspecciones [8]; la generación de modelos previos a la ERS (Casos de Uso, Escenarios) [9][10]; entre otros. Ben Achour et al. [11] determinó que la mitad de los Casos de Uso de un proyecto tenían errores basados en la falta de comprensión de la terminología utilizada. Por otro lado, también se presentan problemas cuando existen modelos previos como es el caso de los Escenarios [12][13]. Estos Escenarios tienen empotrados los requisitos los cuales se extraen de aquellos episodios donde el actor *sistema* tiene un rol. Al especificar los requisitos desde los Escenarios, estos aparecen en la ERS en el mismo orden en cual fueron extraídos y muchos de ellos suelen comprenderse sin dificultad mientras se conserven en la misma secuencia. Esto se debe a que en los requisitos anteriores existe información que lo completa. Pero cuando esos requisitos se separan, por ejemplo, al ordenarlos por algún ítem de *importancia,* pierden contexto y pueden llevar a interpretaciones erróneas. Si bien este problema puede ser subsanado retornando al modelo que le dio origen, requiere de un esfuerzo adicional con un alto riesgo de no hacerlo e interpretar equivocadamente el requisito. Por otro lado, quienes son responsables de las ERS suelen no ser conscientes que existen estos problemas y se valida con el cliente-usuario y se construye el sistema de software con requisitos improcedentes.

En el presente artículo se propone medir cada requisito en particular tomando en cuenta las *características* que lo definen. A tal efecto se utilizó la clasificación de Wiegers [14] que establece las siguientes características: necesario, correcto, conciso, completo, consistente, no ambiguo, verificable, trazable y modificable. Para determinar el grado de cumplimiento se definió una métrica cuantitativa para cada requisito en particular utilizando el Método Goal Question Metric (GQM) [3]. Se construyó un conjunto de preguntas que, además de medir, permite conocer con exactitud el problema de descripción existente, por lo que actúa como una suerte de verificación que permite la rápida corrección del requisito. Cabe destacar que este método es útil para trabajar con cada requisito en particular, pero para liberar la ERS se deben evaluar las características y atributos de la especificación, pero este punto no es tratado en el presente artículo.

El artículo está organizado de la siguiente manera: en la sección 2 se explica el modelo GQM; en la sección 3 se describen las características con las cuales se construye el modelo propuesto; en la sección 4 se define el modelo GQM para evaluar las características de cada requisito; en la sección 5 se analizan los resultados obtenidos y en la sección 6 las conclusiones y trabajos futuros.

2. Goal Question Metric (GQM)

El modelo GQM fue desarrollado por Basili [3] para medir diferentes metas de una organización y saber si las mismas fueron alcanzadas. Para ello, primero debe especificar las metas, rastrear esos objetivos a los datos y finalmente, proporcionar un marco para interpretar los datos en relación con el objetivo. El método se lo utiliza buscando una mejora evolutiva de la calidad. El resultado de la aplicación del enfoque GQM es la especificación de un sistema de medición dirigido a un conjunto particular de problemas y un conjunto de reglas para la interpretación de los datos de medición.

Para Basili una medición puede ser más satisfactoria si es diseñada teniendo en cuenta las metas, y las preguntas ayudan a medir si se está alcanzando en forma exitosa la meta definida. Se busca mejorar la calidad y confiabilidad reduciendo costos, riesgos y mejorando tiempos.

El modelo de medición GQM tiene tres niveles:

1) **Nivel conceptual (OBJETIVO/GOAL)**

Un objetivo se define para un objeto, por una variedad de razones, con respecto a varios modelos de calidad, desde varios puntos de vista, en relación con un ambiente particular.

Los objetos de medición son:

Productos: Artefactos, entregables y documentos que se producen durante el ciclo de vida del sistema; Por ejemplo, especificaciones, diseños, programas, conjuntos de pruebas.

Procesos: actividades relacionadas con el software normalmente asociadas con el tiempo; P.ej. especificar, diseñar, probar, entrevistar.

Recursos: Elementos utilizados por los procesos para producir sus salidas; P.ej. personal, hardware, software, espacio de oficina.

2) **Nivel operativo (PREGUNTA/QUESTIONS)**

Se utiliza un conjunto de preguntas para caracterizar la forma la evaluación/logro de una meta específica que se va a realizar en base a algún modelo caracterizador. Las preguntas tratan de caracterizar el objeto de medición (producto, proceso, recurso) con respecto a un problema de calidad y determinar su calidad desde el punto de vista seleccionado.

3) **Nivel cuantitativo (MÉTRICA/METRIC)**

Un conjunto de datos está asociado con cada pregunta para responderla de forma cuantitativa.

Los datos pueden ser:

Objetivo: Si dependen únicamente del objeto que se está midiendo y no en el punto de vista desde el que se toman; Por ejemplo, el número de versiones de un documento, horas del personal dedicadas a una tarea, tamaño de un programa.

Subjetivos: Si dependen tanto del objeto que se está midiendo como el punto de vista desde el que se toman; Por ejemplo, legibilidad de un texto, nivel de satisfacción del usuario.

3. Características de los requisitos del software

Para construir el método fue necesario comprender cuál era el problema real. Para ello se examinaron algunos requisitos al azar y simplemente se los leyó. Durante este proceso aparecieron muchas dudas que no permitían comprender en su totalidad el servicio descripto. Para ilustrar este procedimiento se describen algunos ejemplos:

Requisito #1: *"El sistema debe asignar un número identificatorio a cada carrera".* Observación: ¿Cómo se compone ese número idenficatorio?

Requisito #2: *"Se permitirá el registro de pedidos de compra con datos obligatorios incompletos, los cuales podrán completarse posteriormente modificando el pedido. Antes de poder aprobarse los datos del pedido deben estar completos".* Observación: ¿Cómo se puede aceptar datos obligatorios incompletos, si son obligatorios? ¿Para aprobar el pedido deben estar completos los obligatorios o todos los datos? Además de las dudas, se puede desagregar en varios requisitos.

Requisito #3: *"El sistema debe permitir imprimir etiquetas de manera unitaria".* Observación: ¿se debe imprimir etiquetas de un solo producto o imprimir una única etiqueta de un único producto?

Requisito #4: *"El sistema deberá generar reportes de manera semanal, mensual y anual, según sea la exigencia".* Observación: ¿Cómo se determina la exigencia?

Las dudas u observaciones que aparecieron en los requisitos analizados se clasificaron según las *características* de Wiegers [12] en conciso, necesario, verificable, correcto, completo, no ambiguo, trazable, consistente y modificable. Pero algunas observaciones no pudieron incorporarse a esta clasificación. Por lo tanto, se agregaron las siguientes tres características:

- Indivisible.
- Autónomo.
- Punto de vista.

Un requisito es indivisible cuando no se puede extraer más de una funcionalidad del mismo. Su granularidad es la menor. Un requisito es autónomo cuando se lo puede comprender por sí mismo, sin depender de otros requisitos. Esto no se relaciona con las dependencias entre requisitos sino con la comprensión del servicio que describe. Finalmente, el punto de vista es para homogeneizar las descripciones, de esta manera todos los requisitos tienen la misma perspectiva y esto facilita la lectura y por consiguiente, su comprensión. En general para una ERS la perspectiva es la del producto ("El sistema debe …"), para las Historias de Usuario es la del usuario, etc. En la Fig.1 se puede observar la clasificación completa utilizada para crear el modelo GQM.

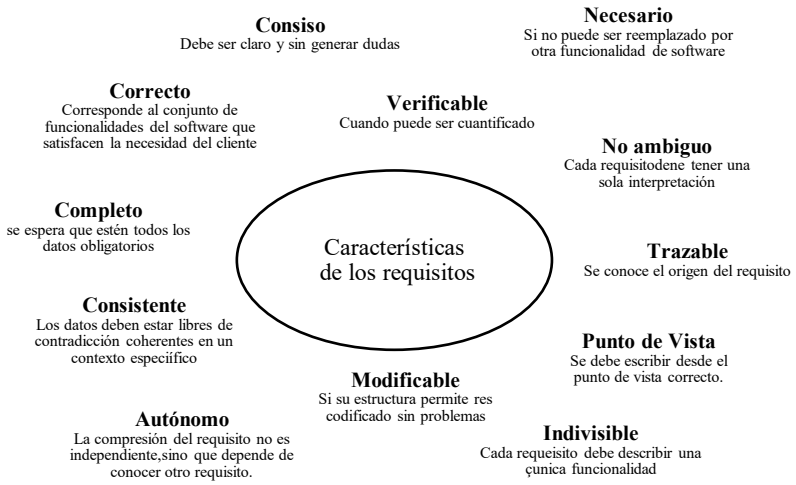

Fig. 1. Características de los requisitos

4. Modelo GQM para evaluar la calidad de los requisitos

Con todas las características de la sección anterior se construyó el modelo GQM definiendo los tres niveles. A continuación, se describe cada uno:

1) Nivel conceptual (OBJETIVO / GOAL)
   Objetivo: determinar la calidad de cada requisito de software.

2) Nivel operativo (PREGUNTA / QUESTIONS)
   Tipo de Respuesta: SI/NO

La Tabla 1 muestra las preguntas para cada característica y el peso asignado a cada una, que puede ser 0 o 1. En este caso solo se definió una pregunta por cuestiones de espacio, pero es posible profundizar en algunas características repartiendo el peso entre todas las preguntas formuladas.

Pregunta y **Peso** de cada característica:

Tabla 1. Nivel Operativo del GQM

Características	Preguntas	Peso en la fórmula
Correcto	**(P1)** ¿está dentro del alcance del Objetivo del Sistema?	RTA: NO=1
Conciso	**(P2)** ¿Es de fácil comprensión? (se puede comprender en una primera lectura)	RTA: NO=1
Necesario	**(P3)** ¿Puede ser reemplazado por otra funcionalidad?	RTA: SI=1
Verificable	**(P4)** ¿Están los parámetros/valores que permiten aplicar un método cuantificable?	RTA: NO=1
Completo	**(P5)** ¿Están los datos necesarios para implementar el requisito?	RTA: NO=1
Consistente	**(P6)** ¿Existe alguna contradicción en el propio requisito?	RTA: SI=1
Autónomo	**(P7)** ¿Alcanza con analizar este requisito solo? (sin conocer otros requisitos)	RTA: NO=1
Indivisible	**(P8)** ¿La realización del requisito corresponde a una única funcionalidad del software?	RTA: NO=1
Punto de Vista	**(P9)** ¿Está escrito desde la perspectiva correcta?	RTA: NO=1
No ambiguo	**(P10)** ¿Puede tener más de una interpretación?	RTA: SI=1
Trazable	**(P11)** ¿Se comprende el origen del requisito?	RTA: NO=1
Modificable	**(P12)** ¿Es posible de ser modificado forma fácil, completa y consistente?	RTA: NO=1

3) Nivel cuantitativo (MÉTRICA / METRIC)

Fórmula: 0 <= (∑ preguntas) / Cantidad de preguntas <= 1

La misma fórmula es utiliza para evaluar las características de un requisito y luego, se puede aplicar la misma fórmula para evaluar todo el requisito.

Evaluación:

La Tabla 2 muestra los rangos que permiten evaluar cada característica y luego, cada requisito. Puede observarse que estos valores pueden ajustarse rápidamente para contextos más críticos o particulares.

Tabla 2. Nivel Cuantitativo del GQM

Rangos x CARACTERÍSTICA		
Aceptado	A revisión	Rechazado
0	>=0.1 y <=0.99	=1

Rangos x REQUISITO		
Aceptado	A revisión	Rechazado
0	>=0.1 y <=0.39	>=0.4

5. Análisis Cuantitativo

Para realizar las pruebas del método se utilizaron cinco casos reales:

1) ERS#1 - Sistema de Organización Eventos
2) ERS#2 - Sistema de Gestión de Títulos
3) ERS#3 - Sistema de Rematriculación
4) ERS#4 - Sistema de Control de Calidad
5) ERS#5 – Gestión de Tiendas

En la ERS#1y ERS#5 los requisitos fueron descriptos directamente desde el contexto, mientras que la ERS#2, ERS#3 ERS#4 fueron obtenidas desde el modelo Escenarios, donde los requisitos están empotrados en los episodios. En estos casos, los requisitos extraídos tienen la característica de no presentar conflictos ya que se obtienen desde un modelo verificado y validado. Además, cuentan con un glosario, el cual se construye muy tempranamente en el proceso de requisitos utilizado.

ID_REQUISITO: #4
Descripción: El sistema debe proponer al alumno diferentes opciones de inscripción de acuerdo a si debe recursar materias o no. Si recursa, el sistema debe obligar a definir un turno para estas materias y luego completar las restantes. Si no recursa, el sistema debe permitir que elija el turno y las materias troncales, MOFE, MOGE y obligaciones académicas según corresponda a su carrera, ciclo y a la carga existente de la oferta académica.

Característica	Peso	Observaciones
Correcto	0	
Conciso	0	
Necesario	0	
Verificable	0	
Completo	1	No se entiende que se va a controlar en la prematriculación. Tampoco que es “periódico” (cuánto tiempo?)
Consistente	0	
Indivisible	1	Se habla de un proceso automático de pre-matriculación, y además de un envío de mail informando problema y posible solución. Al menos son 2 funcionalidades.
Autónomo	1	No se sabe cómo es el proceso de pre-matriculación automático.
Punto de Vista	0	
No ambiguo	0	
Trazable	1	
Modificable	1	
TOTAL x Requisito	**5**	

Fig. 2. Ejemplos del Nivel Operativo

Cabe aclarar que algunas características tienen una relación muy estrecha, por ejemplo cuando un requisito no está completo genera ambigüedad y a su vez hace dificultosa la trazabilidad. En estos casos se sugiere responder cada pregunta de manera independiente.

A modo de ejemplo, en la Fig.2 se muestra la evaluación de características realizada sobre el Requisito #4 de la ERS#3. Cabe destacar que, en todos los casos, los requisitos han sido tomados textualmente de la especificación correspondiente. La columna "Peso" contiene 0 o 1 de acuerdo a la respuesta de la pregunta de cada característica (ver Fig. 2). En la columna "Observaciones" se describen las dudas que aparecen durante el análisis. Esta columna puede convertirse en un mecanismo de corrección valioso, ya que puede ser entregado a los autores para que se enfoquen directamente en cómo mejorar cada descripción. Finalmente, se suma la columna "Peso" para obtener el "Total x Requisito", este valor indica la calidad del mismo. Este total se utiliza en la columna "Fórmula" de la Tabla 3.

Tabla 3. Nivel Cuantitativo para la ERS#3

Id. Requisito	Fórmula *Total x Requisito / Cantidad de requisitos*	Decisión por requisito *Aceptado = 0 A revisión >= 0.1 y <= 0.49 Rechazado >= 0.5*
#1	1 / 10 = 0,1	A revisión
#2	7 / 10 = 0,7	**RECHAZADO**
#3	2 / 10 = 0,2	A revisión
#4	5 / 10 = 0,5	**RECHAZADO**
#5	6 / 10 = 0,6	**RECHAZADO**
#6	5 / 10 = 0,5	**RECHAZADO**
#7	4 / 10 = 0,4	A revisión
#8	0 / 10 = 0	ACEPTADO
#9	2 / 10 = 0,2	A revisión
#10	2 / 10 = 0,2	A revisión

Como se mencionó al comienzo de esta sección, se evaluaron cinco casos obteniendo los siguientes resultados finales, tal como se muestra en la Tabla 4.

Tabla 4. Resumen de los requisitos

CASOS	Total de Requisitos	Cant. Aceptados	Cant. A Revisión	Cant. Rechazados
ERS#1	11	3	2	6
ERS#2	18	5	8	5
ERS#3	10	1	5	4
ERS#4	18	6	7	5
ERS#5	21	4	8	9

La suma de todos los casos analizados alcanza a 78 requisitos de los cuales solo 19 han sido aceptados sin ninguna observación.

6. Conclusiones y Trabajos Futuros

En este artículo se presenta un método para determinar cuantitativamente la comprensión de cada requisito. Para ello se utiliza el modelo GQM que permite determinar qué características son satisfechas y cuáles no. Como ya se mencionó, las cinco ERS utilizadas han sido tomadas de casos reales, demostrando cómo se está trabajando en la industria con respecto a la especificación de requisitos. En este mismo camino, es importante destacar que cuando se consultó a las empresas acerca de sus documentos de especificación la respuesta fue alarmante, ya que muchas de ellas no contaban con documentos respaldatorios. En algunos casos lo justificaron con que en la actualidad solo hacen mantenimiento del software, en otros casos trabajan con el boca a boca, etc. Cabe destacar tambien la importancia de realizar procesos que garanticen la construccion de los requisitos. Esto puede observarse en la Tabla 4 en los resultados finales, donde las ERS#1 y ERS#5 que fueron realizadas directamente con el usuario, generan más dudas que cuando se construyen desde modelos previos. Cabe destacar que en estos casos la experiencia del ingeniero de requisitos influye directamente en los resultados.

Tabla 5. Porcentaje de características incumplidas

Característica	Cantidad Requisitos con defectos	Porcentaje
Completo	24	31.90
No ambiguo	12	15.38
Autónomo	7	8.98
Indivisible	6	7.69
Consistente	5	6.41
Punto de vista	1	1.28
Conciso	1	1.28
Verificable	1	1.28
Modificable	1	1.28
Correcto	1	1.28
Trazable	0	0
Necesario	0	0
Total	**59**	**76.76**

Como resultado final se presentan los datos en la Tabla 5. Para construir esta Tabla se aplanaron todos los requisitos (78 requisitos) y se agruparon por característica. Se puede observar que el problema de la incompletitud se ha presentado en 24 requisitos, determinando que el 31.90% de los problemas radican en esta característica. Le sigue en importancia la ambiguedad, luego la autonomía, la consistencia y la indivisibilidad y finalmente, el punto de vista y lo conciso. La característica autónomo se da con mayor frecuencia cuando existen modelos previos a la ERS los cuales tienen los requisitos empotrados, ya que la secuencia en el cual aparecen en el modelo suele ser el mismo que en la ERS.

Estos resultados corroboran que siguen existiendo especificaciones con serios problemas de comprensión.

Como trabajo futuro se espera analizar más casos e incluir *Historias de Usuario* y *Casos de Uso*. Si bien las pruebas realizadas sobre ERS son iniciales, los resultados

obtenidos son muy promisorios. También se espera realizar mediciones en la industria y estudiar cómo responde como mecanismo de aprendizaje.

Referencias

1. Ian Sommerville, "Ingeniería de Software", 10th Edition, ISBN 10 9332582696, 2017
2. J. Dick, E. Hull, and K. Jackson (2017), Requirements Engineering". 4ta Edition, Springer, ISBN:9783319610733, 3319610732.
3. V.R. Basili (1992) "Software Modeling and Measurement: The Goal Question Metric Paradigm,", Computer Science Technical Report Series, CS-TR-2956 (UMIACS-TR-9296), University of Maryland, College Park, MD.
4. Boehm, B. (1984) "Model and metrics for software management and engineering", IEEE Computer Society Press, pp. 4-9
5. Birihanu, Ermiyas. (2018). "Analysis of Software Quality Using Software Metrics".
6. International Journal on Computational Science & Applications. 8. 11-20. 10.5121/ijcsa.2018.8502.
7. Berry D. (2004) "Requirements Ambiguity", en el libro "Perspectives on Software Requirements", Kluwer Academic Publishers, Ed. Leite, J.C.S.P., Doorn, J.H. EEUU, ISBN: 1-4020-7625-8, Capítulo 2.
8. Ryan Mike, Wheatcraft, Louis, Dick, Jeremy and Zinni, Rick. (2015). On the Definition of Terms in a Requirements Expression. INCOSE International Symposium. 25. 10.1002/j.2334-5837.2015.00055.x.
9. Kaplan Gladys (2022) "Proceso de Requisitos Validado Empíricamente". Tesis doctoral de la UNLP, Capitulo 13.
10. Del Águila Cano, I. M. (2022). "Fundamentos de Ingeniería de los Requisitos". Editorial Universidad de Almería. ISBN: 9788413511610, 8413511615, pp 68-77.
11. Jacobson, I., Christerson, M., Jonsson, P., Overgaard, G.,Reading, (1992) "Object-oriented
12. Software Engineering - A Use Case Driven Approach", MA: Addison Wesley, Nueva York: ACM Press.
13. Ben Achour, C., Rolland, C., Maiden, N.A.M., Souveyet, C. (1999) "Guiding Use Case Authoring: Results of an Empirical Study", International Symposium On Requirements Engineering (RE´99), Limerick, Irlanda, IEEE Computer Society Press, pp.36-43.
14. Hadad G., Doorn J. y Kaplan G. (2009) "Explicitar Requisitos del Software usando escenarios", WER 09, IX Workshop de Engenharia de Requisitos.
15. Leite, J.C.S.P., Hadad, G.D.S., Doorn, J.H., Kaplan, G.N. (2000), "Scenario Construction Process", Requirements Engineering Journal, Springer-Verlag London Ltd., Vol.5, Nº1, pp. 38-61.
16. K.E. Wiegers, (1999) "Writing Quality Requirements – Process Impact". Disponible en: http://www.processimpact.com/articles/qualreqs.pdf