Revista Cubana de Ciencias Informáticas
Vol. 9, No. Especial CCIA 17, 2015
RNPS: 2301 | ISSN: 2227-1899
http://rcci.uci.cu
Pág. 76-84

Tipo de artículo: Artículo original
Temática: Ingeniería y Gestión de software
Recibido: 05/01/2015 | Aceptado: 20/02/2015

Modelando con UML el proceso de evaluación de productos de software utilizando el enfoque GQM

Modeling with UML the evaluation process of software products using GQM

José Ramón Hernández Vega [1*], Sandra Verona Marcos [2], Sonia Pérez Lovelle [2]

[1*] Centro de Estudios Matemáticos para las Ciencias Técnicas, Instituto Superior Politécnico José Antonio Echeverría, Calle 114 # 11901 e/ Ciclovía y Rotonda Marianao La Habana, Cuba. jvega@cemat.cujae.edu.cu
[2] Facultad de Ingeniería Informática, Instituto Superior Politécnico José Antonio Echeverría, Calle 114 # 11901 e/ Ciclovía y Rotonda Marianao La Habana, Cuba.{sverona, sperezl}@ceis.cujae.edu.cu

[*]**Autor para correspondencia**: jvega@cemat.cujae.edu.cu

Resumen

El enfoque GQM (Meta-Pregunta-Métrica, por sus siglas en inglés) ha sido utilizado en el proceso de evaluación de calidad de productos de software, como instancia de un paradigma de medida. Sin embargo, este enfoque no tiene asociado diagramas o elementos visuales que permitan una mejor comunicación entre los encargados de la evaluación y los desarrolladores, por lo que se hace una propuesta para usar UML (*Unified Modeling Language*) como lenguaje de especificación para describir su estructura y a partir de esto, usar el perfil de pruebas de UML (UTP, UML *Testing Profile*) para la especificación del proceso, mediante la descripción de la arquitectura, el comportamiento, los datos y la gestión de las pruebas.

Palabras clave: calidad de software, proceso de evaluación, GQM, UML, UTP.

Abstract

GQM (Goal-Questions-Metrics) has been used in evaluation of quality process of software products as instance of metric paradigm. But, it has not diagrams or visual elements for communication between evaluators and developers. For this reason there are a proposal to use UML (Unified Modeling Language) as specification language in order to describe their structural aspects and to use UML Testing Profile to describe the process through architectural description, test behavior, test data and test management.

Keywords: *evaluation process, GQM, software quality, UML, UTP.*

Introducción

UML *(OMG, 2012)* fue adoptado como estándar del *Object Management Group* (OMG) en 1997 debido a que representa una colección de las mejores prácticas de ingeniería que han sido probadas con éxito en el modelado de sistemas. Es un lenguaje para la especificación, visualización, construcción y documentación de sistemas, no solo de software. Teniendo en cuenta estas propiedades, es que se presenta esta propuesta en la que se usa este lenguaje para especificar los elementos que conforman el enfoque GQM *(BASILI et al., 2014), (BASILI et al., 2010), (BASILI et al.,*

1994) para garantizar una mejor comunicación entre el equipo de desarrolladores y el equipo encargado de realizar las pruebas y además se incorporan los elementos de su perfil de pruebas (UTP) *(OMG, 2013)* con vistas a incorporar a la documentación del sistema lo relacionado con las pruebas a desarrollar.

Este trabajo cuenta con una presentación del enfoque GQM como herramienta para la evaluación de la calidad de los productos de software, una presentación de UML y su perfil de pruebas, el uso de UML para describir el enfoque GQM, así como los elementos del perfil de pruebas a incorporar a la documentación del sistema a probar y por último, las conclusiones.

Materiales y Métodos

Evaluación de calidad de los productos de software

En los procesos de ingeniería, la medición constituye un elemento clave *(DUJMOVIC, 1982)*. La aplicación de medidas se debe a la necesidad de obtener una mejor comprensión de los atributos en los modelos creados; pero fundamentalmente, estas son empleadas para valorar la calidad de los productos de ingeniería o de los sistemas construidos.

En este sentido, la evaluación de la calidad se ha convertido en una preocupación constante, a la cual se dedican numerosos esfuerzos dentro de las organizaciones. Siendo esta una forma práctica de desarrollar productos y/o procesos que posean una efectiva ubicación en el mercado, pues constituye un modo que ayuda a aumentar sustancialmente la probabilidad de obtener una buena aceptación en los clientes finales *(PRESSMAN, 2002)*.

Es por estas razones que las organizaciones que se dedican al desarrollo de software se han encargado de incluir dentro de sus procesos un espacio para la evaluación de la calidad, definiendo en cada caso, una metodología propia para alcanzar los niveles o estándares puntualmente previstos. Lo cual indica que alcanzar altos niveles de calidad de software, está claramente en correspondencia con el proceso que se utilice para lograrla *(PIATTINI et al., 2010)*.

Sobre el tema Mc Gregor *(Mc GREGOR, 2001)* afirma: *"el aseguramiento de la calidad en una empresa de desarrollo de software debe orientarse hacia la prevención de los defectos y la estandarización de las actividades que se ejecutan durante las diferentes etapas del desarrollo, entre otros aspectos"*.

Para tal fin, a lo largo de los años, se han creado un conjunto de modelos de calidad, dentro de los que se destacan *(Mc CALL et al., 1977), (BOEHM et al., 1978), (IEEE, 1998), (ISO/IEC, 2001), (ISO/IEC, 2010)*. A partir de dichos modelos de evaluación se han construido una colección amplia de métricas, las cuales constituyen hoy un referente importante a la hora de valorar cuantitativamente la calidad de los productos de software. El grupo de estándares y normas creadas recientemente *(ISO/IEC, 2003a), (ISO/IEC, 2003b), (ISO/IEC, 2004)*, dan fe de esta situación.

GQM como soporte al proceso de evaluación de calidad de los productos de software

En la práctica, la colección de métricas para evaluar la calidad de los productos de software es muy amplia, y sigue aumentando. Este fenómeno se evidencia de forma explícita con la norma ISO/IEC 9126-1 *(ISO/IEC, 2001)* y su sucesora ISO/IEC 25010 *(ISO/IEC, 2011)*, ya que esta última incorpora un nuevo grupo de atributos posible a evaluar y por consiguiente un conjunto de métricas que permitan evaluar dichos atributos.

Revista Cubana de Ciencias Informáticas
Vol. 9, No. Especial CCIA 17, 2015
RNPS: 2301 | ISSN: 2227-1899
http://rcci.uci.cu
Pág. 76-84

Por estas razones, es necesario definir una estrategia que dirija el proceso de evaluación hacia la selección correcta de las métricas a valorar. Para tal motivo, frecuentemente, son usados paradigmas de medidas, a fin de seleccionar de una manera más eficiente las métricas que puedan ser aplicables, de forma que estas respondan directamente a los intereses de la evaluación *(NEUKIRCHEN et al., 2008)*. Este es el caso del enfoque Meta-Pregunta-Métrica o GQM por sus siglas en inglés *(OMG, 2012), (BASILI et al., 2014), (BASILI et al., 2010)*.

GQM proporciona una manera útil para definir mediciones tanto del proceso como de los resultados de un proyecto. Considera que un programa de medición puede ser más satisfactorio si es diseñado teniendo en mente las metas u objetivos perseguidos. Este paradigma provee un enfoque que permite establecer un sistema de medición por objetivos para el desarrollo de software, en el cual, el equipo empieza con las metas organizacionales, define los objetivos de medición, plantea preguntas para hacer frente a las metas, e identifica métricas que dan respuestas a las preguntas *(OMG, 2012)*.

Este paradigma tiene un carácter jerárquico, de tal forma que las «metas» identifican lo que se quiere lograr; las «preguntas», dicen si se están satisfaciendo los objetivos y ayuda a comprender cómo interpretarlos; y las «métricas» identifican las mediciones que son necesarias para responder a las preguntas y cuantificar el objetivo *(BASILI et al., 2014)*. Lo mencionado en este sentido puede verse en la Figura 1.

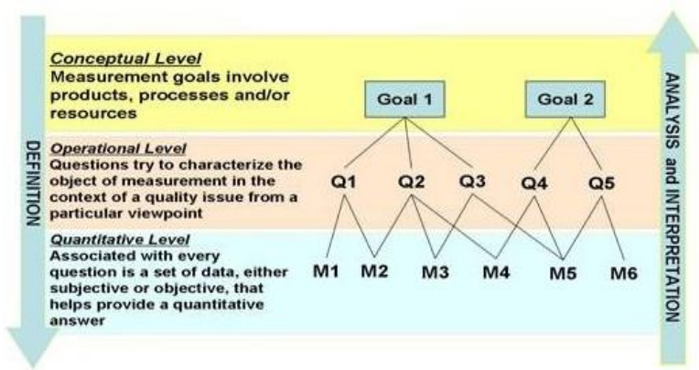

Figura 1. Fundamento jerárquico de GQM *(BASILI et al., 1994)*.

Para su implementación, algunos autores afirman que la aplicación de GQM debe ser vista en términos de fases de las actividades que están integradas con la planificación y gestión de proyectos y que tienen relaciones de dependencia entre sí *(VAN SOLINGEN, 1999)*. Estas fases antes mencionadas son: Planificación, Definición, Recolección de datos, Interpretación; y las relaciones que existen entre ellas pueden verse en la Figura 2.

Revista Cubana de Ciencias Informáticas
Vol. 9, No. Especial CCIA 17, 2015
RNPS: 2301 | ISSN: 2227-1899
http://rcci.uci.cu
Pág. 76-84

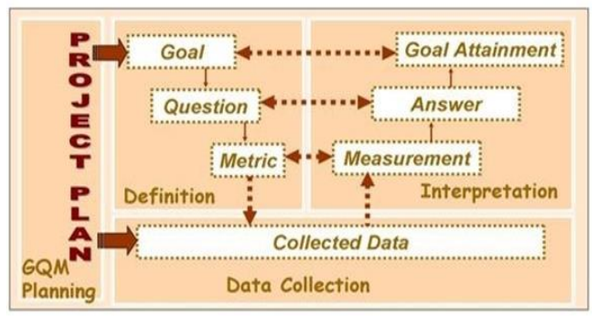

Figura 2. Fases para la implementación GQM *(VAN SOLINGEN, 1999)*.

La Planificación GQM supervisa la implementación de GQM dentro del contexto del proyecto. Por eso no precede a otras fases, más bien, interactúa con estas. Inicialmente la Planificación GQM establece cómo la fase de Definición debe ser implementada y cómo debe comportarse. A continuación, la planificación GQM utiliza la salida de la fase de definición de base para la planificación de los mecanismos de recogida de datos, así como para el análisis y la interpretación. Por tanto, la planificación GQM proporciona los artefactos que sirven como una guía para otras fases, y proporciona la integración necesaria con la planificación del proyecto. Según *(VAN SOLINGEN, 1999)* y *(BASILI et al., 2010)* estos artefactos antes mencionados son:

«Plan GQM»: documento que contiene cada meta de medición y su correspondiente desglose en preguntas y métricas, preservando así las relaciones de los objetivos con las preguntas, y de estas con las métricas. Este documento sienta las bases para avanzar a través de las otras fases de GQM.

«Plan de medidas»: documento donde se define la base real de las medidas que son necesarias para generar las métricas definidas en el plan GQM. También establece los procedimientos detallados para la recogida de los datos de medición y la generación de los indicadores identificados. Debe abordar lo que se recopila de los datos, cómo se va a recoger, por quién y cuándo. El Plan de Medidas incluye una descripción de los medios necesarios para identificar, recoger y validar los datos, por lo cual orienta las actividades de la fase de recopilación de datos.

«Plan de Análisis»: documento donde se precisa la forma de analizar, agregar y presentar los datos de medición recogidos en formas que sean significativas para los grupos de interés. En él se sientan las bases para la fase de interpretación de GQM, proporcionando orientación sobre cómo la información debe ser organizada para facilitar su uso y asegurarse de que el foco permanece en las metas. Un punto importante es que la planificación, para el análisis y la interpretación, se debe hacer antes de la recogida de datos de manera que esté claramente ligada a los objetivos y no fluctúe dependiendo de los valores reales de los datos recogidos. El plan de análisis define el nivel apropiado de abstracción para la presentación de los datos, pues contiene valores esperados de métricas, gráficos y diagramas, lo cual permite a los miembros del equipo de proyecto ir comparando los datos que van obteniendo.

Aplicar GQM como soporte para la evaluación de la calidad de los productos de software en un ambiente de pruebas tercerizadas, donde el equipo de evaluación no forma parte del equipo de proyecto, requiere la creación de un espacio en el contexto del proyecto que permita definir los artefactos necesarios que guíen GQM. En este espacio también es entregada toda la documentación del producto de software útil para la evaluación, dígase requerimientos funcionales y no funcionales, manuales de usuario, etc., definiéndose además el alcance de la prueba de acuerdo a las características del producto de software o la fase de desarrollo en que se encuentra.

Revista Cubana de Ciencias Informáticas
Vol. 9, No. Especial CCIA 17, 2015
RNPS: 2301 | ISSN: 2227-1899
http://rcci.uci.cu
Pág. 76-84

Resultados y Discusión

Especificación de GQM como herramienta de pruebas usando UML

Como se explicaba antes, este uso del enfoque GQM para realizar pruebas de software se realiza en un laboratorio que se dedica a probar software y componentes de software a terceros, por lo que se debe lograr que en la reunión inicial los participantes (desarrolladores y probadores) usen un lenguaje común. Atendiendo a que los desarrolladores presentan como parte de la documentación de la solicitud diagramas de UML y otros artefactos del Proceso Unificado de Software, entonces este debe ser el lenguaje para la especificación de todos los elementos involucrados en el proceso.

La especificación de GQM como herramienta de pruebas usando UML puede dividirse en dos etapas: una en la que se representa la estructura de este enfoque usando artefactos de UML, en particular el diagrama de paquetes para representar su estructura general (ver Figuras 3 y 4), por lo que no debe cambiar mientras no se realicen cambios en los elementos a tener en cuenta. La segunda, es la que cambia para cada software a probar y consiste en incluir los elementos del perfil de pruebas de UML *(OMG, 2013)* a los artefactos usados por los desarrolladores.

En la Figura 3 se pueden observar los paquetes que conforman GQM y su relación con el paquete usado para especificar la norma ISO/IEC 9126 *(ISO/IEC, 2001)* y la estructura interna de este paquete usando un diagrama de clases se presenta en la Figura 4.

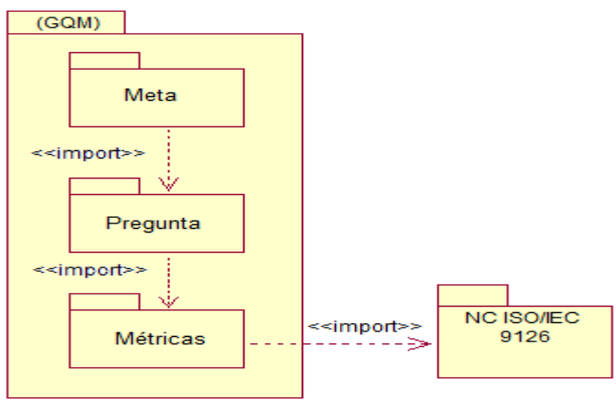

Figura 3. Diagrama de paquetes de UML para representar estructura de GQM.

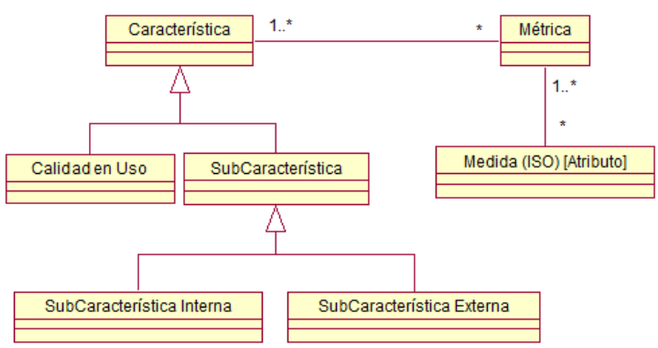

Figura 4. Diagrama clases de UML para representar la estructura del paquete NC ISO/IEC 9126.

Revista Cubana de Ciencias Informáticas
Vol. 9, No. Especial CCIA 17, 2015
RNPS: 2301 | ISSN: 2227-1899
http://rcci.uci.cu
Pág. 76-84

Uso del perfil de UML (UTP)

Una de las principales ventajas de UML es ser un lenguaje general, aunque en ocasiones esto puede ser una desventaja por no poder representar cabalmente situaciones particulares o dominios específicos. Por ello, desde su versión 1.3 ofrece la posibilidad de extensión mediante dos mecanismos, uno de los cuales es el uso de perfiles, que permite crear nuevos dialectos definiendo estereotipos, valores etiquetados y restricciones.

El perfil de pruebas, versión 1.2, permite especificar, visualizar, analizar, diseñar y construir artefactos usados en los procesos de pruebas, independientemente de cualquier metodología, dominio o tipo de sistema *(OMG, 2013)* para lo cual dispone de una biblioteca de tipos predefinidos para especificar la arquitectura, el comportamiento, los datos y la gestión relacionada con el proceso de pruebas.

En la Figura 5 puede verse la relación entre los paquetes definidos anteriormente y el perfil de pruebas, especificando el sistema objeto de prueba (SUT, *System Under Test*, por sus siglas en inglés).

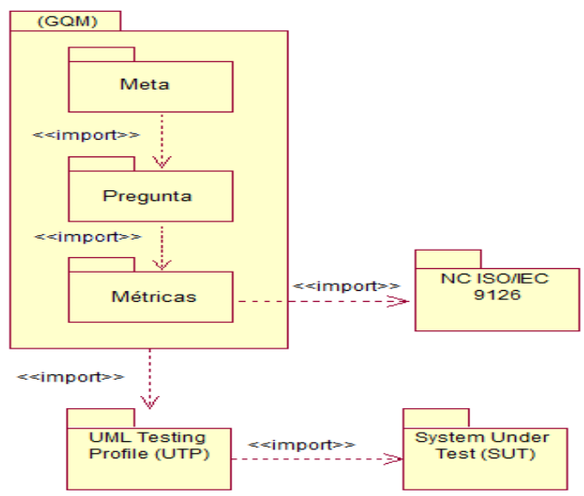

Figura 5. Diagrama de paquetes de UML relacionando GQM y el perfil de pruebas.

Los diagramas que propone el perfil de pruebas incluyen el sistema o componente a probar como una caja negra, pero deben establecerse las relaciones con los diferentes tipos de prueba a realizar y los resultados, por lo que en este caso, se trabaja en una taxonomía de tipos de sistemas o componentes a probar acorde con las definiciones de GQM. Es por ello que en esta primera etapa solo se presenta un diagrama en el que aparecen las relaciones generales entre todos los elementos involucrados en la propuesta general.

Como se puede apreciar en la Figura 6, se muestran los elementos que intervienen en el ambiente de prueba y sus interrelaciones. El proceso de prueba incluye desde la reunión inicial entre probadores y desarrolladores en la que se acuerda todo lo referido a fechas, especificaciones del sistema, casos de prueba, requisitos, etc., hasta la obtención de los resultados. Durante todo el proceso se hará uso de las definiciones realizadas usando el enfoque GQM y de la información a registrar y existente en las bases de datos de los servidores.

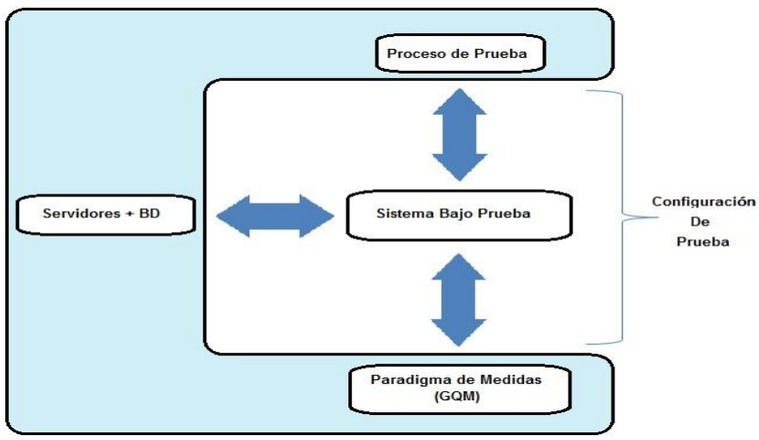

Figura 6. Ambiente de pruebas.

Este perfil da la posibilidad de representar al evaluador *«Arbiter»*, de asignar valores a los resultados de las pruebas, mediante el tipo *«Verdict»*, todo lo relacionado con el tiempo de la prueba, etc.

Si bien se trata de una propuesta, la posibilidad de tener resultados satisfactorios está basada en el uso de este perfil para la especificación de pruebas en disímiles áreas como la de dispositivos *bluetooth (RU DAI et al., 2004)*, líneas de productos de software *(PÉREZ et al., 2011)*, solicitud de créditos *(FAROOQ, 2012)*, etc.

En *(RU DAI et al., 2004)* aparece una metodología para el uso de los elementos de este perfil y en *(DINIZ DA COSTA, 2013)*, se hace una extensión de este perfil para incorporar coordinación entre las diferentes pruebas a un sistema y demuestran su funcionamiento a través de un sistema de comercio electrónico con el uso de multi-agentes.

Conclusiones

El uso de UML como lenguaje de comunicación entre desarrolladores y probadores permite disminuir las barreras de comunicación cuando se usan elementos no tradicionales en este proceso, como es el enfoque GQM con sus abstracciones.

Poder usar extensiones de UML mediante el uso de perfiles está demostrado que permite poder tener más elementos de un dominio específico y en particular este de las pruebas, se ha demostrado que existe una comunidad atenta a su evolución.

Referencias

ACOSTA-MENDOZA, N.; GAGO-ALONSO, A.; MEDINA-PAGOLA, J.E. Frequent approximate subgraphs as features for graph-based image classification. Knowledge-Based Systems, 2012, 27, 381-392.

BORGELT, C. Canonical forms for frequent graph mining. In: 30th Annual Conference of the German Classification Society, Universitat Berlin, Springer-Verlag, 2006, 337-349.
CONWAY, J.; GUY, R. The Book of Numbers. New York, Copernicus, Springer-Verlag, 1996. 310 pages. BAPAT, R. Graphs and Matrices. New Delhi, Hindustan Book Agency, India, 2010. 171 pages

DIESTEL, R. Graph Theory. Electronic Edition, Springer-Verlag, New York, 2000.

GAGO-ALONSO, A.; MEDINA-PAGOLA, J.E.; CARRASCO-OCHOA, J.A.; MARTÍNEZ-TRINIDAD, J.F. Full duplicate candidate pruning for frequent connected subgraph mining. Integrated Computer-Aided Engineering, 2010a, 17(3): 211-225.

GAGO-ALONSO, A.; PUENTES-LUBERTA, A.; CARRASCO-OCHOA, J.A.; MEDINA-PAGOLA, J.E.; MARTÍNEZ-TRINIDAD, J.F. A new algorithm for mining frequent connected subgraphs based on adjacency matrices. Intelligence Data Analysis, 2010b, 14 (3), 385-403.

HARARY, F.: Graph Theory. Addison-Wesley, Reading, MA, 1969, 178-180.

HUAN, J.; WANG, W.; PRINS, J. Efficient mining of frequent subgraphs in the presence of isomorphism. In: 3rd IEEE International Conference on Data Mining, Melbourne, FL, IEEE Computer Society, 2003, 549-552.

INOKUCHI, A.; WASHIO, T.; MOTODA, H. An apriori-based algorithm for mining frequent substructures from graph data. In: 4th European Conference on Principles of Data Mining and Knowledge Discovery, Lyon, France, Springer-Verlag, 2000, 13-23.

JIANG, C.; COENEN, F.; ZITO, M. A survey of frequent subgraph mining algorithms, The Knowledge Engineering Review, 2013, 28: 75-105.

KURAMOCHI, M.; KARYPIS, G. Frequent Subgraph Discovery. In: 1st IEEE International Conference on Data Mining, San Jose, CA, IEEE Computer Society, 2001, 313-320.

LI, R.; WANG, W.: REAFUM: Representative Approximate Frequent Subgraph Mining. In: SIAM International Conference on Data Mining, Vancouver, BC, Canada, 2015. ISSN 2167-0099.

MANSO, M.; PELLINO, S.; PETROSINO, A.; ROZZA, A. A Novel Graph Embedding Framework for Object Recognition. In: Computer Vision - ECCV 2014 Workshops, 2014, 341-352.

MCKAY, B. D. Practical graph isomorphism. Congressus Numerantium, 1981, 30: 45-87.

NIJSSEN, S.; KOK, J.N. A quickstart in frequent structure mining can make a difference. In: 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, ACM, 2004, 647-652.

VO, B.; NGUYEN, D.; NGUYEN, T.L. A Parallel Algorithm for Frequent Subgraph Mining. In: International Conference on Computer Science, Applied Mathematics and Applications, Metz, France, 2015, 163-173.

YAN, X.; HAN, J. gSpan: Graph-based substructure pattern mining. In: International Conference on Data Mining, Maebashi, Japan, IEEE, 2002, 721-724.