Recuperación de Arquitecturas de Software: Un Mapeo Sistemático de la Literatura

Martín E. Monroy⁽¹⁾, José L. Arciniegas⁽²⁾ y Julio C. Rodríguez⁽¹⁾

(1) Universidad de Cartagena, Facultad de Ingeniería, Programa de Ingeniería de Sistemas, Piedra de Bolívar, Bolívar – Colombia (e-mail: mmonroyr@unicartagena.edu.co, jrodriguezz@unicartagena.edu.co)

(2) Universidad del Cauca, Departamento de Telemática, Sector Tulcán, Cauca – Colombia (e-mail: jlarci@unicauca.edu.co)

Recibido Dic. 15, 2015; Aceptado Feb. 17, 2016; Versión final Mar. 3, 2016, Publicado Oct. 2016

Resumen

En este artículo se presentan los resultados de un mapeo sistemático de la literatura sobre el tema de recuperación de vistas arquitectónicas de sistemas software, el cual se llevó a cabo aplicando una propuesta metodológica establecida en la literatura. Se identificaron los estudios existentes sobre el tema, especificando el tipo de estudio realizado, su propósito, las técnicas de recuperación que utilizan, los aspectos y elementos de la arquitectura que recuperan y los mecanismos que utilizan para representar las vistas arquitectónicas recuperadas, entre otros aspectos. Se concluye que sería útil contar con un mecanismo unificado y consistente para representar los resultados obtenidos por las técnicas de recuperación de arquitecturas, que facilite su interpretación, análisis y reutilización.

Palabras clave: recuperación de arquitectura; ingeniería inversa; mapeo sistemático; arquitectura de software.

Software Architecture Recovery: A Systematic Mapping Study

Abstract

This paper presents the results of a systematic mapping review about the recovery of architectural views from a software system, which was carried out using a methodological proposal found in the literature. Existing studies on the subject were identified, specifying the type of research, its purpose, the used recovery techniques, the aspects and architectural elements that is recovered, and the models used to represent the recovered architectural views, among others. It is concluded that it would be useful to have a unified and consistent mechanism for representing the results obtained by architectural recovery techniques, which facilitates their interpretation, analysis and reuse.

Keywords: architecture recovery; reverse engineering; systematic review; software architecture.

INTRODUCCIÓN

La evolución del software se logra a través del mantenimiento del producto, sin embargo, esta actividad representa altos costos para el proceso de desarrollo (Koskinen, 2010). Por otra parte, Cuadrado et al. (2008) afirman que aproximadamente el 50% del tiempo de mantenimiento se utiliza para el entendimiento del sistema. Esta situación se presenta porque el conocimiento sobre el sistema se encuentra registrado en forma explícita en la documentación o de manera implícita en la mente de los expertos que lo desarrollaron. Desafortunadamente, con frecuencia no se dispone de dicho conocimiento, sobre todo cuando se trata de sistemas heredados que carecen de documentación o se encuentra desactualizada y no se cuenta con el equipo de expertos que lo crearon (Demeyer et al., 2009; Cuenca et al., 2008).

Por otra parte, debido al cambio continuo que genera el círculo virtuoso entre los avances científicos y tecnológicos, cada día se hace más imperante para un producto software tener capacidad de mantenimiento: capacidad del producto software para ser modificado efectiva y eficientemente, debido a necesidades evolutivas, correctivas o perfectivas (ISO/IEC 25000, 2005). Esto se logra si el producto es modular y tiene capacidad para ser: analizado, probado, modificado y reutilizado. Adicionalmente, para efectos de control de calidad se requiere garantizar la coherencia entre el código fuente y los modelos arquitectónicos y de diseño propuestos (Müller et al., 2000).

Ante esta situación, la ingeniería inversa se ha convertido en la principal solución, porque permite recuperar el conocimiento de un producto software para facilitar su comprensión (Tonella et al., 2007; Rugaber y Stirewalt, 2004). También facilita la reutilización de código, el control de clonación no autorizado, el análisis y entendimiento del código malicioso (Treude et al., 2011). Además, hace posible la coexistencia sincronizada del sistema y los modelos que lo representan. Esto la ha convertido en uno de los mayores retos actuales de la ingeniería de software (Favre et al., 2009).

La recuperación de arquitectura es un enfoque de la ingeniería inversa cuyo principal objetivo es reconstruir las vistas arquitectónicas de un producto software (Ducasse y Pollet, 2009). Según el estándar ISO/IEC/IEEE 42010:2011, la arquitectura comprende los conceptos o propiedades fundamentales de un sistema en su entorno plasmados en sus elementos, relaciones, y en los principios que rigen su diseño y evolución. La descripción de la arquitectura es un producto de trabajo utilizado para expresar la arquitectura. Un punto de vista arquitectónico es un producto de trabajo en el que se establecen los convenios para la construcción, interpretación y uso las vistas arquitectónicas para enmarcar los intereses específicos sobre el sistema. Una vista arquitectónica es un producto de trabajo que expresa la arquitectura de un sistema desde la perspectiva de un conjunto de intereses específicos del sistema, para lo cual utiliza un tipo de modelo como forma de representación (grafos, diagramas, redes de Petri, etc).

Existen varias propuestas que definen los puntos de vista y las vistas que describen la arquitectura de un sistema software (Hofmeister et al., 1999; Kruchten, 1995; Smolander, 2002; ISO/IEC/IEEE, 2011), lo que obliga a seleccionar un enfoque para representar la arquitectura, el cual depende del contexto y la situación (Smolander, 2002). Esta circunstancia incrementa la complejidad del proceso de recuperación de la arquitectura, porque exige para cada enfoque tareas adicionales, que se podrían obviar si existiera unidad de criterios para identificar y representar los elementos de la arquitectura.

Con el propósito de afrontar dicha complejidad Koschke hace una revisión de la situación sobre el tema en el año 2005, el cual actualizó en el 2009, en la que concluye que sería útil un catálogo sobre las técnicas de recuperación de arquitectura detallando cuándo y cómo deben ser utilizadas. De igual forma, Ducasse y Pollet hacen un estudio sobre el tema en el año 2007, el cual actualizaron en el 2009, obteniendo como resultado una clasificación de los enfoques de recuperación de arquitectura, estructurada en los siguientes ejes: el propósito, el proceso, las entradas, las técnicas y las salidas. A pesar de la profundidad de estos estudios y de la rigurosidad científica con que se hicieron, no se detallan los aspectos relacionados con los elementos de la arquitectura que se recuperan ni los mecanismos que utilizan para representarlos.

Conocer los elementos de la arquitectura que se recuperan y los mecanismos que utilizan las técnicas de recuperación de arquitectura para representarlos (Ducasse y Pollet, 2009), es importante porque facilita la interpretación de los resultados obtenidos en el proceso de recuperación, reflejando las vistas arquitectónicas recuperadas. Además, hace más eficiente el trabajo de investigación en este campo, porque brinda la oportunidad de integrar las propuestas existentes; al mismo tiempo que facilita la definición de lineamientos y criterios que hagan posible la unificación de los mecanismos de representación de los elementos recuperados por cada técnica (Favre, 2004). Esto también contribuye a la definición de técnicas de análisis arquitectónico sobre los resultados obtenidos en los procesos de recuperación de arquitecturas (Zhang et al., 2009; Peake et al., 2013).

Por lo anterior, se planteó como objetivo de la presente investigación identificar los elementos de la arquitectura que recuperan las técnicas de recuperación de arquitectura y los mecanismos que utilizan para representarlos. Para lograr dicho objetivo se aplicó la metodología que se explica a continuación, logrando los resultados descritos posteriormente. Finalmente se hace un análisis de los resultados y se presentan las conclusiones.

METODOLOGÍA

Se hizo un mapeo sistemático de la literatura aplicando la propuesta metodológica establecida por Kitchenham et al. (2011) y Petersen et al. (2008), que consistió en la realización de los siguientes pasos: 1) definición de las preguntas de investigación, 2) búsqueda de los estudios primarios, 3) selección de los estudios aplicando los criterios de inclusión y exclusión establecidos, 4) clasificación de los artículos y 5) extracción y agregación de datos. El propósito del estudio consistió en identificar las técnicas de recuperación de arquitectura existentes hasta el momento, los elementos de la arquitectura que recuperan y los tipos de modelos que utilizan para representarlos, con el ánimo de establecer los desafíos existentes al respecto. Para orientar el mapeo sistemático se definieron las siguientes preguntas de investigación:

1) ¿Cuáles son los estudios existentes acerca de la recuperación de vistas arquitectónicas?. Esto permite establecer un inventario sobre los trabajos realizados en el campo de la recuperación de arquitecturas, a partir del cual se fundamentan los resultados de esta investigación; 2) ¿Qué tipos de estudio se han realizado?; 3) ¿Cuál es el propósito de cada estudio?, al responder las dos últimas preguntas se pueden clasificar los trabajos realizados según su tipo y propósito, identificando con claridad aquellos que corresponden a propuestas conceptuales, para hacer un análisis focalizado sobre las técnicas de recuperación que utilizan, los elementos de la arquitectura que recuperan y los tipos de modelos que aplican como forma de representación, para lo cual se plantean las preguntas; 4) ¿Qué técnicas de recuperación utilizan aquellos estudios que plantean propuestas conceptuales para recuperar arquitecturas?; 5) ¿Qué aspectos de la arquitectura recupera cada estudio?; 6) ¿Qué mecanismos utilizan para representar las vistas arquitectónicas recuperadas?; 7) ¿Cuáles son los principales investigadores en el tema?; y 8) ¿Cuáles son principales medios donde se publican estudios sobre el tema?, con la intención de identificar los expertos en el tema y los medios que más han publicado al respecto, facilitando el trabajo a futuras investigaciones.

La búsqueda de los estudios primarios se realizó en las siguientes bases de datos bibliográficas: IEEEExplore, ACM Digital Library, SpringerLink, ScienceDirect, Scopus, Wiley online Library y CiteSeer. En cada una de ellas se aplicaron las expresiones de búsquedas obtenidas a partir de la formulación: $[(cadena1 \text{ AND } cadena3) \text{ OR } (cadena4 \text{ AND } cadena1) \text{ OR } (cadena4 \text{ AND } cadena2)]$, donde: $cadena1 = \text{"architecture"}$, $cadena2 = \text{"architectural views"}$, $cadena3 = [(\text{"retrieval"} \text{ OR } \text{"recovery"} \text{ OR } \text{"reconstruction"} \text{ OR } \text{"discovery"})]$ y $cadena4 = [(\text{"Reconstructing"} \text{ OR } \text{"reverse"} \text{ OR } \text{"reversing"} \text{ OR } \text{"retrieving"} \text{ OR } \text{"recovering"} \text{ OR } \text{"capturing"} \text{ OR } \text{"discovering"} \text{ OR } \text{"runtime"} \text{ OR } \text{"extract"} \text{ OR } \text{"extracting"} \text{ OR } \text{"mining"})]$.

Esto implicó realizar 26 búsquedas en cada base de datos, para un total de 182 búsquedas que se realizaron en los campos: título, resumen y palabras clave, obteniendo un total de 4050 documentos encontrados. Se seleccionaron inicialmente los documentos que cumplieron con los siguientes criterios de inclusión: Capítulos de libros, reportes técnicos, literatura gris y artículos de revistas que presenten resultados de estudios empíricos, sin restricción de fecha de publicación. Para depurar el listado obtenido se aplicaron los siguientes criterios de exclusión: Documentos duplicados, aquellos cuyo tema principal no es la recuperación de arquitecturas de software, los que hayan sido publicados en revistas o actas de conferencias no arbitradas y los documentos de opinión. En la tabla 1 se relaciona la cantidad de artículos seleccionados por cada base de datos consultada.

Tabla 1: Cantidad de artículos seleccionados por base de datos bibliográfica

Base de datos	Encontrados	Seleccionados	Porcentaje
IEEEExplore	223	106	45%
Scopus	443	47	20%
ACM	52	29	12%
CiteSeer	164	25	11%
ScienceDirect	22	13	5%
Wiley online Library	3059	10	4%
SpringerLink	87	6	3%
Total	4050	236	100%

Para evaluar la fiabilidad de la aplicación de los criterios de inclusión/exclusión, se seleccionó una muestra de 70 documentos encontrados en la búsqueda de IEEEExplore, correspondiente al 36% del total de documentos encontrados en esta base de datos y al 75% de los documentos seleccionados en la misma. Esta muestra fue analizada independientemente por dos revisores para determinar la fiabilidad inter-evaluador mediante el estadístico Kappa Cohen (Carletta, 1996). Se obtuvieron los resultados presentados en la tabla 2, dando un grado de fiabilidad satisfactorios (Kappa = 0,821). Esto indica la existencia de criterios claros que no denotan divergencias representativas entre los revisores (Fleiss et al., 2013).

Tabla 2: Fiabilidad de los criterios de inclusión/exclusión

		Evaluador 1		Total
		Si	No	
Evaluador 2	Si	39	2	41
	No	4	25	29

La clasificación de los artículos se realizó tomando como referente las preguntas de investigación establecidas, lo que permitió llegar a las siguientes categorías: Tipos de estudio, técnicas de recuperación utilizadas, aspecto de la arquitectura recuperado, mecanismos utilizados para representar las vistas arquitectónicas recuperadas, principales investigadores en el tema y principales medios de publicación. La extracción de los datos fue realizada por cada uno de los autores en forma independiente a partir de la lectura de cada uno de los documentos, llenando los datos correspondientes a las categorías mencionadas previamente. Posteriormente se hizo la validación mediante la confrontación de los datos obtenidos, identificando y resolviendo las diferencias encontradas.

RESULTADOS

Los resultados que responden a las seis primeras preguntas establecidas se encuentran detallados en la tabla 3, que constituye un inventario sobre los trabajos realizados en el campo de la recuperación de arquitecturas, desde comienzos de la década de los noventa hasta la realización de esta investigación. En la primera columna aparecen los autores con un número consecutivo, que sirve para ubicar en la tabla 4 el documento que se ha analizado. De esta manera se da respuesta a la primera pregunta de investigación. En la segunda columna se presenta el año de publicación para organizar cronológicamente en forma ascendente los documentos encontrados. En la tercera columna está el tipo de estudio, tomando como posibles valores los establecidos por Tonella et al., (2007): No experimento (NE), reporte de experiencia (RE), estudio de caso (EC), cuasi-experimento (QE), estudio a partir de la observación (EO), revisión sistemática (RS) y experimento aleatorio (EA), lo que da respuesta a la segunda pregunta.

La cuarta columna muestra el propósito del estudio (Tonella et al., 2007): Propuesta conceptual (PC), prueba de conceptos (PbC), comparación (C) y revisión de literatura (RL), dando respuesta a la tercera pregunta. Las preguntas cuatro, cinco y seis sólo aplican para aquellos estudios cuyo propósito principal es plantear una propuesta conceptual con respecto a la recuperación de arquitecturas. La quinta columna indica la técnica de recuperación utilizada. En la realización del mapeo se encontraron las siguientes técnicas: Análisis dinámico (AD), análisis estático (AE), anotaciones en el código (AC), atributos de calidad (AtC), clustering (C), coincidencia de patrones (CP), concept location (CL), heurística (H), inteligencia artificial (IA), minería de datos (MD), reconocimiento de patrones (RP), reflexión (R), teoría de grafos (TG). Con esto se responde a la cuarta pregunta de investigación. Para aquellos estudios que no hacen claridad sobre la técnica aplicada se utiliza la notación NI que significa no indica. La sexta columna presenta el aspecto de la arquitectura que se recupera: Estructura (E), comportamiento (C) o ambas (A) (Canfora et al., 2011), mientras que la séptima columna dice el mecanismo que utilizan para representar las vistas arquitectónicas recuperadas, quedando resueltas la quinta y sexta pregunta.

Tabla 3: Relación de estudios seleccionados

Autores	Año	Tipo estudio	Propósito	Técnica Recuperación	Aspecto	Mecanismo usado para representar las vistas
Bass, L. (1)	1992	RE	PC			
Müller, H.A y M. Orgun (2)	1993	EC	PC	TG	E	Grafos
Harris, D. et al. (3)	1995	RE	PC	C	E	Grafos
Chase, M.P. et al. (4)	1996	EC	PbC			
Gall, H. et al. (5)	1996	RE	PC	H	A	Grafos
Harris, D. et al. (6)	1996	RE	PC	C	E	Grafos
Prywes, N. y P. Rehmet (7)	1996	EC	PC	H	A	Grafos
Mendonca, N.C. y J. Kramer (8)	1996	RE	C			
Mendonca, N.C. y J. Kramer (9)	1997	RE	C			

Tabla 3: (Continuación)

<i>Autores</i>	<i>Año</i>	<i>Tipo estudio</i>	<i>Propósito</i>	<i>Técnica Recuperación</i>	<i>Aspecto</i>	<i>Mecanismo usado para representar las vistas</i>
Mendonca, N.C. y J. Kramer (10)	1997	RE	C			
Eixelsberger, W. et al. (11)	1997	RE	PC	H	A	Grafos
Koen De Hondt (12)	1998	EC	PC	C	A	Grafos
Mendonca, N.C. y J. Kramer (13)	1998	EX	PbC			
Duenas, J.C. et al. (14)	1998	RE	PbC			
Mendonca, N.C. y J. Kramer (15)	1998	EX	PC	CP	A	Grafos
Cook, J. E. y A.L. Wolf (16)	1998	RE	PC	H	C	Redes de Petri
Jerding, D. y Rugaber (17)	1998	RE	PC	H	C	Diagrama de eventos
Chase, M.P. et al. (18)	1998	EC	PbC			
Kazman, R. (19)	1998	RE	C			
Melissa, P. C. et al. (20)	1998	EX	PbC			
Chung-Horng Lung (21)	1998	EC	PC	C	E	Grafos
Eixelsberger, W. et al. (22)	1998	EC	PC	AtC	E	ASDL
Bowman, I. y R. Holt (23)	1998	EC	PC	H	A	Grafos
Guo G. Y. et al. (24)	1999	EC	PC	RP	A	Grafos
Fiutem, R. et al. (25)	1999	EX	PC	TG	E	Grafos
Mendonca, N.C. y J. Kramer (26)	1999	EC	PC	TG	A	Grafos
von Mayrhauser, A. et al. (27)	1999	RE	PbC			
Richner, T. y S. Ducasse (28)	1999	EC	PC	C	A	Grafos
Carriere, S. et al. (29)	1999	RE	PbC			
Woods, S. et al. (30)	1999	RE	C			
Sartipi, K. et al. (31)	2000	EX	PC	C	E	Grafos
Bojic, D. y D. Velasevic (32)	2000	RE	PC	CL	A	UML
Bril, R. J. et al. (33)	2000	EX	PbC			
Bojic, D. y D. Velasevic (34)	2000	EC	PC	CL	A	UML
Sartipi, K. y K. Kostas (35)	2001	EX	PC	C	E	Grafos
Svetinovic, D. y M. Godfrey (36)	2001	RE	PC	H	A	No indica
Sartipi, K. (37)	2001	RE	PbC			
Mendonca, N.C. y J. Kramer (38)	2001	EC	PC	CP	A	Grafos
Lei, D. y N. Medvidovic (39)	2001	EC	PC	H	A	UML
Stoermer, C. y L. O'Brien (40)	2001	EC	PC	MD	A	Grafos
Gröne, B. et al. (41)	2002	EC	PbC			
Riva, C. (42)	2002	RE	PC	CL	C	Grafos
Ivkovic, I. y M.W. Godfrey (43)	2002	RE	PC	H	A	UML
Hassan, A.E. y R.C. Holt (44)	2002	RE	PC	C	A	UML
Riva, C. y J.V. Rodriguez (45)	2002	RE	PC	CL	A	UML
Riva, C. y Y. Yang (46)	2002	RE	PC	H	E	XML
Hogganvik, I. y E. Molstad (47)	2002	EC	PC	H	A	UML
Pinzger, M. y H. Gall (48)	2002	EC	PC	CP	E	Grafos
Stoermer, C. et al. (49)	2002	RE	PC			
Pinzger, M. et al. (50)	2002	EC	PbC			
Svetinovic, D. y M. Godfrey (51)	2003	EC	PC	H	A	No indica
Löwe, W y J. Lundberg (52)	2003	EX	PC	TG	E	Grafos
Sartipi, K. y K. Kostas (53)	2003	EC	PC	C	E	Grafos
Lutz, R. y G. Gannod (54)	2003	RE	PbC			
Kazman, R. et al. (55)	2003	RE	PC	C	A	Grafos
Lundberg, J. y W. Löwe (56)	2003	RE	PC	TG	E	Grafos
Qiao, B. et al. (57)	2003	RE	PC	H	E	UML y XMI
Bril, R. J. et al. (58)	2003	EC	PbC			
Ivkovic, I. y M.W. Godfrey (59)	2003	EC	PC	H	A	No indica
Riva, C. y C. del Rosso (60)	2003	EC	PC	H	E	No indica
Ettore, M. et al. (61)	2003	EX	PC	IA	A	Texto / Identificadores
Koschke, R. y D. Simon (62)	2003	EC	PC	R	E	Grafos
Li, Q. y P. Chen (63)	2003	RE	PC	IA	E	UML
Eisenbarth, T. et al. (64)	2003	EC	PC	CL	A	Grafos
Ran, A. y R. Lencevicius (65)	2003	EX	PC	H	C	Registro de tiempos
Stoermer, C. et al. (66)	2003	EC	PC	AtC	A	Grafos
Sartipi, K. y K. Kostas (67)	2003	EC	PC	CP	E	Grafos
Sartipi, K. y K. Kostas (68)	2003	EX	PC	CP	A	Grafos
Minmin, H. et al. (69)	2003	RE	PC	H	A	UML
Sartipi, K. (70)	2003	EC	PC	CP	E	Grafos
Jing, L. et al. (71)	2004	EX	PbC			

Tabla 3: (Continuación)

<i>Autores</i>	<i>Año</i>	<i>Tipo estudio</i>	<i>Propósito</i>	<i>Técnica Recuperación</i>	<i>Aspecto</i>	<i>Mecanismo usado para representar las vistas</i>
Lencevicius, R. y A. Ran (72)	2004	RE	PC	H	C	Registro de tiempos
Pinzger, M. et al. (73)	2004	EC	PC	H	A	UML, Grafos, Texto
Bauer, M. y M. Trifu (74)	2004	EX	PC	C	E	UML
Favre, Jean-Marie (75)	2004	RE	PC	C	A	Metamodelo
Andrey A. Terekhov (76)	2004	EC	PbC			
Hong, Y. et al. (77)	2004	EC	PbC			
Oiming, T. et al. (78)	2004	EC	PC	H	E	Grafos
Favre, Jean-Marie (79)	2004	RE	PC	C	A	UML
Zhang, X. et al. (80)	2004	RE	PC	TG	E	Grafos
Vasconcelos, A. y C. Werner (81)	2004	EC	PC	CP	C	UML
van Deursen, A. et al. (82)	2004	EC	PC	H	A	Grafos
Maqbool, O. y H. A. Babri (83)	2004	EX	PC	C	E	Grafos
Sarmiento, C. y S. Takahashi (84)	2004	EC	PbC			
Pashov, I. y M. Riebisch (85)	2004	EC	PC	CL	A	UML
Bril, R. J. et al. (86)	2005	EC	PC	R	E	Grafos
Li, Q. et al. (87)	2005	EC	PC	AD	A	Grafos / PSG
Wu, L. et al. (88)	2005	RE	PC	AD	A	Grafos
Bradley, S. et al. (89)	2005	RE	PbC			
Andritsos, P. y V. Tzerpos (90)	2005	EX	PC	C	A	Grafos
Maqbool, O. y H. A. Babri (91)	2005	EX	PC	C	E	Grafos
Haqqie, S. y A. A. Shahid (92)	2005	EC	PC	IA	E	Grafos
Sun C. et al. (93)	2005	EC	PC	H	A	UML
Koschke, R. (94)	2005	RS	RL			
O'Brien, L. et al. (95)	2005	EC	PC	H	E	Grafos
Gorton, I. and L. Zhu (96)	2005	RE	C			
Vasconcelos, A. y C. Werner (97)	2005	EX	PC	C	A	UML
Sartipi, K. et al. (98)	2006	EC	PbC			
Sartipi, K. et al. (99)	2006	EC	PC	C	A	UML
Arciniegas, J. et al. (100)	2006	EC	PC	CP	A	UML
Maqbool, O. y H. A. Babri (101)	2006	EX	PC	C	E	Grafos
Stringfellow, C. et al. (102)	2006	EC	PbC			
Schmerl, B. et al. (103)	2006	EC	PC	RP	C	ACME
Yu, P. et al. (104)	2006	RE	PC	R	C	ACME
Stoermer, C. et al. (105)	2006	EC	PC	AtC	A	Grafos
Bouneffa, M. et al. (106)	2006	RE	PC	NI	E	GXL, ACME
Lungu, M. et al. (107)	2006	EX	PC	AC	E	Grafos
Ronny, R. et al. (108)	2006	EC	PC	H	E	No indica
Vladimir Jakobac (109)	2006	EC	PC	C	A	Grafos
Gang, H. et al. (110)	2006	EX	PC	R	C	ABC/ADL
Lo, D. y K. Siau-Cheng (111)	2006	EX	PbC			
Lungu, M. y M. Lanza (112)	2006	EC	PbC			
Guo, J. et al. (113)	2006	EC	PC	AE	A	Grafos
Maweed, Y. et al. (114)	2006	RE	PC	H	E	GXL, ACME
Medvidovic, N. y V. Jakobac (115)	2006	EC	PC	C	A	UML
Lee, S. y S. Kang (116)	2006	EC	PC	R	A	Grafos
Sartipi, K. y N. Dezhkam (117)	2007	EC	PC	CP	A	Grafos
Vasconcelos, A. y C. Werner (118)	2007	EX	PC	AD	A	UML y XML
Maqbool, O. y H. A. Babri (119)	2007	EX	PC	IA	E	Grafos
Ping, Y. et al. (120)	2007	EX	PC	H	C	ACME
Maqbool, O. y H. A. Babri (121)	2007	EX	PbC			
Asif, N. (122)	2007	EC	PC	H	E	UML
Dugerdil, P. y S. Jossi (123)	2007	EX	PC	C	A	UML
Rasool, G. y N. Asif (124)	2007	RE	PC	H	A	Grafos
Stoermer, C. (125)	2007	EC	PC	AtC	A	No indica
Staiger, S. (126)	2007	EX	PC	H	E	IML
Pollet, D. et al (127)	2007	RS	RL			
Cuadrado, F. et al. (128)	2008	EC	PbC			
Abi-Antoun, M. (129)	2008	EC	PC	AC	C	Grafos
Yu Zhou (130)	2008	RE	PC	TG	C	Grafos
Roy, B. y T. Graham (131)	2008	EC	PC	H	E	Grafos
de Niz, D. (132)	2008	RE	PC			
Khodamoradi, K. et al. (133)	2008	RE	PC	H	E	ACME

Tabla 3: (Continuación)

<i>Autores</i>	<i>Año</i>	<i>Tipo estudio</i>	<i>Propósito</i>	<i>Técnica Recuperación</i>	<i>Aspecto</i>	<i>Mecanismo usado para representar las vistas</i>
Fontana, A. et al. (134)	2008	EX	PC			
Anton, J. et al. (135)	2008	EC	PC	H	E	Grafos
Buckley, J. y A. P. Legar (136)	2008	EC	PbC			
Chardigny, S. et al. (137)	2008	EC	PC	C	E	Grafos
Chardigny, S. et al. (138)	2008	EC	PC	C	E	Grafos
Zhiyong, H. et al. (139)	2008	RE	PC	R	C	Grafos
Abi-Antoun, M. y J. Aldrich (140)	2008	RE	PC	AC	C	Grafos
Abi-Antoun, M. (141)	2008	EC	PC	AC	C	Grafos
Fontana, A. et al. (142)	2008	EX	PbC			
Abi-Antoun, M. y J. Aldrich (143)	2008	RE	PbC			
Abi-Antoun, M. y J. Aldrich (144)	2008	RE	PbC			
Zhang, L. et al. (145)	2009	EC	PC	H	E	Grafos
Kang, S. et al. (146)	2009	EC	PC	H	A	Grafos
Riva, C. et al. (147)	2009	RE	PC	H	A	UML
Koschke, R. (148)	2009	RS	RL			
Bittencourt, R. y D. Guerrero(149)	2009	EC	PbC			
Santonu, S. et al. (150)	2009	EX	PC	C	E	Grafos
Frenzel, P. y R. Koschke (151)	2009	EC	PC	R	E	Grafos
Hasan, S. et al. (152)	2009	RE	PC	H	E	Grafos
Anquetil, N. et al. (153)	2009	EC	PC	H	E	Grafos
Feiler, P. (154)	2009	RE	PbC			
Pirzadeh, H. et al. (155)	2009	EX	PC			
Ducassee, S. y. Pollet (156)	2009	RS	RL			
Patel, C. et al. (157)	2009	EC	PC	C	E	UML
Abi-Antoun, M. y J. Aldrich (158)	2009	EC	PC	AE	C	Grafos
Abi-Antoun, M. y J. Aldrich (159)	2009	EC	PC	AE	C	Grafos
Fontana, A. et al. (160)	2010	EX	PC	IA	E	UML
Vazquez, G. et al. (161)	2010	EX	PC	AtC	E	UML
Fontana, A. et al. (162)	2010	RE	PC	IA	E	UML
Corazza, A. et al. (163)	2010	EC	PC	C	E	Grafos
Scanniello, G. et al. (164)	2010	EC	PC	TG	E	UML y XML
Tae-hyung, K. et al. (165)	2010	EC	PC	R	C	Grafos
Abi-Antoun, M. y J. Barnes (166)	2010	RE	PC	AC	C	ACME
Scanniello, G. et al. (167)	2010	EC	PC	TG	E	UML y XML
Risi, M. et al. (168)	2010	EX	PC	C	E	Grafos
Verjus, H. et al. (169)	2010	RE	PC	C	E	Grafos
El Handouni, A. et al. (170)	2010	RE	PC	CL	E	Grafos
Abi-Antoun, M. et al. (171)	2010	EX	PbC			
Schrettner, L. et al. (172)	2010	RE	PC	NI	E	Grafos
Yuxin, W. et al. (173)	2010	EX	PC	C	E	Grafos
Akthar, S. y S. Rafi (174)	2010	EX	PC	CP	E	Grafos
Koschke, R. (175)	2010	EC	PC	R	E	Grafos
Woollard, D. et al. (176)	2010	EX	PC	C	E	Grafos
Qifeng, Z. et al. (177)	2010	EX	PC	C	E	Grafos
de Brito, H. et al. (178)	2010	EC	PC	TG	C	Grafos
Maras, J. et al. (179)	2010	RE	PbC			
Muhammad, S. et al. (180)	2010	EX	PbC			
Şora, I. et al. (181)	2010	EX	PC	C	E	Grafos
Chardigny, S. y A. Seriai (182)	2010	RE	PC	C	E	Grafos
Akthar, S. y S. Rafi (183)	2010	EX	PC	TG	E	Grafos
Yongwang, Z. et al. (184)	2010	RE	PC	TG	C	Grafos
Scanniello, G. et al. (185)	2010	EC	PC	C	E	Grafos
Abi-Antoun, M. y Z. Hailat (186)	2011	EC	PbC			
Seriai, A. y S. Chardigny (187)	2011	EC	PC	IA	E	UML
Ros, J.P. y R. S. Sangwan (188)	2011	RE	PC	H	E	CRD tree
Fontana, A. y M. Zanoni (189)	2011	EX	PC	IA	E	UML
Callo, T. (190)	2011	EC	PC	AD	C	XML
Callo, T. et al. (191)	2011	EC	PC	H	C	XML
Schmidt, F. y S. MacDonell (192)	2011	EX	PC	C	E	No indica
Sajani, H. et al. (193)	2011	EC	PC	C	E	Grafos
Ganesan, D. et al. (194)	2011	EC	PC	H	A	Grafos
Vasconcelos, A. y C. Werner(195)	2011	EC	PC	AD	A	UML y XML

Tabla 3: (Continuación)

<i>Autores</i>	<i>Año</i>	<i>Tipo estudio</i>	<i>Propósito</i>	<i>Técnica Recuperación</i>	<i>Aspecto</i>	<i>Mecanismo usado para representar las vistas</i>
Naseem, R. et al. (196)	2011	EX	PC	C	E	Grafos
Lehmann, G. et al. (197)	2011	RE	PC			
Erdemir, U. et al. (198)	2011	EX	PC	C	E	Grafos
Pavankumar, K. et al. (199)	2011	EX	PC	MD	E	Grafos
Nagy, C. et al. (200)	2011	RE	PbC			
Constantinou, E. et al. (201)	2011	EX	PC	C	E	Grafos
Bibi, M. y O. Magboll (202)	2011	EX	PC	IA	E	Grafos
Garcia, J. et al. (203)	2012	EC	PC			
von Detten, M (204)	2012	RE	PbC			
Platenius, M.C. et al. (205)	2012	RE	PC	C	E	Grafos
Sajani, H. (206)	2012	EX	PC	IA	E	Grafos
Nour, A. et al. (207)	2012	EC	PC	R	C	Grafos
Lei, Z. et al. (208)	2012	EC	PC	R	A	No indica
Muhammad, S. et al. (209)	2012	EX	PbC			
Rothlisberger, D. et al. (210)	2012	EX	PC	H	C	CCT
Weinreich, R. et al. (211)	2012	RE	PC	H	A	UML
Abi-Antoun, M. et al. (212)	2012	RE	PbC			
Kobayashi, K. et al. (213)	2012	EC	PC	C	E	Grafos
Moshkenani, Z.S. et al. (214)	2012	EC	PC	IA	E	Grafos
Boussaidi, G.E. et al. (215)	2012	EX	PC	C	E	Grafos
Chao-Tung, S. Y. Dowming (216)	2012	RE	PC	H	E	No indica
Li, H. et al. (217)	2012	EX	PC	C	E	Grafos
Risi, M. et al. (218)	2012	EX	PC	C	E	Grafos
Garcia, J. et al. (219)	2013	EX	PbC			
Nielsen, J. y M. Lykke (220)	2013	RE	C			
Gandrille, E. et al. (221)	2013	EC	PC	R	C	Grafos
Chun, Y. et al. (222)	2013	EC	PC	C	E	UML
Cuesta, C. et al. (223)	2013	RE	PC			
Fritz Solms (224)	2013	EC	PC	H	A	UML
Acher, M. et al. (225)	2013	RE	PC	H	E	Feature model
Buckley, J. et al. (226)	2013	EC	PbC			
Yuanfang, C. et al. (227)	2013	EX	PC	C	E	Matriz estructura de diseño
Maffort, C. (228)	2013	RE	PC	MD	E	No indica
Song, H. et al. (229)	2013	RE	PC	H	C	MOF y QVT
Forster, T. et al. (230)	2013	EC	PC	R	C	Grafos
von Detten, M. et al. (231)	2013	EC	PbC			
Sora, I. (232)	2013	EX	PC	C	E	Grafos
Zhu, J. et al. (233)	2013	EX	PC	C	E	Grafos
Venkatesh, K. y T. Reys (234)	2013	EX	PbC			
Peake, I. et al. (235)	2013	EC	PC	C	A	Grafos
Mircea, L. et al. (236)	2014	EC	PbC			

Para responder la séptima pregunta se tomó como criterio la cantidad de artículos publicados por cada investigador, siendo el mínimo valor cinco. Los resultados obtenidos se presentan en la tabla 5, donde se discrimina la cantidad de artículos en los que cada autor aparece como principal o en calidad de coautor, además del año de la última publicación. Finalmente, al responder la última pregunta de investigación se evidencia que el 45% de los trabajos han sido publicados en siete conferencias y tres revistas relacionadas en la tabla 6, a partir de la cual se puede inferir que el principal medio de publicación sobre recuperación de arquitecturas de software son las conferencias.

Tabla 4: Referencia a estudios seleccionados

<i>N°</i>	<i>Título</i>	<i>N°</i>	<i>Título</i>
1	A metamodel for the runtime architecture of an interactive system: the UIMS tool developers workshop	2	A reverse-engineering approach to subsystem structure identification
3	Recognizers for extracting architectural features from source code	4	Analysis and presentation of recovered software architectures
5	Architecture recovery in ARES	6	Extracting architectural features from source code
7	Recovery of software design, state-machines and specifications from source code	8	Requirements for an effective architecture recovery framework
9	A quality-based analysis of architecture recovery environments	10	Quality-based analysis of architecture recovery environments

Tabla 4: (Continuación)

N°	Título	N°	Título
11	Software Architecture Recovery of Embedded Software	12	A Novel Approach to Architectural Recovery in Evolving Object Oriented System
13	An Experiment in Distributed Software Architecture Recovery	14	Architecture recovery for software evolution
15	Developing an approach for the recovery of distributed software architectures	16	Event-based detection of concurrency
17	Extraction of architectural connections from event traces	18	Managing recovered function and structure of legacy software components
19	Perils of reconstructing architectures	20	Recovering software architecture from multiple source code analyses
21	Software architecture recovery and restructuring through clustering techniques	22	Software architecture recovery of a program family
23	Software architecture recovery using Conway's law	24	A software architecture reconstruction method
25	ART: an architectural reverse engineering environment	26	Component module classification for distributed software understanding
27	Experience with a reverse architecture approach to increase understanding	28	Recovering high-level views of object-oriented applications from static and dynamic information
29	Software architectural transformation	30	The Perils and Joys of Reconstructing Architectures
31	A pattern matching framework for software architecture recovery and restructuring	32	A use-case driven method of architecture recovery for program understanding and reuse reengineering
33	Maintaining a legacy: towards support at the architectural level	34	Use-case driven method of architecture recovery for program understanding and reuse reengineering
35	A graph pattern matching approach to software architecture recovery	36	A Lightweight Architecture Recovery Process
37	Alborz: a query-based tool for software architecture recovery	38	An approach for recovering distributed system architectures
39	Focus: a light-weight, incremental approach to software architecture recovery and evolution	40	MAP - mining architectures for product line evaluations
41	Architecture recovery of Apache 1.3--A case study	42	Architecture reconstruction in practice
43	Architecture recovery of dynamically linked applications: a case study	44	Architecture recovery of Web applications
45	Combining static and dynamic views for architecture reconstruction	46	Generation of architectural documentation using XML
47	INCO- Open Source Architecture Recovery	48	Pattern-supported architecture recovery
49	Practice patterns for architecture reconstruction	50	Revealer: a lexical pattern matcher for architecture recovery
51	A Lightweight Process for Architecture Recovery: From Code to Domain Requirements and Back Again	52	A Low-Level Analysis Library for Architecture Recovery
53	A user-assisted approach to component clustering	54	Analysis of a software product line architecture: an experience report
55	Architecture reconstruction guidelines	56	Architecture Recovery by Semi-Automatic Component Identification
57	Bridging legacy systems to model driven architecture	58	Embedding architectural support in industry
59	Enhancing domain-specific software architecture recovery	60	Experiences with software product family evolution
61	Feed-forward and recurrent neural networks for source code informal information analysis	62	Hierarchical Reflexion Models
63	Implementing architecture recovery by using improved genetic algorithm	64	Locating features in source code
65	Making sense of runtime architecture for mobile phone software	66	Moving towards quality attribute driven software architecture reconstruction
67	On modeling software architecture recovery as graph matching	68	Pattern-based Software Architecture Recovery
69	Reconstructing Software Architecture for J2EE Web Applications	70	Software architecture recovery based on pattern matching
71	An experimental study of two graph analysis based component capture methods for object-oriented systems	72	Applying fixed priority scheduling in practice
73	Architecture recovery for product families	74	Architecture-aware adaptive clustering of OO systems

Tabla 4: (Continuación)

N°	Título	N°	Título
75	CaCOphoNy: metamodel-driven software architecture reconstruction	76	Dealing with architectural issues: a case study
77	DiscoTect: A system for discovering architectures from running systems	78	Extraction and visualization of architectural structure based on cross references among object files
79	Metamodel-Driven Architecture Recovery	80	Refining code-design mapping with flow analysis
81	Software Architecture Recovery based on Dynamic Analysis	82	Symphony: view-driven software architecture reconstruction
83	The weighted combined algorithm: a linkage algorithm for software clustering	84	Understanding CASE generated legacy applications: A case study
85	Using feature modeling for program comprehension and software architecture recovery	86	Architectural support in industry: A reflection using C-POSH
87	Architecture recovery and abstraction from the perspective of processes	88	Coping with legacy system migration complexity
89	Dynamically discovering architectures with DiscoTect	90	Information-theoretic software clustering
91	Interpreting clustering results through cluster labeling	92	Mining Design Patterns for Architecture Reconstruction using an Expert System
93	ReArchJBs: a tool for automated software architecture recovery of JavaBeans-based applications	94	Reconstruction of software architectures: A literature and methods overview on the state of the science
95	Supporting Migration to Services using Software Architecture Reconstruction	96	Tool support for just-in-time architecture reconstruction and evaluation: an experience report
97	Towards a Set of Application Independent Clustering Criteria within an Architecture Recovery Approach	98	Alborz: An Interactive Toolkit to Extract Static and Dynamic Views of a Software System
99	An Orchestrated Multi-view Software Architecture Reconstruction Environment	100	Architecture reasoning for supporting product line evolution: An example on security
101	Automated software clustering: An insight using cluster labels	102	Comparison of software architecture reverse engineering methods
103	Discovering Architectures from Running Systems	104	Dynamic software architecture oriented approach to online evolution
105	Model-centric software architecture reconstruction	106	On implementing interoperable and flexible software evolution activities
107	Package patterns for visual architecture recovery	108	Recovering architectural assumptions
109	Recovering Architectural Views of OO Systems	110	Runtime recovery and manipulation of software architecture of component-based systems
111	SMaRTIC: towards building an accurate, robust and scalable specification miner	112	Softwrenaut: exploring hierarchical system decompositions
113	Static analysis based software architecture recovery	114	Using Graph Rewriting Systems for Automating Software Evolution Activities
115	Using software evolution to focus architectural recovery	116	Verifying a software architecture reconstruction framework with a case study
117	An Amalgamated Dynamic and Static Architecture Reconstruction Framework to Control Component Interactions 259	118	Bayesian Learning for Software Architecture Recovery
119	Architecture recovery and evaluation aiming at program understanding and reuse	120	Expectation, experiment and reflection on internet software evolution
121	Hierarchical Clustering for Software Architecture Recovery	122	Recovery of architecture artifacts
123	Reverse-architecting legacy software based on roles: An industrial experiment	124	Software Quality Attribute Analysis by Architecture Reconstruction (SQUA3RE)
125	Software Architecture Recovery	126	Static Analysis of Programs with Graphical User Interface
127	Towards A Process-Oriented Software Architecture Reconstruction Taxonomy	128	A Case Study on Software Evolution towards Service-Oriented Architecture
129	A field study in static extraction of runtime architectures	130	A runtime architecture-based approach for the dynamic evolution of distributed component-based systems
131	An iterative framework for software architecture recovery: An experience report	132	Architectural Concurrency Equivalence with Chaotic Models
133	Architectural styles as a guide for software architecture reconstruction	134	Can design pattern detection be useful for legacy systemmigration towards SOA?
135	Documenting after the fact: Recovering architectural design decisions	136	Encapsulating targeted component abstractions using software Reflexion Modelling

Tabla 4: (Continuación)

N°	Título	N°	Título
137	Extraction of Component-Based Architecture from Object-Oriented Systems	138	Quality-Driven Extraction of a Component-based Architecture from an Object-Oriented System
139	Software Architectural Reflection Mechanism for Runtime Adaptation	140	Static Conformance Checking of Runtime Architectural Structure
141	Static extraction and conformance checking of the runtime architecture of object-oriented systems	142	The MARPLE Project- A Tool for Design Pattern Detection and Software Architecture Reconstruction
143	Tool support for statically checking the structural conformance of an object-oriented system to its runtime architecture	144	Tool support for the static extraction of sound hierarchical representations of runtime object graphs
145	A biting-down approach to hierarchical decomposition of object-oriented systems based on structure analysis	146	A framework for tool-based software architecture reconstruction
147	A profile-based approach for maintaining software architecture: an industrial experience report	148	Comparison of Graph Clustering Algorithms for Recovering Software Architecture Module Views
149	Architecture reconstruction: Tutorial on reverse engineering to the architectural level	150	Discovery of architectural layers and measurement of layering violations in source code
151	Extending the reflexion method for consolidating software variants into product lines	152	FLORA: a framework for decomposing software architecture to introduce local recovery
153	JavaCompExt: Extracting Architectural Elements from Java Source Code	154	Modeling the implementation of state-based system architectures
155	Quality of the Source Code for Design and Architecture Recovery Techniques: Utilities are the Problem	156	Software Architecture Reconstruction: A Process-Oriented Taxonomy
157	Software Clustering Using Dynamic Analysis and Static Dependencies	158	Static extraction and conformance analysis of hierarchical runtime architectural structure using annotations
159	Static extraction of sound hierarchical runtime object graphs	160	.NET Reverse Engineering with MARPLE
161	A case-based reasoning approach to derive object-oriented models from software architectures	162	A model proposal for program comprehension
163	A Probabilistic Based Approach towards Software System Clustering	164	An approach for architectural layer recovery
165	An Interactive Change Impact Analysis Based on an Architectural Reflexion Model Approach	166	Analyzing security architectures
167	Architectural layer recovery for software system understanding and evolution	168	Architecture Recovery Using Latent Semantic Indexing and K-Means: An Empirical Evaluation
169	BeeEye: A framework for constructing architectural views	170	Component-based architecture recovery from object oriented systems via relational concept analysis
171	Developer refinement of runtime architectural structure	172	Development of a Methodology, Software -- Suite and Service for Supporting Software Architecture Reconstruction
173	Improved Hierarchical Clustering Algorithm for Software Architecture Recovery	174	Increasing the efficiency of the software architecture recovery through spanning tree based maximal graph mining technique
175	Incremental Reflexion Analysis	176	Kadre: domain-specific architectural recovery for scientific software systems
177	Object-oriented software architecture recovery using a new hybrid clustering algorithm	178	On-the-Fly and Non-invasive Extraction of Runtime Architectures Using Hierarchical Object Graphs
179	phpModeler - A Web Model Extractor	180	Role of relationships during clustering of object-oriented software systems
181	Software architecture reconstruction: An approach based on combining graph clustering and partitioning	182	Software architecture recovery process based on object-oriented source code and documentation
183	Software architecture recovery with error tolerance and penalty cost in graph	184	Towards a Graph Grammar Based Verification Approach for Runtime Constrained Evolution of Service-Oriented Architectures
185	Using the Kleinberg Algorithm and Vector Space Model for Software System Clustering	186	A Case Study in Extracting the Runtime Architecture of an Object-Oriented System
187	A genetic approach for software architecture recovery from object-oriented code	188	A Method for Evidence-Based Architecture Discovery
189	A tool for design pattern detection and software architecture reconstruction	190	A top-down approach to construct execution views of a large software-intensive system

Tabla 4: (Continuación)

N°	Título	N°	Título
191	A top-down strategy to reverse architecting execution views for a large and complex software-intensive system: An experience report	192	An Automatic Architecture Reconstruction and Refactoring Framework
193	Application Architecture Discovery - Towards Domain-driven, Easily-Extensible Code Structure	194	Architecture Reconstruction and Analysis of Medical Device Software
195	Evaluating reuse and program understanding in ArchMine architecture recovery approach	196	Improved Similarity Measures for Software Clustering
197	Meta-modeling runtime models	198	Object Oriented Software Clustering Based on Community Structure
199	Software Architecture Recovery through Graph Mining Technique and Reduction of Complexity Involved in A * Algorithm through Stable Marriage Problem	200	Solutions for Reverse Engineering 4GL Applications, Recovering the Design of a Logistical Wholesale System
201	Towards Open Source Software System Architecture Recovery Using Design Metrics	202	Version information support for software architecture recovery
203	A framework for obtaining the ground-truth in architectural recovery	204	Archimatrix: A Tool for Deficiency-Aware Software Architecture Reconstruction
205	Archimatrix: Improved Software Architecture Recovery in the Presence of Design Deficiencies	206	Automatic software architecture recovery: A machine learning approach
207	Characterizing real-time reflexion-based architecture recovery: an in-vivo multi-case study	208	Detecting anti-patterns in Java EE runtime system model
209	Evaluating relationship categories for clustering object-oriented software systems	210	Exploiting Dynamic Information in IDEs Improves Speed and Correctness of Software Maintenance Tasks
211	Extracting and Facilitating Architecture in Service-Oriented Software Systems	212	Extraction of ownership object graphs from object-oriented code: an experience report
213	Feature-gathering dependency-based software clustering using Dedication and Modularity	214	Improving Naïve Bayes classifier for software architecture reconstruction
215	Reconstructing Architectural Views from Legacy Systems	216	Software architecture recovery and re-documentation tool of a Hospital Information System
217	Software architecture recovery based on Weighted Hierarchical Clustering	218	Using fold-in and fold-out in the architecture recovery of software systems
219	A comparative analysis of software architecture recovery techniques	220	A tool analysis in architectural reconstruction
221	Design and runtime architectures to support autonomic management	222	Efficient software clustering technique using an adaptive and preventive dendrogram cutting approach
223	Evolution styles: using architectural knowledge as an evolution driver	224	Experiences with using the systematic method for architecture recovery (SyMAR)
225	Extraction and evolution of architectural variability models in plugin-based systems	226	JITTAC: A Just-in-Time tool for architectural consistency
227	Leveraging design rules to improve software architecture recovery	228	Mining Architectural Patterns Using Association Rules
229	Modeling and maintaining runtime software architectures	230	Recovering Component Dependencies Hidden by Frameworks-Experiences from Analyzing OSGi & Qt
231	Reengineering component-based software systems with Archimatrix	232	Software Architecture Reconstruction Through Clustering: Finding the Right Similarity Metric
233	Software architecture recovery through similarity-based graph clustering	234	Software-Architecture Recovery from Machine Code
235	Towards Reconstructing Architectural Models of Software Tools by Runtime Analysis	236	Evolutionary and collaborative software architecture recovery with Softwareonaut

ANÁLISIS DE LOS RESULTADOS

En coherencia con las preguntas establecidas para el mapeo literario, se hace un análisis fundamentado en los resultados estadísticos y en función de las siguientes variables y sus posibles relaciones: Propósito del estudio, aspecto de la arquitectura que se recupera, técnica de recuperación utilizada, elementos de la arquitectura recuperados y mecanismos utilizados para representar las vistas arquitectónicas recuperadas. A partir de los resultados obtenidos se observa que durante la última década se tiene una producción promedio de 17 artículos por año, lo cual indica la vigencia y relevancia del tema en el contexto de la ingeniería de software. Por otra parte, con respecto al tipo de estudio realizado se reporta una tendencia a presentar estudios de caso (EC 42.4%), siendo menos frecuentes los reportes de experiencia (RE 30.5%) y los experimentos (EX 25.4%), mientras que sólo se identificaron 4 revisiones sistemáticas (RS) que corresponden al 1.7% de los documentos encontrados.

Tabla 5: Principales investigadores en el tema.

Nombre	Cantidad de publicaciones			Año última publicación
	Autor principal	Coautor	Total	
Abi-Antoun, M.	11	0	11	2012
Sartipi, K.	10	0	10	2007
Maqbool, O.	5	4	9	2012
Aldrich, J.	0	8	8	2009
Mendonca, N. C.	7	0	7	2001
Koschke, R.	4	3	7	2010
Kazman, R.	2	5	7	2006
Kramer, J.	0	7	7	2001
Riva, C.	5	1	6	2009
Scanniello, G.	3	3	6	2012
Gall, H.	1	5	6	2004
Fontana, A.	5	0	5	2011
Stoermer, C.	5	0	5	2007
Medvidovic, N.	1	4	5	2013
O'Brien, L.	1	4	5	2006
Babri, H. A.	0	5	5	2007
Kostas, K.	0	5	5	2003

Tabla 6: Principales medios de publicación sobre el tema.

Medio de publicación	Tipo	Cantidad	%
European Conference on Software Maintenance and Reengineering (CSMR)	Conferencia	23	10%
Working Conference on Reverse Engineering (WCRE)	Conferencia	20	8%
International Conference on Software Maintenance (ICSM)	Conferencia	9	4%
International Conference on Software Engineering (ICSE)	Conferencia	9	4%
Working Conference on Software Architecture (WICSA)	Conferencia	9	4%
International Workshop on Program Comprehension	Conferencia	8	3%
International ACM Sigsoft conference on Quality of software architectures QoSA	Conferencia	8	3%
Journal software maintenance and evolutions: Research and practice	Revista	8	3%
IEEE Transactions on Software Engineering	Revista	6	3%
Information and Software Technology	Revista	6	3%

En relación con el propósito del estudio, los resultados revelan que la gran mayoría presentan propuestas conceptuales (77.1%), casi una quinta parte (18.2%) se centran en probar conceptos, mientras que sólo el 3% son comparaciones y el 1.7% hacen revisión literaria, como se observa en la tabla 7. Se encontraron distintos tipos de propuestas conceptuales: Propuesta de métodos, propuesta de métodos y desarrollo de herramientas, definición de marcos de referencia, definición de algoritmos, mejora de algoritmos, presentación de patrones, entre otras; de las cuales la mayoría de estudios corresponde a la definición de un marco de referencia (Ver tabla 7). Los primeros cinco tipos de propuestas conceptuales enunciadas se centran directamente en la recuperación de arquitecturas, por eso, más adelante se hace un análisis particular para este grupo conformado por 174 documentos. También se determinó que los estudios orientados a probar conceptos lo hacen a través del desarrollo de una herramienta, la aplicación de un método o la evaluación a nivel de comparación entre varios métodos.

En cuanto al aspecto de la arquitectura que se recupera, se determinó que la mitad (51,7%) de los estudios se centran en la recuperación de la estructura y una tercera parte (32.2%) recupera tanto la estructura como el comportamiento, mientras que sólo una sexta parte (16.1%) se centra en la recuperación del comportamiento, como se observa en la tabla 8. Este bajo porcentaje de investigaciones orientadas exclusivamente a recuperar el comportamiento podría tener su origen en los retos establecidos por Canfora et al. (2011) al momento de recuperar el comportamiento de un sistema software: Habilidad para compilar y ejecutar programas, habilidad para seleccionar entradas representativas, habilidad para buscar información relevante en las trazas de ejecución y habilidad para realizar análisis de caja negra.

También se determinó que las técnicas de recuperación más utilizadas son Clustering y la Heurística (ver tabla 9), entendiendo a esta última como la combinación de varias técnicas o la presentación de propuestas originales como: "morder abajo" (Zhang et al., 2009), procesos ligeros (Svetinovic y Godfrey, 2001),

conjunto de activos arquitectónicos de dominio (Ivkovic y Godfrey, 2003), recuperación basada en evidencias (Ros y Sangwan, 2011), recuperación incremental (Roy y Graham, 2008), recuperación a partir de trazas de eventos (Cook y Wolf, 1998), recuperación basada en reglas de abstracción (Qiao et al., 2003), recuperación basada en componentes (Anquetil et al., 2009; Sun et al., 2005), recuperación basada en metamodelos (Ran y Lencevicius, 2003), análisis a partir de la interfaz gráfica de usuario (Staiger, 2007) y análisis a partir de XML (Riva y Yang, 2002). Para el caso concreto de la recuperación del comportamiento las técnicas más utilizadas son la heurística y la reflexión.

Tabla 7: Detalle del propósito de los estudios encontrados por tipo de estudio realizado

Propósito	Detalle del propósito	Tipo de estudio				Total	%
		RE	EX	EC	RS		
Propuesta conceptual (77.1%)	Proponer método	6	9	18	0	33	14,0%
	Proponer método y desarrollar herramienta	7	6	12	0	25	10,6%
	Definir un marco de referencia	32	20	47	0	99	41,9%
	Definir un algoritmo	2	6	2	0	10	4,2%
	Mejorar algoritmo	0	5	2	0	7	3,0%
	Presentar patrones	1	0	0	0	1	0,4%
	Otra	4	2	1	0	7	3,0%
Prueba de conceptos (18.2%)	Evaluación de métodos	0	6	3	0	9	3,8%
	Aplicar método	7	1	7	0	15	6,4%
	Desarrollo de herramienta	6	5	8	0	19	8,1%
Comparación (3%)	Evaluar herramienta	7	0	0	0	7	3,0%
Revisión literaria (1.7%)	Establecer la situación actual	0	0	0	4	4	1,7%
Totales		72	60	100	4	246	100%
		30,5%	25.4%	42.4%	1.7%	100%	

Tabla 8: Detalle propuestas conceptuales por aspecto recuperado

Propósito	Detalle del propósito	Aspecto			Total	%
		E	C	A		
Propuesta conceptual (77.1%)	Proponer método	14	6	13	33	19,0%
	Proponer método y desarrolla herramienta	12	5	8	25	14,4%
	Definir un marco de referencia	48	17	34	99	56,9%
	Definir algoritmo	9	0	1	10	5,7%
	Mejorar algoritmo	7	0	0	7	4,0%
Totales		90	28	56	174	100%
		51,7%	16.1%	32.2%	100.0%	

Otro aspecto importante en la recuperación de la arquitectura corresponde a los elementos recuperados. A partir del mapeo sistemático se determinó que una tercera parte de los estudios concentran su atención en la recuperación de módulos (ver tabla 10), otro porcentaje igual, además de recuperar los componentes establecen las relaciones de comunicación entre ellos (sus conectores). En menor porcentaje se recupera subsistemas (8%), conceptos (6.9%), capas (5.2%), trazas de ejecución (3.4%), decisiones arquitectónicas (2.9%), patrones de diseño (2.3%), concurrencia y rendimiento (1.1) y sólo en un 0.6% se recupera: servicios, procesos y comunicación entre ellos, antipatrones en java y problemas de conformidad entre el diseño y la implementación.

Al establecer la relación entre los elementos recuperados y la técnica de recuperación utilizada (ver tabla 11), se observa que sólo la técnica de reconocimiento de patrones (Schmerl et al., 2006; Guo et al., 1999) se ha utilizado exclusivamente para recuperar componentes y conectores, mientras que en los demás casos se puede recuperar más de un elemento de la arquitectura, evidenciándose la tendencia de recuperar módulos para representar la estructura aplicando diversas técnicas de recuperación, siendo la más utilizada Clustering. Por otra parte, para representar el comportamiento la mayoría de las propuestas se limitan a recuperar componentes y conectores, que no describen en forma completa este aspecto de la arquitectura, lo que dificulta un análisis más detallado sobre los elementos recuperados.

Tabla 9: Técnicas de recuperación utilizadas según aspecto recuperado

Técnica de recuperación	Aspecto			Total	%
	E	C	A		
AD: Análisis dinámico		1	4	5	2,9%
AC: Anotaciones en el código		4		4	2,3%
AE: Análisis estático		2	1	3	1,7%
AtC: Atributos de calidad	2		3	5	2,9%
C: Clustering	37		13	50	28,7%
CP: Coincidencia de patrones	4	1	5	10	5,7%
CL: Concept Location	1	1	5	7	4,0%
H: Heurística	20	8	19	47	27,0%
IA: Inteligencia artificial	10		1	11	6,3%
MD: Minería de datos	2		1	3	1,7%
R: Reflexión	4	7	2	13	7,5%
RP: Reconocimiento de patrones		1	1	2	1,1%
TG: Teoría de grafos	8	3	1	12	6,9%
NI: No indican	2			2	1,1%
Total	90	28	56	174	100%

Tabla 10: Elementos recuperados según aspecto recuperado

Elementos recuperados	Aspecto			Total	%
	E	C	A		
Módulos	56			56	32,2%
Capas	9			9	5,2%
Subsistemas	14			14	8,0%
Servicios		1		1	0,6%
Objetos		6		6	3,4%
Componentes y conectores		14	42	56	32,2%
Trazas de ejecución		5	1	6	3,4%
Concurrencia y rendimiento		2		2	1,1%
Procesos y comunicación entre ellos			1	1	0,6%
Conceptos	3		9	12	6,9%
Decisiones arquitectónicas	3		2	5	2,9%
Patrones de diseño	4			4	2,3%
Antipatrones en java			1	1	0,6%
Problemas de conformidad entre el diseño y la implementación	1			1	0,6%
Total	90	28	56	174	100%

Por otra parte, los elementos recuperados se representan utilizando distintos mecanismos para mostrar las vistas arquitectónicas. El mapeo realizado permitió identificar que la forma de representación más usada es el grafo (59,8%), seguida por un grupo de trabajos que utilizan lenguajes de descripción de arquitectura como UML (20,7%), ACME (4,0%), ABC/ADL (0,6%), ASDL (Architecture Structure Description Language) (0,6%), mientras que otros utilizan lenguajes de marcado como XML (0,6%) o lenguajes propuestos por los autores como es el caso de IML (Intermediate Language) (Staiger, 2007). También se encontró una propuesta (0,6%) bajo el enfoque de desarrollo dirigido por modelos que utiliza el meta-metamodelo MOF (Meta Object Facility) y el lenguaje QVT (Query/View/Transformation).

Adicionalmente se identificaron propuestas que utilizan registro de tiempos (Lencevicius y Ran, 2004; Ran y Lencevicius, 2003), redes de Petri (Gröne et al., 2002; Cook y Wolf, 1998), Calling Context Tree (Röthlisberger et al., 2012), Concept-Requirement-Decision (CRD) Tree (Ros y Sangwan, 2011), Diagrama de eventos (Jerding y Rugaber, 1998), Feature model (Acher et al., 2013), Matriz de la estructura de diseño (Cai et al., 2013), Metamodelos (Favre, 2004) y texto con identificadores (Merlo et al., 2003).

Tabla 11: Elementos recuperados según técnica de recuperación

Elementos recuperados	Técnica de recuperación													Total	
	AD	AC	AE	AtC	C	CP	CL	H	IA	MD	R	RP	TG	NI	
Módulos				2	23	4	1	10	4	1	4		6	1	56 32,7%
Capas					6			1					2		9 5,2%
Subsistemas					8			3	2					1	14 8,0%
Servicios													1		1 0,6%
Objetos		3	2										1		6 3,4%
Componentes y conectores	3	1	1	2	11	6	4	16		1	7	2	2		56 32,7%
Trazas de ejecución								5			1				6 3,4%
Concurrencia y rendimiento	1							1							2 1,1%
Procesos y comunicación entre ellos	1														1 0,6%
Conceptos				1	2		2	6	1						12 5,7%
Decisiones arquitectónicas								5							5 2,9%
Patrones de diseño									4						4 2,3%
Antipatrones en java											1				1 0,6%
Problemas de conformidad entre el diseño y la implementación										1					1 0,6%
Total	5	4	3	5	50	10	7	47	11	3	13	2	12	2	174 100%
	2,9%	2,3%	1,7%	2,9%	28,7%	5,7%	4,0%	27,0%	6,3%	1,7%	7,5%	1,1%	6,9%	1,1%	100%

Teniendo en cuenta que la forma de representar un elemento arquitectónico determina su precisión semántica, se estableció la relación entre los elementos recuperados y su forma de representación para cada uno de los estudios encontrados en el mapeo, observando que los estudios que utilizan UML se limitan a mostrar los elementos que representan el comportamiento del sistema sólo por medio de componentes y conectores, que si bien, es un modelo que muestra aspectos relacionados con la comunicación, omite información relevante para entender los detalles, tales como la secuencia, la concurrencia y los tiempos, entre otros, a partir de los cuales se puede hacer un análisis más completo. Otro aspecto que se infiere es la tendencia a representar los distintos elementos recuperados a través de grafos, que si bien son fáciles de construir e interpretar, tienen una capacidad semántica limitada para representar los aspectos estructurales y de comportamiento.

CONCLUSIONES

Los resultados presentados permiten concluir lo siguiente:

- 1) Sería útil contar con un mecanismo unificado y consistente para representar los resultados obtenidos por las técnicas de recuperación de arquitecturas, que facilite su interpretación, análisis y reutilización (Gorton y Zhu, 2005).
- 2) Existe una amplia gama de propuestas conceptuales para recuperar la arquitectura, no obstante, las vistas arquitectónicas de comportamiento recuperadas hasta el momento aún no proporcionan la semántica suficiente, que permita realizar análisis más detallados sobre la arquitectura.
- 3) Hay una tendencia a recuperar componentes como elementos representativos de la arquitectura, lo cual, si bien es válido, conlleva a una sobrecarga del término que dificulta su interpretación (Ducasse y Pollet, 2009).
- 4) Se identificaron trece (13) técnicas de recuperación de arquitecturas, siendo la más usada Clustering, seis (6) tipos de lenguaje para describir la arquitectura y diez (10) formas para representar los elementos arquitectónicos, siendo los grafos los utilizados con mayor frecuencia.
- 5) Se corrobora el planteamiento de Koschke (2009) sobre la necesidad un catálogo de las técnicas de recuperación de arquitectura detallando cuándo y cómo deben ser utilizadas.
- 6) La recuperación de la arquitectura es un tema vigente que interesa a la comunidad desarrolladora de software.

AGRADECIMIENTOS

Este trabajo es parte de los resultados parciales de la tesis doctoral titulada Marco de referencia para la recuperación y análisis de vistas arquitectónicas de comportamiento, desarrollada por Martín Monroy Ríos en el programa de Doctorado de la Universidad del Cauca. Agradecemos a la Red Latinoamericana de Coordinación y Cooperación para unificar buenas prácticas de desarrollo de Software - Proyecto SPU CoCoNet-LA por su apoyo.

REFERENCIAS

- Acher, M.; Cleve, A.; Collet, P.; Merle, P.; Duchien, L. y P. Lahire, Extraction and evolution of architectural variability models in plugin-based systems, *Software and Systems Modeling*: 13(4), 1367-1394 (2014)
- Anquetil, N.; Royer, J.; André, P.; Ardourel, G.; Hnětynka, P.; Poch, T. y V. Petraşcu, JavaCompExt: Extracting architectural elements from java source code, *Proceedings - Working Conference on Reverse Engineering, WCRE*, 317-318 (2009)
- Cai, Y.; Wang, H.; Wong, S. y L. Wang, Leveraging design rules to improve software architecture recovery, *QoSA 2013 - Proceedings of the 9th International ACM Sigsoft Conference on the Quality of Software Architectures*, 133-142 (2013)
- Carletta, J., Assessing agreement on classification tasks: the kappa statistic, *Computational linguistics*: 22(2), 249-254 (1996)
- Cook, J. E. y A. L. Wolf, Event-based detection of concurrency, *Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering*, 35-45 (1998)
- Cuadrado F.; García B.; Dueñas J. y H. A. Parada, A Case Study on Software Evolution towards Service-Oriented Architecture, *IEEE Computer Society, 22nd International Conference on Advanced Information Networking and Applications*, Gino-wan City, Okinawa, Japan, 25 al 28 de marzo (2008)

- Cuenca, L.I.; Ortiz A. y A. Boza, Desarrollo de una Herramienta Software para la Vista de Información de la Arquitectura CIMOSA, doi: 10.4067/S0718-07642008000300014, Información Tecnológica: 19(3), 97-106 (2008)
- Demeyer, S.; Ducasse S. y O. Nierstrasz, Object-Oriented Reengineering Patterns. 342, Square Bracket Associates, Berna, Suiza (2009)
- Ducasse, S. y D. Pollet, Software architecture reconstruction: A process-oriented taxonomy, IEEE Transactions on Software Engineering: 35(4), 573-591 (2009)
- Favre, J., CacOphoNy: Metamodel-driven software architecture reconstruction, Proceedings - Working Conference on Reverse Engineering, WCRE, 204-213 (2004)
- Favre L.; Martinez L y C. Pereira, MDA-Based Reverse Engineering of Object Oriented Code, Enterprise, Business-Process and Information Systems Modeling, 29, Springer, 251–263 (2009)
- Fleiss, J. L.; Levin, B. y M. C. Paik, Statistical methods for rates and proportions, John Wiley & Sons (2013)
- Gorton, I. y L. Zhu, Tool support for just-in-time architecture reconstruction and evaluation: An experience report, Proceedings - 27th International Conference on Software Engineering, ICSE05, 514-523 (2005)
- Gröne, B; Knöpfel A. y R. Kugel, Architecture recovery of Apache 1.3--A case study, International Conference on Software Engineering Research and Practice (2002)
- Guo, G. Y.; Atlee, J. M. y R. Kazman, A software architecture reconstruction method, Springer US, 15-33 (1999)
- Hofmeister, C.; Nord, R, y D. Soni, Applied Software Architecture, Reading, MA: Addison-Wesley (1999)
- ISO/IEC/IEEE 42010. Systems and software engineering — Architecture description, International Organization for Standardization, Ginebra, Suiza (2011)
- ISO/IEC 25000. Software product Quality Requirements and Evaluation (SQuaRE) -- Guide to SQuaRE, International Organization for Standardization, Ginebra, Suiza (2005)
- Ivkovic, I. y M. Godfrey, Enhancing domain-specific software architecture recovery, 11th IEEE International Workshop on Program Comprehension IEEE, 266-273 mayo (2003)
- Jerding, D. y Rugaber, S., Extraction of architectural connections from event traces, In Proceedings of the ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering (1998)
- Kitchenham, B.A.; D. Budgen, y O.P. Brereton, Using mapping studies as the basis for further research. A participant-observer case study, Information and Software Technology: 53(6), 638–651 (2011)
- Koschke, R. Architecture reconstruction: Tutorial on reverse engineering to the architectural level (2009)
- Koschke, R., Reconstruction of software architectures: A literature and methods overview on the state of the science. [Rekonstruktion von Software-Architekturen: Ein Literatur- und Methoden-Überblick zum Stand der Wissenschaft] Informatik - Forschung Und Entwicklung: 19(3), 127-140 (2005)
- Koskinen, J., Software maintenance costs, University of Jyväskylä (2010)
- Kruchten, P.B., The '4+1' View Model of Architecture", IEEE Software, 12(6), 45–50 (1995)
- Lencevicius, R. y A. Ran, Applying fixed priority scheduling in practice, Proceedings of the Fourth International Workshop on Software and Performance, WOSP'04, 156-160 (2004)
- Merlo, E.; McAdam, I. y R. De Mori, Feed-forward and recurrent neural networks for source code informal information analysis. Journal of Software Maintenance and Evolution, 15(4), 205-244 (2003)
- Müller H. A.; Jahnke J. H.; Smith D. B.; Storey M.; Tilley S. R. y K. Wong, Reverse Engineering: A Roadmap. Conference on The Future of Software Engineering, Limerick, Ireland, 47 – 60 (2000)

- Peake, I.; Blech, J. O. y L. Fernando, Towards Reconstructing Architectural Models of Software Tools by Runtime Analysis. In EESSMOD@ MoDELS, 43-48 (2013)
- Petersen, K.; Feldt R.; Mujtaba S. y M. Mattsson, Systematic Mapping Studies in Software Engineering, 12th International Conference on Evaluation and Assessment in Software Engineering (EASE), 1-10, Bari, Italy junio (2008)
- Pollet, D.; Ducasse, S.; Poyet, L.; Alloui, I.; Cîmpan, S. y H. Verjus, Towards a process-oriented software architecture reconstruction taxonomy, in Proceedings of the European Conference on Software Maintenance and Reengineering, CSMR, 137-148 (2007)
- Qiao, B.; Yang, H.; Chu, W. C. y B. Xu, Bridging legacy systems to model driven architecture, Proceedings - IEEE Computer Society's International Computer Software and Applications Conference, 304-309 (2003)
- Ran, A. y R. Lencevicius, Making sense of runtime architecture for mobile phone software, Proceedings of the Joint European Software Engineering Conference (ESEC) and SIGSOFT Symposium on the Foundations of Software Engineering (FSE-11), 367-370 (2003)
- Riva, C. y Y. Yang, Generation of architectural documentation using XML, In Ninth Working Conference Reverse Engineering, 161-169 (2002)
- Ros, J. P. y R. S. Sangwan, A method for evidence-based architecture discovery, Proceedings - 9th Working IEEE/IFIP Conference on Software Architecture, WICSA, 342-345 (2011)
- Röthlisberger, D.; Härry, M.; Binder, W.; Moret, P.; Ansaloni, D.; Villazón, A. y O. Nierstrasz, Exploiting dynamic information in IDEs improves speed and correctness of software maintenance tasks, IEEE Transactions on Software Engineering: 38(3), 579-591 (2012)
- Roy, B. y T. Graham, An iterative framework for software architecture recovery: An experience report (2008)
- Rugaber S. y K. Stirewalt, Model-Driven Reverse Engineering, IEEE Software: 21(4), 45-53 (2004)
- Schmerl, B.; Aldrich, J.; Garlan, D.; Kazman, R. y H. Yan, Discovering architectures from running systems, IEEE Transactions on Software Engineering: 32(7), 454-466 (2006)
- Smolander, K., Four metaphors of architecture in software organizations: finding out the meaning of architecture in practice, In Empirical Software Engineering, International Symposium 211-221 (2002)
- Song, H.; Huang, G.; Wu, Y.; Chauvel, F.; Sun, Y. ; Shao, W.; y H. Mei, Modeling and maintaining runtime software architectures, Ruan Jian Xue Bao/Journal of Software: 24(8), 1731-1745 (2013)
- Staiger, S., Static analysis of programs with graphical user interface, Proceedings of the European Conference on Software Maintenance and Reengineering, CSMR, 252-261 (2007)
- Sun, C.; Cao, J.; Zhou, J.; Jin, M.; Liu, C. y Y. Shen, ReArchJBs: A tool for automated software architecture recovery of JavaBeans-based applications, Proceedings of the Australian Software Engineering Conference, ASWEC, 270-280 (2005)
- Svetinovic, D. y M. Godfrey, A lightweight architecture recovery process, Software Architecture Recovery and Modelling, Stuttgart, Germany (2001)
- Tonella, P.; Torchiano, M.; Du Bois, B. y Systä, T., Empirical studies in reverse engineering: state of the art and future trends, Empirical Software Engineering: 12(5), 551-571 (2007)
- Treude C.; Figueira F. y M. Storey, An Exploratory Study of Software Reverse Engineering in a Security Context, 18th Working Conference on Reverse Engineering, Limerick, Ireland, 184-188 (2011)
- Zhang, L.; Luo, J.; Li, H.; Sun, J. y H. Mei, A biting-down approach to hierarchical decomposition of object-oriented systems based on structure analysis, Journal of Software Maintenance and Evolution: 22(8), 567-596 (2009)