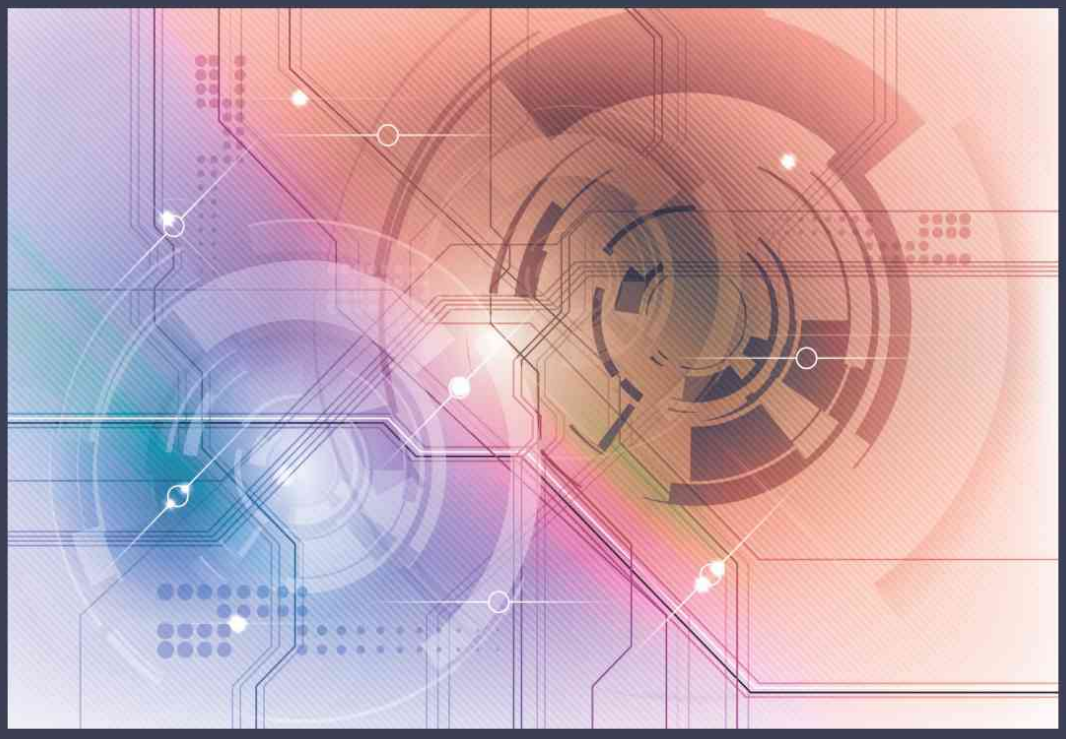
Métodos de investigación en ingeniería del software

https://yolibrospdf.com/programacion.html



Marcela Genero Bocco
José A. Cruz-Lemus
Mario G. Piattini Velthuis

Ra-Ma®

https://yolibrospdf.com/programacion.html

Métodos de investigación en ingeniería del software

Marcela Genero Bocco

José A. Cruz Lemus

Mario G. Piattini Velthuis

Ra-Ma®

La ley prohíbe
Copiar o Imprimir este libro

MÉTODOS DE INVESTIGACIÓN EN INGENIERÍA DEL SOFTWARE

ISBN de Edición en Papel: 978-84-9964-507-0

Editado por:
RA-MA, S.A. Editorial y Publicaciones
Calle Jarama, 33, Polígono Industrial IGARSA
28860 PARACUELLOS DE JARAMA, Madrid
Teléfono: 91 658 42 80
Fax: 91 662 81 39
Correo electrónico: editorial@ra-ma.com
Internet: www.ra-ma.es y www.ra-ma.com

Maquetación: Gustavo San Román Borrueco
Diseño Portada: Antonio García Tomé

ISBN: 978-84-9964-461-5

E-Book desarrollado en España en septiembre de 2014

A "mi Mario", con quien he experimentado infinidad de vivencias felices: y deseo se sigan replicando ;-).

Marcela Genero Bocco

A mis padres: con admiración.

José Antonio Cruz-Lemus

A Marifé Escolano: quien me inició en el mundo de la investigación empírica.

Mario Piattini Velthuis

ÍNDICE

AUTORES

MARCELA GENERO BOCCO

Profesora Titular de Universidad en el Departamento de Tecnologías y Sistemas de Información de la Universidad de Castilla-La Mancha, en Ciudad Real, España. Acreditada por la ANECA como Catedrática de Universidad en enero de 2012. Es Licenciada en Ciencias de la Computación, por la Universidad Nacional del Sur, en Bahía Blanca, Argentina (1989) y Doctora en informática en la Universidad de Castilla-La Mancha (2002).

Tiene numerosas publicaciones en revistas de prestigio internacional: *International Journal on Software Engineering and Knowledge Engineering*, *Information Software and Technology*, *Data and Knowledge Engineering*, Journal *of Software Maintenance*, *Journal of Systems and Software*, *Data and Knowledge Engineering*, *Software Quality Journal*, *Empirical Software Engineering*, *Information Sciences*, *Journal of Database Management*, *Software and System Modelling*, *ACM Transactions on Software Engineering and Methodology*, entre otras.

Ha participado como editora, junto con Mario Piattini y Coral Calero de los siguientes libros: "*Information and Database Quality*" (publicado por Kluwer) y "*Metrics for Software Conceptual Models*" (publicado por *Imperial College Press*).

Ha sido coautora de trabajos presentados en diferentes conferencias internacionales, como: SEKE, ER, CAISE, METRICS, ISESE, ESEM, MODELS, etc. y ha participado en numerosos comités de programas en congresos

internacionales (EASE, ESEM, ICEIS, CAISE, METRICS, ISESE, RCIS, etc.). Participa activamente como revisora de artículos en revistas de prestigio internacional y en comités de programa de congresos de prestigio nacional e internacional.

Ha sido Presidenta del Comité de Programa del congreso EASE 2012, que es uno de los principales congresos sobre el uso de estudios empíricos en la ingeniería del *software*. Ha organizado lo simposios doctorales PROFES 2012 y IDOESE 2012, y varios *workshops* relacionados con la calidad de modelos conceptuales y los estudios empíricos (IWQCM 2002, 2003 (dentro del ER), EESSMod 2011, 2012, 2013 (dentro del MODELS)). Ha organizado tres tutoriales sobre estudios empíricos en la ingeniería del *software* y el modelado dentro de las JISBD 2004 y el MODELS 2012 y 2013) y el congreso EASE 2012 que es uno de los más relevantes e temas relaciona

Ha liderado numerosos proyectos relacionados con la calidad del *software*, financiados por convocatorias regionales y nacionales y ha dirigido varias tesis doctorales. Sus principales áreas de investigación son: calidad en el modelado conceptual, beneficios del modelado usando UML, métodos de investigación en informática, medición en la ingeniería del *software*, validación empírica de tecnologías *software*, técnicas de análisis y agregación de datos empíricos, etc.

Es miembro desde el año 2004 de la red en la que participan prestigiosos investigadores y profesionales sobre ingeniería del *software* empírica (ISERN).

JOSÉ ANTONIO CRUZ LEMUS

Doctor e Ingeniero en informática por la Universidad de Castilla-La Mancha. En la actualidad es Profesor Contratado Doctor para el Departamento de Tecnologías y Sistemas de Información en la Escuela Superior de informática, de la misma universidad y está acreditado como Profesor Titular de Universidad por ANECA desde febrero de 2012.

Cuenta con publicaciones en prestigiosas revistas internacionales como *Empirical Software Engineering*, *Information Sciences*, *Information and Software Technology*: *Software and Systems Modelling* o *ACM Transactions on Software Engineering and Methodology*, entre otras.

Autor de numerosos trabajos en distintos congresos internacionales (MODELS, ISESE, SEKE, ER, ESEM) y presidente del comité organizador del congreso EASE 2012.

MARIO PIATTINI VELTHUIS

Doctor y Licenciado en informática por la Universidad Politécnica de Madrid. Licenciado en Psicología por la Universidad Nacional de Educación a Distancia. Máster en Auditoría informática (CENEI). Máster en Gestión de Proyectos (Universidad de Washington). Especialista en la Aplicación de Tecnologías de la Información en la Gestión Empresarial (CEPADE-UPM). CISA, CISM, CRISC, y CGEIT por la ISACA. Diplomado en Calidad por la Asociación Española para la Calidad. CSQE por ASQ. Auditor Jefe 15504 por AENOR, y CMMI, ITIL y TMap *Foundations*.

Ha trabajado como consultor para numerosos organismos y empresas, entre las que destacan: Ministerio de Industria y Energía, Ministerio de Administraciones Públicas, Siemens-Nixdorf, Unisys, Hewlett-Packard, Oracle, ICM, Atos-Ods, Soluziona/Indra, Sistemas Técnicos de Loterías, Cytsa, etc. Ha sido Socio-Fundador de las empresas Cronos Ibérica, S.A., Kybele Consulting, S.L. y Alarcos Quality Center, S.L.

Ha sido profesor en las Universidades Complutense y Carlos III de Madrid, director de varios *masters* y cursos de especialización, y ha impartido un centenar de cursos en empresas y organismos. Actualmente es Catedrático de Universidad de Lenguajes y Sistemas Informáticos en la Escuela Superior de informática de la Universidad de Castilla-La Mancha (UCLM), donde dirige el grupo de investigación Alarcos, especializado en Calidad de Sistemas de Información. También es Director del Instituto de Tecnologías y Sistemas de Información de la UCLM.

Ha dirigido/participado en más de 60 proyectos de investigación, dirigido 40 tesis doctorales, y participado como miembro o Presidente de Comité de Programa de un centenar de congresos. Ha publicado 50 libros en temas relacionados con Ingeniería del *Software* y Bases de Datos, y más de 150 artículos en revistas indexadas en el JCR, así como un centenar de comunicaciones en congresos de prestigio. Se encuentra entre los 15 "*Top scholars in the field of systems and software engineering* (2004-2008)", según el *Journal of Systems and Software* (Wong *et al.*, 2011).

Ha sido Coordinador del Área de Ciencias de la Computación y Tecnología informática de la Agencia Nacional de Evaluación y Prospectiva, Coordinador científico del área de Tecnologías de la Información y de las Comunicaciones para la elaboración del Plan Nacional de I+D+I 2008-2011 y Colaborador de la Subdirección General de Formación e Incorporación de Investigadores de la Dirección General de Investigación y Gestión del Plan Nacional de I+D+i.

PRÓLOGO

Hace ya unos años surgió con fuerza un debate cargado de interesantes matices entre quienes nos dedicamos a la docencia e investigación en informática en el marco de la universidad. Ese debate, que nos hizo mirarnos a nosotros mismos y cuestionarnos la perspectiva desde la que impartíamos nuestra docencia, simplemente trataba de analizar la naturaleza de la actividad de los/las profesionales en informática. La cuestión era: ¿es la informática una ciencia o una ingeniería?

Esa pregunta tenía en aquel momento importantes implicaciones, ya que se estaban redactando los nuevos planes de estudio adaptados al marco normativo que el gobierno derivó del acuerdo de Bolonia y, más importante todavía, se estaba discutiendo si la informática, como otros estudios profesionales, entre ellos las ingenierías, tendría reconocidas atribuciones profesionales en exclusiva, lo que sería inmediatamente un arma frente al intrusismo profesional y una cierta garantía de revalorización de nuestra actividad, si se conseguía que los proyectos de desarrollo de software tuvieran que ser firmados por una persona con titulación en informática y colegiada en un colegio de Ingeniería informática.

Dilucidar la cuestión sobre la naturaleza científica o técnica de la informática, despertaba también otros debates ya más directamente relacionados con la problemática docente. Concretamente, parecía que de la respuesta a la pregunta "informática, ¿ciencia o ingeniería?" se podría desprender la decisión adecuada sobre cuántas asignaturas básicas de matemáticas y electrónica se debían incluir en los planes de estudio, o sobre el lenguaje de programación más adecuado para iniciarse en el diseño de soluciones software.

Sin embargo, a lo largo de todo aquel debate, incluso cuando se bajaba al nivel de discutir la estructura de los planes de estudio, nunca se le prestó atención a los aspectos epistemológicos o simplemente metodológicos de la investigación. No me refiero aquí a investigación solo en el sentido "científico" del término, sino a algo mucho más sencillo, cotidiano y necesario, al proceso de averiguar, consiguiendo *evidencias contrastables*, si cierta hipótesis pudiera ser cierta o no.

En las ciencias humanas como la psicología, se presta una gran atención a la metodología a seguir para poder lograr esas evidencias contrastables que permitan refutar o no una determinada hipótesis. Quizá ese interés en las facultades de la psicología de todo el mundo, en dar a los futuros profesionales una correcta formación en métodos de investigación, se deba a lo difícil de medir las variables relevantes en psicología, aunque es preciso reconocer aquí que esa ciencia ha avanzado muchísimo en el desarrollo de técnicas de observación y medición de la conducta y de métodos para poder contrastar las hipótesis formuladas.

Pero, ¿a qué viene en este punto hablar de psicología o hacer un parangón entre las necesidades formativas en informática y las Ciencias Humanas?. La respuesta se basa en un hecho evidente que quizá, por tan evidente, no se ha tenido suficientemente en cuenta, al menos, en el diseño de la formación de los/as futuros profesionales: ***hacer software es una actividad de las personas, no de las máquinas***. Somos las personas las que hacemos los análisis, los diseños, la programación, las pruebas y el mantenimiento del software y por ello para validar si una determinada técnica o metodología de desarrollo del *software* es mejor que otra, tenemos que "*medir*" cómo las personas se desenvuelven con cada una de ellas. Esa necesidad de medir la conducta de personas para poder tener información contrastable de utilidad en informática se da en muchos otros ámbitos dentro de nuestra disciplina.

En la actualidad se está acometiendo la *gamificación* de entornos de trabajo como un medio de motivar a los/as desarrolladores/as y mejorar su productividad, e incluso como forma de obtener medidas objetivas de evaluación de su eficiencia, pero para saber si ciertas técnicas de gamificación consiguen realmente los resultados que se proponen tendremos que saber medir si la satisfacción, motivación y rendimiento de las personas que usan dichos entornos realmente se incrementa.

Pero el interés de los métodos de investigación derivados de ciencias humanas no se acaba en su aplicación a la observación del rendimiento de quienes desarrollan *software*. Pensemos en los estudios sobre usabilidad, o amigabilidad de interfaces de usuario. Una investigación rigurosa, sobre si ciertas estrategias de diseño de interfaces es realmente mejor que otra, requiere la realización de experimentos con usuarios, que se deben planear controlando parámetros como

experiencia previa de los usuarios, nivel de motivación, etc., para poder garantizar que las variaciones observadas en facilidad de uso, satisfacción de la experiencia de uso, etc., (variable dependiente) están causadas directamente por los diferentes diseños de la interfaz de usuario (variable independiente). Pero tanto las técnicas para medir esas variables, como la metodología para diseñar los experimentos, requieren una formación que, en la actualidad, es ajena a la formación que se imparte en las facultades de informática.

No acaban aquí los ámbitos de la informática que requieren de esas técnicas de investigación para su avance. Hoy en día el perfilado de usuarios es un tema crucial para acometer tareas de *Recomendación* y *Reputación* en redes sociales. En *Recuperación de Información*, tanto en aplicaciones de prensa digital como en ámbitos Web, de nuevo, el análisis del comportamiento de las personas que usan el sistema es crucial para el diseño de los algoritmos que deciden qué contenidos presentar a cada quien.

Se podrían poner más ejemplos en los que la utilización de métodos de investigación, como los que se presentan en esta obra, son imprescindibles para conseguir las evidencias contrastables que permitan decidir cómo hacer mejor informática, pero baste resumir todas esas situaciones en dos contextos raramente ajenos a cualquiera de nuestras actividades profesionales. Aquellos en donde queremos saber, de entre diferentes alternativas, cuál va a ser la más adecuada para mejorar la experiencia de quienes usan el sistema, y aquellos en donde queremos saber, de entre diferentes alternativas, cuál va a ser más la adecuada para mejorar la productividad de quienes lo desarrollan.

Sin embargo, la formación sobre la metodología para "medir" adecuadamente, o en general, para abordar el diseño de experimentos o de simples evaluaciones empíricas, está completamente ausente de nuestros planes de estudio, en donde, paradójicamente, sí abundan las asignaturas en estadística que servirían para analizar los datos obtenidos en dichos experimentos.

Este libro viene a llenar ese vacío presentado, de un modo sencillo y completamente enfocado a la Ingeniería del Software, los métodos de investigación, diseño de experimentos, técnicas de medición, etc. más adecuados, no solo para poder hacer investigación orientada a avanzar el estado del arte en ingeniería del *software*, sino incluso para que un jefe de proyecto pueda acumular evidencias que le permitan decidir si cambiar o no de entorno o de herramientas para mejorar la productividad de su equipo. Así, en el esta obra se exponen desde las técnicas de creación de cuestionarios y encuestas, hasta las de diseño de experimentos orientados a evaluar el efecto de las variables que se quieren investigar. Como ya se indicó muchos de esos métodos y técnicas provienen, como no podía ser de otro modo, de la investigación en ciencias humanas pero se

presentan aquí completamente orientados a la investigación en informática en general y en Ingeniería del Software en particular.

Para terminar este prólogo a la que considero una obra de gran valor formativo y calidad didáctica, volvamos a la ya vieja discusión sobre la naturaleza de la informática. Si la informática es ciencia, no se puede concebir sin la adecuada metodología de investigación, hoy gran ausente en la formación de grado. Pero si la informática es ingeniería, igualmente necesita comprobar la utilidad de las nuevas tecnologías que surjan, y la investigación tecnológica también es investigación. Por ello, con independencia del debate sobre si la formación de los futuros/as profesionales se debe orientar más a aspectos fundamentales o aplicados, con independencia de si se debe empezar a enseñar programación con Java o con C, con independencia sobre si es mejor formar en complejidad algorítmica o en modelado y diseño de software, lo que es indudable es que todos los futuros profesionales deberían tener una formación, al menos básica, en los métodos de investigación que aquí se proponen.

Abrigo el deseo y la esperanza de que la lectura de esta obra facilite, a los profesionales actuales y futuros, herramientas que considero imprescindibles para mejorar su actividad profesional y que, por otro lado, sirva para sensibilizar a quienes tienen responsabilidades en el diseño de la formación en informática sobre la necesidad de introducir estas metodologías de la investigación en los estudios de grado en informática.

A Coruña, 20 de Julio de 2014

Nieves R. Brisaboa

PREFACIO

Hace veinte años que se celebró la primera reunión de la red ISERN (*International Software Engineering Research Network*), cuyo manifiesto de creación señala que: "La ingeniería del *software* es una disciplina relativamente nueva e inmadura. Con el fin de madurar, necesitamos adoptar una visión experimental que nos permita observar y experimentar con las tecnologías, comprender sus debilidades y fortalezas, adaptar las tecnologías a los objetivos y características de proyectos específicos, y empaquetarlos junto con la experiencia obtenida empíricamente con el fin de mejorar su potencial de reutilización en proyectos futuros". En el grupo Alarcos estuvimos preocupados desde que iniciamos nuestra investigación en mantenimiento y métricas de *software*, en el año 1997, por adoptar esa visión experimental rigurosa que nos ayudara a determinar las mejores técnicas y medidas que permitieran a las organizaciones mejorar la productividad y la calidad del *software*. Para ello al principio aplicamos técnicas "prestadas" de la Psicología, disciplina, que al igual que otras muchas, concede desde hace bastante más tiempo que la informática, la importancia que se merecen a los aspectos metodológicos de la investigación.

Afortunadamente a principios de la década de los 2000 ya se publicaban los primeros libros (como el de Wohlin y colaboradores y el de Juristo y Moreno) y los primeros artículos (como los de Kitchenham y Pfleeger) que nos guiaban a la hora de hacer experimentos o encuestas en el área de la ingeniería del *software*. En estos últimos diez años, por un lado, los investigadores han desarrollado toda una serie de guías y técnicas que nos permiten llevar a cabo la investigación de manera rigurosa; y, por otra, las organizaciones y los "profesionales" han empezado a darse cuenta de la necesidad de contrastar experimentalmente muchas de las creencias y nuevas técnicas en el área de la ingeniería del *software*, concediendo cada vez más importancia a la ingeniería del *software* basada en evidencias (EBSE: *Evidence-*

Based Software Engineering) que se puede considerar una evolución de la ingeniería del *software* empírica (ESE: *Empirical Software Engineering*).

Sin embargo, y a pesar de la importancia que ha cobrado la ESE y los numerosos talleres de trabajo (*workshops*), conferencias, redes, proyectos y publicaciones que han demostrado su necesidad y validado su utilidad; todavía existen muchos investigadores y profesionales de la ingeniería del *software* que desconocen las técnicas experimentales y las pautas para su aplicación. De hecho en los últimos años estamos impartiendo numerosos seminarios y cursos de doctorado sobre el tema, que consideramos debiera incluirse aunque sea resumidamente en los propios grados y másteres de ingeniería del *software*, ya que la investigación nos permite comprender mejor la naturaleza del *software* y de los procesos para su creación, desarrollo y evolución.

Los objetivos que nos hemos propuesto en esta obra son los siguientes:

- Presentar de forma clara y precisa los principales métodos de investigación aplicables en ingeniería del *software*.
- Mostrar ejemplos concretos de aplicación de los métodos.
- Dar a conocer los principales problemas en su utilización.
- Proporcionar bibliografía y recursos que puedan ayudar a una utilización más efectiva de los métodos.

Todo ello esperamos que contribuya a incrementar la rigurosidad de la investigación que se lleva a cabo en ingeniería del *software* y permita potenciar la transferencia de tecnología en este campo, al proporcionar a las organizaciones y empresas evidencias sobre las mejoras y ventajas que pueden ofrecer las técnicas de la ingeniería del *software*.

CONTENIDO

La obra consta de siete capítulos.

En el primero, se introduce la necesidad de utilizar métodos rigurosos en proyectos e investigaciones en ingeniería del *software*, se presentan las características de los diferentes métodos, la contextualización de los estudios empíricos, el uso de teorías en ingeniería del *software* y se abordan algunos problemas importantes en la investigación como los relativos a la ética y a la colaboración entre universidad y empresa.

A continuación, empieza la I parte del libro dedicada a los métodos primarios, que son aquellos que se utilizan para obtener evidencia empírica sobre un tema de interés en el ámbito de la ingeniería del *software*.

El capítulo 2, aborda el diseño y la realización de encuestas, así como el análisis de los datos obtenidos de las mismas.

Los experimentos son el objeto de estudio del capítulo 3, incluyendo el proceso experimental, la realización de réplicas y familias de experimentos, como así también la agregación de experimentos a través de meta-análisis.

El capítulo 4, expone las principales cuestiones relacionadas con los estudios de casos, incluyendo algunas cuestiones relacionadas con los métodos etnográficos. Mientras que en el capítulo 5, se ofrece una panorámica sobre la investigación-acción.

El siguiente capítulo inaugura la Parte II del libro dedicada a los métodos secundarios, aquellos que se encargan de recopilar y sintetizar de manera sistemática y rigurosa toda investigación existente sobre un tema de interés. Concretamente el Capítulo 6, aborda un tema fundamental a la hora de plantear un proyecto o tesis doctoral o iniciar cualquier investigación, la realización de revisiones sistemáticas de la literatura.

El siguiente capítulo (el capítulo 7), da paso a la parte III del libro donde se exponen algunas experiencias reales que combinan algunos métodos anteriores, un método para la creación y validación de métricas y la combinación de investigación-acción y estudios de casos.

El libro incluye en cada capítulo, una interesante bibliografía recomendada así como sitios web y herramientas de interés, y al final del libro se presenta la bibliografía que ha servido de referencia y una lista de los acrónimos utilizados.

ORIENTACIÓN A LOS LECTORES

Aunque un conocimiento en profundidad de los métodos de investigación puede estar reservado a expertos en la materia, nuestro propósito al presentar este libro ha sido dirigirnos a una audiencia mucho más amplia que comprende:

- Directores e investigadores de proyectos de investigación.
- Doctorandos que desean realizar su tesis doctoral de manera rigurosa.

- Autores de artículos científicos en ingeniería del *software* que quieran mejorar sus trabajos desde el punto de vista metodológico.
- Personal informático en general (jefes de proyecto, analistas, consultores, etc.) que quieran evaluar o mejorar alguna técnica nueva antes de implantarla en su organización.

Debido a la diversidad de la audiencia, el estudio de esta obra puede realizarse de maneras muy distintas, dependiendo de la finalidad y conocimientos previos del lector. Cada capítulo puede ser consultado de manera autónoma, sin tener que seguir el orden que se ha establecido.

AGRADECIMIENTOS

Queríamos expresar nuestro agradecimiento a todas las personas que nos han acompañado en estos casi veinte años que llevamos aprendiendo en este tema:

Nuestros compañeros del Grupo Alarcos de la UCLM, que han compartido sus conocimientos y experiencias en métodos de investigación.

Barbara Kitchenham quien siempre ha estado predispuesta para clarificarnos muchísimos aspectos, en especial, los relativos a las revisiones sistemáticas de la literatura.

Esperanza Manso, nuestro "lazarillo en estadística", que siempre nos ayuda y aconseja tan bien, en temas relacionados con el análisis de datos.

Claes Wohlin, Dag Sjøberg, Helen Sharp, Dieter Rombach, Daniela Cruzes, Tore Dybå, Guilherme Travassos, Matin Höst, Sandro Morasca, Michel Chaudron, Natalia Juristo, Sira Vegas, Oscar Dieste, Giuseppe Visaggio, Teresa Baldassarre y Danilo Caivano, que han tenido la consideración de venir a nuestro centro en Ciudad Real a explicarnos de primera mano, y muchas veces en primicia, los últimos avances logrados y sus experiencias en métodos de investigación.

Los miembros de la red ISERN con quienes hemos compartido las reuniones anuales de la red, en las cuales hemos mantenido fructíferas discusiones que nos han permitido enriquecer nuestros conocimientos sobre ingeniería del *software* empírica.

Los profesores que nos han invitado a impartir cursos relacionados con la ingeniería del *software* empírica en las universidades de Deusto, Sevilla, Rey Juan Carlos, Murcia y Alicante de España, Torino, Tor-Vergata, Sannio y Bari-Aldo

Moro de Italia, Innsbruck de Austria, Namur de Bélgica, ORT de Uruguay y La Plata de Argentina; como así también a los alumnos que participaron en dichos cursos.

Los profesores que nos han permitido realizar réplicas de experimentos y a los alumnos que han participado en ellas.

Un agradecimiento especial a Nieves Brisaboa, Gestora de la Subdirección General de Proyectos de Investigación del Área Científica de Tecnologías Informáticas (TIN), por haber aceptado escribir el prólogo a esta obra.

Por último, nos resta dar las gracias a Sandra Ramírez por sus valiosas sugerencias que, como en otras muchas ocasiones, han contribuido a mejorar considerablemente este libro, y a la editorial Ra-Ma, especialmente a Jesús Ramírez, por su apoyo y confianza.

Marcela Genero Bocco

José Antonio Cruz-Lemus

Mario Piattini Velthuis

Ciudad Real: *Julio 2014*

Capítulo 1

INVESTIGACIÓN EN INGENIERÍA DEL SOFTWARE

1.1 NECESIDAD DE CONOCIMIENTO RIGUROSO EN INGENIERÍA DEL SOFTWARE

Como señalan Basili *et al.* (2001) prácticamente todas las organizaciones que desarrollan y mantienen *software* comparten las siguientes necesidades:

- Comprender los procesos y productos.
- Evaluar los éxitos y fracasos.
- Aprender de las experiencias.
- Empaquetar experiencias exitosas.
- Reutilizar experiencias exitosas.

Esto es debido a que las organizaciones dedicadas a la ingeniería del *software* requieren y generan grandes cantidades de conocimiento, de diverso tipo sobre los productos, procesos y proyectos *software*, así como sobre el personal involucrado a los largo de todo el ciclo de vida del *software*. En este sentido, Lindvall y Rus (2003) afirman que la gestión del conocimiento permite "producir mejor *software*, de una forma más rápida y económica, así como tomar mejores decisiones".

Ahora bien, comprender una disciplina implica aprendizaje, es decir, observación, reflexión y encapsulación de conocimiento, construcción de modelos (del dominio de aplicación, de los procesos para resolver problemas, etc.), experimentación, y evolución de los modelos con el tiempo. Desde un punto de vista científico, la investigación en ingeniería del *software* se centra en conocer la naturaleza de los procesos, productos e interrelaciones entre ellos en el contexto de un sistema *software* o de un sistema organizacional (en el caso de sistemas de información). Para lo que es necesario emplear diferentes métodos de investigación en la ingeniería del *software*.

Sin embargo, en informática en general, y en la ingeniería del *software* en particular, no se le ha concedido la importancia que se merece a los métodos de investigación, y se han argumentado muchas excusas para no investigar. En la Tabla 1.1 se presentan las excusas más comunes, según Tichy (1998), y sus correspondientes refutaciones.

Excusas	**Refutación**
El método científico no es aplicable.	Para entender el proceso de la información, los científicos informáticos deben observar los fenómenos y formular y probar explicaciones.
El nivel de experimentación actual es suficiente.	Comparando con otras ciencias, los científicos informáticos validan un porcentaje mínimo de sus afirmaciones.
Los experimentos tienen un coste muy alto.	Se pueden llevar a cabo experimentos significativos con presupuestos pequeños.
Las demostraciones son suficientes.	Las demostraciones sólo ilustran un potencial pero no demuestran nada.
La experimentación ralentiza el progreso.	Aumentar el porcentaje de artículos con validación significativa es una buena forma de acelerar el progreso.
Las tecnologías cambian demasiado rápido.	Si una cuestión se vuelve irrelevante de forma rápida es que no estaba bien planteada o definida.

Tabla 1.1. Excusas para no investigar en ingeniería del software y sus refutaciones (*Tichy*: *1998*)

Como señalan Juristo y Moreno (2001), no se concede suficiente importancia a la ingeniería del *software*, probablemente porque:

- Los desarrolladores no están instruidos sobre la necesidad y la importancia de contrastar ideas contra la realidad.
- Los desarrolladores no son capaces de entender los datos de un experimento o cómo fueron analizados por otros porque carecen de los conocimientos estadísticos.

Nosotros creemos además que hasta hace bien poco la ingeniería del *software*, tanto en el ámbito académico como profesional, se movía muchas veces por modas; lo que Alan Davis denominó "*Lemmingeniería del Software*", usando al metáfora de los leminos (Lemmini); lo que no favorecía en absoluto el rigor experimental. Afortunadamente, y en parte debido a los grandes y sonoros fracasos de la ingeniería del *software*; los actuales directivos de las empresas se están dando cuenta de la necesidad de un mayor gobierno del desarrollo y mantenimiento del *software*, lo que ha potenciado la ingeniería del *software* basada en la evidencia.

Es cierto que la fabricación del *software* no es análoga a la fabricación industrial debido al gran peso del factor humano en el desarrollo y mantenimiento de *software*. Por eso no debemos dejarnos "despistar" por esta metáfora tan utilizada; y observar como otras disciplinas, como la psicología, pedagogía, sociología, etc. han aplicado con mucho éxito desde hace varios años métodos de investigación que pueden "adaptarse" y "adoptarse" con éxito en nuestro campo.

1.2 TIPOS DE MÉTODOS DE INVESTIGACIÓN

Los métodos de investigación se pueden clasificar de diversas maneras, una primera clasificación en función del nivel de evidencia que proporcionan, puede ser:

- **Métodos primarios**: son aquellos métodos utilizados para realizar estudios con el objetivo de obtener evidencia empírica sobra un tema de interés (experimentos, encuestas, estudios de casos, etc.). Por consiguiente, a los estudios llevados a cabo utilizando estos métodos se los denomina *estudios primarios*.
- **Métodos secundarios**: se refiere a los métodos que permiten recopilar de manera sistemática y rigurosa los estudios primarios relacionados con una pregunta de investigación específica, con el objetivo de sintetizar la evidencia disponible para responder a dicha pregunta. En la ingeniería del *software* se utilizan, las revisiones y mapeos sistemáticos de la literatura como métodos de investigación secundarios. Este tipo de métodos se utilizan para producir *estudios secundarios* (revisiones o mapeos sistemáticos) y *estudios terciarios* (revisiones sistemáticas de revisiones sistemáticas).

Por lo que respecta a los métodos de investigación primarios, además de la clásica distinción entre *inductivos* (si a partir de la observación se induce la teoría) o *deductivos* (si partiendo de la teoría se plantean hipótesis y finalmente observaciones), existen otras varias. Una clasificación ampliamente aceptada es la que divide a los métodos en cuantitativos y cualitativos, atendiendo principalmente al tipo de datos que se manejan. Sin embargo, siguiendo a Runeson *et al.* (2012), preferimos diferenciar entre métodos flexibles o fijos. En los fijos todos los parámetros se definen cuando se inicia la investigación, mientras que en los flexibles pueden cambiar durante el curso de la investigación.

Los métodos de investigación fijos (por ejemplo, experimentos) son especialmente adecuados para el estudio de fenómenos u objetos naturales, mientras que el estudio de fenómenos culturales y sociales requiere otro tipo de métodos, que no se basen en experimentos ni teorías formales, sino en entrevistas, documentos, impresiones y reacciones del investigador, etc. y son los métodos flexibles (como los estudios de casos o la investigación-acción). Los primeros se basan principalmente en datos cuantitativos (números) y los segundos en datos cualitativos (palabras, descripciones, diagramas, fotos, etc.), aunque también pueden usar datos cuantitativos. En ingeniería del *software* pueden utilizarse ambos tipos de métodos, aunque hasta hace poco han prevalecido los fijos.

Por otro lado, atendiendo a su propósito se pueden distinguir cuatro tipos de investigaciones:

- **Exploratoria**, busca qué es lo que sucede y genera ideas e hipótesis para nuevas investigaciones.
- **Descriptiva**, describe el estado actual de una situación o fenómeno.
- **Explicativa**, busca la explicación para una situación o problema (en forma de relación causal).
- **Mejora**, intenta mejorar algún aspecto del fenómeno estudiado.

También el método depende de la perspectiva de la investigación o teoría subyacente. Así, por ejemplo, podemos adoptar una aproximación *positivista*, buscando evidencia de proposiciones formales, variables medibles, hipótesis de prueba, etc.; *crítica* o *interpretativa*, con el fin de entender el fenómeno mediante la interpretación de los participantes.

En Wieringa (2013) se comparan tres clasificaciones de métodos de validación, las propuestas en Zelkowitz y Wallace (1998), en Glass *et al.* (2001) y la propia de Wieringa (2013) (ver Tabla 1.2).

Wieringa (2013)	Zelkowitz y Wallace (2013)	Glass *et al*. (2001)
Métodos de validación		
Opinión del experto		
Experimento de caso-único	Simulación Análisis dinámico	Experimento de campo Experimento de laboratorio (con *software*) Simulación
Investigación en acción técnica	Estudio de caso	Investigación-acción
Experimento para establecer diferencias estadísticas	Experimento replicado Experimento en entorno artificial	Experimento de campo Experimento de laboratorio (con personas)
Otros métodos de investigación		
Estudio de caso observacional	Estudio de caso Estudio de campo	Estudio de caso Estudio de campo
Método de meta-investigación	Búsqueda de literatura	Revisión/análisis de la literatura
Métodos de medición		
Métodos para recolectar datos	Seguimiento de proyectos Datos heredados Lecciones aprendidas	Etnografía
Técnicas de inferencia		
Técnicas para inferir información a partir de los datos	Análisis estático	Análisis de datos Teoría fundamentada Hermenéutica Análisis de protocolo

Tabla 1.2. Resumen de métodos de validación y otros métodos según Wieringa (2013)

Aunque existen decenas de métodos de investigación, este libro se centrará en los cuatro más utilizados en ingeniería del *software*:

- **Encuestas**. Son investigaciones que proporcionan una visión general, mediante la recogida de información estandarizada de una población específica o una muestra representativa de la misma (sujetos del estudio), por medio de un cuestionario o entrevista.
- **Experimentos**. Un experimento controlado o cuasi experimento (si los sujetos no se asignan aleatoriamente), se caracteriza por medir el efecto de manipular una variable en otra variable, habiendo aleatorizado las posibles variables perturbadoras.

- **Estudios de caso**. Son estudios de campo u observacionales, son investigaciones en las que no hay aleatoriedad de variables perturbadoras ni representatividad de la muestra. Este tipo de método se basa en varias fuentes de evidencia para investigar una instancia o un número pequeño de ellas de un fenómeno de ingeniería del *software* en su contexto real, no estando claramente definidos los límites entre el fenómeno y el contexto. Dentro de este tipo consideramos los denominados métodos etnográficos.
- **Investigación-acción (IA)**. Es un tipo de investigación que pretende cambiar el fenómeno estudiado y en el que es fundamental la inmersión del investigador en la realidad que se está investigando.

1.3 COMPARATIVA ENTRE LAS ESTRATEGIAS EMPÍRICAS

Además de las consideraciones que hemos realizado en el apartado anterior, es posible realizar una comparación de las estrategias de acuerdo a los siguientes factores (ver Tabla 1.3):

- **Control de la ejecución**. Describe cuánto control tiene el investigador sobre la investigación. Por ejemplo, en un estudio de caso los datos son tomados durante la ejecución de un proyecto. Si se decide suspender el estudio, por ejemplo por razones económicas, el investigador no puede continuar con el estudio. Por el contrario, en un experimento el investigador tiene control pleno sobre la ejecución.
- **Control de la medición**. Es el grado en el que el investigador puede decidir las medidas que deben ser recogidas y las que deben ser incluidas o excluidas durante la ejecución del estudio. En una encuesta no es posible incluir medidas.
- **Coste de la investigación**. Dependiendo de la estrategia elegida, el coste asociado a la investigación varía. Esto está relacionado, por ejemplo, con el tamaño de la investigación y las necesidades de recursos. La encuesta es la estrategia de menor coste, ya que no requiere grandes cantidades de recursos.
- **Facilidad de réplica**. Se refiere a la facilidad con la que podemos replicar la situación que estamos investigando. Gracias a la posibilidad de replicación de los experimentos sus resultados son más generalizables.
- **Tiempo**. En cuanto a la duración de la investigación.

De todas maneras hay que tener en cuenta que se trata más bien de un continuo en el que se pueden ir clasificando los diferentes métodos a lo largo de estas dimensiones. De hecho, hay que tener en cuenta que los experimentos parten del control y se extienden hacia el realismo igual que sucede con la IA técnica, mientras que los estudios de caso y la IA parten del realismo en busca de mayor control.

Factor	Encuesta	Experimento	Estudio de caso	Investigación en acción
Tipo de diseño	Fijo	Fijo	Flexible	Flexible
Objetivo Principal	Descriptivo	Explicativo	Exploratorio	Mejora
Naturaleza de los datos	Cuantitativa Cualitativa	Cuantitativa	Cualitativa Cuantitativa	Cualitativa Cuantitativa
Control de la ejecución	No	Si	No	No
Control de la medición	No	Si	Si	Si
Coste	Bajo	Medio	Alto	Muy Alto
Facilidad de réplica	Alto	Alto	Bajo	Bajo
Tiempo	Horas	Días/meses	Meses/Años	Meses/Años

Tabla 1.3. Comparativa de estrategias empíricas

Una investigación completa, combinará varios métodos con el fin de reforzar la validez interna y externa de los resultados y recoger la mayor evidencia posible y fiable sobre la técnica, herramienta, método, o cualquier otro tipo de artefacto que se esté investigando. En el capítulo 7 se muestran varios ejemplos del uso de métodos de investigación de manera combinada.

Easterbrook *et al.* (2008) presentan algunas directrices para poder seleccionar la estrategia empírica apropiada según los objetivos del problema investigado.

1.4 CONTEXTUALIZACIÓN DE LA INVESTIGACIÓN

Es importante contextualizar los resultados obtenidos tras una investigación, o como dice Dybå (2013) contextualizar la evidencia empírica, para poder determinar el alcance de la aplicabilidad de los resultados obtenidos. Ya que no es posible obtener resultados válidos universalmente, habrá que determinar en qué contexto concreto son válidos. Y para definir el contexto es útil seguir las buenas prácticas periodísticas definiendo: quién, qué, cuándo, dónde y porqué (Johns, 2006).

Qué es lo que funciona, para quién, dónde, cuándo, y por qué es la gran pregunta de la ingeniería del *software* basada en la evidencia. Aunque la investigación empírica parece aún preocuparse de identificar relaciones universales que son independientes de cómo interactúan los entornos de trabajo y otros contextos con los procesos importantes para la práctica del *software*. Las preguntas como "¿Cuál es el mejor?" parecen prevalecer. Por ejemplo, "¿Qué es mejor: programación en solitario o programación por pares? ¿Probar al inicio o al final?". Sin embargo, estas preguntas universales no tienen sentido ya que sus respuestas dependen del entorno, la configuración y de los objetivos de los proyectos estudiados. Por ejemplo, los entornos de las organizaciones de *software* son diferentes, al igual que sus tamaños, tipos de clientes, países o geografía e historia. Además, los factores humanos que subyacen en la cultura de la organización difieren de una organización a otra, y también influyen en la forma en la que el *software* se desarrolla. Por lo tanto, reconocer estos aspectos y la manera en la que se interrelacionan es importante para la utilización eficaz de la investigación en la práctica. Sin embargo, la naturaleza de estas relaciones es poco conocida, y en consecuencia, no podemos asumir a priori que los resultados de un estudio en particular, sean aplicables fuera del contexto específico en el que se llevó a cabo.

Dybå *et al.* (2012) ofrecen una visión general de cómo el contexto afecta a la investigación empírica y proponen una manera de contextualizar mejor la evidencia empírica para que otros puedan entender qué es lo que funciona, para quién, dónde, cuándo, y por qué. La Tabla 1.4 presenta un paralelismo con los temas de investigación en ingeniería del *software*. Esta perspectiva amplia (que denomina "ómnibus") es más importante que centrarse en variables específicas, como hacen la mayoría de los estudios empíricos hoy en día. El contexto ómnibus muestra una perspectiva periodística y el contexto discreto refleja la visión

tradicional orientada a variables. En esta visión tradicional, además de ser crucial la selección de qué variables de contexto considerar, el número de ellas es crítico, con el fin de evitar una complejidad combinatoria.

Contexto "ómnibus"				
¿Qué? – Fenómeno	¿Quién? – Sujetos	¿Dónde? – Ubicación	¿Cuándo? – Tiempo	¿Por qué? – Razón fundamental
Contexto discreto				
Técnico Complejidad Tecnología Tarea/sistema …	Social Habilidad individual Autonomía del equipo Estructura de la organización …	Ambiental Incertidumbre Comunidad Mercado …		

Tabla 1.4. Dimensiones de los contextos en la ingeniería del software

Tenemos que cambiar el foco de la atención desde un enfoque basado en listas de comprobación para contextualizar la evidencia empírica a un enfoque más dinámico de la práctica del *software*. En lugar de ver un conjunto de variables discretas que estáticamente rodean partes de la práctica, podemos ver las relaciones entre la evidencia empírica y el contexto como un proceso que surge y cambia a través del tiempo y el espacio. Es crucial reconocer que cualquier definición de contexto puede ocurrir sólo en relación con una situación específica de la práctica.

Dado que ningún estudio ofrece una infinidad de factores contextuales y combinaciones a considerar, la decisión en cuanto a los parámetros por los que contextualizar no debe ser diferente de la decisión con respecto a las variables de control. Ambas decisiones deben basarse en los objetivos y las teorías relacionadas con el fenómeno bajo estudio.

La contextualización requiere de la inmersión en el fenómeno bajo estudio, lo que significa que los investigadores deben invertir considerable tiempo en la práctica que están intentando entender.

1.5 ASPECTOS ÉTICOS

Aunque la investigación en ingeniería del *software* no presenta muchas cuestiones éticas, como puede ser en el caso de la medicina o la genética, es necesario tenerla en cuenta.

Históricamente, pocos investigadores han prestado atención a este tema, aunque ya hace más de una década Singer y Vinson (2001) hicieron una importante llamada de atención sobre los aspectos éticos en la investigación en ingeniería del *software*.

En la actualidad, lamentablemente, ni las agencias de financiación ni las universidades suelen contemplar los aspectos éticos en la investigación en ingeniería del *software*. Por otro lado, los códigos deontológicos de asociaciones como, por ejemplo, IEEE o de algunas sociedades informáticas específicas, tampoco suelen contemplar los aspectos éticos de la investigación en este campo.

Como señala Sieber (2001), si no tenemos en cuenta los aspectos éticos, se puede causar daños a algunos de los implicados (*stakeholders*) en la investigación:

- **Personas**, **empresas e investigadores** si no se respeta su propiedad intelectual.
- **Ingenieros de *software*** que participan o eligen no participar en la investigación.
- **Estudiantes** a los que se les pide que sirvan como sujetos.

Este autor señala diferentes tipos de riesgos en la investigación en ingeniería del *software*:

- **Inconveniencia**, como aburrimiento, frustración o pérdida de tiempo por parte de los sujetos.
- **Riesgo psicológico**, debido a la preocupación sobre posibles críticas a la forma de trabajar, pérdida de reputación, falta de confidencialidad, la forma en que los resultados de la investigación pueden influir en su carrera (para los profesionales) o notas (en el caso de estudiantes).
- **Riesgo social**, como la desaprobación por parte de los pares (colegas) o la estigmatización al revelarse cierta información.

- **Riesgo económico**, además de la pérdida de propiedad intelectual ya mencionada, pérdida de empleo, de oportunidad o de ingresos.
- **Riesgo legal**, debido a demandas por daños causados a la reputación de la empresa o de individuos

Por todo ello, una de las cuestiones más importantes es la de informar a los sujetos de la naturaleza de la investigación y de sus posibles riesgos, sobre todo sobre los procedimientos que se seguirán en lo concerniente a la difusión de los resultados, especialmente la privacidad y confidencialidad.

Para ello lo ideal es utilizar un consentimiento informado en el que se especifiquen todas estas cuestiones y se asegure una comunicación continua entre el investigador y los sujetos ante cualquier duda que pueda surgir. Recientemente, Runeson *et al.* (2012) proponen incluir un escrito con información a los participantes en este sentido:

"*Esta investigación la lleva a cabo el Instituto XXX*: *explicación del Instituto*:....

Se ha firmado un contrato lo que significa que la información es confidencial:...

> *El Prof. YYY es el Director del Instituto y se le puede contactar por teléfono...mail..*

Queremos enfatizar que:

- su participación es completamente voluntaria

- es libre de rechazar contestar cualquier pregunta

- es libre de retirarse en cualquier momento

Los resultados serán totalmente confidenciales y disponibles solo a los miembros del equipo de investigación o: *si tuviera lugar una evaluación de la calidad*: *a evaluadores con las mismas condiciones de confidencialidad*. *Puede haber extractos de los resultados pero bajo ninguna circunstancia su nombre o identificación serán incluidos en el informe*."

Hay que tener en cuenta además que los investigadores suelen firmar contratos o convenios de investigación en los que suele haber cláusulas como las que se muestra en la Figura 1.1.

SÉPTIMA. CONFIDENCIALIDAD

Ambas partes se comprometen a no difundir de ninguna forma la información técnica científica o comercial a la que hayan podido tener acceso durante el desarrollo del trabajo, sin que conste autorización expresa de la otra parte, mientras esas informaciones no sean de dominio público o su revelación sea requerida judicialmente.

OCTAVA. PUBLICIDAD DE LOS RESULTADOS

Cuando una de las partes desee utilizar resultados parciales o finales, en su totalidad o parcialmente, para su publicación como artículo, exposición en conferencia o congreso, o en cualquier otra modalidad de difusión, habrá de solicitar la conformidad de la otra parte mediante carta certificada dirigida al responsable de la misma en el seguimiento del Contrato.

La otra parte deberá responder en un plazo máximo de treinta días, comunicando su autorización, sus reservas o su disconformidad sobre la información contenida en el artículo, conferencia, etc. Transcurrido dicho plazo sin obtener respuesta, se entenderá que el silencio es la tácita autorización para su difusión.

En toda publicación, conferencia o informe en el que se haga uso de resultados parciales o finales de este trabajo, figurarán siempre los autores del trabajo en su condición de tales, y como inventores en el caso de patentes. En cualquiera de estos casos, se hará siempre referencia al presente contrato.

Figura 1.1. Cláusulas ejemplo de convenios o contratos (*fuente UCLM*)

A pesar de todo esto, a veces resulta difícil que no se identifique a las organizaciones o sujetos, y mucho más aún, sin reducir la replicabilidad de la investigación (o la calidad de los datos de la misma). Para paliar estos problemas Becker-Kornstaedt (2001) propone manipular los datos utilizando principalmente dos técnicas: anonimización (eliminar los nombres o características únicas que permitan identificar a una persona o grupo) y saneamiento (*sanitization*), eliminar toda información sensible para que pueda ser distribuida a una audiencia mayor.

En cualquier caso, conviene tener siempre presente que las organización tiene derecho a que no se desvelen ciertas deficiencias en sus prácticas de ingeniería del *software*, y los sujetos que participan en las investigaciones a que no se revele nada sobre su rendimiento, o de lo que comentan al investigador, por ejemplo, cuando admiten que no se siguen los procedimientos, técnicas o procesos establecidos como piensa la dirección de la empresa.

1.6 COLABORACIÓN EN INVESTIGACIÓN ENTRE LA INDUSTRIA Y LA UNIVERSIDAD

En este apartado comentaremos, en primer lugar, algunos aspectos que dificultan la colaboración entre la universidad y la industria, para a continuación presentar un modelo para materializar la transferencia de tecnología entre universidad y algunos factores que potencian el éxito en esta transferencia.

1.6.1 Dificultades para la colaboración en la investigación

La ingeniería del *software*, como cualquier otra disciplina científica aplicada como la química o la medicina, tiene dos objetivos fundamentales: aumentar el conocimiento (teórico) para entender por qué las cosas ocurren en un área particular de interés; y mejorar las técnicas (práctico) de forma que los resultados de la investigación sean útiles. Por lo tanto, es fundamental establecer una buena base teórica en ingeniería del *software*, pero siempre con el objetivo de obtener resultados útiles en la práctica. Esta no es la situación habitual, en la que a menudo se proponen nuevas ideas, métodos, etc. sin aplicación en la práctica, existiendo una clara desconexión entre la investigación teórica y su aplicación (Moody, 2000) tal y como se puede observar en la Figura 1.2.

Si el objetivo es eliminar esta barrera entre la teoría y la práctica en ingeniería del *software* es necesario que la investigación esté orientada a objetivos prácticos, y que la industria del *software* aplique los resultados obtenidos en la investigación.

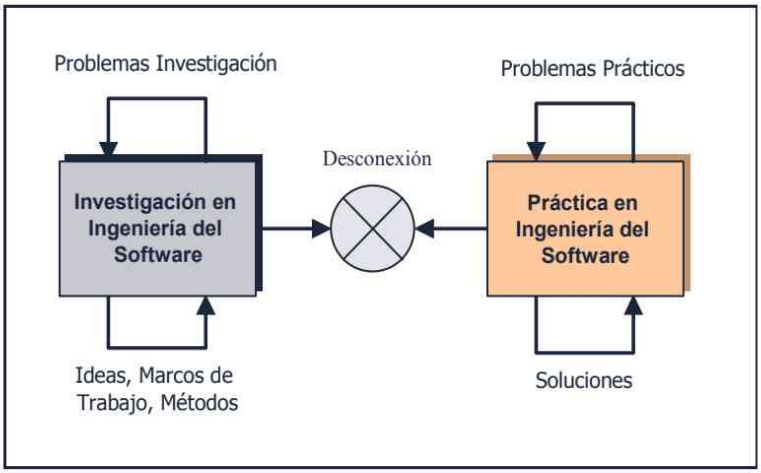


Figura. 1.2. Desconexión entre investigación y práctica en ingeniería del software (*Moody*: *2000*)

En efecto, es importante que se utilicen los conocimientos científicos en la resolución de problemas técnicos; pensando que "ni todo es teoría ni todo resolver problemas técnicos". Somos conscientes, que es difícil conseguir el equilibrio entre estos dos extremos; en este sentido, Kock *et al.* (2002) señala que hay expertos que afirman que la investigación que no es relevante para la industria tiene un efecto muy negativo; pero, por otro lado, que muchas veces se quiere "disfrazar" de investigación "relevante" labores de consultoría llevadas a cabo sin ningún rigor científico.

Nosotros compartimos que es necesario y deseable como señala Botella (2001), que "*hay otro fin que debería sustentar siempre la investigación pública y tiene que ver con su destinatario final. Entiendo que la investigación pública ha de tener un fin social que contribuya al progreso, y la investigación tecnológica ha de ser el motor de la industria, aportar soluciones a los problemas planteados por ella*". Lamentablemente, en muchas ocasiones la investigación se concibe exclusivamente en función del propio beneficio de los investigadores y con el único fin de publicar en las revistas más importantes (Kock *et al.*, 2002).

La relación con la industria es necesaria, y como señala Parnas (1998) muy conveniente, ya que la industria puede aportar al investigador problemas muy interesantes en los que profundizar desde un punto de vista teórico. Como confiesa alguien tan poco "sospechoso" de ser investigador aplicado y a la vez uno de los mayores expertos ("formales") en el campo de las bases de datos, como puede ser J. Ullman: "*creo que uno de mis más grandes fallos ha sido gastar demasiado tiempo haciendo cosas de lápiz y papel y no salir fuera a conocer los sistemas. Soy demasiado viejo y vago ahora para hacerlo y probablemente debería haberme disciplinado en años anteriores a hacerlo*" (Ullman, 2001). Denning (2002) también destaca que "*el papel natural de los científicos informáticos es la custodia*

del núcleo de conocimientos intelectuales y científicos del campo. Este importante papel debe ser desempeñado por alguien si la profesión de TI ha de alcanzar la coherencia deseada. Sin embargo: esto no ocurrirá de forma automática. Ocurrirá si los científicos informáticos aprenden a estrechar sus lazos con las aplicaciones comerciales: las interacciones con otros campos y las preocupaciones de sus clientes. Esto puede ser un abismo demasiado ancho de cruzar para muchos científicos informáticos".

Esta relación necesaria entre investigación académica e industria es difícil ya que, como señala Runeson (2012), los principales productos de trabajo de la investigación académica son publicaciones en revistas y conferencias que no están diseñadas como canal de comunicación entre la academia y la industria, por lo que se necesita una comunicación más directa para obtener beneficios mutuos de colaboración. Dentro de esta comunicación, uno de los aspectos más efectivos, pero a la vez más difíciles de conseguir, es la realización de tesis doctorales "industriales" que bien puedan ser realizadas por los propios profesionales de la industria o bien se trate de doctorandos que se incorporen a las empresas una vez finalizada su tesis doctoral[1].

1.6.2 Transferencia de tecnología entre universidad e industria

Existen multitud de modelos para la transferencia de tecnología, uno de los más conocidos en ingeniería del *software* es el propuesto por Gorschek *et al.* (2008), que puede verse en la Figura 1.3.

Figura 1.3. Modelo de transferencia de tecnología en ingeniería del software (Gorschek et al.: 2008)

[1] Hay que destacar en este sentido programas como "Torres Quevedo", del Ministerio de Economía y Competitividad, que en España facilitan la inserción de doctores en las empresas.

En este modelo se identifican los siguientes siete pasos:

1. Identificar áreas de mejora potenciales basadas en necesidades industriales, debiendo la investigación conectar con las necesidades para obtener el compromiso necesario.
2. Formular una agenda de investigación, de acuerdo con un conjunto de necesidades priorizadas.
3. Formular una solución candidata, que sea realista y se adapte a la situación y prácticas de la empresa.
4. Llevar a cabo la validación en laboratorio, en un entorno experimental, normalmente utilizando alumnos como sujetos.
5. Llevar a cabo la validación estática, presentando la solución candidata en la industria y recogiendo la retroalimentación de los profesionales.
6. Llevar a cabo la validación dinámica, mediante proyectos piloto.
7. Lanzar la solución, incorporándola en el proceso real de la empresa.

El reto más importante para lograr una transferencia efectiva, no es sólo conseguir la colaboración con la Unidad de I+D de la organización, sino sobre todo que las propuestas sean asumidas por los departamentos de desarrollo de *software* en proyectos reales. Nuestra experiencia en este sentido ha sido muy positiva en varios casos, el más significativo con la empresa Indra, multinacional de tecnología líder en España y una de las principales de Europa y Latinoamérica que cuenta con más de 42.000 profesionales y con clientes en 118 países. Indra tiene una red de más de veinte centros especializados en el desarrollo de *software*, denominados "*Software Labs*" repartidos por todo el mundo: además de en España, en Europa del Este (Moldavia, Eslovaquia), Latinoamérica (México, Argentina, Colombia, Brasil y Panamá), Sudeste asiático (Filipinas) y Australia. En marzo de 2000 se firmó un acuerdo de colaboración entre la empresa y nuestra universidad, que dio lugar en 2001 a la firma de un convenio específico de colaboración para la creación de un Centro Mixto de Investigación y Desarrollo, en el propio campus de la UCLM, en el que trabajan más de 500 ingenieros informáticos, que llevan a cabo proyectos de I+D que permiten la creación y el desarrollo de nuevas tecnologías, que posteriormente son adoptadas por la empresa.

En Sandberg *et al.* (2011) se presenta el resultado de otra fructífera relación entre industria y academia en lo que denominan CPR (*Collaborative Practice Research*), en que de manera conjunta trabajan los investigadores y los profesionales, utilizando experimentos, estudios de casos e investigación-acción. En su opinión los principales factores críticos de éxito son:

- Involucrar a la dirección en la formulación del problema y la gestión y conducción de la investigación.
- Tener acceso a los mejores empleados y competencias de la empresa.
- Ser capaz de comunicar ideas, progreso, resultados, etc.
- Escoger temas de interés durante un considerable periodo de tiempo.
- Abordar problemas reales de la industria.
- Alinear los resultados con los objetivos de la industria.
- Obtener resultados con un impacto real en la práctica.
- Generar nuevas ideas, conocimiento, patentes, procesos, métodos, técnicas y publicaciones.

Recientemente, en Wohlin *et al.* (2012) se confirman estos factores, y se señalan los tres más críticos:

- Compromiso y soporte por parte de la dirección de la empresa.
- Existencia de un patrocinador en la empresa.
- Actitud y habilidades sociales del investigador.

Estos investigadores, también analizan las diferencias de percepción entre los académicos y los profesionales, destacando la importancia que dan estos últimos al compromiso de los investigadores para contribuir y ayudarles, y su focalización en los resultados.

En Pareto *et al.* (2012) también se recogen lecciones aprendidas similares para conseguir una colaboración fructífera entre la industria y la academia:

1. No presionar, en el sentido de que los investigadores a veces apremiamos demasiado a la industria con herramientas o teorías que la industria no ve (todavía) necesarias o útiles.
2. Escuchar, ya que tanto los investigadores como los profesionales necesitan comprender mejor sus necesidades y formas de trabajo.
3. Conocer su tiempo, es decir, saber qué le interesa al profesional en cada momento.
4. No ser aburrido, quizás enfatizando demasiado los aspectos teóricos que pueden no interesar al profesional.

5. Ser generalista, en el sentido de conocer las teorías, herramientas, procesos, arquitecturas, etc. y también los temas empresariales de organización, mercados, etc.
6. Correr riesgos, y salirnos de nuestra zona de confort probando cosas nuevas.
7. Aprender a bailar el "*quickstep*", ya que la industria se mueve muy deprisa.
8. Ser ágil, satisfaciendo al cliente, trabajando "diariamente", y abrazando el cambio constante.

1.7 USO DE TEORÍAS EN LA INGENIERÍA DEL SOFTWARE

Construir y usar teorías, son prácticas muy establecidas en las ciencias maduras, como medio para obtener y acumular conocimiento y que pueden beneficiar tanto a los investigadores como a la industria como muestra la Figura 1.4. En la ingeniería del *software* el uso de teorías ha sido poco investigado, aunque como dicen Johnson *et al* (2012) "*la ingeniería del software está llena de teorías implícitas*: *sólo es necesario que salgan a la luz y que se las estudio con el rigor científico que merecen*".

Figura 1.4. Utilidad de las teorías para la investigación y la industria

A continuación, resumiremos algunos artículos que se han realizado sobre el uso de teorías en la ingeniería del *software*. En Hannay *et al.* (2007) se presenta una revisión sistemática de la literatura sobre el uso de teorías en los experimentos, que recopiló 103 artículos publicados en las principales revistas y conferencias en el ámbito de la ingeniería del *software* durante los años 1993 y 2002. Esta revisión de la literatura pone de manifiesto que de los 103 artículos encontrados, 24 usan un

total de 40 teorías para explicar las relaciones causa-efecto, investigadas. Sólo 2 de esas 24 teorías se utilizaron en más de un artículo: la teoría "*Probabilistic model of Perspective-based reading and Checklist-based reading inspection*" (Modelo probabilístico de lectura basada en perspectivas e inspección de lectura basada en listas de control) y la "*Theory of cognitive fit*" (Teoría del ajuste cognitivo). La mayoría de los artículos que usan algún tipo de teoría, lo hacen en el diseño experimental para justificar las preguntas de investigación e hipótesis, otros las usan para dar explicaciones a los resultados obtenidos, y unos pocos para probar o modificar la teoría. Hall *et al.* (2009) realizaron un estudio similar al de Hannay *et al.* (2007) en el que investigaron el uso de teorías en estudios sobre la motivación de los ingenieros de *software*. Uno de sus descubrimientos fue que muchos de los 92 estudios analizados no se habían basado en las teorías clásicas de la motivación que se originaron en las ciencias sociales.

Además existen algunas iniciativas interesantes como la del SEMAT (*Software Engineering Method and Theory*, *www.semat.org*) y GUTSE (*Grand Unified Theory of Software Engineering*, *http://books.google.es/books/about/The_grand_unified_theory_of_software_eng.html?id=TLcceL3NEiMC&redir_esc=y*), que demuestran que el uso de teorías en la ingeniería del *software* está cobrando cada vez más interés.

Comentaremos a continuación algunas propuestas que permiten clasificar teorías o que permiten definir los elementos que debe tener una teoría.

Un paso importante para usar y/o construir teorías en la ingeniería del *software* es conocer los tipos de teorías existentes. En este sentido Gregor (2006) ha contribuido proponiendo una clasificación que consiste en cinco tipos de teorías:

- **Análisis**. Las teorías de este tipo describen el objeto de estudio, e incluyen por ejemplo, taxonomías, clasificaciones y ontologías.
- **Explicación**. Este tipo de teorías explican por qué ocurre algo, pero carecen de poder predictivo.
- **Predicción**. Estas teorías predicen que va a ocurrir sin dar explicaciones, por ejemplo, en términos de modelos matemáticos o probabilísticos.
- **Explicación y predicción**. Estas teorías combinan las dos teorías precedentes, y es lo que se suele llamar "teorías con base empírica".
- **Diseño y acción**. Estas teorías describen como hacer las cosas e incluyen principios de diseño. Este tipo de teorías por lo general no proporcionan los mecanismos para explicar las relaciones causa-efecto que se

investigan. En cambio, se limitan a postular la existencia de la relación, a menudo sin considerar el factor humano, que es un aspecto esencial en los experimentos llevados a cabo en el contexto de la ingeniería del *software*. Por ello, muchos cuestionan si estas se deben considerar realmente teorías.

Además de la conveniencia de usar teorías en la investigación empírica en la ingeniería del *software*, se pueden usar los estudios empíricos para construir teorías, como señalan Sjøberg *et al.* (2008). La descripción de una teoría debe estar dividida en cuatro partes:

- **Constructos** (¿cuáles son los elementos básicos?).
- **Proposiciones** (¿cómo interactúan los constructos?).
- **Explicaciones** (¿por qué son las proposiciones a las que se refiere?).
- **Alcance** (¿en qué universo del discurso es aplicable la teoría?).

En la ingeniería del *software* se supone que una teoría explica un fenómeno que ocurre en la ingeniería del *software*. La situación típica en la ingeniería del *software* es que un actor aplica tecnologías para llevar a cabo ciertas actividades en un sistema de *software* (existente o en proyecto). En particular, cada constructo debe pertenecer a, o derivarse de una de las cuatro clases arquetipo: Actor, Tecnología, Actividad y Sistema *software*, como se muestra en la Tabla 1.5.

Clase arquetipo	Subclases
Actor	Individuo, equipo, proyecto, organización o industria
Tecnología	Modelo, método, técnica, herramienta o lenguaje
Actividad	Planificar, crear, modificar o analizar (un sistema *software*)
Sistema *software*	Los sistemas *software* se pueden clasificar considerando diversas dimensiones, como tamaño, complejidad, dominio de aplicación, sistemas de gestión, de tiempo real, empotrados, etc.

Tabla 1.5. Marco definido por Sjøberg et al. (2008) para construir teorías

Además Sjøberg *et al.* (2008) proponen un proceso para construir teorías que consta de cinco actividades y muestran un ejemplo de cómo construir una

teoría para el desarrollo de grandes proyectos basados en UML a partir de estudio de caso exploratorio. Las actividades de este proceso son:

- Definir los constructos de la teoría.
- Definir las proposiciones de la teoría.
- Ofrecer explicaciones para justificar la teoría.
- Determinar el alcance de la teoría.
- Probar la teoría a través de estudios empíricos.

Para concluir podemos decir que, aunque se reconoce la necesidad de teorías en la ingeniería del *software*, ninguna de las propuestas existentes ha tenido gran impacto.

La teorías son importantes para la conceptualización y comunicación del conocimiento dentro de un campo de investigación, y son útiles cando se agregan estudios existentes o se planifican réplicas. Las teorías también se pueden usar para comunicarse con los profesionales en la toma de decisiones, ya se trate de decisiones estratégicas de la tecnología o decisiones de proyectos basadas en predicciones. Por lo tanto, es fundamental construir teorías en la ingeniería del *software*, para que la ingeniería del *software* se convierta en un campo maduro de la ciencia.

1.8 LECTURAS RECOMENDADAS

- **Endres**, **A**. y **Rombach**, **D**. (2003). *A handbook of software and systems engineering*: *empirical observations*: *laws and theories*. Pearson. En este libro se presentan diversas leyes y teorías de la ingeniería del *software* que damos por ciertas, pero que nunca han sido investigadas ni validadas con rigor, por lo que nos invita a pensar en la necesidad de contar con buenos métodos de investigación que nos permita avanzar en nuestra disciplina con unos buenos fundamentos.
- **León**, **O**. **G**. y **Montero**, **I**. (2012). *Métodos de investigación en psicología y educación*. Madrid: McGraw-Hill. Otras disciplinas como la Psicología y la Educación conceden importancia a los métodos de investigación desde hace mucho más tiempo que la informática. Este libro es un clásico en estas áreas y explica de manera didáctica y amena los diferentes conceptos y métodos utilizados en la investigación.

1.9 SITIOS WEB RECOMENDADOS

- *http://www.apa.org/ethics/code/*

  Es el sitio de la *American Psychological Association* (APA) en el que se puede encontrar su código ético (*Ethical Principles of Psychologists and Code of Conduct*), es muy interesante leerlo para reflexionar sobre los diferentes aspectos éticos, especialmente su estándar número 8 relativo a investigación y publicación.

- *http://www.research.umn.edu/consent/*

  Este sitio la Universidad de Minnesota ofrece un pequeño tutorial sobre el consentimiento informado.

- *http://start.aisnet.org/?CodeofResearch*

  En este sitio se encuentra la última versión del Código de Conducta en Investigación de la *Association for Information Systems* (AIS) que puede servirnos de guía a la hora de abordar algunos dilemas éticos en la investigación en ingeniería del *software*.

Capítulo 2

ENCUESTAS

2.1 INTRODUCCIÓN

Las encuestas son, probablemente, el método de investigación más utilizado por todo el mundo. De hecho, es muy común que en nuestra vida diaria se nos ofrezca, muy a menudo, participar en una encuesta por nuestro papel de electores, consumidores o usuarios de servicios. Este uso tan extendido de las encuestas puede hacer que las investigaciones basadas en encuestas parezcan una forma fácil y directa de conseguir información importante sobre productos, contextos, procesos, personas, etc. Sin embargo, la realidad no es esa.

Una encuesta es un método empírico que se utiliza para recopilar información de o sobre personas para describir, comparar o explicar su conocimiento, sus actitudes o su comportamiento. También se pueden utilizar encuestas para la descripción de las características de métodos o herramientas.

En la mayoría de los casos, los datos relativos a la encuesta provendrán de cuestionarios. Pero los cuestionarios, por sí solos, no constituyen la encuesta. De hecho, una encuesta es un proceso más complejo formado por una serie de actividades bien definidas (Kitchenham y Pfleeger, 2008) que se enumeran a continuación y que se irán explorando a lo largo de este capítulo:

- Establecer los objetivos de la encuesta.
- Diseñar la encuesta.

- Desarrollar el cuestionario.
- Evaluar y validar el cuestionario.
- Obtener los datos de la encuesta.
- Analizar los datos obtenidos.
- Reportar los resultados.

2.2 PROCESO DE REALIZACIÓN DE ENCUESTAS

En este apartado se describen las tareas que es recomendable realizar en cada una de las actividades del proceso para la realización de encuestas.

2.2.1 Establecer los objetivos de la encuesta

Establecer los objetivos es siempre el primer paso de una encuesta o de cualquier otro tipo de investigación. En el caso concreto de las encuestas, cada objetivo debería establecerse como una frase relativa a los resultados esperados tras realizar la propia encuesta.

Algunos ejemplos válidos de objetivos podrían ser determinar el lenguaje de programación más atractivo para un determinado grupo de programadores de *software* o la metodología de desarrollo más utilizada por las empresas de un país.

El origen de los objetivos puede ser diverso y, por ejemplo, un objetivo puede surgir como resultado de una duda que haya aparecido tras una búsqueda en la literatura, puede que haya algún tipo de necesidad que se perciba por parte de los investigadores responsables y se pretenda explorar las distintas posibilidades asociadas a ella o puede, incluso, que los haya planteado directamente algún experto.

En cualquier caso, los objetivos de las encuestas deben ser lo más claros posibles y han de poder medirse. Dichos objetivos determinarán la mayoría del resto de actividades del proceso de realización de encuestas, de ahí que deban establecerse cautelosamente.

2.2.2 Diseñar la encuesta

Los dos tipos de diseño de encuestas más utilizados son (Kitchenham y Pfleeger, 2008):

- **Encuestas transversales**: en este tipo de estudio, se pide información a los participantes en un instante determinado. Por ejemplo, podríamos sondear a todos los miembros de la plantilla de una factoría de desarrollo de *software* a las 11:00 horas para saber qué actividades están llevando a cabo en ese preciso momento. Esta información nos daría una instantánea de qué ocurre en una organización en un momento determinado. La mayoría de encuestas que se realizan en la ingeniería del *software* son de este tipo.
- **Encuestas longitudinales**: en este tipo de estudio la meta se establece a más largo plazo y se trata de conocer la evolución a través del tiempo de una determinada población. Podemos encontrarnos con dos variantes principales de encuestas longitudinales, según si la población que elegimos en los distintos momentos en los que se realiza la encuesta es la misma o varía.

Hay algunos otros tipos más complejos de diseño de encuestas, por ejemplo, diseños que comparan poblaciones diferentes u otros que pretenden medir el impacto de un cambio. Si se está interesado en alguno de estos diseños se puede consultar *Shaddish et al.* (2002).

Otro aspecto importante a tener en cuenta a la hora de diseñar una encuesta es cómo se va a administrar. Existen, de nuevo, diversas opciones (Kitchenham y Pfleeger, 2008):

- Cuestionarios auto-administrados (vía Internet).
- Encuestas telefónicas.
- Entrevistas personales.

Elegir una u otra de estas opciones determinará el tipo de preguntas que podrán realizarse en la encuesta. Además, algunas estrategias para obtener datos fiables, como el orden de las preguntas o el vocabulario utilizado, también vienen influidas por cómo se vaya a administrar la encuesta.

Cabe destacar que en el campo de la ingeniería del *software*, los cuestionarios auto-administrados son la opción más utilizada, por lo que será este tipo el que se estudiará en este capítulo.

2.2.3 Desarrollar el cuestionario

En muchas disciplinas, los investigadores suelen basarse en cuestionarios ya existentes o, como mucho, en ligeras variaciones de los mismos. Este enfoque es útil puesto que los cuestionarios existentes ya se han validados con anterioridad y permite, además, comparar los resultados que se obtengan en el estudio con los previamente obtenidos.

Sin embargo, en la ingeniería del *software*, la mayoría de las investigaciones parten de cero, por lo que utilizar material ya existente no suele ser una opción viable y es necesario que los investigadores desarrollen sus propios cuestionarios.

Una encuesta pretende obtener respuestas sobre una serie de preguntas por una razón determinada. Por tanto, a la hora de diseñar un cuestionario conviene partir del objetivo de la investigación, si bien traducir directamente objetivos a preguntas rara vez lleva a un cuestionario realmente útil, por lo que, si se pretende conseguir una encuesta realmente efectiva habrá que diseñar correcta y cautelosamente los cuestionarios que se utilizarán para la recolección de los datos.

En el resto de sub-apartados se comentarán las distintas tareas a llevar a cabo para desarrollar correctamente el cuestionario. En concreto, la consulta de la literatura relevante a la investigación que se esté llevando a cabo a través de la encuesta, la elección del tipo de las preguntas del cuestionario, el diseño de las mismas, así como el diseño de las respuestas a las preguntas, el formato a seguir por los cuestionarios y, por último pero no por ello menos importante, los aspectos motivacionales a tener en cuenta para conseguir una participación suficiente en la encuesta, en cuanto a cantidad y calidad.

2.2.3.1 CONSULTA DE LA LITERATURA RELEVANTE

Toda buena investigación que se precie, ha de comenzar con esta fase que pretenderá conseguir un doble objetivo: por un lado se tratará de identificar qué otros estudios se han llevado a cabo con anterioridad sobre el mismo tema, por otro, será de gran ayuda conocer cómo se han conseguido los datos en estos estudios previos, si es que los hay. Más en concreto, habrá que centrarse en la posibilidad de conseguir los cuestionarios o cualquier otro método de recolección de datos que se haya realizado con anterioridad en investigaciones relacionadas con el mismo tema.

2.2.3.2 TIPOS DE PREGUNTAS

En un cuestionario pueden encontrarse dos tipos de preguntas: abiertas o cerradas. Una pregunta abierta permite al encuestado expresar la respuesta utilizando sus propias palabras, mientras que una pregunta cerrada proporciona una lista de posibles respuestas a la pregunta.

Cada tipo de pregunta tiene sus ventajas e inconvenientes. Las preguntas abiertas permiten que el encuestado se exprese libremente y utilizando el lenguaje que estime más conveniente pero, a la vez, ese mismo hecho se vuelve un inconveniente al permitir que se produzcan interpretaciones erróneas sobre las respuestas obtenidas o que se obtenga una gran cantidad de información irrelevante y no deseada. En resumen, las preguntas abiertas suelen ser difíciles de codificar y analizar, por lo que suele ser mucho más conveniente que el cuestionario esté formado por tantas preguntas cerradas como se pueda.

2.2.3.3 DISEÑO DE LAS PREGUNTAS

Una vez se tiene claro qué preguntar, es necesario pensar detenidamente cómo se quieren plantear las preguntas. Las preguntas han de ser precisas, carentes de ambigüedad y perfectamente comprensibles por parte de todas aquellas personas que vayan a responderlas. Para conseguir estas características conviene tener en cuenta los siguientes factores (Kitchenham y Pfleeger, 2008):

- El lenguaje utilizado es el apropiado para las personas que responderán la encuesta y cualquier término potencialmente ambiguo se deberá definir explícitamente.
- Se evitarán errores gramaticales, de puntuación o de deletreo.
- En cada pregunta se abordará un único concepto para conseguir preguntas concisas y concretas.
- Se evitará el uso de calificadores ambiguos.
- Nunca se utilizarán jergas ni expresiones coloquiales.
- Las preguntas podrán ser positivas o negativas, evitando dobles negaciones.
- Se evitará preguntar sobre sucesos acaecidos hace mucho tiempo.
- Se evitarán preguntas sensibles que incomoden a los participantes.
- Las preguntas se podrán responder de manera sencilla para evitar que los participantes se frustren al no conocer la respuesta a las mismas.

2.2.3.4 DISEÑO DE LAS RESPUESTAS

Las respuestas a las preguntas de un cuestionario suelen ser alguna de las contenidas en la siguiente lista:

1. Valores numéricos (p.ej. la edad).
2. Categorías (p.ej. el tipo de trabajo).
3. Respuestas SI/NO.
4. Escalas ordinales.

En el caso de los valores numéricos no se suelen encontrar dificultades, pero sí conviene hacer algunas recomendaciones sobre el resto.

En las respuestas enmarcadas en categorías será necesario que dichas categorías cumplan ciertos requisitos (Kitchenham y Pfleeger, 2008):

- Ser exhaustivas pero no demasiado largas.
- Ser mutuamente excluyentes.
- Permitir una selección múltiple si fuera necesario.
- Incluir una categoría "Otros" si no se cubren explícitamente todas las posibles respuestas.

Las respuestas SI/NO son especialmente problemáticas. Contienen un carácter restrictivo (sólo dos respuestas posibles y opuestas) y son poco fiables (la misma persona puede dar distintas opuestas en momentos diferentes). Es por ello que en respuestas que expresen actitudes o preferencias sea mucho más interesante utilizar respuestas en una escala ordinal. Hay tres tipos de escala:

1. Escalas de conformidad, p.ej. desde totalmente de acuerdo hasta totalmente en desacuerdo. Este tipo de escala también se conoce como escala de Likert (Likert, 1932).
2. Escalas de frecuencia, p.ej. desde nunca hasta siempre.
3. Escalas de evaluación, p.ej. desde terrible hasta excelente.

En la mayoría de los casos se recomienda utilizar una escala de entre cinco y siete posibles valores (Lethbridge, 1998). Además, conviene explicitar dichos valores y no utilizar números para etiquetarlos, puesto que pueden llevar a

interpretaciones por parte de los participantes que introduzcan un sesgo en los resultados (Krosnick, 1990). Como mucho, si se estima conveniente se puede incorporar una numeración a las etiquetas lingüísticas, especialmente si la diferencia entre dichas etiquetas es muy pequeña.

Un último aspecto a tener en cuenta sobre las respuestas es la posibilidad de incluir una opción NS/NC, es decir, "no sabe o no contesta". No hay un acuerdo en la comunidad científica sobre este aspecto aunque, como norma general, se recomienda no incluir dicha opción si los participantes han sido seleccionados porque deberían poder responder a todas las preguntas.

2.2.3.5 FORMATO DE LOS CUESTIONARIOS

Para conseguir que el aspecto físico de los cuestionarios que se entreguen en papel (aunque la mayoría de ellas son directamente aplicables a un cuestionario web) no introduzca complejidad innecesaria en la encuesta existen, entre otras, las siguientes recomendaciones (Kitchenham y Pfleeger, 2008):

- Dejar un espacio para que los participantes dejen sus comentarios sobre el cuestionario.
- Usar espacio entre las distintas preguntas.
- Utilizar un tamaño de fuente entre 10 y 12 puntos.
- Evitar el uso de cursivas.
- Enfatizar la información que lo requiera mediante el uso de negritas, subrayado y mayúsculas.
- No dividir entre páginas las instrucciones, una pregunta o las respuestas asociadas a una pregunta.

El orden en que se plantean las preguntas también es importante. Lo normal es seguir un orden lógico, haciendo que las preguntas más sencillas se encuentren al principio y que el nivel de dificultad avance según los participantes van rellenando el cuestionario.

En muchos casos, los cuestionarios contienen preguntas demográficas. Si bien no hay consenso sobre dónde colocar estas preguntas, deberán colocarse al principio o al final del cuestionario.

2.2.3.6 MOTIVACIÓN

Uno de los principales retos a la hora de abordar la realización de una encuesta es conseguir motivar a los participantes para que respondan de manera fiel a una encuesta en la que, en la mayoría de los casos, no han pedido participar.

En líneas generales, las personas estarán motivadas y darán respuestas completas y precisas siempre y cuando perciban que los resultados del estudio les serán útiles. Es por ello que se recomienda incluir cierta información clave en el cuestionario de la encuesta (Kitchenham y Pfleeger, 2008), como por ejemplo:

- ¿Cuál es el objetivo del estudio?
- ¿Cómo puede ser relevante para los participantes?
- ¿Por qué es importante la participación de cada individuo?
- ¿Cómo y por qué se eligió a cada participante?
- ¿Cómo se implementa la confidencialidad de las respuestas?

2.2.4 Evaluar y validar el cuestionario

Una vez se han definido las preguntas del cuestionario, los investigadores suelen pensar que ya se puede pasar a realizar la encuesta y obtener los datos de la misma. Éste es un error muy común que debe evitarse, ya que el conjunto de preguntas del cuestionario constituye únicamente el punto de partida de la construcción de los cuestionarios y, una vez creado, es esencial que se evalúe (Litwin, 1995).

En esta actividad de evaluación, o pre-test como también se conoce, se pretenden diversos objetivos (Kitchenham y Pfleeger, 2008):

- Comprobar que las preguntas se entienden correctamente.
- Evaluar el índice probable de respuestas.
- Evaluar la fiabilidad y validez del cuestionario, a través de grupos de discusión y/o de estudios pilotos.
- Comprobar que el método de análisis de datos que se utilizará será compatible con las respuestas que se van a obtener.

En los grupos de discusión se cuenta con un conjunto de personas representativo de aquellos que utilizarán los resultados de la encuesta o de aquellos que la realizarán (o incluso una mezcla de ambos). Se trata de que estos individuos rellenen el cuestionario e identifiquen cualquier problema asociado a él. Así, los grupos de discusión deberían identificar preguntas que faltan o que sobran, así como preguntas o instrucciones ambiguas.

En los estudios piloto, las encuestas se realizan tal cual se tienen diseñadas, pero utilizando una muestra mucho más pequeña de participantes. Estos estudios pretenden detectar problemas en el propio cuestionario, el tiempo permitido para responder, etc.

Una vez se ha construido y validado el cuestionario asociado a la encuesta conviene comenzar a documentar la misma (Bourque y Fielder, 1995). Normalmente, se suele puede comenzar con un documento descriptivo inicial, denominado especificación del cuestionario, que debería contener:

- Los objetivos del estudio.
- Una descripción del porqué de cada pregunta.
- Motivos para haber adoptado o adaptado preguntas de otras fuentes, incluyendo las citas apropiadas.
- Una descripción del proceso de evaluación.

Más tarde, cuando el cuestionario se haya publicado, se debe ir actualizando la información relativa a los datos de los participantes, cómo se administró el cuestionario, cómo se procesaron las preguntas, etc.

Un buen motivo para ir preparando esta documentación en paralelo con el desarrollo de la encuesta es que, en la mayoría de los casos, la recolección de datos puede extenderse mucho en el tiempo y puede que pasen meses entre que se distribuyen los cuestionarios y se comienzan a analizar los resultados. Esta dilación puede hacer que se olviden fácilmente detalles sobre la creación de los cuestionarios o sobre la gestión de los mismos.

2.2.5 Obtener los datos

A la hora de conseguir los datos, normalmente es imposible contar con las respuestas de toda la población implicada en el estudio, de ahí que haya que recurrir a una *muestra* de la misma, con la esperanza de que sus respuestas representen a las respuestas que hubiera dado el conjunto completo.

Dentro de los métodos de selección de la muestra, destacan tres tipos principales (Kitchenham y Pfleeger, 2008):

- **Métodos probabilísticos**. En estos métodos, cada elemento de la población objetivo tiene una probabilidad conocida y distinta de cero de ser incluido en la muestra.
- **Métodos basados en grupos** (*clusters*). En estos casos, cada individuo de la población objetivo pertenece a un grupo bien definido.
- **Métodos no probabilísticos**. Estos métodos se utilizan cuando los sujetos que realizarán la encuesta se eligen porque son fácilmente accesibles por parte de los investigadores responsables o porque se tiene alguna justificación para creer que son representativos de la población objetivo.

Con el fin de obtener una muestra representativa para una encuesta hay tres factores esenciales a tener en cuenta: ausencia de sesgos y grado de adecuación de los participantes y la relación coste-efectividad que presentan.

No hay que perder de vista varios aspectos a la hora de seleccionar quién va a participar en una encuesta. El primero de ellos es el tamaño de la muestra. Si el tamaño no es suficientemente grande, las conclusiones no serán razonables y será difícil generalizarlas. Eso sí, no hay una ecuación que marque exactamente cómo de grande debería ser una muestra (Fowler, 2002) ni hay una fórmula que permita comprobar cuándo un tamaño de la muestra pasa de ser no representativo a representativo. Lo que se recomienda es que se tenga una muestra lo más representativa (es decir, de un tamaño mayor) posible de todos los subgrupos de individuos que pudiera haber en la población.

En cualquier caso y, a pesar de lo imposible de la tarea, determinar el número de sujetos que participarán en la encuesta no es lo único importante. Cualquier encuesta que se precie ha de indicar su tasa de respuesta, es decir, el porcentaje de sujetos contactados para realizar la encuesta que finalmente completaron los cuestionarios. Aun así, una tasa de respuesta alta no implica que se vaya a conseguir resultados más precisos (Krosnick, 1990).

Encontrar un compromiso entre el tamaño de la muestra y la tasa de respuesta de una encuesta no es un proceso sencillo y en muchos casos dependerá de factores ajenos a los investigadores. Conviene, por tanto, utilizar los recursos con los que se cuente, como enviar recordatorios en listas de correo o, llegado el caso, individualmente a aquellos participantes potenciales que aún no hayan completado los cuestionarios.

2.2.6 Analizar los datos

Una vez diseñada y llevada a cabo la encuesta es el momento de pasar a analizar los datos obtenidos. En este apartado se tratarán los puntos más importantes relativos al análisis de los mismos. En concreto, se estudiará la validación de los datos obtenidos, la división en las respuestas en grupos homogéneos y, finalmente, el análisis de los datos obtenidos, tanto ordinales como nominales.

2.2.6.1 VALIDACIÓN DE LOS DATOS

Antes de embarcarse en ningún análisis, es necesario comprobar que los datos con los que se cuenta son completos y consistentes. Para ello, conviene contar con una política de gestión de cuestionarios inconsistentes. Si se observa que la mayoría de los participantes han respondido a todas las preguntas, probablemente convenga rechazar aquellos que estén incompletos. Si, por el contrario, hay una o más preguntas cuya respuesta se ha omitido sistemáticamente por parte de los participantes, convendrá rechazar esa(s) pregunta(s) en concreto.

Podría darse el caso de que, aun estando incompletos, convenga analizar todos los cuestionarios, lo que nos llevará a tener un tamaño de la muestra diferente para las preguntas. Si ocurre esta situación, habrá que documentarlo convenientemente y explicitar el tamaño de la muestra para cada pregunta. Conviene tener en cuenta que este enfoque se podrá utilizar para calcular estadísticos simples o comparar medias, pero no para estudios de correlación o regresión.

Para evitar estas situaciones es muy importante que los cuestionarios se validen con anterioridad a la realización de la encuesta.

2.2.6.2 DIVISIÓN DE LAS RESPUESTAS

En ocasiones, antes de comenzar a analizar será conveniente dividir y agrupar las preguntas para contar con grupos homogéneos. Estos agrupamientos se suelen hacer en función de la información demográfica que se ha obtenido de los participantes, por ejemplo, basándose en la edad, la localización geográfica, la experiencia en algún campo concreto, etc., según convenga en cada caso.

2.2.6.3 ANÁLISIS DE DATOS ORDINALES Y NOMINALES

Los datos numéricos generados por una encuesta se podrán analizar utilizando las técnicas habituales, sin embargo, puede que los datos ordinales y nominales generen algún problema adicional, por lo que serán éstos los que trataremos en este sub-apartado.

Si se ha utilizado un cuestionario que se responde en una escala ordinal, se suele convertir dicha escala en sus valores numéricos correspondientes (p.ej. desde 1 hasta 7) y analizar dichos datos como si fueran datos numéricos simples. Es una solución razonable pero que viola las reglas matemáticas para analizar datos ordinales, lo que conlleva un riesgo que puede dar lugar a unos resultados imprecisos.

Para evitar esta situación existen tres posibilidades principales:

1. Utilizar las propiedades de una distribución polinomial, estimar la proporción de la población en cada categoría y determinar el error estándar de la estimación (Moses, 2000).
2. Convertir una escala ordinal en una variable dicotómica. Por ejemplo, si se pretende calcular qué proporción de la población está de acuerdo con una determinada afirmación, pueden reagruparse las respuestas afirmativas (totalmente de acuerdo, de acuerdo, etc.) en una única variable (1) y el resto en otra (0) y utilizar las propiedades de una distribución binomial.
3. Utilizar el coeficiente de correlación de *Spearman* o la *tau de Kendall* para comprobar asociaciones entre variables de escala ordinal (Siegel y Castellan, 1998).

2.2.7 Reportar los resultados

En la mayoría de las ocasiones, se tratará de publicar la información generada durante la realización de la encuesta y el análisis de sus resultados de la misma en alguna revista o congreso o incluso como un informe técnico asociado a algún proyecto de investigación.

Hay una serie de elementos que deben incluirse en el documento que servirá como documentación de la encuesta. Se trata de los siguientes (Fink, 2003):

- Título, autores, patrocinadores, localización y fecha.
- Introducción, en la que se debe dejar bien claro cuál es el problema o la necesidad a resolver e incluso las preguntas de investigación y/o las hipótesis a ser validadas.
- Características de la encuesta:
  - Tipo de cuestionarios y justificación de la elección.
  - Contenido del cuestionario: número de preguntas, descripción del contenido de las preguntas, tipos de respuestas, etc.
  - Escalas utilizadas, valoración de las preguntas y si se han agrupado preguntas.
  - Fiabilidad y validez del cuestionario, incluyendo los estudios piloto realizados, el tiempo de respuesta y cualquier otro indicador de calidad que se haya utilizado.
- Características de los administradores de la encuesta.
- Literatura relevante sobre el tema de investigación.
- Diseño de la encuesta, muestra escogida y análisis de resultados.
- Relación de los resultados con los objetivos de la encuesta.
- Conclusiones alcanzadas, implicaciones derivadas de la encuesta y trabajo futuro.

2.3 FIABILIDAD Y VALIDEZ DE LAS ENCUESTAS

Una encuesta es fiable cuando, por más veces que se repita, se obtienen siempre unos resultados similares. La fiabilidad de una encuesta puede comprobarse repitiendo exactamente la misma encuesta (o con ligeras variaciones en cuanto a la redacción o el orden de presentación de las preguntas, por ejemplo) a los mismos sujetos y verificando si las respuestas obtenidas son similares.

A nivel estadístico se suele utilizar el coeficiente alfa de *Cronbach* (Cronbach, 1951) que, en función de su valor, indica la correlación de las distintas preguntas. Se suele recomendar un valor mínimo del coeficiente de 0,7 para poder afirmar que los distintos elementos están correlacionados.

En cuanto a la validez de las encuestas, se suelen estudiar distintos tipos (Kitchenham y Pfleeger, 2008):

- **Validez de contenido**: establece una visión subjetiva sobre lo apropiado que es el contenido de los cuestionarios en relación a los sujetos que los realizan.
- **Validez de criterio**: establece la capacidad del cuestionario de distinguir a qué grupo pertenece cada uno de los sujetos que lo responden.
- **Validez del constructo**: se refiere a cómo de bien se consigue medir a través del cuestionario aquello que se pretende medir.

Desafortunadamente, la importancia que en el campo de la ingeniería del *software* se le da al estudio de la fiabilidad y la validez de las encuestas es aún bastante escasa. Aun así, existen algunos ejemplos donde sí que se llevan a cabo, como el que se presenta en Dybå (2000). En este trabajo se presenta un estudio de la fiabilidad a través del estudio de diversas correlaciones entre los elementos de la encuesta y un estudio de la validez desde los tres puntos de vista comentados con anterioridad.

Algunos otros ejemplos de encuestas que incluyen estudios sobre la validez y la fiabilidad son: Humphrey y Curtis (1991) y Ropponen y Lyytinen (2000).

2.4 EJEMPLO DE ENCUESTA

A continuación, y a modo de ejemplo, se presenta una encuesta publicada en Carrillo de Gea *et al.* (2012) y que se realizó con el objetivo de conseguir información acerca de en qué modo y hasta qué punto las herramientas *software* destinadas a la Ingeniería de Requisitos (IR) dan soporte al propio proceso de la IR a través de sus distintas capacidades. Con el fin de facilitar la lectura, en los siguientes apartados se adaptará el contenido del artículo a las distintas actividades del proceso que se ha presentado en este capítulo.

2.4.1 Establecer los objetivos de la encuesta

El principal objetivo del trabajo es arrojar luz en el estado del arte relativo a las capacidades de las herramientas de la IR. Más formalmente se define el objetivo usando la plantilla GQM (*Goal-Question-Metric*) (Basili y Rombach, 1988), que se muestra en la Tabla 2.1.

Objeto de estudio	Herramientas de IR
Propósito	Caracterizar las principales características de las herramientas de IR y evaluar el escenario actual de las herramientas de IR.
Enfoque de calidad	Efectividad, coste, y presencia de las herramientas de IR.
Perspectiva	Investigadores y compradores de herramientas de IR.
Contexto	Este estudio se realiza con fabricantes de herramientas de IR y herramientas que aparecieron en alguna de las base de datos consideradas.

Tabla 2.1. Objetivo de la encuesta

Para conseguir el objetivo, en primer lugar se creó un marco de clasificación consistente en 146 *items* basándose en el informe técnico ISO/IEC TR24766 (ISO, 2009). Dicho informe presenta las características o capacidades recomendables que debería tener cualquier herramienta de soporte a la IR. El marco de clasificación propuesto consta de 8 categorías, cada una varios *ítems* o grupos (ver ejemplo en la Tabla 2.2).

Categoría	Grupo de capacidades
Elicitación de requisitos	Captura; Listas de comprobación, plantillas; Importación y exportación desde y hacia otras fuentes; Documentación de la elicitación.
Análisis de requisitos	Análisis de requisitos de calidad; Análisis de viabilidad; Análisis de atributos; Análisis y gestión de riesgos.
Especificación de requisitos	Documentación de la especificación de requisitos.
Modelado de requisitos	Análisis del modelado; Lenguajes de modelado y de especificación.
Verificación y validación (V&V) de requisitos	V&V.
Gestión de requisitos	Línea base de requisitos; Gestión de cambios en los requisitos; Gestión de proyectos; Modelo de datos abierto o cerrado.
Trazabilidad de requisitos	Trazabilidad; Flexibilidad en las trazas; Trazabilidad bidireccional; Análisis de la trazabilidad.
Otras capacidades	Información de gestión de las herramientas de requisitos; Interfaz de usuario gráfica; Integración de datos.

Tabla 2.2. Ejemplo del marco de clasificación

A partir de estas categorías, los autores establecen como hipótesis de trabajo que las herramientas de la IR actuales dan un soporte adecuado a:

- H_1: Elicitación de requisitos.
- H_2: Análisis de requisitos.
- H_3: Especificación de requisitos.
- H_4: Modelado de requisitos.
- H_5: Verificación y validación (V&V) de requisitos.
- H_6: Gestión de requisitos.
- H_7: Trazabilidad de requisitos.
- H_8: Otras capacidades.
- H_9: Todas las características anteriores.

Se establece además, que cada una de las características será tratada como una variable en la encuesta. Para calcular el valor de cada variable se tomará el porcentaje de características de cada categoría que presenta cada herramienta estudiada, en función de las respuestas afirmativas que los participantes en la encuesta den sobre las distintas funcionalidades que incorporan las herramientas. Se calculará también un valor global para cada herramienta que englobará todas las categorías establecidas.

Además, se introducen otras 2 variables para cada herramienta: el coste individual por licencia y las licencias en uso.

2.4.2 Diseñar la encuesta

En cuanto al diseño de la encuesta, se trata de una encuesta transversal, realizada on-line en un período de 2 meses.

Se establecen una serie de valores numéricos que se utilizan para la evaluación de los datos recogidos a través de la encuesta.

En primer lugar, para cada categoría (c) y cada herramienta (t) determinada se establece si se tiene en cuenta o no (si es un participante) siempre y cuando se cumpla el criterio que se muestra en la siguiente ecuación:

$$participante(t{:}\,c) = \begin{cases} verdadero{:}\; NA(t{:}\,c) \geq NQ(c)/2 \\ falso{:}\;\; en\ caso\ contrario \end{cases}$$

Es decir, que el número de respuestas dadas superaba el 50% de preguntas relativas a una determinada categoría.

Si se consideraba que una herramienta (t) participaba en los cálculos relativos a una determinada categoría (c), se calculaba su puntuación según la fórmula:

$$puntuación(t{:}\,c) = \frac{\sum_{q=1}^{NQ(c)} puntuación(t{:}\,q)}{NQ(c)}$$

Esta puntuación obtenía un valor en porcentaje sobre el grado de cumplimiento de cada herramienta en cada categoría establecida. Los valores de las puntuaciones obtenidas para cada herramienta en cada categoría se discretizaron según los siguientes intervalos:

$$discretizar(s) = \begin{cases} muy\ alto, & s \in (0{,}875, 1] \\ alto, & s \in (0{,}625, 0{,}875] \\ medio, & s \in (0{,}375, 0{,}625] \\ bajo, & s \in (0{,}125, 0{,}375] \\ muy\ bajo, & s \in (0, 0{,}125] \end{cases}$$

A su vez, para calcular la puntuación global de una herramienta en una determinada categoría se estableció la siguiente fórmula:

$$puntuación(c) = \frac{\sum_{t=1}^{NP(c)} puntuación(t{:}\,c)}{NP(c)}$$

El valor *NP*(c) corresponde al número de participantes en la categoría c.

Con el fin de validar las hipótesis del estudio, se estableció la veracidad de las mismas siempre que la puntuación obtenida superara al 70% de respuestas positivas en esa categoría.

Para cuantificar el precio en dólares de cada licencia (p) y el número de licencias (l) de cada herramienta se utilizaron las siguientes fórmulas:

$$cuantificar(p) = \begin{cases} 0{:}875: & p > 1000 \\ 0{:}625: & p \in (501{:}1000] \\ 0{:}375: & p \in [100{:}500] \\ 0{:}125: & p < 100 \\ \text{sin respuesta}: & p \text{ sin respuesta} \end{cases}$$

$$cuantificar(l) = \begin{cases} 0{:}875: & l > 10000 \\ 0{:}625: & l \in (1001{:}10000] \\ 0{:}375: & l \in (101{:}1000] \\ 0{:}125: & l < 100 \\ \text{sin respuesta}: & l \text{ sin respuesta} \end{cases}$$

2.4.3 Desarrollar el cuestionario

Ya se comentó la importancia de contextualizar la investigación que se está desarrollando a través de un estudio bibliográfico de aquellos temas relacionados con ella. En este artículo, este estudio se centra en tres puntos principales, estudiando los trabajos relevantes ya publicados sobre cada uno de ellos: herramientas existentes de la IR, marcos de trabajo para la comparación de herramientas de la IR y, finalmente, encuestas sobre herramientas de la IR.

El cuestionario de la encuesta contenía un total de 146 preguntas, 126 recogían información técnica de las funcionalidades de cada herramienta y las 20 restantes información básica de gestión como el nombre de la herramienta, el fabricante, la versión actual, etc. La gran mayoría de las preguntas era de respuesta cerrada, aunque se incorporaron algunas de respuesta abierta para que los participantes pudieran, por ejemplo, añadir alguna funcionalidad de la herramienta que no se hubiera recogido en la lista de preguntas. A modo de ejemplo, se presentan varias preguntas en la Tabla 2.3, agrupadas por categoría. Todas estas preguntas eran cerradas y admitían como posibles respuestas "Sí", "No", "No lo *sé*" o "*Sin respuesta*". *El cuestionario está disponible en http://www.um.es/giisw/EN/re-tools-survey/part1.pdf.*

Categoría	Grupo de capacidades	Pregunta
Elicitación	Captura de requisitos	¿Da la herramienta soporte a la captura de requisitos permitiendo que el usuario almacene y gestione la documentación de las entrevistas, talleres y otras observaciones?
Análisis	Análisis de atributos	¿Da la herramienta soporte al rastreo de cualquier atributo definido por el usuario o proporcionado por la herramienta a través de la detección y la marca de atributos ausentes o perdidos?
Especificación	Documentación de la especificación de requisitos	¿Da la herramienta soporte a la documentación de la especificación de requisitos manteniendo una relación de requisitos a través de los riesgos que hayan surgido y los que se hayan mitigado?
Modelado	Análisis del modelado	¿Da la herramienta soporte al análisis del modelado evaluando los requisitos basados en objetivos de negocio?
Verificación y validación	Verificación y validación	¿Da la herramienta soporte a la verificación y validación generando informes de excepción en los requisitos que no tengan casos de plan de verificación y casos de plan de verificación que no estén vinculados a los requisitos?
Gestión	Gestión de proyectos	¿Da la herramienta soporte a la gestión de proyectos registrando, monitorizando e informando sobre el estado del proceso general de gestión de requisitos?
Trazabilidad	Trazabilidad	¿Da la herramienta soporte a la trazabilidad a través del mantenimiento automático de trazas?
Otras capacidades	Interfaz gráfica de usuario	¿Da la herramienta soporte a interfaces gráficas de usuario permitiendo el uso de navegadores web?

Tabla 2.3. Ejemplos de preguntas

En el artículo se explicita el problema detectado al haber, en algunos casos, respuestas del estilo "no lo sé" o "sin respuesta". La solución que se adoptó consistía en tener en cuenta únicamente una categoría, si se habían respondido (bien positiva o negativamente) al menos al 50% de las preguntas relativas a esa categoría.

2.4.4 Evaluar y validar el cuestionario

Los autores elaboraron el cuestionario en un período de 3 meses, incluyendo 5 ciclos de revisión del mismo. Durante estas revisiones, se discutieron y acordaron la formulación de todas las preguntas hasta que se consiguió que todas fueran claras y simples.

Además, una persona del equipo de investigación se dedicó a comprobar que todos los aspectos técnicos relativos a la realización on-line de la encuesta funcionaban correctamente.

2.4.5 Obtener los datos de la encuesta

Para la gestión de los cuestionarios se utilizó la herramienta *LimeSurvey*[2]. Para reclutar a las personas que participarían en la encuestas contactaron mediante correo electrónico con los distintos desarrolladores y/o proveedores de 100 herramientas, para invitarles a participar en la encuesta. Tras un proceso de otros 2 meses se consiguió que 38 de ellos participaran aportando los datos sobre cada una de sus herramientas, debiendo reducir el conjunto original de herramientas candidatas de 100 a 94 porque no pudieron conseguir la dirección de correo electrónico de sus representantes.

2.4.6 Analizar los datos obtenidos

Para establecer correlaciones entre variables, los autores utilizaron el coeficiente de correlación de Pearson (Ott y Longnecker, 2010) y como herramientas de cálculo estadístico y procesamiento de datos utilizaron el paquete *SPSS 19.0* y la herramienta *MS-Office Excel 2007*.

Los distintos resultados obtenidos se desgranan en los siguientes apartados, comenzando con una descripción de los participantes a través de la información de gestión recogida en la encuestas, continuando con el análisis de correlación entre las variables definidas, el contraste de hipótesis y finalizando con el procedimiento de validación de datos.

---

[2] *www.limesurvey.org*

2.4.6.1 PARTICIPANTES

Finalmente se consiguieron examinar un total de 38 herramientas de las 94 candidatas, lo que supone una participación del 40,4%.

Con las preguntas que recogían información de gestión de las herramientas se realizó un análisis que se resume gráficamente en la Figura 2.1.

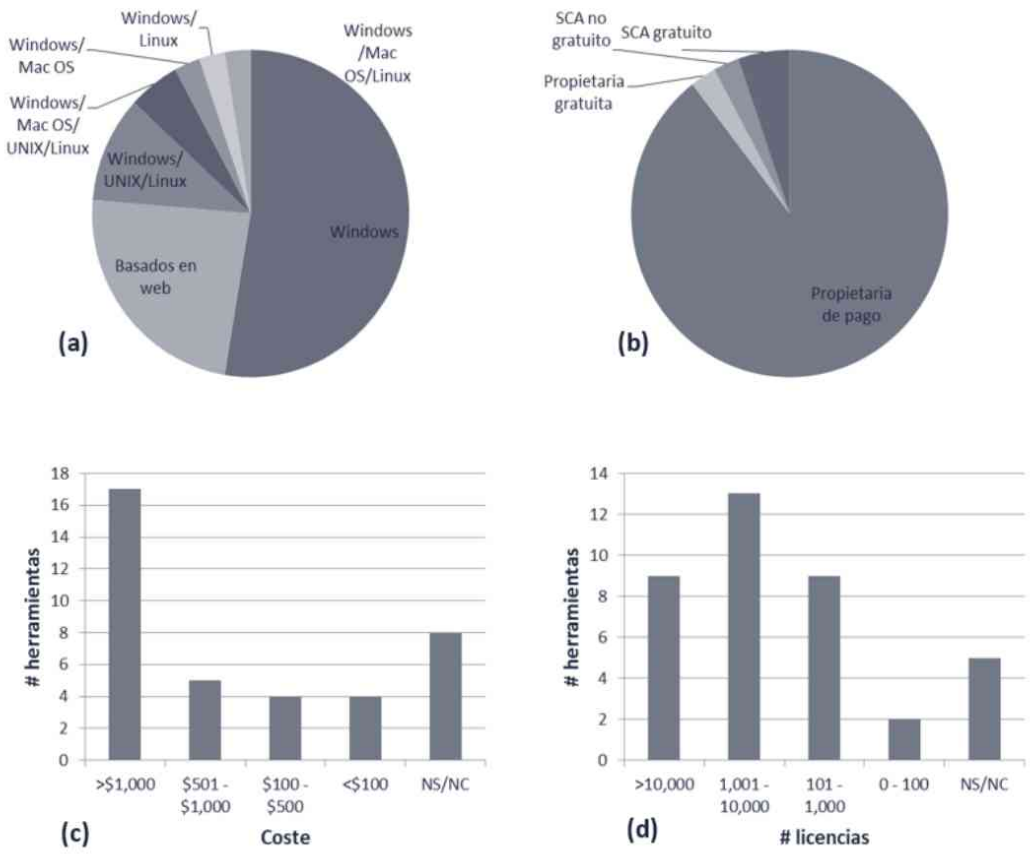


Figura 2.1. Datos relativos a las herramientas, Plataforma software requerida (*a*), *Tipo de licencias* (*b*), *Coste individual por licencia* (*c*) *y Licencias en uso* (*d*)

De la Figura 2.1 se puede extraer la siguiente información:

- Con respecto a la plataforma *software*, hay una gran predominancia de sistemas Windows como se puede observar en la Figura 2.1(a). Los clientes basados en la Web también son comunes, con el propósito de facilitar el acceso colaborativo a los recursos. Otros sistemas operativos como UNIX o Linux, y particularmente Mac, tienen una presencia más limitada.

- Las licencias eran mayoritariamente propietarias y no gratis (ver Figura 2.1 (b)), con poca influencia del *software* libre. Con respecto al promedio de coste por licencia (n dólares), la mayoría de las herramientas cuestan $1.000 o más como se puede observar en la Figura 2.1(c).
- Finalmente el número de licencias en uso es aproximadamente entre 1.001 y 10.000 (ver Figura 2.1 (d), aunque hay un número considerable de herramientas más extendidas (más de 10.000 licencias) y otro grupo importante de herramientas con menos representación en el mercado (entre 101 y 1.000 licencias).

2.4.6.2 CORRELACIONES ENTRE VARIABLES

Para medir la correlación entre las variables estudiadas se utilizó un test de correlación bivariado. En la Figura 2.2 se presentan los resultados obtenidos. En esta figura cada celda contiene dos valores, el valor de *r* que indica la fortaleza y la dirección (-+) de la correlación (son mejores los valores mayores), y "*" o "**" indica que la hipótesis nula H_0 se pude rechazar, lo que significa que las variables están correlacionadas. El segundo valor es el número de pares en la muestra.

En este tipo de análisis, un valor positivo elevado representa una correlación positiva igualmente elevada entre las 2 variables involucradas. A modo de ejemplo, mencionamos la correlación de la elicitación con el análisis (0,723) o entre la elicitación y la especificación (0,800), entre otras. Esto implica que los valores de las variables correlacionadas, aumentan o disminuyen simultánea y proporcionalmente.

Si analizamos el Coste, vemos que la mayor correlación la tiene con la especificación de requisitos, aunque si observamos la correlación con la puntuación global no es significativa con es positiva (0,358).

Finalmente, comentar la correlación entre el número de licencias en uso y el resto de variables. Se puede observar en Figura 2.2 una alta correlación positiva con la categoría de otras capacidades (0,513) Y por sin embargo, no hay correlación entre el número de licencias en uso y el coste medio por licencia (0,243).

Matriz de correlación: ELicitación; Análisis; eSPecificación; Modelado; V&V; Gestión; Trazabilidad; Otras; Global; Coste por licencia; Licencias en uso; N: tamaño de la muestra

	EL	AN	SP	MO	VV	GE	TR	OT	GL	CO	LI
EL	1										
N	35										
AN	0.763**	1									
N	34	35									
SP	0.800**	0.653**	1								
N	35	35	36								
MO	0.470**	0.499**	0.629**	1							
N	33	33	34	34							
VV	0.792**	0.740**	0.641**	0.521**	1						
N	27	27	27	25	29						
GE	0.729**	0.725**	0.716**	0.592**	0.694**	1					
N	31	31	32	31	28	34					
TR	0.776**	0.618**	0.781**	0.656**	0.791**	0.737**	1				
N	31	31	32	30	29	33	34				
OT	0.777**	0.610**	0.718**	0.628**	0.678**	0.801**	0.732**	1			
N	29	29	29	28	28	31	31	31			
GL	0.903**	0.799**	0.869**	0.799**	0.830**	0.934**	0.913**	0.872**	1		
N	25	25	25	25	25	25	25	25	25		
CO	0,269	0.336*	0.545**	0.404*	0,169	0,094	0.329*	0,147	0,358	1	
N	29	29	30	28	23	27	27	24	21	30	
LI	0,185	0,068	0,285	-0,006	-0,054	0,245	0,012	0.513**	0,183	0,243	1
N	31	31	32	30	25	30	30	27	22	29	33

* correlación significativa a nivel 0,05

** correlación significativa a nivel 0,01

Figura 2.2. Correlaciones entre variables

2.4.6.3 EVALUACIÓN DE LAS HIPÓTESIS

Para hacer el contraste de hipótesis se utilizó un análisis de los estadísticos descriptivos. La Figura 2.3 muestra las puntuaciones para cada una de las capacidades consideradas, obtenidas para las 38 herramientas cuyos fabricantes contestaron a la encuesta.

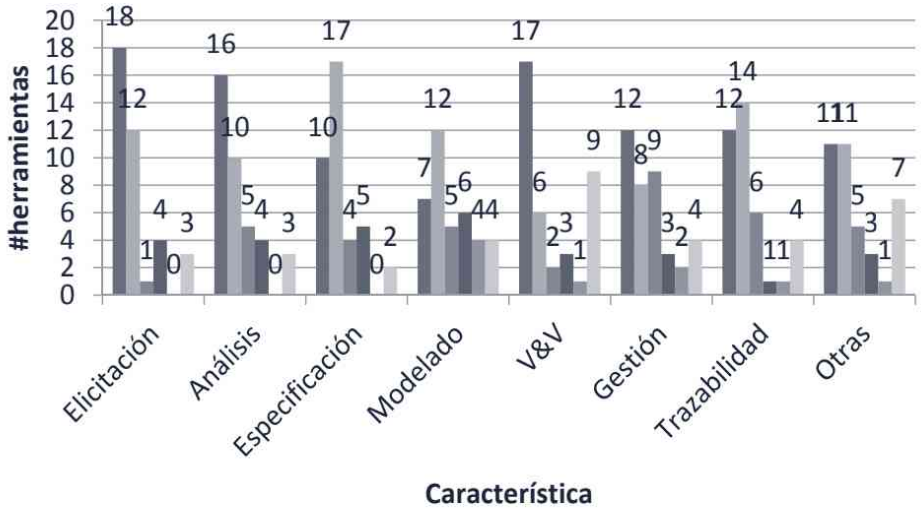


Figura 2.3. Puntuaciones obtenidas por las herramientas (*valores de izquierda a derecha, muy alta, alta, media, baja, muy baja, sin participantes*)

Como resumen de los resultados obtenidos en relación a las hipótesis planteadas, únicamente las hipótesis H_4 y H_6 (las relativas al modelado y la gestión de los requisitos) fueron rechazadas, por lo que el resto de hipótesis se aceptaron. Esto significa que las herramientas estudiadas sobre IR dan soporte adecuado al modelado y a la gestión de los requisitos, pero no así al resto de características estudiadas.

Por último, la Figura 2.4 muestra las puntuaciones globales obtenidas por las distintas herramientas, que muestra que las puntuaciones globales en general fueron altas (72% obtuvieron una puntuación alta o muy alta), lo que demuestra que el nivel de capacidades de las herramientas era alto, por lo que la hipótesis H_9 se puedo ser aceptadas. No obstante, los valores individuales obtenidos para cada una de las capacidades, como comentamos previamente no todos fueron favorables.

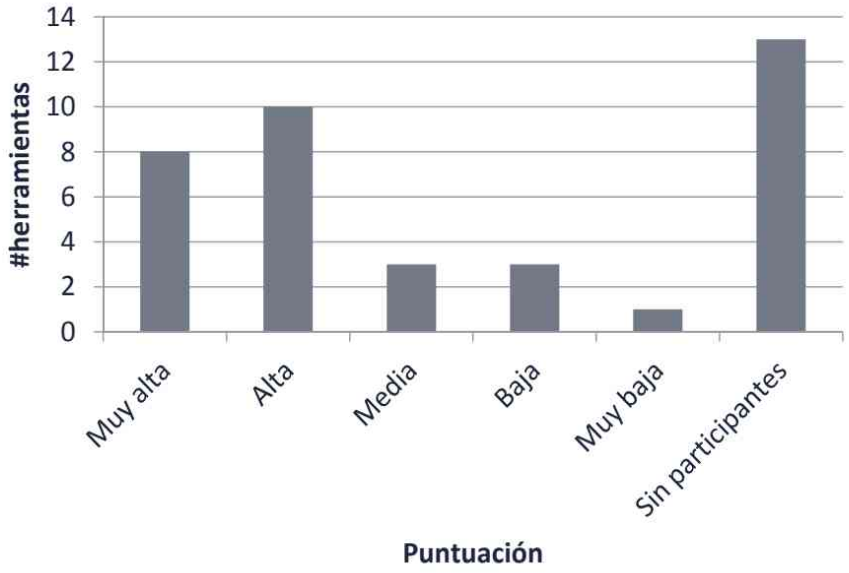


Figura 2.4. Puntuaciones globales de las herramientas

2.4.6.4 VALIDACIÓN DE LOS DATOS

Se presentan en este apartado las posibles anomalías que detectaron respecto de los datos. Principalmente exponen que la evaluación de cada herramienta se llevaba a cabo por sus fabricantes, lo que podía influir en un grado de subjetividad superior a lo normal con el fin de conseguir una puntuación positiva de la herramienta. Con el fin de mitigar este problema, los autores realizaron un estudio seleccionando de manera aleatoria cinco de las 38 herramientas para comprobar que las respuestas que habían recibido se ajustaban con la realidad. También se realizó un análisis denominado "fiabilidad entre

evaluadores" (*interrater reliability*) que permite con el propósito de determinar el grado de consistencia entre las puntuaciones y para medir la fiabilidad de los datos obtenidos a través de las repuestas de los encuestados. Tras hacer este análisis se llegó a la conclusión que las puntuaciones obtenidas eran consistentes y fiables.

2.4.7 Limitaciones del ejemplo

Como es habitual en cualquier estudio empírico se analizaron aquellos aspectos que podían amenazar la validez de los resultados obtenidos:

- **Validez interna**: Se refiere a la fiabilidad de la encuesta. La validez del material recolectado a través del cuestionario es altamente dependiente de la experiencia de los encuestados. La mayoría de los encuestados son profesionales con varios de experiencia en el uso de las herramientas de IR. El riesgo de maduración se tuvo en cuenta y por ello se diseñó un cuestionario que no requiriera más de 20 minutos para completarlo. A pesar de esto, hubo 3 encuestados que comenzaron a completar el cuestionario y lo abandonaron. Otro aspecto que se intentó mitigar es el sesgo que puede producirse por el hecho de que las herramientas fueran evaluadas por sus propios fabricantes, para ello se tuvo especial cuidado en el diseño del cuestionario y además se hizo un análisis de la fiabilidad entre evaluadores y se encontró un alto grado de acuerdo.
- **Validez externa**: Se refiere a la generalización de los resultados a entornos industriales. La selección de las herramientas se hizo en listas conocidas, evitando así considerar herramientas poco conocidas. Y los encuestados eran sus fabricantes, con lo que consideramos que tanto las herramientas como los encuestados se consideran adecuados. Sin embargo hay algunas herramientas relevantes, que no se pudieron incluir, como es el caso de Enterprise Architect, aunque se intentó en reiteradas ocasiones contactar con sus fabricantes. Además se consideraron capacidades de las herramientas escogidas de un marco específico diseñado por expertos para la evaluación de herramientas de IR y que se adecuo para considerar algunos aspectos específicos necesarios en la industria (por ejemplo, modelos de datos abiertos, integración de datos, etc.). Sin embargo, puede que nos sean las capacidades deseables por los usuarios de las herramientas. Los autores de este estudio creen que la evidencia obtenida se puede generalizar a entornos de IR industriales específicos y los resultados de este estudio pueden utilizarse por los ingenieros de requisitos, teniendo en cuenta que la tecnología evoluciona y que pueden surgir nuevas herramientas y que las herramientas existentes pueden incorporar nuevas capacidades.

- **Validez de la conclusiones**: El tamaño de la muestra (38 herramientas) es pequeño para producir un poder estadístico aceptable y es innegable que existan herramientas que no se consideraron. Por ello, no es recomendable considerar como definitivos los resultados obtenidos. Es aconsejable continuar este estudio con una mayor cantidad de herramientas. No obstante, el número de participantes representa un porcentaje relevante de la comunidad de fabricantes de herramientas de IR, incluyendo herramientas de empresas establecidas en tres continentes (Asia, América y Europa). Además el estudio realizó siguiendo un proceso y se publicó con un alto nivel de detalle para permitir que el proceso sea reproducible. Aunque puede ocurrir que en el futuro que el número de herramientas obtenido en las búsquedas varíe en el futuro.

2.4.8 Conclusiones del ejemplo

En el ejemplo presentado se realizó un estudio bibliográfico sobre las herramientas de soporte a la IR. También se creó un marco de clasificación, basado en el informe técnico ISO/IEC TR 24766 para establecer las características deseables de estas herramientas. Además se llevó a cabo una encuesta con el objetivo de analizar hasta qué punto las herramientas existentes para la IR cubren las capacidades deseables. Los hallazgos obtenidos en esta encuesta pueden ser útiles para investigadores porque les proporcionan una idea del estado del arte sobre herramientas de IR y también para los profesionales, ya que les ayudará a ser conscientes de las características más frecuentes que contienen las herramientas de IR existentes y a conocer su coste, el número de licencias en uso, etc. Esta información puede ser de gran interés al momento de decidir qué herramienta IR se debe utilizar en sus organizaciones.

Entre otros, los principales hallazgos obtenidos son:

- Respecto de los resultados obtenidos en el análisis de correlación, demuestran que hay una fuerte correlación entre muchas de las características deseables en las herramientas, lo que lleva a pensar que si una herramienta se comporta de manera deseable con respecto a una de las características, con mucha probabilidad lo hará también con el resto.
- La característica que mejor se trata, en general, es la elicitación de requisitos y en concreto la captura de requisitos, aunque existe deficiencia en el soporte de la herramientas a las plantillas y lista de comprobación utilizadas en la elicitación de requisitos. El análisis, la especificación, la trazabilidad, la V&V de los requisitos también se tratan adecuadamente.

- Por último, indicar que las herramientas estudiadas dan peor soporte a los aspectos relacionados con el modelado, la gestión de requisitos y otras capacidades. Esto indica que estas capacidades deberían ser mejoradas en las herramientas de requisitos.
- Las puntaciones obtenidas para el modelado fueron menores que las obtenidas en la especificación y otras categorías. Esto se puede deber a que las herramientas de requisitos han estado tradicionalmente más orientadas a requisitos textuales, en lenguaje natural en comparación con notaciones de modelado como BPMN, UML, E/R.
- Con respecto a la categoría de gestión de requisitos, se detectó la falta de mecanismos para modelo de datos abierto. Al parecer, las herramientas de IR actuales no dan soporte tales características, a pesar de que le proporcionan tanto a los desarrolladores como los usuarios muchos beneficios importantes: apoyo para aumentar la comunicación y la automatización, una amplia personalización para el usuario final, secuencias de comandos y funciones de macro, agentes inteligentes externos y tutores, comandos potentes para buscar y reemplazar, buscar y reemplazar comandos, fácil suministro de correctores ortográficos conocidos, marcas semánticas, interfaces alternativas sin reimplementación, la capacidad de tener los *plug-ins* que operan en el mismo espacio, y una significativamente mayor reutilización de código común para los implementadores. Algunos de los participantes que han mencionado la posibilidad de dar soporte a modelos de datos abiertos han señalado que se puede hacer por medio de APIs específicas.
- A veces, sobre todo en las grandes organizaciones, existen diversas fuentes de datos que contienen datos críticos de la empresa. Por otra parte, la gestión de estos datos dispersos depende de sistemas diferentes. Esta diversidad de fuentes de datos es causada por muchos factores que se encuentran típicamente en proyectos de Desarrollo Global de *Software*, como la falta de coordinación entre las diferentes partes de la organización, las diferentes tasas de adopción de tecnologías nuevas, fusiones y adquisiciones, y la distancia geográfica entre los grupos de colaboradores. Por lo tanto, las herramientas de IR deben ofrecer mecanismos con los que combinar la información de estos sistemas diversos, sobre todo en ambientes de trabajo distribuidos.

2.5 OTROS EJEMPLOS DE ENCUESTAS

Entre otros ejemplos de encuestas que pueden encontrarse en la literatura, caben destacar los siguientes: Grossman *et al.* (2005), Conradi *et al* (2005), Jørgensen y Moløkken-Østvold (2006), Rodríguez *et al.* (2012) y Vizcaíno *et al.* (2013).

2.6 LECTURAS RECOMENDADAS

- **Fink**, **A**. (2003). *The survey kit*. *SAGE Publications*. Se trata de una colección de libros que pretende ser una enciclopedia sobre la realización de encuestas en general. Resulta interesante al estar dividido en distintos libros específicos para distintos aspectos relativos a las encuestas, si bien no se enmarca específicamente en el campo de la ingeniería del *software*.

- **Ciolkowski**, **M**., **Laitenberger**, **O**., **Vegas**, **S**. y **Biffl**, **S**. (2003). *Practical experiences in the design and conduct of surveys in empirical software engineering*. ESERNET 2001-2003, LNCS 2765, pp. 104-128. En este trabajo se presenta un proceso para preparar, realizar y analizar una encuesta, basada en cuestionarios, específicamente en campo de la ingeniería del *software* con el fin de proporcionar a los investigadores del área una aproximación sistemática y disciplinada a la hora de realizar encuestas y detectar y evitar obstáculos a la hora de realizar encuestas y conseguir resultados interesantes.

- **Kitchenham**, **B**. **A**. y **Pfleeger**, **S**. **L**. (2008). *Personal opinion surveys*. *Chapter* 3 in: Shull, F., Singer, J., Sjøberg, D.I.K. (eds.) *Guide to Advanced Empirical Software Engineering*: *Springer*. Capítulo de referencia en la actualidad en la comunidad acerca del diseño y realización de encuestas. Este capítulo tiene como origen la serie de artículos que los autores publicaron en la revista *Software Engineering Notes* entre 2001 y 2003, llamados *Principles of Survey Research*.

2.7 HERRAMIENTAS Y SITIOS WEB RECOMENDADOS

Existen multitud de herramientas y páginas web que permiten realizar encuestas *online*. Entre las más utilizadas y que ofrecen sus servicios de forma gratuita, cabe destacar:

- *http://www.limesurvey.org/*
- *https://www.murvey.com/*
- *http://kwiksurveys.com/*
- *http://freeonlinesurveys.com/*

Algunas otras herramientas ofrecen funcionalidades limitadas de manera gratuita, pero es necesario adquirir una licencia para acceder a todas las posibilidades que ofrecen. Entre ellas destacan:

- *http://es.surveymonkey.com*
- *http://www.zoomerang.com/*
- *http://www.surveygizmo.com/*

Por último, indicar que cualquier paquete estadístico que se utilice normalmente (R, SPSS, etc.) será más que suficiente para realizar el análisis de los datos obtenidos durante la realización de una encuesta.

Capítulo 3

EXPERIMENTOS

3.1 CARACTERÍSTICAS DE LOS EXPERIMENTOS

Un experimento en ingeniería del *software* es una investigación empírica que manipula una variable (denominada independiente o factor) del entorno o fenómeno estudiado midiendo el efecto que tiene sobre otra variable denominada variable dependiente. Existen dos tipos de experimentos, los controlados en los que los tratamientos se asignan a los sujetos de manera aleatoria; y los denominados cuasi experimentos, cuando esta aleatorización no es posible. De aquí en adelante hablaremos en términos genéricos de experimentos, refiriéndonos indistintamente a ambos tipos de experimentos (controlados y cuasi).

Los experimentos pueden ser "orientados a las personas" (*human-oriented experiments*) u "orientados a la tecnología" (*technology-oriented experiments*). En los primeros los sujetos aplican diferentes tratamientos a los objetos; por ejemplo, dos métodos de inspección se aplican sobre dos códigos fuente. En los "orientados a la tecnología", generalmente se aplican diferentes herramientas a diferentes objetos; por ejemplo al mismo programa se le aplican dos técnicas de generación de casos de prueba diferentes.

Podremos utilizar los experimentos para confirmar el conocimiento convencional, explorar relaciones entre sucesos, evaluar la precisión de modelos, validar métricas, etc. En todas estas aplicaciones de los experimentos lo que se persigue es probar hipótesis, ya que si los resultados de un experimento contradicen una hipótesis, dichas hipótesis pueden rechazarse.

La principal fortaleza de los experimentos es que a través de ellos se puede investigar en qué situaciones son ciertas las afirmaciones y pueden servir para recomendar en qué contextos son útiles ciertos estándares, métodos y herramientas. Esto tiene relación con la contextualización de la evidencia empírica mencionada en el capítulo 1.

Para llevar a cabo un experimento, es necesario seguir un proceso experimental en el que se detallen las actividades a realizar, qué debe hacerse y cuáles son las entradas y salidas de cada actividad. El proceso experimental que se presenta a continuación se centra en experimentos, pero las mismas actividades básicas deben realizarse en cualquier tipo de estudio empírico. La principal diferencia es cómo se realizan cada una de las tareas de cada actividad, por ejemplo, el diseño de un experimento, una encuesta o un estudio de caso es diferente, pero todos deben diseñarse. Esto significa que las actividades básicas son las mismas, pero cada actividad deberá adaptase según el objetivo específico de cada tipo de estudio.

Este proceso no se lleva a cabo en cascada, ya que no es necesario terminar una actividad para poder comenzar la siguiente, de hecho, el orden de las actividades en el proceso sólo indica el orden de comienzo de cada actividad. El proceso es iterativo, ya que en ciertos casos es necesario volver atrás para refinar una actividad previa antes de continuar con el experimento. Obviamente, no se podrá volver atrás para refinar el objetivo o la planificación una vez comenzada la ejecución del experimento.

El proceso experimental propuesto en Wohlin *et al.* (2012), consta de cinco actividades cada una con sus correspondientes tareas, como se puede ver en la Figura 3.1.

3.2 PROCESO EXPERIMENTAL

El punto partida de todo experimento es saber que realmente el experimento es el método apropiado para evaluar el fenómeno en el que estamos interesados, es decir, se ha de estar convencido de que un experimento es la manera apropiada para responder la pregunta que se está investigando.

A continuación se describen las tareas que componen las cinco actividades del proceso experimental.

Figura 3.1. Visión global del proceso experimental (*Wohlin et al., 2012*)

3.2.1 Definición del alcance

En esta actividad se han de definir los objetivos del experimento. Para definir correctamente todos los aspectos importantes del experimento, antes de pasar a la planificación y posterior ejecución, es aconsejable utilizar la plantilla GQM (*Goal-Question-Metric*) para definición de objetivos (Basili y Rombach, 1988). Esta plantilla es la siguiente:

- **Analizar** <*Objeto*(*s*) *de estudio*> -¿qué es lo que se estudia?
- **con el propósito de** <*Propósito*> - ¿qué intención tiene el estudio?
- **con respecto a** <*Aspecto de calidad*> - ¿qué efecto se estudia?
- **desde el punto de vista de** <*Perspectiva*> - ¿quién se ve afectado?
- **en el contexto de** <*Contexto*> - ¿dónde, cómo y por quién se lleva a cabo el estudio?

Así, por ejemplo, podemos encontrar la definición del objetivo de un experimento como: "Analizar un método orientado a objetos y otro estructurado con el propósito de evaluar, con respecto a la productividad desde el punto de vista de los investigadores en el contexto de estudiantes de grado y postgrado de una determinada universidad desarrollando un sistema *software*".

La Tabla 3.1 muestra algunos ejemplos de los elementos que constituyen la plantilla para la definición de objetivos.

Objeto de estudio	Propósito	Aspecto de calidad	Perspectiva	Contexto
Producto	Caracterizar	Efectividad	Desarrollador	Sujetos
Proceso	Supervisar	Costo	Mantenedor	Objetos
Modelo	Evaluar	Fiabilidad	Director de proyectos	
Métrica	Predecir	Mantenibilidad	Cliente	
Teoría	Controlar	Portabilidad	Usuario	
	Cambiar		Investigador	

Tabla 3.1. Ejemplos de los elementos de la plantilla para la definición de objetivos

3.2.2 Planificación

Tras definir los objetivos del experimento, se realiza la planificación del mismo para tener una noción clara de cómo se va a llevar a cabo. La planificación se divide, a su vez, en seis tareas, que se comentan a continuación.

3.2.2.1 SELECCIÓN DE CONTEXTO

El contexto determina el entorno en el que se ejecutará el experimento y puede caracterizarse según cuatro dimensiones: 1) *off-line* vs. *on-line*, según se realice en proyectos que están siendo ejecutados actualmente o no; 2) estudiantes vs. profesionales, 3) problemas "de juguete" vs. proyectos reales; 4) específico vs. general, según se pretenda conseguir resultados válidos para un contexto específico o para un propósito general. La elección tomada en cada dimensión determinará el contexto en el cual son aplicables los resultados obtenidos en el experimento y también la posibilidad de generalización de los mismos.

Con el objetivo de obtener resultados más generales, los experimentos deberían ser ejecutados en proyectos reales llevados a cabo por profesionales. Aunque esta puede ser la situación ideal, en algunos casos puede ser arriesgada. Si queremos comparar tecnologías emergentes versus tecnologías tradicionales, es aconsejable realizar un experimento en un entorno académico con estudiantes y una vez se tenga una primera evidencia de que la tecnología emergente es mejor, ir más allá y replicar el experimento en entornos industriales. Höst *et al.* (2000) y Kitchenham *et al.* (2002) justifican y explican que en ciertas circunstancias es aconsejable realizar experimentos con estudiantes. Si estamos interesados en evaluar el uso de tecnologías por ingenieros de *software* no expertos o principiantes los experimentos con estudiantes son válidos. Además hay que tener en cuenta que en ciertas tecnologías emergentes los alumnos de último curso suelen tener más conocimiento que los profesionales.

3.2.2.2 FORMULACIÓN DE HIPÓTESIS

El objetivo de un experimento puede expresarse como una hipótesis a probar. Una hipótesis es una teoría provisional o una suposición que se cree que explica el comportamiento que se pretende explorar.

Normalmente, las hipótesis que se quieren rechazar a través del experimento se especifican como hipótesis nulas (H_0). Las hipótesis alternativas (H_1) se plantean en el caso de que se rechacen las hipótesis nulas. Para probar las

hipótesis existen numerosos test estadísticos que se presentan en el apartado 3.2.4. La prueba de hipótesis implica diferentes tipos de errores: el test estadístico puede rechazar una hipótesis cierta siendo falsa (error de Tipo I) o no rechazar una hipótesis siendo falsa (error de Tipo II). El tamaño del error depende de varios factores, que deben tenerse en cuenta al planificar el experimento. Por ejemplo, la habilidad del test estadístico de revelar un patrón cierto, siendo H_0 falsa, en los datos recolectados en el experimento, lo que se conoce como potencia del test.

3.2.2.3 SELECCIÓN DE VARIABLES

En un experimento podemos encontrar diversos tipos de variables:

- **Variables independientes** (*factor*, *state*, *predictor*): variables cuyos valores se cambian para estudiar qué efecto producen esos cambios. Se denomina tratamiento a cada uno de los posibles valores (también denominados niveles) de la variable independiente.
- **Variables dependientes** (*response*): variables que se estudian para comprobar el efecto de los cambios en las variables independientes.
- **Variables controladas** (*controlled*): variables independientes controladas en un nivel fijo.
- **Variables enmascaradas** (*confounded*): variables no controladas que varían simultáneamente con las variables independientes.
- **Variables aleatorias** (*randomized*): variables no controladas que han de tratarse como un error aleatorio.

Una vez definido el objetivo según la plantilla GQM, debemos elegir las variables que vamos a manipular (variables independientes) y que vamos a medir (variables dependientes) en el experimento. Para vincular el objetivo con las hipótesis, los parámetros de la plantilla GQM se relacionan con las variables que intervienen en las hipótesis. Así pues, los factores que caracterizan el <objeto de estudio> constituyen las variables independientes, mientras que las variables dependientes miden el <aspecto de calidad>.

La selección de las variables requiere también conocer su escala de medición y el rango de valores que pueden tomar. Además de seleccionar las variables independientes y dependientes, habrá que tratar de minimizar el efecto del resto de variables presentadas.

3.2.2.4 SELECCIÓN DE SUJETOS

Un sujeto es aquella persona que aplica los tratamientos del experimento. La selección de sujetos (o muestreo) tiene un efecto muy importante en los resultados experimentales, por lo que debe considerarse cuidadosamente. Para poder generalizar los hallazgos del estudio, la población de sujetos debe ser representativa. El muestreo se puede realizar usando técnicas probabilísticas (muestreo aleatorio o sistemático) o no probabilísticas (muestreo por conveniencia o por cuota).

Además de la representatividad, el tamaño de la muestra también influye, ya que cuanto mayor sea, menor será la probabilidad de cometer un error al generalizar los resultados. El tamaño de la muestra también tiene influencia sobre la selección del test estadístico que se va a utilizar para analizar los datos, por ello es necesario considerar durante la planificación cómo se van a analizar los datos.

3.2.2.5 ELECCIÓN DEL DISEÑO

Éste es un paso crucial, ya que un mal diseño puede invalidar cualquier estudio bien intencionado. Además, un diseño apropiado es la base que permite la realización de réplicas correctas.

En la relación causa-efecto, que la variable independiente ejerce sobre la dependiente pueden mediar otras fuentes extrañas de variación que se deben tener en cuenta al diseñar un experimento. Algunas de estas fuentes de error se deben a la variabilidad que existe entre los participantes, la diferencia de concentración debido a ruidos molestos que los sujetos de un aula pueden sufrir con respecto a los que participan en otra clase, los fallos técnicos en el funcionamiento de los ordenadores, etc. Como, en realidad, la eliminación completa no es posible, hay que intentar disminuir esta variabilidad controlando tantas variables como sea posible. Mediante la aplicación de técnicas de control, tales como asignación aleatoria, bloqueo y equilibrado, se logra minimizar el error experimental. Estas técnicas se detallan a continuación:

- **Aleatorización**: Este principio se refiere a la asignación de manera aleatoria de los tratamientos a los sujetos, y también a la selección de los sujetos de manera aleatoria dentro de una muestra representativa.

- **Bloqueo**: Algunas veces tenemos un factor que probablemente tiene un efecto sobre la variable dependiente, pero no estamos interesados en este efecto. Si el efecto del factor es conocido y controlable, podemos utilizar el bloqueo para incrementar así la precisión del experimento. El bloqueo se utiliza para eliminar el efecto no deseado de ese factor en la comparación de los tratamientos. Dentro de un bloque, el efecto no deseado es el mismo y podemos estudiar el efecto de los tratamientos dentro de cada bloque, pero no entre bloques. Por ejemplo, podríamos considerar como ejemplo de este tipo de factor a la experiencia. Si tenemos sujetos con mucha experiencia en un tratamiento y otros que no. Para minimizar el efecto de la experiencia, los sujetos se deberían dividir en dos grupos, uno con experiencia y otro sin experiencia. Otra alternativa es formar grupos equilibrados con respecto a la experiencia.
- **Balanceo o equilibrado**: Este principio se cumple cuando cada tratamiento se asigna al mismo número de sujetos. El balanceo es deseable, porque simplifica y favorece los análisis estadísticos, aunque no es necesario.

Pueden encontrarse distintos tipos diseños experimentales en función del propósito del experimento, el número de variables independientes, el número de tratamientos, la cantidad de tratamientos que se asignen a cada sujeto, etc. La Figura 3.2. presenta una taxonomía con la que se clasifican los experimentos en ingeniería del *software* (Otero y Dolado, 2000), en la que se consideran dos criterios para clasificar los tipos de diseño. El primer criterio consiste en considerar el número de variables independientes que se van a manipular, con lo que se diferencia entre diseño simple (una única variable independiente) y diseño complejo o factorial (dos o más variables independientes). En un diseño simple se tendrán tantos tratamientos como niveles se tengan de la variable independiente y en un diseño complejo o factorial se tendrán k^n tratamientos, siendo k el número de variables independientes y n el número de niveles de cada una de ellas. El segundo criterio es el número de tratamientos que se asignan a cada sujeto. Es decir cuando se asigna un sólo tratamiento por sujeto se denomina diseño inter-sujetos (IES) y en el diseño intra-sujetos (IAS) los sujetos reciben todos los tratamientos.

Evidentemente, cada tipo de diseño tendrá sus pros y sus contras y por ello su elección se debe analizar, justificar y documentar adecuadamente en la planificación.

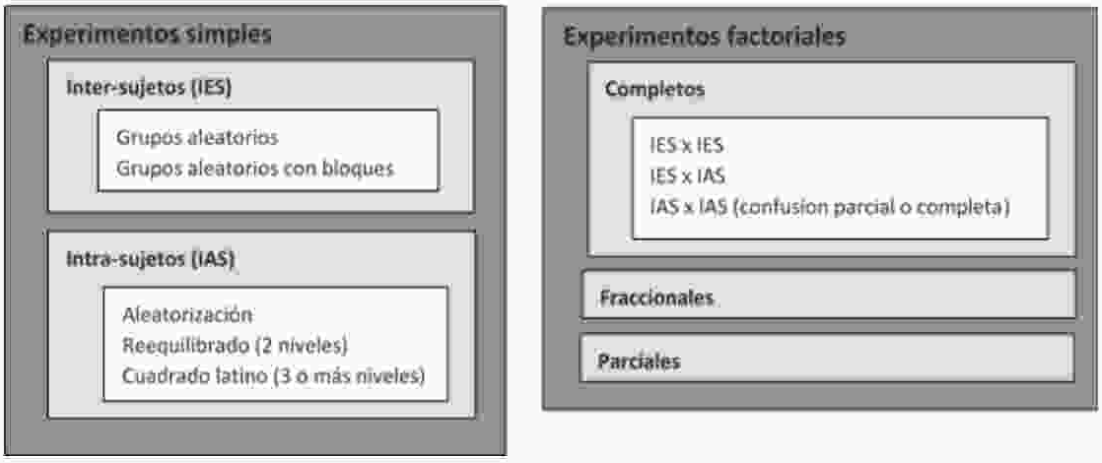


Figura 3.2. Tipos de diseño experimental (*Otero y Dolado, 2000*)

En Otero y Dolado (2000) se puede encontrar una explicación detallada de la taxonomía presentada en la Figura 3.2 con referencias a artículos que presentan ejemplos de experimentos de cada tipo. También en Wholin *et al.* (2012), Montgomery (2000) y en Juristo y Moreno (2001) se presenta una explicación detallada ilustrada con ejemplos sobre los diferentes tipos de diseños experimentales.

A modo de ejemplo, a continuación, explicaremos un tipo de diseño que suele ser el más utilizado en ingeniería del *software*. Supongamos que queremos comparar la productividad de los sujetos cuando programan en Java o en C++. La variable independiente es el lenguaje de programación, con dos tratamientos (Java, C++), y la variable dependiente la productividad (medida como líneas de código/hora). Lo ideal sería que cada sujeto use ambos lenguajes (diseño intra-sujetos), aunque si la aplicación a desarrollar lleva mucho tiempo puede producirse el efecto fatiga, lo que podría producir un efecto negativo en la productividad de los sujetos. Y además habría que considerar la realización de dos aplicaciones distintas, es decir considerar otra variable independiente con dos tratamientos (Aplicación A y Aplicación B), porque si un sujeto recibe la misma aplicación para desarrollar en ambos lenguajes, se produciría claramente un efecto de aprendizaje. Por ello se deben considerar dos aplicaciones, pero con similar complejidad y tamaño, para que no sea la aplicación en si la que influye en la productividad, sino el uso de un lenguaje u otro. Además es recomendable, entregar los tratamientos a los sujetos cambiando el orden. Se podrían formar cuatro grupos balanceados de sujetos, y el experimento se realizaría en dos ejecuciones, como muestra la Tabla 3.2. Este tipo de diseño corresponde al diseño factorial 2^2 completo IAS-IAS con confusión parcial, ya que cada sujeto recibirá dos tratamientos (y no los cuatro del diseño factorial 2^2) uno correspondiente al desarrollo en Java y otro a C++ pero cada uno correspondiente a una aplicación distinta.

		Aplicación	
		A	B
Lenguaje	Java	G1	G2
	C++	G3	G4

(Ejecución 1)

		Aplicación	
		A	B
Lenguaje	Java	G4	G3
	C++	G2	G1

(Ejecución 2)

Tabla 3.2. Ejemplo de un diseño factorial completo: IAS-IAS con confusión parcial

Por restricciones de tiempo, puede justificarse elegir un diseño inter-sujetos, es decir que un grupo de sujetos use Java y otro grupo C++. En este tipo diseño hay que tener mucho cuidado que los sujetos tengan la misma experiencia en Java y C++, porque si no la experiencia podría ser un factor que sesgue los resultados obtenidos. Lo que se suele hacer es realizar un test antes de la ejecución del experimento donde se recoge la experiencia de los sujetos, para formar grupos equilibrados en cuanto al factor experiencia (diseño simple inter-sujetos con reequilibrado). Si la población es muy grande y no hay posibilidad de hacer un test previo, se pueden formar grupos aleatorios (diseño simple inter-sujetos con aleatorización).

En el apartado 3.3 se muestra un experimento cuyo diseño es un diseño simple inter-sujetos reequilibrado, y el apartado 3.7 diseño factorial 2^2 completo IAS-IAS con confusión parcial, que suelen ser los más utilizados en la ingeniería del *software*.

3.2.2.6 INSTRUMENTACIÓN

El objetivo de la instrumentación es dotar de medios para realizar el experimento y para su seguimiento sin afectar el control del experimento. Los resultados de un experimento deberían ser los mismos independientemente de cómo se instrumente. Si los instrumentos afectan a los resultados del experimento, el experimento no será válido.

Hay tres tipos de instrumentos en un experimento: objetos experimentales, guías e instrumentos de medición, que deben elegirse y desarrollarse adecuadamente para cada experimento específico antes de ejecutarlo. Los objetos experimentales pueden ser, especificaciones de requisitos, diagramas de diseño, documentos con código, etc. Las guías sirven para guiar a los sujetos en la

realización del experimento y pueden ser, entre otras, descripciones de procesos y listas de comprobación. Los instrumentos de medición se utilizarán para recoger los datos y pueden ser formularios, entrevistas, etc.

3.2.2.7 EVALUACIÓN DE LA VALIDEZ

Un tema fundamental relativo a los resultados de un experimento, es cómo de válidos son los resultados encontrados. La evaluación de la validez no se puede dejar para el final, ya a la hora de planificar un experimento hay que analizar las posibles amenazas que se pueden producir e intentar palearlas en la manera de lo posible. Los tipos de amenazas a la validez son (Campbell y Cook, 1979):

- **Validez interna**: Define el grado de confianza en una relación causa-efecto entre los factores de interés y los resultados observados, es decir, el grado con el que pueden extraerse conclusiones en la relación causa-efecto (variables independientes-dependientes). Las amenazas a la validez interna comprenden temas que pueden indicar una relación casual aunque realmente no haya ninguna. Algunos factores que influyen en la validez interna son cómo se seleccionan y se agrupan los sujetos, como se les trata durante el experimento, si ocurre algún evento inesperado durante la realización, cómo son los materiales utilizados, si los sujetos abandonan la ejecución del experimento o se cansan al realizarlo, etc.
- **Validez externa**: Representa el grado hasta el que los resultados alcanzados pueden generalizarse teniendo en cuenta la población utilizada y otros parámetros de la investigación. Cuanto mayor sea, más se pueden generalizar los resultados a la práctica real en la ingeniería del *software*. Las amenazas a esta validez incluyen, principalmente, la posibilidad de generalizar los resultados experimentales fuera del contexto del experimento. La validez externa no sólo se ve afectada por el diseño experimental elegido, sino también por los objetos y los sujetos experimentales. Destacan tres riesgos principales: no contar con los sujetos adecuados como participantes, realizar el experimento en un entorno equivocado y realizarlo con una temporalización que afecte a los resultados.
- **Validez de constructo**: Define hasta dónde las variables miden correctamente los constructos teóricos de las hipótesis. Una amenaza a esta validez es la ausencia de pruebas teóricas que afirmen que las variables dependientes o las independientes realmente miden aquellos

conceptos que pretenden medir. Por ejemplo, el número de cursos que ha recibido un alumno sobre una determinada tecnología, es una medida muy poco significativa de la experiencia de los alumnos. Quizá sea más apropiado medir la experiencia como el número de años de experiencia en el uso práctico de dicha tecnología. Y de esta forma se mejora la validez de constructo.

- **Validez de la conclusión**: Define hasta dónde las conclusiones son estadísticamente válidas, es decir cuan correcta es la conclusión entre la relación entre el tratamiento y la variable dependiente. Algunas amenazas a la validez de las conclusiones pueden ser: bajo poder estadístico, violar las suposiciones de los test estadísticos, perseguir o "pescar" determinado resultado, falta de fiabilidad de las medidas, etc.

En muchos casos, es difícil paliar todas las amenazas, lo importante es describir explícitamente cuáles y cómo se han paliado y cuáles no y porqué. Además puede ocurrir que no todas las amenazas sean igual de importantes, por lo que puede ser necesario priorizarlas, ya que al intentar paliar una, se pueda perjudicar a otra.

En Wohlin *et al.* (2012), se presenta una lista de posibles amenazas dentro de cada tipo de amenazas. Neto y Conte (2013) han realizado un análisis sobre los diferentes tipos de amenazas considerados en la literatura y proponen una lista de acciones para intentar controlarlas.

3.2.3 Operación

Una vez se ha diseñado y planificado un experimento, debe llevarse a cabo para recoger los datos que se han de analizar posteriormente. Esto es lo que se denomina operación de un experimento.

Durante esta actividad se llevan a cabo tres tareas que se detallan a continuación.

3.2.3.1 PREPARACIÓN

Antes de comenzar el experimento, es necesario contar con personas que deseen formar parte de él como sujetos. Es imprescindible que estén motivados para participar durante toda la realización del experimento para conseguir resultados fiables. Se deben tener en cuenta algunos aspectos éticos como:

1) Contar con el consentimiento de los sujetos para participar en el experimento.

2) El rendimiento de cada sujeto en la ejecución del experimento debe ser confidencial.

3) Ofrecer algún incentivo a los sujetos por la participación en el experimento.

4) Desvelar todos los detalles del experimento a los sujetos, siempre que estos no puedan sesgar los resultados. En este último caso, los detalles del experimento se pueden desvelar al final de la ejecución del mismo.

Es fundamental que durante la preparación, se entrene a los sujetos seleccionados para realizar el experimento tanto en el tema objeto de estudio, como en las tareas que deberán realizar en el experimento. Es esencial que los sujetos entiendan claramente todo el material experimental y todas las tareas que deberán realizar. En estas sesiones de entrenamiento se les puede pedir a los sujetos que rellenen un cuestionario sobre datos personales, experiencia, y preguntas específicas para evaluar su conocimiento. Los resultados obtenidos en este cuestionario se suelen utilizar para formar grupos de sujetos balanceados según la experiencia y conocimiento.

También es necesario que todos los instrumentos experimentales estén preparados, lo que incluye los objetos, las guías y los formularios y herramientas de medición. Para asegurarnos que todos los instrumentos experimentales son correctos y están listos para ejecutar el experimento, es aconsejable hacer una ejecución del experimento de prueba (comúnmente llamado estudio piloto) con unos pocos sujetos (que luego no realizarán el experimento).

3.2.3.2 EJECUCIÓN

Para ejecutar un experimento lo ideal, aunque no siempre posible, es reunir a todos los sujetos, para que todos realicen el experimento en el mismo lugar. Esto proporciona diversas ventajas, como que los datos son más sencillos de recoger y que la persona que supervisa el experimento se encuentra presente durante la realización del experimento para solventar cualquier duda que le pueda surgir a los sujetos que participan en el experimento.

Antes de ejecutar el experimento, se suelen agrupar los sujetos y se les explica detalladamente todas las tareas que deben realizar y en qué orden. Si no es posible reunir a todos los sujetos para la ejecución del experimento, esta información deberá proporcionarse a los sujetos vía correo electrónico o a través de una página web.

La recolección de datos se podrá realizar de un modo totalmente manual con papel y lápiz, hasta un modo totalmente automatizado con herramientas.

3.2.3.3 VALIDACIÓN DE LOS DATOS

Una vez se han recogido los datos, el experimentador ha de comprobar que son razonables y que se han recogido correctamente.

Durante la ejecución del experimento se pueden detectar sujetos que no se comportan adecuadamente y se debe tomar nota de esto, porque sus resultados pueden ser anómalos y a la hora de validar los datos recolectados, posiblemente habrá que descartarlos.

3.2.4 Análisis e Interpretación

Después de recoger los datos obtenidos tras la realización del experimento, se deben analizar e interpretar correctamente. Hay tres aspectos principales a la hora de elegir entre las distintas técnicas de análisis: la naturaleza de los datos recogidos, el motivo de los experimentos y el tipo de diseño experimental. Es fundamental conocer por ejemplo la escala de medición de los datos y la distribución de los mismos para encontrar el test estadístico apropiado.

La interpretación cuantitativa se puede realizar llevar a cabo realizando las siguientes tareas: estadísticos descriptivos, reducción de datos (exclusión de valores atípicos) y contraste de hipótesis. Los estadísticos descriptivos, como la media, la desviación estándar, el máximo, el mínimo, etc., se utilizan para caracterizar los datos. En la Tabla 3.3 se pueden ver los test estadísticos más adecuados para realizar el contraste de hipótesis, en función del tipo diseño experimental seleccionado y la distribución de los datos. En caso de que la distribución sea normal se utilizarán test paramétricos y en caso contrario no paramétricos.

Tipo de diseño	Test paramétricos	Test no paramétricos
Un factor, un tratamiento		Test binomial Chi 2
Un factor, dos tratamientos	Test t Test F Test t emparejado	Mann-Whitney Chi 2 Wilcoxon
Un factor, más de dos tratamientos	ANOVA	Kruskal-Wallis Chi 2
Más de un factor	ANOVA	

Tabla 3.3. Test estadísticos a utilizar para los contrastes de hipótesis según el diseño experimental seleccionado

Para realizar el análisis de datos se pueden utilizar paquetes estadísticos o también técnicas utilizadas en el ámbito de la minería de datos para realizar lo que comúnmente se conoce como descubrimiento de conocimiento (*Knowledge Discovery*).

Más detalles sobre cómo realizar el análisis de datos se puede encontrar en Wohlin *et al.* (2012), Maxwell (2002), Juristo y Moreno (2001) y en libros específicos de estadística.

3.2.5 Presentación y difusión

Cuando se realiza un experimento se suelen presentar los hallazgos del mismo, lo que puede realizarse mediante una comunicación en un congreso o en una revista, un informe para la toma de decisiones, o como un paquete para replicación del experimento o como material educativo. En cualquier caso, es muy importante no olvidar ningún aspecto importante de cada una de las actividades del proceso experimental descritas previamente. Una buena documentación del experimento permitirá que otros investigadores puedan replicarlo o basarse en el conocimiento empírico que se haya adquirido durante la realización del experimento.

Jedlitschka *et al.* (2008) proponen unas directrices sobre cómo reportar experimentos, con el objetivo de homogeneizar la manera en la que se presentan los experimentos sin que se omita información relevante. Estas directrices incluyen los siguientes elementos: Título, Autores, Resumen estructurado, Palabras claves,

Introducción, Antecedentes, Planificación del experimento, Ejecución, Análisis, Discusión, Conclusiones y Trabajo Futuro, Agradecimientos, Referencias y Apéndices. De acuerdo al tipo de foro en el que se pretende difundir el experimento, puede ser importante enfatizar unos aspectos y también la longitud de la publicación puede impedir incluir toda la información. Por ello es importante poner el material experimental y si es posible un reporte técnico, con todos los detalles, disponible en una página web.

3.3 EJEMPLO DE UN EXPERIMENTO

A continuación se describe un experimento publicado en Fernández-Sáez *et al.* (2013) y que se ha desarrollado siguiendo el proceso experimental expuesto. Este experimento se replicó y se presentó como una familia de experimentos en Fernández-Sáez *et al.* (2014).

El objetivo de este experimento es saber qué tipos de diagramas UML son más útiles para mantener el código. Se consideraron diagramas de clases y de secuencia, con diferente origen: diagramas de diseño (*Forward Design diagrams*) o diagramas obtenidos utilizando técnicas de ingeniería inversa (*Reverse Engineered diagrams*).

El experimento se llevó a cabo en la Universidad de Sevilla, en España, en noviembre del año 2011. Todo el material utilizado en el experimento se encuentra disponible en: *http://alarcos.esi.uclm.es/originUMLmaintenance/*

A continuación se presenta el experimento describiendo todas las tareas del proceso experimental descrito previamente, aunque en algunos casos cambiaremos el orden de las tareas, para facilitar la comprensión.

3.3.1 Definición del alcance

Utilizando la plantilla GQM para la definición de objetivos, el objetivo del experimento es "Analizar el origen de los diagramas UML con el propósito de evaluar con respecto a su ayuda para mantener el código fuente desde el punto de vista de las personas que realizan el mantenimiento en el contexto de alumnos de la ingeniería informática, de la Universidad de Sevilla".

3.3.2 Planificación

3.3.2.1 SELECCIÓN DEL CONTEXTO

Los objetos experimentales consisten en los diagramas de clases y de secuencia UML y el código Java de un producto *software*. Los diagramas se obtuvieron a partir de dos orígenes diferentes:

- **II**: Diagramas UML obtenidos automáticamente a través de ingeniería inversa (II).
- **D**: Diagramas UML construidos manualmente en la fase de diseño.

En el primer caso se mantiene el código fuente utilizando como documentación los diagramas UML construidos en la fase de diseño. Y en el segundo caso no están disponibles los diagramas UML, por lo que es necesario obtenerlos a través de una técnica de II. Sólo se utilizaron diagramas de clases y de secuencia, porque se pueden obtener vía II y porque son dos de los diagramas UML más utilizados para diseñar productos *software*.

Los diagramas utilizados modelan un sistema de gestión de un centro deportivo que permite a los usuarios alquilar servicios (pista de tenis, etc.). El código Java implementa la aplicación del centro deportivo desarrollado como parte de una tesis de máster de un estudiante de la Universidad de Castilla-La Mancha. Es una aplicación cliente-servidor, cuyo código contiene 5123 líneas de código (LoC), por lo que se puede considerar como una aplicación pequeña pero realista. Los requisitos de mantenimiento fueron propuestos por el director del alumno de máster. En el caso de los diagramas D se disponía de 4 diagramas de clases con un total de 16 clases y 21 diagramas de secuencia con un total de 226 mensajes. En el caso de los diagramas II se disponía de 4 diagramas de clases con 21 clases y 11 diagramas de secuencia con 191 mensajes. El número de diagramas de secuencia y respectivos mensajes muestra son indicadores de que los diagramas son más grandes y complejos, Los diagramas II se obtuvieron utilizando la herramienta "*IBM Rational Software Architect*".

Y como sujetos se decidió escoger alumnos de quinto curso de la Ingeniería informática, como representantes de ingenieros *software* principiantes. Se presentarán más detalles de los sujetos en el siguiente apartado.

3.3.2.2 SELECCIÓN DE SUJETOS

En el experimento, participaron 40 alumnos del quinto curso de la Ingeniería informática en la Universidad de Sevilla, en España. En el momento de ejecución del experimento, estos alumnos estaban cursando la asignatura Ingeniería de *Software* III. Consideramos que el conocimiento adquirido sobre UML y Java, en este curso y en otros previos, fue suficiente para que los sujetos pudieran entender los diagramas y el código fuente del sistema seleccionado. Además se corroboró a través de un cuestionario que los sujetos completaron previamente, que todos tenían el mismo nivel de conocimiento. Los sujetos participaron voluntariamente y fueron seleccionados por conveniencia, es decir todos los sujetos disponibles en la asignatura Ingeniería de *Software* III. Para evitar el temor a la evaluación, que podrían causar amenazas a la validez del experimento, los sujetos no fueron evaluados por su rendimiento en el experimento. Así mismo, para evitar absentismo se les dijo a los alumnos que en el examen final podrían tener que realizar tareas similares a las del experimento.

Como sugieren Höst *et al.* (2000) y Kitchenham *et al.* (2002), se consideró apropiado contar con alumnos para la realización del experimento, debido a que las tareas que debían realizar no requerían una gran experiencia en entornos industriales. Y además el trabajar con alumnos, tiene algunas ventajas en este caso, como por ejemplo, que todos los alumnos tenían un nivel bastante homogéneo de conocimientos previos, la disponibilidad de un gran número de sujetos y tener la oportunidad de tener una prueba inicial sobre la veracidad o no de las hipótesis estudiadas.

3.3.2.3 SELECCIÓN DE VARIABLES

La variable independiente (también denominada factor principal) es el *Origen* de los diagramas, que es una variable nominal que toma dos valores (tratamientos): D (diseño), II (ingeniería inversa).

La variable dependiente es la *Mantenibilidad*, medida utilizando las siguientes medidas:

- **Efectividad de la Mantenibilidad** (*MEffec*): Esta medida está relacionada con la corrección de las respuestas, representa la habilidad de mantener correctamente el sistema y se calcula de la siguiente

manera: (*Número de tareas correctas – Número de tareas realizadas*) / *Número de tareas*. Se considera que mientras mayor sea su valor, mayor será la efectividad de la mantenibilidad.

- **Eficiencia de la Mantenibilidad** (*MEffic*): Esta variable representa está relacionada con el tiempo y con la corrección, y representa el número de tareas realizadas correctamente por unidad de tiempo. Se calcula de la siguiente manera: (Número de tareas correctas – Número de tareas realizadas)/tiempo.

3.3.2.4 FORMULACIÓN DE HIPÓTESIS

Se formularon las siguientes hipótesis relacionadas con las dos variables dependientes propuestas:

- $\mathbf{H_{1,0}}$: No existe una diferencia significativa en la *Efectividad* de los sujetos al realizar tareas de mantenimiento del código fuente, cuando usan diagramas obtenidos en la fase de diseño o diagramas obtenidos vía ingeniería inversa. $H_{1,1}$:$\neg H_{1,0}$
- $\mathbf{H_{2,0}}$: No existe una diferencia significativa en la *Eficiencia* de los sujetos al realizar tareas de mantenimiento del código fuente, cuando usan diagramas obtenidos en la fase de diseño o diagramas obtenido vía ingeniería inversa. $H_{2,1}$: $\neg H_{2,0}$

El objetivo del análisis estadístico será rechazar las hipótesis nulas y posiblemente aceptar las alternativas.

3.3.2.5 ELECCIÓN DEL DISEÑO

Se seleccionó un diseño balanceado inter-sujetos, es decir cada tratamiento se asignó a un único sujeto y cada tratamiento se asignó al mismo número de sujetos, en lugar de seleccionar un diseño intra-sujetos por restricciones de tiempo. Se intentaron mitigar ciertas amenazas inherentes al diseño inter-sujeto, considerando algunas sugerencias proporcionadas en Wohlin *et al.* (2012). Por ejemplo, para mitigar el efecto de la experiencia, en la primera sesión del experimento (sesión de entrenamiento) los sujetos debían rellenar un cuestionario sobre sus conocimientos previos y de acuerdo a la puntuación obtenida en este

cuestionario se asignaron aleatoriamente los sujetos a cada tratamiento, intentando tener un nivel de experiencia balanceado en cada uno de los dos grupos (G1: Diagramas II, G2: Diagramas D).

Todos los sujetos deberían realizar las mismas tareas, pero en diferente orden, para mitigar el efecto que pueda tener el orden de las tareas en el rendimiento de las tareas. Dicho orden se obtuvo aleatoriamente y eran los mismos para ambos grupos, hecho que permite mitigar además los efectos producidos por el aprendizaje.

3.3.2.6 INSTRUMENTACIÓN

En esta tarea, se prepararon los objetos experimentales que constaban de los diagramas de clases y de secuencia en sus dos versiones D e II y el código correspondiente a la aplicación seleccionada. También se diseñaron las tareas que deberían realizar los sujetos, con el fin de poder contrastar las hipótesis formuladas. Se consideraron tres tareas de mantenimiento adaptativo (para mejorar el sistema) y dos de mantenimiento correctivo (para corregir defectos), ambas incluían sólo la modificación del código fuente (ver Tabla 3.4). Las tareas seleccionadas se consideraron representativas de tareas de mantenimiento que pueden surgir en entornos industriales. Como los sujetos debían hacer las tareas manualmente usando papel y lápiz y considerando la dificultad de modificar el código fuente en papel, se les entregó una plantilla especial para estructurar la realización de las tareas de mantenimiento y así ser más fáciles de realizar, entender y corregir. Los sujetos debían utilizar diferentes plantillas dependiendo si la tarea requería mantener, una clase, un método o un atributo. Estas plantillas están disponibles junto a todo el material experimental en *http://alarcos.esi.uclm.es/originUMLmaintenance/*

También durante la instrumentación, se diseñó un cuestionario post-experimento (ver Tabla 3.5), que deberían completar los sujetos una vez realizadas las tareas de mantenimiento. Este cuestionario que constaba de 15 preguntas con respuestas de tipo Likert de cinco puntos, preguntas abiertas y preguntas tipo test de selección simple. El objetivo de este cuestionario era obtener la percepción de los usuarios sobre la ejecución del experimento, que podría ser útil para interpretar y explicar los resultados obtenidos.

Tareas	Resumen de las tareas	Tipo de mantenimiento	Máxima puntuación
T1	Cuando alguno de los servicios del centro deportivo no esté disponible, se deben cancelar todas las reservas de este tipo de servicio.	Correctivo	4 puntos
T2	El centro deportivo debe registrar el/los números de teléfono de sus clientes.	Adaptivo	5 puntos
T3	El sistema deberá emitir un ticket con las reservas de cada cliente en una determinada fecha.	Adaptivo	5 puntos
T4	Cuando se borre un socio o miembro del centro deportivo sus cuentas pendientes se den mantener en el sistema.	Correctivo	2 puntos
T5	El sistema deberá permitir registrar los datos de los instructores del centro deportivo.	Adaptivo	6 puntos

Tabla 3.4. Resumen de las tareas de mantenimiento

N	Descripción de la preguntas	Valoración de las respuestas
P1	Dificultad de las tareas	(1-5)
P2	El entrenamiento fue suficiente para realizar las tareas	(1-5)
P3	La claridad del material entregado	(1-5)
P4	Los objetivos de las tareas estaban claros para mi	(1-5)
P5	Las tareas que realicé estaban claras para mi	(1-5)
P6	No tuve dificultad para leer los diagramas.	(1-5)
P7	No tuve dificultad al leer el código fuente.	(1-5)
P8	El nivel de detalle de los diagramas era adecuado para realizar las tareas	(1-5)
P9	Los diagramas de clase entregados fueron útiles	(1-5)
P10	Indique en qué tareas no fueron útiles los diagramas de clases, justificando por qué	Pregunta abierta
P11	Los diagramas de secuencia entregados no fueron útiles	(1-5)
P12	Indique en qué tareas no fueron útiles los diagramas de secuencia, justificando por qué	Pregunta abierta
P13	Tuve tiempo suficiente para realizar las tareas	Tipo test
P14	¿Cuánto tiempo (en porcentaje) estuvo observando los diagramas?	Tipo test
P15	¿Cuánto tiempo (en porcentaje) estuvo observando el código fuente?	Tipo test

1 = muy de acuerdo; 2 = de acuerdo; 3 neutral; 4 = en desacuerdo; 5 = muy en desacuerdo (P2, P3, P4, P5, P6, P7, P9, P11)
1 = muy alto; 2 = alto; 3 = adecuado; 4 = bajo; 5= muy bajo (P8)
1= muy difícil; 2=difícil; 3=media; 4=fácil; 5 = muy fácil(P1)
1= muy claro; 2 = claro; 3 = adecuado; 4 = no claro; 5= muy poco claro (P3)
A = más tiempo necesario ; B = menos tiempo necesario; C = tiempo suficiente (P13)
A. <20%; B. >=20% y <40%; C. >=40% y <60%; D. >=60% y <80%; E. >=80% (P14, P15)

Tabla 3.5. Cuestionario post-experimento

3.3.3 Operación

A continuación se describirán las tres tareas de esta actividad, que son la preparación, la ejecución y la validación de los datos.

3.3.3.1 PREPARACIÓN

Dos semanas antes de la ejecución del experimento, con el fin de revisar el material experimental y el tiempo necesario para realizar el experimento, se realizó un estudio piloto con 6 estudiantes de doctorado de la Universidad de Castilla-La Mancha en la Escuela Superior de informática de Ciudad Real, España. Para la realización del experimento se utilizó el material diseñado y se asignó a 3 sujetos un tratamiento (Diagramas D y código fuente) y a otros 3 el otro tratamiento (Diagramas II y código fuente). Para la realización del estudio piloto los sujetos no tenían limitación de tiempo. El estudio piloto sirvió para corregir algunos errores de redacción y se detectó que dos horas eran suficientes para ejecutar el experimento, que era el tiempo del que se disponía.

Una vez se modificó el material de acuerdo a consideraciones que surgieron tras el estudio piloto, en una primera sesión realizada el día anterior a la ejecución del experimento se llevó a cabo el entrenamiento, en el cual se repasaron conceptos de UML y de Java, se realizó de manera conjunta con el experimentador un ejemplo de tareas similares a las que tendrían que realizar en el experimento y para finalizar esta sesión los sujetos tuvieron que completar el cuestionario de conocimientos previos.

3.3.3.2 EJECUCIÓN

El experimento se realizó en una segunda sesión, en un aula de clase, y los sujetos estaban supervisados por el profesor de la asignatura y por el

experimentador y no se permitió la comunicación entre los sujetos. Se les indicó que cualquier duda se le debía comunicar al experimentador.

Al inicio de esta sesión, los sujetos fueron divididos en dos grupos, G1 y G2, utilizando los resultados del cuestionario de conocimientos previos, para lograr grupos balanceados con respecto a la experiencia. Luego los sujetos recibieron el material para realizar las tareas de mantenimiento, los sujetos de G1 recibieron los diagramas D y el código fuente y los sujetos de G2 recibieron los diagramas II y el código fuente.

Se les indicó que no podían mirar los diagramas y el código antes de empezar a realizar las tareas. El orden de ejecución debía ser el siguiente: primero debían anotar el tiempo de inicio y luego comenzar a resolver las tareas de mantenimiento en el orden indicado y una vez finalizadas debían anotar el tiempo de fin. En este caso se les pido que sean precisos al apuntar estos tiempos y que en la mitad de las tareas no se pusieran a hacer otra cosa, porque sino el tiempo no sería real y podrían invalidar la ejecución del experimento. Para registrar el tiempo debían utilizar el reloj proyectado en una pantalla, anotando horas, minutos y segundos.

Una vez realizadas las tareas de mantenimiento, los sujetos debían entregar todo el material y a continuación al experimentador les entregó el cuestionario post-experimento. Para la realización de todas las tareas disponían como máximo de dos horas.

Para evitar posibles sesgos no se desvelaron las hipótesis bajo estudio, antes de la realización del experimento, aunque se les dijo que este experimento era parte de una investigación sobre el beneficio de utilizar UML en el mantenimiento. Además se les dijo que el resultado de las tareas sería confidencial y que no serían evaluados por su rendimiento en el experimento, pero que tareas similares se iban a incluir en el examen final de la asignatura. También se les comunicó que se les informaría cuando se finalizara la redacción del informe que reporta este experimento y sus resultados; y que los autores se lo podrían facilitar bajo petición.

3.3.3.3 VALIDACIÓN DE LOS DATOS

Después de la ejecución del experimento, el experimentador recolectó todos los datos en una tabla diseñada para tal fin, en el que se le otorgaba a cada tarea una puntuación considerando el puntaje máximo asignado (ver Tabla 3.4) y restando las tareas no realizadas correctamente. Las tareas incorrectas no se puntuaron negativamente. No se detectaron valores atípicos, aunque si se observó que no todos los sujetos finalizaron todas las tareas por falta de tiempo.

3.3.4 Análisis e Interpretación

El procedimiento seguido para analizar los datos es el siguiente:

1. Realizar un estudio de los estadísticos descriptivos correspondientes a las medidas de la variable dependiente (*MEffec* y *MEffic*) para describir y resumir los valores de las mismas.
2. Con el fin de decidir si utilizar test paramétricos o no-paramétricos para probar las hipótesis se analiza si los datos tienen una distribución normal o no, utilizando el test de Kolmogorov-Smirnov y también se analiza la homogeneidad de las varianzas con el test de Levene.
3. Contrastar las hipótesis utilizando los test estadísticos apropiados según los resultados obtenidos en el paso anterior.
4. Analizar los datos del cuestionario post-experimento utilizando gráficos de barras y el test T cuando sea oportuno de acuerdo a la naturaleza de los datos.

Al aplicar los test estadísticos para las pruebas de hipótesis se decidió aceptar una probabilidad del % de cometer un error de Tipo I.

A continuación se presentan los resultados obtenidos en el análisis de datos, que se realizó utilizando el paquete estadístico *SPSS* (SPSS, 2003).

3.3.4.1 ANÁLISIS DE LOS ESTADÍSTICOS DESCRIPTIVOS

La Tabla 3.6 muestra los estadísticos descriptivos correspondientes a las dos medidas de la mantenibilidad, *MEffec* y *MEffic*. En esta tabla se presentan las siguientes columnas: N representa el número de sujetos, ($\overline{X}$) la media, la mediana y DS la desviación estándar, agrupadas por el tipo de Origen, II y D.

Analizando el contenido de la Tabla 3.6, se puede observar que las medias son mejores cuando los sujetos usan diagramas D (construidos en la fase de diseño) para ambas medidas, aunque la diferencia es muy pequeña con respecto a las medias de los diagramas II.

Origen	**N**	***MEffec***			***MEffic***		
		$\overline{X}$	Mediana	DS	$\overline{X}$	Mediana	DS
II	20	0,641	0,6818	0,165	0,00270	0, 00283	0,00079
D	20	**0,650**	0,6818	0,148	**0,00273**	0, 00303	0,00072

Tabla 3.6. Estadísticos descriptivos de MEffec y MEffic

3.3.4.2 PRUEBAS DE HIPÓTESIS RELACIONADAS CON EL ORIGEN DE LOS DIAGRAMAS

Como la distribución de los datos no era normal y no había homogeneidad de varianzas se decidió probar las hipótesis relacionadas con el *origen* de los diagramas, usando el test no-paramétrico U de Mann Whitney. En las Tablas 3.7 y

3.8. se muestran los resultados de aplicar el test U de Mann-Whitney: la columna *Origen* representa a la variable independiente, *p-valor* es el nivel de significación, *po* es la potencia observada estimada, *te* es el tamaño del efecto y *r* representa si se puede o no rechazar la hipótesis nula.

A continuación se presentan las pruebas de hipótesis considerando los datos obtenidos en las tareas de mantenimiento en general y también se analizarán por separado los datos obtenidos para las tareas de mantenimiento adaptativo y correctivo.

3.3.4.3 PRUEBA DE LAS HIPÓTESIS DE LA EFECTIVIDAD DEL MANTENIMIENTO: *MEFFEC* ($H_{1,0}$)

Teniendo en cuenta los resultados mostrados en la Tabla 3.7 no se puede rechazar la hipótesis nula $H_{1,0}$, debido a que el p-valor es 0,957, que no es menor 0,05. Esto demuestra que parece que los diferentes orígenes de los diagramas UML no afectan en la efectividad del mantenimiento, es decir en cuán bien realizaron los sujetos las tareas de mantenimiento. La potencia observada (*po*) es baja, y puede ser debido a que el tamaño del efecto (*te*) es pequeño, por ello las conclusiones obtenidas no se pueden considerar como demasiado firmes.

MEffec			
p-valor	po	te	r
0,957	0,054	0,001	NO

Tabla 3.7. Resultados del test U de Mann-Whitney para MEffec

También se probó la hipótesis sobre *MEffec* para cada tipo de tarea de mantenimiento, obteniendo resultados no significativos, 0,606 fue el p-valor para el mantenimiento adaptativo y 0,119 para el correctivo.

3.3.4.4 PRUEBA DE LAS HIPÓTESIS DE LA EFICIENCIA DEL MANTENIMIENTO *MEFFIC* ($H_{2,0}$)

En la Tabla 3.8 se puede observar que no existe un efecto significativo (el p-valor es 0,534, que no es menor que 0,05) con respecto al *Origen* de los diagramas y la Eficiencia del Mantenimiento. Y también en este caso el *po*, es bajo, por lo que las conclusiones tampoco son firmes.

MEffic			
p-valor	po	te	r
0,534	0,051	0,0003	NO

Tabla 3.8. Resultados del test de U de Mann-Whitney para MEffic

Los resultados obtenidos para el mantenimiento adaptativo y correctivo no son significativos, los p-valores obtenidos son 0,449 y 0,290 respectivamente.

También se probaron las hipótesis considerando sólo el tiempo, y tampoco se obtuvieron valores significativos.

3.3.4.5 RESULTADOS DEL CUESTIONARIO POST-EXPERIMENTO

El análisis de las repuestas recogidas en el cuestionario post-experimento revelan que los sujetos no consideraron suficiente las dos horas asignadas para la realización del experimento (ver Figura 3.3) y que la mayoría percibió las tareas de dificultad media (ver Figura 3.4), independientemente del tratamiento recibido. Aunque a través del estudio piloto, se detectó que dos horas eran suficientes, puede haber ocurrido que los sujetos que realizaron el experimento, al ser estudiantes de grado, tuvieran menos experiencia que los alumnos de doctorado y por ello necesitaran más tiempo. Un 10% más de los sujetos del grupo G2 (Diagramas II) no terminaron las tareas, en comparación con el grupo G1 (Diagramas D).

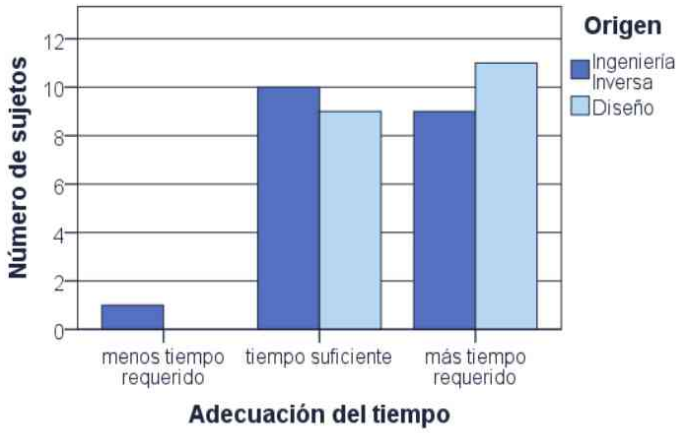


Figura 3.3. Percepción de los sujetos con respecto a la adecuación del tiempo

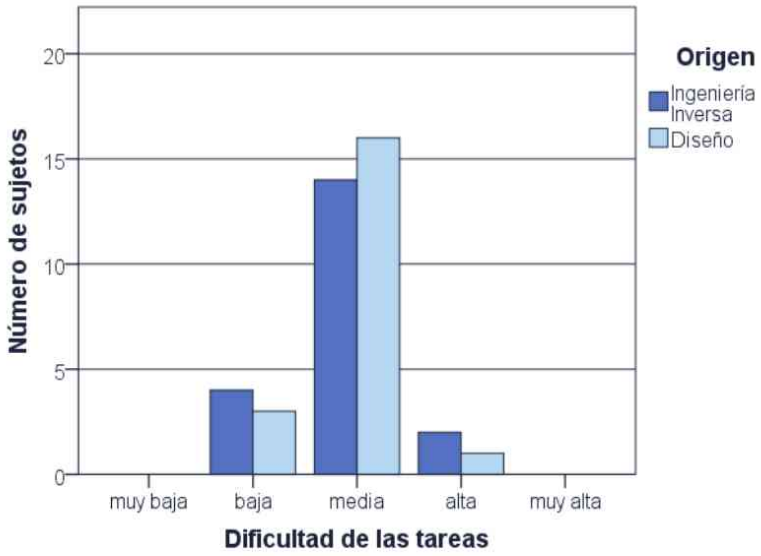


Figura 3.4. Percepción de los sujetos sobre la dificultad de las tareas

En la pregunta sobre la adecuación del nivel de detalle, la mayoría de los sujetos que recibieron los Diagramas D estuvieron de acuerdo con el nivel de detalle de los diagramas recibidos; sin embargo, la mayoría de los sujetos que recibieron los Diagramas II indicaron que los diagramas tenían demasiado nivel de detalle (ver Figura 3.5).

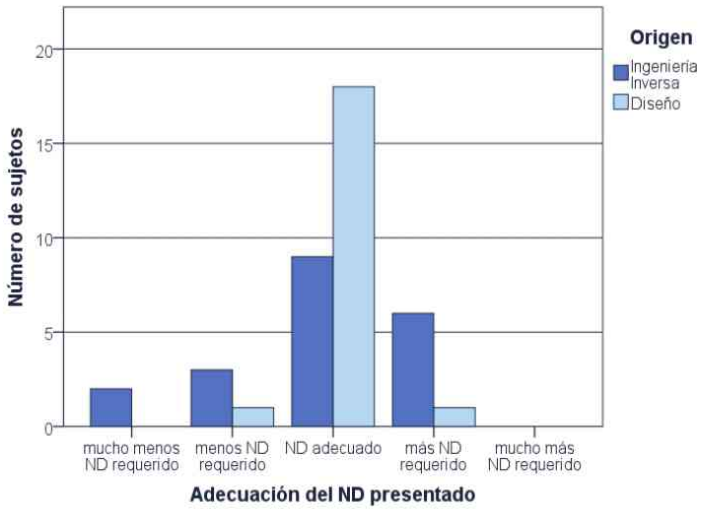


Figura 3.5. Percepción de los sujetos sobre la adecuación del nivel de detalle

Como se muestra en la Figura 3.6, los sujetos que recibieron los Diagramas D tuvieron menos dificultades al leer los diagramas utilizados, en comparación con el grupo que recibió los Diagramas II. Además, dada la naturaleza de los datos, probamos si había diferencia significativa en la dificultad percibida por los sujetos dependiendo si recibieron los Diagramas D o II, a través del test T. Para probar el test T, comparamos las repuestas de los sujetos (de 1 a 5) agrupadas considerando los diagramas que recibieron (D o II). El resultado del test T muestra una diferencia significativa (p-valor = 0,001, menor que 0,05) y la potencia observada es alta (0,957). Estos resultados nos permiten concluir, que efectivamente los sujetos que recibieron los Diagramas II tuvieron más dificultad al leer los diagramas.

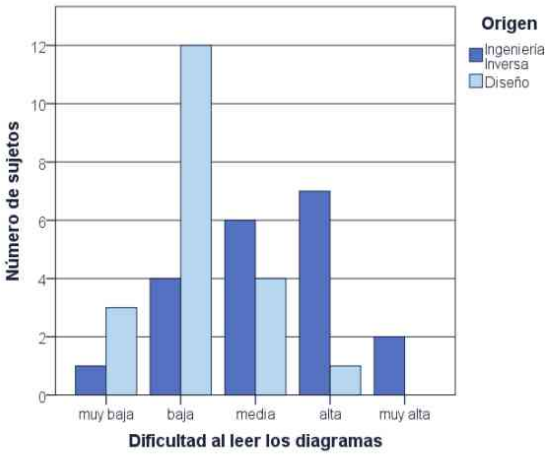


Figura 3.6. Percepción de los sujetos sobre la dificultad al leer los diagramas

También en el cuestionario post-experimento, los sujetos debían indicar cuan útiles les habían resultado los diagramas para resolver las tareas. Los diagramas de clases resultaron útiles en ambos grupos, en más o menos la misma proporción (ver Figura 3.7). Con respecto a los diagramas de secuencia, 15 sujetos de los 20 que recibieron los Diagramas II, indicaron que no les resultaron útiles y que eran muy difíciles de entender, en oposición a 6 sujetos en el grupo de los Diagramas D (ver Figura 3.8). Esto se puede haber producido debido a que los diagramas de secuencia obtenidos por ingeniería inversa tienen un elevado nivel de detalle.

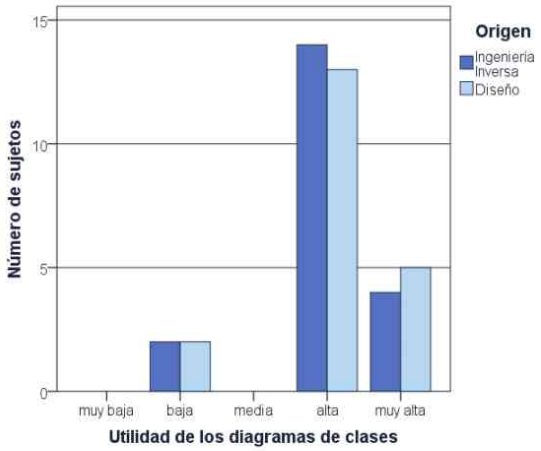


Figura 3.7. Percepción de los sujetos sobre la utilidad de los diagramas de clases utilizados

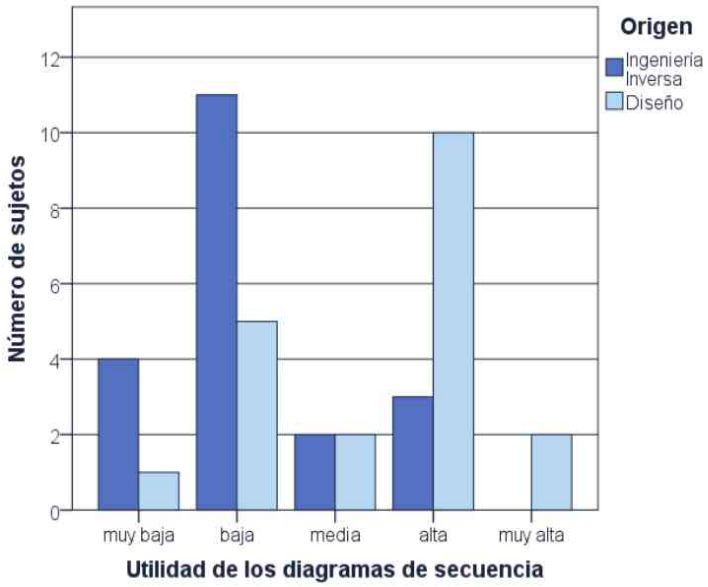


Figura 3.8. Percepción de los sujetos sobre la utilidad de los diagramas de secuencia utilizados

Con el objetivo de obtener información acerca de si los sujetos utilizaban los diagramas o no a la hora de mantener el código fuente, se les preguntó que artefacto habían utilizado para resolver cada tarea (código fuente, diagramas de clases, diagrama de secuencia). La mayoría de los sujetos utilizó el código fuente para resolver las tareas (el 90% de los sujetos del grupo que recibió los Diagramas II y el 86% de los sujetos del grupo que recibió los Diagramas D), lo que estaba dentro de lo previsto.

La mayoría de los sujetos también utilizaron los diagramas de clases (El 80% de los sujetos del grupo que recibió los Diagramas II y el 74% del grupo de los Diagramas D), porcentajes que son coherentes con lo respondido acerca de la utilidad de los diagramas que recibieron (ver Figura 3.9). En el caso del grupo de los Diagramas II, los sujetos utilizaron los diagramas de clases en la misma proporción para el mantenimiento correctivo y para el mantenimiento adaptativo. Mientras que en el grupo de los Diagramas D, los sujetos utilizaron un 7% más los diagramas de clases en el mantenimiento correctivo.

Con respecto a los diagramas de secuencia su uso fue muy bajo en general, tan sólo el 33% de los sujetos los utilizaron, en el mismo porcentaje para los dos grupos (D e II). Este es consistente con la percepción de los sujetos del grupo de los diagramas II, sobre la utilidad de los diagramas de secuencia recibidos, mostrada en la Figura 3.10. En el caso del grupo de los Diagramas D, existe una leve inconsistencia, porque aunque los consideraron útiles no los utilizaron. En ambos grupos utilizaron más los diagramas de secuencia para el mantenimiento correctivo (una diferencia del 20% y el 27% respectivamente).

3.3.4.6 RESUMEN Y CONCLUSIONES DEL ANÁLISIS DE DATOS

Los estadísticos descriptivos mostraron que los sujetos que utilizaron los diagramas obtenidos en la fase de diseño obtuvieron valores levemente mejores tanto en la *Efectividad* como en la *Eficiencia* del *Mantenimiento*, lo que indica, en cierta medida, que los diagramas de diseño mejoran el mantenimiento del código fuente.

Con respecto a los resultados de las pruebas de hipótesis, tanto la *Eficiencia* como la *Efectividad* del *Mantenimiento*, no se vieron afectadas significativamente por el *Origen* de los diagramas UML, es decir los resultados obtenidos no permitieron rechazar las hipótesis nulas porque los niveles de significación eran mayores que 0,05. Las potencias observadas de los test eran bajas, por ello la posibilidad de cometer un error al aceptar la hipótesis nula era alta. Por ello, no se pueden obtener resultados concluyentes y es necesario realizar réplicas.

Los resultados del cuestionario post-experimento, que recogió las percepciones de los sujetos, mostraron mejores resultados para los diagramas obtenidos en la fase de diseño, estos diagramas les resultaron más fáciles de entender, por tener menor nivel de detalle y por ello los utilizaron más a la hora de mantener el código.

Los resultados obtenidos nos llevan a recomendar, con cierta precaución, que es conveniente seguir un desarrollo centrado en modelos y mantener actualizados los diagramas UML, especialmente los diagramas de clases. Estas recomendaciones son válidas en el contexto de ingenieros de *software* principiantes y sistemas pequeños relacionados con dominios conocidos. Para corroborar la validez de los resultados obtenidos y para poder generalizarlos es necesario realizar réplicas, que los autores de esta investigación han realizado y publicado en Fernández-Sáez *et al.* (2014).

3.3.5 Amenazas a la validez

A continuación se comentan algunos aspectos que pueden haber atentado contra la validez del experimento y cómo se intentaron mitigar:

- **Validez externa**: La validez externa que tiene que ver con la posibilidad de generalización de los resultados, se puede haber visto afectada, por haber utilizado estudiantes como sujetos, por no considerarlos representativos de los profesionales, aunque las tareas que debían realizar no requerían un alto nivel de experiencia en la industria. Por ello podemos considerar los resultados como válidos en el ámbito de ingenieros de *software* principiantes, siguiendo recomendaciones de la literatura (Höst *et al.*, 2000; Kitchenham *et al.*, 2002). Otro aspecto que puede amenazar la validez externa tiene que ver con los objetos utilizados. En este caso no existió tal amenaza porque tanto los diagramas UML como el código fuente seleccionados pertenecen a un caso real, representativo de un sistema real pequeño y relacionado con un dominio conocido. Para aumentar la validez externa es recomendable realizar réplicas con profesionales en entornos industriales considerando sistemas de mayor tamaño y de dominios poco conocidos.
- **Validez interna**: Los aspectos relacionados con las amenazas a la validez externa se trataron de mitigar al diseñar el experimento: Cada grupo se formó balanceando el nivel de experiencia, considerando los resultados obtenidos en el cuestionario sobre conocimientos previos. Además, los sujetos manifestaron en el cuestionario post-experimento que el material que recibieron y las tareas estaban bien explicadas, hecho

que también se había corroborado en el estudio piloto. Para evitar el abandono de los sujetos, es decir que asistieran a la primera sesión pero no a la segunda, se les dijo que tareas similares tendrían en el examen final de la asignatura Ingeniería del Software III. Para mitigar el efecto de aprendizaje se varió el orden de las tareas que debían realizar, y para evitar el temor a ser evaluados, se les dijo a los sujetos que no serían evaluados por el rendimiento en su experimento. También se prohibió la comunicación entre los sujetos para evitar plagios, supervisándolos por el experimentador durante la ejecución del experimento.

- **Validez de constructo**: Las medidas seleccionadas son medidas utilizadas normalmente para medir efectividad y la eficiencia y el cuestionario post-experimento fue diseñado utilizando cuestionarios y escalas estándares.
- **Validez de la conclusión**: Para asegurar la validez de las conclusiones se seleccionaron los test estadísticos apropiados teniendo en cuenta la naturaleza de los datos.

3.4 FAMILIAS DE EXPERIMENTOS

Los experimentos o estudios aislados, difícilmente proporcionan información suficiente para responder a las preguntas que definen una investigación. Por ello, es interesante que los experimentos formen parte de familias de estudios, más que considerar los experimentos aislados (Basili *et al.*, 1999). Estas familias de experimentos nos pueden permitir extraer conclusiones relevantes sobre hipótesis, que no podrían sugerir los estudios individualmente. Los estudios que conforman las familias de experimentos deben ser planificados adecuadamente como réplicas de los experimentos originales. El concepto de réplica se verá en el apartado 3.4. Puesto que la planificación de estudios individuales ya sabemos cómo abordarla, tal y como se indica en la Figura 3.1, debemos ampliar este proceso al caso de estudios que forman parte de una familia de experimentos.

En este apartado presentaremos un proceso, basado en las guías propuestas en Ciolkowski *et al.* (2002), para definir de forma sistemática una familia de experimentos. Comprende tres pasos fundamentales en cualquier estudio empírico, que son preparación, ejecución y análisis, como se muestra en la Figura 3.9.

La fase de preparación comprende la definición de uno o más objetivos, que se analizarán posteriormente, con objeto de garantizar la coherencia de los datos en cuanto a que sean comparables los de todos los estudios. Todos éstos tendrán un marco común (definición del contexto), incluido un plan común GQM

para la medición. El segundo paso del proceso es desarrollar los estudios individuales, utilizando para ello tanto el contexto, como la definición y material de la familia. Es verdad que se deben seguir los mismos pasos que en cualquier experimento individual, pero en este caso el esfuerzo de preparación se reduce, ya que se reutiliza información común a toda la familia. En la Figura 3.9 se muestra como se reutilizan la definición del contexto, el marco del diseño y los materiales de la familia. El análisis de cada experimento puede sugerir cambios en el contexto para ampliar el marco de la familia. Por ejemplo, pueden considerarse más variables o utilizar material adicional.

El último paso consiste en utilizar los datos de todos los estudios de la familia para analizarlos conjuntamente, o lo que se suele llamar "agregación o integración" de resultados, que se abordará en el apartado 3.6.

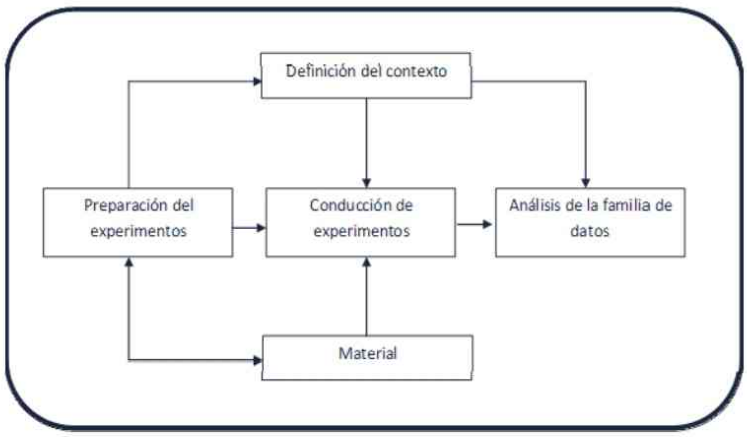


Figura 3.9. Fases del desarrollo de una familia de experimentos (*Ciolkowski et al., 2002*)

Esta introducción a las familias de experimentos, muestra que aunque la definición de las mismas cueste más esfuerzo que la de un experimento aislado, a cambio promete resultados más beneficiosos y útiles. Los mayores costes se deben, entre otros, a la precisión con que debemos definir el contexto y la variabilidad que admite. A cambio, el investigador se provee de material y documentación reutilizable en otros experimentos. Con ello además, consigue un valor adicional para esos experimentos que, al formar parte de la familia, proporcionará resultados con un alcance mayor, esto es, información añadida a un cuerpo de conocimiento representado por la propia familia. Por último, las familias de experimentos proporcionan respuestas a preguntas cuyo alcance sobrepasa el ámbito que puede cubrir un experimento aislado, por ejemplo qué factores de contexto influyen en los resultados, o para mejorar la potencia de los test. En definitiva, es una forma de generalizar los resultados a través de los estudios.

3.5 RÉPLICAS

La realización de réplicas de los estudios empíricos es una actividad esencial para la construcción de conocimiento en cualquier ciencia empírica. Como señaló Karl Popper, "*No tomamos nuestras propias observaciones muy en serio, o las aceptamos como observaciones científicas, hasta que no las hemos repetido y probado*" (Popper 1959). Lindsay y Ehrenberg argumentan que las réplicas "*... que se necesitan no sólo para validar nuestros resultados: pero lo más importante: para establecer el rango de condiciones radicalmente diferentes en las que los resultados son válidos: y las excepciones predecibles*" (Lindsay y Ehrenberg 1993).

En la ingeniería del *software*, el primer artículo que publicó una réplica de un experimento fue publicado en el año 1994 (Daly *et al.* 1994). Alrededor del mismo periodo, Brooks *et al.* (1995) elaboraron un conjunto de principios para la replicación de estudios en la ingeniería del *software*. A finales de los años 90, Basili *et al.* (1999) presentaron un marco para organizar conjuntos de experimentos relacionados (familias) y la generación de conocimiento a partir de ellos. Estas obras seminales inspiraron y guiaron a los investigadores para realizar réplicas, y también a los investigadores a estudiar las cuestiones relacionadas con la realización y presentación de las réplicas. Recientemente, la comunidad de la ingeniería del *software* empírica comenzó a tratar cuestiones importantes sobre las réplicas, como el papel de los paquetes de laboratorios para apoyar la realización de réplicas (Kitchenham, 2008; Shull *et al.* 2008), la falta de incentivos para realizar réplicas (Kitchenham, 2008), la importancia del conocimiento tácito (Shull *et al.*, 2002), aspectos sobre la comunicación entre los investigadores (Vegas *et al.*, 2006), la necesidad de contar con directrices específicas sobre cómo reportar las réplicas (Carver 2010), la dificultad de realizar réplicas con sujetos humanos (Lung *et al.*, 2008; França *et al.* (2010), el papel de los diferentes tipos de réplicas (Gómez *et al.*, 2010a,b; Krein y Knutson 2010), la generación de conocimiento a través de las diferencias encontradas en las réplicas (Juristo y Vegas, 2009), y cómo gestionar las interacciones entre los experimentadores cuando se realizan réplicas (Juristo *et al.*, 2013).

Como vemos, el tema de la réplicas está cobrando cada vez más interés, en este sentido da Silva *et al.* (2014) han realizado un mapeo sistemático de la literatura (SMS) incluyendo artículos publicados entre los años 1994 y 2010, para tener una visión más global de qué se ha hecho hasta ahora en el campo de las réplicas en la ingeniería del *software*, centrándose concretamente en réplicas de experimentos entre los años 1994 y 2010. En este mapeo sistemático encontraron 96 artículos incluyendo 133 réplicas, de las cuales el 70% de las réplicas se publicaron a partir del año 2004 y fueron réplicas internas. El 55% de las réplicas estaban relacionadas con los siguientes temas: requisitos de *software*, construcción

de *software* y calidad de *software*. Se llega a la conclusión, de que si bien la comunidad de la ingeniería del *software* empírica está preocupada en aspectos metodológicos de las réplicas, como reflejan los artículos citados anteriormente, esto no concuerda luego con el poco número de réplicas que realmente se llevan a cabo. Por lo que concluyen que todavía se necesitan mayores incentivos para realizar réplicas externas, mejores estándares para reportar estudios empíricos y sus réplicas, y agendas de investigación colaborativas que contribuyan a acelerar la realización y publicación de las réplicas.

Un evidencia más, de la creciente interés en la comunidad de los investigadores sobre la realización de réplicas y del estudio de diferentes aspectos de las réplicas, es la organización de tres ediciones consecutivas del taller internacional sobre réplicas "*International Workshop on Replication in Empirical Software Engineering Research* (RESER)", en los últimos años.

Si bien hemos incluido las réplicas, dentro del capítulo de experimentos, también se pueden replicar otros tipos de estudios empíricos, pero por su naturaleza y como se mostró en la Tabla 1.2, los experimentos y las encuestas son los más fáciles de replicar.

Existen varias clasificaciones de tipos de réplicas. Por ejemplo Brooks *et al.* (1995) definen:

- **Réplica interna**: réplicas realizadas por las personas que realizaron el experimento original.
- **Réplica externa**: réplicas realizadas por personas que no tuvieron nada que ver con la realización del experimento original.

En (Lindsay y Ehrenberg, 1993), se distinguen dos tipos de réplicas:

- **Réplica similar** (*closed*): Cuando todas las condiciones del experimento original se mantienen lo mas similares posible.
- **Réplica diferenciada** (*differentiated*): Cuando no se mantienen las condiciones del experimento original.

Y Basili *et al.* (1999) proponen una clasificación más detallada:

- Réplicas que no varían las hipótesis. Las réplicas de este tipo no varían ni las variables dependientes ni las variables independientes del experimento original. Dentro de este tipo de réplicas se puede distinguir entre:

  - Estrictas, que duplican el experimento original con la mayor precisión posible, son necesarias para incrementar la fiabilidad en la validez de la conclusión del experimento.

  - Réplicas que modifican la forma en que el experimento se realiza, con el fin de incrementar nuestra confianza en los resultados experimentales, cambiando la forma en que se controlan las amenazas a la validez interna.

- Réplicas que varían las hipótesis. Aunque cambian algunas variables permanecen en el mismo nivel de especificidad que el experimento original. En este caso se puede distinguir entre:

  - Réplicas que varían variables intrínsecas a los objetos del estudio (independientes). Estas réplicas incorporan nuevas variables independientes, modificando así la especificación del proceso a estudiar.

  - Réplicas que varían variables intrínsecas (dependientes) al objetivo de la evaluación. Estas réplicas permiten cambiar la forma en que se mide un atributo o constructo-efecto de interés, así se puede entender qué dimensiones afectan o son más importantes para ese atributo en cuestión.

  - Réplicas que varían las variables de contexto del entorno en el se evalúa la solución. Estos estudios permiten identificar los aspectos del entorno que pueden afectar a los resultados del proceso en investigación, por lo tanto nos ayudan a entender la validez externa.

  - Réplicas que extienden la teoría. Ayudan a determinar los límites de la efectividad de un proceso, pues permiten hacer grandes cambios en los procesos, los productos y/o los modelos del contexto para ver si los principios básicos siguen cumpliéndose.

3.6 AGREGACIÓN DE RESULTADOS

La síntesis o agregación cuantitativa de los resultados obtenidos en familias de experimentos se realiza comúnmente a través de técnicas de meta-análisis. Según Glass *et al.* (1981) "*El meta-análisis se refiere al análisis del análisis... al análisis estadístico de una colección de resultados procedentes de estudios individuales*: *y cuyo propósito es integrar dichos resultados*. *Supone una alternativa rigurosa frente a las discusiones meramente narrativas sobre los*

resultados de una colección de estudios...". Es decir el meta-análisis se refiere a un conjunto de técnicas estadísticas que se utilizan para analizar los resultados obtenidos en múltiples estudios empíricos. Al combinar los resultados de varios estudios experimentales, el meta-análisis permite generar conocimiento más general y fiable que el de los resultados obtenidos por los estudios individuales, ya que dicho conocimiento está sustentado por una mayor cantidad de evidencia empírica.

Si todos los estudios incluidos en el proceso de meta-análisis, fueran igualmente precisos y utilizaran exactamente las mismas variables dependientes, bastaría con promediar los resultados de cada uno de ellos para obtener así una conclusión final (Borenstein *et al.*, 2007). Sin embargo, en la práctica no todos los estudios tienen la misma precisión, por ello cuando se los combine se debe asignar un mayor peso a los estudios que permiten obtener información más fiable. Esto se logra combinando los resultados mediante un promedio ponderado (Cochrane, 2003). Por otra parte, para poder solucionar los problemas vinculados a la no uniformidad de las variables dependientes, los métodos de meta-análisis expresan sus resultados mediante un índice denominado "tamaño del efecto", el cual es un estimador de la magnitud de relación entre un tratamiento y una variable dependiente (Cochrane, 2003) y es aplicable a cualquier medida de la diferencia de los resultados de dos grupos. Por ello, el objetivo del meta-análisis es encontrar el tamaño del efecto global obtenido a partir de los tamaños de los efectos de cada estudio individual y reflejará el grado en el que el fenómeno bajo estudio está presente en la población en su conjunto.

El método de meta-análisis para variables continuas (las más utilizadas en ingeniería de *software*) más común, es el método paramétrico de diferencias medias ponderadas (en inglés *Weighted Mean Difference*) (Hedges y Olkin, 1985). No obstante existen otros métodos alternativos menos difundidos para el cálculo del tamaño de efecto:

- **Paramétricos** de Proporción de Respuesta (en inglés *Response Ratio*) (Gurevitch y Hedges, 2001),
- **No paramétricos** de Proporción de Respuesta (Gurevitch y Hedges, 2001) y Conteo de votos (en inglés Vote Counting) (Hedges y Olkin, 1985).

Un estudio empírico debe cumplir con las siguientes características para poder ser incluido en un estudio de meta-análisis (Pickard *et al.*, 1998):

- Ser del mismo tipo, por ejemplo experimentos controlados.
- Tener las mismas hipótesis.

- Tener las mismas medidas para las variables independientes y dependientes.
- Reportar los mismos factores explicativos.

Un estudio de meta-análisis consiste en los siguientes pasos:

- Decidir que estudios se incluirán en el meta-análisis.
- Extraer el tamaño del efecto del estudio primario publicado, o estimarlo en el caso de que no esté publicado.
- Combinar los tamaños del efecto de los estudios primarios para estimar el tamaño del efecto global.

Existen numerosos ejemplos de uso de meta-análisis en familias de experimentos realizadas en la ingeniería del *software*. Los primeros trabajos sobre el uso de meta-análisis en ingeniería del *software* aparecieron a finales de los años 90 (Porter y Johnson, 1997; Miller, 1999; Pickard *et al.*, 1998; Laitenberger *et al.*,1999; Hayes, 1999, entre otros). Más recientemente han aparecido otros como: Dyba *et al.* (2007), Cruz-Lemus *et al.* (2009; 2011) y Scaniello *et al.* (2013), etc.

También en cierto que no siempre se puede aplicar el meta-análisis para agregar resultados cuantitativos de experimentos, por no cumplir con las restricciones impuestas para poder hacer el meta-análisis, por ello en Ciolkowski (2009) y Dieste y Juristo (2011) se presentan propuestas de agregación alternativas.

Un estudio más exhaustivo de métodos cuantitativos de agregación de experimentos en ingeniería de *software* se puede encontrar en Dieste *et al.* (2008).

A modo de ejemplo, a continuación describiremos la familia de experimentos publicada en Cruz-Lemus *et al.* (2009) que presenta un estudio de meta-análisis. Además, en el capítulo 7, se presenta otra familia de experimentos para validar un conjunto de medidas para diagramas de clases UML y la agregación de sus resultados utilizando meta-análisis.

3.7 EJEMPLO DE UNA FAMILIA DE EXPERIMENTOS

A continuación presentaremos una visión global de la familia de experimentos publicada en Cruz-Lemus *et al.* (2009), luego la descripción de cada serie de experimentos, las factores que pueden amenazar la validez de los experimentos realizados y finalmente un meta-análisis llevado a cabo para agregar los resultados obtenidos. De cada serie de experimentos se detallarán

resumidamente solo las principales características, ya que el principal objetivo es presentar de manera detallada el estudio de meta-análisis.

3.7.1 Visión global de la familia de experimentos

Esta familia de experimentos consta de tres series de experimentos: 1) Primer experimento y su réplica (E1, R1), 2) Segundo experimento y su réplica (E2, R2) y 3) Tercer experimento (E3). Algunas de sus características se presentan en la Figura 3.10, como así también su cronología. Todo el material experimental se encuentra disponible en *http://alarcos.inf-cr.uclm.es/CSExperiments/*.

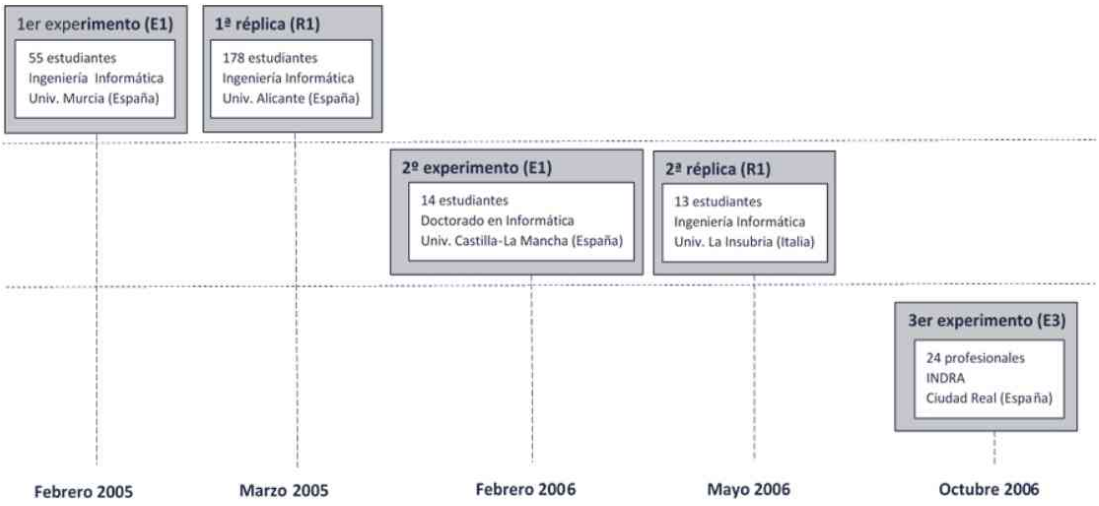


Figura 3.10. Cronología de la familia de experimentos

El primer experimento y su réplica (E1 y R1) se llevaron a cabo en dos universidades españolas en 2005. Los materiales y las tareas a realizar fueron muy simples y el conocimiento previo de los estudiantes de grado utilizados como sujetos no fue avanzado. Estos estudios proporcionaron algunos resultados iniciales que luego fueron reforzados con los otros experimentos de la familia.

El segundo experimento y su réplica (E2 y R2) se llevaron a cabo en dos universidades, una en España y otra en Italia, en 2006. El conocimiento previo que tenían los estudiantes de Italia fue similar al que tenían los estudiantes de los experimentos anteriores (E1 y R1), pero los sujetos españoles eran estudiantes de doctorado y tenían más experiencia en el modelado de sistemas. Además, se mejoraron los materiales y las tareas asignadas a los sujetos, especialmente con el uso de la teoría CTML (*Cognitive Theory of Multimedia Learning*) (Mayer 2001) para evaluar el conjunto completo de variables del diseño experimental.

La principal diferencia entre los primeros experimentos y sus réplicas (E1, R1, E2, y R2) respecto al tercer experimento (E3) es que en éste último lo sujetos experimentales fueron profesionales informáticos. Otra característica que hace que el tercer experimento fuera distinto, es el que los materiales y las tareas se renovaron y mejoraron aun más.

En los estudios E1 y R1, se utilizó como variable dependiente la *Efectividad de la Comprensibilidad*, definida como la capacidad de comprender el material presentado correctamente. En los estudios de E2, R2 y E3, hemos añadido dos nuevas variables relacionadas con la teoría CTML, la *Retención* y la *Transferencia*. La *Retención* se define como la memorización del material que se presenta; mientras que la *Transferencia* es la capacidad de utilizar los conocimientos adquiridos a través del material recibido para resolver problemas relacionados que no son pueden responder directamente. Estas tres variables se midieron a través de tareas diferentes basadas en cuestionarios. Los valores de la *Efectividad de la Comprensibilidad* (*CEfec*), la *Transferencia* (*CTrans*), y de la *Retención* (*CReten*) se calcularon como el número de respuestas correctas dividido el número de preguntas aplicados a cada cuestionario correspondiente.

También se midió el tiempo necesario para completar cada tarea, pero finalmente se decidió no usarlo, debido a que según opinión de los propios autores y de expertos, el tiempo en sí mismo no es un buen indicador de la comprensibilidad. El tiempo proporciona información sobre cuán rápido se realiza una tarea, pero no sobre lo bien que se ha realizado.

Teniendo en cuenta el tipo de diseños experimentales utilizados y el tratamiento de los estudios, se consideró utilizar para realizar las pruebas de las hipótesis el test denominado ANOVA (Análisis de la varianza), utilizando como umbral de significación estadística $\alpha = 0{,}05$. Por lo que se rechazaron las hipótesis nulas en el caso de que los test estadísticos den una significación estadística (p-valor) de los resultados que no sea superior a 0,05. También se estudió la potencia del test estadístico cuando se obtuvieron resultados estadísticamente no significativos. Todos los análisis estadísticos se realizaron con el paquete estadístico *SPSS* (SPSS, 2003).

3.7.2 Primer experimento y su réplica (E1 y R1)

Comenzaremos explicando los aspectos comunes a E1 y R1, y a continuación se explicarán los detalles específicos de cada uno. Una explicación más detallada de estos estudios empíricos se puede encontrar en Cruz-Lemus *et al.* (2005).

Antes de la ejecución del experimento, los sujetos recibieron una breve sesión de entrenamiento, en la que el investigador presentó los principales elementos de los diagramas de transición de estados UML y mostró dos ejemplos de las tareas experimentales a realizar. Estos ejemplos, así como los realizados en el resto de los experimentos y réplicas, eran neutrales con respecto a la variable independiente (el uso o no de estados compuestos) ya que un ejemplo contenía estados compuestos y el otro no.

Los sujetos se asignaron al azar a dos grupos, denominados Grupo A y Grupo B. Se utilizaron dos dominios, uno relacionado con el funcionamiento de un cajero automático y el otro de una llamada telefónica. Para cada dominio, se utilizaron dos diagramas conceptualmente idénticos, uno utilizando estados compuestos y el otro no.

En la primera parte del experimento, se utilizó el dominio del cajero automático, en la que los sujetos del Grupo A recibieron un diagrama sin estados compuestos, mientras que los sujetos en el Grupo B recibieron un diagrama con estados compuestos. En la segunda parte del experimento, se utilizó el dominio de la llamada de teléfono y los sujetos en el grupo A recibieron un diagrama con estados compuestos y los del Grupo B un diagrama sin estados compuestos. El diseño del experimento se resume en la Tabla 3.9.

Este proceso de asignación de los sujetos a los 4 tratamientos diferentes, obtenidos mediante la combinación de las variables independientes (Dominio y Estados Compuestos) corresponde a un diseño factorial 2x2 IASxIAS con confusión parcial (ver Figura 3.2), porque dentro de un Dominio, la variable Estados Compuestos cambia conjuntamente con el grupo de sujetos, y de esta manera se intenta paliar el efecto de aprendizaje. Efecto que también se intenta paliar, asignando a la mitad de los sujetos de cada grupo primero los diagramas con estado compuestos y luego los diagramas sin estados compuestos, y la otra mitad de sujetos los recibieron en orden inverso.

		Dominio	
		Cajero automático	Llamada telefónica
Estados Compuestos	Con EC	Grupo A	Grupo B
	Sin EC	Grupo B	Grupo A

Tabla 3.9. Diseño experimental de E1 y R1

A cada diagrama se le adjuntaron seis preguntas, que para cada dominio eran las mismas tanto para los diagramas con estados compuestos como para los sin estados compuestos. Estas preguntas tenían que ver con la navegación entre estados y los efectos que produce la navegación.

Para aumentar la motivación y el interés por parte de los sujetos, el profesor explicó a los alumnos que los ejercicios del experimento podrían ser similares a los que encontrarían en su examen final de la asignatura. Para no sesgar los resultados, no se dio a conocer el objetivo de este experimento ni el de los siguientes, antes de la realización de los mismos.

En E1 y R1 se consideró como variable dependiente la *Efectividad de la Comprensibilidad* (*CEfec*) medida como el número de respuestas correctas/número de preguntas.

3.7.2.1 PRIMER EXPERIMENTO (E1)

Los sujetos que participaron en este experimento eran alumnos de cuarto curso de Ingeniería informática, que habían cursado la asignatura de Ingeniería de Software en la que habían aprendido técnicas de modelado, incluyendo UML. Un resumen de las de características de este experimento se presentan en la Tabla 3.10 y los estadísticos descriptivos obtenidos en el análisis de datos se presentan en la Tabla 3.11.

Hipótesis nula	H_0: El uso de estados compuestos no mejora la Efectividad de la Comprensibilidad de un diagrama de transición de estados UML. H_1: $\neg H_0$
Lugar de realización	Universidad de Murcia (España)
Fecha	Febrero de 2005
Sujetos	55 estudiantes de la Ingeniería informática (28 en el Grupo y 27 en el Grupo B)
Variable dependiente	Efectividad de la Comprensibilidad medida a través de *CEfec*
Variables independientes	Estados compuestos (ConEC, SinEC), Dominio (cajero automático, llamada telefónica).

Tabla 3.10. Características de E1

Dominio	Cajero automático		Llamada telefónica	
EC	Con (n=27)	Sin (n=28)	Con (n=28)	Sin (n=27)
CEfec	0,9506 (0,0082)	0,9702 (0,0042)	0,9286 (0,0840)	0,9506 (0,1014)

Tabla 3.11. Media y desviación estándar (entre paréntesis) de la Efectividad de la Comprensibilidad para E1

Los resultados de la Tabla 3.11 muestran que los sujetos que trabajaron con diagramas de estados UML sin estados compuestos obtuvieron mejores valores para la *CEfec*. La prueba de hipótesis, que se presenta en la Tabla 3.12, se hizo con un ANOVA que es el test estadístico más apropiado considerando el diseño experimental seleccionado.

Fuente	df	F	p-valor	Potencia observada
Dominio	1	1,628	0,205	0,244
EC	1	1,628	0,205	0,244

Tabla 3.12. Resultado del ANOVA para E1

En la Tabla 3.12 y el resto de tablas de este apartado relacionadas con el ANOVA, se muestran los resultados del test estadístico denominado *test F de Fischer* donde la columna *Fuente* representa las variables independientes, *df* los grados de libertad, *F* es el valor del test estadístico, *p-valor* es el nivel de significación estadística obtenido, y *Potencia observada* es la potencia estimada del test basada en α=0,05. Observando los valores obtenidos no podemos rechazar la hipótesis nula H_0 (a nivel α=0,05), es decir no hay efecto del uso de estados compuestos. La potencia observada del test, es pequeña, probablemente por el valor pequeño del tamaño del efecto, por lo que podríamos asumir una probabilidad estimada de cometer un error de Tipo II en nuestras afirmaciones de 0,756 (1 - 0,244). Aunque los resultados no son concluyentes, parecen indicar que no existe un impacto apreciable del uso de estados compuestos en la *Efectividad de la Comprensibilidad* de los diagramas de estado.

3.7.2.2 RÉPLICA DEL PRIMER EXPERIMENTO (R1)

Los sujetos que participaron en esta réplica eran alumnos del segundo curso de ingeniería informática y no estaban muy familiarizados con el modelado con UML, dado que en el momento que se realizó el experimento estaban cursando el primer curso de ingeniería de *software*. En la Tabla 3.13 se presentan las principales diferencias entre esta réplica y el experimento original (E1 y R1 respectivamente).

Lugar de realización	Universidad de Alicante (España)
Fecha	Marzo 2005
Sujetos	178 estudiantes de ingeniería informática (89 en cada grupo)

Tabla 3.13. Principales diferencias entre R1 y E1

El nivel de experiencia de los sujetos en el modelado con UML era mucho menor en R1 que en E1, debido a que la mayoría tenía sólo unos pocos meses de experiencia, y no habían trabajado con estados compuestos aún. El conocimiento que tenían sobre los estados compuestos lo adquirieron durante la sesión de entrenamiento realizada antes de la ejecución del experimento. El análisis de datos, cuyos resultados se muestran en las Tablas 3.13 y 3.14, se realizó utilizando los mismos test estadísticos que en el experimento original.

Según muestra la Tabla 3.14 la mayoría de los sujetos realizaron las tareas correctamente cuando recibieron diagramas sin estados compuestos y sobre el dominio del cajero automático. Análogamente con E1, los sujetos obtuvieron mejores valores de *CEfec* cuando trabajaron con diagramas sin estados compuestos.

Dominio	**Cajero automático**		**Llamada telefónica**	
EC	Con (n=89)	Sin (n=89)	Con (n=89)	Sin (n=89)
CEfec	0,9663 (0,0719)	0,9625 (0,1028)	0,8978 (0,1188)	0,9491 (0,0991)

Tabla 3.14. Media y desviación estándar (entre paréntesis) de la Efectividad de la Comprensibilidad para E1

En la Tabla 3.15 se muestran los resultados del ANOVA realizado para los datos obtenidos en R1.

Fuente	**df**	**F**	**p-valor**	**Potencia observada**
Dominio	1	1,873	0,183	0,260
EC	1	2,340	0,139	0,313

Tabla 3.15. Resultado del ANOVA para R1

Se puede observar que no existe un efecto del dominio ni del uso de estados compuestos y también la potencia es baja. Si se rechaza la hipótesis nula, se estaría asumiendo una probabilidad estimada de 0,687 de cometer un error de Tipo II. Al igual que en E1, los resultados no son concluyentes aunque parecen

indicar que existe un impacto apreciable del uso de estados compuestos en la *Efectividad de la Comprensibilidad* de los diagramas de estados.

3.7.2.3 CONCLUSIONES DE E1 Y R1

El principal objetivo de E1 y R1 fue estudiar el efecto del uso de estados compuestos en los diagramas de transición de estados UML en la *Efectividad de la Comprensibilidad*. Teniendo en cuenta los resultados obtenidos, no se puede concluir algo definitivo, debido a que los resultados no son estadísticamente significativos y las potencias observadas de los test fueron bajas. Sin embargo, se pudo observar que aparentemente el uso de estados compuestos no hace que los sujetos realicen las tareas de comprensión sobre los diagramas de transición de estados UML, más correctamente.

3.7.3 Segundo experimento y su réplica (E2 y R2)

Observando los resultados obtenidos en E1 y R1, se revisaron el diseño experimental, las tareas y el material entregado a los sujetos. Leyendo otras investigaciones realizadas con experimentos en el campo del modelado conceptual (Bodart *et al.*, 2001; Gemino y Wand, 2005), se decidió mejorar las tareas experimentales con el objetivo de capturar mejor la comprensión que los sujetos tuvieron sobre los diagramas. Por ello se decidió llevar a cabo otra serie de experimentos, incorporando, como se comentó previamente, el uso de la teoría CTML. Se llevó a cabo un segundo experimento (E2) y su posterior réplica (R2), en los que además de considerar a *CEfec* se consideraron dos nuevas variables tomadas de la teoría CTML: *CTrans* and *CReten*.

Tanto en E2 como en R2, los sujetos recibieron una sesión de entrenamiento antes de llevar a cabo el experimento, sobre el uso de los elementos principales de los diagramas de transición de estados UML, dado que muchos de los sujetos no habían utilizado diagramas de estados desde hacia tiempo. También en esta sesión de entrenamiento, se presentaron algunos ejemplos de las tareas que iban a tener que realizar en el experimento, para que los sujetos tuvieran claro qué tareas iban a tener que realizar. En esta sesión los sujetos podían preguntar al investigador sus dudas y expresar todas las sugerencias y comentarios que creyeran oportuno.

Como en E1 y R1 se usaron cuatro diagramas relacionados con dos dominios diferentes, aunque se cambió la llamada telefónica por el reloj despertador.

El diseño experimental fue idéntico al de E1 y R1. Para cada dominio se utilizaron dos diagramas con idéntico contenido semántico, uno con estados compuesto y el otro no. Cada sujeto recibió dos diagramas, uno con estados compuestos y el otro no, relacionados ambos con diferentes dominios. Así se obtuvieron dos grupos como muestra en la Tabla 3.16.

		Dominio	
		Cajero automático	**Reloj despertador**
Estados compuestos	**SinEC**	Grupo A	Grupo B
	ConEC	Grupo B	Grupo A

Tabla 3.16. Diseño experimental de E2 y R2

Cada sujeto tenía que realizar tres tareas, cada una relacionada con una de las variables dependientes seleccionadas.

- **Tarea 1**. Los sujetos tenían que completar el cuestionario 1, que consistía en 7 preguntas que eran las mismas dentro de cada dominio, independientemente al uso o no de estados compuestos. Las preguntas se referían a la navegación entre estados, el valor de las variables, etc. En esta tarea los sujetos podían consultar el diagrama para contestar las preguntas. A través de estas tareas se estudió el efecto sobre la variable *CEfec*.
- **Tarea 2**. Los sujetos tenían que completar el cuestionario 2, que consistía en 5 preguntas, en las que se les preguntó a los sujetos cómo funcionaba el modelo, es decir preguntas que eran más específicas que en el cuestionario previo. En este caso no se les permitió mirar el diagrama para contestar el cuestionario. Lo podían mirar previamente, pero se les retiró antes de tener que resolver esta tarea. A través de esta tarea se midió la variable *CTrans*.
- **Tarea 3**. En esta tarea del tipo "llenar los espacios en blanco", los sujetos recibieron un texto que representaba los requisitos del sistema modelado, pero contenía espacios en blanco que debían ser completados sin utilizar el diagrama. Los diagramas se les quitaron a los sujetos antes de tener que completar la tarea 2 y no se les volvieron a entregar. A través de esta tarea se estudió el efecto sobre la variable *CReten*. Estos dos tipos de pruebas son similares a las usadas en otros estudios como (Gemino y Wand 2005; Khatri *et al.*, 2006) que tratan sobre la comprensión de modelos usando CTML.

Los sujetos se asignaron al azar a cada uno de los dos grupos, recibiendo en primer lugar el cuestionario 1 y el diagrama correspondiente al grupo al que fueron asignados. Después de 20 minutos los sujetos debían entregar al instructor el cuestionario 1 completo, y a continuación recibieron el cuestionario 2 y 3 y tenían 20 minutos para completar ambos cuestionarios.

Una vez realizadas ambas tareas se recogió el material, y llevo a cabo el mismo proceso, en este caso con el segundo diagrama, que estaba relacionado con diferente dominio (cajero automático/reloj despertador) que el primero y también con diferente uso de los estados compuestos (con/sin). El procedimiento llevado a cabo en la ejecución de E2 y R2 se presenta en la Figura 3.11.

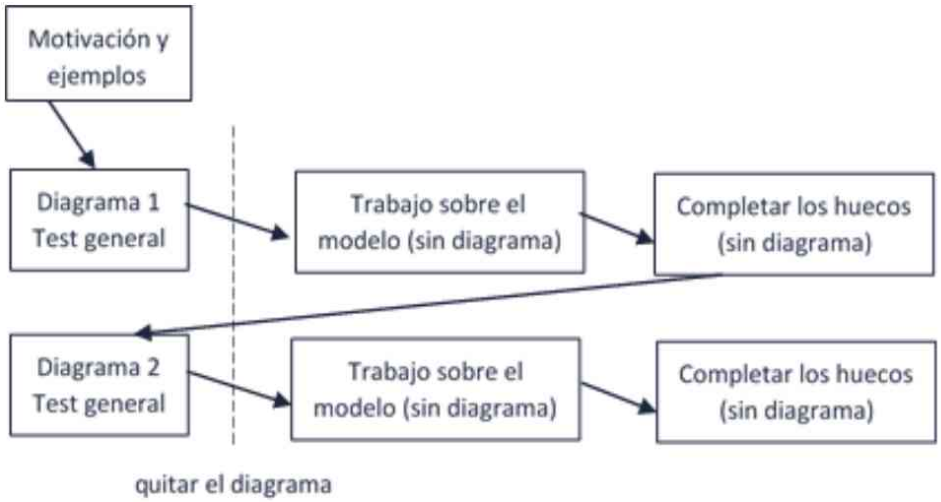


Figura 3.11. Procedimiento de E2 y R2

A continuación se presentan los detalles específicos de E2 y R2.

3.7.3.1 SEGUNDO EXPERIMENTO (E2)

Las principales características de E2 se presentan en la Tabla 3.17.

Hipótesis nulas	H_{0a}: el uso de estados compuestos no mejora *CEfec* cuando los sujetos tratan de entender los diagramas de transición de estados UML. H_{1a}: $\neg H_{0a}$ H_{0b}: el uso de estados compuestos no mejora *CTtrans* cuando los sujetos tratan de entender los diagramas de transición de estados UML. H_{1b}: $\neg H_{0b}$ H_{0c}: el uso de estados compuestos no mejora *CReten* cuando los sujetos tratan de entender los diagramas de transición de estados UML. H_{1c}: $\neg H_{0c}$
Lugar de realización	Universidad de Castilla-La Mancha en Ciudad Real, España.

Fecha	Febrero de 2006
Sujetos	14 estudiantes de doctorado (7 en cada grupo)
Variable dependiente	La comprensibilidad de los diagramas de transición de estados UML, medida a través de las medidas *CEfec*, *CTrans* y *CReten*.
Variables independientes	Estados Compuestos (ConEC, SinEC) y Dominio(Cajero automático, reloj despertador)

Tabla 3.17. Características de E2

Los estadísticos descriptivos se muestran en la Tabla 3.18, concretamente se muestran la media y la desviación estándar para cada grupo.

Dominio	**Cajero automático**		**Reloj despertador**	
EC	Con (n=7)	Sin (n=7)	Con (n=7)	Sin (n=7)
CEfec	0,6837 (0,0761)	0,7959 (0,0687)	0,7857 (0,0643)	0,8673 (0,0453)
CTrans	0,6429 (0,0684)	0,6349 (0,0898)	0,6905 (0,0765)	0,5774 (0,0683)
CReten	0,6071 (0,0798)	0,6143 (0,0643)	0,5714 (0,1359)	0,4444(0,1118)

Tabla 3.18. Media y desviación estándar (entre paréntesis) para E2

Los valores obtenidos para *CEfec* son menores que en el estudio previo (E1 y R1). Una posible explicación para esto es que, el material había sido modificado incluyendo más preguntas y más difíciles de responder.

Existe una tendencia clara en *CEfec*, los sujetos obtienen mejores resultados en los diagramas sin estados compuestos, independientemente del dominio utilizado. No obstante, para la variable *CTrans* la situación es opuesta, se obtuvieron mejores resultados en los diagramas con estados compuestos. Y para la variable *CReten*, los resultados fueron diferentes, dependiendo del dominio.

Los resultados de aplicar un ANOVA para hacer las pruebas de las hipótesis se muestran en la Tabla 3.19.

		CEfec			***CTrans***			***CReten***		
Fuente	**df**	**F**	**p-valor**	**Potencia observada**	**F**	**p-valor**	**Potencia observada**	**F**	**p-valor**	**Potencia observada**
Dominio	1	1,873	0,183	0,260	0,004	0,948	0,050	1,043	0,317	0,166
EC	1	2,340	0,139	0,313	0,643	0,430	0,120	0,354	0,557	0,088

Tabla 3.19. Resultado del ANOVA para E2

En todos los casos (*CEfec*, *CTrans* y *CReten*), las variables no se vieron afectadas por el dominio o el uso de estados compuestos. Las potencias observadas de los test estadísticos fueron bajas, por ello la posibilidad de cometer un error aceptando la hipótesis nula es alta. Por ello, los resultados no son concluyentes.

En E2, los resultados obtenidos para las tres variables coinciden con los obtenidos en E1 y R1 para *CEfec*.

3.7.3.2 RÉPLICA DEL SEGUNDO EXPERIMENTO (R2)

Las principales diferencias entre R2 y E2 se detallan en la Tabla 3.20.

Lugar de realización	Universidad de La Insubria (Como, Italia)
Fecha	Mayo de 2006
Sujetos	13 estudiantes de la ingeniería informática (6 o 7 en cada grupo)

Tabla 3.20. Características de R2

En la Tabla 3.21, se presentan la media y la desviación estándar obtenidas para cada grupo en R2.

Dominio	**Cajero automático**		**Reloj despertador**	
EC	**Con (n=7)**	**Sin (n=6)**	**Con (n=6)**	**Sin (n=7)**
CEfec	0,9082(0,0895)	0,9167 (0,1311)	0,9476 (0,1075)	0,8673 (0,01197)
CTrans	0,5556 (0,2128)	0,7407 (0,0907)	0,6667 (0,1900)	0,7262 (0,2136)
CReten	0,5571 (0,1272)	0,6167 (0,1329)	0,6296 (0,2781)	0,5873 (0,2775)

Tabla 3.21. Media y desviación estándar (entre paréntesis) para R2

En esta réplica, la única variable que muestra una clara tendencia en favor de los diagramas sin estados compuestos, independientemente del dominio, es *CTrans*. En el caso de las otras dos variables, los sujetos obtuvieron diferentes resultados dependiendo del dominio y del uso de estados compuestos.

Comparando los resultados con los obtenidos en E2, se puede observar un incremento en las medias, hecho que se podría explicar por el tamaño pequeño de las muestras en cada ambos estudios, además del hecho de que no fueron aleatorios. Esto podría probablemente indicar que podríamos encontrar dos grupos con diferentes conocimientos en la población seleccionada.

El test de la hipótesis se hizo como en los otros experimentos mediante un ANOVA, cuyos resultados se presentan en la Tabla 3.22.

		CEfec			*CTrans*			*CReten*		
Fuente	**df**	**F**	**p-valor**	**Potencia observada**	**F**	**p-valor**	**Potencia observada**	**F**	**p-valor**	**Potencia observada**
Dominio	**1**	0,367	0,550	0,089	0,437	0,515	0,097	0,066	0,800	0,057
EC	**1**	0,110	0,743	0,062	2,807	0,107	0,362	0,010	0,919	0,051

Tabla 3.22. Resultado de t-test en R2

Otra vez, en todos los casos (*CEfec*, *CTrans* y *CReten*) los resultados demuestran que no se ven afectados ni por el dominio ni por el uso de estados compuestos. También en este caso la potencia observada es baja, por ello la posibilidad de cometer un error al aceptar la hipótesis nula es alta, lo que hace que no podamos considerar los resultados concluyentes.

3.7.3.3 CONCLUSIONES DE E2 Y R2

Considerando los resultados obtenidos, no podemos tampoco extraer conclusiones definitivas, porque las pruebas de las hipótesis no son estadísticamente significativas y las potencias observadas son bajas. No obstante parece haber una clara tendencia, indicando que el uso de estados compuestos no parece mejorar significativamente *CTrans*.

3.7.4 Tercer experimento (E3)

En este experimento, nuevamente se revisó y mejoró el material y las tareas experimentales. Y además se contó con la participación de profesionales para la realización de este experimento.

El experimento se llevó a cabo con profesionales que trabajan en la factoría de *software* de Indra ubicada en Ciudad Real, España.

3.7.4.1 DISEÑO DE E3

La Tabla 3.23 resume las principales características de E3.

Hipótesis nula	H_{0a}: el uso de estados compuestos no mejora *CEfec* cuando los sujetos tratan de entender los diagramas de transición de estados UML. H_{1a}: $\neg H_{0a}$ H_{0b}: el uso de estados compuestos no mejora *CTrans* cuando los sujetos tratan de entender los diagramas de transición de estados UML. H_{1b}: $\neg H_{0b}$ H_{0c}: el uso de estados compuestos no mejora *CReten* cuando los sujetos tratan de entender los diagramas de transición de estados UML. H_{1c}: $\neg H_{0c}$
Lugar de realización	Ciudad Real (España).
Fecha	Octubre 2006.
Sujetos	24 profesionales (12 en cada grupo).
Variable dependiente	La comprensibilidad de lo diagramas de transición de estados UML, medida a través de: *CEfec*, *CTrans* y *CReten*.
Variables independientes	Estados compuestos (SinEC, ConEC).

Tabla 3.23. Características de E3

En este experimento se consideró un único dominio (diseño inter-sujetos) reloj digital, que tiene un tamaño y complejidad mayor al de los dominios considerados en los estudios previos. En este caso también se formaron dos grupos al azar pero intentando crear grupo equilibrados en cuanto al factor de la experiencia, como se detallará más adelante. La hipótesis y el procedimiento experimental fueron similares a E2 y R2. A continuación explicaremos en más detalles el procedimiento experimental y los resultados.

3.7.4.2 PROCEDIMIENTO DE E3

El experimento se dividió en dos sesiones, durante dos días. La primera sesión, de aproximadamente dos horas, tuvo lugar en la tarde del primer día y la segunda sesión en la mañana siguiente. Para que los sujetos tuvieran un conocimiento previo lo más homogéneo posible, la primera sesión se inició con un seminario sobre "Modelado dinámico con UML". Veinticinco profesionales asistieron a la primera sesión y se les proporcionó un resumen de los principales conceptos de los aspectos dinámicos en el modelado en general, y en UML en particular. La última parte del seminario se centró en los diagramas de transición de estados UML, aunque no se hizo mención a ningún aspecto que pudiera desvelar la relación entre este seminario previo, y la sesión de ejecución del experimento.

Después del seminario, el investigador mostró varios ejemplos de diagramas de transición de estados UML con sus respectivos cuestionarios, cuyo objetivo era que los alumnos entendiesen la semántica de los diagramas. A continuación se les entregó un diagrama de transición de estados con un cuestionario pre-experimento que denominados "Cuestionario 0", similar al utilizado en la tarea 1, de E2 y R2. Los resultados obtenidos tras la corrección de este cuestionario se utilizaron para formar dos grupos balanceados y equilibrados con respecto al nivel de conocimiento sobre los diagramas de transición de estado UML. El procedimiento seguido para formar los grupos se comenta más adelante.

Los sujetos también completaron un cuestionario subjetivo y anónimo en la que se incluyeron algunos datos personales (edad, sexo, etc.) y su experiencia en el modelado, programación orientada a objetos, uso de UML, etc. Estos datos indicaron que aunque la mayoría de ellos habían desarrollado un *software* orientado a objetos, sólo la mitad de los sujetos había utilizado anteriormente UML en proyectos reales, y esto sólo una o dos veces. La media de experiencia en el desarrollo orientado a objetos era de dos años.

Una vez que finalizó la primera sesión se corrigió el Cuestionario 0 y de acuerdo a los resultados obtenidos se asignaron los sujetos a cada uno de los dos grupos, intentando equilibrar los grupos de acuerdo a los resultados obtenidos, de la siguiente manera: se ordenaron los sujetos de acuerdo al número de respuestas correctas y al tiempo utilizado para responder, y los que ocupaban una posición impar fueron asignados al Grupo A y los restantes al Grupo B. Así se obtuvieron dos grupos balanceados como se muestra en la Tabla 3.24.

Grupo	N	Preguntas correctas		Tiempo	
		Media	Desviación estándar	Media	Desviación estándar
A	13	5,3846	1,3409	628,62	122,88
B	12	5,4583	1,0967	620,67	135,18

Tabla 3.24. Media y desviación estándar para cada grupo obtenida en el Cuestionario 0

La mañana siguiente tuvo lugar la ejecución del experimento, que inició con la formación de los grupos, y a continuación comenzó la ejecución del experimento de manera similar a E2 y R2. Un alumno del Grupo A no asistió, por lo que ambos grupos quedaron con 12 sujetos. En primer lugar los alumnos fueron asignados a cada grupo.

A continuación, los sujetos recibieron un diagrama de estados UML y el Cuestionario 1, que era exactamente el mismo para el diagrama con estados compuestos y sin estados compuestos. Para evitar los posibles efectos de aprendizaje, se optó por un diseño inter-sujetos, es decir, cada sujeto se le asignó un sólo diagrama. Los sujetos del Grupo A recibieron un diagrama modelado utilizando estados compuestos y los del Grupo B un diagrama sin estados compuestos.

Al igual que en E2 y R2, se utilizó este cuestionario para medir la *CEfec*. Las preguntas del Cuestionario 1 cubrían todas las diferentes partes del diagrama para que podamos asegurarnos de que todas las partes del diagrama habían sido leídas por los sujetos antes de que les retirasen el diagrama. Esta tarea duró 25 minutos.

Cuando se completó esta tarea, se recogieron todos los diagramas y los cuestionarios y se les entregaron los Cuestionarios 2 y 3. El Cuestionario 2 se utilizó para medir la variable *CReten* y consistió en un texto con 10 espacios en blanco para completar el documento con la especificación del modelo del reloj digital. Los sujetos tuvieron 15 minutos para esta tarea.

El Cuestionario 3 se utilizó para medir la variable *CTrans* y consistió en 6 tareas basadas en información que no se puede extraer directamente del diagrama, sino que se puede inferir a partir de la misma. Como esta tarea era la más complicada, los sujetos tuvieron 35 minutos para resolverla.

Una vez que finalizaron el Cuestionario 3, se recogieron todos los materiales y se repartió un cuestionario para recoger sus impresiones acerca de la dificultad de los cuestionarios y de los puntos positivos y negativos que se habían encontrado durante la realización del experimento.

La Figura 3.12 resume el procedimiento seguido para la ejecución de E3.

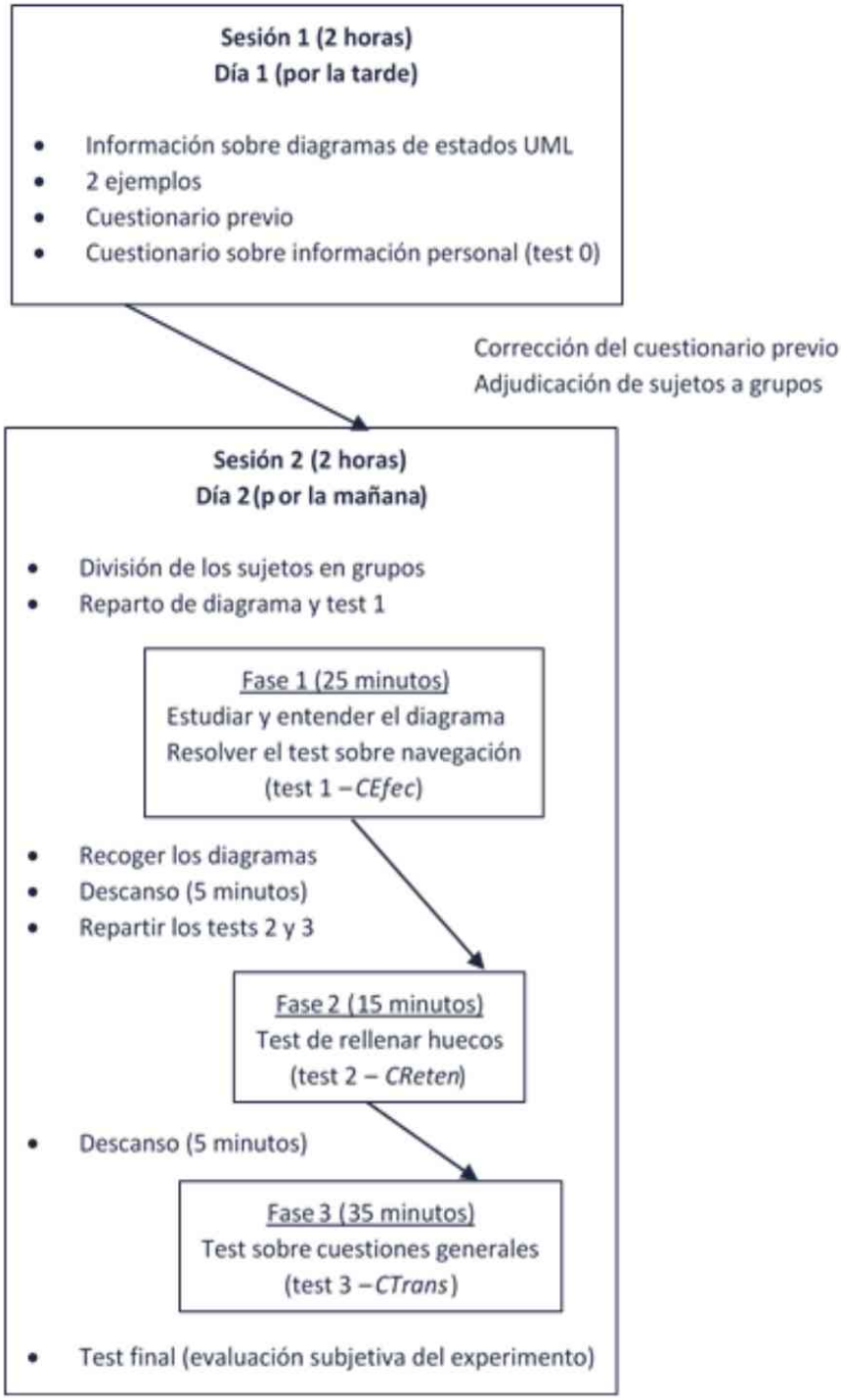


Figura 3.12. Procedimiento de E3

3.7.4.3 ANÁLISIS E INTERPRETACIÓN DE DATOS PARA E3

Como en los experimentos previos, comenzamos mostrando los estadísticos descriptivos en la Tabla 3.25, que indican que los resultados obtenidos para *CEfec* y *CTrans* son mejores cuando los sujetos trabajaron con los diagramas con estados compuestos, mientras para *CReten* se obtuvieron mejores valores en el caso de diagramas sin estados compuestos.

EC	**Con (n=12)**	**Sin (n=12)**
CEfec	0,7500 (0,1446)	0,6417 (0,1564)
CTrans	0,3690 (0,2079)	0,2619 (0,1677)
CReten	0,7750 (0,1390)	0,8917 (0,0515)

Tabla 3.25. Media y desviación estándar (entre paréntesis) para E3

A continuación para hacer las pruebas de hipótesis se utilizó el t-test, que produce los mismos resultados que un ANOVA cuando se comparan dos grupos (ver Tabla 3.26).

		CEfec			***CTrans***			***CReten***		
Fuente	**df**	**F**	**p-valor**	**Potencia observada**	**F**	**p-valor**	**Potencia observada**	**F**	**p-valor**	**Potencia observada**
EC	1	257,628	0,000	1,000	34,442	0,000	1,000	762,338	0,000	1,000

Tabla 3.26. Resultado del t-test para E3

La Tabla 3.26 muestra que existe un efecto estadísticamente significativo para las tres variables, *CEfec*, *CTrans* y *CReten*. El uso de estados compuestos mejora la *CEfec* y *CTrans*, y empeora la *CReten*. Esto significa que los estados compuestos son útiles para una mejor comprensión del diagrama (*CEfec*) y para llevar a cabo tareas relacionadas con el diagrama (pero que no se pueden responder directamente a partir del mismo diagrama), pero que no son útiles para memorizar el diagrama (*CReten*).

Estos resultados contrastan con los obtenidos en los experimentos presentados anteriormente. En este caso, una vez más la intención era evaluar cómo el uso de estados compuestos afecta la comprensibilidad de los diagramas de transición de estados UML, pero usando profesionales de la industria del *software* en lugar de estudiantes y también se aumentó la complejidad de las tareas a

realizar. Muy probablemente, estos dos factores pueden haber afectado a los resultados obtenidos. Además de esto, se debe recordar que se utilizó las habilidades de los sujetos para equilibrar su distribución en grupos.

3.7.5 Amenazas a la validez de la familia de experimentos

En este apartado, se explican algunas cuestiones que pueden poner en peligro la validez de los experimentos, teniendo en cuenta los cuatro tipos de amenazas que hemos visto en este capítulo:

3.7.5.1 VALIDEZ DE LAS CONCLUSIONES

En E1, R1, E2 y R2, el poder estadístico fue bajo, hecho que no nos permite rechazar hipótesis erróneas sin un alto grado de incertidumbre.

3.7.5.2 VALIDEZ INTERNA

El número de sujetos implicados no era grande. Sin embargo, se identificó una tendencia clara en un sólo caso, en el que los participantes eran profesionales. Esto nos lleva a interpretar que los estados compuestos parecen ser un elemento que requiere un cierto nivel de madurez o experiencia para ser utilizado correctamente. Creemos que los estudiantes probablemente no habían adquirido esa madurez, mientras que los profesionales sí. Este puede haber sido un factor determinante en los resultados que aquí se presentan.

3.7.5.3 VALIDEZ DE CONSTRUCTO

Las medidas se construyeron sobre la base de las directrices establecidas en CTML, y por ello se cree que se han medido de manera apropiada.

3.7.5.4 VALIDEZ EXTERNA

Los diagramas que se utilizaron en este estudio, representan modelos relativamente simples y es muy posible que si se hubieran utilizado diagramas representativos, de los que se usan en la industria, se podrían haber obtenido resultados diferentes. Estos resultados son válidos para diagramas y sujetos con características similares a los usados en esta familia de experimentos. Para poder generalizar los resultados similares a toda la población de diseñadores que usan diagramas de transición de estados es necesario realizar más estudios empíricos.

3.7.6 Estudio de meta-análisis

La Tabla 3.27 resume los resultados del ANOVA realizado para estudiar el efecto del dominio y del uso de estados compuestos sobre tres las variables dependientes que miden la comprensibilidad de los diagramas de transición de estados UML. En esta tabla la columna "*p*" indica el p-valor y la columna "*po*" la potencia observada.

	ANOVA *CEfec*				**ANOVA *CTrans***				**ANOVA *CReten***			
	Dominio		**EC**		**Dominio**		**EC**		**Dominio**		**EC**	
	p	**po**	**p**	**po**	**p**	**p**	**p**	**po**	**p**	**po**	**p**	**po**
E1	0,205	0,244	0,205	0,244	---	---	---	---	---	---	---	---
R1	0,183	0,260	0,139	0,313	---	---	---	---	---	---	---	---
E2	0,183	0,260	0,139	0,313	0,948	0,050	0,430	0,120	0,317	0,166	0,557	0,088
R2	0,550	0,089	0,743	0,062	0,515	0,097	0,107	0,362	0,800	0,057	0,919	0,051
E3	---	---	**0:000**	1,000	---	---	**0:000**	1,000	---	---	**0:000**	1,000

Tabla 3.27. Resumen del ANOVA (*en negrita figuran los resultados significativos*)

Sólo en el experimento E3 se obtuvieron resultados significativos, que indican que los estados compuestos mejoran comprensión de cómo funciona el diagrama de transición de estados (*CEfec*) y también mejoran el rendimiento de las tareas relacionadas con la adquisición de conocimiento a partir del diagrama (*CTrans*), pero los estados compuestos no mejoran la memorización del diagrama (*CReten*).

Los resultados obtenidos tras el análisis individual de cada experimento no son concluyentes, por ello se decidió integrarlos, a través de un estudio de meta-análisis, llevado a cabo con la herramienta *Comprehensive Meta-Analysis* (Biostat, 2006).

Como se comentó en el apartado 3.6, el meta-análisis es un conjunto de técnicas estadísticas para la combinación de los diferentes tamaños del efecto de los experimentos individuales para obtener un efecto global de una variable independiente. Como las medidas pueden provenir de diferentes entornos y no ser homogéneas, se debe obtener una medida estandarizada para cada estudio, y estas medidas se deben combinar para obtener el tamaño del efecto global. En nuestro estudio, la variable independiente es el uso de estados compuestos y nos interesa conocer cómo afecta a la comprensibilidad de los diagramas de transición de estados UML.

En este meta-análisis se utilizó el valor medio de EC(con) menos el valor medio de EC(sin), y a partir de estos valores se obtuvieron la medida *g de Hedges* (Hedges y Olkin 1985; Kampenes *et al.*, 2007; Ellis, 2010), que es la medida que se consideró como estandarizada y se puede utilizar para sintetizar los estudios que presentan los efectos del tratamiento en diferentes escalas. Este valor expresa la magnitud de los efectos del tratamiento, EC, en nuestro caso, en relación con las desviaciones estándar dentro de los grupos.

La medida *g de Hedges* es una media ponderada cuyos pesos dependen del tamaño de la muestra, como muestra la siguiente ecuación:

$$\overline{Z} = \frac{\sum_i w_i z_i}{\sum_i w_i}$$

donde $w_i = 1/(n_i-3)$ y n_i es el tamaño de la muestra del i-ésimo experimento.

Cuanto mayor sea el valor de la *g de Hedges* es, mayor será las correspondientes diferencias de medias. En ingeniería de *software*, se puede clasificar el tamaño del efecto en tres categorías: pequeño, mediano y grande (Kampenes *et al.*, 2007).

Además una vez que se calcula el tamaño del efecto global, se puede proporcionar otros dos valores, el intervalo de confianza y el p-valor, que nos permitirán decidir a cerca de la hipótesis del meta-análisis, que en nuestro caso serían las siguientes:

- **H_{0a}**: el uso de estados compuestos (EC) no tiene influencia sobre *CEfec*.
  H_{1a}: $\neg H_{0a}$
- **H_{0b}**: el uso de estados compuestos (EC) no tiene influencia sobre *CReten*.
  H_{1b}: $\neg H_{0b}$
- **H_{0c}**: el uso de estados compuestos no tiene influencia sobre *CTrans*.
  H_{1c}: $\neg H_{0c}$

La Tabla 3.28 resume los resultados obtenidos en el meta-análisis, mostrando para cada estudio y dominio los valores de la g de Hedges y del tamaño del efecto. Concretamente las celdas relacionadas con el tamaño del efecto muestran:

- El valor del tamaño del efecto, clasificado como pequeña, mediano y grande. Este valor se basa en la diferencia estandarizada entre dos medias. Por ejemplo, un tamaño del efecto de 0,5 indica que la media de EC(con) es la mitad de la desviación estándar mayor que la media de EC(sin). Considerando la variable *CEfec*, un efecto positivo significa que el uso de estados compuesto mejora la *Efectividad de la Comprensibilidad*, mientras que un efecto negativo significa lo contrario. Por ejemplo, hay un efecto negativo de *CEfec* en E1 en el dominio del cajero automático, como muestra el valor negativo de la g de Hedges, pero hay un tamaño del efecto positivo en R1 para el mismo dominio. Para estudios en ingeniería de *software*, se puede considerar, que tamaños del efecto entre 1,01 y 3,40 son grandes, entre 0,38 y 1,00 son medianos y aquellos entre 0 y 0,37 son pequeños (Kampenes *et al.*, 2007).
- Una indicación sobre si el resultado es estadísticamente significativo (S) o no (NS). Se puede observar que el tamaño del efecto global es únicamente significativo para *CEfec*.

Estudio	**Dominio**	***CEfec***		***CTrans***		***CReten***	
		g	**Tamaño del efecto**	**g**	**Tamaño del efecto**	**g**	**Tamaño del efecto**
E1	Cajero automático	-2,986	Grande S	No aplicable		No aplicable	
	Llamada telefónica	-0,233	Pequeño NS				
R1	Cajero automático	0,042	Pequeño NS				
	Llamada telefónica	-0,467	Mediano S				
E2	Reloj despertador	-1,345	Grande S	1,460	Grande S	0,955	Mediano NS
	Cajero automático	-1,450	Grande S	0,093	Pequeño NS	-0,092	Pequeño NS
R2	Reloj despertador	1,023	Grande S	-0,272	Pequeño NS	0,142	Pequeño NS
	Cajero automático	-0,072	Pequeño NS	-1,021	Grande NS	-0,427	Mediano NS
E3	Reloj digital	0,695	Mediano NS	0,548	Mediano NS	-1,075	Grande S
Tamaño del efecto global		-0,333	Pequeño S	0,194	Pequeño NS	-0,193	Pequeño NS

Tabla 3.28. Resultados del meta-análisis

Además se ilustran los resultados del meta-análisis gráficamente en las Figuras 3.13, 3.14 y 3.15, que muestran la medida g de Hedges con un intervalo de confianza de 95%, para cada variable dependiente (*CEfec*, *CTrans* y *CReten* respectivamente). Las figuras se muestran en inglés puesto que la herramienta utilizada para realizar el meta-análisis no soporta otro leguaje.

No todos los estudios contribuyen igualmente a la conclusión general, que está representada por el diamante en la última fila de las figuras. Cada uno de los estudios recibe un peso específico en el meta-análisis, es decir, el tamaño del efecto del estudio, representado en la figuras a través de un cuadrado. Las estimaciones de los estudios son más precisos cuanto más grande sea el tamaño de la muestra, y por ello contribuyen en mayor medida en el tamaño del efecto global. Sin embargo, el tamaño de la muestra no es el único factor que contribuye al peso de un estudio. El peso de un estudio es proporcional al área del cuadrado correspondiente en las figuras.

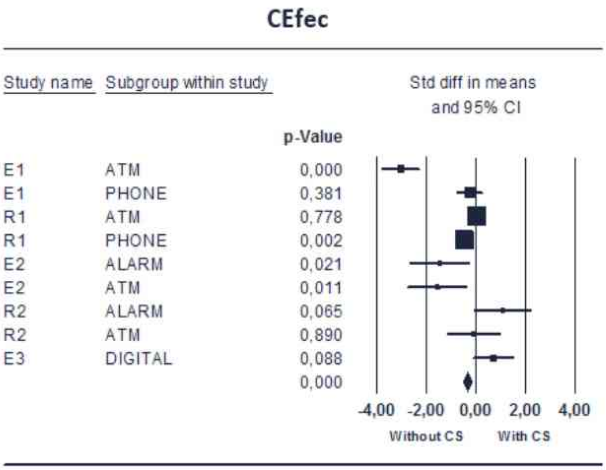


Figura 3.13. Meta-análisis para CEfec

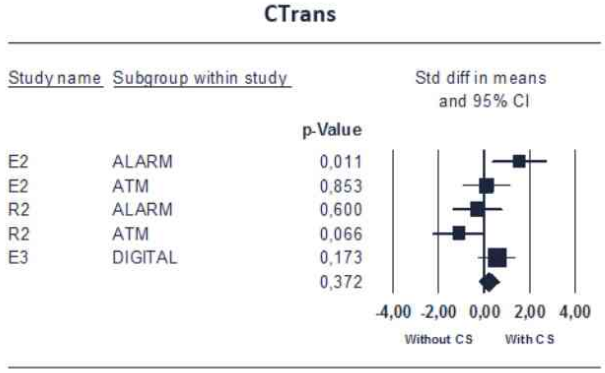


Figura 3.14. Meta-análisis para CTrans

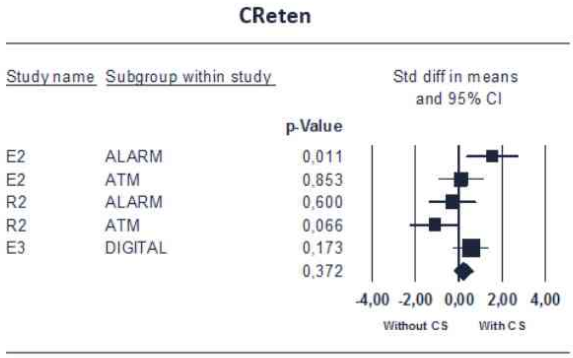


Figura 3.15. Meta-análisis para CReten

Además, se realizó un nuevo meta-análisis, excluyendo E3, por dos motivos. En primer lugar, la descripción y amenazas a la validez de E3, hacen que sea un estudio bastante diferente a los demás, ya que se usaron material y tareas más difíciles y los participantes fueron profesionales. Por otra parte, el error observado en los tres meta-análisis fue el más alto en E3. En la Tabla 3.29, que muestra los resultados de este segundo meta-análisis, se puede observar cómo se han modificado los valores de la *g de Hedges*.

	CEfec			***CTrans***			***CReten***		
	g	**Tamaño del efecto**	**p-valor**	**g**	**Tamaño del efecto**	**p-valor**	**g**	**Tamaño del efecto**	**p-valor**
CS Excluyendo E3	-0,383	Mediano	0,000	0,037	Pequeño	0,890	0,137	Pequeño	0,597
CS Incluyendo E3	-0,333	Mediano	0,000	0,194	Pequeño	0,384	0,193	Pequeño	0,382

Tabla 3.16. Resultado del meta-análisis excluyendo E3

No obstante, las conclusiones con respecto a los p-valores son similares a las obtenidas en el estudio de meta-análisis anterior, que son las siguientes:

- El uso de estados compuesto hace que la *Efectividad de la Comprensibilidad* (*CEfec*) disminuya su valor, con un tamaño del efecto mediano (-0,383), es decir, la media de *CEfec* cuando no se usan estados compuestos es 0,383 veces la desviación estándar que cuando se usan estados compuestos y el p-valor es 0,000. Este tamaño del efecto es menor si tenemos en cuenta todos los experimentos, ya que en E3 los estados compuestos mejoran la comprensibilidad.

- El uso de estados compuestos no influye en la realización de tareas relacionadas con el diagrama (*CTrans* p-valor = 0,890). Cuando incluimos E3, el tamaño del efecto y el p-valor están más a favor de la idea de que los estados compuestos ayudan a mejorar la *Transferencia*, pero esto no es suficiente para sea estadísticamente significativa.
- Por último, el uso de estados compuestos no tiene ninguna influencia en la memorización de los diagramas (*CReten* p-valor = 0,597). En este caso, cuando se incluye E3 el tamaño del efecto y el p-valor se inclinan para indicar que los estados compuestos tienen influencia, pero esto no es estadísticamente significativo.

Para concluir, los principales hallazgos de esta familia de experimentos, realizada en el contexto de diagramas de transición de estados UML relativamente pequeños (entre 10 y 25 estados), estudiantes de grado y profesionales con dos años de media de experiencia en el desarrollo de *software* orientado a objetos son:

- La idea inicial que tuvieron los autores y que es la más comúnmente aceptada en el campo de la ingeniería de *software*, es que el uso de los estados compuestos ayuda a que los diagramas de transición de estados UML sean más comprensibles. Sin embargo, los resultados del meta-análisis muestran que el uso de estados compuestos tiene una influencia negativa en la *Efectividad de la Comprensibilidad* (*CEfec*) de los diagramas, es decir, la forma en que los sujetos entienden directamente cómo funciona el diagrama. Este resultado va en contra de la sabiduría convencional. Sin embargo, los resultados individuales de E3 están a favor de esta afirmación. Sospechamos que la razón podría ser que los sujetos con más experiencia son capaces de tomar ventaja de los beneficios del uso de los estados compuestos. Cuando hay una falta de experiencia, podría ser más difícil de entender y gestionar el uso de los estados compuestos. También parece que cuanto más complicadas sean las tareas a realizar, más útil será el uso de estados compuestos.
- Los resultados globales no muestran un claro efecto, ya sea en el uso o no uso de los estados compuestos relacionados con los conceptos de *Transferencia*, es decir, la capacidad de utilizar los conocimientos adquiridos a partir del material recibido para resolver problemas relacionados pero que no se pueden resolver directamente a partir del material recibido (*CTrans*), y de *Retención*, es decir, la capacidad de memorizar el material que está siendo presentado (*CReten*). Pero en particular los resultados obtenidos en E3 muestran que los estados compuestos mejoran la *Transferencia* de los diagramas. Como ya hemos comentado, se sospecha que la causa podría ser que cuando los sujetos son más experimentados, pueden beneficiarse de la utilización de los

estados compuestos, de lo contrario el efecto es el contrario. Al mirar la *Retención* (*CReten*), los resultados de E3 presentan un efecto negativo y parece que es mejor no usar los estados compuestos para memorizar diagramas.

A pesar de que el meta-análisis parece mejorar los resultados de los estudios individuales, después de un largo y riguroso período de experimentación no se pueden extraer resultados concluyentes sobre si los estados compuestos son beneficiosos o no para la comprensión de los diagramas de transición de estados UML, en el contexto mencionado. Sin embargo, nuestros resultados muestran que el uso de estados compuestos puede no ser siempre beneficioso, como se creía inicialmente.

En cualquier caso, se necesita una mayor investigación en las siguientes direcciones para favorecer a la validez externa (generalización de los resultados):

- Estudio de la hipótesis que expresamos sobre el efecto que los estados compuestos pueden tener sobre las personas sin destrezas en su uso.
- Ampliación del número de profesionales en estudios futuros, que refuercen la validez de las conclusiones, ya que el tamaño de la muestra de la E3 (24 profesionales) es pequeña en comparación con otros estudios en la familia.
- Uso de diagramas y tareas más complejas correspondientes a proyectos reales relacionados con dominios de sistemas de tiempo real.

3.8 LECTURAS RECOMENDADAS

- Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B. (2012). *Experimentation in software engineering*. Springer. Esta es la segunda edición del primer libro que se publicó sobre la experimentación adaptada a la ingeniería de *software*, cuya primera edición fue publicada por la editorial Kluwer en el año 2000. Si bien la mayor parte del libro se dedica a explicar los pasos del proceso experimental, también trata otros temas como: revisiones sistemáticas de la literatura, encuestas y casos de estudios. Este libro es el más referenciado en publicaciones sobre estudios empíricos en ingeniería de *software*.
- Juristo, N., Moreno, A. (2001). *Basics of software engineering experimentation*. Kluwer. Este es el segundo libro que se publicó específicamente sobre el tema de experimentación en la ingeniería de

software. Presenta en profundidad los tipos diseños experimentales y las técnicas estadísticas para el análisis de datos.

- Forrest, S., Janice, S y Sjøberg, D. (Eds.) (2008). *Guide to advanced empirical software engineering*. *Springer*. Este libro es un compendio de capítulos escritos por diferentes autores sobre temas específicos relacionados la ingeniería del *software* empírica. Está dividido en tres secciones, métodos de investigación, fundamentos prácticos y creación de conocimiento. Destacar de este libro el capítulo 8 "*Reporting Experiments in Software Engineering*" (Jedlitschka, A., Ciolkowski, M., Dietmar P.), que es uno de los más referenciados y que presenta consejos sobre cómo reportar experimentos.

3.9 SITIOS WEB RECOMENDADOS

- *http://isern.iese.de/Portal/*

  Este es el portal de la "*International Software Engineering Research Network* (*ISERN*)", formada por prestigiosos miembros del mundo académico y también industrial de diferentes países de todos los continentes. El principal objetivo de la red es fomentar el uso de métodos empíricos en la investigación realizada en la ingeniería del *software* para consolidar la ingeniería del *software* como una disciplina basada en la evidencia.

3.10 HERRAMIENTAS RECOMENDADAS

En el Grupo Alarcos se ha construido la herramienta *Empirical-WebGen* (Noviello *et al.*, 2008) para llevar a cabo experimentos. Es una herramienta gratuita que está disponible *http://webgen.webportalquality.com/*. Existen además otras herramientas para dar soporte al proceso experimental, como *SESE* (Arisholm *et al.*, 2002; Karahasanovic *et al.*, 2005), *Ginger2* (Torii *et al.*, 1999) y *eSEE* (Travassos *et al.*, 2008), entre otras.

Para realizar el análisis de datos se suelen utilizar los siguientes paquetes estadísticos: *R* (*http://www.r-project.org/*) que es gratuito y el paquete *SPSS* (*www.ibm.com/software/analytics/spss/*).

Para realizar el meta-análisis la herramienta más utilizada es *Comprehensive Meta-Analysis* (*http://www.meta-analysis.com/*).

Capítulo 4

ESTUDIOS DE CASO

4.1 INTRODUCCIÓN

Tradicionalmente se habla de estudio de caso como un método empírico destinado a investigar fenómenos contemporáneos en su propio contexto (Benbasat *et al.*, 1987; Robson, 2002; Yin, 2014). A partir de ahí, hay distintos matices que se añaden a esta definición, como que se suelen utilizar múltiples fuentes de evidencias (Robson, 2002), la imprecisión que puede aparecer entre los límites del fenómeno y su contexto (Yin, 2014) o la ausencia de control experimental junto con la recolección de información de unas pocas entidades (personas, grupos u organizaciones) (Benbasat *et al.*, 1987).

Otra definición, que trata de unificar las tres anteriores establece que un estudio de caso en ingeniería del *software* es una investigación empírica que hace uso de múltiples fuentes de evidencia para investigar una instancia (o un pequeño número de instancias) de un fenómeno contemporáneo relacionado con la ingeniería del *software* dentro de su contexto real, específicamente cuando las fronteras entre el fenómeno y su contexto no pueden definirse claramente (Runeson *et al.*, 2012).

Los estudios de caso no dan lugar, como resultado, a relaciones causales como ocurre con los experimentos, sino que permiten comprender más en profundidad el fenómeno que se está estudiando en su contexto real. Precisamente ahí radica la principal funcionalidad de los estudios de caso, en la capacidad de proporcionar resultados de investigación a partir de proyectos del mundo real.

Precisamente esta interacción con el mundo real supone la principal dificultad de los estudios de caso.

Los estudios de caso se caracterizan por (Runeson *et al.*, 2012):

- Ser un método de investigación flexible, ya que han de tratar con las complejas y dinámicas características de los fenómenos del mundo real, como ocurre en el campo de la ingeniería del *software*.
- Sus conclusiones, tanto cualitativas como cuantitativas, se basan en una clara cadena de evidencias, recogida de múltiples fuentes de una forma planeada y consistente.
- Añaden conocimiento al ya existente, basándose en una teoría previamente establecida o estableciendo una si no la hubiera con anterioridad.

4.2 PROCESO DE REALIZACIÓN DE ESTUDIOS DE CASO

A la hora de llevar a cabo un estudio de caso, el proceso más comúnmente utilizado se basa en las siguientes cinco actividades (Runeson *et al.*, 2012): Diseñar y planificar el estudio de caso, Preparar la recogida de datos, Recoger los datos, Analizar e interpretar los datos recogidos e Informar sobre los resultados obtenidos. En general estas actividades son similares a las de otros estudios empíricos, como por ejemplo el proceso experimental presentado en el capítulo 3, aunque existen algunas peculiaridades propias de los estudios de caso que describiremos a continuación.

4.2.1 Diseñar y planificar el estudio de caso

Un estudio de caso es un tipo de investigación flexible, lo que no exime de la necesidad de una correcta planificación para llevarlo a cabo.

A la hora de planificar un estudio de caso es necesario que, al menos, se tengan en cuenta los siguientes elementos (Robson, 2002):

- **Objetivo**: ¿qué se pretende conseguir?
- **El caso y Unidad de Análisis**: ¿qué se va a estudiar?
- **Teoría**: marco de referencia en el que se encuadra el estudio.

- **Preguntas de investigación**: ¿qué hay que saber?
- **Método**: ¿cómo se van a recoger los datos?
- **Estrategia de selección**: ¿dónde hay que buscar los datos?

El objetivo del estudio de caso puede ser, por ejemplo, de tipo exploratorio, descriptivo, explicativo, o de mejora. El objetivo es formulado por naturaleza de manera más general y menos precisa que los diseños de investigación fijos. Inicialmente, el objetivo es más como un punto de enfoque que se desarrolla durante el estudio. Las preguntas de investigación establecen lo que se necesita saber con el fin de cumplir con el objetivo del estudio. Al igual que en el objetivo, las preguntas de investigación se desarrollan durante el estudio y se concretan en preguntas específicas de investigación durante las iteraciones de estudio (Andersson y Runeson, 2007).

En la ingeniería de *software* el caso puede ser un proyecto de desarrollo de *software*, que es la opción más sencilla. Alternativamente, puede ser un individuo, un grupo de personas, un proceso, un producto, una política, un papel en la organización, un evento, una tecnología, etc. El proyecto, el individuo, el grupo, etc. también pueden constituir una unidad de análisis dentro de un caso. Los estudios de caso sobre "programas de juguete" o similares, por supuesto, se excluyen debido a que su contexto no es cercano a la realidad. Sin embargo, se pueden usar como estudios pilotos para probar el diseño, y que sirva como preparación para futuro estudios de caso.

Yin (2014) establece una distinción entre estudios de caso holísticos, en los que el caso se estudia como un todo, y estudios de caso embebidos (*embedded*): en los que dentro del mismo caso se estudian distintas unidades de análisis (ver Figura 4.1).

Decidir si definimos un estudio consistente en dos casos como holístico o embebido depende de lo que definamos como contexto y objetivos de la investigación. Por ejemplo, si estudiamos dos proyectos en dos empresas diferentes y sobre dos dominios de aplicación diferentes, ambos utilizando prácticas ágiles. Por un lado, los proyectos se pueden considerar dos unidades de análisis en un estudio de caso embebido si el contexto es las empresas de *software* en general y el objetivo de la investigación es el estudio de las prácticas ágiles. Por otro lado, si se considera que el contexto sea una compañía específica o dominio de aplicación específico, se deben considerar como dos casos holísticos separados.

Como comentamos en el capítulo 1, no es muy frecuente el uso de teorías para fundamentar la investigación en ingeniería de *software*. Sin embargo, definir el marco de referencia del estudio hace que el contexto de la investigación del

estudio de caso sea más claro, y ayuda tanto a los que llevan a cabo la investigación como a los que revisan los resultados de la misma. A falta de una teoría, el marco de referencia, alternativamente se puede expresar en términos de la perspectiva adoptada en la investigación y los antecedentes de los investigadores. La Teoría Fundamentada (*Grounded Theory*) para los estudios de caso no se basa en ninguna teoría específica (Corbin y Strauss, 2008).

Las principales decisiones sobre los métodos para la recolección de datos se definen en el diseño del estudio de caso, a pesar de que las decisiones específicas sobre los procedimientos de recolección de datos se toman después. Lethbridge *et al.* (2005) definen tres categorías de métodos: directos (p.ej. entrevistas), indirectos (p.ej. a través del uso herramientas) e independientes (p.ej. análisis de la documentación). Estos métodos se tratarán en más detalle en el apartado 4.2.2.

En los estudios de caso, el caso y la unidad de análisis deben ser seleccionados intencionalmente. A diferencia de las encuestas y experimentos, en los que los sujetos se seleccionan de la población sobre la que los resultados se intentan generalizar. El propósito de la selección puede ser el estudio de un caso que se espera que sea "típico", "crítico", "revelador" o "único" en algún aspecto (Benbasat *et al*, 1987), y el caso es seleccionado en consecuencia. En un estudio de caso comparativo, las unidades de análisis deben ser seleccionados para tener la variación de las propiedades que el estudio tiene la intención de comparar. Sin embargo, en la práctica, muchos casos se seleccionan sobre la base de la disponibilidad (Benbasat *et al.*, 1987), similar a lo que ocurre en los experimentos (Sjøberg *et al.*, 2005).

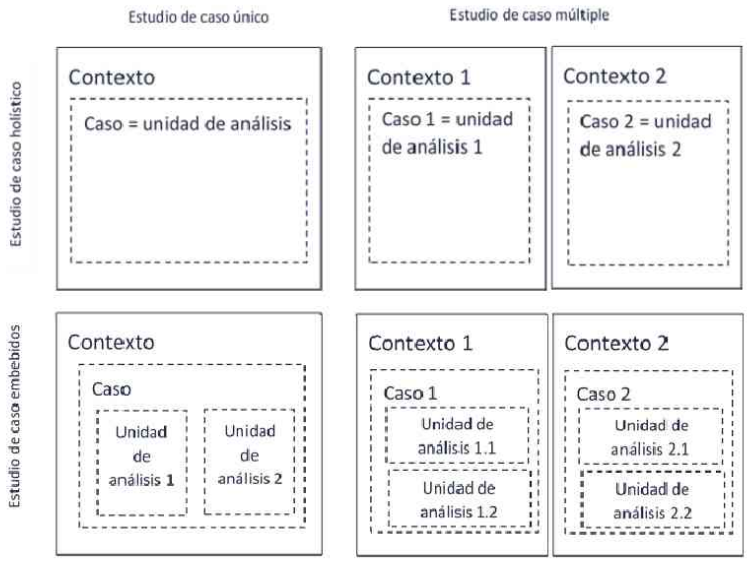


Figura 4.1. Estudios de caso holísticos (*izquierda*) *e embebidos* (*derecha*)

Es aconsejable, definir un protocolo para el estudio de caso que contendrá tanto las decisiones del estudio como los procedimientos de campo que se llevarán a cabo durante su ejecución (Runeson *et al.*, 2012). Es, por tanto, un documento que irá evolucionando a medida que se cambie la planificación del estudio. Además, en relación al protocolo, cabe destacar que:

- Sirve de guía a la hora de recoger los datos, evitando que el investigador olvide recoger datos que se habían planificado recoger.

- El proceso de formulación del protocolo concretiza la investigación en el fase de planificación, lo que puede ayudar al investigador a decidir qué fuentes usar y qué preguntas formular.

- Otros investigadores y cualquier otra persona relevante en el estudio pueden revisar el protocolo y hacer los comentarios que considere oportunos con el fin de reducir el riesgo de perder alguna fuente relevante de información, alguna pregunta interesante o algún rol que añadir.

- Puede servir como una bitácora o diario en el que se registre la recogida y el análisis de los datos junto las decisiones de cambio que se produzcan debido a la naturaleza flexible de la investigación.

A modo de ejemplo, Brereton *et al.* (2008) proponen un esquema de protocolo de un estudio de caso que se resume en la Tabla 4.1.

Apartado	**Contenido**
Antecedentes	Investigación previa. Preguntas de investigación, tanto principales como adicionales.
Diseño	Caso único o múltiple. Diseño holístico o embebido. Objeto del estudio. Proposiciones derivadas de las preguntas de investigación.
Selección	Criterios para la selección del caso.
Procedimientos y roles	Procedimientos de campo. Roles de los miembros del equipo de investigación.
Recogida de datos	Identificación de los datos. Definición del plan de recogida y almacenamiento de los datos.
Análisis	Criterios de interpretación. Conexiones entre los datos y las preguntas de investigación. Explicaciones alternativas.

Validez del plan	Tácticas para reducir las amenazas a la validez.
Limitaciones del estudio	Especificación de otras amenazas a la validez del estudio, que son inherentes al problema investigado en sí y no de la propia planificación del estudio. Por ejemplo, conflicto de intereses.
Informe	Audiencia objetivo.
Calendario	Estimación temporal de los principales pasos a seguir.
Apéndices	Cualquier información adicional detallada.

Tabla 4.1. Posible esquema del protocolo de un estudio de caso (*Brereton et al., 2008*)

4.2.2 Preparar y recoger los datos

En este apartado se presentan conjuntamente la segunda y tercera actividad del proceso de realización de estudios de caso, encargados de la preparación y la recogida de los datos relativos al estudio de caso.

En función del grado de implicación del investigador en la recogida de los datos, se establece la siguiente división de las técnicas de recogida de datos (Lehtbridge *et al.*, 2005):

- **Primer grado**. Métodos de recogida en los que el investigador está en contacto directo con los sujetos y los datos se recogen en tiempo real. En esta categoría se enmarcan las entrevistas, los grupos de discusión y las observaciones basadas en protocolos de "pensar en voz alta" (*think-aloud protocols*) (Owen, 2006).
- **Segundo grado**. Métodos en los que el investigador recoge los datos directamente pero sin interactuar con los sujetos. En este grupo se encuentran el estudio de los registros de uso de herramientas *software* o la observación a través de grabación en video.
- **Tercer grado**. Análisis independiente de artefactos de trabajo donde se utilizan datos previamente disponibles. En este grupo se enmarca el estudio de documentos tales como especificaciones de requisitos e informes de fallos de una organización.

Como parece lógico, los métodos más costosos son los de primer grado, ya que requieren el máximo esfuerzo e implicación tanto de investigadores como de sujetos. A la vez que disminuye el grado de implicación del investigador se reducen los costes de obtención de los datos pero también disminuye el control que el investigador tendrá sobre el proceso de obtención de los datos.

En los siguientes sub-apartados se comentarán algunas de las técnicas más utilizadas a la hora de recoger los datos relativos a un estudio de caso.

4.2.2.1 ENTREVISTAS

Al utilizar entrevistas para recoger los datos, un investigador realiza una serie de preguntas a un conjunto de sujetos sobre las áreas de interés del estudio de caso. En la mayoría de las ocasiones las entrevistas se realizan individualmente, aunque es posible realizar entrevistas grupales.

Las preguntas que establecen el diálogo investigador-sujeto que se produce durante la entrevista, se basan en las preguntas de investigación aunque, por lo general, no se formulan de la misma manera. Estas preguntas podrán ser *abiertas*, si se permite e incluso se invita a que los sujetos proporcionen un amplio y no prefijado rango de respuestas, o *cerradas*, si se ofrece un conjunto de respuestas alternativas.

En función del grado de estructuración de las entrevistas, se establece la siguiente clasificación de las mismas (Robson, 2002):

	No estructuradas	**Semi-estructuradas**	**Completamente estructuradas**
Interés principal	Cómo los individuos experimentan el fenómeno cualitativamente	Cómo los individuos experimentan el fenómeno cualitativa y cuantitativamente	El investigador trata de encontrar relaciones entre conceptos
Preguntas	Manual de entrevista con las áreas en las que centrarse	Mezcla de preguntas abiertas y cerradas	Preguntas cerradas
Objetivo	Exploratorio	Descriptivo y exploratorio	Descriptivo y exploratorio

Tabla 4.2. Tipos de entrevistas

Durante la realización de las entrevistas, se recomienda registrar el proceso de la misma en un formato adecuado de vídeo o audio ya que, aunque pueden tomarse notas, es complicado registrar todos los detalles concretos que suceden durante la propia entrevista. Si se ha registrado la entrevista, conviene transcribirla antes de analizarla e incluso puede ser interesante que el entrevistado examine y apruebe el contenido de la transcripción.

4.2.2.2 OBSERVACIONES

Un ejemplo típico de uso de observaciones, que lo hace particularmente interesante en nuestro caso, consiste en investigar cómo los ingenieros de *software* realizan una determinada tarea. Esta investigación puede realizarse supervisando, a través de una grabación en vídeo, el desempeño de un grupo de ingenieros de *software*, o mediante la aplicación de un protocolo de "pensar en voz alta", en los que el investigador realiza preguntas del estilo "¿qué opinas sobre…?" o "¿cuál es tu estrategia sobre…?". Esto se puede complementar con grabación de audio y de las pulsaciones del teclado.

Los tipos de observaciones se dividen en: 1) Alto o bajo grado de interacción por parte del investigador y 2) Alto o bajo conocimiento de los sujetos observados, como se muestra en la Tabla 4.3.

	Al conocimiento del sujeto observado	**Bajo conocimiento del sujeto observado**
Alto grado de interacción por parte del investigador	Categoría 1	Categoría 2
Bajo grado de interacción por parte del investigador	Categoría 3	Categoría 4

Tabla 4.3. Tipos de observaciones

Las observaciones de la categoría 1 y 2 suelen ser estudios etnográficos o de investigación-acción, en los cuales el investigador es parte del equipo, y no sólo son vistos como investigadores por el resto de integrantes del equipo. La diferencia entre la categoría 1 y 2 es que en la categoría 1 el investigador es visto como un "participante observador" por el resto de integrantes del equipo, y en la categoría 2 es visto como un "participante normal". En la categoría 3, el investigador es sólo visto como un investigador. Las observaciones en la categoría 3, se suelen hacer con técnicas de recogida de datos de primer grado, como por ejemplo el protocolo de pensar en voz alta, como se comentó previamente. En la categoría 4, los sujetos se observan utilizando técnicas de segundo grado como grabaciones de video.

Esta técnica de recolección de datos suele proporcionar un profundo conocimiento sobre el fenómeno que se está estudiando pero, como inconveniente, la cantidad de datos que se genera suele ser muy elevada, por lo que se suele consumir mucho tiempo en el análisis de los mismos.

4.2.2.3 DATOS DE ARCHIVO

Los datos de archivo se refieren, por ejemplo, a documentos de las distintas fases de desarrollo, datos de errores, registros financieros y otras medidas que previamente ha recogido la organización.

Para estos tipos de datos conviene contar con una herramienta de gestión de la configuración, ya que permite almacenar un gran número de documentos e incluso de diferentes versiones de un mismo documento.

4.2.2.4 MÉTRICAS

Hasta aquí, todas las técnicas de recogida de datos se centraban en datos cualitativos, sin embargo los datos cuantitativos también son importantes en un estudio de caso. Los datos se pueden recoger a propósito para el estudio de caso o pueden estar ya disponibles y, simplemente, usarse en el mismo. En este caso el investigador no tiene capacidad de controlar la calidad de los mismos ni si hay datos importantes que no se han recogido.

Los datos disponibles se pueden encontrar en bases de datos de una organización, como por ejemplo: datos de esfuerzo de proyectos anteriores, cifras sobre ventas de productos, métricas sobre la calidad del producto, etc.

4.2.3 Analizar e interpretar los datos recogidos

El análisis de los datos se realiza de manera distinta para datos cualitativos y cuantitativos. Para los datos cuantitativos, se suele realizar un análisis de los estadísticos descriptivos, análisis de correlaciones, modelos predictivos y contraste de hipótesis, mientras que para los datos cualitativos se llevan a cabo técnicas de generación y confirmación de hipótesis.

4.2.3.1 ANÁLISIS DE DATOS CUANTITATIVOS

Como ya se comentó en el capítulo 3, para comprender los datos que se han recogido se suelen utilizar estadísticos descriptivos tales como medias, desviaciones típicas, histogramas y diagramas de caja. El contraste de hipótesis, por su parte, se utiliza para determinar si hay un efecto significativo de uno o varios factores (variables independientes) en una o más variables (variables dependientes).

Cabe resaltar que los métodos para el análisis cuantitativo presuponen un diseño de investigación fijo. Por ejemplo, si una pregunta con una respuesta cuantitativa se cambia a mitad de camino en una serie de entrevistas, esto hace que sea imposible interpretar el valor medio de las respuestas. Además, los conjuntos de datos cuantitativos de los casos individuales tienden a ser muy pequeños, debido al número de los encuestados o a los puntos de medición, lo que causa preocupación especial en el análisis.

4.2.3.2 ANÁLISIS DE DATOS CUALITATIVOS

El objetivo básico de cualquier análisis cualitativo es extraer conclusiones de los datos manteniendo una clara cadena de evidencias, lo que significa que un lector podrá seguir el proceso de obtención de resultados y conclusiones a partir de los datos recogidos (Yin, 2014). Esto significa que se debe detallar la suficiente información de cada actividad del estudio y de todas las decisiones tomadas por el investigador.

Con el fin de reducir sesgos por parte de investigadores individuales, es recomendable que la extracción de conclusiones a partir de los datos la realicen distintos investigadores simultáneamente y que pongan en común sus resultados.

Hay dos partes diferentes a la hora de analizar datos cualitativos (Seaman, 1999):

- **Generación de hipótesis**. Estas técnicas se utilizan para generar hipótesis a partir de los datos. Los investigadores deben realizar un esfuerzo para evitar preconcepciones y estar abiertos a cualquier hipótesis que pueda derivarse de los datos. Como técnicas de generación se pueden utilizar las siguientes: "comparaciones constantes" (*constant comparisons*) y "análisis cruzado de casos" (*cross-case analysis*) (Seaman, 1999).
- **Confirmación de hipótesis**. Estas técnicas se utilizan para confirmar si una hipótesis es válida a través de, por ejemplo, a través del análisis de más datos. Para la confirmación de hipótesis se pueden usar técnicas como la triangulación y la replicación (Seaman, 1999).

En primer lugar, se generan las hipótesis en un primer ciclo del estudio de caso, o con datos de una unidad de análisis, y la confirmación de las hipótesis se puede realizar en un segundo ciclo o a partir de una segunda unidad de análisis. Por lo tanto el análisis de datos cualitativos se realiza en varios pasos (Robson, 2002). En primer lugar, se asigna a partes del texto un código que representa cierto tema, área o constructo. Por lo general, se puede asignar un mismo código a varias partes

del texto, y a una parte del texto se le pueden asignar varios códigos. Se puede así generar una jerarquía de códigos con sub-códigos. El investigador, puede agregar al texto anotaciones y reflexiones. Una vez realizada la codificación, el investigador identifica un primer conjunto de hipótesis. Esto puede ser, por ejemplo, frases similares en diferentes partes del texto, patrones en los datos, diferencias entre subgrupos de sujetos, etc. Las hipótesis identificadas a continuación, se pueden utilizar una vez recogidos más datos. De manera iterativa se van analizando y recogiendo datos en paralelo. Durante este proceso iterativo se pueden realizar algunas generalizaciones y eventualmente se puede construir un cuerpo de conocimiento formalizado, que sea el resultado final de la investigación.

Una técnica comúnmente usada para este tipo de análisis es la tabulación, que consiste en poner en tablas los datos codificados y así poder tener una visión global de los datos. Por ejemplo, en las filas de la tabla se pueden poner los códigos y en las columnas los sujetos entrevistados. Aunque no hay una única manera de construir esta tabla, esto se debe decidir en particular para cada estudio de caso.

Existen herramientas específicas para realizar análisis de datos cualitativos como pueden ser *NVivo* y *Atlas* (ver apartado 4.7). Aunque también se pueden utilizar hojas de cálculo o procesadores de texto para gestionar datos de textuales.

Como se describió anteriormente, es importante utilizar un enfoque estructurado para realizar el análisis de datos cualitativos, aunque el análisis se puede hacer con diferentes niveles de formalismo:

- **Enfoque de inmersión**. Este es el enfoque menos estructurado, se basa más en la intuición y la experiencia interpretativa de los investigadores. Este enfoque es difícil de combinar con el requisito de mantener y comunicar una cadena de evidencia.
- **Enfoque de edición**. Este enfoque incluye la definición de códigos basados en lo hallazgos obtenidos durante el análisis.
- **Enfoque de plantillas**. Este enfoque es más formal e incluye códigos definidos a priori, basados en las preguntas de investigación.
- **Enfoque quasi-estadístico**. Este enfoque es más formal aún e incluye el cálculo de frecuencia de palabras y frases.

En general los enfoques más apropiados para los estudios de casos en el contexto de la ingeniería del *software* son los enfoques de edición y de plantillas. Es difícil de obtener una clara cadena de evidencia basada en enfoques de inmersión informales. Y también es difícil de interpretar, por ejemplo las frecuencias de palabras en textos y entrevistas.

4.2.3.3 VALIDEZ

La validez de cualquier estudio empírico marca la fiabilidad y objetividad de sus resultados. Obviamente, hay que considerar todas las posibles amenazas a validez del estudio antes de la fase de análisis, aunque es en esta fase cuando se comprueba qué posibles problemas se han producido.

Como ocurre con los experimentos (ver capítulo 3), se proponen cuatro factores a tener en cuenta para establecer la validez de un estudio de caso (Yin, 2014):

- **Validez de constructo**. Este aspecto refleja hasta qué punto las medidas que se han realizado se adecúan a lo que el investigador tiene en mente y a lo que se está investigando, en función de las preguntas de investigación.
- **Validez interna**. Este aspecto se basa en estudiar las relaciones causales que hacen que distintos factores afecten a otro factor investigado pero que no se tenga constancia de todos ellos.
- **Validez externa**. Este aspecto marca la capacidad de generalización de los resultados del estudio de caso y el interés que puedan suscitar en personas ajenas al propio caso.
- **Fiabilidad**. Este aspecto indica la dependencia de los datos y su análisis respecto de un investigador específico y la capacidad de replicar el mismo estudio y obtener los mismos resultados.

4.2.4 Informar sobre los resultados obtenidos

Existe un conjunto de características que todo informe sobre un estudio de caso debería incluir (Robson, 2002). Estas características son:

- Contar sobre qué trata el estudio.
- Emitir una opinión personal clara sobre el caso estudiado.
- Proporcionar un historial de la investigación para que pueda saberse qué se ha hecho, cómo y quién lo ha hecho.
- Proporcionar datos básicos para que el lector pueda asegurarse de que las conclusiones son razonables.
- Articular y contextualizar las conclusiones de la investigación.

A la hora de publicar un informe sobre un estudio de caso la estructura lineal analítica (problema, trabajo relacionado, método, análisis y conclusión) es la estructura más aceptada. Aun así se ha definido una estructura que puede servir como esqueleto a cualquier publicación académica sobre un estudio de caso (Jedlitschka y Pfahl, 2005; Runeson *et al.*, 2012), que consiste en los siguientes apartados: Título, Autores, Resumen estructurado, Introducción, Trabajo relacionado, Diseño del estudio de caso, Resultado, Conclusiones y trabajo futuro, Agradecimientos y Referencias.

4.3 EJEMPLO DE ESTUDIO DE CASO

A continuación se presenta un ejemplo de estudio de caso (Fernández-Sáez *et al.*, 2013c), cuyo objetivo principal es investigar el retorno de la inversión (ROI) de utilizar lenguajes de modelado en tareas de mantenimiento de *software*. En particular, el estudio se basa en el lenguaje de modelado UML, ampliamente utilizado en entornos de desarrollo profesional.

El trabajo es el resultado de ocho meses de investigación en una compañía multinacional holandesa. La unidad organizacional que sirvió como foco de esta investigación se localizaba dentro del departamento de desarrollo de servicios de información, donde se produce todo el desarrollo de *software* de la compañía.

4.3.1 Diseño y planificación del ejemplo

La investigación parte de una pregunta principal que guiará todo el diseño y la ejecución del estudio: *"¿Cuál es el ROI de utilizar UML en proyectos de mantenimiento de software?"*.

Desgranando un poco más la pregunta de investigación, puede indicarse que la investigación pretende comprobar si la inversión en UML que realizan algunas compañías se justifica en términos de beneficios para los proyectos de mantenimiento de *software*, como mejoras en la productividad o en la calidad de los productos. Debido a la complejidad de la pregunta principal de investigación se decidió dividirla en dos preguntas independientes:

- **PI1**: ¿Cuál es el coste de utilizar UML en proyectos de mantenimiento de *software*?
- **PI2**: ¿Cuál es el rendimiento financiero de utilizar UML en proyectos de mantenimiento de *software*?

Debido a la naturaleza del campo que se está estudiando, es difícil cuantificar tanto el rendimiento financiero como los costes del uso de UML, por lo que se tratarán de conseguir datos cuantitativos que permitan responder las preguntas de investigación pero, a la vez, también tratarán de conseguir datos cualitativos (basados en opiniones personales) para complementar a los primeros y mejorar la percepción global obtenida.

Los datos cuantitativos se conseguirán a través de la revisión de la documentación de proyectos históricos mientras que la información cualitativa provendrá de la realización de encuestas a sujetos de diferentes roles (ingenieros de *software*, probadores, desarrolladores, etc.) involucrados en diferentes proyectos. Durante estas entrevistas, se obtendrá información sobre cómo los encargados de tareas de mantenimiento utilizan diagramas UML en sus tareas diarias.

Con el fin de alcanzar los objetivos planteados y ser capaces de responder a las preguntas de investigación de una manera más fácil, se plantearon cuatro preguntas más sencillas de responder:

- **P1**: ¿Cuáles son los costes asociados al uso de herramientas relativas al uso de UML? (relacionada con PI1).
- **P2**: ¿Cuáles son los costes asociados a la formación en UML? (relacionada con PI1).
- **P3**: ¿Qué impacto tiene el uso de diagramas UML en las desviaciones (en tiempo y presupuesto) de proyectos de mantenimiento *software*? (relacionada con PI1 y PI2).
- **P4**: ¿Qué impacto tiene el uso de diagramas UML en la comprensión de los encargados del mantenimiento de *software* y en la calidad del producto? (relacionada con PI2).

Finalmente, como se muestra en la Figura 4.2 y aplicando la clasificación de Yin (2014), este estudio de caso puede caracterizarse como único y embebido (*single embedded case study*).

Figura 4.2. Estudio de caso único y embebido

4.3.2 Preparación y recogida de los datos en el ejemplo

A la hora de recoger los datos sobre el uso de UML durante las tareas de mantenimiento se usaron dos fuentes, que corresponden a las distintas unidades de análisis del caso:

- **Archivos de proyecto compartidos por el departamento**. El departamento de desarrollo cuenta con un servidor de ficheros en el que se comparte toda la documentación relativa a los proyectos que desarrolla.
- **Personal de la compañía**. El investigador encargado del estudio de caso era parte del personal de la compañía en el momento de la realización del mismo, por lo que tenía un acceso relativamente sencillo al personal de la compañía y, más en concreto, al personal involucrado en proyectos de mantenimiento de *software*.

A través de la primera fuente se obtuvieron los datos cuantitativos de cada proyecto. En primer lugar se utilizaron los datos de estimación del proyecto, almacenados en una hoja de cálculo. Otro dato de interés consistía en la presencia de diagramas UML en la documentación de los proyectos de mantenimiento, pero fue algo más complicado porque la compañía no posee un repositorio de modelos y hubo que examinar manualmente cada carpeta del proyecto para saber cuáles contenían diagramas UML. En concreto y con tal fin, se examinaron los ficheros relativos al modelado utilizando Enterprise Architect (*http://www.sparxsystems.es/*) y los documentos de texto (en formato propietario o pdf).

También se consiguieron datos cualitativos a través de entrevistas que se mantuvieron con el personal de la compañía. Se dividió a los sujetos entre usuarios y no usuarios de UML y se llevaron a cabo entrevistas semi-estructuradas en las que las mismas (o muy parecidas) preguntas se realizaron a todos los entrevistados. Por último, cabe destacar que la mayoría de las preguntas eran abiertas, es decir, se daba un margen amplio de respuesta a los entrevistados.

A continuación, y a modo de ejemplo, se muestran en la Tabla 4.4 algunas de las preguntas que formaban parte del cuestionario utilizado en el estudio de caso. Algunas preguntas son genéricas y otras dependen de si se usa o no UML.

Preguntas genéricas
1.- ¿Cuáles son sus antecedentes y su experiencia?
2.- ¿Cuál es su rol y cuáles son sus responsabilidades dentro del proyecto?
5.- ¿Con qué frecuencia utiliza usted la documentación?
Si utiliza diagramas UML
6.- ¿Por qué razones utiliza UML? ¿Con qué propósito se utiliza el modelado en UML?
8.- ¿Qué diagramas considera que son los más utilizados para realizar tareas de mantenimiento? ¿Qué diagramas considera que son los más útiles para realizar tareas de mantenimiento?
10.- Cuando mantiene el código, ¿mantiene también los diagramas?
Si la respuesta es sí
10.1.- ¿Cuánto tiempo le lleva?
10.2.- ¿Quién mantiene los diagramas, la misma persona que mantiene el código o una diferente?
11.- ¿Le gusta UML?
12.- ¿Cree que UML tiene ventajas? ¿Cuáles? ¿Y desventajas?
13.- ¿Cree que UML le ayuda a ahorrar tiempo?
14.- ¿Cree que UML ayuda a mejorar la calidad del producto final? ¿Cómo?
18.- ¿Cree que el uso del modelado introduce errores?
Si no utiliza diagramas UML
22.- ¿Le gustaría disponer de diagramas UML?
Si la respuesta es sí
22.1.- ¿Cómo cree que UML le ayudaría a mantener el sistema?
22.2.- ¿Qué beneficios cree que los diagramas UML podrían aportar a su trabajo?
22.3.- ¿Cree que UML ayuda a mejorar la calidad del producto final? ¿Cómo?
22.4.- ¿Qué factores de coste están relacionados a utilizar el modelado UML en un proyecto?

Tabla 4.4. Ejemplo de preguntas de la entrevista

El cuestionario completo se puede descargar de *http://alarcos.esi.uclm.es/download/list-of-questions.pdf*

4.3.3 Análisis e interpretación de los datos del ejemplo

Con respecto a los datos cuantitativos, se utilizaron test estadísticos para establecer las relaciones entre las distintas variables del proceso. En concreto, se utilizaron los test habituales para determinar la normalidad y la homogeneidad de la distribución (*Kolmogorov-Smirnov* y *Levene*), que determinaron que la mejor opción para contrastar las hipótesis era un test no paramétrico y se decidió utilizar el test de Mann-Whitney. Para todos los cálculos estadísticos se utilizó la herramienta *SPSS*.

Para obtener los datos cualitativos, en primer lugar se realizaron las entrevistas, capturando el audio de las conversaciones con una grabadora de sonidos y, posteriormente, transcribiendo el contenido de las mismas con la herramienta Digital *Voice Editor*. A continuación se analizó cada transcripción, resaltando las frases más importantes o sorprendentes a través de la herramienta *NVivo v.10*. Por último, y utilizando la misma herramienta, se codificaron y agruparon en temas más generales las frases resaltadas.

4.3.4 Informe de los resultados obtenidos

Los resultados se obtuvieron a partir de la información histórica de 135 proyectos y de 20 entrevistas personales.

Para respetar la estructura que plantea el trabajo original, presentaremos los resultados relativos a las 4 preguntas de investigación formuladas (ver apartado 4.3.1), en distintos apartados.

4.3.4.1 COSTES RELACIONADOS CON LAS HERRAMIENTAS UML (P1)

Una vez estudiados los datos de los que se disponía, se llegaron a las siguientes conclusiones en función de las distintas herramientas que se utilizaban en el momento de la realización del estudio:

- **Visio**. Un 15% de las personas encargadas de modelar utilizaban esta herramienta para hacerlo. El número de licencias en uso no estaba claro. El precio de cada licencia oscilaba entre los 160€ y los 350€ y no había costes de mantenimiento asociados al uso de esta herramienta.
- **Bizz Design Architect**. Sólo un 5% del personal de modelado utilizaban esta herramienta y sólo había 2 licencias en uso. El coste por licencia no estaba claro aunque en la página web de la herramienta estipulaban un precio por licencia de 5000€ en 2007. Además, el precio de mantenimiento por año ascendía a 6489€.
- **Sparxs Enterprise Architect** (**EA**). El 80% del personal de la compañía dedicado a modelar utilizaba esta herramienta. Había un total de 100 licencias activas, cada una con un coste (ya pagado) de 135€ y un coste de mantenimiento de 4430€ anuales.
- **Erwin**. Se sabía que esta herramienta se había utilizado con fines de modelado, pero no se encontraron datos cuantitativos sobre su uso.

A la luz de estos datos, se puede apreciar cómo la compañía había decidido utilizar EA como herramienta de modelado hacía tiempo y que aún se estaba en fase de adaptación a la herramienta. Esta decisión de utilizar EA parece correcta, ya que es una herramienta específica y completa y, además, más barata que las otras utilizadas. Además, el coste que supone su utilización es asumible en función del presupuesto con el que cuenta la compañía para los proyectos de mantenimiento.

4.3.4.2 COSTES RELACIONADOS CON LA FORMACIÓN EN UML (P2)

Utilizando los datos históricos (entre 2006 y 2012) proporcionados por el personal encargado de gestionar la formación interna y externa de los empleados, se detectaron diversos cursos relacionados con la formación en UML y otros conceptos relacionados (orientación a objetos, Proceso Unificado de Desarrollo, etc.).

El coste total detectado en este apartado ascendía, durante ese período de tiempo, a 24.313 €, de nuevo una cantidad asumible en función del presupuesto disponible.

4.3.4.3 IMPACTO DE UML EN LAS DESVIACIONES DE LOS PROYECTOS (P3)

En este caso, se trataba de saber si la presencia de diagramas UML en la documentación de proyectos de mantenimiento de *software* afectaba de alguna manera las estimaciones en cuanto a tiempo y presupuesto de los proyectos, por lo que se contaba con dos hipótesis de trabajo:

- **$H1_0$**: no hay una diferencia significativa en cuanto a las desviaciones del calendario previsto en proyectos que utilizan UML y otros que no lo utilizan.

- **$H2_0$**: no hay una diferencia significativa en cuanto a las desviaciones del presupuesto previsto en proyectos que utilizan UML y otros que no lo utilizan.

Las variables utilizadas en el proceso de contraste de estas hipótesis fueron:

- **Presencia de UML**. Esta variable se relaciona con la presencia o ausencia de diagramas UML en la documentación del proyecto de mantenimiento *software*. Tiene, en consecuencia, dos posibles valores, positivo y negativo, según los diagramas estén disponibles o no en la documentación del proyecto.
- **Desviación del calendario**. Esta variable se relaciona con la posibilidad de desviarse con respecto al calendario de un proyecto de mantenimiento *software*. Para calcularla se emplea una fórmula que consiste en restar los días planeados de los días utilizados realmente. Además, se utilizaron una serie de medidas relacionadas que se detallan a continuación:
  - *Desviación en días*. Muestra el número de días de desviación entre el calendario inicial y el realmente ejecutado. Al calcular este número, se obtiene un entero negativo si se presupuestaron más días de los que finalmente se necesitaron para realizar el proyecto y un entero positivo si el proyecto se realizó durante más días de los inicialmente planeados. La fórmula de cálculo es: (*fecha de fin real – fecha de fin planeada*) – (*fecha de inicio real – fecha de inicio planeada*).
  - *% Dif. R-P*. Muestra el porcentaje de desviación en el número total de días del proyecto. Al calcular este número, se obtiene un porcentaje positivo si se necesitaron más días que los que se habían previsto y un porcentaje negativo en caso contrario. La fórmula de cálculo es: ((*fecha de fin real – fecha de fin planeada*) – (*fecha de inicio real – fecha de inicio planeada*)) / (*fecha de fin planeada – fecha de inicio planeada*).
  - *% Dif. R-Ac*. Como durante la ejecución de un proyecto, cabe la posibilidad de actualizar el calendario del mismo y añadir o eliminar días del calendario si se detecta que era erróneo, esta medida muestra el porcentaje de desviación en el número total de días del proyecto, pero teniendo en cuenta las actualizaciones que se hubieran producido en el calendario del mismo. La fórmula de cálculo es: (*fecha de fin real – fecha de fin actualizada*) – (*fecha de inicio real – fecha de inicio actualizada*) / (*fecha de fin actualizada – fecha de inicio actualizada*).

  - *Dentro de P días*. Esta medida, que admite valores de verdadero o falso, evalúa si los días planeados y los utilizados realmente coindicen.

  - *Dentro de Ac días*. Esta medida, que admite valores de verdadero o falso, evalúa si los días planeados, y posteriormente actualizados, y los utilizados realmente coindicen.

- **Desviación del presupuesto**. Esta variable se relaciona con la posibilidad de desviarse con respecto al presupuesto de un proyecto de mantenimiento *software*. Para calcularla se emplea una fórmula que consiste en restar la cantidad de dinero presupuestada de la realmente gastada. Además, se utilizaron una serie de medidas relacionadas que se detallan a continuación:

  - *% Dif. Presupuesto R-P*. Muestra el porcentaje de desviación del presupuesto ejecutado con respecto al inicialmente presupuestado. La fórmula de cálculo es: (*presupuesto ejecutado – presupuesto planeado*) / (*presupuesto planeado*).

  - *% Dif. Presupuesto R-Ac*. Muestra el porcentaje de desviación del presupuesto ejecutado con respecto al presupuesto actualizado del proyecto. La fórmula de cálculo es: (*presupuesto ejecutado – presupuesto actualizado*) / (*presupuesto actualizado*).

  - *Presupuesto dentro de P*. Esta medida, que admite valores de verdadero o falso, evalúa si la cantidad económica presupuestada para la realización del proyecto y la realmente ejecutada coindicen.

  - *Presupuesto dentro de Ac*. Esta medida, que admite valores de verdadero o falso, evalúa si la cantidad económica presupuestada para la realización del proyecto, tras ser actualizada, y la realmente ejecutada coindicen.

Se realizó, además, la combinación de algunas de estas variables para obtener otras medidas:

- **Dentro de P días + presupuesto dentro de P**. Esta medida, que admite valores de verdadero o falso, evalúa simultáneamente si, por un lado, los días planeados y los utilizados realmente en la realización del proyecto coindicen y si, a la vez, la cantidad económica presupuestada para la realización del proyecto y la realmente ejecutada coindicen.

- **Dentro de Ac días + presupuesto dentro de Ac**. Esta medida, que admite valores de verdadero o falso, evalúa simultáneamente si, por un lado, los días planeados para la realización de un proyecto tras su actualización y los utilizados realmente coindicen y si, a la vez, la cantidad económica presupuestada, tras ser actualizada, para la realización del proyecto y la realmente ejecutada coindicen.

La Tabla 4.5 muestra los resultados obtenidos al ejecutar los test estadísticos comentados en el apartado 4.3.3 sobre las variables previamente presentadas. La segunda y tercera columnas (Mann-Withney y ANOVA), corresponden a los test estadísticos realizados, la cuarta columna (sig.) corresponde al valor de significación del ANOVA. Un valor inferior a 0,05 en la segunda o cuarta columna significa que la variable en cuestión afecta a la desviación (bien en tiempo o en presupuesto) del plan inicial del proyecto.

Variable	Mann-Withney	ANOVA	sig.
Desviación en días	0,688	0,517	0,478
% Dif. R-P	0,584	0,987	0,329
% Dif. R-Ac	0,633	0,450	0,508
Dentro de P días	0,699	0,145	0,707
Dentro de Ac días	0,929	0,008	0,931
% Dif. Presupuesto R-P	0,383	1,222	0,279
% Dif. Presupuesto R-Ac	0,081	2,223	0,148
Presupuesto dentro de P	0,591	0,281	0,600
Presupuesto dentro de Ac	0,063	3,809	0,061
Dentro de P días + presupuesto dentro de P	0,575	0,307	0,584
Dentro de Ac días + presupuesto dentro de Ac	0,699	0,145	0,707

Tabla 4.5. Test estadísticos sobre las desviaciones

Puede comprobarse cómo ningún valor de los obtenidos es estadísticamente significativo, es decir, inferior a 0,05, lo que implica que no se puede rechazar ninguna hipótesis nula.

La interpretación de estos resultados indica que no se puede afirmar que, al menos en el caso estudiado, no hay una diferencia significativa en cuanto a las desviaciones del calendario ni del presupuesto previstos en proyectos que utilizan UML y otros que no lo utilizan.

4.3.4.4 IMPACTO DE UML EN LA COMPRENSIÓN DE TAREAS DE MANTENIMIENTO Y EN LA CALIDAD DEL PRODUCTO (P4)

En función de los datos obtenidos en las entrevistas realizadas a los trabajadores involucrados en proyectos de mantenimiento *software* se obtuvieron los datos que se presentan en este apartado. Con el fin de mejorar la claridad de la presentación, se hará una presentación de los resultados dividida en distintos temas relacionados con UML.

Uso de UML

Los diagramas UML que los entrevistados indicaron que usaban y la frecuencia con la que se mencionaron son: diagramas de secuencia (80% de los entrevistados), diagramas de clases (60%), diagramas de actividad y de casos de uso (50%), diagramas de despliegue (40%), diagramas de componentes (30%) y diagramas de colaboración (10%). Los entrevistados indicaron, además, que todos los diagramas se utilizan durante el proceso de mantenimiento completo.

Propósito del uso de UML

Se obtuvo una amplia variedad de respuestas a las preguntas "¿Por qué se utiliza UML?" y "¿Para qué se utiliza el modelado con UML?". La respuesta más repetida indicaba el uso de UML como herramienta de comunicación (22% de los entrevistados), pudiendo esta comunicación ser con los miembros del propio equipo, incluyendo los distintos *stakeholders* (8%) o con miembros de otros equipos (5%). UML también se utiliza para comunicar la situación actual del proyecto al personal recién incorporado al mismo (7%). En cualquier caso, las respuestas indican que aunque " […] UML ayuda a mejorar la comunicación, no la puede reemplazar […]".

Otros usos comentados de UML son: mejorar el conocimiento del sistema en mantenimiento (8%), análisis de riesgos (7%) y guía para la realización de pruebas (7%).

Coste del uso de UML

En las distintas entrevistas, se preguntó acerca de posibles factores de coste o inversiones relacionadas al uso de una notación de modelado como UML en una compañía de mantenimiento de *software*.

Los distintos factores de coste detectados y el porcentaje en el que fueron referenciados se muestran en la Tabla 4.6.

Factor de coste	% referencias
Formación en notación UML en herramientas de modelado	33% 22% 5%
Migración	28%
Cambio en la forma de pensar	11%
Uso de herramientas	11%
Cambios en la gestión central	5%
Curva de aprendizaje	5%
Cambio en los procesos	5%

Tabla 4.6. Factores de coste relacionados con el uso de UML

El factor más comúnmente repetido es el coste en formación, puede que por las propias carencias en el conocimiento de UML que tenían los entrevistados.

Ventajas y desventajas de UML

Las principales ventajas y desventajas relativas al uso de UML que se detectaron en las entrevistas se resumen en la Tabla 4.7.

Ventajas	Desventajas
Relativas a las características de UML	
Alto nivel de abstracción Alta adecuación para el diseño de sistemas OO Muestra diferentes puntos de vista Estandarizado	No ejecutable Semántica clara o inexistente Libertad para los estilos, uso de capas, etc. Alto nivel de abstracción Ausencia del punto de vista del usuario Baja capacidad para la definición de SOA (Arquitecturas Orientadas a Servicios). No se obliga a separar el qué del cómo

Relativas al uso de UML	
Ayuda a clarificar los procedimientos	Dificultades para comprender la notación
Ayuda a estructurar la forma de modelar	Dificultades para modelar cosas complejas
Mejora la documentación	Falta de expresividad
Es un lenguaje común con aceptación global	
Es el único lenguaje de modelado que se aprende adecuadamente	
Reduce fallos de interpretación en proyectos globales	

Tabla 4.7. Ventajas y desventajas de UML

Es curioso encontrar cómo el alto nivel de abstracción de UML se menciona, a la vez, como ventaja y desventaja. Puede que los arquitectos perciban la abstracción como beneficiosa mientras que los desarrolladores necesiten diagramas que estén más cerca del código.

Uso de UML y calidad del software

Al preguntar a los entrevistados si el uso de UML ayudaba a mejorar la calidad del producto final, el 100% de los entrevistados respondieron positivamente. En cualquier caso, resulta interesante destacar que se consideró que la calidad del código fuente se relaciona más con la realización correcta de pruebas y la obtención de resultados positivos en ellas.

El 17% de los entrevistados consideró que el uso de UML reduce la introducción de defectos en el código del sistema, mientras que el 8% opinaba justo lo contrario, es decir, que introducía defectos.

Otro dato destacable es que el 42% de los entrevistados opinaba que el uso de UML es muy útil a la hora de encontrar la causa de un problema en el código fuente.

Estandarización

Al preguntar sobre la estandarización en la forma de trabajar, especialmente en las tareas de documentación y la creación de diagramas, un 10% de los entrevistados consideraba que había un exceso de estandarización en la

compañía, mientras que un 37% detectaba una importante falta de la misma, especialmente en aspectos de nomenclatura, uso de capas, estilos y niveles de detalle en el modelado de los sistemas.

Independientemente de su opinión sobre la presencia de estándares en la compañía, la mayoría de los entrevistados (53%) estaba de acuerdo en que existía una carencia en la adecuación del uso de estándares y que eran necesarios mecanismos de incentivación para el uso de los mismos.

4.3.5 Amenazas a la validez

Como en todo estudio empírico, conviene tener en cuenta las posibles amenazas a la validez del mismo que, en este caso son:

- **Validez interna**. La edad, la educación, el rol y la experiencia de los participantes en las entrevistas pueden haber sido factores que hayan influenciado la posición de los mismos a favor o en contra del uso de UML.
- **Validez externa**. La selección de la muestra de individuos que participaron en las entrevistas se realizó de la manera más aleatoria posible.
- **Validez de constructo**. Las transcripciones de las entrevistas se enviaron a los entrevistados para que hicieran las correcciones que consideraran oportunas.
- **Fiabilidad**. La cadena de evidencias desde las entrevistas hasta el análisis de la documentación se llevó a cabo respetando la literalidad de los datos obtenidos para evitar introducir sesgos a través de la interpretación.

4.4 OTROS EJEMPLOS DE ESTUDIOS DE CASO

Además de presentar otro ejemplo en el capítulo 7, destacamos algunos estudios de casos existentes en la literatura, entre otros: Boehm y Ross (1988), Curtis *et al.* (1988), Kitchenham *et al.* (1995), Li *et al.* (2011), Rainer (2011), McLeod *et al.* (2011), Di Bella *et al.* (2013) y Estler *et al.* (2013).

4.5 ESTUDIOS ETNOGRÁFICOS

Pese a que algunas fuentes consideran los estudios etnográficos dentro de las grandes metodologías de investigación, tradicionalmente se consideran como un tipo especializado de estudio de caso en el que se pone el foco en prácticas culturales (Easterbrook *et al.*, 2008) o estudios de larga duración con enormes cantidades de datos participante-observador (Kitchenham *et al.*, 2002).

La etnografía es un tipo de investigación cualitativa que pretende observar una actividad, comprender el punto de vista del informador y hacer lo implícito explícito (Sharp, 2012). El foco principal radica en el punto de vista del informador y se basa en responder a preguntas del estilo ¿qué es importante y qué no lo es?, pero siempre desde el punto de vista del informador, no del investigador. Las mismas preguntas se pueden realizar pero cambiando el adjetivo por relevante, interesante, doloroso, emocionante, etc.

En la etnografía, el investigador se encuentra físicamente en el lugar de la investigación por un período largo de tiempo, por lo que puede observar tanto lo que las personas hacen como lo que dicen que hacen. Así, los estudios etnográficos proporcionan una visión más rica de los aspectos humanos, sociales y organizacionales de los sistemas de información (Myers, 1999). En estos estudios se obtiene un conocimiento más profundo de las personas, de las organizaciones y del contexto en las que las primeras desempeñan su trabajo en las segundas.

Las principales diferencias entre los estudios etnográficos y los estudios de caso son que, en primer lugar, el grado de involucración del investigador en el grupo social que se está estudiando es mucho mayor en los estudios etnográficos y, en segundo lugar, a las fuentes de datos tradicionales de los estudios de caso (entrevistas, datos de las organizaciones, resúmenes, etc.) se complementan con la recogida de datos a través de la observación del investigador participante lo que, como ya se ha comentado, implica que el investigador invierta una cantidad mucho mayor de tiempo en la investigación.

Para concluir este apartado, mencionar algunos ejemplos de estudios etnográficos que pueden resultar interesantes: Sharp (2004), Robinson *et al.* (2007), Passos *et al.* (2012) y Scaniello y Salviulo (2014).

4.6 LECTURAS RECOMENDADAS

- **Runeson**, **P**., **Höst**, **M**., **Rainier**, **A**. y **Regnell**, **B**. *Case study research in software engineering. Guidelines and examples. Wiley*. (2012). Libro de referencia para el diseño y la realización de estudios de caso en el campo de la ingeniería del *software*. Incluye todo el marco teórico de referencia, así como un buen número de ejemplos.

- **Runeson**, **P**. y **Höst**, **M**. (2009). *Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering*, 14(2): 131-164. Este artículo constituye un manual sobre cómo realizar estudios de caso en el ámbito de la ingeniería del *software*. Además de analizar todo el trasfondo teórico de este tipo de estudios empíricos, incluyendo el método de realización de estudios de caso, propone una serie de líneas guía para realizar publicaciones académicas que sirvan para informar sobre los estudios realizados.

- **Verner**, **J**. **M**., **Sampson**, **J**., **Tosic**, **V**., **Bakar**, **N**. **A**. **A**. y **Kitchenham**, **B**. **A**. (2009). *Guidelines for industrially-based multiple case studies in software engineering. Third International Conference on Research Challenges in Information Science*, pp. 313-324. En este trabajo, los autores tratan de homogeneizar las distintas propuestas metodológicas que, hasta ese momento, existían para la realización de estudios de caso relativos a la ingeniería del *software*, pero estableciendo como contexto específico de aplicación la industria.

- **Yin**, **R**. **K**. (2014). *Case study research. 5th edition. Sage*. Libro clásico sobre la investigación basada en estudios de caso. No se enmarca específicamente en la ingeniería del *software* sino en los estudios de caso en general. Ha servido de fuente de inspiración a la mayoría de publicaciones sobre estudios de caso de las distintas ciencias.

- **Hammersley**, **M**, y **Atkinson**, **P**. (2007). *Ethnography*: *principles in practice. 3rd edition*. Taylor & Francis. Este libro presenta una introducción a los estudios etnográficos en general pero puede también aplicarse en el ámbito de la ingeniería de *software*.

4.7 HERRAMIENTAS Y SITIOS WEB RECOMENDADOS

A pesar de que en la mayoría de los casos es más que suficiente contar con un procesador de textos y una hoja de cálculo, existen herramientas *software* específicas que pueden servir de soporte en algunas fases de la realización de un estudio de caso. Por ejemplo, para la realización del análisis de datos cualitativos pueden utilizarse herramientas como *NVivo* (*http://www.qsrinternational.com*) y *Atlas* (*http://www.atlasti.com*).

Una de las herramientas de gestión de la configuración y de versiones más utilizada es *Apache Subversion* (*http://subversion.apache.org/*).

A la hora de transcribir las entrevistas, hay herramientas *software* que permiten automatizar el proceso, como es el caso de *Digital Voice Editor* (*http://esupport.sony.com/p/swu-download.pl?upd_id=5529*).

Por último, puede resultar útil contar con cualquiera de los paquetes estadísticos ya comentados en el capítulo 3.

Capítulo 5

INVESTIGACIÓN - ACCIÓN

5.1 CARACTERÍSTICAS DE LA INVESTIGACIÓN-ACCIÓN

Entre los diversos métodos de investigación cualitativa existentes, la mayoría provenientes del campo de las ciencias sociales, el más utilizado en sistemas de información e ingeniería del *software* es la Investigación-Acción (IA), conocida en inglés como *Action-Research*.

Este método fue propuesto después de la Segunda Guerra Mundial por Kurt Lewin como una forma de investigación que podía enlazar el enfoque experimental de las ciencias sociales con programas de acción social que respondieran a los principales problemas sociales de entonces (Lewin, 1946). Mediante la investigación-acción, se argumentaba que se podían lograr en forma simultánea avances teóricos y cambios sociales. Este método ha obtenido una amplia aceptación y aplicación en la investigación en informática, desde que fue propuesto en el año 1985 para investigar en el área de sistemas de información (Wood-Harper, 1985). Desde principios de los años noventa se empezó a utilizar también de manera explícita en investigaciones relacionadas con la ingeniería del *software*.

Existen diversas definiciones de Investigación-Acción. Algunas de las más significativas son las siguientes:

- **Para McTaggart** (1991) es "la forma que tienen los grupos de personas de preparar las condiciones necesarias para aprender de sus propias experiencias, y hacer estas experiencias accesibles a otros".
- **Para French y Bell** (1996) es "el proceso de recopilar de forma sistemática datos de la investigación acerca de un sistema actual en relación con algún objetivo, meta o necesidad de ese sistema; de alimentar de nuevo con esos datos al sistema; de emprender acciones por medio de variables alternativas seleccionadas dentro del sistema, basándose tanto en los datos como en las hipótesis; y de evaluar los resultados de las acciones, recopilando datos adicionales".
- **Para Wadsworth** (1998) consiste en la participación de "todas las partes involucradas en la investigación, examinando la situación existente (que sienten como problemática), con los objetivos de cambiarla y mejorarla".

De las definiciones anteriores se puede deducir que la IA tiene una doble finalidad: generar un beneficio al "cliente" de la investigación y, al mismo tiempo, generar "conocimiento de investigación" relevante (Kock y Lau, 2001). Una premisa fundamental en esta forma de investigar es que los procesos sociales complejos (y el uso de tecnologías de la información en organizaciones es de este tipo) pueden ser estudiados mejor introduciendo cambios en dichos procesos y observando los efectos de dichos cambios (Baskerville, 1999).

Por tanto, la IA es una forma de investigar de carácter colaborativo que busca unir teoría y práctica entre investigadores y profesionales mediante un proceso de naturaleza cíclica, produciendo nuevos conocimientos útiles en la práctica, que se obtienen mediante el cambio y/o búsqueda de soluciones a situaciones reales que le ocurren a un grupo de profesionales (Avison *et al.*, 1999). Esto se consigue gracias a la intervención de un investigador en la realidad del mencionado grupo. Los resultados de esta experiencia deben ser beneficiosos tanto para el investigador como para los practicantes.

En la práctica, la IA no se refiere a un método de investigación concreto, sino a una clase de métodos que tienen en común las siguientes características (Baskerville, 1999):

- **Orientación a la acción y al cambio**.
- **Focalización en un problema**.

- **Un modelo de proceso "orgánico"** que engloba etapas sistemáticas y algunas veces iterativas.
- **Colaboración entre los participantes**.

5.2 PARTICIPANTES EN LA INVESTIGACIÓN-ACCIÓN

En un análisis más formal de los participantes en IA, Wadsworth (1998) identifica los siguientes cuatro tipos de roles en este método (en algunas ocasiones la misma persona o equipo puede desempeñar más de un rol):

- El **investigador**, el individuo o grupo que lleva a cabo de forma activa el proceso investigador.
- El **objeto investigado**, es decir, el problema a resolver.
- El **grupo crítico de referencia**, aquel para quien se investiga en el sentido de que tiene un problema que necesita ser resuelto y que también participa en el proceso de investigación (aunque menos activamente que el investigador). En él hay tanto personas que saben que están participando en la investigación, como otras que participan sin saberlo.
- El **beneficiario** (en inglés *stakeholder*), aquel para quien se investiga en el sentido de que puede beneficiarse del resultado de la investigación, aunque no participa directamente en el proceso. Puede ser el receptor de documentos, informes, etc. En este grupo, por ejemplo, caben tanto las empresas que se benefician de un nuevo método para resolver problemas en tecnologías de la información, como los técnicos que aplican dicha metodología.

5.3 PROCESO DE LA INVESTIGACIÓN-ACCIÓN

Un proceso de investigación que emplea IA se halla compuesto de grupos de actividades organizadas formando un ciclo característico. Padak y Padak (1994) identifican los siguientes pasos, que deben seguirse en las investigaciones que utilicen este método:

1. **Planificación**: Identificar las cuestiones relevantes, que guiarán la investigación, que deben estar directamente relacionadas con el objeto que se está investigando y ser susceptibles de encontrarles respuesta. En esta actividad se buscan caminos alternativos, líneas a seguir o reforzar algo existente. El resultado es que se definen claramente otros problemas o

situaciones a tratar. Algunos autores (Baskerville, 1997) distinguen entre diagnóstico (identificar los problemas iniciales) y planificación (especificar acciones para resolver dichos problemas).

2. **Acción**: Variación de la práctica, cuidadosa, deliberada y controlada. Se efectúa una simulación o prueba de la solución. Es cuando el investigador interviene sobre la realidad.

3. **Observación**: Recoger información, tomar datos, documentar lo que ocurre. Esta información puede proceder prácticamente de cualquier sitio (bibliografía, medidas, resultados de pruebas, observaciones, entrevistas, documentos, etc.). También se conoce como "evaluación".

4. **Reflexión**: Compartir y analizar los resultados con el resto de interesados, de tal manera que se invite al planteamiento de nuevas cuestiones relevantes y, como añade Wadsworth (1998), "a profundizar en la materia que se está investigando para proporcionar conocimientos nuevos que puedan mejorar las prácticas, modificando éstas como parte del propio proceso investigador, para luego volver a investigar sobre estas prácticas una vez modificadas". También se conoce como "especificación del aprendizaje". En algunas variantes de Investigación-Acción no es una etapa realmente, sino un proceso continuo que ocurre durante todo el tiempo.

Con estas características, el proceso definido por la IA es iterativo, de forma que se va avanzando en soluciones cada vez más refinadas mediante la completitud de ciclos, en cada uno de los cuales se ponen en marcha nuevas ideas, que son puestas en práctica y comprobadas en el ciclo siguiente, tal como se muestra en la Figura 5.1. Este ciclo caracteriza la IA como un proceso reflexivo de aprendizaje y búsqueda de soluciones. El carácter cíclico supone volver a reevaluar o replantear las acciones o caminos a seguir ponderando diagnóstico y reflexión.

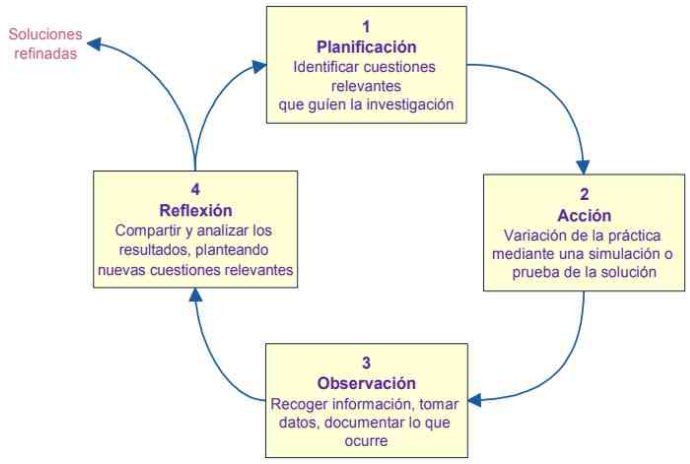


Figura 5.1. Carácter cíclico de la IA

En el campo de la ingeniería del *software*, el cliente de una investigación es normalmente una organización a la cual el investigador suministra servicios (consultoría, ayuda para implementar cambios en las prácticas de desarrollo de *software*, etc.) a cambio de tener acceso a datos de interés para la investigación y, en muchos casos, de recibir financiación (Kock y Lau, 2001). En cualquier caso, el investigador que utiliza IA en la ingeniería del *software* sirve a dos entidades diferentes: el cliente de la investigación y la comunidad científica. Las necesidades de ambos suelen ser muy diferentes y, a veces, opuestas entre sí, por lo que intentar satisfacer ambas demandas es el principal desafío al que todos los investigadores tienen que enfrentarse.

Desafortunadamente, la práctica de la IA en este campo presenta algunas debilidades:

1. La falta de método con que los investigadores y profesionales utilizan y conciben la IA
2. El contexto de consultoría utilizado, que impone una perspectiva demasiado restrictiva al implicar responsabilidades contractuales e intereses organizacionales que pueden ir en contra de lo propuesto por la IA.
3. La ausencia de un modelo de proceso de investigación definido que indique los pasos a seguir.

Todo lo anterior puede tener como consecuencia una falta de rigurosidad del proceso de investigación, por ello, para solventar estos problemas se han propuesto, entre otras, las siguientes alternativas:

- Llevar a cabo la investigación usando una perspectiva de gestión de proyectos.
- Incluir criterios de calidad especialmente concebidos.
- Analizar los factores que inciden en la formalización del proceso.
- Organizar el proceso con una estructura de proyecto.

Combinando estas ideas, Estay y Pastor (2000a) y (2000b) han propuesto "usar gestión de proyectos para mejorar el rigor de un proyecto de IA[3], lo cual se ha traducido en generar una estructura de proyecto que contenga los principales

[3] Estos autores analizan la utilización de la IA en el contexto de los sistemas de información, pero sus propuestas son perfectamente válidas en el campo de la ingeniería del software.

elementos de la IA". Para lograr lo anterior, estos autores plantean la necesidad de adoptar prácticas de gestión, adecuadas a IA-SI, basadas en el PMBOK (*Guide to the Project Management Body of Knowledge*). Se trataría de hacer corresponder la IA con el proyecto, ya que ambos son experiencias de trabajo únicas con resultados finales igualmente únicos y, además, comparten la idea de intervención, es decir, ambos suponen una alteración voluntaria de la realidad. Aunque la intervención en Investigación-Acción produce alteraciones en una práctica de trabajo, también es una forma de obtener datos de la experiencia real que son necesarios para el proceso de investigación. Los mismos autores también han propuesto un modelo de madurez basado en CMM, aplicando prácticas de gestión de proyectos de forma incremental con objeto de garantizar una mejora del rigor y calidad del uso de la IA (Estay y Pastor, 2001).

Se pueden considerar así, en el contexto de la investigación que existen dos realidades (científica/académica y práctica) que interactúan pero que se mueven en planos diferentes. La IA opera sobre esta realidad dual, que se concreta en dos tipos de ciclos de Investigación-Acción para dos tipos de proyectos:

1. **Ciclos orientados a resolver problemas dentro de proyectos de ingeniería de software**. Estos proyectos consisten en el desarrollo de una solución informática (son proyectos informáticos, de desarrollo de *software*, de implantación y/o mantenimiento de sistemas informáticos, etc.). En este caso el investigador se encarga de resolver un problema y la IA aparece como una herramienta más para el desarrollo de sistemas de información.
2. **Ciclos orientados a investigar dentro de proyectos de investigación**. Estos proyectos son esfuerzos intencionados buscando un resultado. En este caso la Investigación-Acción nos ofrece un método de trabajo y una justificación para acercarnos a una determinada realidad con fines de probar una teoría o hipótesis.

Por otro lado, la estructura de proyecto de IA-SI propuesta por Estay y Pastor (2000b) define dos ciclos característicos:

1. **Ciclo orientado a construir una solución para generar nuevo conocimiento útil a los profesionales y mejorar su práctica**. El investigador se conecta con la realidad mediante una intervención. La investigación se utiliza para construir modelos, teorías o conocimiento de manera informada e influida por la realidad. En este ciclo es el interés por resolver un problema lo que origina el interés por la investigación.

2. **Ciclo orientado a gestionar la investigación para producir nuevo conocimiento en la disciplina de ingeniería del software y mejorar la práctica de los investigadores**. En este ciclo es el interés por la investigación el que origina interés por resolver ciertos problemas.

En resumen, la IA puede analizarse desde dos dimensiones complementarias (ver Figura 5.2).

1. Una dimensión "vertical" en función del tipo de proyecto.
2. Una dimensión "horizontal" en función del bi-ciclo típico de la estructura de un proyecto de IA.

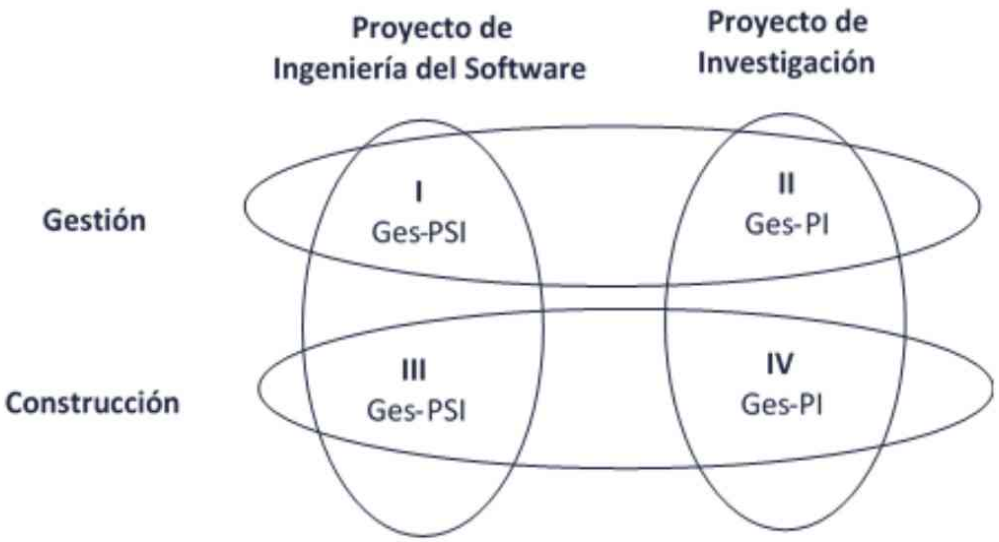


Figura 5.2. Dos dimensiones en la IA

5.4 INVESTIGACIÓN-ACCIÓN CANÓNICA

En Davison *et al.* (2004) se propone una serie de "principios" para asegurar el rigor y la relevancia en las investigaciones que utilizan la denominada la "Investigación-Acción Canónica" (en inglés, CAR; *Canonical Action Research*) en el que se propugna un modelo orientado a procesos iterativo, riguroso y colaborativo para la IA.

Estos principios conllevan una serie de criterios que pueden ser utilizados como criterios para llevar a cabo la IA:

5.4.1 Principio del Acuerdo entre Cliente e Investigador

Este principio hace hincapié en la importancia de que el cliente entienda cómo funciona la IA, para lo que se proponen los siguientes criterios:

- Acuerdo tanto del investigador como del cliente en que la IA canónica es el enfoque apropiado para la situación organizacional.
- Compromiso explícito del cliente con el proyecto.
- Especificar clara y explícitamente: el foco del proyecto de investigación, los roles y las responsabilidades de los miembros de las organizaciones cliente e investigadora, objetivos y medidas de evaluación del proyecto, métodos de recogida y análisis de datos.

5.4.2 Principio del Modelo de Procesos Cíclico

Este principio hace hincapié en el carácter cíclico de la IA, para lo que se proponen los siguientes criterios:

- Seguir el modelo cíclico o justificar cualquier desviación del mismo.
- Llevar a cabo por parte del investigador un diagnóstico independiente de la situación organizacional.
- Las acciones planificadas se basan explícitamente en los resultados del diagnóstico y se implementan y evalúan.
- Llevar a cabo por parte del investigador una reflexión sobre los resultados de la intervención.
- Como resultado de la reflexión se toma un decisión explícita de seguir o no en un ciclo adicional.
- Tanto la salida del investigador como la finalización del proyecto se debe al cumplimiento de los objetivos del proyecto o de alguna otra clara justificación.

5.4.3 Principio de la Teoría

Este principio hace hincapié en que la investigación suele estar guiada por una teoría, para lo que se establecen los criterios siguientes:

Las actividades del proyecto han sido guiadas por una o varias teorías:

- El dominio de investigación, y el establecimiento del problema específico, son relevantes y significativo tanto para la comunidad científica como para el cliente.
- Se ha usado un modelo basado en teoría para derivar las causas del problema observado.
- La intervención planificada sigue este modelo basado en teoría.
- Se ha usado una teoría para evaluar los resultados de la intervención.

5.4.4 Principio del Cambio por medio de la Acción

Este principio hace hincapié en que la esencia de la IA canónica es tomar acciones para cambiar la situación actual, para lo que se establecen los criterios siguientes:

- Tanto el cliente como el investigador están motivados para mejorar la situación.
- Se han especificado el problema y sus causas supuestas como resultado del diagnóstico.
- Se han diseñado las acciones planificadas para abordar las causas supuestas.
- El cliente ha aprobado las acciones planificadas antes de que sean implementadas.
- Se ha evaluado la situación organizacional de forma completa tanto antes como después de la intervención.
- Se han documentado clara y completamente la planificación y la naturaleza de las acciones tomadas.

5.4.5 Principio del Aprendizaje por medio de la Reflexión

Este principio hace hincapié en que la especificación explícita del aprendizaje es la actividad más crítica de la IA, para lo que se establecen los criterios siguientes:

- El investigador reporta informes de progreso al cliente y los miembros organizacionales.

- Tanto el investigador como el cliente reflexionan sobre los resultados del proyecto.
- Las actividades y resultados de la investigación se reportan clara y completamente.
- Los resultados se consideran en términos de sus implicaciones para acciones ulteriores en esta situación, acciones a tomar en dominios de investigación relacionados, para la comunidad científica general, y en cuanto a aplicabilidad general de la propia IA.

5.5 OTRAS CONSIDERACIONES DEL USO DE LA IA EN INGENIERÍA DEL SOFTWARE

La mayor parte de los trabajos de aplicación de la IA en informática se centran en el área de los Sistemas de Información. Así, en Lau (1997) se puede encontrar un resumen del uso de IA-SI, comentando diversos ejemplos publicados por diferentes autores referidos a la construcción y desarrollo de sistemas de información, que también tratan el análisis, diseño, desarrollo e implementación de *software* y a los procesos asociados.

En Baskerville (1999) se hace una introducción al uso de IA en los Sistemas de Información indicando diez formas de utilización y cuatro características que determinan dicha forma de uso: modelo de proceso (iterativo, reflexivo, linear); estructura (rigurosa, fluida); rol del investigador (colaborador, facilitador, experto); y objetivos principales (desarrollo organizacional, diseño de sistemas, conocimiento científico, entrenamiento).

En Dos Santos y Travassos (2011) se presenta una visión general sobre la utilización de la IA en la ingeniería del *software*, diferenciando tres formatos de IA: *colaboración técnica* (cuando se trata de probar una tecnología en un contexto real), *colaboración mutua* (el investigador y los participantes identifican conjuntamente los problemas, sus causas y las posibles intervenciones) y *mejora* (cuando el investigador pretende facilitar actividades de ingeniería del *software*), en los que el investigador actúa como observador, participante y facilitador, respectivamente. Estos autores destacan que la IA puede servir para abordar, de forma pragmática y sin perder rigor científico, el problema de que el conocimiento de ingeniería del *software* puede ser insuficiente en una situación dada, por ejemplo qué procedimiento adoptar en una actividad de diseño *software*.

5.6 EJEMPLO DE INVESTIGACIÓN-ACCIÓN

Como se sabe, uno de los principales problemas en el campo de la informática es el mantenimiento que supone en muchas ocasiones más del 70% del presupuesto de las organizaciones. Este problema se agudizó a finales de los noventa debido al efecto del "año 2000" y de la adopción del euro. En Polo *et al.* (2002a) y Ruiz *et al.* (2002b) se expone la utilización de IA para la construcción de un entorno metodológico y tecnológico para la gestión del proceso de mantenimiento del *software* y que se ha desarrollado en el marco de tres proyectos de I+D en colaboración con dos organizaciones externas. Los resultados principales de este proyecto, desde el punto de vista profesional se recogen en dos libros (Piattini *et al.* 1998 y 2000), y desde el punto de vista investigador en las tesis doctorales de Polo (2000) y Ruiz (2003) y sus correspondientes publicaciones científicas.

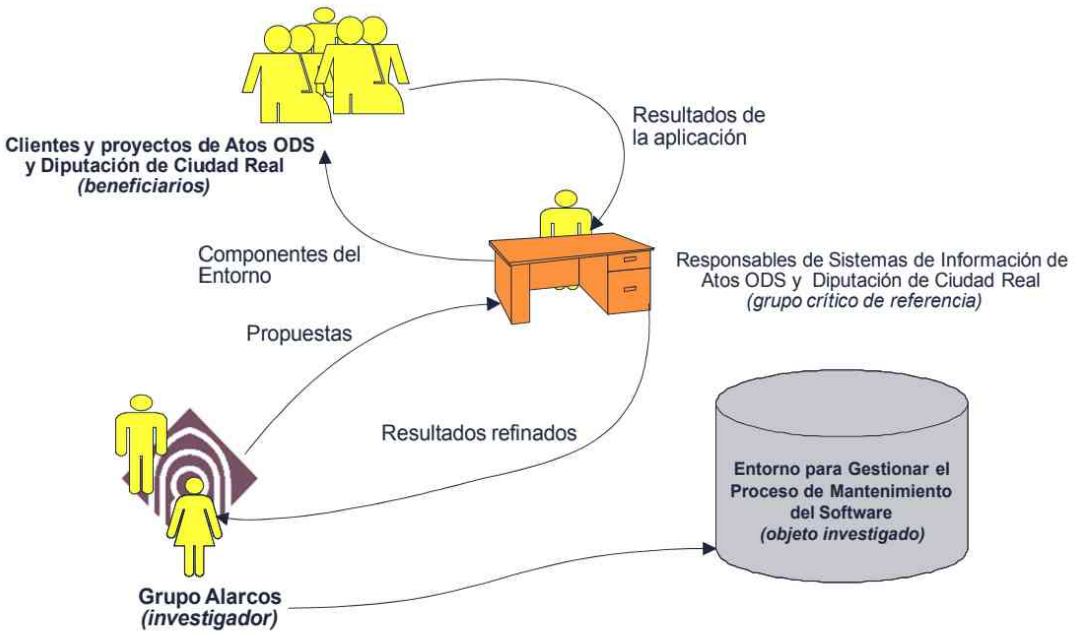


Figura 5.3. Participantes en la investigación-acción

En este trabajo, los cuatro roles diferentes ya comentados han correspondido a los siguientes participantes (en la Figura 5.3 ilustramos las relaciones entre ellos):

a) **Investigador**: el grupo de investigación Alarcos, de la Universidad de Castilla -La Mancha, en Ciudad Real.

b) **Objeto investigado**: el proceso de mantenimiento del *software* (PMS), y más en particular, un entorno para gestionar dicho proceso de forma integrada, con mayor calidad y más sencilla.

c) **Grupo crítico de referencia** (GCR): Ha estado constituido por representantes de las dos empresas que han participado en los proyectos MANTEMA, MPM y MANTIS: Atos ODS S.A. y Excma. Diputación Provincial de Ciudad Real (concretamente el CENPRI, Centro Provincial de informática).

d) **Beneficiarios**: Son las organizaciones que pueden ser beneficiadas por los resultados del trabajo, es decir, todas aquellas que poseen *software* propio o que utilizan *software* de terceros, que está sometido a un proceso de mantenimiento. Más directamente, los beneficiarios son la Diputación de Ciudad Real (que tiene *software* propio que mantener) y los clientes que contratan servicios de mantenimiento de *software* con Atos ODS.

En este caso, se considera que la definición de un entorno para la gestión del PMS es un dominio adecuado para la aplicación de IA, ya que se cumplieron las condiciones idóneas para su uso:

1. El investigador propuso un marco de trabajo teórico que fue aceptado por el grupo crítico de referencia.

2. El investigador trabajó activamente para que los beneficios fueran mutuos, científicos para el investigador y prácticos para el grupo crítico de referencia.

3. El conocimiento obtenido pudo ser aplicado de forma inmediata.

4. La investigación se desarrolló en un proceso típico cíclico e iterativo combinando teoría y práctica.

La puesta en marcha de IA durante el proceso investigador de este trabajo ha supuesto una continua realimentación entre el investigador y el grupo crítico de referencia: el primero estudiaba los problemas planteados y proponía soluciones que eran analizadas y aplicadas por el segundo en sus ambientes de trabajo real; los resultados eran debatidos en común. De esta forma, con cada ciclo (ver Figura 5.4) llevado a cabo se han ido obteniendo soluciones cada vez más refinadas generadas de forma participativa. Estas soluciones se concretaban en propuestas de componentes del Entorno MANTIS que eran analizadas y probadas. Los resultados servían para refinar y mejorar los componentes del Entorno o para incluir nuevos componentes.

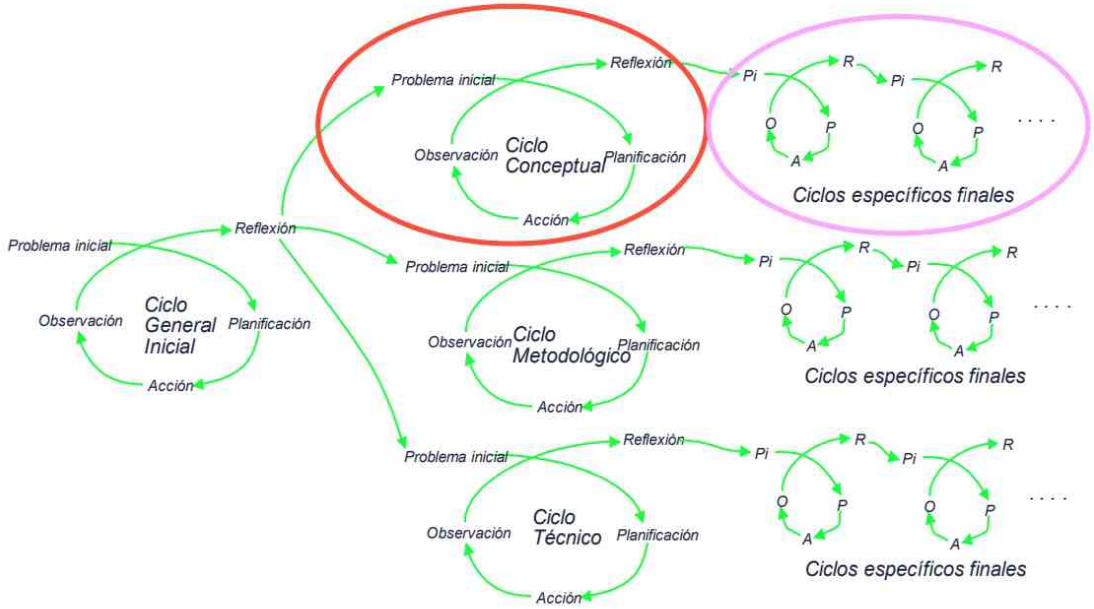


Figura 5.4. Estructura multiciclo con bifurcación utilizada en MANTIS

Este proceso se puede resumir en los siguientes ciclos (ver Figura 5.4):

- **Ciclo general inicial**: Los investigadores y el grupo cíclico de referencia definieron la problemática general de la gestión del proceso de y puesta mantenimiento del *software* y establecieron los objetivos y requisitos generales del entorno necesario (planificación). Se procedió a la búsqueda de toda la información de posible interés al respecto (acción). Su análisis posterior (observación) permitió descubrir que el objeto de estudio tenía una complejidad importante dado que debían tenerse en cuenta múltiples aspectos de naturaleza diferente. La compartición en conjunto (reflexión) entre los investigadores y el grupo crítico de reflexión permitió detectar que las posibles soluciones generales debían consistir en la integración de diversas soluciones parciales a problemas parciales. En resumen, se decidió considerar el Entorno MANTIS como una colección de herramientas de tres tipos diferentes: conceptuales, metodológicas y técnicas (*software*). Este ciclo supuso un estudio muy amplio de los diferentes aspectos que influyen en la gestión de un proceso *software*, en general, y del proceso de mantenimiento, en particular: tecnología de proceso *software*, entornos de ingeniería del *software*, modelado y metamodelado de procesos, mejora de procesos, arquitecturas conceptuales para modelado de procesos y para integración de herramientas *software*, modelos de ciclo de vida del *software*, modelos de gestión de proyectos, etc.

- **Ciclos generales intermedios**: Para cada uno de los tres tipos de herramientas anteriores se realizó un ciclo de IA, con las cuatro etapas conocidas, que pretendía dar respuesta a las siguientes preguntas:

  – *Ciclo Conceptual*: ¿Qué se necesita para manejar la complejidad de la gestión del PMS?; ¿Cómo representar toda la información necesaria en un entorno para gestionar el PMS?. En este ciclo se empezó trabajando con el objetivo de aclarar la terminología utilizada en PMS, terminándose por definir una serie de ontologías (ver Figura 5.5) que pueden encontrarse en Ruiz *et al.* (2003) y García *et al.* (2006).

Figura 5.5. Ontologías del entorno MANTIS

  – *Ciclo Metodológico*: ¿Qué métodos/técnicas son útiles para gestionar el PMS?. En los que se empezó aplicando el estándar ISO 12207 (Polo *et al.*, 1999) e incorporándole actividades relativas al *outsourcing* (Polo *et al.*, 2002b). Más recientemente incluso se ha creado una versión ágil de la metodología de mantenimiento (Pino *et al.*, 2011).

  – *Ciclo Técnico*: ¿Qué herramientas *software* son útiles para gestionar el PMS?, ¿Cuáles de ellas interesa desarrollar ex profeso?. De hecho se desarrollaron un conjunto importante de herramientas como puede verse en la figura 5.6, más detalles pueden encontrarse en las tesis anteriormente citadas y en Polo *et al.* (2001) y Ruiz *et al.* (2002a y 2002b).

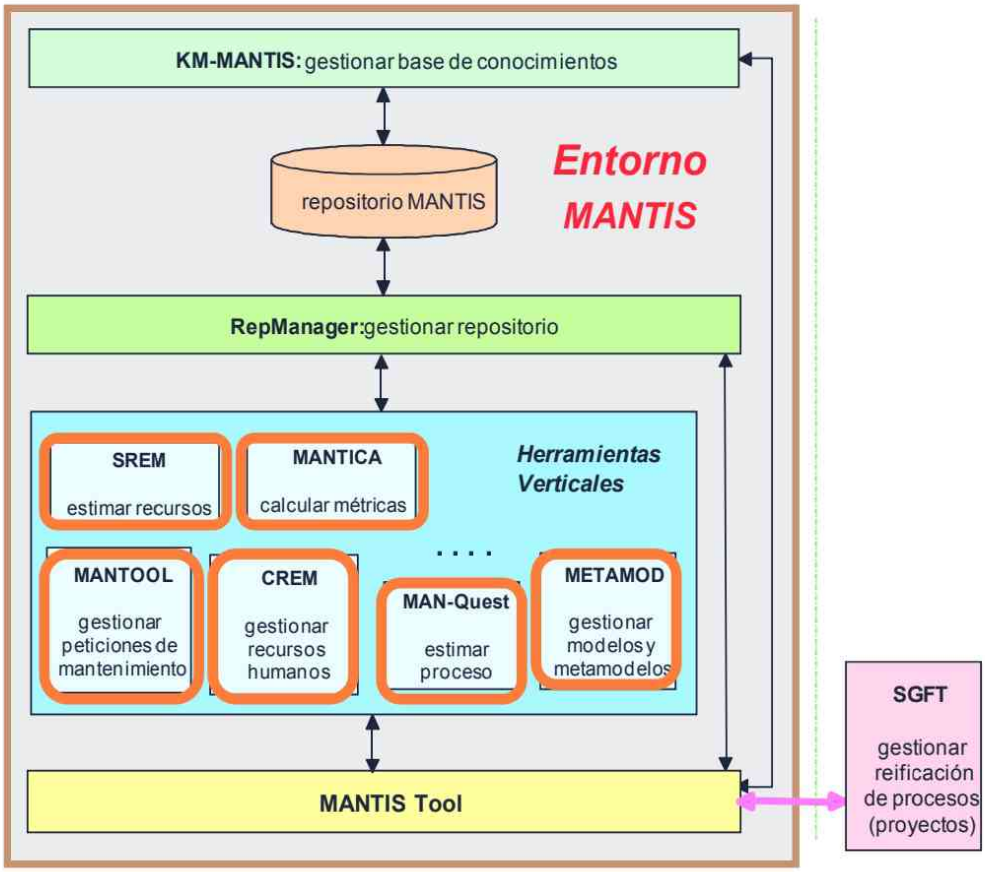


Figura 5.6. Herramientas del entorno MANTIS

- **Ciclos específicos finales**: A partir del momento en que las respuestas anteriores quedaron claras, tanto para los investigadores como para el grupo crítico de referencia, se procedió a realizar ciclos específicos de IA para cada uno de los componentes individuales del Entorno MANTIS (perteneciente a cualquiera de los tres tipos de herramientas). Los ciclos anteriores significan que para la definición del Entorno MANTIS se ha utilizado IA con una estructura de proyecto multicíclica con bifurcación (ver Figura 5.4), que es la más conveniente en los casos de nuevos ciclos resultado de nuevos subproblemas y/o cuando se generan nuevos problemas (McNiff, 1988).

La definición del marco de trabajo general para el Entorno MANTIS (con sus tres tipos de herramientas conceptuales, metodológicas y técnicas) facilitó la gestión de la complejidad del trabajo de investigación y ayudó a la incorporación más activa del grupo crítico de referencia, ya que pudieron dedicarse personas diferentes de las dos organizaciones implicadas en diferentes ciclos específicos finales correspondientes a componentes diferentes de MANTIS.

Un problema encontrado en la realización de los diversos ciclos, fue la existencia de dependencias no conocidas a priori entre unos ciclos y otros. Por ejemplo, para poderse completar el ciclo metodológico se necesitó llegar a un cierto nivel de refinamiento en el ciclo conceptual. Igualmente, el ciclo técnico necesitó un determinado número de iteraciones en los ciclos conceptual y metodológico. En la realización de los ciclos específicos finales de cada componente del Entorno MANTIS, también se descubrieron dependencias que hicieron que el orden temporal en el que fueron avanzando dichos ciclos cambiase continuamente a lo largo del proceso de investigación. Incluso se produjeron algunos casos en que, en mitad del ciclo de un determinado componente, se descubrió la necesidad de añadir a MANTIS un nuevo componente de igual o diferente tipo. Por ejemplo, al realizar una etapa de reflexión en una de las iteraciones del ciclo específico para elegir y definir el componente "arquitectura conceptual" se detectó la necesidad de disponer de una herramienta *software* que permitiera trabajar con modelos y metamodelos de procesos *software* de forma integrada a diferentes niveles de abstracción. Al no descubrir ninguna herramienta en el mercado que cumpliera dichos requisitos se decidió incluir un componente técnico nuevo en el Entorno MANTIS para tal fin, llamado MANTIS-Metamod. Estas situaciones supusieron una realimentación de los ciclos intermedios en el sentido de que la realización de sus ciclos específicos finales supusieron con cierta frecuencia una nueva iteración del ciclo intermedio correspondiente. A lo largo del trabajo de investigación también se han conseguido resultados no previstos inicialmente. Por ejemplo, el reto principal de definir el Entorno MANTIS se abordó en diferentes pasos que, después de diversos refinamientos, permitieron obtener el resultado deseado. La reflexión posterior sobre el camino seguido nos llevó a la conclusión de que habíamos acabado generando una metodología para la definición de entornos de ingeniería del *software*.

En la Tabla 5.1 se muestra un resumen de los hechos más importantes acontecidos durante la realización de los tres proyectos citados. Se han clasificado según el aspecto al que se refieren (de entre todos los abordados en los citados tres proyectos) y se indica si participaron en su realización el investigador (Grupo Alarcos) y el grupo crítico de referencia (GCR), es decir, Atos ODS y/o Diputación de Ciudad Real. En dicha tabla se puede tener una perspectiva global del trabajo llevado a cabo en estos proyectos. También se puede comprobar que los tres proyectos han sido realmente subproyectos de un proyecto global cuyo objetivo general ha sido mejorar la realización y gestión del proceso de mantenimiento del *software* (PMS).

Fecha año-mes	**Aspecto**	**Hecho**	**Roles participantes**	
			GCR	**Invest.**
1997-07		Inicio del proyecto MANTEMA.	X	X
-09	General PMS	Recopilación y estudio de información sobre el PMS.		X
-10	Metodología	Planteamiento de una estructura inicial de la metodología.		X
-11	Metodología	Reuniones técnicas para refinar la estructura inicial.	X	X
-12	Metodología	Propuesta de separar cada tipo de mantenimiento.	X	
1998-01	Metodología	Guía para mantenimiento correctivo.		X
-02	Metodología	Aplicación de esta guía a proyectos reales.	X	
-03	Metodología	Definición de guías para el resto de tipos de mantenimiento.		X
-05	Metodología	División del correctivo en urgente y no urgente.	X	
-05	Metodología	Presentación de una guía de perfectivo.		X
-06	Metodología	Estudio y comentarios a la guía de perfectivo.	X	
-06	Metodología	Presentación de MANTEMA 1 (5 guías técnicas).		X
-07	Metodología	Aplicación de MANTEMA 1 a proyectos reales.	X	
1999-01	Metodología	Estudio de MANTEMA 1 para corregir defectos y añadir nuevas características.		X
-03	Mejora	Búsqueda en la literatura de técnicas de apoyo para la ejecución del proceso.		X
-03	Metodología	Construcción de plantillas para los documentos de mantenimiento.		X
-04	Metodología	Informe de fallos descubiertos durante la aplicación de MANTEMA 1.	X	
-05	Metodología	Petición de integrar los mantenimientos correctivo no urgente, perfectivo, preventivo y adaptativo en un solo tipo (*planificable*), dejando aparte el correctivo urgente (*no planificable*).	X	
-06		Inicio del proyecto MPM.	X	X
-06	Herr. Técnicas	Prototipo de herramienta para gestionar peticiones de modificación (MANTOOL).	X	X
-07	Metodología	Presentación de MANTEMA 2.		X
-09	Metodología	Aplicación de MANTEMA 2 a proyectos reales.	X	

-09	Mejora	Estudio de propuestas existentes para la mejora del PMS.		X
-10		Fin del proyecto MANTEMA.	X	X
-11	Auditoria Control	Propuesta de objetivos de control para auditoria del PMS.		X
-12	Gestión Riesgos	Planteamiento de guía para identificar y estimar riesgos.		X
2000-01		Inicio del proyecto MANTIS.	X	X
-01	Metodología	Refinamiento de MANTEMA 2.	X	X
-02	Entorno MANTIS	Estudio de las propuestas sobre entornos de ingeniería del software.		X
-02	Gestión Riesgos	Refinamiento de guía para identificar y estimar riesgos.	X	X
-03	Metodología	Elaboración de MANTEMA 2.1.		X
-04	Metodología	Prueba de MANTEMA 2.1 en proyectos reales de 4GL y PYMES.	X	
-04	Entorno MANTIS	Propuesta de *framework* para el Entorno MANTIS con componentes de tres tipos: conceptuales, metodológicos y técnicos.		X
-04	Medida	Propuesta de métricas para la estimación del esfuerzo de mantenimiento.		X
-05	Entorno MANTIS	Revisión conjunta de los tres tipos de componentes del Entorno.	X	X
-05	Medida	Refinamiento de métricas para la estimación del esfuerzo de mantenimiento.	X	X
-06	Medida	Propuesta de métricas para la gestión del PMS (reparto de las cargas de trabajo).		X
-07	Arq. Conceptual	Arquitectura conceptual de MANTIS basada en MOF.		X
-07	Medida	Refinamiento de métricas para la gestión del PMS.	X	X
-09	Ontologías	Integración en MANTIS de la ontología del PMS.		X
-09	Flujos de Trabajo	Propuesta de integrar en MANTIS datos sobre la ejecución real de proyectos de mantenimiento.	X	
-10	Mejora	Propuesta de cuestionario basado en CMM para la mejora del PMX.		X
-11	Flujos de Trabajo	Propuesta de utilizar flujos de trabajo para representar la ejecución de los proyectos.		X

-11	Medida	Refinamiento de la propuesta integrada de métricas para el PMS.	X	X
-12	Metamodelos	Propuesta de metamodelo de procesos software general para MANTIS.		X
-12	Herr. Técnicas	Definición de la arquitectura software del Entorno MANTIS.		X
2001-01	Interfaces	Estudio de interfaces metodológicos para integrar los procesos organizacionales y gerenciales (según ISO 15504) en el Entorno MANTIS.		X
-01	Flujos de Trabajo	Refinamiento del uso de flujos de trabajo en el Entorno MANTIS (incluir ontología de los flujos de trabajo).	X	X
-02	Roles y actores	Estudio de la integración de aspectos organizativos (roles, organizaciones y responsabilidades) en el Entorno MANTIS.		X
-02	Mejora	Prueba del cuestionario para la mejora del PMS en proyectos reales.	X	
-03	Ontologías	Ontología de la medida.		X
-03	Medida	Propuesta de integración del proceso de medida en el Entorno MANTIS.		X
-04	Entorno MANTIS	Propuesta general del Entorno MANTIS.		X
-05	Entorno MANTIS	Revisión conjunta de la propuesta de Entorno MANTIS.	X	X
-06	Gestión Proyectos	Propuesta de utilizar PMBOK como modelo para la gestión de proyectos.		X
-06	Herr. Técnicas	Revisión conjunta de la arquitectura software del Entorno MANTIS.	X	X
-06	Mejora	Refinamiento del modelo de mejora del proceso de mantenimiento incluyendo aspectos organizativos y las métricas de gestión y estimación del esfuerzo.	X	X
-07	Mejora	Refinamiento del cuestionario para la mejora del PMS.	X	X
-07	Herr. Técnicas	Propuesta de Gestor del Repositorio (RepManager) de datos y metadatos basado en XMI.		X
-07	Herr. Técnicas	Propuesta de gestor de modelos y metamodelos (Metamod).		X
-07	Interfaces	Refinamiento de la integración de los procesos organizacionales y gerenciales.	X	X

-09	Metamodelo PMS	Versión ampliada del metamodelo de proceso software general para MANTIS incluyendo aspectos de medida y los flujos de trabajo.		X
-09	Herr. Técnicas	Petición de integrar MANTOOL con el Entorno MANTIS.	X	X
-10		Fin del proyecto MPM.	X	X
-10	Herr. Técnicas	Prototipo de Gestor del Repositorio (RepManager).		X
-11	Herr. Técnicas	Propuesta de utilizar Sistemas de Gestión de Flujos de Trabajo comerciales para la reificación de los proyectos.	X	
-11	Herr. Técnicas	Prototipo de gestor de modelos y metamodelos (Metamod).		X
-11	Herr. Técnicas	Propuesta de interfaz integrado para el Entorno MANTIS: herramienta MANTIS-Tool.		X
-12	Entorno MANTIS	Refinamiento de los componentes del Entorno MANTIS.	X	X
-12		Fin del proyecto MANTIS.	X	X

Tabla 5.1. Desarrollo de la IA en los proyectos MANTEMA: MPM y MANTIS

Como se puede concluir revisando la tabla anterior, la aplicación del método de IA ha permitido abordar la complejidad del trabajo a desarrollar mediante refinamientos sucesivos, avanzando desde propuestas generales a otras más concretas, llevando a cabo varios ciclos típicos de planificación-acción-observación-reflexión.

5.7 INVESTIGACIÓN-ACCIÓN TÉCNICA

Recientemente se ha propuesto (Wieringa y Morah, 2012) una variante de la IA, denominada IA "Técnica" (o en inglés, TAR, *Technical Action Research*) como una fusión entre la ciencia del diseño y la IA, que partiendo de un artefacto busca la forma de validarlo. Por lo tanto, es un tipo de investigación, "dirigida por artefacto" en contraposición a la IA tradicional ("dirigida por problema"). En la IA técnica, se crea un artefacto y se empieza llevando a cabo una prueba de concepto (en inglés *proof of concept*), luego probándolo mediante problemas pequeños ("*de juguete*") en circunstancias ideales, y posteriormente, se escalan las condiciones para resolver problemas más realistas, hasta que puede probarse en organizaciones para resolver problemas concretos (ver figura 5.7).

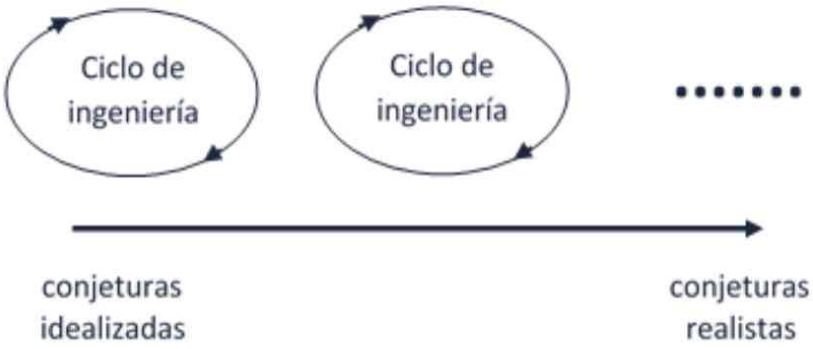


Figura 5.7. Iteraciones en la IA técnica (*Wieringa y Morah*: *2012*)

En la IA técnica es importante distinguir entre el desarrollo del artefacto por parte del investigador, la adquisición de conocimiento por el investigador y la solución del problema del cliente. La adquisición de conocimiento corresponderá a la formulación y contestación de preguntas de investigación específicas que deberán ser evaluadas respecto al grado de certeza de las respuestas. Por su parte, la solución del problema del cliente requerirá la identificación de los *stakeholders* relevantes junto con los objetivos que persiguen, y será evaluada teniendo en cuenta si se produce una mejora respecto a su efectividad y utilidad.

En cada ciclo de ingeniería se pueden distinguir cuatro actividades (ver Figura 5.8):

- **Investigación del problema**, que incluye la identificación de los *stakeholders* y sus objetivos con sus correspondientes criterios; la investigación de los fenómenos relevantes para el problema, y la evaluación de cómo estos fenómenos concuerdan con los objetivos de los *stakeholders*.

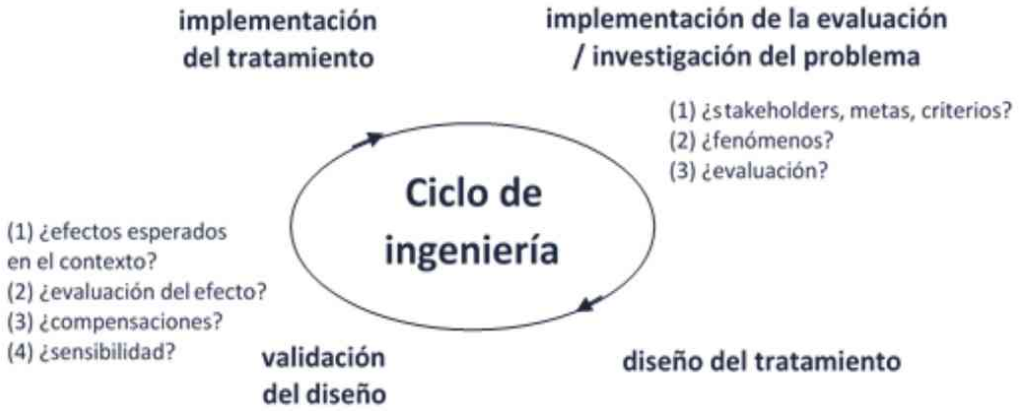


Figura 5.8. Ciclo de ingeniería (*Wieringa y Morah*: *2012*)

- **Diseño del tratamiento**, en este caso el tratamiento consiste en un artefacto que interactúa con el contexto del problema (ver Figura 5.9). En este caso los *stakeholders* son las personas (físicas o jurídicas) afectadas por los artefactos y que son parte del contexto, mientras que los profesionales son las personas que diseñan tratamientos particulares para problemas concretos o que intentan resolver un problema particular.

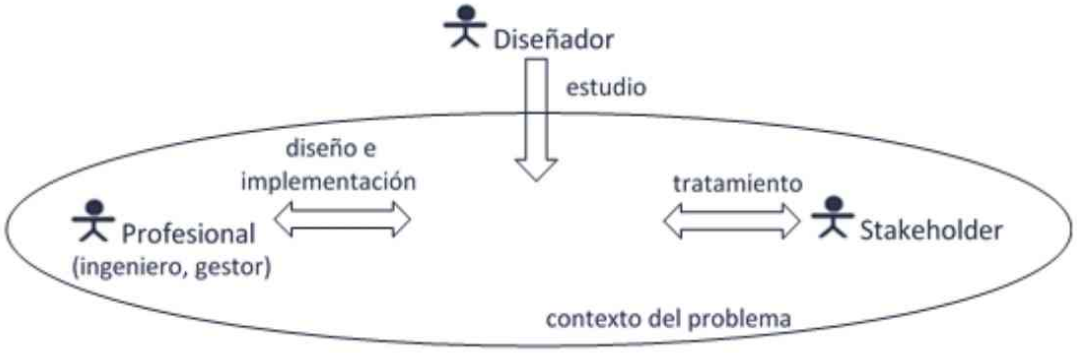


Figura 5.9. Tratamientos y artefactos en la IA técnica (*Wieringa y Morah*: *2012*)

- **Validación del diseño**, actividad que trata de responder a dos cuestiones: efectos esperados del artefacto en el contexto del problema, y valor esperado (en el sentido de como los efectos satisfacen los criterios), Antes de la implementación hay que plantearse también como se comporta este tratamiento comparado con otros posibles, y la sensibilidad del tratamiento, es decir, analizar si sería efectivo y útil si el problema cambia.
- **Implementación del tratamiento y evaluación**, que consiste en la transferencia al entorno y la evaluación respecto a sus *stakeholders*, efectos, valor y sensibilidad al contexto del problema.

5.8 EJEMPLO DE INVESTIGACIÓN-ACCIÓN TÉCNICA

En esta sección presentamos un ejemplo de investigación-acción técnica realizado en el contexto de la arqueología de procesos de negocio, que es la actividad de ingeniería que estudia los procesos de negocio de una organización mediante el análisis de los artefactos *software* existentes en la organización (Pérez-Castillo *et al.*, 2011b). En Pérez-Castillo *et al.* (2011a) se presenta el diseño y construcción de una herramienta *software* para el descubrimiento de procesos de negocio desde sistemas de información heredados denominada MARBLE4, que se

[4] *http://marketplace.eclipse.org/content/marble*

distribuye como una aplicación eclipse bajo licencia (EPL). Desde el punto de vista del problema de investigación, los resultados obtenidos fueron una tesis doctoral (Pérez-Castillo, 2012) y sus correspondientes publicaciones científicas.

5.8.1 Ciclos de IA Técnica en MARBLE

En el diseño, construcción y validación de MARBLE se empleó IA Técnica a lo largo de cuatro ciclos (ver Figura 5.10).

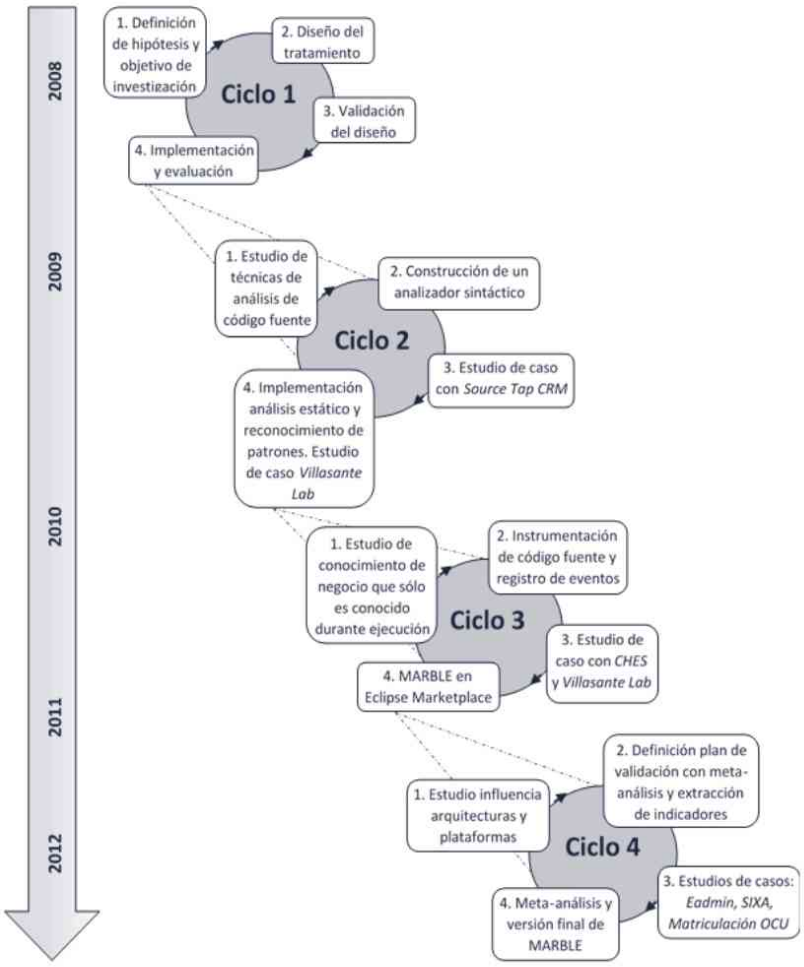


Figura 5.10. Ciclos durante el desarrollo de MARBLE siguiendo la IA Técnica

5.8.1.1 PRIMER CICLO

En este ciclo se construye una primera prueba de concepto y se propone MARBLE como un marco genérico para la modernización *software*. La

investigación del problema en la fase 1 de este ciclo consistió en el análisis preliminar del estado del arte mediante la realización de una revisión sistemática de la literatura y la posterior definición de la hipótesis y el objetivo de investigación. La hipótesis de investigación fue que era factible la construcción de un marco basado en el enfoque ADM (*Architecture Driven Modernization*) para recuperar los procesos de negocio embebidos en diferentes tipos de sistemas de información heredados de forma estandarizada y automática. A partir de esta hipótesis se estableció el objetivo de investigación como: "construir un marco basado en ADM con el cual descubrir y reconstituir modelos de procesos de negocio desde sistemas de información existentes".

En la fase 2, el diseño del tratamiento consistió en la definición de un marco genérico y extensible de acuerdo al enfoque ADM para la ingeniería inversa de sistemas de información heredados hacia modelos de procesos de negocio. El marco genérico está basado en el estándar KDM (en inglés *Knowledge Discovery Metamodel*) (ISO/IEC, 2012), que permite realizar representaciones conceptuales abstractas de las diferentes vistas de la arquitectura de los sistemas heredados. Posteriormente, ese conocimiento se transforma y depura progresivamente hasta llegar a los procesos de negocio (BP, *Business Processes*) subyacentes. Para ello, MARBLE se dividió en cuatro niveles de abstracción definiendo las transformaciones entre ellos (ver Figura 5.11).

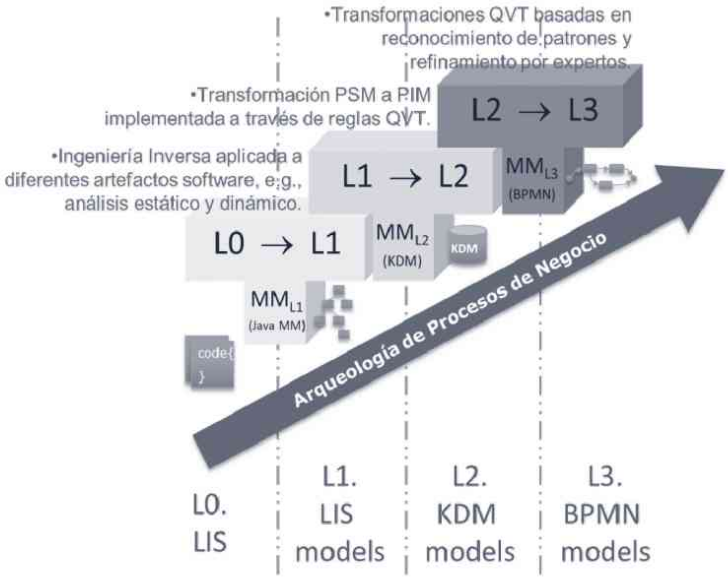


Figura 5.11. Marco genérico y extensible definido en MARBLE

En la tercera fase del primer ciclo, la validación del diseño fue soportada por los beneficios que de forma teórica aporta la adopción del enfoque ADM junto con el uso del estándar KDM. En la cuarta fase de implementación y evaluación se

implementó un prototipo de la herramienta MARBLE como una aplicación de escritorio que permitía analizar código Java y obtener información preliminar de los modelos de procesos de negocio candidatos.

5.8.1.2 SEGUNDO CICLO

En el segundo ciclo del proyecto, de acuerdo a la metodología de IA Técnica se llevó a cabo la construcción de una herramienta para la extracción de procesos de negocio mediante análisis estático de código Java. Esta herramienta se validó empíricamente con de sistemas de ejemplo de tamaño pequeño. La fase de investigación del problema consistió en la definición de una técnica enmarcada dentro del marco genérico de MARBLE para soportar el análisis estático de código fuente heredado y generación de modelos de procesos de negocio. En la segunda fase, el diseño del tratamiento consistió en la construcción de un parser de código java capaz de analizar sintácticamente línea a línea código java heredado y construir un árbol sintáctico abstracto. Este árbol abstracto era después trasformado a un modelo KDM independiente de la plataforma. Finalmente, se diseñó una técnica para el reconocimiento de patrones en modelos KDM y la generación de sub-partes de un diagrama de procesos de negocio. Este parser de código java y la técnica de *pattern matching* son considerados como el artefacto bajo estudio en el segundo ciclo.

En la tercera fase, la validación del diseño anterior consistió en la aplicación a un *Source TAP CRM* (Source Tap, 2009), un pequeño sistema CRM (*Customer Relationship Management*) de código abierto de 8.800 líneas de código. En la fase de implementación y evaluación se refinó la implementación y se incorporó al prototipo que se construyó en el ciclo anterior resultando en una primera versión completa de la herramienta MARBLE. Adicionalmente, se aplicó a otro caso a fin de validar empíricamente la técnica basada en análisis estático de código fuente usando una aplicación web del laboratorio Villasante, dedicado al análisis químico de aguas potables y residuales (Pérez-Castillo *et al.*, 2011c).

5.8.1.3 TERCER CICLO

El tercer ciclo consistió en la incorporación a MARBLE de una técnica de ingeniería inversa basada en el análisis dinámico, es decir, el análisis de sistemas durante su ejecución. La validación se realizó con sistemas reales de tamaño medio. La investigación del problema se centró en detectar y controlar los efectos de ignorar la información de negocio embebida en los sistemas de información heredados que sólo puede ser conocida en tiempo de ejecución. Para ello se estudiaron diferentes posibilidades para obtener información durante la ejecución y su posterior análisis para recuperar procesos de negocio.

La fase de diseño del tratamiento permitió la definición de una técnica consistente en la instrumentación previa del código fuente heredado mediante análisis estático, y la posterior obtención de registros de eventos durante la ejecución de las trazas inyectadas en el código fuente original (Pérez-Castillo *et al.*, 2010). En la fase de validación se utilizó la herramienta en sistemas de información reales, como el sistema CHES de evaluación oncológica usado en varios hospitales de Austria (Pérez-Castillo *et al.*, 2011c). En la última fase, se implementó la técnica de análisis dinámico a la vez que la herramienta MARBLE se migró a un desarrollo orientado a *plug-ins* para Eclipse. Este cambio de arquitectura facilitó la futura extensibilidad y aplicabilidad de MARBLE. La primera versión de MARBLE como aplicación Eclipse se lanzó y se publicó bajo licencia EPL en el repositorio *Eclipse Marketplace*.

5.8.1.4 CUARTO CICLO

En el último ciclo se llevó a cabo un refinamiento de MARBLE y se usó en entornos industriales con grandes sistemas en explotación. En la fase de investigación del problema se analizó los efectos que podían tener las diferentes arquitecturas y plataformas tecnológicas del sistema de información heredado sobre el cual se extraían los procesos de negocio con MARBLE.

Para ello, se diseño en la segunda fase de este ciclo un plan intensivo de validación en la que los últimos refinamientos de MARBLE se probaron con sistemas industriales de tamaño considerable, además de definir el posterior meta-análisis que intenta de sacar conclusiones combinando los resultados de las aplicaciones anteriores.

En la validación de dicho diseño se emplearon tres sistemas adicionales: (1) Eadmin Xunta, una aplicación web encargada de la administración electrónica regional de la consejería de vivienda; (2) SIXA, un sistema de información de apoyo a la docencia implantado en institutos de secundaria; y (3) Una aplicación web para la matriculación universitaria de varias universidades españolas. Todos estos sistemas tenían un tamaño de entre 100.000 y 300.000 líneas de código. El meta-análisis permitió demostrar que los resultados obtenidos seguían la misma tendencia (Pérez-Castillo *et al.*, 2012), la cual fue además positiva de acuerdo a los valores de las medidas esperados. Dicho meta-análisis permitió demostrar la aplicabilidad de MARBLE y facilitar su adopción en la industria.

En la cuarta fase, de implementación y evaluación, se lanzó la segunda versión de *MARBLE Eclipse application*, que aglutinaba pequeñas mejoras y correcciones de errores. MARBLE 2.0 siguió siendo distribuida bajo licencia EPL en *Eclipse Marketplace*.

5.9 LECTURAS RECOMENDADAS

No existe un libro específico que explique cómo aplicar IA en el ámbito de la ingeniería del *software*, pero sí algunos que, aunque se aplican a otros ámbitos, pueden dar algunas sugerencias válidas.

- **Coghlan**, **D**. y **Brannick**, **D**. (2010). *Doing Action Research in your own Organization*: London: Sage. Un libro muy completo que explora los fundamentos, la implementación y los desafíos de la utilización de la IA en las organizaciones.
- **Johnson**, **A**. **P**. (2011). *A short guide to action research*. Allyn and Bacon. Se trata de una guía muy práctica para aplicar la IA.
- **Davison**, **R**., **Martinsons**, **M**., **Ou**, **C**. (2012). *The roles of theory in canonical action research*. MIS Quarterly: 36(3), 763-786. En este artículo se revisa el papel que tienen tanto la teoría focal (la que proporciona la base intelectual para el cambio orientado a la acción) como la teoría instrumental (la que se utiliza para explicar el fenómeno, y facilitar el diagnóstico, el planeamiento y la evaluación) en un proyecto de IA.

5.10 SITIOS WEB RECOMENDADOS

- *http://wwwhome.cs.utwente.nl/~roelw/*

  Se trata del sitio web del profesor Roel Wieringa de la Universidad de Twente, en el que se pueden encontrar varios trabajos relacionados con la IA Técnica y otras reflexiones sobre la investigación en ingeniería del *software*.

5.11 HERRAMIENTAS RECOMENDADAS

No existen herramientas específicas que puedan ayudar a llevar a cabo un proyecto de IA, pero podría valer cualquiera de las usuales en la gestión de proyectos (*MSProject*, *OpenProj, Taskjuggler, GanttProject*, etc.).

Capítulo 6

REVISIONES SISTEMÁTICAS DE LA LITERATURA

6.1 CARACTERÍSTICAS

Una revisión sistemática de la literatura, conocida por sus siglas en inglés SLR (*Systematic Literature Review*), es un medio para de identificar, evaluar e interpretar toda la investigación disponible para responder a unas preguntas de investigación específicas (Kitchenham y Charters, 2007). Una SLR se considera un *estudio secundario* y cada uno de los estudios individuales recopilados en una SLR se denomina *estudio primario*.

Unas de las motivaciones más importantes para la realización de SLRs es la obtención de nuevos hallazgos y la proposición de ideas innovadoras para futuras investigaciones (Zhang y Ali Babar, 2013). Las SLRs difieren principalmente de las revisiones de la literatura tradicionales (que podríamos denominar *Traditional Literature Reviews* o TLRs) por que se planifican formalmente y se ejecutan de manera sistemática y metódica. Una buena revisión sistemática debe ser replicable de forma independiente, por lo que tendrá más valor científico que el de una TLR. Al encontrar, evaluar y resumir toda la evidencia disponible sobre un tema específico de investigación, una SLR puede proporcionar un mayor nivel de validez de sus conclusiones de lo que sería posible en cualquiera de los estudios primarios analizados en la SLR.

Las SLRs han sido más influyentes que las TLRs atendiendo al número de citas que han tenido (Zhang y Ali Babar, 2013), ya que se consideran un instrumento de investigación fiable para recopilar la evidencia empírica sobre un tema específico.

Si bien las SLRs tienen numerosas ventajas, su desventaja es que consumen más tiempo y esfuerzo, por lo que uno de los principales retos de la realización de las SLRs es lograr un equilibrio entre el rigor metodológico y el esfuerzo requerido.

En otras disciplinas como la medicina, es una práctica común realizar SLRs, por ello cuentan con procedimientos bien establecidos para llevarlas a cabo (Sackett *et al.*, 2000; Higgins y Green, 2011). Lo mismo no ocurría en la ingeniería del *software* hasta que Barbara Kitchenham propuso en el año 2004 una primera versión de las directrices que presentan aspectos metodológicos sobre cómo realizar SLRs en la ingeniería del *software* (Kitchenham, 2004), posteriormente mejoradas en el año 2007 (Kitchenham y Charters, 2007), basadas en las existentes en el campo de la medicina.

A partir de entonces el interés sobre el uso de las SLRs como un método de investigación ha ido creciendo vertiginosamente, como lo demuestran los cientos de SLRs publicadas desde el año 2004 en el ámbito de la ingeniería del *software*. Además se han llevado a cabo numerosas investigaciones con el objetivo de mejorar la infraestructura científica y metodológica para dar soporte a las SLRs (Kitchenham y Beretron, 2013. Por ejemplo, sobre cómo mejorar el procedimiento de búsqueda (Kitchenham *et al.*, 2010; Zhang *et al.*, 2011), cómo realizar la evaluación de la calidad de los estudios primarios (Dybå y Dingsøyr, 2008a; Kitchenham *et al.*, 2010b), sobre la evaluación de la fiabilidad de las SLRs como método de investigación (MacDonell *et al.*, 2010), sobre el uso y adopción de las SLRs en la ingeniería del *software* y los retos a los que se enfrentan los investigadores al realizar SLRs (Zhang y Ali Babar, 2013), etc. También se han publicado lecciones aprendidas tras haber realizado SLRs, con el objetivo de compartir conocimientos y experiencias (Brereton *et al.*, 2007; Dybå et al, 2007b; Staples y Niazi, 2007).

Conviene puntualizar antes de continuar, que si bien el objetivo principal de las SLRs es sintetizar la evidencia empírica existente sobre un tema de interés en la ingeniería del *software*, las directrices de Kitchenham y Charters (2007) pueden aplicarse a otras disciplinas y también para recopilar toda la literatura

existente sobre un tema de interés aunque el objetivo no sea meramente la búsqueda de evidencia empírica. Por ello consideramos que el proceso descrito en este capítulo puede ser de gran utilidad especialmente para doctorandos que necesiten realizar el estado del arte del tema que están investigando de manera rigurosa y exhaustiva, guiados evidentemente por sus supervisores o directores.

6.2 PROCESO PARA REALIZAR UNA SLR

Kitchenham y Charters (2007) proponen un proceso para realizar SLRs que consta de tres actividades principales, cada una de ellas con varias tareas secuenciales (ver Figura 6.1), de las que se suele ser necesario realizar varias iteraciones, sobre todo para refinar el protocolo de revisión.

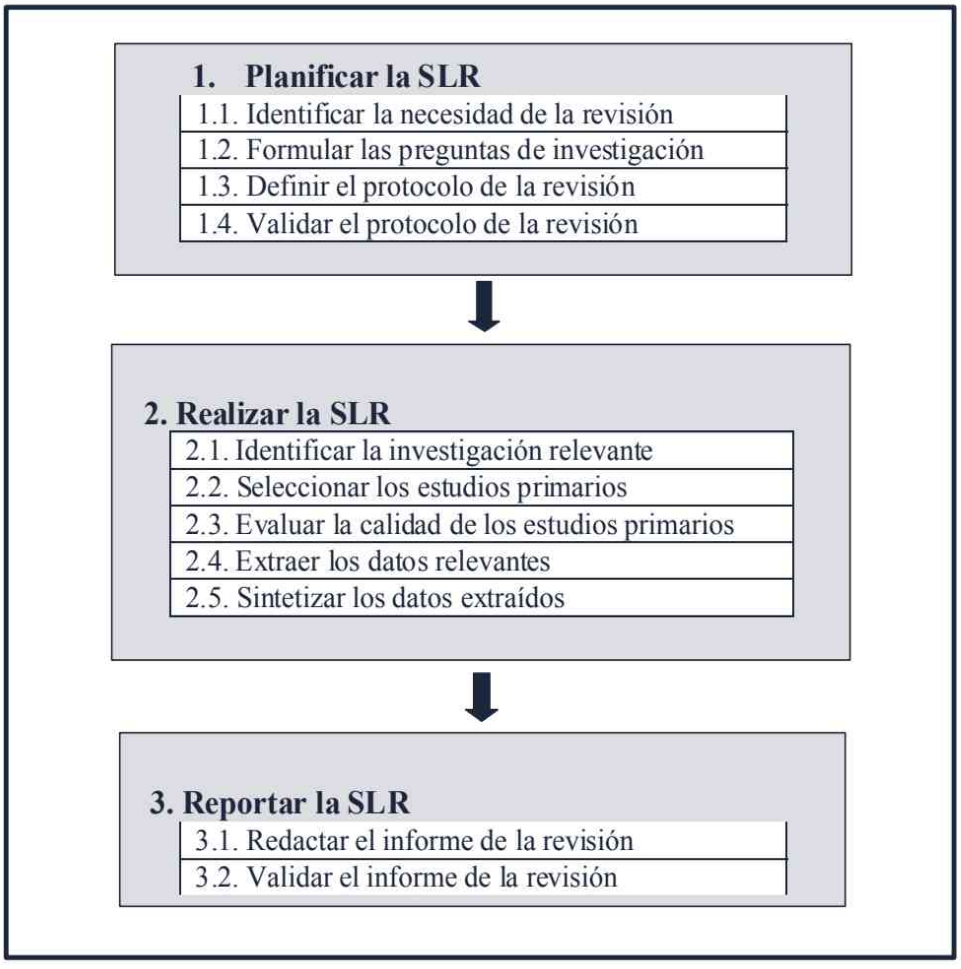


Figura 6.1. Proceso para realizar SLRs

6.2.1 Planificar la revisión

La planificación es una actividad crucial porque de las decisiones tomadas en esta actividad dependerá el correcto desarrollo de la SLR. El principal objetivo de la planificación es especificar todos los aspectos que harán que la revisión sea sistemática y rigurosa, evitando en la manera de lo posible posibles sesgos o ambigüedades, y que se detallan en lo que se denomina *protocolo de la revisión*.

La planificación se divide en cuatro tareas que se detallan a continuación.

6.2.1.1 IDENTIFICAR LA NECESIDAD DE LA REVISIÓN

La necesidad de realizar una SLR surge de la necesidad de resumir cuidadosamente toda la información relevante sobre un tema de interés. Kitchenham y Charters (2007) aconsejan que se busque en primer lugar otras revisiones sistemáticas existentes sobre el tema. Si se encuentran otras revisiones, habrá que analizarlas para determinar si es justificable realizar otra más y además su lectura ayudará en la definición del protocolo de la nueva revisión.

6.2.1.2 FORMULAR LAS PREGUNTAS DE INVESTIGACIÓN

La especificación de las preguntas de investigación es la parte más importante de cualquier revisión sistemática. Estas preguntas dirigirán todo el proceso de revisión:

- En el proceso de búsqueda se deben seleccionar los estudios primarios que sirvan para responder a las preguntas de investigación.
- En el proceso de extracción se deben extraer los datos necesarios para responder a las preguntas de investigación.
- El proceso de análisis de datos debe sintetizar los datos de tal manera que las preguntas pueden ser respondidas.

Algunos de los tipos de preguntas que se pueden responder realizando SLRs en la ingeniería del *software* suelen servir para:

- Evaluar el efecto de una tecnología.
- Evaluar la frecuencia o proporción de un factor de desarrollo de proyectos, tales como la adopción de una tecnología, o la frecuencia o tasa de éxito o fracaso del proyecto.

- Identificar los costes y los factores de riesgo asociados con una tecnología.
- Identificar el impacto de las tecnologías en los modelos de fiabilidad, rendimiento y costes.
- Analizar los costes y beneficios del empleo de tecnologías de desarrollo de *software* específicas.

Algunos aspectos a tener en cuenta en la redacción de las preguntas de investigación incluyen:

- La **población** en la que se recoge la evidencia, por ejemplo: un papel específico de los profesionales en la ingeniería del *software* (programadores, analistas, etc.), una categoría de ingenieros de *software* (considerando su nivel de experiencia), un área de aplicación o un grupo industrial como compañías de telecomunicación o pequeñas empresas dedicadas a las tecnologías de la información, entre otras.
- La **intervención** aplicada en el estudio empírico, es decir la tecnología, herramienta o metodología que se estudia.
- La **comparación**, es la tecnología, herramienta o metodología con la que se compara la intervención. Cuando la tecnología de comparación es la tecnología convencional o de uso común, se conoce como el tratamiento de "control".
- Los **resultados** deben estar relacionados con factores de importancia para los profesionales como la mejora de la fiabilidad, reducción de costes de producción y de tiempo de aparición en el mercado.
- El **contexto** en el que se realiza la comparación (academia o industria), las personas que participan en el estudio (profesionales, estudiantes, etc.) y el tipo de tareas que se deben realizar en el estudio empírico.
- El **tipo de diseño** de los experimentos que se tendrán en cuenta.

6.2.1.3 DEFINIR EL PROTOCOLO DE REVISIÓN

Un protocolo de revisión sistemática es un plan formal y bastante concreto para llevar a cabo la revisión sistemática, que debe definirse para reducir la posibilidad de sesgos. Kitchenham y Charters (2007) recomiendan hacer pruebas durante la definición del protocolo, por ejemplo con respecto a la estrategia de búsqueda y de extracción de datos.

Como la definición del protocolo suele no ser una tarea sencilla y requiere cierta destreza, es conveniente ver cómo se ha definido el protocolo en artículos que hayan publicado SLRs, como por ejemplo los mencionados en la Tabla 6.16 (en el apartado 6.5). También se recomienda leer el protocolo detallado de la SLR publicada en Beechman *et al.* (2008), que se encuentra disponible en *http://uhra.herts.ac.uk/bitstream/handle/2299/989/S70.pdf?sequence=1*

A continuación, se describen con un mayor nivel de detalle los elementos que debe tener el protocolo de revisión.

Antecedentes

Se debe justificar la relevancia de la necesidad de llevar a cabo la revisión. Para ello se debe utilizar información extraída en una tarea previa: "Identificar la necesidad de la revisión".

Preguntas de investigación

En esta parte del protocolo se incluirán las preguntas de investigación definidas previamente.

Estrategia de búsqueda

La definición de la estrategia de búsqueda requiere definir la cadena de búsqueda, el período de búsqueda y decidir las fuentes de búsqueda. La definición de la cadena de búsqueda no es una tarea simple, y suele requerir varias iteraciones hasta que se define la cadena definitiva. La cadena de búsqueda es recomendable que se defina en inglés, debido a que es el lenguaje internacional utilizado en la investigación, aunque, en casos muy justificados sea necesario incluir otro lenguaje. Como aconsejan Brereton *et al.* (2007) se pueden seguir los siguientes pasos para definir la cadena de búsqueda:

- Definir los términos principales.
- Identificar palabras alternativas, sinónimos o términos relacionados con los términos principales.
- Usar el operador lógico OR para incluir las palabras alternativas, sinónimos y términos relacionados.
- Usar el operador lógico AND para enlazar los términos principales.

Antes de definir la cadena de búsqueda es conveniente contactar con expertos en el tema de interés para que nos proporcionen artículos relevantes y luego leer estos artículos para sacar ideas de que términos podrían incluirse en la cadena de búsqueda. Los términos principales también pueden salir de los diferentes elementos que se incluyen en las preguntas de investigación (población, intervención, comparación, etc.). Además de la cadena de búsqueda se debe indicar en qué parte de la publicación se va a buscar, por ejemplo: título, resumen, palabras claves, texto completo. Generalmente se busca en el título y resumen, si el buscador lo permite, y en caso contrario en el texto completo.

Es aconsejable definir cadenas de búsquedas iniciales, aplicarlas en alguna fuente de búsqueda y detectar leyendo el título y el resumen si realmente los artículos encontrados responden a los objetivos de la SLR. Si el número de artículos encontrados es muy grande, puede ser necesario precisar más la cadena de búsqueda.

Las fuentes de búsqueda pueden ser:

- **Bibliotecas digitales**.
- **Revistas**, **conferencias**, **talleres**, etc.
- **Literatura gris**: artículos, reportes técnicos, libros, etc. que normalmente no se encuentran en las bibliotecas digitales y que suelen proporcionar los expertos.

Algunos investigadores prefieren en lugar de buscar en bibliotecas digitales, orientar la búsqueda a revistas y congresos específicos sobre el tema de interés. Esto puede ser viable en el caso de que los investigadores sean expertos en el tema, pero en cualquier caso esta decisión debe justificarse adecuadamente.

También se debe considerar que puede existir un sesgo en las publicaciones, debido a que los resultados positivos son más propensos a ser publicados que los resultados negativos. Para evitar este sesgo, se puede buscar en literatura gris, en actas de congresos y talleres y contactando con expertos preguntándoles si tienen resultados no publicados.

Intuitivamente se piensa que es mejor hacer las búsquedas de manera automática para consumir menos tiempo y para poder incluir mayor número de fuentes. Aunque también se dice que las búsquedas manuales suelen ser más precisas y que dan como resultados artículos de mayor calidad (Kitchenham *et al.*, 2010b; Zhang *et al.*, 2011). Nosotros, creemos que ambas alternativas son complementarias y que se justifican adecuadamente contribuirán a obtener el mayor número posible de artículos relevantes. También puede ser necesario buscar

en la lista de las referencias de los artículos encontrados, sobre todo cuando se encuentran muy pocos artículos relevantes, efecto que se conoce como "bola de nieve" (*snowballing*). Wohlin (2014) explica cómo hacer la búsqueda en la lista de referencias de manera sistemática.

En resumen, la tarea de identificación de la literatura relevante para una SLR es crucial y es fundamental que todas las decisiones tomadas durante la definición de la estrategia de búsqueda sean debidamente justificadas y documentadas. Y es fundamental contar con expertos que nos asesoren a la hora definir la estrategia de búsqueda, y en el caso los doctorandos podrá ser sus directores quienes los asesoren.

Criterios de selección de estudios

Los criterios de inclusión y exclusión de los estudios primarios se deben definir con el objetivo de identificar los estudios primarios que muestren evidencia con respecto a las preguntas de investigación. Con el fin de reducir la probabilidad de sesgo, los criterios de selección se deben establecer durante la definición de protocolo, a pesar de que pueden posteriormente ser refinados durante el proceso de búsqueda. También se debe especificar dentro de los criterios de exclusión, la exclusión de estudios duplicados en caso de que se encontrasen dos estudios publicados en diferentes artículos. Se suele decidir excluir el que esté menos completo.

Procedimiento para la selección de estudios

En el protocolo se debe indicar, por ejemplo, si se aplicarán los criterios de inclusión/exclusión sólo teniendo en cuenta los resúmenes o leyendo el artículo completo. Debido a que la selección de artículos es una tarea que puede consumir mucho tiempo, conviene proponer como procedimiento hacer una primera selección leyendo los resúmenes y en caso de duda leer el artículo completo.

También se debe mencionar quienes serán las personas responsables de aplicar los criterios de inclusión/exclusión. Lo más común suele ser que una sola persona se encargue de esto y que luego otras seleccionen varios artículos aleatoriamente y verifiquen si su inclusión o exclusión se ha realizado correctamente. En caso de que hubiera discrepancias conviene mantener una reunión para resolverlas.

Es conveniente además mantener una lista de los artículos tanto incluidos como excluidos para hacer más transparente el proceso de selección, indicando los motivos que llevaron a incluirlos/excluirlos.

Listas de comprobación y procedimiento para la evaluación de la calidad de los estudios

Además de la definición de criterios de inclusión/exclusión, es importante evaluar la calidad de los estudios primarios con el fin de:

- Proporcionar criterios aún más detallados de inclusión /exclusión.
- Investigar si las diferencias en la calidad proporcionan una justificación para explicar las diferencias en los resultados del estudio.
- Ponderar la importancia de los estudios individuales cuando se sintetizan los resultados.
- Orientar la interpretación de los resultados y determinar la fortaleza o validez de inferencias realizadas.
- Dar recomendaciones para futuras investigaciones.

La evaluación de la calidad se suele hacer a través de listas de comprobación de los factores que deben ser evaluados para cada estudio. Si a cada ítem se le asigna una escala numérica, se puede hacer una evaluación numérica de cada estudio. Lamentablemente no existe un estándar para la evaluación de estudios y la mayoría de las SLRs publicadas hasta 2012 o no evalúan la calidad de los estudios primarios o no lo hacen adecuadamente (Kitchenham y Brereton, 2013). Existen algunos ejemplos de listas de comprobación para evaluar la calidad de los estudios y de su aplicación que se pueden tener en cuenta (Kitchenham y Charters, 2007; Dybå y Dingsøyr, 2008a y 2008b; Fernández-Sáez *et al.*, 2013b; Kitchenham *et al.*, 2012, 2013). Hay que tener en cuenta que no todos los tipos de estudios empíricos (experimentos, estudios de casos, encuestas, etc.) se pueden evaluar de igual manera, por lo que puede ser necesario incluir ítems de evaluación específicos para cada tipo de estudio, aunque aún no se ha propuesto un mecanismo para evaluar la calidad de estudios primarios que incluyan varios tipos de estudios empíricos (Kitchenham y Brereton, 2013). Como parte de la evaluación de la calidad es recomendable además definir algún mecanismo para evaluar la fortaleza, rigor y relevancia de la evidencia que presenta cada estudio (Dybå y Dingsøyr, 2008a; Ivarsson y Gorschek, 2011).

En el caso de que el objetivo de la SLR no se restrinja a la búsqueda de evidencia empírica, los criterios para evaluar su calidad pueden ser muy diversos, como se puede observar en Fernández *et al.* (2011) y Pino *et al.* (2008) o bien la justificación de su calidad se limita a decir que son publicaciones sujetas a un riguroso proceso de revisión (como en el caso de revistas indexadas o congresos de prestigio con revisión por pares).

Estrategia para la extracción de los datos

Para extraer toda la información relevante que sirva para responder a las preguntas de investigación formuladas, es necesario diseñar un formulario de extracción de datos y también puede ser necesario definir un esquema de clasificación. En el formulario de extracción de datos es recomendable incluir: 1) Los metadatos de la publicación, como título, autores, año, tipo de publicación, etc., 2) Campos específicos que sirvan para responder a las preguntas de investigación o para clasificar los artículos de acuerdo al esquema de clasificación definido y 3) Campos que incluyan los ítems definidos en la lista de comprobación para evaluar la calidad. La extracción de datos y la clasificación de los estudios primarios se realizan leyendo el texto completo. Se deberá especificar quiénes serán los responsables de realizar la extracción de datos y quiénes de controlar que la extracción de datos se hace correctamente. Es aconsejable que al menos un investigador escoja aleatoriamente algunos artículos y corrobore si están bien clasificados o no. En el caso de alumnos de doctorado la extracción de datos la hacen los alumnos y sus directores controlan que se haya hecho correctamente. Se debe indicar también cómo se resolverán las discrepancias por ejemplo en la clasificación de artículos.

Síntesis de los datos extraídos

Para la construcción de conocimiento y para llegar a conclusiones sobre la fortaleza de la evidencia empírica sobre algún tema de interés, es necesario comparar y contrastar la evidencia de varios estudios. Por lo tanto, la síntesis de la investigación es fundamental en la ingeniería de *software*, aunque es una de las tareas al realizar las SLRs que más esfuerzo requiere y a la que menos atención se le ha prestado (Cruzes y Dyba, 2011a).

En una SLR la síntesis de los datos se realiza para dar respuesta a las preguntas de investigación formuladas. La síntesis engloba una familia de métodos que se utilizan para sintetizar, integrar, combinar y comparar los resultados de

diferentes estudios sobre un tema específico o pregunta de investigación (Cooper *et al.*, 2009; Dixon-Woods *et al.*, 2005, Noblit y Hare, 1988). Estos métodos encarnan la idea de hacer un nuevo "todo" a partir de las "partes", con el fin de proporcionar nuevos conceptos e interpretaciones de orden superior, nuevos marcos explicativos, argumentos, teorías nuevas o mejoradas o conclusiones. Tales síntesis también pueden servir para identificar áreas cruciales y preguntas para futuros estudios que no se han tratado adecuadamente con la investigación empírica actual.

Si los estudios primarios tienen variables independientes y dependientes similares, puede ser posible agregarlos a través de meta-análisis, que utiliza métodos estadísticos para combinar los tamaños del efecto. Sin embargo, en la ingeniería del *software* los estudios primarios son a menudo demasiado heterogéneos para permitir una síntesis estadística y, en particular, para los estudios cualitativos o estudios que usan varios métodos de investigación, se necesitan diferentes métodos de síntesis de la investigación (Dyba *et al*, 2007), como los presentados en la Tabla 6.1 (Cruzes y Dyba, 2011a), entre otros.

Las características del meta-análisis y un ejemplo de su aplicación se describen en detalle en el capítulo 3, y a continuación introduciremos algunas ideas sobre los métodos de síntesis cualitativos y mixtos.

En Cruzes y Dybå (2011a) se presenta un estudio terciario en el cual se recopilaron 49 SLRs publicadas entre 2005 y julio de 2010, con el objetivo de analizar los métodos de síntesis utilizados. En este estudio más de la mitad de las revisiones localizadas no contenía ningún tipo de síntesis, y aquellos que sí contenían síntesis, habían realizado en su mayoría una síntesis narrativa o temática. Muy pocas las SLRs presentaron un enfoque sólido y riguroso en cuanto a la síntesis de los resultados obtenidos. Ejemplos de algunos métodos de síntesis en el campo de la ingeniería del *software* serían: Pino *et al.*, (2008) y Bjørnson y Dingsøyr (2008) de síntesis narrativa; Dybå y Dingsøyr (2008) de metaetnografía; Beecham *et al.* (2008) y Staples y Niazi (2008) de análisis temático; Ivarsson y Gorschek (2009) de encuesta del caso; y Kitchenham *et al.* (2007) y Turner *et al.* (2010) de análisis comparativo.

Los autores concluyen que, si bien el interés por las SLRs ha ido creciendo, se ha prestado muy poca atención a la síntesis de las investigaciones en el ámbito de la ingeniería del *software*. Esta tendencia tiene que cambiar y para aumentar la importancia y utilidad de la SLRs para la investigación y la práctica, es necesario contar con un repertorio de métodos de síntesis incluyendo ejemplos de aplicación, para que puedan ser utilizados de manera rigurosa al realizar las SLRs.

Método de síntesis	Descripción
Síntesis narrativa (*narrative synthesis*) Rodgers *et al.* (2009)	Se refiere a la realización de un resumen narrativo (en lugar de estadístico) de los resultados obtenidos en los estudios primarios. Se trata de un marco general de descripciones narrativas y ordenación de la evidencia primaria con comentarios e interpretaciones, combinada con herramientas y técnicas específicas que ayudan a aumentar la transparencia y fiabilidad. La síntesis narrativa se puede aplicar tanto a revisiones de estudios cuantitativos o cualitativos.
Meta-etnografía (*meta-ethnography*) Noblit y Hare (1988)	Su objetivo es sintetizar por inducción, interpretación y análisis de la traducción de los estudios primarios con el fin de comprender y obtener ideas, conceptos y metáforas a través de diferentes estudios. Las interpretaciones y explicaciones en los estudios primarios se tratan como datos y se traducen a través de varios estudios para producir una síntesis.
Teoría fundamentada en datos (*grounded theory*) Corbin y Strauss (2008)	La teoría fundamentada en datos es un enfoque de la investigación primaria que describe los métodos de muestreo cualitativo, recopilación de datos y análisis de datos. Incluye fases simultáneas de recogida de datos y análisis, el uso del método de comparación constante, el uso de muestreo teórico, y la generación de una nueva teoría. Trata a los reportes de los estudios como datos sobre los cuáles se puede hacer un análisis para generar para generar temas de orden superior e interpretaciones.
Análisis cruzado de casos (*cross-case analysis*) Miles y Huberman (1994)	El análisis cruzado de casos incluye el uso tablas y gráficos para gestionar y presentar datos cualitativos, sin perder el significado de los mismos, a través de la codificación intensiva. Incluye meta-matrices para dividir y agrupar datos de varias maneras. Los datos de cada estudio primario se resumen y se codifican en temas generales. La evidencia se resume a continuación, considerando la evidencia dentro de cada tema. También se estudian las similitudes y diferencias entre los estudios sintetizados.
Análisis temático (*thematic analysis*) Braun y Clarke (2006); Cruzes y Dybå (2011b)	El análisis temático es un método para identificar y analizar patrones (temas) dentro de los datos. Organiza y describe el conjunto de datos con todo lujo de detalles y con frecuencia interpreta diversos aspectos del tema de investigación. El análisis temático se puede utilizar dentro de diferentes marcos teóricos, y puede ser un método realista que reporta experiencias, significados, y la realidad de los participantes. También puede ser un método constructivista, que examina las maneras en que los acontecimientos, realidades, significados, experiencia, y otros aspectos afectan a la gama de discursos. El análisis temático tiene limitado poder interpretativo más allá de la mera descripción, si no se utiliza dentro de un marco teórico existente.

Análisis de contenido (*content analysis*) Franzosi (2010)	El análisis de contenido es una forma sistemática de clasificación y codificación de los estudios en temas generales, mediante el uso de herramientas de extracción diseñados para ayudar a la reproducibilidad. Las ocurrencias dentro de cada tema se cuentan y se tabulan. Las frecuencias de los datos se determinan en base a especificaciones precisas de las categorías y a la aplicación sistemática de las reglas pertinentes. Sin embargo, las técnicas de conteo de frecuencias del análisis de contenido puede no reflejar la estructura o la importancia del fenómeno subyacente, y los resultados se pueden simplificar y contar los componentes que son fáciles de clasificar en lugar de los que son realmente importantes.
Encuesta del caso (*case survey*) Yin y Heald (1975)	Es un proceso formal para la codificación de forma sistemática de los datos relevantes de un gran número de estudios de casos para el análisis cuantitativo. Se utiliza un conjunto de preguntas cerradas estructuradas para extraer datos de manera que las respuestas puedan ser agregadas para su posterior análisis. Los datos cualitativos se convierten en cuantitativos, por ello se pueden sintetizar de ambas maneras cuantitativa y cualitativamente. Cada estudio primario se trata como un caso específico. Los hallazgos y propiedades de cada estudio se obtienen mediante las preguntas mencionadas y los datos se analizan mediante métodos de análisis de encuestas.
Análisis comparativo (*comparative analysis*) Ragin (1987)	El método de análisis comparativo cualitativo es un método de síntesis mixta que analiza las conexiones causales complejas, utilizando lógica booleana para explicar los resultados sobre la base de una tabla de verdad. El análisis booleano de condiciones necesarias y suficientes para los resultados particulares se basa en la presencia/ausencia de las variables y los resultados independientes en cada estudio primario.
Síntesis agregada (*aggregated synthesis*) Estabrooks *et al.* (1994)	La síntesis agregada es un proceso interpretativo que contiene elementos tanto de la teoría fundamentada como de la meta-etnografía. Trata de mantener el contexto de la investigación original, mejorando de la generalización de los estudios originales mediante la construcción de teorías de rango medio. Por lo tanto, el objetivo de la síntesis agregada es el desarrollo de teorías y la acumulación de conocimientos, lo que pueden explicar y predecir ciertos comportamientos.
Síntesis realista (*realist synthesis*) Pawson *et al.* (2005)	La síntesis realista es una aproximación basada en teoría que abarca la investigación cuantitativa y /o cualitativa de cualquier tipo de evidencia, centrándose en explicar cómo funcionan estas intervenciones y por qué fallan en determinados contextos. La extracción de los datos en la síntesis realista, consiste en un cuestionario de preguntas de referencia para obtener información sobre lo que funciona, para quién y en qué circunstancias. La teoría que subyace en una intervención particular es fundamental para este método.

Meta-resumen cualitativo (*qualitative metasummary*) Sandelowski y Barroso (2007)	El meta-resumen cualitativo consiste en una agregación cuantitativa de resultados cualitativos, cuyo objetivo es discernir la frecuencia de cada resultado y buscar en los resultados con valores más altos, la evidencia de los fundamentos de la replicación para la validez de la investigación cuantitativa y para afirmar haber descubierto un patrón o tema en la investigación cualitativa.
Meta-síntesiss cualitativa (*qualitative metasynthesis*) Sandelowski y Barroso (2007)	La meta-síntesis cualitativa es una integración interpretativa de los resultados cualitativos que se encuentran en forma de descripciones conceptuales temáticas o explicaciones interpretativas. Este método proporciona interpretaciones nuevas de los resultados derivadas de considerar todos los estudios realizados en su conjunto. La validez no reside en la lógica de replicación, sino más bien en la interpretación.
Meta-estudio (*meta-study*) Paterson *et al.* (2001)	Consiste en el análisis de teorías, métodos y hallazgos en la investigación cualitativa, así como la síntesis de estas ideas en nuevas formas de pensar acerca de un tema de interés. El objetivo es transformar la acumulación de los resultados en un cuerpo legítimo de conocimiento con el objetivo de generar teoría y contribuir a la práctica. Este método es único en la medida en que se centra en la importancia de la comprensión de los resultados en cuanto a los métodos y teorías que los impulsan.

Tabla 6.1. Resumen de métodos de síntesis cualitativos y mixtos

Estrategia de divulgación

Se deberá especificar en qué foros (reportes técnicos, revistas, conferencias, páginas web, etc.) se dará difusión a los resultados obtenidos. De acuerdo a la audiencia en la que puedan tener impacto los resultados obtenidos se considerarán foros académicos o industriales.

Calendario del proyecto

Es conveniente hacer una tabla con la planificación temporal de cada una de las actividades e intentar cumplirla para evitar retrasos, aunque es cierto que a priori sin saber el volumen de artículos a revisar no es del todo precisa la estimación que puede realizarse en las etapas iniciales de la revisión. En dicho caso será necesario ajustar dicha planificación.

6.2.1.4 VALIDAR EL PROTOCOLO DE REVISIÓN

Como el protocolo es un elemento crítico para la realización de SLRs, es conveniente que sea evaluado por expertos. Si se cuenta con presupuesto sería conveniente contratar a un grupo de investigadores independientes para la

evaluación del protocolo. Estos mismos expertos pueden ser quienes luego validen el reporte de la SLR. Los alumnos de doctorado pueden presentar el protocolo a sus directores para que lo evalúen.

6.2.2 Realizar la revisión

En esta actividad se pone en práctica todo lo planificado previamente en el protocolo y se obtienen los resultados finales que responderán a las preguntas de investigación. Además es fundamental documentar todas las incidencias y decisiones ocurridas durante las tareas realizadas durante la ejecución de la revisión. Esto hará que la SLR sea replicable y que todas las decisiones tomadas estén disponibles para revisores externos y para quienes deseen utilizar los resultados obtenidos. A continuación, se describen las cinco tareas de esta actividad.

6.2.2.1 IDENTIFICAR LA INVESTIGACIÓN RELEVANTE

El conjunto de publicaciones relevantes para responder a las preguntas de investigación se encuentran siguiendo la estrategia de búsqueda definida en el protocolo.

Debido a las limitaciones que presentan los buscadores de las bibliotecas digitales (Brereton *et al.* 2007), en muchos casos es necesario refinar o adaptar las cadenas de búsqueda. También puede ser necesario refinar las cadenas de búsqueda, incluir nuevas fuentes o cambiar el período de búsqueda ateniendo a descubrimientos realizados al hacer las búsquedas. Por ello es importante documentar las modificaciones realizadas en la estrategia de búsqueda y guardar los resultados a través de sistemas de gestión de referencias como *EndNote*, *BibTex*, etc. Además de los metadatos de cada artículo (título, autores, año, etc.) es importante guardar el resumen.

En esta tarea también se deben detectar los artículos duplicados, encontrados en múltiples fuentes y eliminarlos.

6.2.2.2 SELECCIONAR LOS ESTUDIOS PRIMARIOS

El proceso de selección debe localizar los estudios primarios que muestren evidencia relacionada con las preguntas de investigación. Este proceso debe también seguir lo planificado en el protocolo siguiendo los criterios y el procedimiento para la selección de estudios establecidos.

En el caso de que alguno de los estudios primarios no estén disponibles se podrá contactar con los propios autores o bien solicitar ayuda a otros investigadores que puedan tener acceso a ellos.

Como resultado de esta actividad, se deberá obtener la lista de estudios primarios seleccionados, almacenada en algún sistema de gestión de referencias incluyendo además los ficheros con los artículos en formato electrónico y la lista de estudios no incluidos y la justificación de su exclusión.

6.2.2.3 EVALUAR LA CALIDAD DE LOS ESTUDIOS PRIMARIOS

Una vez seleccionados los estudios primarios se someterá a los mismos a un proceso de evaluación de su calidad siguiendo aplicando la lista de comprobación definida en el protocolo. Como resultado de esta tarea puede ser necesario excluir aquellos artículos que no superen el umbral establecido para considerarlos estudios de calidad.

6.2.2.4 EXTRAER LOS DATOS RELEVANTES

En esta tarea se rellenará el formulario de extracción de datos definido en el protocolo. Es conveniente que la extracción de datos sea validada por otro investigador, al menos seleccionando aleatoriamente algunos estudios intentando resolver las discrepancias en caso de que las hubiera.

Como resultado de esta tarea se obtendrán los formularios de extracción de datos rellenos con la información correspondiente a cada estudio primario seleccionado.

Puede ocurrir que en el proceso de extracción de datos se encuentren duplicados y se tenga que excluir algún estudio. En este caso se seleccionará la publicación más completa. Por ejemplo es frecuente que un experimento se publique en una conferencia y luego el experimento y las réplicas se publiquen, a modo de familia de experimentos, en una revista, por lo que habrá que excluir el estudio publicado inicialmente en una conferencia.

6.2.2.5 SINTETIZAR LOS DATOS EXTRAÍDOS

Una vez recopilados en el formulario de extracción de datos todos los datos relevantes para cada estudio primario se procederá a sintetizarlos utilizando los

métodos establecidos en el protocolo, para dar respuesta a las preguntas formuladas. La síntesis es comúnmente acompañada de tablas y gráficos para ilustrar los resultados.

6.2.3 Reportar la revisión

Finalmente, se realizará un informe que refleje todo el proceso de revisión considerando el medio de divulgación seleccionado al definir el protocolo. En (Kitchenham y Charters, 2007) se presentan la estructura y el contenido recomendables para redactar el reporte de la revisión. Por restricciones de espacio, puede ser necesario complementar la publicación con un reporte técnico indicando la página web donde esté accesible, para incluir información relevante pero muy extensa, como pueden ser la definición completa del protocolo, la lista de estudios primarios, los formularios de extracción de datos rellenos con la información de cada estudio, la evaluación de la calidad de cada estudio, etc.

También es recomendable incluir en el reporte de la revisión apartados relacionados con: 1) Amenazas a la validez, en el que se comenten las limitaciones del proceso de realización de la SLR y 2) Lecciones aprendidas, que intenten reflejar experiencias recogidas durante el proceso de la realización de la SLR y que puedan servir como experiencia para futuros investigadores.

Mientras más información haya disponible, más transparencia se dará a la SLR para los revisores externos o para quienes quieran usar los resultados obtenidos.

Una vez realizado el informe es importante enviarlo a expertos para validarlo, que pueden ser los mismos expertos o directores de doctorandos que validaron el protocolo.

6.3 OTROS TIPOS DE REVISIONES

Existen otros dos tipos de revisiones de la literatura que complementan a las SLRs: los mapeos sistemáticos de la literatura y las revisiones terciarias. Ambos tipos de estudios pueden realizarse siguiendo las directrices de Kitchenham y Charters (2007).

6.3.1 Mapeos sistemáticos de la literatura

Los mapeos sistemáticos de la literatura (en inglés *Systematic Mapping Studies* o SMS) son estudios secundarios con un alcance más amplio que las SLRs, dado que su principal objetivo es proporcionar una visión global sobre un tema de interés (con enfoque empírico o no) e identificar la cantidad y tipo de investigación y resultados disponibles sobre el mismo. Esto permite identificar temas en los que la evidencia empírica sea escasa y sea necesario realizar más estudios empíricos.

Es muy común, en un SMS calcular la frecuencia de publicaciones a lo largo del tiempo para detectar tendencias, o clasificar los artículos encontrados según algún esquema de clasificación predefinido. Los SMSs suelen consumir menos tiempo que las SLRs y sirven a los investigadores como base para llevar a cabo futuras investigaciones, siempre y cuando se realicen con rigor (Kitchenham *et al.*, 2011).

Analizando varios artículos publicados por otros autores y por nosotros mismos llegamos a la conclusión de que no estaba muy clara la diferencia entre SLR y SMS, porque en muchos casos artículos que dicen han realizado una SLR en realidad lo que han llevado a cabo es un SMS. La Tabla 6.2 presenta las principales diferencias entre las SLRs y los SMSs.

Elementos	**SMS**	**SLR**
Objetivos	Clasificación y análisis temático de la literatura sobre un tema específico de la ingeniería de *software*	Identificar las mejores prácticas con respecto a procedimientos, tecnologías, métodos o herramientas específicas, mediante la agregación de información obtenida a partir de estudios empíricos.
Pregunta de investigación	Genérica, relacionada con tendencias de investigación, como por ejemplo: qué investigadores, cuántos estudios, qué tipo de estudios, etc.	Específica, relacionada con resultados de estudios empíricos, como por ejemplo: ¿Es mejor el método/tecnología A que la B?
Proceso de búsqueda	Definido por el área de estudio o de interés	Definido por la pregunta de investigación la cual identifica la tecnología específica que está siendo investigada

Alcance	Amplio – se incluyen todos los artículos sobre un área de interés, pero sólo se extrae de ellos datos para clasificarlos	Centrado – sólo se incluyen artículos que contengan estudios empíricos relacionados con las preguntas de investigación y se extrae de ellos información detallada sobre los resultados obtenidos en cada uno de ellos
Requisitos de la estrategia de búsqueda	A menudo menos estricta si sólo se buscan tendencias, por ejemplo se puede buscar sólo en un conjunto específico de publicaciones, limitándolas a artículos de revistas, o limitándolas a una o dos bibliotecas digitales	Extremadamente exigente– se deben encontrar todos los artículos relevantes. Generalmente además de buscar en las fuentes establecidas, puede ser necesario buscar en las referencias de los estudios primarios seleccionados o consultar a los expertos para incluir el mayor número de artículos posible
Evaluación de la calidad	No es esencial. Al incluir tanto estudios teóricos como empíricos de cualquier tipo, suele ser muy difícil definir un mecanismo de evaluación	Es importante asegurarse de que los resultados se basan en la evidencia de mejor calidad
Resultados	Un conjunto de artículos relacionados con un área de interés clasificados en una serie de dimensiones, especificando el número total de artículos en cada dimensión	Se agregan los resultados de los estudios empíricos para contestar a las preguntas de investigación

Tabla 6.2. Diferencias entre SMSs y SRLs

6.3.2 Revisiones terciarias

En un dominio en el que existan varias SLRs o SMSs puede ser posible realizar un estudio terciario (en inglés *Tertiary Study*), es decir una revisión sistemática de revisiones sistemáticas, con el objetivo de responder a preguntas de investigación más amplias. Una revisión terciaria se puede realizar siguiendo el mismo proceso que para revisiones sistemáticas. La realización de este tipo de revisiones probablemente requiere menos recursos, pero depende de que existan sobre el dominio de estudio suficientes revisiones sistemáticas de calidad. El gran número de SLRs y SMSs que se han realizado en la ingeniería del *software* ha permitido realizar varias revisiones terciarias, como se muestra en la Tabla 6.3.

Estudio terciario	Tema/objetivo	Estrategia de búsqueda	Período de tiempo	Cantidad de SLRs incluidas
Kitchenham *et al.* (2009)	Ingeniería del *software*	Sólo manual	Enero 2004–Junio 2007	20
Kitchenham *et al.* (2010a)	Actualización de la anterior	Sólo automática	Enero 2004–Junio 2008	33
da Silva *et al.* (2011)	Actualización de las dos anteriores	automática + manual	Enero 2008–Diciembre 2009	67
Cruzes y Dybå (2011a)	Métodos de síntesis	Automática e incluye los artículos seleccionados en (1) y (2)	Enero 2005–Julio 2010	49
Zhang y Ali Babar (2013)	Identificar las personas que realizan SLRs/SMSs en ingeniería del *software*	Manual y automática	Enero 2004–Diciembre 2010	144

Tabla 6.3. Resumen sobre estudios terciarios sobre SLRs en la ingeniería del software

También se han realizado numerosas SLRs sobre el desarrollo global de *software* (37 artículos) que se incluyeron en un estudio terciario sobre la mitigación de riesgos en el desarrollo global de *software* (Verner *et al.*, 2014).

6.4 EJEMPLO DE UN MAPEO SISTEMÁTICO DE LA LITERATURA

A continuación se describe un SMS publicado en Fernández-Sáez *et al.* (2011) y que se ha desarrollado siguiendo las directrices de Kitchenham y Charters (2007). Si bien los autores lo publicaron como una SLR, considerando las diferencias entre SMS y SLR presentadas en la Tabla 6.2 y comunicaciones personales que hemos mantenido con Barbara Kitchenham, llegamos a la conclusión de que la revisión de la literatura realizada es realmente un SMS.

6.4.1 Planificar la revisión

En este apartado se presentan cada una de las tareas realizadas durante la planificación.

6.4.1.1 IDENTIFICAR LA NECESIDAD DE LA REVISIÓN

En esta tarea se han evaluado revisiones de la literatura existentes en el contexto de la calidad de modelos conceptuales y el modelado de *software* utilizando UML (Moody, 2005; Genero et. al, 2005; Pretorius y Budgen, 2008; Mohagheghi *et al.*, 2009; Lucas *et al.*, 2009).

El objetivo de este SMS es identificar "¿Qué se ha hecho?" y "¿Qué se debe hacer en el futuro?" en el contexto de la calidad de los modelos UML. Esto contrasta con las revisiones de la literatura existentes que tienen un enfoque más limitado, como son: medidas para diagramas de clases UML, marcos de calidad, investigación empírica, MDA y consistencia. Las revisiones presentadas en Genero *et al.* (2005) y Moody (2005) no describen un proceso de selección sistemático, ni especifican criterios de inclusión/exclusión. Pretorius y Budgen (2008) presentan una revisión sistemática pero en una escala bastante pequeña con sólo 33 artículos seleccionados. En la revisión de Mohagheghi *et al.* (2009) se utilizaron menos fuentes de búsqueda, incluyendo 40 estudios primarios, y el proceso de selección no fue estrictamente sistemático.

Por lo tanto este SMS presentado como ejemplo, difiere de las revisiones de la literatura realizadas previamente en tres aspectos: el objetivo es diferente, la revisión es más amplia y más sistemática y la clasificación de estudios primarios es más refinada.

6.4.1.2 FORMULAR LAS PREGUNTAS DE INVESTIGACIÓN

Se formularon las preguntas de investigación que se presentan en la Tabla 6.4, junto a su motivación.

Preguntas de investigación	Motivación
PI1 ¿Qué tipos de calidad sobre los modelos UML se investigaron?	Descubrir qué tipos de calidad y qué características específicas de calidad de los modelos UML se investigaron.
PI2 ¿Qué tipo de métodos se utilizaron en la investigación relacionada con la calidad de los modelos UML?	Determinar si las propuestas en este campo de investigación son más prácticas o más de investigación básica y también identificar oportunidades para realizar futuras investigaciones.

PI3 ¿Cuál es la naturaleza de los resultados de investigación sobre la calidad de los modelos UML?	Encontrar el tipo de resultados obtenidos en el contexto de la calidad de los modelos UML y evaluar el estado de este campo.
PI4 ¿Cuáles son los objetivos perseguidos en la investigación sobre la calidad de los modelos UML?	Determinar dónde se encuentra la mayor parte del interés de investigación y qué áreas han sido poco investigadas, explorando conceptos básicos, recopilando conocimiento de prácticas actuales o con el objetivo de avanzar la práctica a través de la ciencia del diseño (*Design Science*).
PI5 ¿En qué tipos de diagramas UML se centra la investigación sobre la calidad de modelos UML?	Descubrir en qué tipos de diagramas se ha centrado la investigación, determinar qué diagramas se han considerado más importantes y también encontrar oportunidades para realizar futuras investigaciones.

Tabla 6.4. Preguntas de investigación

6.4.1.3 DEFINIR EL PROTOCOLO DE REVISIÓN

A continuación se describen cada uno de los elementos del protocolo definido para este SMS.

Antecedentes

El desarrollo de *software* cada vez más utiliza enfoques centrados en modelos (Mohagheghi, *et al.*, 2009), hecho fomentado por iniciativas como la "Arquitectura dirigida por modelos" (conocida por sus siglas en inglés *MDA*) (OMG, 2003) y el "Desarrollo dirigido por modelos" (conocido por sus siglas en inglés MDD) (Atkinson y Kühne, 2003) que hacen hincapié en el relevante papel de los modelos en el desarrollo de *software*.

La pregunta relevante ya no es "¿Debemos utilizar el modelado?" sino "¿Cómo debemos hacer el modelado?". Este nuevo enfoque en el proceso de modelado, en lugar de en el producto de *software* resultante de las actividades de desarrollo, pone a la calidad de los modelos en la vanguardia, lo que refleja el creciente interés, tanto en la industria y la academia, en los métodos y técnicas para la evaluación, aseguramiento y mejora de la calidad de los modelos en el desarrollo y mantenimiento de *software* (Mohagheghi *et al.*, 2009).

Si bien ha habido una gran cantidad de investigación sobre la calidad del *software*, son relativamente pocos los trabajos sobre la calidad de los modelos, y no hay consenso sobre el concepto de la calidad de modelos. Los conocimientos

existentes sobre la calidad del *software* no son del todo aplicables a los modelos, ya que los modelos tienen características muy diferentes al código fuente. Por ejemplo los modelos tienen múltiples vistas, se pueden utilizar de manera informal en lugar de formal y precisa, se pueden utilizar en todas las fases del proyecto, etc. Por otro lado, en la investigación en el modelado ha habido una tendencia a centrarse en la mejora del propio lenguaje UML, para tratar casos especiales de modelado, que en la mejora de la calidad de los propios modelos.

Con el fin de avanzar en el campo de la investigación sobre la calidad del modelado, es útil explorar la historia de este campo y determinar su estado actual, localizando, evaluando e interpretando la investigación relevante hasta la fecha relacionada con la calidad de los modelos centrada especialmente en los modelos UML.

Como se indicó en la tarea "Identificar la necesidad de la revisión" existen algunas revisiones de la literatura relacionadas con la calidad de los modelos conceptuales y UML, pero tienen un enfoque restrictivo y no se han realizado siguiendo un proceso sistemático. Todo lo dicho, motivó la realización del SMS presentado en este ejemplo.

Preguntas de investigación

Las preguntas de investigación que se intentarán responder tras la realización de este SMS se presentan en la Tabla 6.4.

Estrategia de búsqueda

Se decidió realizar la búsqueda automática en el período comprendido entre 1997 y 2009 y en las siguientes fuentes: SCOPUS, Science@Direct en el área Computer Science, Wiley InterScience en el área de Computer Science, IEEE Digital Library, ACM Digital Library, y Springer. Como el Object Management Group adoptó a UML en el año 1997 (OMG, 1997), no tendría sentido buscar artículos anteriores a ese año.

La selección realizada sobre los términos principales, sinónimos, palabras alternativas o términos relacionados con los términos principales se presenta en la Tabla 6.5. Queremos aclarar que presentamos los términos en inglés porque así se utilizaron para las búsquedas con el fin de buscar publicaciones en inglés, que es el idioma universal en el campo de la investigación.

Términos principales	Términos alternativos
Quality	*quality OR consistency OR maintainability OR understandability OR completeness OR comprehension OR comprehensibility OR testability OR defect OR effectiveness OR complexity OR readability OR metric OR measure OR efficiency OR validation OR verification OR layout*
UML	*UML OR Unified Modeling Language*
Representation	*Representation OR diagram OR model*

Tabla 6.5. Términos de la cadena de búsqueda

La cadena de búsqueda definida a partir de la Tabla 6.5 es la siguiente:

"(quality OR consistency OR maintainability OR understandability OR completeness OR comprehension OR comprehensibility OR testability OR defect OR effectiveness OR complexity OR readability OR metric OR measure OR efficiency OR validation OR verification OR layout) AND (UML OR Unified Modeling Language) AND (Representation OR diagram OR model)"

La cadena de búsqueda se aplicará en las fuentes indicadas, buscando en el título y en el resumen, en caso de que el buscador lo permita y si no se buscará en el texto completo.

Todas las decisiones tomadas durante la definición de la cadena de búsqueda se tomaron conjuntamente entre los autores del artículo, la mayoría de los cuales se pueden considerar expertos, ya que cuentan con una amplia trayectoria en el campo del modelado conceptual y su calidad.

Criterios de selección de estudios

Se incluirán artículos en inglés que se refieran a la calidad de los diagramas UML y publicados entre 1997 y 2009 en revistas indexadas y en conferencias, congresos o talleres de prestigio con revisión por pares.

Se excluirán tipos de artículos de debate o de opinión, o disponibles sólo en forma de resúmenes o presentaciones en PowerPoint, duplicados (siempre considerando el artículo más completo), cuyo principal contribución no se relacione con la calidad de los modelos UML, o en los que la calidad se mencione por encima de manera introductoria en los resúmenes y también se excluirán artículos relacionados con la calidad del propio lenguaje UML.

Procedimiento para la selección de estudios

Para seleccionar los estudios primarios se aplicarán los criterios de inclusión/exclusión leyendo los resúmenes de los artículos encontrados en la tarea "Identificar la investigación relevante" de la actividad "Realizar la revisión". Si tras leer el resumen quedan dudas sobre la inclusión/exclusión de algún artículo se leerá el artículo completo.

La selección de estudios la realizará la primera autora del trabajo y el segundo autor cogerá aleatoriamente el 30% de los artículos para corroborar si los criterios de inclusión/exclusión se aplicaron correctamente, consultando al resto de autores en caso de haber dudas o discrepancias.

Listas de comprobación y procedimiento para la evaluación de la calidad de los estudios

Como criterio para considerar artículos de cierta calidad, se consideró seleccionar estudios incluidos en revistas indexadas y en conferencias, congresos o talleres de prestigio con revisión por pares.

Estrategia para la extracción de los datos

El formulario de extracción de datos tiene dos partes, la primera con los metadatos de cada estudio primario: título, autores, tipo de publicación, nombre de la conferencia/revista, año, etc. Y la segunda parte que contiene las dimensiones y categorías del esquema definido para clasificar los estudios primarios seleccionados. Este esquema de clasificación se definió teniendo en cuenta cada una de las preguntas de investigación (ver Tabla 6.4) y la literatura relevante sobre la calidad en modelos conceptuales (Piattini *et al.*, 2005; Krogstie, 1998; Lindland *et al.*, 1994; Nelson *et al.* 2001) (ver Tabla 6.6).

Dimensiones	Categorías
Tipo de calidad	Calidad sintáctica: corrección. Calidad semántica: consistencia, compleción, corrección. Calidad pragmática: mantenibilidad, analizabilidad, comprensibilidad, capacidad de prueba, funcionalidad, ejecutabilidad, reusabilidad, complejidad, confiabilidad.
Tipo de evidencia/método de investigación	Empírico: experimento, estudio de caso, encuesta No empírico: especulación, ejemplo, revisión de la literatura
Tipo de resultado de la investigación	Modelo de calidad, notación, método (técnica, metodología, proceso, aproximación, estrategia), marco, herramienta, medida, semántica formal, conocimiento, patrón, vista, lista de comprobación, directriz, regla, convención de modelado
Objetivo de la investigación	Comprender, medir, evaluar, asegurar, mejorar
Tipo de diagrama	Diagrama de clases, diagrama de secuencia, diagrama de actividad, diagrama de casos de uso, diagrama de estados, diagrama de colaboración, diagrama de componentes, diagrama de objetos, diagrama de paquetes, diagrama de despliegue, ningún diagrama específico, nuevos diagramas incluidos en UML 2.0

Tabla 6.6. Resumen del esquema de clasificación

La extracción de datos la realizará la primera autora del artículo y en caso de que hubiera dudas se resolverán entre todos los autores.

Síntesis de los datos extraídos

En primer lugar, se realizará una síntesis cuantitativa considerando el número y/o porcentajes de artículos en cada dimensión/categoría, ilustrándolos a través de tablas y/o gráficos, para de esta manera dar respuesta a cada pregunta de investigación, haciendo una correspondencia uno a uno entre pregunta y dimensión. También se considerará el cruce de dimensiones cuando se considere oportuno.

Además se analizarán: 1) El número de publicaciones por año para detectar y justificar tendencias y 2) El número de publicaciones por tipo de publicación para detectar los foros en los que más se ha publicado y orientar a futuros investigadores sobre los foros más apropiados en los que se puede buscar información o publicar en temas relacionados con la calidad de los modelos UML.

Estrategia de divulgación

Se decidió publicar el reporte de este SMS en la revista *Journal of Database Management*, por ser una revista de prestigio en el campo del modelado conceptual y por estar indexada en el *Journal of Citation Report*.

Calendario del proyecto

Por el propio conocimiento de los autores sobre el tema investigado, se preveía encontrar muchos artículos por lo que fue imposible hacer una planificación temporal. Sólo se sabía la fecha de inicio que fue en Julio de 2007.

6.4.1.4 VALIDAR EL PROTOCOLO DE REVISIÓN

El protocolo fue revisado por los 5 autores del artículo publicado en Fernández-Sáez *et al.* (2011) para asegurar que se habían tenido en cuenta todos los aspectos relevantes para lograr los objetivos de este SMS, considerando además las recomendaciones que proporcionan las directrices de Kitchenham y Charters (2007).

6.4.2 Realizar la revisión

Este SMS se llevó a cabo en dos fases, en una primera fase se localizaron artículos encontrados entre 1997 y 2007, y como el proceso se prolongó demasiado, en el momento de escribir el artículo para el *Journal of Database Management* el período de búsqueda había quedado desfasado y se decidió entonces llevar a cabo la segunda iteración para incluir artículos hasta diciembre de 2009. La cronología del proceso para la realización de este SMS se muestra en la Tabla 6.7 y a continuación se detallan las tareas llevadas a cabo.

6.4.2.1 IDENTIFICAR Y SELECCIONAR LOS ESTUDIOS PRIMARIOS

Se encontraron 1500 artículos entre los años 1997 y 2007, aplicando la estrategia de búsqueda definida en el protocolo. Debido a las limitaciones que ofrecen ciertas fuentes de búsqueda, en el caso de que no permitieran usar cadenas de búsqueda complejas, se tuvieron que diseñar cadenas específicas para cada fuente y manipular los resultados de las búsquedas para obtener los mismos resultados que pudieran haber sido obtenidos utilizando la cadena de búsqueda original. La búsqueda se hizo en el título y el resumen del artículo, excepto en

aquellas fuentes de búsqueda que no lo permitían, en las cuales se tuvo que buscar en el texto completo.

Para cada fuente de búsqueda se guardaron: las cadenas de búsqueda, los metadatos de los artículos encontrados (título, autores, año de publicación, etc.) y los resúmenes de los mismos.

De los primeros 1500 artículos, después de leer sus resúmenes y excluir los que no tenían nada que ver con la calidad de los modelos UML, sólo quedaron 483 de los cuales se excluyeron 144 por estar duplicados. A continuación, se aplicaron los criterios de inclusión/exclusión a los 339 artículos restantes, leyendo el texto completo. Se excluyeron además artículos que se referían a la medición del tamaño funcional de lo diagramas UML, por no considerarlos relacionados con la calidad de los modelos. En total en esta tarea se seleccionaron 215 y adicionalmente se refinaron tanto el esquema de clasificación como el formulario de extracción de datos.

Como se comentó anteriormente, al escribir el artículo para enviar al *Journal of Database Management* había pasado mucho tiempo, y el periodo de búsqueda había quedado un poco desactualizado, por lo que en marzo de 2010 se decidió extender la búsqueda para incluir también artículos publicados en los años 2008 y 2009. Inicialmente se encontraron 979 artículos más, a los cuales se le aplicó el mismo proceso que a los artículos encontrados en la búsqueda inicial, quedando finalmente 103 artículos, que sumados a los anteriores hacen un total de 318 artículos.

Como se planificó en el protocolo, la identificación y selección de estudios primarios la realizó la primera autora del artículo, y el segundo autor escogió aleatoriamente el 30% de los artículos para corroborar su correcta selección. Las dudas que surgieron durante la selección de artículos se resolvieron entre todos los autores.

6.4.2.2 EXTRACCIÓN DE DATOS

Primero se comenzó a extraer los metadatos y a clasificar los 215 artículos seleccionados durante la primera fase, leyendo el texto completo. Para la clasificación se utilizó el esquema de clasificación presentado en la Tabla 6.5. Durante esta tarea se resolvieron algunas dudas y se decidió excluir 18 artículos más, quedando un total de 193 estudios primarios. En una segunda fase se realizaron las mismas tareas finalizando el proceso con 73 estudios primarios. La lista completa con los 266 artículos seleccionados en ambas fases está disponible en la siguiente página web: *http://alarcos.esi.uclm.es/SLR-QualityUMLModels*.

Mes y año	Planificar	Realizar	Reportar	Resultados
Primera fase				
Julio de 2007	Definición del protocolo			Protocolo del SMS
Septiembre de 2007		Identificación de la literatura relevante (hasta septiembre de 2007)		Tabla con los metadatos de los artículos encontrados en cada fuente de búsqueda, incluyendo los resúmenes. (1500 artículos) Cadenas de búsqueda aplicadas en cada fuente de búsqueda
		Selección de estudios leyendo títulos y resúmenes		Tabla con los metadatos de los artículos seleccionados en cada fuente de búsqueda, incluyendo los resúmenes. (483 artículos)
Marzo de 2008		Recuperación de los ficheros de cada estudio primario		Repositorio de artículos (483 artículos)
Abril de 2008		Eliminación de duplicados		Tabla con los metadatos de los artículos (399 artículos).
Julio de 2008	Refinamiento del protocolo	Prueba de la extracción de datos		Formulario de extracción con el esquema de clasificación refinado.
Agosto de 2008		Selección de estudios y extracción de datos leyendo el texto completo		Formulario de extracción de datos completado con la clasificación de 215 estudios primarios
Febrero de 2009		Resolución de dudas en la clasificación de los estudios primarios		Formulario de extracción de datos revisado, con la clasificación de los estudios primarios. (193 artículos).
Marzo de 2009		Síntesis de los datos		Tablas, gráficos e interpretación de los resultados de los 193 artículos.
Julio de 2009			Informe piloto	

Segunda fase				
Marzo de 2010		Actualización de la búsqueda de la literatura relevante (hasta diciembre de 2009)		Tabla con los metadatos de los artículos encontrados en cada fuente de búsqueda, incluyendo los resúmenes. (979 artículos).
Marzo de 2010		Selección de estudios primarios leyendo los títulos y resúmenes		Tabla con los metadatos de los artículos seleccionados (140 artículos).
		Recuperación de los ficheros de cada estudio primario		Repositorio de artículos (140 artículos).
		Eliminación de duplicados		Tabla con los metadatos de los artículos (103 artículos).
Febrero de 2010		Selección de estudios y extracción de datos leyendo el texto completo		Formulario de extracción de datos con la clasificación de los estudios primarios (103 artículos)
Marzo de 2010		Resolución de dudas en la clasifican de los estudios primarios		Formulario de extracción de datos revisado, con la clasificación de los estudios primarios (73 artículos)
Abril de 2010		Síntesis de datos		Tablas, gráficos e interpretación de los resultados de los 266 artículos (193 + 73 artículos).
Julio de 2010			Reporte final	

Tabla 6.7. Cronología de las actividades del SMS

6.4.2.3 SÍNTESIS DE LOS DATOS

A continuación se presenta la síntesis de la información contenida en el formulario de extracción de datos, con el objetivo de responder a las preguntas de investigación formuladas. Además de mostrar datos cuantitativos a través de tablas y gráficos, al final se presenta la interpretación de los resultados obtenidos y se presentan algunas sugerencias inferidas de la síntesis.

Tipo de calidad de los modelos UML (PI1)

Según refleja la Tabla 6.8 la mayor parte de los esfuerzos de investigación se ha concentrado en la calidad semántica (50,75%), seguido de la calidad pragmática (38,72%), con relativamente poco esfuerzo de investigación sobre la calidad sintáctica (5,64%). Hay unos pocos artículos que tratan más de un tipo de calidad, por ejemplo, seis artículos se ocupan de la calidad y uno sólo de calidad sintáctica y semántica, seis se ocupan de la calidad semántica y pragmática, y sólo uno de los tres tipos de calidad.

Tipo de calidad	Número	Porcentaje
Sintáctica	15	5,64%
Semántica	135	50,75%
Pragmática	103	38,72%
Sintáctica+ Semántica	6	2,26%
Sintáctica + Pragmática	0	0,00%
Semántica + Pragmática	6	2,26%
Sintáctica + Semántica + Pragmática	1	0,38%
Total	266	100,00%

Tabla 6.8. Porcentaje de artículos relacionados con cada tipo de calidad

Como se muestra en la Tabla 6.6 cada tipo de calidad tiene asociado varias características de calidad. El número y porcentaje de artículos para cada característica de calidad se muestra en la Tabla 6.9. Notar que como algunos artículos se refieren a más de una característica de calidad la suma total de artículos supera los 266, que es el número total de estudios primarios seleccionados.

Sintáctica	**Número**	
Corrección	21	100,00%
	Total	21
Semántica	**Número**	
Consistencia	113	62,09%
Compleción	14	7,69%
Corrección	55	30,22%
	Total	182
Pragmática	**Número**	
Mantenibilidad	24	19,35%
Analizabilidad	1	0,81%
Comprensibilidad	78	62,90%
Capacidad de prueba	2	2,61%
Funcionalidad	4	3,23%
Ejecutabilidad	2	1,61%
Reusabilidad	1	0,81%
Complejidad	11	8,87%
Confiabilidad	1	0,81%
	Total	124

Tabla 6.9. Resultados por características dentro de cada tipo de calidad

La consistencia semántica ha sido con diferencia la característica de calidad más investigada. Los artículos clasificados en ésta categoría investigan principalmente cuestiones de consistencia que pueden ocurrir cuando los modelos se construyen utilizando varios tipos de diagramas UML. Le siguen artículos sobre la corrección semántica, mientras que sólo 14 artículos se dedican a la compleción semántica.

Con respecto a la calidad pragmática, la comprensibilidad encabeza la lista, y aunque distante le sigue en segundo lugar la mantenibilidad. Además de la complejidad (11 artículos), las otras características de calidad pragmática se abordan en sólo muy pocos artículos.

Método de investigación (PI2)

Los resultados según el método de investigación utilizado se muestran en la en la Tabla 6.10. Hay que tener en cuenta que el total (278) es mayor que el número total de artículos seleccionados en el SMS (266), hecho que se debe a que hay artículos que se clasificaron en más de una categoría. Por ejemplo, en un mismo artículo se llevaron a cabo una encuesta y un estudio de casos. Los métodos de investigación pueden ser empíricos o no empíricos. Los métodos empíricos contienen un 29,86% de los artículos y los no empíricos un 70,14%. La categoría no empírica incluye trabajos que no muestran ningún tipo de evidencia empírica sobre la utilidad que intentan prestar las propuestas realizadas, la mayoría se limitan a mostrar ejemplos de su aplicación.

Método de investigación	Número	Porcentaje	Sintáctica		Semántica		Pragmática	
Empírico	**83**	**29,86%**	**2**	**9,09%**	**19**	**12,84%**	**62**	**57,41%**
Experimento	66	23,74%	2	9,09%	9	6,08%	55	50,93%
Estudio de caso	15	5,40%	0	0,00%	9	6,08%	6	5,56%
Encuesta	2	0,72%	0	0,00%	1	0,68%	1	0,93%
No empírico	**195**	**70,14%**	**20**	**90,91%**	**129**	**87,16%**	**46**	**42,59%**
Especulación	26	9,35%	2	9,09%	19	12,84%	5	4,63%
Ejemplo	169	60,79%	18	81,82%	110	74,32%	41	37,96%
Revisión de la literatura	0	0,00%	0	0,00%	0	0,00%	0	0,00%
Total	278		22		148		108	

Tabla 6.10. Resultados por tipo de evidencia

Más de la mitad de artículos seleccionados en este SMS (60,79%) utilizan ejemplos para ilustrar las propuestas presentadas y existen relativamente pocos artículos sobre investigación empírica, El experimento es el método empírico que se utiliza con mayor frecuencia, mientras que las encuestas y revisiones de la literatura están casi ausentes. Analizando un poco más a fondo los artículos que presentan experimentos, se llegó a la conclusión de que el 71,67% de ellos se centran en el impacto que tienen diferentes métodos o estilos en la comprensibilidad de los modelos (esta información no se muestra en la Tabla 6.10). La investigación experimental tiende a llevarse a cabo en mayor medida con alumnos universitarios de tercer, cuarto o quinto año de la carrera de informática, mientras que es menos frecuente la participación de profesores universitarios (5,56%) o de profesionales en la industria (22,22%). Además, el uso de alumnos y también el uso de problemas de "juguete" impiden en muchos casos generalizar los resultados encontrados.

Al combinar esta pregunta de investigación con la pregunta sobre el tipo de calidad (PI1), se observa que el 55 de estos artículos presentan experimentos controlados para corroborar hipótesis sobre la calidad pragmática. La mayoría de artículos, por amplia diferencia, presentan investigación no empírica relacionada con la calidad semántica, presentando modificaciones relacionadas con la calidad de los modelos o que extienden UML y demuestran el problema y/o su utilidad usando uno o más ejemplos. Del total de artículos sobre calidad semántica que presentan ejemplos, el 70,27% se centra en aspectos de la consistencia.

Resultado de la investigación (PI3)

La síntesis cuantitativa según el tipo de resultados que proponen, se muestra en la Tabla 6.11, en la que nuevamente se puede observar que el total es mayor que 266, debido a que hay artículos que se pueden clasificar en más de una categoría. Por mucha diferencia, la propuesta más frecuente es la de un nuevo método, relacionados con la validación, verificación o transformación de modelos, etc. Le siguen aunque distante, los artículos que producen nuevo conocimiento. Estos son en gran parte los artículos que emplean métodos de investigación empíricos (PI2), ya que se puede considerar que la verificación de hipótesis produce nuevo conocimiento.

Tipo de resultado	Número	Porcentaje
Semántica formal	3	1,01%
Marco	3	1,01%
Conocimiento	55	18,46%
Método	119	39,93%
Medida	28	9,40%
Notación	10	3,36%
Patrón	4	1,34%
Modelo de calidad	1	0,34%
Herramienta	50	16,78%
Vista	3	1,01%
Lista de comprobación, regla, directriz, convención de modelado	22	7,38%
Total	298	100,0%

Tabla 6.11. Resultados por tipo de resultado

Siguiendo de cerca al conocimiento, y en tercer lugar, se encuentran los artículos que proponen nuevas herramientas, como herramientas para la comprobación automática de la consistencia entre los diagramas dentro de un

modelo UML, la comprobación basada en modelos, herramientas de visualización, etc. Los artículos sobre medidas presentan una variedad de medidas para medir diferentes características de los modelos, como tamaño, complejidad, consistencia, etc. El quinto tipo más común de artículos es aquel en el que se presentan reglas, convenciones de modelado, directrices y listas de comprobación, y el resto de tipos de resultado son muy escasos.

Si nos centramos en las cinco categorías señaladas (ver Tabla 6.12), podemos ver que las propuestas de métodos, herramientas y reglas se refieren sobre todo a la calidad semántica, mientras que las propuestas que incluyen conocimiento o medidas se relacionan en mayor medida con la calidad pragmática. Para los estudios que producen conocimiento, este resultado es consistente con el hallazgo de que éstos emplean sobre todo experimentos que se centran en la calidad pragmática y específicamente en la comprensibilidad.

Tipo de calidad	Método	Conocimiento	Herramienta	Medida	Lista de comprobación: regla: directriz: convención de modelado
Pragmática	**18,25%**	**76,06%**	**22,03%**	**91,18%**	**24,0%**
Confiabilidad	0,73%	0,00%	0,00%	0,00%	0,0%
Ejecutabilidad	0,73%	0,00%	3,39%	0,00%	0,0%
Funcionalidad	1,46%	2,82%	0,00%	2,94%	0,0%
Mantenibilidad	3,65%	9,86%	3,39%	26,47%	0,0%
Reusabilidad	0,73%	0,00%	0,00%	0,00%	0,0%
Complejidad	0,00%	1,41%	1,69%	23,53%	4,0%
Capacidad de prueba	0,00%	0,00%	1,69%	2,94%	0,0%
Comprensibilidad	10,95%	60,56%	11,86%	35,29%	20,0%
Analizabilidad	0,00%	1,41%	0,00%	0,00%	0,0%

Semántica	**74,45%**	**19,72%**	**62,71%**	**8,82%**	**72,0%**
Compleción	4,38%	7,04%	3,39%	0,00%	8,0%
Consistencia	55,47%	9,86%	38,98%	5,88%	48,0%
Corrección	14,60%	2,82%	20,34%	2,94%	16,0%
Sintáctica	**7,30%**	**4,23%**	**15,25%**	**0,00%**	**4,0%**
Corrección	7,30%	4,23%	15,25%	0,00%	4,0%

Tabla 6.12. Cruce de tipo de resultado con tipo de calidad y característica de calidad

De los métodos que se ocupan de la calidad semántica (ver Tabla 6.12), podemos ver que la mayoría son propuestas para mejorar la consistencia de los diagramas UML. La misma observación es válida para las herramientas que se proponen para la calidad semántica, la mayoría de ellos se centran en la consistencia, aunque un porcentaje sustancial de las herramientas propuestas se refieren a la corrección semántica. Además, la mayoría de las reglas, convenciones de modelado, directrices y listas de comprobación están relacionadas con la calidad semántica, sobre todo con la consistencia.

La mayoría de los artículos que proponen medidas para la calidad pragmática, proponen medidas para evaluar o predecir la mantenibilidad de los diagramas UML, mientras que el siguiente mayor porcentaje se concentra en la medición de la comprensibilidad. Estas dos categorías están estrechamente relacionadas ya que antes de modificar un diagrama éste debe ser comprendido correctamente. Es de destacar que el 76% de los artículos en estas dos categorías incluye una validación de las medidas a través de uno o más experimentos controlados o un estudio de caso, además de la propia definición de las medidas (ver Tabla 6.12).

Objetivo de la investigación (PI4)

El propósito de investigar los objetivos de investigación perseguidos es determinar en qué se centra el interés de la investigación sobre la calidad de modelos UML y descubrir áreas que han sido poco investigadas. Como muestra la Tabla 6.13, hay 123 artículos (45,9%) relacionados con el aseguramiento de la calidad, 85 artículos (31,7%) con la evaluación de la calidad y 38 artículos (14.2%)

con la medición de la calidad. Las otras dos categorías juntas, mejorar y comprender, contienen menos del 9% de los artículos.

Objetivo de investigación	Número	Porcentaje
Mejorar	15	5,64%
Asegurar	122	45,49%
Medir	38	14,29%
Evaluar	85	31,95%
Comprender	7	2,63%
Total	266	100,0%

Tabla 6.13. Resultados por objetivo de investigación

La investigación sobre el aseguramiento de la calidad y la evaluación de la calidad de modelos UML suman más de las tres cuartas partes de los artículos publicados (77,44%). Esto no es sorprendente, porque el aseguramiento de la calidad es un tema muy importante. Los otros temas de investigación son importantes para avanzar en el estado del arte de la calidad de los modelos UML, pero se ven menos reflejados en los resultados de este SMS. Esto puede significar que un determinado tema no ha sido suficientemente estudiado o que aún tiene que encontrar la aceptación en revistas, o se pueden dar ambos casos.

Diagrama UML (PI5)

Mientras que más del 60% de los trabajos incluidos en este SMS se centró en la calidad de un determinado tipo de diagrama UML, casi el 40% examinó los diagramas UML en su conjunto (ver Tabla 6.14). La versión original de UML (1997) tenía nueve tipos de diagramas diferentes para modelar sistemas desde diferentes puntos de vistas. UML 2.0 introdujo cuatro diagramas nuevos, haciendo un total de 13 diagramas.

El diagrama de clases es el más estudiado, seguido por los diagramas de transición de estados y de secuencia. La investigación sobre la calidad de los modelos UML ha prestado menos atención a los diagramas de casos de uso y los diagramas de actividad. Muy pocos de los estudios primarios seleccionados se centran en diagramas de colaboración, de componentes y de paquetes. Este resultado se puede deber a que los diagramas de clases, de transición de estados y de secuencia han existido mucho antes de que apareciera UML. No se encontraron artículos relacionados con la calidad de los tipos de diagramas nuevos, incorporados en UML 2.0.

Tipo de diagrama	Número	Porcentaje
Diagrama de clases	83	25,30%
Diagrama de secuencia	34	10,37%
Diagrama de actividad	15	4,57%
Diagrama de casos de uso	21	6,40%
Diagrama de transición de estados	55	16,77%
Diagrama de colaboración	8	2,44%
Diagrama de componentes	3	0,91%
Diagrama de objetos	2	0,61%
Diagrama de paquetes	3	0,91%
Diagrama de despliegue	1	0,30%
Ningún diagrama específico	103	31,40%
Nuevos diagramas de UML 2.0	0	0,0%
	328	100,0%

Tabla 6.14. Resultados por tipo de diagrama

Resultados adicionales

Además del conocimiento del estado del arte sobre la calidad de los modelos UML es interesante conocer la evolución a lo largo del tiempo y también los foros que más han acogido publicaciones sobre el tema. Como se muestra en la Figura 6.2, hay una clara progresión en el número de publicaciones que aparecen cada año, cifra que puede demostrar que el interés por este tema ha ido creciendo con el tiempo, llegando a su punto más alto en 2007.

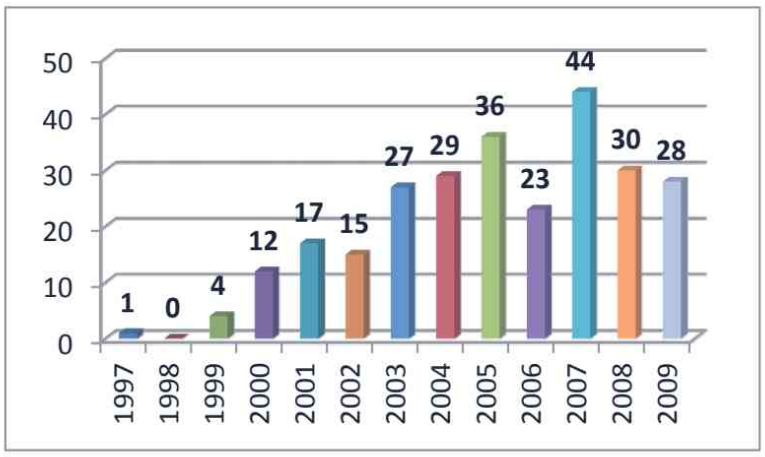


Figura 6.2. Evolución de artículos por año

Al analizar los foros en los que se publica, se encontró que el 58,6% de los artículos (157) se publicaron en conferencias, el 34,3% en revistas (92) y el 7,1% en talleres (19). El primer artículo en revista apreció en 1999, y su número ha ido creciendo hasta 2009 momento en el que se publicaron el mayor número de artículos, 16. Esta progresión en el número de artículos publicados, manifiesta sin lugar a dudas que la calidad de los modelos UML se ha considerado un "tema candente" de investigación.

La Tabla 6.15 muestra los foros con el mayor número de artículos publicados sobre la calidad de los modelos UML. Los tres primeros, "*International Conference on Model Driven Engineering Languages and Systems* (MODELS)", que originalmente se llamó UML, "*Electronic Notes in Theoretical Computer Science*" e "*Information and Software Technology*" tienen 16, 15 y 9 artículos cada uno, representando en total el 15,04% de los artículos publicados. El que le sigue es "*nternational Conference on Software Engineering* (ICSE)", que tiene 8 artículos, representando el 3,01% del total.

Foro de la publicación	**Número**	**Porcentaje**
International Conference on Model Driven Engineering Languages and Systems (*MODELS*: *originalmente UML*)	16	6,04%
Electronic Notes in Theoretical Computer Science	15	5,66%
Information and Software Technology	9	3,40%
International Conference on Software Engineering (*ICSE*)	8	3,02%
6.4.2.4 ER Workshops	7	2,64%
6.4.2.5 ACM SIGSOFT Software Engineering Notes	6	2,26%
6.4.2.6 Journal of Systems and Software	6	2,26%
6.4.2.7 ACM Symposium on Software Visualization (*SoftVis*)	5	1,89%
6.4.2.8 Empirical Software Engineering	5	1,89%
6.4.2.9 International Conference on Automated Software Engineering (*ASE*)	5	1,89%
6.4.2.10 Asia-Pacific Software Engineering Conference (*APSEC*)	4	1,51%
6.4.2.11 Australian Software Engineering Conference (*ASWEC*)	4	1,51%

Tabla 6.15. Número de artículos por foro en el que han sido publicados

Interpretación de los resultados

Una vez realizada la síntesis cuantitativa se ha realizado la interpretación de la misma con el fin de explicar los resultados.

Sobre la pregunta de investigación 1 "¿Qué tipos de calidad sobre los modelos UML se investigaron?" los resultados muestran un orden claro, lo que puede indicar la importancia relativa que los investigadores atribuyen a cada tipo de calidad. El orden es: (1) la calidad semántica (es decir, la corrección y compleción de los modelos con respecto al sistema a ser modelado), (2) la calidad pragmática (es decir, los aspectos relacionados con el uso de los modelos), y (3) calidad sintáctica (es decir, la corrección sintáctica de los modelos). Mientras que es difícil de explicar por qué la calidad semántica ha recibido más atención por parte de los investigadores que la calidad pragmática, excepto tal vez por el enfoque que han tenido los artículos sobre la consistencia de los modelos (113 de 266 documentos), que tratamos de interpretar a continuación. Además, menos del diez por ciento de los trabajos tratan la calidad sintáctica, pero este resultado no es sorprendente. Esto es porque la mayoría, si no todas, las herramientas de modelado obligan que se cumpla la corrección sintáctica de forma automática, de modo que si un elemento particular no es sintácticamente correcto, la herramienta no va a permitir que se introduzca.

La investigación sobre la calidad semántica se ha centrado principalmente en cuestiones relacionadas con la consistencia. Una razón plausible para la atención que se le ha prestado a aspectos de la consistencia, es que UML tiene 13 diagramas diferentes, y algunos solapan en cuanto a su semántica y finalidad. Además, se ofrece poca orientación sobre cuándo utilizar estos diferentes tipos de diagramas. Por ello parece lógico que los investigadores hayan estudiado formas de hacer frente a las posibles incoherencias que pueden surgir cuando se utilizan varios diagramas (de diferentes tipos) para modelar el mismo sistema, desde diferentes puntos (aunque a veces relacionados e incluso solapados) de vista.

La investigación sobre la calidad semántica ha prestado menos atención a las cuestiones de la corrección semántica (es decir, ¿Se modela correctamente el sistema?), y sobre todo a la compleción semántica (es decir, ¿Se han modelado todos los elementos del sistema relevantes?). Este resultado puede ser debido a la considerable dificultad de proponer nuevos enfoques para asegurar que un modelo es completo y correcto cuando se compara con un dominio, con descripciones de los requisitos del sistema o de la estructura y los flujos de procesos, etc. Aunque es importante garantizar la consistencia entre diagramas, se puede argumentar que sin la atención adecuada a la compleción y corrección de los modelos no se va a contribuir a la construcción de un "buen" sistema.

Con respecto a la calidad pragmática, la investigación ha puesto de relieve la comprensibilidad, y, en menor medida, la mantenibilidad de los modelos UML. Hay varias razones posibles para esto. En primer lugar, estas características de calidad pueden ser relativamente más fáciles de operacionalizar en la investigación que otras características como la funcionalidad, la complejidad y la reusabilidad, por ejemplo a través de preguntas de comprensión y tareas de modificación. En segundo lugar, la investigación sobre aspectos de calidad pragmática se ha centrado principalmente en el uso de modelos UML para facilitar la comunicación entre las partes interesadas. Los modelos pueden verse como instrumentos que llevan la información de una parte a otra (lo que significa que tienen que ser fácil de entender). Esta observación puede deberse a que la calidad pragmática originalmente se definió como "el grado en que se entiende un modelo" (Lindland *et al.*, 1994). Una vez que un modelo es comprendido, se pueden realizar modificaciones, por lo que el siguiente paso lógico es investigar la facilidad de mantenimiento de los modelos.

Respecto a la pregunta de investigación 2 "¿Qué tipo de métodos de investigación se han utilizado en la investigación relacionada con la calidad de los modelos UML?", el hallazgo de que sólo 29,86% de los artículos respaldan sus afirmaciones con un estudio empírico es más que llamativo. La mayoría de los otros artículos emplean uno o más ejemplos para ilustrar el problema investigado y la solución propuesta, pero estos ejemplos no se pueden ver como evaluaciones de las propuestas. Una razón plausible para la baja presencia de los estudios empíricos es que este SMS también examinó artículos de conferencias y talleres, que en general son menos exigentes en cuanto a la exhaustividad de la investigación en comparación con las revistas. Es común que en los artículos de conferencias y talleres (que suelen tener restricciones de longitud) se presente el problema investigado y se proponga una solución para abordarlo, pero que la evaluación empírica se mencione como trabajo futuro.

La mayoría de los artículos que presenten una evaluación empírica son estudios que emplean experimentos. El uso de otros métodos (encuestas, revisión de la literatura) es raro. Por lo que, la evaluación empírica de las propuestas sobre la calidad de los modelos UML es claramente una oportunidad de futura investigación. Un número cada vez mayor de este tipo de estudios también indican que el campo está madurando. La realización de "buenos" estudios de casos, también es otra oportunidad ya que han sido poco explotados en la investigación relacionada con la calidad de los modelos UML.

Respecto a la pregunta de investigación 3 "¿Cuál es la naturaleza de los resultados de investigación sobre la calidad de los modelos UML?", casi la mitad de los trabajos revisados proponen un método relacionado con algún aspecto de calidad de los modelos UML, sobre todo relacionados con la calidad semántica y

en particular con la consistencia. Otros resultados típicos de la investigación incluyen (en orden decreciente de frecuencia) son: herramientas, medidas, y lo que podría ser descrito como instrumentos que proporcionan orientación a los modeladores (reglas, convenciones de modelado, directrices y listas de comprobación).

Las herramientas también apuntan en gran medida a la calidad semántica, mientras que la mayoría de las medidas se encargan de medir características de la calidad pragmática como la comprensibilidad y mantenibilidad.

Una minoría de los artículos, menos de uno de cada cinco, tenía como objetivo aumentar o confirmar los conocimientos existentes sobre la calidad de los modelos UML. Estos artículos en su mayoría presentan experimentos que evalúan la eficacia y eficiencia de propuestas realizadas sobre la calidad pragmática, en particular la comprensibilidad, o también su objetivo es validar medidas y la búsqueda de relaciones entre variables de la calidad de los modelos UML.

Creemos que estos resultados son una indicación de que la investigación sobre la calidad de los modelos UML se lleva a cabo principalmente siguiendo la tradición de la ciencia del diseño, centrándose en el desarrollo de nuevos artefactos que avanzan el estado de la práctica y su objetivo es ofrecer soluciones a los problemas del mundo real experimentados por los modeladores (por ejemplo, el problema de cómo garantizar la consistencia entre diferentes diagramas UML que componen un modelo). La falta de estudios de evaluación podría ser una consecuencia de la decisión de incluir también conferencias y artículos de talleres en este SMS, como se señaló antes. Sin embargo, si esta evaluación no está presente (es decir, no se continúa con la publicación en revistas de la investigación completa incluyendo la evaluación empírica), indicaría un ciclo de investigación de la ciencia del diseño incompleto, lo que podría interpretarse como un indicio de inmadurez. La falta de estudios sobre "recopilación de conocimiento" puede entonces indicar un enfoque prematuro al proporcionar instrumentos para hacer frente a la calidad, sin un conocimiento profundo de la naturaleza de la calidad de los modelos UML y los factores que pueden influir.

La pregunta de investigación 4 "¿Cuáles son los objetivos perseguidos en la investigación sobre la calidad de los modelos UML?", se responde diciendo que el objetivo más importante es el aseguramiento de la calidad, que examina la manera de asegurar que el proceso de modelado produce un modelo de calidad. Esto es seguido de cerca por la evaluación de la calidad, que compara mediciones de calidad con experiencias del mundo real, lo que está más probablemente

relacionado con una tendencia a proponer nuevos métodos, herramientas y otros instrumentos de calidad (PI3) a través de métodos de investigación no empíricos (PI2). Sólo el tres por ciento de los artículos tenía como objetivo aumentar el conocimiento sobre la calidad del modelo. Así también se desprende de los resultados de PI3 donde los resultados de la investigación también tenían relativamente pocos trabajos sobre el conocimiento. El objetivo de la comprensión parece ser de poca importancia directa, aunque la medición y evaluación son importantes y también pueden ayudar a adquirir una mejor comprensión.

Respecto a la pregunta de investigación 5 "¿En qué tipos de diagramas UML se centra la investigación sobre la calidad de modelos UML?", el cuarenta por ciento de los artículos revisados no se enfocaban en algún diagrama UML en particular, por lo que la investigación presentada en estos artículos es a nivel general. Los artículos relacionados con algún diagrama en particular investigan, en orden de frecuencia, diagramas estructurales (casi exclusivamente diagramas de clases), diagramas de comportamiento (principalmente diagramas de transición de estados) y diagramas de interacción (sobre todo los diagramas de secuencia). Los tipos de diagramas que fueron recientemente introducidos en UML 2.0 no se han investigado aún desde una perspectiva de calidad, y los diagramas de interacción (por ejemplo, diagramas de colaboración) han recibido muy poca atención en la investigación.

Estos resultados reflejan en gran medida la frecuencia con que se utilizan los distintos diagramas en la práctica. Las investigaciones indican que los diagramas que más se usan en el modelado de *software* son, en orden decreciente de frecuencia, diagramas de casos de uso, diagramas de clases, diagramas de secuencia y diagramas de transición de estados (Dobing y Parsons, 2006; Erickson y Siau, 2007; Grossman *et al.*, 2005). Erickson y Siau (2007) concluyen que los diagramas UML más importantes son: diagramas de clases, de casos de uso, de secuencia y de transición de estados y se deben incluir en el "núcleo de UML". Cabe destacar que el diagrama que se utiliza más en la práctica, el diagrama de casos de uso, ha recibido poca atención por parte de la investigación de calidad de los modelos UML. Una posible razón para esto es que la investigación de la calidad de los modelos en general se ha centrado principalmente en diagramas estructurales o modelos de datos, y se ha prestado mucha menos atención a los modelos que representan el comportamiento del sistema y la interacción (Recker *et al.*, 2007),

Concluyendo se puede decir que los resultados obtenidos en este SMS están alineados con los obtenidos en la revisión de la literatura presentada en (Moody, 2005). Esta revisión de cuarenta artículos identificó doce grandes temas

teóricos y prácticos de la investigación existente: proliferación de propuestas, los diferentes niveles de generalidad, y la falta de pruebas empíricas, de adopción en la práctica, de acuerdo en los conceptos y la terminología, de coherencia con campos relacionados y estándares, de medidas, de procedimientos de evaluación, de directrices de mejora, de conocimiento acerca de las prácticas, y un enfoque en los modelos estáticos y en la calidad del producto. Los resultados indican que las cuestiones identificadas por Moody para la investigación de calidad de modelado conceptual, en general, tienen una aplicación a la investigación de la calidad de los modelos UML en particular.

Sugerencias de trabajo futuro

Los autores de este SMS, en base a su interpretación (en cierta manera subjetiva), proponen las siguientes sugerencias y reflexiones sobre el trabajo futuro:

- Se debe dedicar más esfuerzo a la investigación empírica sobre la calidad de los modelos en general y en particular de los modelos UML. Hay una proliferación de herramientas y extensiones de UML, pero pocos indicios de que estas herramientas y extensiones realmente mejoren la calidad de los modelos UML. Se necesita un esfuerzo coordinado de la investigación empírica, el uso de meta-análisis para la integración de los experimentos y la experimentación utilizando diagramas de proyectos del mundo real, con el fin de construir un cuerpo sólido de conocimiento. También para fomentar la réplica de los estudios empíricos es fundamental que el material experimental esté disponible.
- Se necesita fomentar la colaboración entre academia e industria. Lamentablemente la investigación llevada a cabo en la academia parece tener poca conexión con los problemas del mundo real. Los problemas, diagramas y proyectos del mundo real deben nutrir a la investigación, y la investigación básica tiene que ser capaz de ser aplicada fácilmente.
- La investigación sobre la calidad de los modelos UML parece concentrarse en tres tipos de calidad (sintáctica, semántica y pragmática), sin embargo, no hay consenso sobre las características de calidad abordadas ni sobre su definición.
- El tema de la calidad de los modelos UML tiene que madurar, con muchos más artículos revisados por pares publicados en las principales revistas.

6.4.3 Reportar la revisión

Como estaba previsto, en esta última actividad se redactó el artículo que fue enviado y finalmente aceptado para su publicación en la revista *Journal of Database Management* (Fernández-Sáez *et al.*, 2011). En esta publicación además de reflejar cada una de las actividades descritas, se incluyeron otros apartados que presentaremos a continuación. Como información complementaria se colgó en una página web (*http://alarcos.esi.uclm.es/SLR-QualityUMLModels*) la lista con los 266 estudios primarios seleccionados.

6.4.3.1 AMENAZAS A LA VALIDEZ

Las principales amenazas a la validez de un SMS son: el sesgo en la selección de artículos, la inexactitud en la extracción de datos y la clasificación errónea (Sjøberg *et al.*, 2005). Somos conscientes de que es imposible lograr una cobertura completa de todo lo escrito sobre un tema. Se utilizaron seis fuentes digitales, incluyendo revistas, conferencias y talleres relevantes en el campo de la ingeniería de *software*. El alcance de las revistas y conferencias que se tratan en este SMS es suficientemente amplio para alcanzar una exhaustividad razonable en el campo estudiado. No se incluyeron documentos adicionales, tales como literatura gris (informes técnicos, libros, etc.), ya que tienden a ser fuentes secundarias. La mayoría de la literatura gris o bien tiene su origen en artículos revisados por pares o se convertirá en artículos revisados en un futuro. Puede ocurrir que se hayan quedado artículos relevantes sin incluir, pero según el conocimiento de los autores en el tema, creemos que no hay muchos de estos casos.

Para ayudar a garantizar un proceso de selección imparcial, las preguntas de investigación se han definido de antemano, se organizó la selección de artículos como un proceso de múltiples etapas y participaron cinco investigadores en este proceso. Como se discutió anteriormente, las decisiones para seleccionar los estudios primarios de este SMS, las tomaron varios investigadores expertos en el tema bajo estudio y en el proceso se siguieron reglas rigurosas. Otro reto es que no existe un estándar de características de calidad, o métodos de la ingeniería del *software* empírica que se puedan utilizar para extraer las características de calidad y los métodos de investigación de manera consistente.

La duplicación de artículos, es una amenaza potencial a la hora de calcular la frecuencia de artículos y los estadísticos, aunque como en este SMS al menos dos personas revisaron la selección de artículos, no creemos que hayan quedado duplicados sin detectar.

La extracción de datos la hizo una sola persona (el primer autor del artículo) y la verificación el segundo autor. Los desacuerdos se resolvieron mediante discusión, con la participación del resto de autores, cuando se creyó oportuno. La extracción de datos y la clasificación es difícil en principio, debido a múltiples motivos como, la falta de terminología estándar y normas para la presentación de los estudios empíricos y para definir las características de calidad en ingeniería de *software*, lo que puede haber llevado a cometer algunos errores en la extracción de datos y esto puede haber dado lugar a una clasificación errónea. Sin embargo, creemos que el proceso de extracción y selección fue riguroso y que siguieron las directrices establecidas en Kitchenham y Charters (2007). También creemos que el hecho de que varios expertos realicen la clasificación, minimiza el riesgo de obtener una clasificación errónea. El esquema de clasificación que se utilizó en este SMS se puede utilizar como punto de partida por futuros investigadores.

6.4.3.2 LECCIONES APRENDIDAS

Por lo general, no es posible juzgar la relevancia de un estudio revisando únicamente el resumen, debido a que la calidad de los resúmenes en los artículos publicados en el ámbito de la ingeniería del *software* suele ser muy pobre. Por ello es conveniente seleccionar los artículos en base a la lectura del texto completo. El uso adecuado de resúmenes estructurados contribuirá a redactar resúmenes de mayor calidad (Budgen *et al.*, 2011). Los resúmenes estructurados deben contener los siguientes apartados: 1) Contexto (la justificación de la relevancia del tema abordado), 2) Objetivos (los objetivos principales perseguidos), 3) Método (el método de investigación seguido y la propuesta realizada para cumplir los objetivos, 4) Resultados (los principales resultados obtenidos) y 5) Conclusiones (las conclusiones que se pueden inferir a partir de los resultados obtenidos).

Debido a la limitación de los motores de búsqueda, se observó que una cadena tan larga como la propuesta, no se podía buscar directamente. Por lo tanto, era necesario adaptar el texto a buscar a cada fuente de búsqueda mediante el fraccionamiento de la cadena de búsqueda original y la combinación de los resultados de forma manual. Los motores de búsqueda que se usan actualmente en la ingeniería del *software* no están diseñados para soportar revisiones sistemáticas, contrario a lo que ocurre en medicina.

Estas lecciones aprendidas coinciden en gran medida con las lecciones aprendidas publicadas en Brereton *et al.* (2007), por lo que creemos que al ser problemas comunes deberían ser solucionados por la comunidad de investigadores de la ingeniería del *software*.

6.5 OTROS EJEMPLOS

Dentro de las 144 revisiones de la literatura (incluyendo tanto SLRs y SMSs), publicadas entre 2004 y 2010 las más citadas se muestran en la Tabla 6.16 (Zhang y Ali Babar, 2013). Las citas se buscaron en *Google Scholar* en Febrero de 2012.

Referencia	Tipo de revisión	Tema	Publicación	Año	Número de citas
Dybå y Dingsøyr (2008b)	SLR	Desarrollo ágil de *software*	IST	2008	339
Jørgensen y Shepperd (2007)	SMS	Estimación de costes de *software*	IEEE TSE	2007	277
Sjøberg *et al.* (2005)	SMS	Experimentos controlados en la ingeniería del *software*	IEEE TSE	2005	248
Jørgensen (2004)	SMS	Estimación del esfuerzo de desarrollo realizada por expertos	JSS	2004	218
Kagdi *et al.* (2007)	SMS	Explotación de un repositorio sobre evolución del *software*	JSME	2007	125
Kitchenham *et al.* (2007)	SLR	Comparación de estimación de costes utilizando datos de una misma compañía o cruzando datos de varias compañías	IEEE TSE	2007	115

Tabla 6.16. SLRs y SMSs más citados en ingeniería de software

6.6 LECTURAS RECOMENDADAS

- **Kitchenham**, **B**. y **Charters**, **S**. (2007). *Guidelines for performing systematic literature reviews in software engineering*, *Keele University*, EBSE-2007-01. (*http://cdn.elsevier.com/promis_misc/525444systematicreviewsguide.pdf*). Este reporte técnico, del cual existe una primera versión del año 2004, es el documento de referencia utilizado como guía para realizar SLRs. Su principal objetivo las directrices para realizar de manera metodológica y rigurosa revisiones de la evidencia empírica actual en el ámbito de la ingeniería de *software*. Está dirigido principalmente a investigadores de ingeniería de *software*, incluyendo estudiantes de doctorado.

- **Petticrew**, **M**. y **Roberts**, **H**. (2006). *Systematic reviews in the social science*: *A practical guide*. *Blackwell Publishing*. Este libro presenta consejos prácticos, ejemplos y bibliografía relevante sobre cómo realizar SLRS en el ámbito de las ciencias sociales. Para elaborar las directrices de Kitchenham y Charters (2007) se tuvo en cuenta este libro, entre otras fuentes.

- **Dybå**, **T**. y **Dingsøyr**, **T**. (2008b). *Empirical studies of agile software development*: *A systematic review*. *Information and Software Technology*, 50 (9-10), 833-859. Este es un muy buen ejemplo de una SLR .y la SLR más referenciada (Zhang y Ali Babar, 2013). Además presenta una minuciosa evaluación de la calidad de cada uno de los estudios primarios y analiza además la fortaleza de la evidencia recopilada con respecto al desarrollo ágil del *software*.

6.7 SITIOS WEB RECOMENDADOS

- *http://www.journals.elsevier.com/information-and-software-technology/*

  Este es el sitio web de la revista *Information and Sofware Technology*, revista pionera en la promoción de la publicación de SLRs en la ingeniería del *software*. Hasta la fecha es la revista que cuenta con el mayor número de SLRs y SMSs publicados.

6.8 HERRAMIENTAS RECOMENDADAS

Según nuestro conocimiento existen tres herramientas de libre acceso, que dan soporte a todas las actividades del proceso propuesto en Kitchenham y Charters (2007), para realizar tanto mapeos como revisiones sistemáticas de la literatura:

- El Grupo Alarcos ha desarrollado la herramienta *SLR-Tool* que está disponible en *http://alarcosj.esi.uclm.es/SLRTool*. Detalles de su diseño y funcionalidad se pueden encontrar en Fernández-Sáez *et al.* (2010).

- En la Universidad de Middlesex de Londres se ha propuesto otra herramienta disponible en *http://slrtool.org/v0/*

- Bowes *et al.* (2012) han propuesto la herramienta *SluRp* disponible en *https://bugcatcher.stca.herts.ac.uk/SLuRp*

Un estudio más exhaustivo sobre las herramientas que dan soporte total o parcialmente al proceso para realizar SLRs se puede encontrar en Marshall y Brereton (2013) y en Al-Zubidy y Carver (2014).

Capítulo 7

COMBINACIÓN DE MÉTODOS

En este capítulo se presentan dos métodos que se han aplicado en diferentes proyectos y tesis de investigación, que combinan algunas técnicas experimentales que se han presentado en los capítulos anteriores.

7.1 MÉTODO PARA LA INVESTIGACIÓN DE MEDIDAS DE SOFTWARE

Existe un gran número de medidas para capturar atributos de los procesos y productos *software*, que tradicionalmente se han realizado confiando en la sabiduría de los expertos. Esta situación ha conducido frecuentemente a cierto grado de imprecisión en las definiciones, propiedades y suposiciones de las medidas, haciendo que el uso de las medidas sea difícil, la interpretación peligrosa y los resultados de muchos estudios de validación contradictorios.

La mayoría de las medidas propuestas han fracasado, debido generalmente a que carecían de atributos necesarios para ser válidas y útiles para el propósito para el que fueron creadas. La creación de una medida debe seguir un proceso metodológico que permita obtener medidas adecuadas a nuestros propósitos.

Desde los años setenta han aparecido un gran número de medidas para capturar atributos del *software* de una forma cuantitativa. Sin embargo, muy pocas han sobrevivido con éxito la fase de definición y han resultado útiles. Esto se debe a múltiples problemas relativos a la validez teórica y empírica de muchas medidas, algunos los cuales se detallan a continuación (Briand *et al.*, 1999b):

- Las medidas no se definen siempre en un contexto en el que sea explícito y esté bien definido su objetivo de interés. Por ejemplo: la reducción del esfuerzo de desarrollo o la reducción de los fallos presentes en los productos *software*.

- Incluso si el objetivo es explícito, las hipótesis experimentales, a menudo no se hacen explícitas, por ejemplo. ¿Qué se pretende deducir del análisis? ¿Es creíble el resultado?

- Las definiciones de las medidas no siempre tienen en cuenta el entorno o contexto en el que serán aplicadas, por ejemplo ¿Se puede utilizar una medida definida para un entorno no orientado a objetos en un contexto orientado a objetos?

- No siempre es posible realizar una validación teórica adecuada de la medida porque el atributo que queremos medir no siempre está bien definido, por ejemplo, la noción de complejidad.

- Un gran número de medidas nunca se han validado empíricamente, por ejemplo, ¿qué medida de tamaño predice mejor el esfuerzo en un entorno de desarrollo?

Por ello en el grupo Alarcos hemos ido definiendo y refinando un método para la definición de medidas válidas (Calero *et al.*, 2001; Serrano *et al.*, 2002; Reynoso *et al.*, 2010) que se ha utilizado en varias tesis doctorales (Calero, 2001; Martínez, 2001; Genero, 2002; García, 2004; Serrano, 2004; Cruz, 2007; Reynoso, 2007; Mora, 2011) y sus respectivas publicaciones; así como en la creación de una *spinoff* del grupo, Alarcos Quality Center, S.L. que ha conseguido crear el primer laboratorio acreditado por ENAC para la evaluación de la calidad del *software*[5].

7.1.1 Método de trabajo

La definición de las medidas debe basarse en objetivos de medida claros y siguiendo las necesidades de la organización. Teniendo en cuenta todas las consideraciones expuestas, en la Figura 7.1 se muestra el proceso de definición de

[5] *http://www.alarcosqualitycenter.com/*

medidas, de manera que se puedan conseguir medidas válidas y útiles para productos *software*. En la Figura 7.1 se puede observar que el método consta de diversas fases que van desde la identificación de los objetivos y las hipótesis de trabajo hasta la aplicación y posterior retirada de una medida. En dicha figura, las flechas continuas representan el flujo de las medidas y las discontinuas representan el flujo de información a lo largo de todo el proceso. Este proceso consta de cinco etapas principales:

- **Identificación**: Se definen los objetivos que se persiguen a la hora de crear la medida y se plantean las hipótesis de cómo se llevará a cabo la medición. Sobre los elementos de esta etapa (objetivos e hipótesis) se basarán todas las etapas siguientes. Como resultado de esta etapa se generan los requisitos que debe cumplir la medida.

- **Creación**: Se realiza la definición de la medida y su validación teórica y empírica. Esta etapa es una de las más importantes y larga pues como abarca un proceso iterativo del que debe salir una medida válida tanto formal como empíricamente.

- **Aceptación**: Una vez obtenida una medida válida, suele ser necesario pasar por una etapa de aceptación de la medida en la que se harán pruebas en entornos reales, de manera que podamos comprobar si la medida cumple los objetivos deseados dentro del campo de aplicación real.

- **Aplicación**: Una vez que tengamos una medida aceptada, la utilizaremos dentro del campo de la aplicación para la que fue diseñada.

- **Acreditación**: Es la última etapa del proceso, que discurre en paralelo con la fase de aplicación y tiene como objetivo el mantenimiento de la medida, de manera que se pueda adaptarla al entorno cambiante de aplicación. Como consecuencia de esta etapa, puede que una medida sea retirada, porque ya no sea útil en el entorno en el que se aplica o que se reutilizada para iniciar el proceso de nuevo.

A continuación, se detallan las etapas que componen el método propuesto.

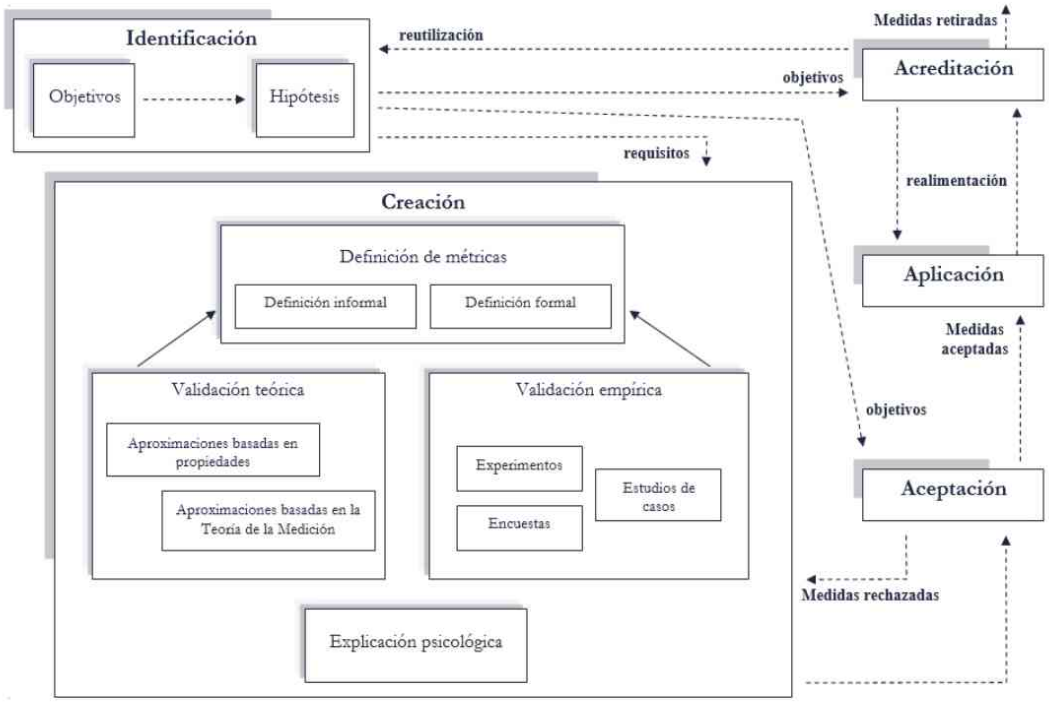


Figura 7.1. Método de investigación para la definición de medidas

7.1.2 Identificación

En esta etapa se pretende identificar los objetivos de la medida y las hipótesis en las que nos basamos para crear las medidas. Los objetivos indican lo que se pretende conseguir con la utilización del proceso de medida y representan la razón por la que se llevará el proceso de medida (el "porqué"). Las hipótesis son la forma en la que se pretende llevar a cabo la medida (el "cómo"), identificando la información que se debe manejar para conseguir alcanzar los objetivos deseados. Este proceso suele estar basado en la experiencia y el conocimiento de los expertos y puede utilizar mecanismos basados en GQM (*Goal-Question-Metric*).

Más concretamente, se llevan a cabo las siguientes actividades:

- Seleccionar la entidad de estudio, es decir, el producto, proceso, proyecto o recurso que se pretende caracterizar midiendo sus atributos.
- Determinar el foco de la calidad, es decir, los atributos en los cuales centraremos las medidas, por ejemplo, mantenibilidad, portabilidad, etc.

- Determinar los objetivos GQM a nivel conceptual; Analizar el "objeto del estudio" con el "propósito" respecto al "foco de calidad" desde el "punto de vista".
- Determinar las propiedades estructurales a estudiar, por ejemplo, acoplamiento, cohesión, tamaño, etc.
- Identificar las abstracciones para medir las propiedades estructurales, por ejemplo, si se trata del acoplamiento identificar los diferentes tipos de conexiones que constituyen acoplamiento, el lugar del impacto del acoplamiento, su granularidad, etc.
- Refinar los objetivos en preguntas a nivel operacional.

Como resultado de esta fase se deben obtener los requisitos que debe cumplir la medida, que serán utilizados en la etapa de creación. Además, como se observa en la Figura 7.1, los objetivos serán utilizados en las etapas de aceptación, aplicación y acreditación.

7.1.3 Creación

El proceso de creación es aquel en el que a partir de los requisitos obtenidos en la etapa de identificación se creará una medida válida, lista para ser aplicada en entornos reales. Como se puede observar en la Figura 7.1, el proceso de creación de las medidas es evolutivo e iterativo y se subdivide en varias etapas intermedias. Como resultado de la retroalimentación, las medidas deben ser redefinidas de acuerdo a las validaciones, teóricas o empíricas, fallidas.

Al final de la etapa de creación, las medidas se considerarán válidas y aquellas que no sean válidas, serán descartadas.

7.1.3.1 DEFINICIÓN

Es el primer paso de esta fase que debe realizarse considerando las características del producto que vamos a medir y la experiencia de los profesionales. Es recomendable conseguir esta definición de una forma metodológica, aplicando, por ejemplo, la aproximación GQM en el que se puede definir el propósito de la medida que se define, el punto de vista del usuario y el contexto de uso. También puede ser aconsejable utilizar algún metamodelo para definir las medidas.

En la definición se deben considerar objetivos claros, es decir, realizar una definición de la medida orientada al objetivo para evitar obtener una definición de la medida que no cumple con el objetivo deseado.

Por otro lado, aunque en una primera etapa definamos la medida en lenguaje natural, Es deseable que la definición de las medidas se realice de manera formal para evitar ambigüedades.

7.1.3.2 VALIDACIÓN TEÓRICA

El objetivo principal de la validación teórica es comprobar si la idea intuitiva acerca del atributo que está siendo medido se refleja en la medida. Esto se hace analizando los requisitos que deben ser satisfechos cuando estamos midiendo. Además la validación teórica proporciona información relacionada con las operaciones matemáticas y estadísticas que pueden ser realizadas con la medida, lo cual es esencial cuando tengamos que trabajar con ella.

Lamentablemente no existe un estándar para la validación formal a través del cual obtener la información matemática de las medidas definidas, sin embargo, hay dos tendencias principales en la validación: los marcos basados en aproximaciones axiomáticas (que definen formalmente propiedades deseables de las medidas para un atributo *software* concreto) (Briand *et al.*, 1996; Weyuker, 1988) y los que se basan en la teoría de la medida (Whitmire, 1997; Zuse, 1998; Poels y Dedene, 2000a) cuyo objetivo es obtener la escala matemática a la que pertenece una medida, y por tanto sus transformaciones admisibles, estadísticos y test aplicables y especifican un marco general en el que las medidas deben ser definidas.

7.1.3.3 VALIDACIÓN EMPÍRICA

El objetivo de esta etapa es probar la utilidad práctica de las medidas propuestas utilizando encuestas, experimentos y estudios de casos.

7.1.3.4 EXPLICACIÓN PSICOLÓGICA

Idealmente deberíamos ser capaces de explicar la influencia de los valores de las medidas desde un punto de vista psicológico. Por ejemplo, utilizando el concepto de complejidad cognitiva, entendida como el esfuerzo mental de una persona que está tratando con un artefacto *software*. Esto lleva consigo dos

ventajas principales: es útil para definir la razón de cada definición de medida, ya que muchas medidas se relacionan con limitaciones cognitivas (Klemola, 2000) y además nos proporciona un conocimiento muy importante a la hora de explicar los hallazgos experimentales.

7.1.4 Aceptación

Suele ser necesaria la existencia de una fase de pruebas en laboratorio en la que se realice una experimentación sistemática en entornos reales y con usuarios reales para verificar si cumple los objetivos buscados dentro de un entorno de trabajo real (esta etapa se diferencia de los estudios de casos en que en éstos últimos no se suele trabajar en el entorno final de aplicación). En definitiva intenta encontrar si las medidas "válidas" que se consiguieron al final de la fase de creación son aceptables en entornos de aplicación reales, teniendo en cuenta los objetivos obtenidos en la etapa de identificación.

Esta etapa debe ser realizada con proyectos no críticos y con riesgos controlados. Idealmente debería usarse en proyectos piloto de manera que el fracaso de aceptación de la medida no suponga un fracaso en un proyecto importante.

Si conseguimos demostrar que la medida sigue cumpliendo los objetivos, estaremos en disposición de pasar a la etapa de aplicación, si no es así, deberemos volver a la etapa de creación.

7.1.5 Aplicación

En esta etapa utilizaremos la medida ya aceptada en el entorno real. Esta fase discurrirá en paralelo con la fase de acreditación.

7.1.6 Acreditación

Esta última fase del proceso es una etapa dinámica que persigue el aseguramiento de la medida y la mejora continua de la misma, en función de cómo evoluciona el entorno de aplicación, de manera que podamos seguir cumpliendo los objetivos que se perseguían al principio del método.

En ocasiones el entorno puede variar tanto (por ejemplo, pasar de un entorno estructurado a uno orientado a objetos) que la medida no sea aplicable, en

este caso, la medida debería ser descartada y el conocimiento adquirido durante su tiempo de vida debería realimentarse a la etapa de identificación de manera que podamos crear una medida adecuada para el nuevo entorno cumpliendo los objetivos perseguidos. Además al utilizar la experiencia de la utilización de la medida descartada, tendremos más probabilidades de formular hipótesis correctas en la etapa de identificación.

7.2 EJEMPLO DEL MÉTODO: MEDIDAS PARA DIAGRAMAS DE CLASES UML

A continuación se presentará cómo se ha aplicado el método descrito en la tesis doctoral de Genero (2002) para la definición de medidas para diagramas de clases UML.

7.2.1 Identificación

Dado que el mantenimiento es una de las etapas del desarrollo de *software* que consume más recursos, el principal objetivo es medir la mantenibilidad de los diagramas de clases UML, debido a que como se menciona en la literatura la mantenibilidad de los diagramas puede afectar la mantenibilidad de producto de *software* final. Utilizando la plantilla GQM para la definición de objetivos (Basili y Rombach, 1988), el objetivo perseguido para la definición de las medidas para diagramas de clases UML es:

Analizar	Diagramas de clases UML
con el propósito de	Evaluar
con respecto a su	Mantenibilidad
desde el punto de vista de	Diseñadores de *software*
en el contexto de	Empresas de desarrollo de *software*

Aunque el objetivo es medir la mantenibilidad de los diagramas de clases UML, la mantenibilidad es un atributo externo de la calidad que no se puede medir directamente en un diseño de alto nivel. Por ello el objetivo es definir medidas para la complejidad estructural (atributo interno) de los diagramas de clases UML y luego a través de experimentación validar si estas medidas están relacionadas con la mantenibilidad.

Concretamente este trabajo se basa en el modelo presentado en la Figura 7.2 (Briand *et al.*, 1999a; ISO, 2001), que constituye la base teórica para el desarrollo de modelos cuantitativos para relacionar atributos internos y externos y se ha utilizado en numerosos estudios empíricos en el área de las propiedades

estructurales de los artefactos *software* (El-Emam *et al.*, 2001; Poels y Dedene, 2000b). La hipótesis es que las propiedades estructurales (como complejidad estructural y el tamaño) de un diagrama de clases UML, tienen un efecto es su complejidad cognitiva. La complejidad cognitiva se puede definir como la carga mental que producen los diagramas de clases UML en las personas que tienen que tratar con él (por ejemplo, modeladores, mantenedores). Esto significaría que un diagrama de clases UML con una complejidad cognitiva alta será más difícil de comprender y de modificar, afectando así a su mantenibilidad.

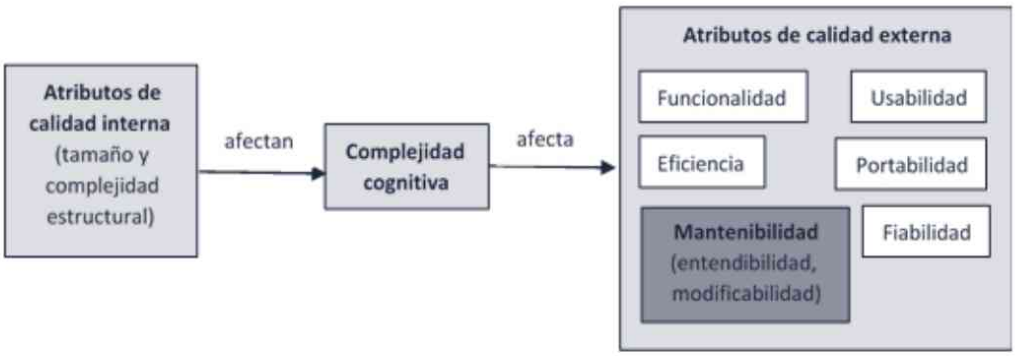


Figura 7.2. Relación entre las propiedades estructurales: la complejidad cognitiva y los atributos externos de la calidad

A partir del modelo de la Figura 7.2, primeramente se definen medidas para la complejidad estructural de los diagramas de clases UML. Se consideró que la complejidad estructural del diagrama de clases UML está determinada por los diferentes elementos que lo componen, como las clases, atributos, relaciones, etc. Fenton ha demostrado formalmente que una sola medida de la complejidad no puede captar todos los posibles aspectos o puntos de vista sobre complejidad (Fenton, 1994), por ello no es recomendable definir una sola medida para la complejidad estructural de los diagramas de clases UML.

Antes de definir las medidas que aquí se presentan, se realizó un exhaustiva revisión de la literatura, y se llegó a la conclusión de que no existían medidas validadas que midieran la complejidad estructural de los diagramas de clases debido al uso de los diferentes tipos de relaciones que proporciona UML.

7.2.2 Creación

A continuación se presentará la definición de las medidas y la validación teórica y empírica.

7.2.2.1 DEFINICIÓN

La definición de las medidas de la complejidad estructural de los diagramas de clases UML se presenta en la Tabla 7.1.

Nombre	Definición
Número de asociaciones (*NAssoc*)	El número total de asociaciones.
Número de agregaciones (*NAgg*)	El número total de relaciones de agregación dentro de un diagrama de clases (cada par todo-parte en una relación de agregación).
Número de dependencias (*NDep*)	El número total de relaciones de dependencia.
Número de generalizaciones (*NGen*)	El número total relaciones de generalización dentro de un diagrama de clases (cada par padre-hijo en una relación de generalización).
Número de jerarquías de agregación (*NAggH*)	El número total de jerarquías de agregación (estructuras todo-partes) en un diagrama de clases.
Número de jerarquías de generalización (*NGenH*)	El número total de jerarquías de generalización dentro de un diagrama de clases.
Máxima profundidad de herencia (*MaxDIT*)	Es el máximo valor de *DIT* (Profundidad del árbol de herencia) calculado para cada clase del diagrama de case. El valor de DIT para cada clase dentro de una jerarquía de generalización es el paso más largo desde la clase hasta la raíz de la jerarquía.
Máxima altura de agregación (*MaxHAgg*)	Es el valor máximo de *HAgg* obtenido para cada clase del diagrama de clases. El valor de *HAgg* para una clase dentro de una jerarquía de agregación es el paso más largo desde la clase hasta las hojas.

Tabla 7.1. Medidas para la complejidad estructural de los diagramas de clases UML

7.2.2.2 VALIDACIÓN TEÓRICA

Para realizar la validación teórica de las medidas presentadas en la Tabla 7.1, se seleccionaron dos marcos teóricos:

- El marco *DISTANCE* basado en la teoría de la medición (Poels, 1999; Poels y Dedene, 2000a) con el objetivo de determinar la escala de las medidas.
- El marco de Briand *et al.* (1996), basado en propiedades, para determinar los atributos que miden las medidas definidas.

En la Tabla 7.2 se resume la validación teórica realizada.

Medidas	Marco basado en propiedades	Marco DISTANCE
NAssoc	Complejidad	Ratio
NAgg	Complejidad	Ratio
NGen	Complejidad	Ratio
NDep	Complejidad	Ratio
NGenH	Tamaño	Ratio
NAggH	Tamaño	Ratio
MaxDIT	Longitud	Ratio
MaxHAgg	Longitud	Ratio

Tabla 7.2. Resumen de la validación teórica

7.2.2.3 VALIDACIÓN EMPÍRICA

En este apartado se persigue el objetivo de validar empíricamente las medidas propuestas (ver Tabla 7.1) y además se añaden medidas tradicionales de tamaño (ver Tabla 7.3).

Nombre	Definición
Número de clases (*NC*)	El número total de clases
Número de atributos (*NA*)	El número total de atributos
Número de métodos (NM)	El número total de métodos

Tabla 7.3. Medidas de tamaño para diagramas de clases UML

Como se comentó anteriormente, la definición de las medidas tuvo en cuenta el modelo presentado en la Figura 7.2, por ello concretamente la familia de experimentos se realizó para corroborar las siguientes hipótesis basadas en las flechas de la Figura 7.2: 1) El tamaño y la complejidad estructural de los diagramas de clases afecta la complejidad cognitiva y 2) La complejidad cognitiva afecta a dos sub-características de la mantenibilidad, la comprensibilidad y la modificabilidad de los diagramas de clases UML.

Para medir el contenido de cada caja de la Figura 7.2 se definieron algunas medidas que se introducirán más adelante.

Una vez descritas las características de la familia de experimentos y llevado a cabo el análisis individual de los datos, se presentará un estudio de meta-análisis con el objetivo de extraer conclusiones más sólidas.

La familia de experimentos, que se llevó a cabo siguiendo el proceso experimental descrito en el capítulo 3, consistía de 5 experimentos (ver Tabla 7.4).

Estudios	Sujetos	Universidad	Fecha	Curso
E1	72	Universidad de Sevilla (España)	Marzo 2003	4to
R1	28		Marzo 2003	
E2	38	Universidad de Castilla-La Mancha (España)	Abril 2003	3ro
R21	23	Universidad de Sannio (Italia)	Junio 2003	4to
R22	71	Universidad de Valladolid (España)	Septiembre 2005	3ro

Tabla 7.4. Características de la familia de experimentos

A continuación resumiremos la principales características del diseño y ejecución del experimento y su análisis de datos tanto individual como la integración a través de meta-análisis.

Planificación de los experimentos

A continuación se describe el marco común a todos los experimentos:

- **Preparación**. Esta familia de experimentos persigue los siguientes objetivos:
  - *Objetivo 1*: Analizar la complejidad estructural de los diagramas de clases UML con respecto a su relación con la complejidad cognitiva desde el punto de vista de los modeladores y diseñadores de *software* en un contexto académico.
  - *Objetivo 2*: Analizar la complejidad cognitiva de los diagramas de clases UML con respecto a su relación con la comprensibilidad y modificabilidad desde el punto de vista de los modeladores y diseñadores de *software* en un contexto académico.
- **Definición del contexto**. Se consideraron estudiantes de grado como sujetos experimentales con conocimientos suficientes para resolver las tareas requeridas.
- **Material**. El material experimental consistía en un conjunto de diagramas de clases UML adecuados para los objetivos de los experimentos. Estos diagramas cubrieron un amplio rango de valores de la medidas (ver Tablas 7.1 y 7.3) considerando tres tipos de diagramas:

Difíciles de mantener (D), Fácil de mantener (F) y Moderadamente difícil de mantener (M). Algunos de ellos se diseñaron específicamente para los experimentos y otros se obtuvieron de aplicaciones reales. Cada diagrama tenía documentación adjunta, conteniendo, entre otras cosas, 4 tareas de comprensión y 4 tareas de modificación.

Ejecución de cada experimento y sus réplicas

Ahora vamos a explicar el plan experimental de los diferentes miembros de la familia de los experimentos. Las variables consideradas para la medición de la complejidad estructural y tamaño eran el conjunto de 11 medidas presentadas en las Tablas 7.1 y 7.3.

Para medir la complejidad cognitiva se utilizó la medida *CompSub*, definida como la percepción subjetiva de los sujetos con respecto a la complejidad de los diagramas con los que tienen que trabajar durante las tareas experimentales. Consideramos *CompSub* a ser una medida de la complejidad cognitiva. Los valores admisibles de esta variable son: muy simple, moderadamente simple, media, moderadamente compleja y muy compleja. Para medir la comprensión y modificabilidad de diagramas de clases UML se consideró hemos considerado el tiempo (en segundos) que tardó cada sujeto para completar las tareas de comprensión y modificabilidad. Llamamos a estas medidas: *Tiempo de comprensión* y *Tiempo de modificación*.

Se eligió un diseño balanceado inter-sujetos, en el que cada sujeto trabajó con un único diagrama que le fue asignado de manera aleatoria.

Se formularon las siguientes hipótesis derivadas a partir de los objetivos de la familia de experimentos:

- $\mathbf{H_{0,1}}$: La complejidad estructural y el tamaño de los diagramas de clases UML no están correlacionados con la complejidad cognitiva. $H_{1,1}$: $\neg$ $H_{0,1}$
- $\mathbf{H_{0,2}}$: La complejidad cognitiva de los diagramas de clases UML no está correlacionada con su comprensibilidad y modifcabilidad. $H_{1,2}$: $\neg$ $H_{0,2}$

Todos los experimentos se supervisaron y se realizaron en un tiempo limitado. Más detalles de los mismos se puede encontrar Genero *et al.* (2004; 2007). Para el análisis de datos se utilizó el paquete SPSS (SPSS, 2003) y para el estudio de meta-análisis la herramienta *Comprehensive Meta-Analysis* (Biostat, 2006).

Primer experimento (E1) y su réplica (R1)

Los resultados obtenidos tras de corroborar las hipótesis fueron:

- Correlación entre la complejidad estructural y el tamaño y la complejidad cognitiva (Hipótesis 1- Objetivo 1). En E1 se encontró una correlación significativa a un nivel de significación 0,05 entre la variable *CompSub* y las 11 medidas, mientras que en R1 existió dicha correlación significativa excepto para las medidas *NM*, *NGen* y *MaxDIT*.
- Correlación entre la complejidad cognitiva y la comprensibilidad y la modificabilidad (Hipótesis 2 – Objetivo 2). La Tabla 7.5 indica que para E1, la complejidad subjetiva esta significativamente correlacionada con la comprensibilidad, mientras que para R1 los resultados no fueron significativos. Al mismo tiempo, no se encontró correlación con el tiempo de modificación de los diagramas de clases UML. Esto puede haber ocurrido debido a que los sujetos basaron su percepción sobre la dificultad teniendo en cuenta las primeras tareas que tuvieron que realizar, que en este caso fueron tareas de comprensión.

Variables correlacionadas	**E1 (n=62)**		**R1(n= 22)**	
	$\rho_{spearman}$	**p-valor**	**$\rho_{spearman}$**	**p-valor**
CompSub vs Comprensibilidad	0,266	0,037	0,348	0,111
CompSub vs Modificabilidad	0,132	0,306	0,270	0,217

Tabla 7.5. Resultados obtenidos en E1 y R1 para el objetivo 2 (*valores significativos en negrita*)

Segundo experimento (E2) y sus réplicas (R21 y R22)

Los objetivos y variables son las mismas que en los estudios previos, pero los diagramas fueron diferentes y se mejoró el contexto y el diseño. Más detalle sobre estos estudios se puede encontrar en Genero *et al.* (2007). Además de las variables presentadas para toda la familia, en estos estudios le consideraron las siguientes variables dependientes:

- **CompCorreción** = número de tareas de comprensión correctas/número de tareas realizadas.
- **CompCompleción** = número de tareas de comprensión correctas / número total de tareas de completión.

- **ModifCorrección** = número de tareas de modificación correctas / número de tareas de modificación realizadas.
- **ModifCompleción** = número de tareas de modificación correctas / número de tareas de modificación.

Al igual que en los estudios previos, se eligió un diseño inter-sujetos, aunque en este caso se formaron grupos equilibrados según el nivel de experiencia de los sujetos. Antes de ejecutar los experimentos los sujetos debieron completar un cuestionario con datos personales y preguntas para medir su experiencia con los diagramas de clases UML, cuyos resultados sirvieron para formar los grupos.

Se excluyeron los datos de aquellos sujetos que obtuvieron medidas de la corrección y compleción por debajo de un 75%.

Al realizar los test de la hipótesis se obtuvieron los siguientes resultados:

- Correlación entre la complejidad estructural y el tamaño con respecto a la complejidad cognitiva (Hipótesis 1- Objetivo 1). La Tabla 7.6 resume las medidas que están correlacionadas significativamente con la variable *CompSub*. Esto refleja que se obtuvieron resultados favorables que admiten una correlación entre la complejidad estructural, y el tamaño de los diagramas de clases UML con respecto a su complejidad cognitiva. La mayoría de las medidas estaban significativamente correlacionadas con la complejidad subjetiva en los diferentes estudios, especialmente las medidas relacionadas con las jerarquías de herencia.

Estudio	Medidas correlacionadas significativamente
E2	NC, NAssoc, NGen, NGenH, MaxDIT (5 de 11)
R21	Todas excepto *NM*: *NGenH* y *MaxAgg* (8 de 11)
R22	Todas excepto *NM* (10 de 11)

Tabla 7.6. Resultados para el objetivo en E2: R21 y R22

- Correlación entre la complejidad cognitiva y la comprensibilidad y la modificabilidad (Objetivo 2). La Tabla 7.7 indica que para todos los estudios, la complejidad subjetiva está significativamente correlacionadas con la comprensibilidad. Para la modificabilidad, los resultados no fueron significativos.

Variables correlacionadas	E2			R21			R22		
	$\rho_{spearman}$	p-valor	N	$\rho_{spearman}$	p-valor	N	$\rho_{spearman}$	p-valor	N
CompSub vs Comprensibilidad	0.343	0.049	33	0.410	0.065	21	0.353	0.003	70
CompSub vs Modificabilidad	0.337	0.099	25	0.156	0.500		0.165	0.173	

Tabla 7.7. Resultados relacionados con el objetivo 2 para E2: R21 y R22

Estudio de meta-análisis

A continuación, presentaremos el meta-análisis realizado con los datos de los cinco experimentos presentado en Manso *et al.* (2009). No entraremos demasiados detalles teóricos sobre el meta-análisis ya que se presentarán en el capítulo 3. En este meta-análisis se utilizaron los coeficientes de correlación (r_i) que, una vez transformados (transformación de *Fisher*), proporcionan los tamaños del efecto que tienen una distribución normal (z_i), que permite que sean más fáciles de usar. El tamaño del efecto global se obtiene usando la medida *g* de Hedges (Hedges y Olkin 1985; Kampenes *et al.*, 2007; Ellis, 2010), al igual que en el ejemplo presentado en el capítulo 3.

Resultados del meta-análisis

Primero, se ha realizado el meta-análisis para cada par medida-*CompSub*, considerando como hipótesis nula que la correlación está por debajo del 0. En la Tabla 7.8 se presenta la estimación global del coeficiente de correlación, un intervalo de confianza del 95%, el p-valor y el valor de la *g* de Hedges, incluyendo la clasificación del tamaño del efecto en grande (G), mediano (M) o pequeño (P).

H0: ρ≤0	Correlación (ρ) Tamaño del efecto global	Límite inferior	Límite superior	p-valor	g de Hedges
NC	0,566	0,464	0,653	0,0000	1,322(G)
NA	0,541	0,435	0,632	0,000	1,219(G)
NM	0,177	0,040	0,307	0,012	0,339(P)
NAssoc	0,566	0,465	0,653	0,000	1,318(G)
NAgg	0,481	0,368	0,581	0,000	1,051(M)
NDep	0,484	0,371	0,584	0,000	1,060(M)
NGen	0,484	0,371	0,584	0,000	1,018 (G)
NGenH	0,422	0,302	0,529	0,000	0,903 (M)
NAggH	0,393	0,270	0,504	0,000	0,814 (M)
MaxDIT	0,492	0,379	0,590	0,000	1,080 (G)
MaxHAgg	0,360	0,233	0,474	0,000	0,734 (M)

Tabla 7.8. Meta-análisis para las medidas-CompSub

Los resultados observados en la Tabla 7.8 están a favor de la existencia de correlación positiva entre la complejidad cognitiva y las 11 medidas que miden la complejidad estructural (ver Tabla 7.1) y el tamaño (ver Tabla 7.3) de los diagramas de clases UML. En efecto, la mayoría de los tamaños del efecto son medianos o grandes, a excepción de *NM*, que es pequeño. Las medidas de tamaño que más influencia tienen sobre la complejidad cognitiva son *NC* y *NA*, mientras que las de complejidad estructural que tiene más influencia sobre la complejidad cognitiva están relacionadas con las agregaciones (*NAgg*) y las generalizaciones (*NGen* y *MaxDIT*). Esto nos lleva a concluir que aquellos diagramas con muchas clases y atributos tendrán una mayor complejidad cognitiva, al igual que los que hagan mucho uso de mecanismos de herencia y agregación.

Con respecto a las hipótesis derivadas del objetivo 2, la Tabla 7.9 muestra que existe correlación entre la complejidad cognitiva y el Tiempo de comprensión y de modificación. En ambos casos el tamaño del efecto es mediano, pero la estimación de la correlación es mayor para la comprensión que para la modificación. Por ello, podemos concluir que mientras mayor sea la complejidad cognitiva de los diagramas de clases UML, más difícil será su comprensión y modificación.

H0: ρ≤0	Correlación (ρ) Tamaño del efecto global	Límite inferior	Límite superior	p-valor	g de Hedges
Tiempo de comprensión	0,330	0,200	0,449	0,000	0,684 (M)
Tiempo de modificación	0,186	0,044	0,320	0,011	0,368(M)

Tabla 7.9. Meta-análisis de CompSub-Tiempo de comprensión y Tiempo de modificación

A modo de ejemplo se muestra un gráfico que muestra el resultado del meta-análisis para la relación entre algunas medidas y *CompSub*, así como la relación entre la comprensibilidad y la complejidad cognitiva.

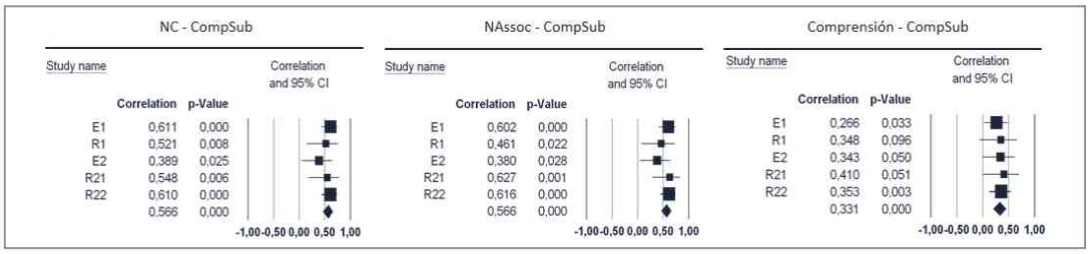


Figura 7.3. Meta-análisis para NC-CompSub: *NAssoc-CompSub y CompSub-Comprensión*

Conclusiones del meta-análisis

Los resultados obtenidos permiten validar el modelo presentado en la Figura 7.2 junto a las respectivas medidas utilizadas. Por un lado podemos decir que, la complejidad estructural está correlacionada con la complejidad cognitiva, especialmente debido al uso de asociaciones y generalizaciones. Y por otro lado podemos concluir que la complejidad cognitiva afecta tanto a la comprensibilidad como a las modificabilidad, pero en mayor medida a la comprensibilidad. Estos resultados son útiles ya que pueden servir para controlar ciertos atributos de calidad de los diagramas de clases UML en la etapa de modelado. También los resultados obtenidos pueden tener implicación para la enseñanza, ofreciendo información de que elementos de UML pueden tener mayor influencia en el esfuerzo de comprender y modificar los diagramas de clases UML. En el caso de tener diseños alternativos de diagramas de clases UML, es recomendable seleccionar el que minimice estos elementos. Además las medidas relacionadas con las asociaciones y generalizaciones se podrían utilizar para construir modelos de predicción como se ha presentado en (Genero *et al.*, 2007).

7.2.3 Aceptación

Las medidas definidas para la complejidad estructural (ver Tabla 7.1.) y otras medidas tradicionales de tamaño (ver Tabla 7.3) se consideraron en el proyecto MEDUSAS (financiado por el CDTI), llevado a cabo por miembros del Grupo Alarcos y varias empresas. El objetivo principal de este proyecto fue la creación y validación de modelos de calidad y medidas para la mantenibilidad, usabilidad y seguridad del *software*. En este proyecto se refinaron las medidas propuestas, se crearon plantillas y herramientas para su recolección y se definieron valores umbrales para las medidas.

7.2.4 Aplicación

Una vez refinadas las medidas fueron utilizadas en estudios de casos llevados durante la ejecución del proyecto MEDUSAS. Una vez finalizados los estudios de casos, se extendió la utilización de las medidas a distintas empresas desarrolladoras de *software*.

7.2.5 Acreditación

De momento, las empresas que usan las medidas para diagramas de clases UML, pretenden seguir aplicándolas, aunque el lenguaje UML siga evolucionando parece que las medidas propuestas serán válidas y que, en todo caso, deberán completarse con otras nuevas.

7.3 MÉTODO PARA LA MEJORA DE PROCESOS SOFTWARE

En este apartado, describimos una combinación de los métodos Investigación-Acción y Estudios de casos, que fue la estrategia empleada en el proyecto COMPETISOFT (Oktaba *et al.*, 2007), que ha permitido incrementar el nivel de competitividad de las pequeñas empresas *software* de Iberoamérica. Después de una revisión sistemática de la literatura, se combinaron los métodos de investigación-acción y estudios de caso, para desarrollar, aplicar, validar y refinar los componentes que forman parte de su marco metodológico en pequeñas empresas.

Los resultados principales de este proyecto, desde el punto de vista profesional se recogen en el libro publicado por Oktaba *et al.* (2008), y desde el punto de vista investigador en la tesis doctoral de Pino (2010) y sus correspondientes publicaciones científicas.

7.3.1 Mejora de procesos en PyMEs

En primer lugar se llevó a cabo una revisión sistemática de la literatura (Pino *et al.*, 2008), con el fin de contestar a la pregunta de investigación: ¿Qué enfoques existen relativos a la mejora de procesos *software* que se centren en PyMEs y que presenten estudios de casos?

Para ello se usaron como cadenas básicas de búsqueda las presentadas en la Tabla 7.10:

1	"software process improvement" AND (small AND (enterprises OR organizations OR companies OR team OR firms OR settings))
2	(small AND (enterprises OR organizations OR companies OR team OR firms OR settings)) AND (CMM OR CMMI or 15504 OR SPICE OR 9001 OR 12207)

Tabla 7.10. Cadenas de búsqueda

Después de aplicar el método descrito en el capítulo 6, se encontraron 743 estudios, de los 400 fueron no repetidos y 45 se obtuvieron como primarios (ver Tabla 7.11).

Fuentes	Estudios					%
	Fecha de búsqueda	Encontrados	No repetidos	Relevantes	Primarios	
Science@Direct	06/03/2006	157	42	4	4	8,9
Wiley InterScience	31/03/2006	76	30	17	15	33,3
IEEE Digital Library	03/04/2006	208	128	22	19	42,2
ACM Digital Library	18/04/2006	272	170	7	1	2,2
Procedings	30/04/2006	30	30	6	6	13,3
Total		743	400	56	45	100

Tabla 7.11. Distribución de artículos por fuente de búsqueda

A partir de estos estudios se obtuvieron resultados diversos como: la tendencia de publicaciones sobre mejora de procesos *software* en PyMEs, las empresas involucradas, los modelos utilizados (CMM, CMMI, IDEAL, ISO 15504, etc.), los procesos en los que se centra la mejora, los factores críticos de éxito, etc.

7.3.2 Marco metodológico de COMPETISOFT

El Marco Metodológico de COMPETISOT, describe tres componentes: el Modelo de Referencia de Procesos, el Modelo de Mejora de Procesos y el Modelo de Evaluación de Procesos (ver Figura 7.4).

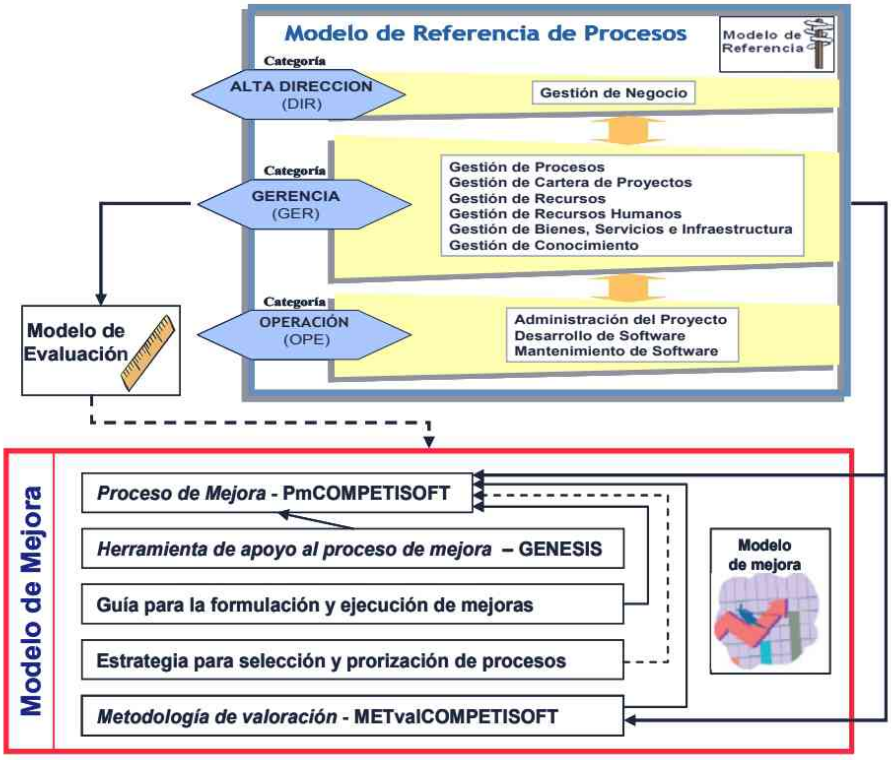


Figura 7.4. Marco Metodológico de COMPETISOFT

El Modelo de Referencia agrupa los procesos en tres categorías: Alta dirección, Gestión y Operación. La categoría de Alta Dirección incluye un proceso que engloba las prácticas relacionadas con la gestión de la empresa. La Categoría de Gerencia, compuesta por seis procesos, incluye las prácticas necesarias para la gestión de procesos, proyectos y recursos en función de las directrices establecidas desde la Alta Dirección. En la Categoría de Operación se incluyen las prácticas de los proyectos de desarrollo y mantenimiento de *software*.

En el Modelo de Referencia las actividades relacionadas con la mejora de procesos están descritas de forma general en el proceso de Gestión de Procesos. Sin embargo para guiar de manera explícita y detallada la implementación de estas prácticas en el contexto de las pequeñas organizaciones, se desarrolló el Modelo de Mejora.

Con el propósito de permitir el reconocimiento mutuo de las evaluaciones formales de COMPETISOFT entre diferentes países se sugiere que cada país defina su propio Modelo de Evaluación, el cual debe ser conforme con la norma internacional ISO/IEC 15504 (ISO, 2004).

7.3.3 Investigación-acción en COMPETISOFT

La Figura 7.5 muestra la identificación de los elementos de investigación-acción en el proyecto COMPETISOFT.

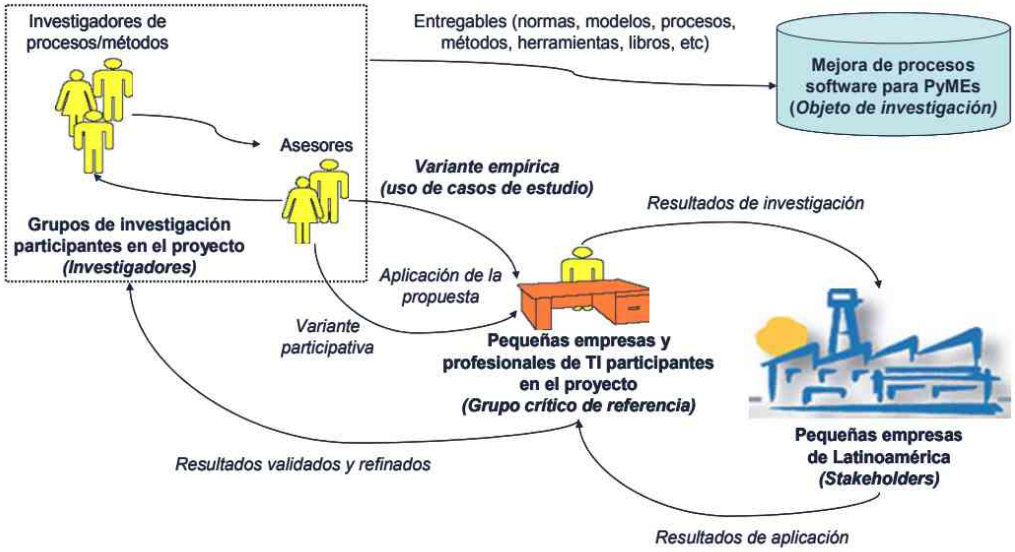


Figura 7.5. Aplicación de investigación-acción y estudios de casos en COMPETISOFT

En COMPETISOFT el *objeto investigado* o problema a resolver es la mejora de procesos *software* en el contexto de las pequeñas empresas. Los *investigadores* que han llevado a cabo de forma activa el proceso investigador, son los de las diferentes universidades participantes del proyecto. Este grupo fue dividido en dos: (i) *investigadores de procesos/métodos*, que fueron los encargados de desarrollar los componentes del marco metodológico de COMPETISOFT, y (ii) *asesores*, que fueron los investigadores de campo encargados de llevar a cabo la aplicación del marco metodológico en las empresas del *grupo crítico de referencia*. El *grupo crítico de referencia* involucra a las pequeñas empresas y sus profesionales de tecnologías de la información participantes del proyecto. En este

grupo se aplicaron las propuestas de investigación desarrolladas para abordar el objeto investigado, con el objetivo de tener una realimentación inicial para refinar, mejorar y validar las propuestas. Finalmente, los *beneficiarios de la investigación* (*stakeholders*) realizada son las pequeñas empresas desarrolladoras de *software* de Iberoamérica.

En lo que respecta a las actividades empleadas para el método de Investigación-Acción, se siguieron los pasos descritos en el capítulo 5:

- **Planificación**. En esta actividad se distinguió entre diagnóstico (identificar los problemas iniciales) y la planificación en sí (especificar acciones para resolver dichos problemas). El problema inicial identificado en COMPETISOFT fue que las pequeñas empresas intentaban mejorar sus procesos usando estándares y modelos propuestos por instituciones como el SEI e ISO, que no resultan adecuados para este tipo de organizaciones. Para resolver esta situación en COMPETISOFT se asignaron los roles y planificaron las actividades conducentes a desarrollar un marco metodológico para mejora de procesos ajustado a la realidad socio-económica de las pequeñas empresas iberoamericanas, el cual integrara diferentes propuestas desarrolladas previamente en el contexto iberoamericano sobre mejora de procesos.

- **Acción**. En COMPETISOFT los investigadores intervinieron en el grupo crítico de referencia en diferentes casos.

- **Observación**. Para documentar lo ocurrido, tomar datos y recolectar información de la intervención realizada en el grupo crítico de referencia se utilizó el método de estudio de casos.

- **Reflexión**. Fue un proceso continuo, que ocurrió durante toda la ejecución del proyecto, para compartir y analizar los resultados obtenidos con los involucrados en la investigación. Esta actividad permitió reflexionar y profundizar en las prácticas relacionadas con la mejora de procesos *software* llevadas a cabo en las pequeñas empresas. Además, mediante la realimentación obtenida de las personas involucradas en el proyecto, se logró refinar, mejorar y estabilizar los componentes propuestos por el marco metodológico.

En COMPETISOFT se han utilizado las variantes participativa y empírica de investigación-acción; participativa porque el *grupo crítico de referencia* puso en práctica con apoyo del *asesor* los modelos del marco metodológico (productos de investigación) realizados por los *investigadores de procesos/métodos*, compartiendo con ellos sus efectos y resultados; y empírica porque con el apoyo

del *asesor* y mediante la utilización del método de estudio de casos, el *grupo crítico de referencia* ha realizado un registro amplio y sistemático de sus acciones usando estos modelos.

7.3.4 Estudio de casos en COMPETISOFT

El método de estudio de casos se ha utilizado para conducir la aplicación del marco metodológico de COMPETISOFT en el contexto real de ocho pequeñas empresas. A continuación se describen los estudios de casos en términos de: antecedentes, diseño, sujetos de investigación, procedimiento de campo, recolección de datos, análisis, validez y limitaciones. La Figura 7.6 muestra las actividades, responsables y línea de tiempo de los estudios de casos.

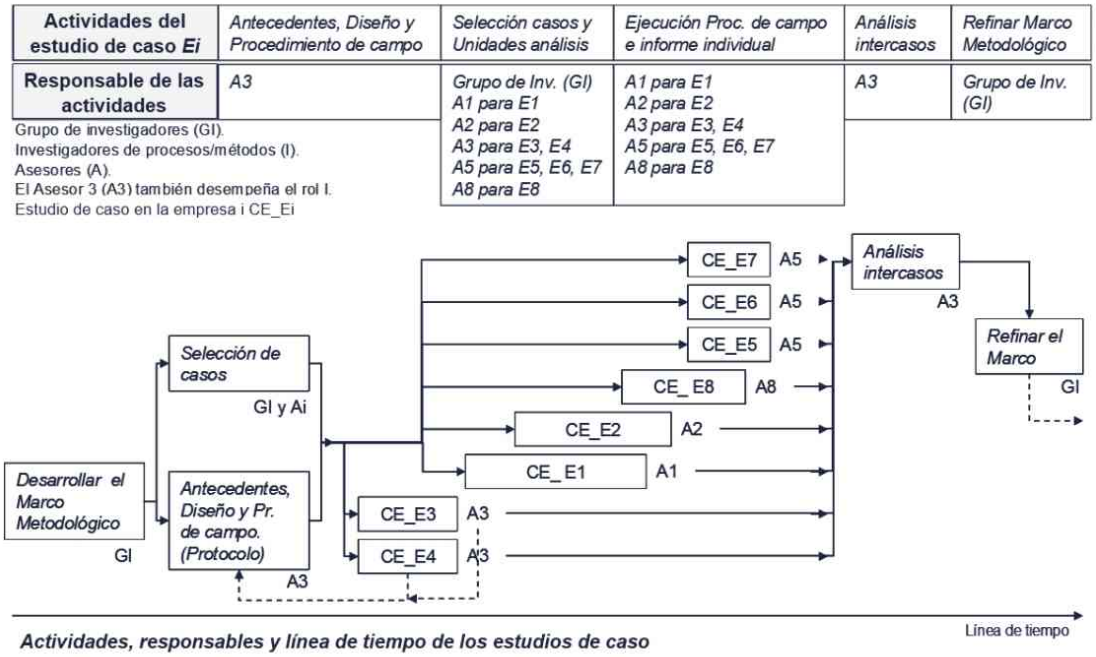


Figura 7.6. Visión general de los estudios de caso llevados a cabo en COMPETISOFT

7.3.4.1 ANTECEDENTES

La *pregunta de investigación principal* (PP) y *las preguntas de investigación adicionales* (PA) que dirigen los estudios de casos, se presentan en la Tabla 7.12.

Preguntas de investigación (principal y adicionales)	
PP	¿Es el marco metodológico de COMPETISOFT adecuado para llevar a cabo esfuerzos de mejora de procesos en pequeñas empresas *software*?
PA1	¿Es el esfuerzo de aplicar mejora de procesos *software* siguiendo el marco metodológico de COMPETISOFT apropiado para las pequeñas empresas?
PA2	¿El uso del marco metodológico de COMPETISOFT posibilita incrementar la capacidad de los procesos de las pequeñas organizaciones?

Tabla 7.12. Preguntas de investigación de los estudios de caso

7.3.4.2 DISEÑO

El tipo de diseño del estudio de casos para este trabajo, es de múltiples casos embebidos, ya que el marco metodológico de COMPETISOFT ha sido aplicado en el contexto de ocho diferentes empresas con el objetivo de mejorar uno o varios de los procesos (unidades de análisis diferentes) descritos por el Modelo de Referencia de COMPETISOFT. El objeto de estudio es el marco metodológico para la mejora de procesos *software* desarrollado en el proyecto COMPETISOFT (específicamente el Modelo de Referencia y el Modelo de Mejora). Las medidas usadas para indagar sobre las preguntas de investigación son: (i) el esfuerzo de las actividades realizadas para llevar a cabo la mejora de los procesos seleccionados por las pequeñas empresas (estas actividades están descritas en el proceso PmCOMPETISOFT del Modelo de Mejora), y (ii) el nivel de capacidad de los procesos seleccionados para mejorarlos (este nivel de capacidad de mide utilizando METvalCOMPETISOFT del Modelo de Mejora).

7.3.4.3 SUJETOS DE INVESTIGACIÓN Y UNIDADES DE ANÁLISIS

El *criterio para la selección de los estudios de casos* fue: pequeñas empresas participantes del proyecto comprometidas con la realización de un ciclo de mejora de procesos de al menos tres meses. Así que para estos estudios de casos se trabajó con ocho empresas de Argentina, Chile, España y Colombia, que hacían parte del grupo crítico de referencia (ver Tabla 7.13). En cada una de estas empresas, las *unidades de análisis* fueron: (i) los procesos a mejorar seleccionados del Modelo de Referencia (procesos bajo intervención), y (ii) las actividades y estrategias descritas por el Modelo de Mejora para guiar la mejora de procesos en estas empresas.

Emp	País	Emple	Exper.	Área principal de mercado
E1	Argentina	8 (7)	16 años / N&I	Desarrollo de productos *software* a medida mediante tecnología de última generación
E2	Chile	18 (12)	10 años / N&I	Desarrollos de productos *software* y sistemas para la industria agrícola (vino y alimentos)
E3	España	7 (6)	5 años / N	Desarrollo de aplicaciones WEB
E4	España	21 (15)	13 años / N	Desarrollo de productos *software* mediante contratos y acuerdos con organizaciones públicas
E5	Colombia	4 (4)	3 años / N	Desarrollo de aplicaciones WEB para gestión y control de sistemas de gestión de calidad
E6	Colombia	6 (6)	3 años / N	Desarrollo de aplicaciones WEB orientadas a servicios agropecuarios
E7	Colombia	4 (4)	2 años / N	Desarrollo de aplicaciones para dispositivos móviles
E8	Argentina	12 (5)	4 años / N&I	Desarrollo de productos *software* a medida en el área de gestión comercial

Emple: Número de empleados en la empresa (Personal en desarrollo y mantenimiento).
Exper: Número de años de existencia de la Emp./Alcance del mercado de sus productos (Nacional–N/Internacional–I).

Tabla 7.13. Características de las organizaciones involucradas en los estudios de caso

Durante la actividad de reflexión de Investigación-Acción, se consensuó que para llevar a cabo el primer ciclo de mejora las empresas debían tener en cuenta las siguientes directrices:

- Seleccionar para mejorar cualquiera de los procesos de la categoría de operación del Modelo de Referencia, con el fin de alinearse con el perfil básico del informe técnico ISO/IEC 29110 (ISO, 2011). Es decir, las *unidades de análisis* serían los procesos: Administración del Proyecto (AP), Desarrollo de Software (DS) y/o Mantenimiento de Software (MS).
- Utilizar el proceso PmCOMPETISOFT (otra *unidad de análisis*) y los demás componentes de Modelo de Mejora para guiar la implantación de la mejora de los procesos escogido por las empresas.

7.3.4.4 PROCEDIMIENTO DE CAMPO

El procedimiento que rige las actividades de campo de los estudios de casos está directa y estrechamente relacionado con las actividades, sub-procesos, roles y productos de trabajo del proceso PmCOMPETISOFT. Una descripción general de este proceso se puede ver en la Figura 7.7, la descripción detalla se puede encontrar en Pino *et al.* (2007).

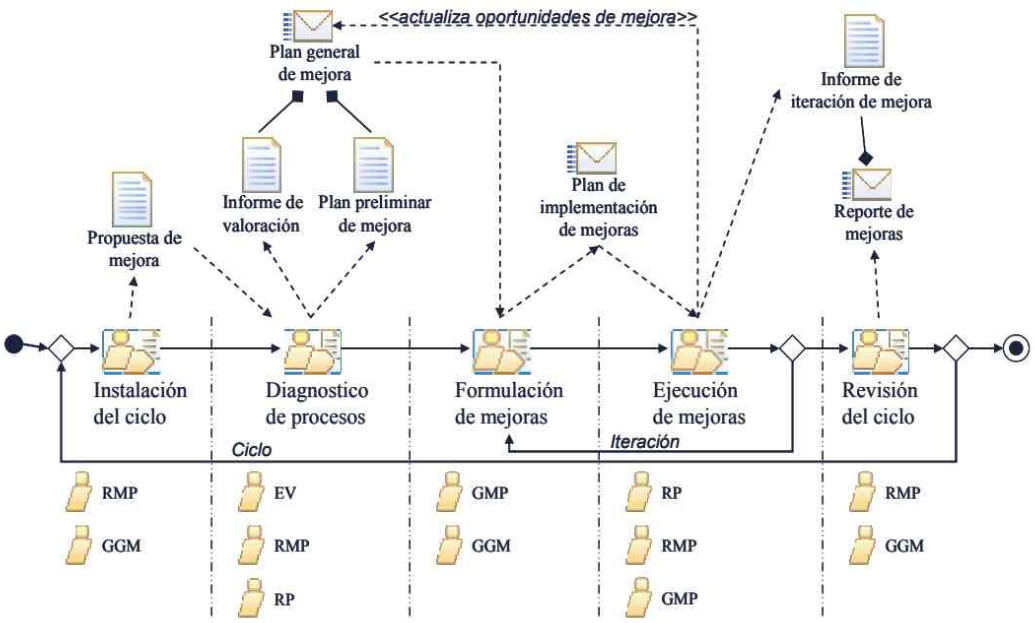


Figura 7.7. Procedimiento de campo que rige las actividades de los estudios de casos

En el procedimiento para llevar a cabo las mejoras hay cinco roles: Responsable de mejora de procesos (RMP), Grupo de gestión de mejora (GGM), Evaluador (EV), Responsables del proceso (RP) y Grupo de mejora de procesos (GMP). Cada ciclo de mejora consta de cinco actividades: Instalación, Diagnóstico, Formulación, Ejecución y Revisión (las actividades de Formulación y Ejecución forman la iteración de mejora). Los productos de trabajo son: la propuesta de mejora, el plan general de mejora (formado por el informe de valoración y el plan preliminar de mejora), el plan de implementación de mejoras y el reporte de mejora (formado por los informes de cada iteración realizada). Cada producto de trabajo tiene su propia plantilla auto-contenida.

Cada una de las empresas comenzó el ciclo de mejora de procesos apoyada por un *asesor* en mejora de procesos del grupo de *investigadores* de COMPETISOFT. Además, cada empresa asignó de acuerdo a sus características los roles descritos por PmCOMPETISOFT (en algunas empresas una persona desempeñó más de un rol). El *asesor* apoyó al GGM en la actividad de instalación, fue el evaluador en la actividad de Diagnóstico, e hizo parte de GMP en la iteración de mejora.

7.3.4.5 RECOLECCIÓN DE DATOS

La recolección de datos se hizo mediante la utilización de plantillas auto-contenidas de los productos de trabajo, un ejemplo de las cuales se puede ver en la Figura 7.8. A partir de la información registrada en estas plantillas, se ha

consolidado: (i) el nivel inicial (al comenzar el ciclo de mejora) y final (después de haber realizado la intervención mediante los estudios de casos) de la capacidad de los procesos de las empresas (ver Tabla 7.14), y (ii) el esfuerzo de llevar a cabo la mejora de procesos siguiendo el Modelo de Mejora y teniendo como referente el Modelo de Referencia (ver Tabla 7.15). Además cada asesor escribió un informe individual de su estudio de caso.

Mejora de Procesos para Fomentar la Competitividad de la Pequeña y Mediana Industria del Software de Iberoamérica **- Proyecto COMPETISOFT (Financiado por CYTED)**

Plan Preliminar de Mejora. Se planifica de manera general las iteraciones a realizar para mejorar los procesos de la organización.

Ciclo de mejora

Nombre de la empresa	
Nombre del proyecto de mejora	
Nombre del responsable de la empresa	
Nombre del responsable de COMPETISOFT	

Nivel de capacidad actual y esperado de los procesos a mejorar en este Ciclo

Procesos a mejorar	*Nivel de Capacidad Actual*	*Nivel de Capacidad Esperado*

Iteraciones del Ciclo de mejora

Número de iteraciones del ciclo de mejora	

Para llevar a cabo la mejora en los procesos descritos para el primer ciclo de mejora se han definido X iteraciones, de la siguiente forma:

Iteración	Proceso	Duración	Desde
1			
…			

Planeación general de las iteraciones del Ciclo de mejora.

Plan de manejo de riesgos del Ciclo de mejora

Plan de capacitación del Ciclo de mejora

Plan de mediciones del Ciclo de mejora

Estimación del esfuerzo realizado en esta actividad

Fecha	*Actividad*	*Nombre o Rol de las personas involucradas*	*Horario*	*Tiempo Asesor (minutos)*	*Tiempo Empresa (minutos)*
			Total separado		
			Total consolidado		

Otra información relevante (Registre aquí otra información que considere relevante)

Aprobación del Plan Preliminar de Mejora

Firma del Responsable de Mejora de la Empresa
Fdo:

Firma del asesor de COMPETISOFT
Fdo:

Sugerencias de mejora a esta plantilla y la actividad asociada

Figura 7.8. Plantilla del plan preliminar de mejora

Emp.	Valor	Nivel de Capacidad de los Procesos								
		OPE			DIR	GER				
		DS	AP	MS	GN	GP	GCP	GRH	GC	GBSI
E1	Inicial	-	2	-	-	-	-	-	-	-
	Final	1	2	*	1	1	1	1	1	1
E2	Inicial	0	1	0	-	-	-	-	-	-
	Final	1	2	*	*	-	-	-	-	-
E3	Inicial	0	0	-	-	-	-	-	-	-
	Final	1	*	-	-	-	-	-	-	-
E4	Inicial	0	0	-	-	-	-	-	-	-
	Final	1	*	-	-	-	-	-	-	-
E5	Inicial	1	0	-	-	-	-	-	-	-
	Final	1*	1	-	-	-	-	-	-	-
E6	Inicial	1	1	-	-	-	-	-	-	-
	Final	1	1*	-	-	-	-	-	-	-
E7	Inicial	0	0	-	-	-	-	-	-	-
	Final	1	1	-	-	-	-	-	-	-
E8	Inicial	0	0	-	-	-	-	-	-	-
	Final	0*	1	-	-	-	-	-	-	-

* Nuevas prácticas base de este proceso han sido introducidas - Proceso no valorado

Categorías del modelo de referencia: **OPE**: Operación, **DIR**: Alta Dirección, **GER**: Gerencia.

Procesos: **DS**: Desarrollo de Software, **AP**: Administración del Proyecto, **MS**: Mantenimiento de Software, **GN**: Gestión de Negocios, **GP**: Gestión de Procesos, **GCP**: Gestión de Cartera de Proyectos, **GRH**: Gestión de Recursos Humanos, **GC**: Gestión de Conocimiento, **GBSI**: Gestión de Bienes, Servicios e Infraestructura

Tabla 7.14. Capacidad inicial y final de los procesos a mejorar por las empresas

La información relacionada con la capacidad de los procesos se obtuvo después de analizar y sintetizar los datos de los procesos a mejorar con respecto a los atributos de proceso: AP1.1 Realización del proceso, AP2.1 Gestión de la realización y AP2.2 Gestión de productos de trabajo, definidos en el método de evaluación de METvalCOMPETISOFT (conforme con la norma ISO 15504-2).

		Empresa							
		E1	**E2**	**E3**	**E4**	**E5**	**E6**	**E7**	**E8**
Esfuerzo (Horas x 1 persona)	Asesor	40	89	15	41	42	38	65	71
	Empresa	264	255	39	47	27	11	23	16
	Total	304	344	54	88	69	49	88	87
Duración del ciclo (semanas)		24	20	12	12	10	10	10	16

Tabla 7.15. Esfuerzo involucrado en cada ciclo de mejora de cada empresa

La información relacionada con el esfuerzo general involucrado en llevar a cabo el primer ciclo de mejora se obtuvo de sintetizar los datos que se tenían individualmente sobre cada una de las actividades del procedimiento de campo.

7.3.4.6 ANÁLISIS

La Tabla 7.14 muestra que las ocho empresas han incrementado el nivel de capacidad de sus procesos de Desarrollo de Software y/o Administración del Proyecto, entre otros. Este incremento ha permitido a unas empresas alcanzar el siguiente nivel de capacidad de sus procesos bajo intervención, y a otras incrementar el valor de alguno de sus atributos de proceso AP1.1, AP2.1 o AP2.2 (aunque no hayan logrado pasar al siguiente nivel). El incremento de capacidad de cada proceso bajo intervención, se observó al encontrar nuevas prácticas base establecidas en estos procesos, las cuales fueron registradas en los reportes de mejora de cada empresa.

Basados en el análisis de los datos presentados en la Tabla 7.14, hay evidencia que el Marco Metodológico de COMPETISOFT responde favorablemente a la pregunta de investigación PA2, es decir: a través de la aplicación del Marco Metodológico de COMPETISOFT en las pequeñas empresas se han introducido nuevas prácticas de desarrollo de *software* que les permitió incrementar la capacidad de sus procesos.

El esfuerzo presentado en la Tabla 7.15, tiene sentido considerando las características de cada empresa individualmente. De esta tabla se puede extraer el esfuerzo invertido por semana en horas/persona para llevar a cabo el ciclo de mejora en cada empresa. Este esfuerzo incluyendo el tiempo del asesor es para E1 12.7 h, E2 17.2 h, E3 4.5 h, E4 7.3 h, E5 6.9 h, E6 4.9 h, E7 8.8 h y E8 5.4 h. El grupo de empleados involucrados en el ciclo de mejora de cada una de estas empresas fue capaz de asumir este esfuerzo particular relacionado con las iniciativas de mejora sin generar contratiempos en sus actividades diarias. Esta capacidad de asumir adecuadamente el esfuerzo de llevar a cabo la mejora de

procesos es una evidencia de que el Marco Metodológico de COMPETISOFT es apropiado para las pequeñas empresas (*pregunta de investigación* PA1).

Además, las propias empresas han reportado que el trabajo de mejora realizado les ha traído diferentes beneficios, entre los más significativos se encuentran:

- Dar el paso de unos procesos *software* caóticos e impredecibles a unos tangibles, que se usan actualmente para el desarrollo de sus proyectos. Iniciando así el camino de un enfoque de desarrollo de *software* centrado en procesos.
- Comenzar a generar una base de conocimiento relacionada con: (i) los procesos mejorados y sus productos de trabajo asociados, (ii) la forma como se mejoraron, y (iii) la instanciación de éstos en proyectos de la empresa. Esto permite disponer de información histórica que sirve para tomar decisiones.
- Abordar la cultura de mejora de procesos *software* al interior de la organización como instrumento para asegurar la calidad de sus productos. Es así como algunas empresas se certificaron posteriormente bajo la norma ISO 9001 o CMMI nivel 2.

Basado en las experiencias recopiladas de los estudios de casos, el incremento de la capacidad de los procesos, el esfuerzo involucrado para la mejora, y los beneficios descritos por las empresas, podemos considerar que el Marco Metodológico de COMPETISOFT es adecuado para llevar a cabo esfuerzos de mejora de procesos en pequeñas empresas (respondiendo así a la *pregunta de investigación principal*).

Para tratar las amenazas a la validez de los estudios de casos se han considerado diferentes asuntos, los cuales se describen a continuación:

- El diseño del estudio de caso y el plan de colección de datos fueron confrontados con la lista de chequeo para estudios de casos en ingeniería de *software* propuesto por Höst y Runeson (2007). Se observó que el diseño y el plan de colección de datos cumplen los ítems propuestos en un alto porcentaje.
- Para la validez de constructo se ha utilizado múltiples fuentes de evidencia: registro de archivos, entrevistas y observación participativa, obtenidas desde diferentes roles participantes en el ciclo de mejora. Además, se ha mantenido una cadena de evidencia permitiendo la

trazabilidad entre preguntas de investigación, tópicos del protocolo, datos almacenados, evidencias y análisis.

- Con respecto a la validez interna, se puede determinar que la decisión de utilizar en las empresas el Marco Metodológico ha conducido a que éstas incrementen la capacidad de los procesos que han seleccionado mejorar.
- Para la validez externa, inicialmente se aplicó el Marco Metodológico en las empresas E3 y E4 con el apoyo de un único asesor (quien es también parte del grupo de investigadores de procesos/métodos). Siempre se realizó primero las actividades del procedimiento de campo en la empresa E4 y a continuación en la E3. Esto permitió revisar, validar y refinar el protocolo y el procedimiento de campo. Luego se distribuyó el material de replicación del estudio de casos a los diferentes asesores para su conducción en las restantes seis empresas.
- Para la fiabilidad el investigador de procesos/métodos desarrolló el material de replicación del estudio de casos y lo distribuyó entre los asesores. Se observó que siguiendo este material para la conducción de los estudios de casos en las empresas E1, E2, E5, E6, E7 y E8 se llegó a similares hallazgos y conclusiones que los obtenidos en los dos primeros estudios de casos de las empresas E3 y E4.

En cuanto a las limitaciones a considerar en los estudios de casos, podemos destacar:

- El tamaño de la población, que puede limitar el poder de generalización de los resultados obtenidos. Aunque estas empresas son representativas de la industria del *software* de Iberoamérica, el número de empresas participantes es un porcentaje bajo de la población en general.
- Sesgos de los estudios de caso debido: (i) a la manipulación de eventos y datos por parte del asesor, y (ii) al desarrollo no natural porque están siendo observados.

7.4 LECTURAS RECOMENDADAS

- **Creswell**, **J**. **W**. (2009). *Research Design*. *Qualitative*: *Quantitative and Mixed Methods Approaches*, Los Angeles: Sage. En el capítulo 10 de este libro se comenta cómo planificar el uso de métodos de investigación mixtos, combinando métodos cuantitativos con cualitativos.

- **Venkatesh**, **V**., **Brown**, **S**., **Bala**, **H**. (2013). *Bridging the qualitative – quantitative divide*: *guidelines for conducting mixed methods research in information systems*. MIS Quarterly, 37(1), 21-54. Este artículo presenta un conjunto de directrices sobre:

  1) Cuándo conviene utilizar métodos de investigación mixtos combinando métodos cuantitativos y cualitativos,
  2) Cómo integrar los resultados obtenidos usando métodos mixtos y
  3) Cómo validar los resultados obtenidos.

7.5 SITIOS WEB RECOMENDADOS

- *http://alarcos.esi.uclm.es/*

  Se trata del sitio web del grupo Alarcos donde se pueden encontrar decenas de artículos en los que se exponen los métodos de investigación que utilizamos en diferentes áreas de la ingeniería del *software*.

7.6 HERRAMIENTAS RECOMENDADAS

Todas las herramientas de los capítulos anteriores.

ACRÓNIMOS

4GL. *Fourth Generation Language* (*Lenguaje de Cuarta Generación*)

ACM. *Association for Computing Machinery*

ADM. *Arquitecture-Driven Modernization*

AENOR. Asociación Española de Normalización y Certificación

AIS. *Association for Information Systems*

ANECA. Agencia Nacional de Evaluación de la Calidad y Acreditación

ANOVA. *Analysis of Variance*

AP. Administración del Proyecto

APA. *American Psychological Association*

API. *Application Programming Interface*

ASQ. *American Society for Quality*

BP. *Business Processes*

BPMN. *Business Process Modeling Notation*

CAR. *Canonical Action Research*

CDTI. Centro para el Desarrollo Tecnológico e Industrial

CMM. *Capability Maturity Model*

CMMI. *Capability maturity Model Integration*

CPR. *Collaborative Practice Research*

CRM. *Customer Relationship Management*

CTML. *Cognitive Theory of Multimedia Learning*

CYTED. Programa Iberoamericano de Ciencia y Tecnología para el Desarrollo

D. Diseño

DIR. Alta Dirección

DS. Desarrollo de Software

E/R. *Entity Relationship*

EA. *Enterprise Architect*

EBSE. *Evidence-Based Software Engineering*

EC. Estados Compuestos

ENAC. Entidad Nacional de Acreditación

ESE. *Empirical Software Engineering*

EV. Evaluador

GBSI. Gestión de Bienes, Servicios e Infraestructura

GC. Gestión de Conocimiento

GCP. Gestión de Cartera de Proyectos

GCR. Grupo Crítico de Referencia

GER. Gerencia

GGM. Grupo de Gestión de Mejora

GMP. Grupo de Mejora de Procesos

GN. Gestión de Negocios

GP. Gestión de Procesos

GQM. *Goal-Question-Metric*

GRH. Gestión de Recursos Humanos

GUTSE. *Grand Unified Theory of Software Engineering*

I+D. Investigación y desarrollo

I+D+i. Investigación, Desarrollo e innovación

IA. Investigación-Acción

IAS. Intra-sujetos

IES. Inter-sujetos

II. Ingeniería Inversa

IR. Ingeniería de Requisitos

ISERN. *International Software Engineering Research Network*

ISO. Organización Internacional de Estandarización

ITIL. *Information Technology Infrastructure Library*

KDM. *Knowledge Discovery Metamodel*

LOC. Líneas de Código

MaxDIT. Máxima profundidad de herencia

MaxHAgg. Máxima altura de agregación

MDA. *Model-Driven Architecture*

MDD. *Model-Driven Development*

MS. Mantenimiento de Software

NAgg. Número de agregaciones

NAggH. Número de jerarquías de agregación

NAssoc. Número de asociaciones

NDep. Número de dependencias

NGen. Número de generalizaciones

NGenH. Número de jerarquías de generalización

OMG. *Object Management Group*

OO. Orientación a Objetos

OPE. Operación

p. p-valor

PMBOK. *Project Management Body of Knowledge*

PMP. Responsable de mejora de procesos

PMS. Gestión del Proceso de Mantenimiento del Software

po. potencia observada

PYMES. Pequeñas y Medianas Empresas

ROI. Retorno de la Inversión

RP. Responsables del proceso

SEI. *Software Engineering Institute*

SEMAT. *Software Engineering Method and Theory*

SLR. *Systematic Literature Review*

SMS. *Systematic Mapping Study*

SOA. *Service-Oriented Architecture*

SPSS. *Statistical Package for the Social Sciences*

TLR. *Traditional Literature Reviews*

UCLM. Universidad de Castilla-La Mancha

UML. *Unified Modelling language*

V&V. Verificación y validación

BIBLIOGRAFÍA

Al-Zubidy, **A**. y **Carver**, **J**. (2014). *Review of systematic literature review tools*. *Informe Técnico SERG-2014-03*. *University of Alabama*.

Andersson, **C**. y **Runeson**, **P**. (2007). *A spiral process model for case studies on software quality monitoring - method and metrics*. *Software Process*: *Improvement and Practice*: 12(2), 125-140.

Arisholm, **E**., **Sjøberg**, **D**., **Carelius**, **G**. y **Lindsjørn**, **Y**. (2002). *A web-based support environment for software engineering experiments*. *Nordic Journal of Computing*: 9(4), 231-247.

Atkinson, **C**. y **Kühne**, **T**. (2003). *Model-driven development*: *A metamodeling foundation*. *IEEE Software*: 20(5), 36-41.

Avison, **D**., **Lan**, **F**., **Myers**, **M**. y **Nielsen**, **A**. (1999). *Action research*. *Communications of the* ACM, 42(1), 94-97.

Basili, **V**. **R**., **Costa**, **P**. **Lindvall**, **M**., **Mendonca**, **M**., **Seaman**, C., **Tesoriero**, **R**. y **Zelkowitz**, **M**. (2001). *An experience management system for a software engineering research organization*. *Annual NASA Goddard Software Engineering Workshop*: 26-35.

Basili, **V**. y **Rombach**, **D**. (1988). *The* TAME *project*: *towards improvement-oriented software environments*. IEEE *Transactions on Software Engineering*: 14(6), 758-773.

Basili, **V**., **Shull**, **F**. y **Lanubile**, **F**. (1999). *Building knowledge through families of experiments*. IEEE *Transactions on Software Engineering*. 25(4), 456-473.

Baskerville, **R**. (1997). *Distinguishing action research from participative case studies*. *Journal of Systems and Information Technology*: 1(1), 25-45.

Baskerville, **R**. (1999). *Investigating information systems with action research*. *Communications of the Association for Information Systems*: 2(19).

Becker-Kornstaedt, **U**. (2001). *Descriptive software process modeling-how to deal with sensitive process information*. *Empirical Software Engineering*: 6, 353-367.

Beecham, **S**., **Baddoo**, **N**., **Hall**, **T**., **Robinson**, **H**. y **Sharp**, **H**. (2008). *Motivation in software engineering*: *a systematic literature review*. *Information and Software Technology*: 50(9–10), 860–878.

Benbasat, **I**., **Goldstein**, **D**. **K**. y **Mead**, **M**. (1987). *The case research strategy in studies of information systems*. *MIS Quaterly* 11(3), 369-386.

Biostat, **Inc**. (2006). *Meta-Analysis v2*. *http://www.meta-analysis.com*

Bjørnson, **F**. y **Dingsøyr**, **T**. (2008). *Knowledge management in software engineering*: *a systematic review of studied concepts*: *findings and research methods used*. *Information and Software Technology*: 50(11), 1055–1068.

Bodart, **F**., **Patel**, **A**., **Sim**, **M**. y **Weber**, **R**. (2001). *Should optimal properties be used in conceptual modelling*? *A theory and three empirical test*. *Information Systems Research* 12(4), 384-405.

Boehm, **B**. **W**. y **Ross**, **R**. (1988). *Theory-W software project management*: *a case study*. *International Conference on Software Engineering* (ICSE), 30-40.

Borenstein, **M**., **Hedges**, **L**. y **Rothstein**, **H**. (2007). *Meta-Analysis Fixed Effect vs*. *random effec*t. (*www.Meta-Analysis.com*).

Botella, **P**. (2001). La investigación en ingeniería de *software* en nuestro país, ¿va bien?. adenda al libro de actas de las VI Jornadas de Ingeniería del Software y Bases de Datos (Díaz, O., Illarramendi, A., Piattini, M. Eds.).

Bourque, **L**. y **Fielder**, **E**. (1995). *How to Conduct Self-administered and Mail Surveys*, *Thousand Oaks*: *CA*: *Sage Publications*.

Bowes, **D**., **Hall**, **T**. y **Beecham**, **S**. (2012). *SLuRp*: *a tool to help large complex systematic literature reviews deliver valid and rigorous results*. *International Workshop on Evidential Assessment of Software Technologies* (EAST), 33-36.

Braun, **V**. y **Clarke**, **V**. (2006). *Using thematic analysis in psychology*. *Qualitative Research in Psychology*: *3*: 77–101.

Brereton, **P**. **Kitchenham**, **B**., **Budgen**, **D**., **Turner**, **M**. y **Khalil**, **M**. (2007). *Lessons from applying the systematic literature review process within the software engineering domain*. *Journal of Systems and Software*: 80(1), 571–583.

Brereton, **P**., **Kitchenham**, **B**. **A**. y **Budgen**, **D**. (2008). *Using a protocol template for case study planning*. *International Conference on Evaluation and Assessment in Software Engineering* (EASE).

Briand, **L**., **Arisholm**, **S**., **Counsell**, **F**., **Houdek**, **F**. y **Thévenod-Fosse**, **P**. (1999a). *Empirical studies of object-oriented artifacts*: *methods*: *and processes*: *State of the art and future directions*. *Empirical Software Engineering*: 4(4), 387-404.

Briand, **L**., **Morasca**, **S**. y **Basili**, **V**. (1996). *Property-based software engineering measurement*. *IEEE Transactions on Software Engineering*: 22(1), 68-86.

Briand, **L**., **Morasca**, **S**. y **Basili**, **V**. (1999b). *Defining and validating measures for object-based high-level design*. IEEE *Transactions on Software Engineering*: 25, 722-743.

Brooks, **A**., **Daly**, **J**., **Miller**, **J**., **Roper**, **M**. y **Wood**, **M**. (1995) *Replication of Experimental Results in Software Engineering*. Informe Técnico EFoCS-17-95 [RR/95/193], *Dept*. *of Computer Science*: *Univ*. *of Strathclyde*.

Budgen, **D**., **J**. **Burn**, **A**. **J**. y **Kitchenham**, **B**. (2011). *Reporting computing projects through structured abstracts*: *a quasi-experiment*. *Empirical Software Engineering* 16(2), 244-277.

Calero, **C**. (2001). Definición de un conjunto de medidas para la mantenibilidad de bases de datos relacionales, activas y objeto-relacionales. Tesis doctoral. Universidad de Castilla-La Mancha.

Calero, **C**., **Piattini**, **M**. y **Genero**, **M**. (2001). *Method for obtaining correct metrics*. *International Conference on Enterprise and Information Systems* (ICEI*S*): Vol. 2, 779-784.

Campbell, **D**. **T**. y **Cook**, **T**. **D**. (1979). *Quasi-Experimentation*: *Design and Analysis Issues for Field Settings*. *Houghton Mifflin*.

Carrillo de Gea, **J**. **M**, **Nicolás**, **J**., **Fernández Alemán**, **J**. **L**., **Toval**, **A**., **Ebert**, **C**. y **Vizcaíno**, **A**. (2012). *Requirements engineering tools*: *Capabilities*: *survey and assessment*. *Information and Software Technology* 54, 1142-1157.

Carver, **J**. (2010). *Towards reporting guidelines for experimental replications*: *A proposal*. *First International Workshop on replication in Empirical Software Engineering Research* (RESER).

Ciolkowski, **M**. (2009). *What do we know about perspective-based reading*? *An approach for quantitative aggregation in software engineering*. *International Symposium in Empirical Software Engineering and Measurement* (ESEM): 133-144.

Cochrane Collaboration. (2003). *Cochrane Reviewers' Handbook*. Versión 4.2.1.

Conradi, **R**., **Li**, **J**., **Slyngstad**, **O**., **Kampenes**, **V**. y **Bunse**, **C**. (2005). *Reflections on conducting an international survey of software* engineering. *International Symposium on Empirical Software Engineering* (ISESE), 214-223.

Cooper, **H**., **Hedges**, **L**. y **Valentine**, **J**. (2009). T*he handbook of research synthesis and meta-analysis*. *Second edition*, Russell Sage Foundation.

Corbin, **J**. y **Strauss**, **A**. (2008). *Basics of qualitative research*: *techniques and procedures for developing grounded theory*, Sage.

Cronbach, **L**. **J**. (1951). *Coefficient alpha and internal structure of tests*. *Psychometrika*: 16(3), 297–334.

Cruz, **J**. **A**. (2007). *A measurement-based approach for assessing UML statechart diagrams understandability*. Tesis doctoral. Universidad de Castilla-La Mancha.

Cruzes, **D**. y **Dybå**, **T**. (2011a). *Research synthesis in software engineeringA tertiary study*. *Information and Software Technology*, 53(5), 440-455.

Cruzes, **D**. y **Dybå**, **T**. (2011b). *Recommended steps for thematic synthesis in software engineering*. *International Symposium on Empirical Software Engineering and Measurement* (ESEM): 275-284

Cruz-Lemus, **J**. **A**., **Genero**, **M**., **Danilo Caivano**, **Abrahão**, **S**., **Insfrán**, **E**. y **Carsí J**. **A**. (2011). *Assessing the influence of stereotypes on the comprehension of*

UML sequence diagrams: *A family of experiments*. *Information and Software Technology* 53(12), 1391-1403.

Cruz-Lemus, **J**. **A**., **Genero**, **M**., **Manso**, **M**. **E**. y **Piattini**, **M**. (2005). *Evaluating the effect of composite states on the understandability of UML statechart diagrams*. *International Conference on Model-Driven Engineering*: *Languages and Systems* (MoDELS): 113-125.

Cruz-Lemus, **J**. **A**., **Genero**, **M**., **Manso**, **M**. **E**., **Morasca**, **S**. y **Piattini**, **M**. (2009). *Assessing the understandability of UML statechart diagrams with composite states - A family of empirical studies*. *Empirical Software Engineering* 14(6), 685-719.

Curtis, **B**., **Krasner**, **H**. y **Iscoe**, **N**. (1988). *A field study of the software design process for large systems*. *Communications of the* ACM: 31(11), 1268-1287.

Da Silva, **F**., **Santos**, **A**., **Soares**, **S**., **França**, **C**. y **Cleviton**, **V**. (2011). *Six years of systematic literature reviews in software engineering*: *An updated tertiary study*. *Information and Software Technology*: 53(9), 899-913.

Da Silva, **F**., **Suassuna**, **M**., **França**, **C**., **M**. **Grubb**, **A**., **Gouveia**, **T**., **Monteiro**, **C**. y **Dos Santos**, **I**. (2014). *Replication of empirical studies in software engineering research*: *a systematic mapping study*. *Empirical Software Engineering*. 19, 501-557.

Daly, **J**., **Brooks**, **A**., **Miller**, **J**., **Roper**, **M**. y **Wood**, **M**. (1994) .*Verification of results in software maintenance through external replication*. IEEE *International Conference on Software Maintenance* (ICSM): 50–57.

Davison, **R**. **M**., **Martinsons**, **M**. **G**. y **Kock**, **N**. (2004). *Principles of canonical action research*. *Information Systems Journal* 14, 65-86.

Denning, **P**. (2002). *Flatlined*. *The profession of IT*. *Communications of the* ACM: 45(6), 15-19.

Di Bella, **E**., **Fronza**, **I**., **Phaphoom**, **N**., **Sillitti**, **A**., **Succi**, **G**. y **Vlasenko**, **J**. (2013). *Pair Programming and Software Defects-A Large*: *Industrial Case Study*. IEEE *Transactions on Software Engineering*. 39(7), 930-953.

Dieste, **O**. y **Juristo**, **N**. (2011). *Systematic review and aggregation of empirical studies on elicitation techniques*. IEEE *Transaction in Software Engineering*: 37(2), 283-304.

Dieste, **O**., **García**, **R**. y **Fernández**, **E**. (2008). Aggregation process for *software engineering*. *Informe técnico disponible en http://oa.upm.es/4501/*

Dixon-Woods, **M**., Agarwal, S., Jones, D., Young, B. y Sutton, A. (2005). *Synthesising qualitative and quantitative evidence*: *a review of possible methods*. *Journal of Health Services Research and Policy*: 10(1), 45–53.

Dobing, **B**. y **Parsons**, **J**. (2006). *How* UML *is used*?. *Communications of the* ACM, 49(5), 109-113.

Dos Santos, **P**. y **Travassos**, **G**. (2011), *Action research can swing the balance in experimental software engineering*. *Advances in Computers*: 83, 205-276.

Dybå, **T**. (2005). *An empirical investigation of the key factors for success in software process improvement*. IEEE *Transactions on Software Engineering*: 31(5), 410–424.

Dybå, **T**. (2013). Contextualizing empirical evidence. *IEEE Software*: *30*(*1*): 81-83.

Dybå, **T**. y **Dingsøyr**, **T**. (2008a). *Strength of evidence in systematic reviews in software engineering*. *International Symposium on Empirical Software Engineering and Measurement* (ESEM): 78–187.

Dybå, **T**. y **Dingsøyr**, **T**. (2008b). *Empirical studies of agile software development*: *A systematic review*. *Information and Software Technology*: 50(9-10), 833-859.

Dybå, **T**., **Arisholm E**., **Sjøberg D**. **I**. **K**., **Hannay**, **J**. **E**. y **Shull**, **F**. (2007). *Are two heads better than one*? *On the effectiveness of pair programming*. IEEE *Software*: 24(6),10-13.

Dybå, **T**., **Dingsøyr**, **T**. y **Hanssen**, **G**. (2007). *Applying systematic reviews to diverse study types*: *an experience report*. *International Symposium on Empirical Software Engineering and Measurement* (ESEM): 225–234.

Dybå, **T**., **Sjøberg**, **D**. y **Cruzes**, **D**. (2012). *What works for whom*: *where*: *when*: *and why*?: *On the role of context in empirical software engineering*. *International Symposium in Empirical Software Engineering and Measurement* (ESEM): 19-28.

Easterbrook, **S**., **Singer**, **J**., Storey, M. y Damian, D. (2008). *Selecting empirical methods for software engineering research*. Capítulo 11 del libro *Guide to Advanced Empirical Software Engineering*. *Springer* (Forrest, S., Janice, S, Sjøberg, D. Eds.).

El-Emam, **K**., **Benlarbi**, **S**., **Goel**, **N**. y **Rai**, **S**. (2001). *The confounding effect of class size on the validity of object-oriented metrics*, IEEE *Transactions on Software Engineering*: 27(7), 630-650.

Ellis, **P**. (2010). *The essential guide to effect sizes*: *Statistical power*: *meta-analysis*: *and the interpretation of research results*. *Cambridge University P*ress.

Erickson, **J**. y **Siau**, **K**. (2007). *Theoretical and practical complexity of modeling methods*. *Communications of the* ACM, 50(8): 46-51.

Estabrooks, **C**., **Field**, **P**. y **Morse**, **J**. (1994). *Aggregating qualitative findings*: *an approach to theory development*. *Qualitative Health Research*: 4 (4), 503–511.

Estay, **C**. y **Pastor**, **J**. (2000a). *Improving action research in information systems with project management*. *Americas Conference on Information Systems* (ACIS), 1558-1561.

Estay, **C**. y **Pastor**, **J**. (2000b). *Towards a project structure for Action-Research in Information Systems*. *Annual Business and Information Technology Conference* (BIT).

Estay, **C**. y **Pastor**, **J**. (2001). Un modelo de madurez para investigación-acción en sistemas de información. *Jornadas de Ingeniería del Software y Bases de Datos* (JISBD), 265-281.

Estler, **H**., **Nordio**, **M**., **Furia**, **C**., **Meyer**, **B**. y **Schneider**, **J**. (2013). *Agile vs. structured distributed software development*: *A case study*. *Empirical Software Engineering*.

Fenton, **N**. (1994). *Software measurement*: *A necessary scientific basis*. IEEE *Transactions on Software Engineering*: 20(3), 199-206.

Fernández, **A**., **Insfrán**, **E**. y **Abrahão**, **S**. (2011). *Usability evaluation methods for the web*: *A systematic mapping study*. *Information and Software Technology*: 53(8), 789-817.

Fernández-Sáez, **A**., **Chaudron**, **M**. y **Genero**, **M**. (2013). *Exploring costs and benefits of sing UML on maintenance*: *Preliminary findings of a case study in a large IT department*. *International Workshop on Experiences and Empirical Studies in Software Modelling* (EESSMod), 33-42,

Fernández-Sáez, **A**., **Chaudron**, **M**., **Genero**, **M**. y **Ramos**, **I**., (2013). *Are forward designed or reverse-engineered UML diagrams more helpful for code*

maintenance?: *A controlled experiment*. *Evaluation and Assessment I Software Engineering* (EASE), 60-71.

Fernández-Sáez, **A**., **Genero**, **M**. y **Chaudron**, **M**. (2013). *Empirical studies concerning the maintenance of UML diagrams and their use in the maintenance of code*: *A systematic mapping study*. *Information and Software Technology*: 55(7): 1119-1142.

Fernández-Sáez, **A**., **Genero**, **M**. y **Romero**, **F**. (2010). SLR-*Tool* - *A tool for performing systematic literature reviews*. *International Joint Conference on Software Technologies* (ICSOFT), 2, 157-166.

Fernández-Sáez, **A**., **Genero**, **M**., **Chaudron**, **M**., **Caivano**, **D**. y **Ramos**, **I**. *Are forward designed or reverse-engineered UML diagrams more helpful for code maintenance*?: *A family of experiments*. *Information and Software Technology*. (Aceptado pendiente de publicación).

Fowler, **F**. **J**. (2002). *Jr*. *Survey Research Methods*. *Third Edition*. *Thousand Oaks*: *CA*. *Sage Publications*.

França, **C**., **Cunha**, **P**. y **Da Silva**, **F**. (2010). *The effect of reasoning strategies on success in early learning of programming*: *lessons learned from an external experiment replication*. *Evaluation and Assessment in Software Engineering* (EASE).

Franzosi, **R**. (2010). *Quantitative narrative analysis*: *Sage*.

French, **W**. **L**., **Bell**, **C**. **H**. (1996). *Organizational development*: *Behavioral science interventions for organization improvement*. *London*: *Prentice Hall*.

García, **F**. (2004). FMESP: Marco de trabajo integrado para el modelado y la medición de los procesos *software*. Tesis doctoral. Universidad de Castilla-La Mancha.

García, **F**., **Bertoa**, **M**., **Calero**, **C**., **Vallecillo**, **A**., **Ruiz**, **F**., **Piattini**, **M**. y **Genero**, **M**. (2006). *Towards a consistent terminology for software measurement*. *Information and Software Technology* 48(8), 631-644.

Gemino, **A**. y **Wand**, **Y**. (2005). *Are forward designed or reverse-engineered* UML *diagrams more helpful for code maintenance*?: *A controlled experiment*. *Data and Knowledge Engineering 55*: 301-326.

Genero, **M**. (2002). *Defining and validating metrics for conceptual models*. Tesis doctoral. Universidad de Castilla-La Mancha.

Genero, **M**. **Manso**, **M**. **E**. y **Piattini**, **M**. (2004). *Early indicators of* UML *class diagrams understandability and modifiability*. *International Symposium on Empirical Software Engineering* (ISESE), 207-216.

Genero, **M**., **Fernández-Sáez**, **A**., **Nelson**, **H**., **Poels**, **G**. y **Piattini**, **M**. (2011). *Research review*: *a systematic literature review on the quality of* UML *model*s. *Journal of Database Management*, 22(3), 46–70.

Genero, **M**., **Manso**, **M**. **E**., **Visaggio**, **A**., **Canfora**, **G**. y **Piattini**, **M**. (2007). *Building measure-based prediction models for* UML *class diagram maintainability*. *Empirical Software Engineering*: 12(5), 517-549.

Genero, **M**., **Piattini**, **M**. y **Calero**, **C**. (2005). *A survey of metrics for* UML *class diagrams*. *Journal of Object Technology*: 4(9), 59-92.

Glass, **G**.**V**., **McGaw**, **B**. **y Smith**, **M**.**L**. (1981). *Meta-Analysis in Social Research*. *Sage Publications*.

Glass, **R**., **Vessey**, **L**. **y Ramesh**, **V**. (2001). *Research in Software Engineering*: *an empirical study*. Informe Técnico TR105-1. *Information Systems Department*: *Indiana University*.

Gómez, **G**., **Omar**, **S**., **Juristo**, **N**. **y Vegas**, **N**. (2010). *Replication*: *reproduction and re-analysis*: *three ways for verifying experimental findings*. *Proceedings of the 1st International Workshop on Replication in Empirical Software Engineering Research* (RESER), 42–44.

Gómez, **G**., **Omar**, **S**., **Juristo**, **N**. **y Vegas**, **N**. (2010). *Replications types in experimental disciplines*. *Empirical Software Engineering and Measurement* (ESEM), 1–10.

Gorschek, **T**., **Garre**, **P**., **Larsson**, **S**. **y Wohlin**, **C**. (2006). *A model for technology transfer in practice*. IEEE *Software*: 23(6), 88-95.

Gregor, **S**. (2006). *The nature of theory in information systems*. *MIS Quarterly*: 30(3), 491-506.

Grossman, **M**., **Aronson**, **J**.**E**. **y McCarthy**, **R**.**V**. (2005). *Does UML make the grade*? *Insights from the software development community*. *Information and Software Technology* 47(6), 383-397.

Grossman, **M**., **Aronson**, **J**.**E**. **y McCarthy**, **R**.**V**. (2005). *Does UML make the grade*? *Insights from the software development community*. *Information and Software Technology*: 47(6), 383-397.

Gurevitch, **J**. **y Hedges**, **L**. (2001). *Meta-analysis*: *Combining results of independent experiments*. *Design and analysis of ecological experiments* (S.M. Scheiner and J. Gurevitch Eds.), 347–369. *Oxford University Press*: *Oxford*.

Hannay, **J**., **Sjøberg**, **D**. **y Dybå**, **T**. (2007). *Systematic review of theory use in software engineering experiments*. IEEE *Transactions on Software Engineering*: 33(2), 87-107.

Hayes, **W**. (1999). *Research synthesis in software engineering*: *A case for meta-analysis*. IEEE *International Symposium on Software Metrics*, 143-151.

Hedges, **L**.**V**. **y Olkin**, **I**. (1985). *Statistical methods for meta-Analysis*. Academia *Press*.

Higgins, **J**. **y Green**, **S**. (2011). *The Cochrane Collaboration*, *Cochrane handbook for systematic reviews of interventions* (versión 5.1), Informe Técnico.

Höst, **M**. **y Runeson**, **P**. (2007). *Checklists for software engineering case study research*. *International Symposium on Empirical Software Engineering and Measurement* (ESEM): 479-482.

Höst, **M**., **Regnell**, **B**. **y Wohlin**, **C**. (2000). *Using students as subjects- a comparative study of students and professionals in lead-time impact assessment*. *Empirical Software Engineering*: 5(39), 201-214.

Humphrey, **W**. **y Curtis**, **B**. (1991). *Comments on 'a critical look'*. IEEE *Software* 8(4), 42-46.

ISO/IEC (2012). ISO/IEC 19506. *Knowledge Discovery Meta-model* (*KDM*): *v1.1* (*Architecture-Driven Modernization*). *http://www.iso.org/iso/ iso_catalogue/ catalogue_ics/catalogue_detail_ics.htm*?*ics1=35&ics2=080&ics3=&csnumber*=32625, ISO/IEC: 302.

ISO/IEC JTC 1 SC 7, **ISO/IEC TR 24766**. (2009). *Information Technology – Systems and Software Engineering .– Guide for Requirements Engineering tool Capabilities*: ISO, *Geneva*: *Switzerland*: *first ed*.

ISO-IEC (2001). ISO/IEC 9126. *Information technology - Software product quality*.

ISO-IEC (2004). ISO-IEC 15504. *Information technology - Process assessment*.

ISO-IEC (2011). ISO-IEC 29110. *Software engineering - lifecycle profiles for very small entities* (VSEs).

Ivarsson, **M**. **y Gorschek**, **T**. (2009). *Technology transfer decision support in requirements engineering research*: *a systematic review of* REj. *Requirement Engineering*: 14(3), 155–175.

Ivarsson, **M**. **y Gorschek**, **T**. (2011). *A method for evaluating rigor and industrial relevance of technology evaluations*. *Empirical Software Engineering*: 16(3), 365-395.

Jedlitschka, **A**. **y Pfahl**, **D**. (2005). *Reporting guidelines for controlled experiments in software engineering*. *International Symposium on Empirical Software Engineering* (ESEM), 95-104.

Johns, **G**. (2006). The *essential impact of context on organizational behavior*. *Academy of Management Rev*: 31(2), 386–408.

Johnson, **P**., **Ekstedt**, **M**. **y Jacobson**, **I**. (2012). *Where's the Theory for Software Engineering*?. IEEE *Software*: 29(5), 96.

Jørgensen, **M**. (2004). *A review of studies on expert estimation of software development effort*. *Journal of Systems and Software*: 70(1-2), 37-60.

Jørgensen, **M**. **y Moløkken-Østvold**, **K**. (2006). *How large are software cost overruns*? *A review of the* 1994 CHAOS *report*. *Information and Software Technology* 48(4), 297-301.

Jørgensen, **M**. **y Shepperd**, **M**. (2007). *A systematic review of software development cost estimation studies*. IEEE *Transactions on Software Engineering*: 33(1), 33-53.

Juristo, **N**. **y Moreno**, **A**. (2001). *Basics of Software Engineering Experimentation*. Kluwer.

Juristo, **N**., **Vegas**, **S**., **Solari**, **M**., **Abrahão**, **S**. **y Ramos**, **I**. (2013). *A process for managing interaction between experimenters to get useful similar replications*. *Information and Software Technology* 55(2), 215-225.

Kagdi, **H**., **Collard**, **M**. **y Maletic**, **J**. (2007). *A survey and taxonomy of approaches for mining software repositories in the context of software evolution*. *Journal of Software Maintenance*: 19(2), 77-131.

Kampenes, **V**., **Dybå**, **T**., **Hannay**, **J**. **y Sjøberg**, **D**. (2007). *A systematic review of effect size in software engineering experiments*. *Information and Software Technology*: 49(11-12), 1073-1086.

Kampenes, **V**., **Dybå**, **T**., **Hannay**, **J**. **y Sjøberg**, **D**. (2009). *A systematic review of quasi-experiments in software engineering*. *Information and Software Technology*: 51(1), 71-82.

Karahasanovic, **A**., **Anda**., **B**., **Arisholm**, **E**., **Hove**, **S**.**E**., **Jørgensen**, **M**., **Sjøberg**, **D**. **y Welland**, **R**. (2005). *Collecting feedback during software engineering experiments*. *Empirical Software Engineering*: 10(2), 113-147.

Khatri, **V**., **Vessey**, **I**., **Ramesh**, **V**., **Clay**, **P**. **y Park**, **S**. (2006). *Understanding conceptual schemas*: *Exploring the role of application and is domain knowledge*. *Information Systems Research*: *17*(*1*): 81-99.

Kitchenham, **B**. (1996). *DESMET*: *a method for evaluating software engineering methods and tools*. Informe Técnico TR96-09, *Dept*. *of Computer Science*: *University of Keele*.

Kitchenham, **B**. (2008). *The role of replications in empirical software engineering - a word of warning*. *Empirical Software Engineering* 13(2), 219-221.

Kitchenham, **B**. **A**. **y Pfleeger**, **S**. **L**. (2008). *Personal opinion surveys*. Capítulo 3 del Libro *Guide to advanced empirical software engineering*: Shull, F., Singer, J., Sjøberg, D.I.K. (eds.), Springer.

Kitchenham, **B**. **A**., **Pfleeger**, **S**. **L**., **Pickard**, **L**. **M**., **Jones**, **P**. **W**., **Hoaglin**, **D**. **C**., **El Emam**, **K**. **y Rosenberg**, **J**. (2002b). *Preliminary guidelines for empirical research in Software Engineering*. IEEE *Transactions in Software Engineering*: 28(8), 721-734.

Kitchenham, **B**. **y Brereton**, **P**. (2013). *A systematic review of systematic review process research in software engineering*. *Information and Software Technology*: 55(12), 2049-2075.

Kitchenham, **B**. **y Charters**, **S**. (2007), *Guidelines for performing systematic literature reviews in software engineering*, Informe Técnico EBSE-2007-01, *Keele University*.

Kitchenham, **B**., **Brereton**, **P**., **Budgen**, **D**., **Turner**, **M**., **Bailey**, **J**. **y Linkman**, **S**. (2009). *systematic literature reviews in software engineering – a systematic literature review*. *Information and Software Technology*: 51(1), 7–15.

Kitchenham, **B**., **Brereton**, **P**., **Turner**, **M**., **Niazi**, **M**., **Linkman**, **S**., **Pretorius**, **R**. **y Budgen**, **D**. (2010b). *Refining the systematic literature review process - two participant-observer case studies*. *Empirical Software Engineering*: 15(6), 618-653.

Kitchenham, **B**., **Budgen**, **D**. **y Brereton**, **P**. (2011). *Using mapping studies as the basis for further research – A participant-observer case study*. *Information and Software Technology*: 53(6), 638–651.

Kitchenham, **B**., **Mendes**, **E**. **y Travassos**, **G**. (2007). *Cross versus within-company cost estimation studies*: *A systematic review*. IEEE *Transactions on Software Engineering*: 33(5), 316-329.

Kitchenham, **B**., **Pfleeger**, **S**., **Pickard**, **L**., **Jones**, **P**., **Hoaglin**, **D**., **El-Emam**, **K**. **y Rosenberg**, **J**. (2002a). *Preliminary guidelines for empirical research in software engineering*. IEEE *Transactions on Software Engineering*: 28(8), 721-734.

Kitchenham, **B**., **Pretorius**, **R**., **Budgen**, **D**., **Brereton**, **P**., **Turner**, **M**., **Niazi**, **M**., **y Linkman**, **S**. (2010a). *Literature reviews in software engineering – a tertiary study*. *Information and Software Technology*: 52(8), 792–805.

Kitchenham, **B**., **Sjøberg**, **D**., **Dybå**, **T**. **Pfahl**, **D**., **Brereton**, **P**., **Budgen**, **D**., **Höst**, **M**. **y Runeson**, **P**. (2012).*Three empirical studies on the agreement of reviewers about the quality of software engineering experiments*. *Information and Software Technology*: 54(8), 804-819.

Kitchenham, **B**., **Sjøberg**, **D**., **Dybå**, **T**., **Brereton**, **P**., **Budgen**, **D**., **Höst**, **M**. **y Runeson**, **P**. (2013). *Trends in the quality of human-centric software engineering experiments-a quasi-experiment*. IEEE *Transactions on Software Engineering*: 39(7), 1002-1017.

Kitchenham, **B**.**A**., **Pickard**, **L**.**M**. **y Pfleeger**, **S**.**L**. (1995). *Case studies for method and tool evaluation*. IEEE *Software*: 12(4), 52-62.

Klemola, **T**. (2000). *A cognitive model for complexity metrics*. *Workshop on Quantitative Approaches in Object-Oriented Software Engineering* (Celebrado en el ECOOP). *Cannes*: *France*. Springer-Verlag.

Kock, **N**. **y Lau**, **F**. (2001). *Information systems action research*: *Serving two demanding masters*. *Information Technology and People* (*Special issue on Action Research in Information Systems*): 14(1), 6-11.

Kock, **N**., **Gray**, **P**., **Hoving**, **R**., **Klein**, **H**., **Myers**, **M**. **y Rockart**, **J**. (2002). *Is research relevance revisited*: *subtle accomplishment*: *unfilled promise*: *or serial hypocrisy*?. *Communications of the Association for Information Systems*: 8, 330-346.

Krein, **J**. **y Knutson**, **C**. (2010). *A case for replication*: *Synthesizing research methodologies in software engineering*. *Proceedings of the 1st International Workshop on Replication in Empirical Software Engineering Research* (RESER).

Krogstie, **J**. (1998). *Integrating the understanding of quality in requirements specification and conceptual modeling*. ACM SIGSOFT *Software Engineering Notes*: 23(1), 86-91.

Krosnick, **J**.**A**. (1990). *Survey research*. *Annual Review of Psychology*: 50, 537–567.

Laitenberger O., **El-Emam K**. **y Harbich T**. (1999). *An internally replicated quasy-experimental comparison of checklist and perspective-based reading of code documents*. Informe Técnico 006.99/e, IESE.

Lau, **F**. (1997). *A review on the use of action research in information systems studies*. En Lee, A.S., Liebenau, J. y Degross, J.I. *Information systems research*: *Information systems and qualitative research*. Chapman and Hill. London, 31-68.

Lethbridge, **T**. (1998). *A survey of the relevance of computer science and software engineering education*. *International Conference on Software Engineering Education* (ICSE): IEEE *Computer Society Press*.

Lethbridge, **T**. **C**., **Sim**, **S**. **E**. **y Singer**, **J**. (2005). *Studying software engineers*: *data collection techniques for software field studies*. *Empirical Software Engineering* 10, 311-341.

Lewin, **K**. (1946). *Action research and minority problems*. *Journal of Social Issues* 2, 34-46.

Li, **Z**., **Madhavji**, **H**., **Murtaza**, **S**. **S**., **Gittens**, **M**., **Miranskyy**, **A**. **V**., **Godwin**, **D**. **y Cialini**, **E**. (2011). *Characteristics of multiple-component defects and architectural hotspots*: *a large system case study*. *Empirical Software Engineering*: 16(5), 667-702.

Likert, **R**. (1932). *A technique for the measurement of attitudes*. *Archives of Psychology*: 140, 1-55.

Lindland, **O**. **I**., **Sindre**, **G**. **y Sølvberg**, **A**. (1994). *Understanding quality in conceptual modeling*. IEEE *Software*: 11(2), 42-49.

Lindsay, **R**. **M**. **y Ehrenberg**, **A**. (1993). *The design of replicated studies*. *Am Stat*: 47(3), 217–228.

Lindvall, **M**. **y Rus**, **I**. (2003). *Lessons learned from building experience factories for software organizations*. *Wissensmanagement*: 59-63.

Litwin, **M**. (1995). *How to measure survey reliability and validity*?. *Thousand Oaks*, CA. *Sage Publications*.

Lucas, **F**. **J**., **Molina**, **F**. **y Toval**, **A**. (2009). *A systematic review of* UML *model consistency management*. *Information and Software Technology*, 51(12), 1631-1645.

Lung, **J**. **Aranda**, **J**., **Easterbrook**, **S**. **y Wilson**, **G**. (2008). *On the difficulty of replicating human subjects studies in software engineering*. *International Conference on Software Engineering* (ICSE), 191–201.

MacDonell, **S**., **Shepperd**, **M**., **Kitchenham**, **B**. **y Mendes**, **E**. (2010). *How reliable are systematic reviews in empirical software engineering*?. IEEE *Transactions on Software Engineering*: *36*(*5*), 676-687.

Manso, **M**., **Cruz-Lemus**, **J**., **Genero**, **M**. **y Piattini**, **M**. (2009). *Empirical validation of measures for UML class diagrams*: *A meta-analysis study*. *Models in Software Engineering*: LNCS 5421, 303-313.

Marshall, **C**. **y Brereton**, **P**. (2013). *Tools to support systematic literature reviews in Software Engineering*: *A mapping study*. *International Conference on Empirical Software Engineering and Measurement* (ESEM).

Martínez, **A**. (2001). Medidas para asegurar la mantenibilidad de entornos de cuarta generación. Tesis doctoral. Universidad de Castilla-La Mancha.

Maxwell, **K**. (2002). *Applied Statistics for Software Managers*. *Software Quality Institute Series*. *Prentice Hall*.

Mayer, **R**. **E**. (2001). *Multimedia Learning*. *Cambridge University Press*.

McLeod, **L**., **MacDonell**, **S**. **y Doolin**, **B**. (2011). *Qualitative research on software development*: *a longitudinal case study methodology*. *Empirical Software Engineering* 16(4), 430-459.

McNiff, **J**. (1988). *Action research*. *Principles and practice*. London (UK): Macmillan.

McTaggart, **R**. (1991). *Principles of participatory action research*. *Adult Education Quarterly*: *41*(*3*).

Miles, **M**. **y Huberman**, **A**. (1994). *Qualitative data analysis*: *An expanded source book*, *Sage*.

Miller, **J**. (1999). *Can results from software engineering experiments be safely combined*? IEEE METRICS, 152-158

Mohagheghi, **P**., **Dehlen**, **V**. **y Neple**, **T**. (2009). *Definitions and approaches to model quality in model-based software development - A review of literature*. *Information and Software Technology*, 51(12), 1646-1669.

Montgomery, **D**. (2000). *Design and analysis of experiments*. *Fifth edition*. *Wiley*.

Moody, **D**. (2000). *Building links between* IS *research and professional practice*: *improving the relevance and impact of IS research*. *International Conference on Information Systems* (ICIS), 351-360.

Moody, **D**. (2005). *Theoretical and practical issues in evaluating the quality of conceptual models*: *current state and future directions*. *Data and Knowledge Engineering*, 55(3), 243-276.

Mora, **B**. (2011). SMF: Marco de Trabajo basado en MDE para la medición genérica del *software*. Tesis Doctoral. Universidad de Castilla-La Mancha.

Moses, **J**. (2000). *Bayesian probability distributions for assessing measurement of subjective software attributes*. *Information and Software Technology*: 42(8), 533–546.

Myers, **M**. (1999). *Investigating information systems with ethnographic research*. *Communications of the Association for Information Systems* (2), *Article 3*.

Nelson, **H**., **Monarchi**, **D**. **y Nelson**, **K**. (2001). *Ensuring the "goodness" of a conceptual representation*. *European Conference on Software Measurement and* ICT *Control* (FESMA).

Noblit, **G**. **y Hare**, **R**. (1988). *Meta-ethnography*: *synthesizing qualitative studies*, *Sage*.

Novillo, **F**., **García**, **F**., **Rolòn**, **E**., **Ruiz**, **F**. **y Piattini**, **M**. (2008). *Empirical* WEBGEN; *A web-based environment for the automatic generation of surveys and experiments*. *International Conference on Evaluation and Assessment in Software Engineering* (EASE).

Oktaba, **H**., **García**, **F**., **Piattini**, **M**., **Pino**, **F**., **Alquicira**, **C**. **y F**. **Ruiz**. (2007). *Software process improvement*: *The* COMPETISOFT *Projec*t. IEEE *Computer*: 40(10), 21-28.

Oktaba, **H**., **Piattini**, **M**., **Pino**, **F**. **J**., **Orozco**, **M**. **J**. **y Alquicira**, **C**. (2008). COMPETISOFT: *Mejora de procesos software para pequeñas y medianas empresas y Proyectos*. RA-MA.

OMG (1997). *Object Management Group - UML*. Recuperado en *http://www.uml.org/*

OMG (2003). *MDA Guide* (*Vol*. *Versión 1.0.1*.). Recuperado en *http://www.omg.org/docs/omg/03-06-01.pdf*

Otero, **M**. **C**. **y Dolado**, **J**. **J**. (2000). Diseño experimental en ingeniería del *software*, Capítulo 3 del libro *Medición para la Gestión en la Ingeniería del Software*. RAMA, 51-72.

Ott, **L y Longnecker**, **M**. (2010). *An introduction to statistical methods and data analysis*. *Cengage Learning*: *Sixth edition*: Brooks/Cole, 56–139

Owen, **S**., **Brereton**, **P**. **y Budgen**, **D**. (2006). *Protocol analysis*: *a neglected practice*. *Communications of the ACM 49*(*2*): 117-122.

Padak, **N**. **y Padak**, **G**. (1994). *Guidelines for planning action research projects*. de *http://archon.educ.kent.edu/Oasis/Pubs/0200-08.html*

Pareto, **L**., **Genero**, **M**. **y Chaudron**, **M**. (2012). Tutorial T5: *Empirical methods in Software Engineer*ing. Celebrado dentro del congreso *International Conference on Model Driven Engineering Languages and Systems* (*MODELS*).

Parnas, **D**. **L**. (1998). *Software engineering programs are not Computer Science programs*. *Annals of Software Engineering*, 6, 19-37.

Passos, **C**., **Cruzes**, **D**. **S**., **Dybå**, **T**. **y Mendonça**, **M**. **G**. (2012). *Challenges of applying ethnography to study software practices*. *International Symposium on Empirical Software Engineering and Mesasurement* (ESEM): 9-18.

Paterson, **B**., **Thorne**, **S**., **Canam**, **C**. **y Jillings**, **C**. (2001). *Meta-study of qualitative health research*: *A practical guide to meta-analysis and meta-synthesis*, Sage.

Pawson, **R**., **Greenhalgh**, **T**., **Harvey**, **G**., **y Walshe**, **K**. (2005). *Realist review – a new method of systematic review designed for complex policy interventions*. *Journal of Health Services Research and Policy*: 10(1), 21–34.

Pérez-Castillo, **R**. (2012). MARBLE: *Modernization approach for recovering business processes from legacy information systems*. Tesis doctoral. Universidad de Castilla-La Mancha.

Pérez-Castillo, **R**., **Cruz-Lemus**, **J**. **A**., **García-Rodríguez de Guzmán**, **I**. **y Piattini**, **M**. (2012). *A family of case studies on business process mining using* MARBLE. *Journal of Systems and Software* 86(6), 1370–1385.

Pérez-Castillo, **R**., **Fernández-Ropero**, **M**., **García Rodríguez de Guzmán**, **I**. **y Piattini**, **M**. (2011a). MARBLE. *A business process archeology tool*. IEEE *International Conference on Software Maintenance* (ICSM), 578-581.

Pérez-Castillo, **R**., **García-Rodríguez de Guzmán**, **I**. **y Piattini**, **M**. (2011b). *Business process archeology using* MARBLE. *Information and Software Technology*: 53, 1023–1044.

Pérez-Castillo, **R**., **Weber**, **B**., **García Rodríguez de Guzmán**, **I**. **y Piattini**, **M**. (2011c). *Generating event logs from non-process-aware systems enabling business process mining*. *Enterprise Information System Journal*: 5(3), 301–335.

Pérez-Castillo, **R**., **Weber**, **B**., **García Rodríguez de Guzmán**, **I**. **y Piattini**, **M**. (2010). *Toward obtaining event logs from legacy code*. *business process managemen*t, *Workshops* (*BPI'*) *Lecture Notes in Business Information Processing* 66 – (*Part* 2), 201–207.

Pfleeger, **S**. **L**. **y Kitchenham**, **B**. **A**. (2001). *Principles of Survey Research*. *Part 1*: *Turning lemons into lemonade*. *Software Engineering Notes*: 26(6), 16-18.

Piattini, **M**., **Genero**, **M**., **Poels**, **G**. **y Nelson**, **J**. (2005). *Towards a framework for conceptual modelling quality*. Capítulo 1 del libro *Metrics for software conceptual models*. I. C. *Press*.

Piattini, **M**., **Ruiz**, **F**., **Polo**, **M**., **Bastanchury**, **T**., **Fernández**, **I**., **Martínez**, **M**. **A**. **y Villalba**, **J**. (1998). Mantenimiento del *software*: conceptos, métodos: herramientas y *outsourcing*. RA-MA.

Piattini, **M**., **Villalba**, **J**., **Ruiz**, **F**., **Bastanchury**, **T**., **Polo**, **M**., **Martínez**, **M**. **A**. **y Nistal**, **C**. (2000). Mantenimiento del *software*: *Modelos*: técnicas y métodos para la gestión del cambio. RA-MA.

Pickard, **L**., **Kitchenham**, **B**. **y Jones**, **B**. (1998). *Combining empirical results in Software Engineering*. *Information and Software Technology*: 40(14), 811-821.

Pino, **F**. (2010). *Integrated framework for software process improvement in small organizations*. Tesis doctoral. Universidad de Castilla-La Mancha.

Pino, **F**. **J**., **Ruiz**, **F**., **García**, **F**. **y Piattini**, **M**. (2011). *A software maintenance methodology for small organizations*: Agile_MANTEMA. *Journal of Software*: *Evolution and Process*: *Journal of Software*: *Evolution and Process*, 24(8), 851–876.

Pino, **F**., **García**, **F**. **y Piattini**, **M**. (2008). *Software process improvement in small and medium software enterprises*: *a systematic review*. *Software Quality Journal*, 16(2), 237-261.

Pino, **F**., **Vidal**, **J**., **García**, **F**. **y Piattini**, **M**. (2007). Modelo para la implementación de mejora de procesos en pequeñas organizaciones *software*. Jornadas de Ingeniería del Software y Bases de Datos (JISBD), 326-33.

Poels, **G**. (1999). *On the formal aspects of the measurement of object-oriented software specifications*. Tesis doctoral. *Faculty of Economics and Business Administration*. *Katholieke Universiteit Leuven*, Bélgica.

Poels, **G**. **y Dedene**, **G**. (2000a). *Distance-based software measurement*: *necessary and sufficient properties for software measures*. *Information and Software Technology*: 42(1), 35-46.

Poels, **G**. **y Dedene**, **G**. (2000b). *Measures for assessing dynamic complexity aspects of object-oriented conceptual schemes*. *International Conference on Conceptual Modelling* (ER), 499-512.

Polo, **M**, **Piattini**, **M**., **Ruiz**, **F**. **y Calero**, **C**. (1999). MANTEMA: *A complete rigorous methodology for supporting maintenance based on the* ISO/IEC 12207 *Standard*. *European Conference on Software Maintenance* (CSMR),178-181.

Polo, **M**. (2000). MANTEMA. Una metodología para el mantenimiento del *software*. Tesis Doctoral. Universidad de Castilla-La Mancha.

Polo, **M**., **Piattini**, **M**. **y Ruiz**, **F**. (2001). MANTOOL: *A tool for supporting the software maintenance process*. *Journal of Software Maintenance and Evolution*: *Research and Practice*: 13(2), 77-95.

Polo, **M**., **Piattini**, **M**. **y Ruiz**, **F**. (2002a). *Using a qualitative research method for building a software maintenance methodology*. *Software Practice and Experience*: 32(13), 1239-1260.

Polo, **M**., **Piattini**, **M**. **y Ruiz**, **F**. (2002b). *Integrating outsourcing in the maintenance process*. *Information Technology and Management*, 3(3), 247-269.

Popper K. (1959). *The Logic of Scientific Discovery*. Hutchinson & Co. 513.

Porter, **A**. **A**., **y Johnson**, **P**. **M**. (1997). *Assessing software review meetings*: *Results of a comparative analysis of two experimental studies*. IEEE *Transactions on Software En-gineering*: 23(3), 129-145.

Pretorius, **R**. **y Budgen**, **D**. (2008). *A mapping study on empirical evidence related to the models and forms used in the UML*. *International Symposium on Empirical Software Engineering and Measurement* (ESEM), 342-344.

Ragin, **C**. (1987). *The comparative method*: *Moving beyond qualitative and quantitative strategies*, *University of California Press*.

Rainer, **A**. (2011).*The longitudinal*: *chronological case study research strategy*: *A definition and an example from IBM Hursley Park*. *Information Software Technology*: 53(7), 730-746.

Recker, **J**., **Rosemann**, **M**. **y Krogstie**, **J**. (2007). *Ontology- versus pattern-based evaluation of process modeling languages*: *A comparison*. *Communications of the Association for Information Systems*: 20(48), 774-799.

Reynoso, **L**. (2007). *A measurement-based approach for assessing the influence of import-coupling on the maintainability of* OCL *expressions*. Tesis doctoral. Universidad de Castilla-La Mancha.

Reynoso, **L**., **Genero**, **M**. **y Piattini**, **M**. (2010). *Refinement and extension of* SMDM, *a method for defining valid measures*. *Journal of Universal Computer Science*: 16(21), 3210-3244.

Robinson, **H**., **Segal**, **J**. **y Sharp**, **H**. (2007). *Ethnographically-informed empirical studies of software practice*. *Information and Software Technology* 49(6), 540-551.

Robson, **C**. (2002). *Real World Research*: *a resource for social scientists and practitioners-researchers*: *Second edition*. *Blackwell*.

Rodgers, **M**., **Sowden**, **A**., **Petticrew**, **M**., **Arai**, **L**., **Roberts**, **H**., **Britten**, **N**. **y Popay**, **J**. (2009). *Testing methodological guidance on the conduct of narrative synthesis in systematic reviews*. *Evaluation*: 15(1), 49–74.

Rodríguez, **P**., **Markkula**, **J**. **Olivo**, **M**. **y Turula**, **K**. (2012). *Survey on agile and lean usage in Finnish software industry*. *International Symposium on Emprical Software Engineering and Measurement* (ESEM), 139-148.

Ropponen, **J**. **y Lyytinen**, **K**. (2002). *Components of software development risk*: *how to address them*. *A project manager survey*. IEEE *Transactions on Software Engineering*: 26(2), 98-112.

Ruiz, **F**. (2003). MANTIS: Definición de un entorno para la gestión del mantenimiento del *software*. Tesis Doctoral. Universidad de Castilla-La Mancha.

Ruiz, **F**., **García**, **F**., **Piattini**, **M**. **y Polo**, **M**. (2002b). *Environment for managing software maintenance projects*. Capítulo 10 del libro *Advances in Software Maintenance Management*: *Technologies and Solutions*. Idea Group Publishing, 255-290.

Ruiz, **F**., **Piattini**, **M**., **García**, **F**. **y Polo**, **M**. (2002a). *An* XMI-*Based repository for software process meta-modeling*. *International Conference of Product Focused Software Development and Process Improvement* (PROFES), 546-558.

Ruiz, **F**., **Vizcaíno**, **A**., **Piattini**, **M**. **y García**, **F**. (2003). *An ontology for the management of software maintenance projects*. *International Journal of Software Engineering and Knowledge Engineering*: 14(3), 323-349.

Runeson, **P**. (2012). *It takes two to tango - an experience report on industry - Academia collaboration*. *International Conference on Software Testing* (ICST), 872-877.

Runeson, **P**. **y Höst**, **M**. (2009). *Guidelines for conducting and reporting case study research in software engineering*. *Empirical Software Engineering*: 14(2), 131-164.

Runeson, **P**., **Host**, **M**., **Rainer**, **A**. **y Regnell**, **B**. (2012). *Case study research in Software Engineering*: *Guidelines and examples*. *Wiley*.

Sackett, **D**., **Straus**, **S**., **Richardson**, **W**., **Rosenberg**, **W**. **y Haynes**, **R**. (2000). *Evidence-based medicine*: *How to practice and teach* EBM, *Churchill Livingstone*: *Edinburgh*.

Sandberg, **A**., **Pareto**, **L**. **y Arts**, **T**. (2011). *Agile collaborative research*: *Action principles for industry-academia collaboration*. IEE*E Software*, 28(4), 74-83.

Sandelowski, **M**. **y Barroso**, **J**. (2007). *Handbook for synthesizing qualitative research*, *Springer*.

Scanniello, **G**. **y Salviulo**, **F**. (2014). *Dealing with Identfiers and Comments in Source Code Comprehension and Maintenance*: *Results from an Ethnographically-informed Study with Students and Professionals*. *International Conference on Evaluation and Assessment in Software Engineering* (EASE).

Scanniello, **G**., **Gravino**, **C**., **Genero**, **M**., **Cruz-Lemus**, **J**. **y Scaniello**, **G**. (2014). *On the impact of* UML *analysis models on source code comprehensibility and modifiability*. ACM *Transactions on Software Engineering and Methodology*: 23(2).

Seaman, **C**. **B**. (1999). *Qualitative methods in empirical studies of Software Engineering*. IEEE *Transactions in Software Engineering*: 25(4), 557-572.

Serrano, **M**. (2004). Definición de un conjunto de medidas para asegurar la calidad de los almacenes de datos. Tesis doctoral. Universidad de Castilla-La Mancha.

Serrano, **M**., **Piattini**, **M**., **Calero**, **C**., **Genero**, **M**. **y Miranda**, **D**. (2002). Un método para la definición de medidas de *software*. *Métodos de Investigación y Fundamentos Filosóficos en Ingeniería del Software y Sistemas de Información* (MIFISIS). Taller celebrado dentro de JISBD.

Shaddish, **W**.**R**., **Cook**, **T**.**D**. **y Campbell**, **D**.**T**. (2002). *Experimental and quasi-experimental designs for generalized causal inference*, *Houghton Mifflin Company*: *New York*.

Sharp, **H**. (2004). *An ethnographic study of XP practice*. *Empirical Software Engineering* 9(4), 353-375.

Sharp, **H**. (2012). *Using ethnography in empirical software engineering*. *Keynote at the International Conference on Evaluation and Assessment in Software Engineering* (EASE).

Shull, **F**., **Basili**, **V**., **Carver**, **J**., **Maldonado**, **J**. **C**., **Travassos**, **G**. **H**., **Mendonça**, **M**. **y Fabbri**, **S**. (2002). *Replicating software engineering experiments*: *addressing the tacit knowledge problem*. *International Symposium on Empirical Software Engineering* (ISESE), 7-16.

Shull, **F**., **Carver**, **J**., **Vegas**, **S**. **y Juristo**, **N**. (2008). *The role of replications*. *Empirical Software Engineering*: 13(2), 211-218.

Sieber, **J**. (2001). *Protecting research subjects*: *employees and researchers*: *Implications for Software Engineering*. *Empirical Software Engineering*: 6, 329-341.

Siegel, **S**. **y Castellan**, **N**. **J**. (1998). *Nonparametric statistics for the behavioral science*, *Second edition*: *McGraw-Hill Book Company*: *New York*.

Singer, **J**. **y Vinson**, **N**. (2001). *Why and how research ethics matters to you*. *Yes*: *you*!. *Empirical Software Engineering*: 6, 287-290.

Sjøberg, **D**., **Dybå**, **T**., **Anda**, **B**. **y Hannay**, **J**. (2008). *Building theories in software engineering* . Capítulo 12 del libro *Guide to Advanced Empirical Software Engineering*. *Springer* (*Forrest*: *S*.: *Janice*: *S*.: *Sjøberg*: *D*. *Eds*.).

Sjøberg, **D**., **Hannay**, **J**., **Hansen**, **O**., **Kampenes**, **V**., **Karahasanovic**, **A**., **Liborg**, **N**. **y Rekdal**, **A**. (2005). *A survey of controlled experiments in Software Engineering*. IEEE *Transacions on Software Engineering*: 31(9), 733-753.

Source Tap (2009). *Source Tap CRM*. *http://sourcetapcrm.sourceforge.net/*.

SPSS (2003). *SPSS 12.0*: *Syntax reference guide*. Chicago, USA: SPSS Inc.

Staples, **M**. **y Niazi**, **M**. (2007). *Experiences using systematic review guidelines*. *The Journal of Systems and Software*: 80(9), 1425–1437

Staples, **M**. **y Niazi**, **M**. (2008). *Systematic review*: *systematic review of organizational motivations for adopting* CMM-based SPI. *Information and Software Technology*: 50(7–8), 605–620.

Tichy, **W**. (1998). *Should computer scientists experiment more*? IEEE *Computer*: 31(5), 32-40.

Torii, **K**., **Nakakoji**, **K**., **Takada**, **Y**., **Takada**, **S**. **y Shima**, **K**. (1999). Ginger2: *An environment for computer-aided empirical software engineering*. IEEE *Transactions on Software Engineering*: (25)4, 474–492.

Travassos, **G**. **H**., **Dos Santos**, **P**. **S**. **M**., **Mian**, **P**., **Neto**, **P**. **G**. **M**. **y Biolchini**, **J**. (2008). *An environment to support large scale experimentation in software engineering*. IEEE *International Conference on Engineering of Complex Computer Systems* (ICECCS), 193-202.

Turner, **M**., **Kitchenham**, **B**., **Brereton**, **P**., **Charters**, **S**., **y Budgen**, **D**. (2010). *Does the technology acceptance model predict actual use*? *A systematic literature review*. *Information and Software Technology*: 52(5), 463–479.

Ullman, **J**. (2001). *Jeffrey d*. *Ullman speaks out on the future of higher education startups*: *database theory*: *and more*. SIGMOD *Record*: 30 (3), 86-94.

Vegas, **S**., **Juristo**, **N**., **Moreno**, **A**., **Solari**, **M**. **y Letelier**, **P**. (2006). *Analysis of the influence of communication between researchers on experiment replication*. *International Symposium on Empirical Software Engineering* (ISESE), 28-37.

Verner, **J**. **M**., **Brereton**, **O**. **P**., **Kitchenham**, **B**. **A**., **Turner**, **M**., **y Niazi**, **M**. (2014). *Risks and risk mitigation in global software development*: *A tertiary study*. *Information and Software Technology*: 56(1), 54-78.

Verner, **J**. **M**., **Sampson**, **J**., **Tosic**, **V**., **Bakar**, **N**. **A**. **A**. **y Kitchenham**, **B**. **A**. (2009). *Guidelines for industrially-based multiple case studies in software engineering*. *International Conference on Research Challenges in Information Science* (RCIS), 313-324.

Vizcaíno, **A**., **García**, **F**., **Villar**, **J**. **C**., **Piattini**, **M**. **y Portillo**, **J**. (2013). *Applying Q-methodology to analyse the success factors in* GSD. *Information and Software Technology*: 55(7), 1200-1211.

Wadsworth, **Y**. (1998). *What is participatory Action Research*? *Action research international*: *paper 2*.

Watts, **P**. **y Stenner**, **P**. (2005). *Doing Q methodology*: *Theory*: *method and interpretation*: Vol. 2.

Weyuker, **E**. (1988). *Evaluating software complexity measures*. IEEE *Transactions Software Engineering*: 14(9), 1357-1365.

Whitmire, **S**. (1997). *Object oriented design measurement*. John Wiley & Sons, Inc.

Wieringa, **R**. (2013). *Empirical research methods for technology validation*: *Scaling up to practice*. *The Journal of Systems and Software*. (*Aceptado pendiente de publicación*).

Wieringa, **R**. **y Morah**, **A**. (2012). *Technical action research as a validation method in information systems design science*. *International Conference on Design Science Research in Information Systems and Technology* (DESRIST), LNCS 7286, 220-238.

Wohlin, **C**. (2014). *Guidelines for snowballing in systematic literature studies and a replication in Software Engieenring*. *International Conference on Evaluation and Assessment in Software Engineering* (EASE), 321-330.

Wohlin, **C**., **Aurum**, **A**., **Angelis**, **L**., **Phillips**, **L**., **Dittrich**, **Y**., **Gorschek**, **T**., **Grahn**, **H**., **Henningsson**, **K**., **Kågström**, **S**., **Low**, **G**., **Rovegard**, **P**., **Tomaszewski**, **P**., **Van Toorn**, **C**. y **Winter**, **J**. (2012b). *The success factors powering industry-academia collaboration*. IEEE *Software*: 29(2), 67-73.

Wohlin, **C**., **Runeson**, **P**., **Höst**, **M**., **Ohlsson**, **M**. **C**., **Regnell**, **B**. y **Wesslén**, **A**. (2012a). *Experimentation in Software Engineering*. Springer.

Wong, **W**., **Tse**, **T**., **Glass**, **R**., **Basili**, **V**., y **Chen**, **T**. (2011). *An assessment of systems and software engineering scholars and institutions* (2003-2007 and 2004-2008). *Journal of Systems and Software*: 84(1), 162-168.

Wood-Harper, **A**. **T**. (1985). *Research methods in information systems*: *Using action research*. Capítulo del libro *Research Methods in Information Systems*, (*Mumford*: *E*. *Hirschheim*: *R.A*.: *Fitzgerald*: *G*. *y Wood-Harper*: *A.T*. *Eds*.): *North-Holland*: *Amsterdam*.

Yin, **R**. **K**. (2014). *Case study research*. *Fifth edition*. *SAGE*.

Yin, **R**. y **Heald**, **K**. (1975). *Using the case survey method to analyze policy studies*. *Administrative Science Quarterly*: 20, 371–381.

Zelkowitz, **M**. y **Wallace**, **D**. (1998). *Experimental models for validating technology*. *Computer*: 31, 23-31.

Zhang, **H**. y **Ali Babar**, **M**. (2013). *Systematic reviews in software engineering*: *An empirical investigation*. *Information and Software Technology*: 55(7), 1341–1354.

Zhang, **H**., **Ali Babar**, **M**. y **Tell**, **P**. (2011). *Identifying relevant studies in software engineering*. *Information and Software Technology*: 53(6), 625–637.

Zuse, **H**. (1998). *A framework of software measurement*. *Berlin*. *Walter de Gruyter*.

Métodos de investigación en ingeniería del software

En estos últimos años, dentro de la Ingeniería del Software Empírica, los investigadores han desarrollado una serie de guías y técnicas que permiten llevar a cabo la investigación de manera rigurosa. Por otra parte, las organizaciones y los profesionales han empezado a darse cuenta de la necesidad de contrastar experimentalmente muchas de las creencias y nuevas técnicas en el área de la ingeniería del software, concediendo cada vez más importancia a la ingeniería del software basada en evidencias (EBSE, *Evidence-Based Software Engineering*) y a la ingeniería del software empírica (ESE, *Empirical Software Engineering*).

En el grupo Alarcos, desde nuestra creación, nos hemos esforzado por adoptar esa rigurosa visión experimental, fruto de la cual proponemos esta obra; cuyo objetivo principal es presentar de forma clara y precisa los métodos de investigación aplicables en ingeniería del software, mostrando ejemplos concretos de su aplicación, dando a conocer los principales problemas en su utilización, y los recursos que puedan ayudar a una utilización más efectiva de estos métodos. En el libro se abordan tanto las principales técnicas de investigación primaria (encuestas, experimentos, estudios de caso e investigación-acción) como las revisiones sistemáticas de la literatura y la combinación de métodos.

Todo ello esperamos que contribuya a incrementar la rigurosidad de la investigación que se lleva a cabo en ingeniería del software y permita potenciar la transferencia de tecnología en este campo, al proporcionar a las organizaciones y empresas evidencias sobre las mejoras y ventajas que pueden ofrecer estas.

https://yolibrospdf.com/programacion.html

9 788499 645070

ra-ma.es

Ra-Ma®