Prof. Edgar Serna M.

Gestión de la Ingeniería de Requisitos Integrando Principios del Pensamiento Complejo

Editorial Instituto Antioqueño de Investigación
Medellín - Antioquia

Prof. Edgar Serna M.

Gestión de la Ingeniería de Requisitos Integrando Principios del Pensamiento Complejo

ISBN: 978-958-52333-9-3

© 2021 Editorial Instituto Antioqueño de Investigación

Medellín, Antioquia

Catalogación en la publicación – Biblioteca Nacional de Colombia

Serna M., Edgar
Gestión de la ingeniería de requisitos integrando principios del pensamiento complejo [recurso electrónico] / Edgar Serna M. -- 1a. ed. -- Medellín: Instituto Antioqueño de Investigación, 2021.
Archivo en formato digital (pdf). -- (Ingeniería y ciencia)

Incluye referencias bibliográficas.

ISBN 978-958-52333-9-3

1. Ingeniería del software 2. Ingeniería de requisitos I. Título II. Serie

CDD: 005.1 ed. 23 CO-BoBN– a1067756

Investigación Científica
ISBN: 978-958-52333-9-3
DOI: 10.5281/zenodo.4445052
Hecho el Depósito Legal Digital

Gestión de la Ingeniería de Requisitos Integrando Principios del Pensamiento Complejo
Serie: Ingeniería y Ciencia
Editorial Instituto Antioqueño de Investigación

Edición 1: enero 2021
Publicación electrónica gratuita

Copyright © 2021 Instituto Antioqueño de Investigación IAI™. Salvo que se indique lo contrario, el contenido de esta publicación está autorizado bajo Creative Commons Licence CC BY-NC-SA 4.0.

Maquetación: Instituto Antioqueño de Investigación IAI
Diseño: IAI, Medellín, Antioquia.
Editorial Instituto Antioqueño de Investigación es Marca Registrada del *Instituto Antioqueño de Investigación*. El resto de marcas mencionadas en el texto pertenecen a sus respectivos propietarios.

La información, hallazgos, puntos de vista y opiniones contenidos en esta publicación son responsabilidad de los autores y no reflejan necesariamente los puntos de vista del Instituto Antioqueño de Investigación IAI; no se garantiza la exactitud de la información proporcionada en este texto.

Ni el autor, ni la Editorial, ni el IAI serán responsables de los daños causados o presuntamente causados directa o indirectamente por el contenido en este libro.

Diseño, edición y publicación
Editorial Instituto Antioqueño de Investigación
http://fundacioniai.org/index.php/editorial.html

Instituto Antioqueño de Investigación IAI
http://fundacioniai.org
contacto@fundacioniai.org

© 2021 Editorial Instituto Antioqueño de Investigación

CONTENIDO

	PRESENTACIÓN	2
I.	INTRODUCCIÓN	3
II.	MARCO REFERENCIAL DE CONCEPTOS SOCIO-INGENIERILES NECESARIOS PARA EL ANÁLISIS	12
1.	Pensamiento Complejo	12
2.	Requisitos	13
3.	Ingeniería de Requisitos	15
3.1	Gestión de la Ingeniería de Requisitos	17
4.	Sistema	19
5.	Complejidad	20
5.1	Transdisciplinariedad	22
5.2	Multidimensionalidad	25
6.	Gestión del Conocimiento	26
7.	Fiabilidad del Software	27
III.	MARCO TEÓRICO COMO RECURSO DE COMPRENSIÓN	30
1.	Información	30
2.	Comunicación	31
3.	Pensamiento sistémico	33
4.	Transdisciplinariedad	34
5.	Multidimensionalidad	35
6.	Gestión del conocimiento	37
IV.	ESTADO DEL ARTE DESDE LA INGENIERÍA DE REQUISITOS Y EL PENSAMIENTO COMPLEJO	39
V.	DISEÑO METODOLÓGICO	48
VI.	RESULTADOS	52
1.	Principios del Pensamiento Complejo para gestionar la Ingeniería de Requisitos	52
1.1	Multidimensionalidad	53
1.2	Transdisciplina	56
VII.	MODELO PARA GESTIONAR LA INGENIERÍA DE REQUISITOS INTEGRANDO PRINCIPIOS DEL PENSAMIENTO COMPLEJO	61
1.	Etapa temprana	62
1.1	Conocimiento del dominio y el contexto	62
1.2	Comprensión del problema	63
1.3	Conformación del equipo de trabajo	64
1.4	Identificación y resolución de barreras	64
1.5	Sensibilización de las partes interesadas	65
2.	Etapa de elicitación	65
3.	Etapa de desarrollo	67
4.	Etapa de gestión	70
5.	Etapa de especificación	71
6.	Validación y verificación del modelo	74
VIII.	CONCLUSIONES	76
IX.	LIMITACIONES A LA INVESTIGACIÓN	81
	REFERENCIAS	84

PRESENTACIÓN

Somos una Sociedad Software-Dependiente, porque la mayoría de nuestras actividades y relaciones están condicionadas o dependen casi exclusivamente de este desarrollo tecnológico. Pero, al parecer, los procedimientos con los cuales se crea el software todavía no han superado la llamada *crisis del software* de los años 60 y se continúa entregando con deficiencias en calidad, especialmente en temas de seguridad y fiabilidad. Este hecho genera problemas en las empresas y en la sociedad, porque las decisiones comerciales, de inversión y de procedimientos en salud, por ejemplo, no siempre ofrecen los resultados esperados y, en muchas ocasiones, resultan en pérdidas económicas considerables o, lo que es peor, en pérdidas de vidas humanas. Por otro lado, los problemas actuales que se pueden solucionar con productos software han incrementado su complejidad, debido a que presentan una alta interacción entre diferentes disciplinas y áreas del conocimiento y, para resolverlos, es necesario modificar la forma como se lleva a cabo su desarrollo. Por todo esto es apremiante mejorar la fiabilidad del software y, como se evidencia en el marco teórico y el estado del arte, varios autores consideran que las metodologías actuales no responden a esta necesidad.

La primera fase del desarrollo de software se conoce como *Ingeniería de Requisitos* y constituye la base sobre la que se realiza las demás, por lo que se considera la más importante del proceso. Una opción de solución a la problemática de la calidad del software es incorporar, a las metodologías tradicionales, principios del Pensamiento Complejo a través de propuestas para gestionar esta fase. De esta forma los ingenieros podrían construir escenarios y modelar soluciones que representen con mayor credibilidad la vida real del sistema y de modelar soluciones que se acerquen mejor a un contexto de solución seguro.

En este libro se identifica la problemática de la fiabilidad del software y se presenta un método para gestionar la Ingeniería de Requisitos dividido en etapas, al que se incorpora los principios de la Transdisciplinariedad y la Multidimensionalidad del Pensamiento Complejo, para estructurar una propuesta inter-relacional lógica, orientada a gestionar esta fase del desarrollo de software. Se trata de una innovación a la forma como se lleva a cabo esta fase del ciclo de vida, con el objetivo de ayudar a mejorar la fiabilidad y seguridad del producto final. Cada uno de estos principios se analiza desde los aportes de diversos autores y se incorpora las posibilidades de conjugación a lo establecido para la Ingeniería de Requisitos. De esta manera se integran a las etapas del método de forma complementaria a los objetivos que cada una busca en la gestión de los requisitos del sistema.

Prof. **Edgar Serna M.**
Es Profesor, Investigador y Asesor empresarial. Como Científico es Analista, Lógico y Visionario. En lo académico se desempeña como Profesor universitario e Investigador, y profesionalmente como Asesor empresarial en el diseño, implementación y mantenimiento de Arquitecturas de TI, y en Innovación Educativa. Sitio web: http://www.eserna.com/.

Estoy convencido de que el aprendizaje es para la vida y que a los niños: 1) hay que dejarlos vivir, para que entiendan el universo, 2) formarlos como personas, para que encuentren su ubicación en el universo, y 3) capacitarlos como profesionales, para que comprendan e intervengan el universo.

I
INTRODUCCIÓN

En gran medida los adelantos tecnológicos de la *Era Digital* han sido posibles gracias a uno de sus propios productos: el software, que se convirtió en pilar para lograrlos desde que se inició la revolución tecnológica, a mediados de la década de los ochenta. Actualmente, casi la totalidad de las actividades humanas depende de ellos y cada día se incrementan y diversifican, por lo que es posible afirmar que esta Sociedad es *Software-Dependiente*. Aunque esa dependencia es una realidad ampliamente aceptada y reconocida, el proceso de *fabricación*[1] de este producto todavía presenta falencias en calidad, eficiencia y seguridad (Ramingwong, 2011). En parte, porque la misma actividad y sus procedimientos no están regulados profesionalmente, además, porque la formación y el trabajo en Ingeniería del Software parece no tener una línea de investigación bien definida. Por eso se necesita regular y formalizar su ejercicio profesional, una meta que comienza desde la comprensión de las consecuencias de liberar productos que no satisfacen las necesidades de los usuarios (Serna, 2013a).

La Ingeniería del Software[2] determina la metodología del desarrollo de software y sienta las bases y principios para elaborar el producto a través de una serie de fases que, en conjunto, se denomina *ciclo de vida* (Pressman, 2005). En la primera de estas fases, la Ingeniería de Requisitos, se recogen, analizan y documentan las necesidades del cliente, denominadas *requisitos*, y que el producto deberá satisfacer, por lo que se convierte en la base de todas las demás y se considera la más importante[3]. Aunque se ha propuesto diversos modelos para gestionarla, generalmente están orientados a modelar cada sistema como un todo. Además, intentan solucionarlo holísticamente y, hasta el momento, son pocas las iniciativas que integran principios y teorías de otras áreas del conocimiento para ayudar a mejorar el proceso, aunque con diversos resultados (Parviainen, Tihinen y Solingen, 2005; Negele et al., 2005). En parte, esta forma de trabajar hace que los productos se retrasen, que no cumplan con el presupuesto, o que no satisfagan los requisitos, porque su fiabilidad[4] y calidad no son las esperadas (Serna, 2012a). Por otra parte, debido a la dependencia social del software, la ingeniería para construirlos debería garantizar que no tengan defectos o fallas que pongan en riesgo la vida o las inversiones de los usuarios.

Esta última característica es importante, porque el software se puede considerar, directa o indirectamente, como uno de los productos de más alto consumo social. Basta mencionar la popularidad y uso de Internet, y el hecho de que las transacciones bancarias, la salud, el transporte, las comunicaciones y la labor científica están computarizadas, es decir, necesitan software para funcionar. Este producto hace parte de una nueva cultura que abarca el quehacer humano de este siglo que se ha denominado Era Digital, en la que Internet rompió los límites de las fronteras físicas para que las economías y las comunicaciones se globalizaran. Charles Snow (1959) se adelantó a este tiempo y estableció que el mundo se encaminaba a una separación de culturas: por un lado, la cultura científica (ciencia) y por el otro la humanista (sociedad). Definió que a medida que los desarrollos tecnológicos aparecían y se establecían, el lenguaje para su comprensión se restringía a la cultura científica, mientras que la otra cultura se convertía en consumidora. Eso es

[1] Tener en cuenta que el software es el único desarrollo tecnológico que no se construye, sino que se desarrolla, porque es un producto intangible.
[2] Esta disciplina todavía no se acepta como Ingeniería y en muchas ocasiones solamente se aplica como proceso, no como área ingenieril.
[3] El producto de esta fase es el documento de Especificación de Requisitos que contiene una descripción detallada de las necesidades del cliente en términos que se puedan traducir a un lenguaje de programación.
[4] Se refiere a la probabilidad de que el producto funcione correctamente en un entorno y por un período de tiempo determinado.

precisamente lo que sucede hoy con el software, porque solamente lo comprenden sus desarrolladores, mientras que la sociedad en general se limita a utilizarlo y a confiar en que haga bien lo que se necesita que haga.

El modelo tradicional de la Ingeniería del Software se estructura en fases que en términos generales son: 1) Ingeniería de Requisitos, 2) Diseño, 3) Desarrollo, 4) Verificación y Validación, 5) Liberación, y 6) Mantenimiento. La primera es la más importante de ellas, porque sienta las bases sobres las que se construirá las demás en una relación de dependencia incremental, es decir, sino se satisface la anterior no será posible alcanzar la siguiente, aunque se lleve a cabo de forma relacional (de acuerdo con los postulados de la programación). Por eso es importante contar con un modelo que facilite las etapas de esta fase: elicitación, desarrollo, gestión y especificación de requisitos. De esta manera sería posible entregar bases sólidas para la construcción del producto. Ofreciendo otra visión de esta relación entre el software y el aspecto humano de uso otros autores han presentado sus puntos de vista, por ejemplo, Canclini et al. (1996) opinan que los Tratados de Libre Comercio entre naciones generan invasiones y dependencias culturales, porque los productos del país dominante en la relación se superponen en la cultura del otro de manera impositiva. Entre esos productos se encuentran el software y el idioma, que obligan al país dominado a generar dependencias casi innecesarias, como lo que sucede con los teléfonos móviles, los computadores y los dispositivos para diversión. Esta es una cuestión que se debe analizar desde el punto de vista de la fiabilidad y la seguridad del software que se consume, pero a la vez de la necesidad de consumirlo.

Por su parte, Canclini (1995) afirma que esta manera de consumir ha alterado las posibilidades y formas de *ser ciudadanos*. En parte porque las fronteras físicas desaparecieron y porque la libre elección ya no es posible, debido a que el sistema global se ha convertido en impositivo. Es decir, la sociedad entró en un consumo sin interrogantes ni elecciones, porque, como lo expresaba Snow (1959), la cultura tecno-científica les es ajena y su única función es presionar botones y esperar respuestas. Canclini (2004) también sostiene que las transformaciones recientes hacen *tambalear* las arquitecturas de la multiculturalidad, porque la imposición de los desarrollos tecnológicos, a través de la globalización, involucra culturas que no están preparadas para aprovecharlos. El resultado es una sociedad desarraigada en su propio contexto y limitada al uso de productos que poco le ayudan a su desarrollo. Entonces, se van convirtiendo en autómatas que no comprenden que su seguridad es vulnerada y que poco a poco pierden su libre albedrío. Entre esos productos el software tiene un papel sobresaliente, porque no tiene la seguridad ni la fiabilidad suficientes para ser consumido de esta forma y porque en la mayoría de ocasiones no es necesario.

En otra relación interesante: la industria cultural y el software (Canclini, 2006), el autor describe los vertiginosos cambios ocurridos en las últimas décadas en la relación cultura-sociedad. Además, describe cómo esos cambios han hecho que la cultura actual se oriente a la inversión, el mercado y la globalización bajo la administración de programas que responden automáticamente a los deseos de las personas. Aquí cabe interrogarse: ¿qué sucede cuando esa asesoría es equivocada? Existen diversos ejemplos en los que la sociedad pierde inversiones, adquiere objetos inverosímiles o realiza transacciones equivocadas, solamente porque un producto software le *recomendó* hacerlo.

Por otro lado, las necesidades del cliente integran un universo de fuentes, sistemas y tecnologías que, al tratar de analizarlas con una visión holística, el ingeniero del software no alcanza a comprenderlas y modelarlas para desarrollar software que las satisfaga completamente. Por eso se hace necesario integrar nuevos principios en el proceso, con los que sería posible estructurar un modelo[5] para gestionar la Ingeniería de Requisitos de forma diferente y, posiblemente, con

[5] Un modelo de gestión es un esquema o marco de referencia para la realización o consecución de una meta. Tradicionalmente, se conforma de procesos, tecnología y personas interrelacionados a través de estrategias específicas. En este caso, se asume como un esquema teórico de un sistema para atender los procesos de la compleja realidad de la Ingeniería de Requisitos.

mejores resultados. De esta manera el cliente y el usuario podrían expresar sus necesidades con mayor precisión y claridad, y el ingeniero las comprendería y modelaría de mejor forma para las fases siguientes. Pero, lo más importante es que se podría entregar un producto más acorde con su contexto de aplicación y que cumpla en alto grado la fiabilidad y la seguridad esperadas.

Una cuestión que tienen en cuenta las metodologías tradicionales para el desarrollo de software se refiere a la integración en el proceso del ciclo de vida de principios desde otras áreas del conocimiento (McDermid, 1993; Lemberger y Morel, 2012), que podrían aportar al logro del objetivo de un producto software fiable. Pero, en muchos casos, esto no es posible, porque la industria no parece interesada en innovar debido a que los beneficios no los obtendría en el corto plazo, además, porque las universidades no están formando a los profesionales con la capacidad suficiente para hacerlo. En otros casos, porque mayoritariamente se prefiere usar herramientas de automatización para el desarrollo, sin dejar espacio al aporte del razonamiento humano. Pero, aunque se reconoce la necesidad de propuestas innovadoras y aplicables con ciertos niveles de facilidad, escasean los investigadores que se aventuran a proponerlas. Lo que recuerda las palabras de Mary Shelley (2014: 51): *Los antiguos maestros de esta ciencia prometieron lo imposible y sus experiencias y prácticas fueron nulas. Por ello quizá los científicos modernos prometen muy poco*.

Pensando en todo lo anterior, en este libro se desarrolla una idea en la que se propone la integración de principios desde áreas no-ingenieriles a la gestión de la Ingeniería de Requisitos. Para eso se tiene en cuenta: 1) *Complejidad*, porque los problemas del mundo que se pueden resolver con productos software son complejos, y se necesita una visión diferente para comprenderlos y presentarles una solución; 2) *Transdisciplinariedad*, porque estos problemas complejos no se originan desde ni se relacionan con una sola disciplina; 3) *Multidimensionalidad*, porque los sistemas que van a resolver estos problemas relacionan situaciones y responden necesidades que tienen orígenes y destinos multidimensionales; 4) *Pensamiento Complejo*, porque los ingenieros de software deben interpretar adecuadamente el contexto de estos problemas complejos; 5) *Razonamiento lógico*, porque es necesario razonar lógicamente para desarrollar adecuadamente el software; y 6) *Capacidad lógico-interpretativa y abstractiva*, porque es una herramienta esencial para modelar, comprender y navegar el mundo, y para entender cómo funciona el cerebro, porque a fin de cuentas es el órgano con el que se razona antes de tomar decisiones para resolver lógicamente los problemas (Serna y Polo, 2014).

La Ingeniería de Requisitos es la fase encargada de entregarles a los desarrolladores las herramientas necesarias para que conviertan las necesidades del cliente en una solución software. Si bien la comunidad reconoce su importancia y aunque desde hace tiempo diversos investigadores se han mostrado inquietos y han propuesto cómo gestionarla de mejor forma (Bubenko, 1995; Al-Rawas y Easterbrook, 1996; Sommerville y Sawyer, 1997; Nguyen y Swatman, 2000; Satria, 2003; Gervasi y Zowghi, 2005; Birk y Heller, 2007; Firesmith, 2007; Ramingwong, 2011; Serna, 2011a; Vijayan y Raju, 2011; Haße y Michielsen, 2013), todavía no se estructura un modelo que brinde la posibilidad de diseñar una especificación de requisitos fiable y segura (Kossiakoff et al., 2011). Una propuesta de solución para este problema es que los ingenieros de software eliciten los requisitos y generen la especificación integrando teorías y principios, como los descritos anteriormente. De esta forma podrían tener una mejor visión de las relaciones entre los diferentes sistemas, los individuos y los desarrollos tecnológicos para modelar soluciones más cercanas a lo esperado, tal como lo han experimentado visionarios como Lemberger y Morel (2012), Mahaux y Canon (2012), Scharnhorst, Börner y Besselaar (2012) y Curlee y Gordon (2014).

El primer paso en este objetivo es lograr comprender una serie de conceptos relacionados con la temática de esta investigación, de los cuales se presenta a continuación una breve descripción y que en el Capítulo II tendrán la ampliación necesaria. En este orden de ideas hay que empezar por

el concepto *sistema* porque, aunque la Teoría General de Sistemas (Bertalanffy, 1969) definió la *generalidad* del mismo, no reveló su carácter *totalitario*. Actualmente, todo se considera un sistema, pero su significado por sí solo no es suficiente para determinar el alcance epistemológico del término en toda su complejidad conceptual. Visto desde los planteamientos de las Ciencias Computacionales, un sistema es una estructura organizada con un propósito determinado, conformada por un conjunto de elementos interrelacionados e interdependientes que continuamente se influencian entre sí para mantener su actividad y existencia. En otras palabras, es un conjunto complejo de dimensiones e interacciones que buscan un fin común, y no un todo homogéneo y armónico, como se trata de interpretar en la Teoría de Sistemas. Como lo demuestra Edgar Morin (1982) ese holismo surge desde el mismo principio, como una simplificación del reduccionismo al que se opone, es decir, una simplificación de y una reducción para la totalidad. Aquí es donde fallan los ingenieros de software, porque la mayoría no establece principios propios para dilucidar el concepto sistema con el que trabajan.

Otro término es *Complejidad*, porque lo que generalmente se reconoce como complejo, suele ser simplemente complicado, enredado o confuso, que no se puede describir dado el número de mediciones, operaciones y cálculos que requiere. Pero un sistema se puede explicar en términos de principios simples (Downey, 2012) y es en este sentido que la Complejidad es útil en la Ingeniería de Requisitos (Siddiqi, 1996; Sharma y Kushwaha, 2010). Un modelo para gestionar esta fase, en el que se involucren principios de otras áreas, se convertiría en una herramienta de ayuda para los profesionales y sería un aporte para comenzar a ver la Teoría de la Complejidad de otra manera.

Por otro lado, los requisitos tienen inmersa la característica de volatilidad, lo que dificulta hacerles seguimiento y mantenerlos actualizados en la modelación de la solución. Aquí es importante tener en cuenta lo que propone Edgar Morin (1991) en el Método Científico: la *auto-eco-re-organización*. Como propiedad inherente de los sistemas para constituirse, evolucionar y adquirir las entradas del contexto, este principio se puede aprovechar para estructurar un modelo de gestión que atienda esa volatilidad, pero comprendiendo que: 1) los sistemas software son *finitos*, por lo que la recursividad que define la auto-organización generaría sus propios principios de estructuración y funcionamiento, necesarios para determinar la vida útil; 2) la eco-organización le aportaría a la Ingeniería de Requisitos un soporte para comprender las *dependencias internas y externas* que tienen los sistemas con el contexto, y para identificar las entradas y salidas del mismo; y 3) debido a que los sistemas son *dinámicos*, la re-organización se utilizaría para programar sus actualizaciones y realizar las modificaciones, sin afectar drásticamente su objetivo central. Estos principios se convierten en la herramienta básica para modelar las soluciones y atender las necesidades de los usuarios, porque *traduciría* las diferentes interpretaciones del problema en un modelo integrador de visiones y de contextos para desarrollar una solución.

Por su parte, la concepción ontológica de las *realidades espaciales* de Roux (2002), en la que un territorio siempre estará definido por características objetivables como fragmento físico del espacio, se puede re-direccionar en la Ingeniería de Requisitos para: 1) identificar y descomponer el problema en sus componentes y características unidimensionales; 2) caracterizar las superposiciones en el sistema mediante su multidimensionalidad; y 3) establecer las relaciones múltiples al interior y el exterior del sistema, para re-conocer la multidependencia y la multidimensionalidad de sus componentes. Analizar los requisitos desde esta concepción le permitirá al ingeniero comprender el sistema como una realidad ontológica, en la que cada sub-sistema tiene sus propias dimensiones y relaciones, a la vez que se integra multidimensionalmente con otros mediante un proceso continuo de entradas-salidas con y para el contexto.

Moriello (2002) sostiene que los sistemas no son independientes, sino que se sitúan en un contexto que los circunda y envuelve total y absolutamente. Por lo tanto, se encuentran en un constante proceso de intercambio que los modifica, redefine y construye de acuerdo con las

entradas y salidas que se producen en él. Esta característica se podría asimilar a lo que sucede en la Ingeniería de Requisitos, porque el sistema que se resuelve está inmerso en un contexto complejo con el que tiene intercambios y relaciones permanentes. Esto origina la volatilidad de los requisitos, a la vez que la necesidad de mantener una actualización constante de sus relaciones y redefiniciones. Ese amplio contexto, en el que el sistema se origina y del que genera entradas a la vez que proporciona salidas, es multidimensional, por lo que sus interrelaciones se deben comprender para presentarle una solución que, en el proceso, se convertirá en parte del mismo contexto en el que se origina el problema. Si los ingenieros no modelan adecuadamente estas relaciones complejas, la solución que definan atenderá al sistema fuente como un componente aislado, y no podrán integrarlo al universo ontológico del que se desprende.

Actualmente, los productos software se desarrollan mediante un ciclo de vida holístico, en el que las metodologías no tienen en cuenta las múltiples fuentes desde las cuales se originan sus requisitos, ni las diversas dimensiones que hacen parte del problema y de la solución. Al poner en tela de juicio la idea clásica del orden y al derrumbar la representación de objetos sustanciales que responden a planes trazados y organizados para alcanzar logros, la Teoría de la Complejidad define la *Multidimensionalidad* como principio. Además, debido a que el contexto unidimensional clásico no representa ni integra la forma en que vive la sociedad de este siglo, se necesita una nueva manera de abordar la dimensión cognoscitiva para comprender y resolver problemas, con la idea de afrontar esa realidad con pensamiento integrador (Xiong, 2011).

Para Penrose (1991), Morin (1994), Zemelman (2003), De Silva y Carlsson (2004) y Giraldo (2004), la Multidimensionalidad permite una comprensión total del mundo, es decir, ayuda a desenmarañar la madeja de hilos en que se confunde el conocimiento a medida que avanza el siglo XXI. Esto es posible porque hace una relación de todo lo que implica ese tejido para lograr que las personas, a través de procesos interpretativos, desarrollen una definición unívoca del mundo y para que asimilen el conocimiento como global (Richardson, 2010). Esto es importante porque hoy se asume que la dificultad de pensar y vivir es una cuestión que no tiene retroceso, especialmente porque los desarrollos tecnológicos no facilitan esa conjunción (y el software es uno de ellos).

Por otro lado, en este siglo los mundos natural, social, económico y político se caracterizan cada vez más por su inestabilidad, volatilidad y cambios disruptivos, conformando un mundo en el que lo improbable, lo imprevisto y lo francamente catastrófico parece ocurrir con alarmante regularidad. Para Taleb (2007) estos acontecimientos son atípicos, ocurren más allá de los reinos de la expectativa normal, suelen tener consecuencias dramáticas en la vida cotidiana y abundan prácticamente en todos los aspectos de la sociedad: desde lo natural a lo político, lo económico y lo social. A esos problemas en ese mundo es que la Ingeniería del Software debe encontrarles soluciones. Lo que no se sabe, o no se espera, es que parecen empeñados en frustrar los mejores planes de los ingenieros y perturbar de innumerables maneras la vida cotidiana de los usuarios. Por eso, la ingeniería está obligada a reevaluar la comprensión de cómo funciona el mundo y a aprender a adaptarse y a hacerles frente a esos desafíos, dada la incertidumbre que los acompaña. En otras palabras, vivimos en un mundo genuinamente complejo, multifacético, global y entrelazado, por lo que se debe reorientar las herramientas que disponemos para intentar mitigarlo.

Un mundo complejo exige un nuevo tipo de pensamiento, es decir, pensar en la intimidad y la inmediatez como experiencias vividas (Morin, 1977; Nishida, 1992; James, 1996; Ruskin, 2012); reconocer y abrazar el desorden inherente, las contradicciones y el carácter desconcertante de la realidad (James, 1996a; Morin, 2008) en la que los problemas desbordan las categorías familiares de pensamiento (Morin, 1977; James, 1996a; Whitehead, 1997; Bergson, 2010); y comprender el esquema dominante en el que se niega la legitimidad (Ehrenzweig, 1971; Said, 1979; Marcuse, 1991; Lacan, 1998). Además, reconocer que se debe permanecer constantemente alerta a los

supuestos que se dan por sentados, y que continúan ejerciendo un férreo control sobre los hábitos del pensamiento tradicional.

De acuerdo con Edgar Morin (1977), este tipo de pensamiento debe ubicar lo impensado, que lo ordena y controla, y desarrollar un principio que respete y revele sus misterios. Esto implica el reconocimiento de que todas las formas de ver y conocer implican el acto simultáneo de entender y comprender, para evitar la ceguera inevitable de reconocer sin saber la causa. Al dirigir la atención a lo impensado el Pensamiento Complejo aumenta la conciencia de la ignorancia, por lo que hay que empezar por extinguir las falsas certezas que establecen ignorancia, incertidumbre y confusión. Este es el punto radical de partida que debe tomar la Ingeniería de Requisitos, para pensar realmente en la complejidad de los problemas del mundo a los que presenta soluciones.

De otro lado, a la ciencia, como conocimiento obtenido mediante la observación y el razonamiento sistemáticamente estructurado y desde el que se deducen principios y leyes generales, la define Thomas Kuhn (1996) como la búsqueda basada en uno o más logros científicos, que una comunidad particular reconoce durante un tiempo como base para la práctica posterior. Este concepto de ciencia prevé un periodo de tiempo en el que las actividades científicas se desarrollan enmarcadas en un paradigma predominante. Pero la ciencia, debido a que en muchos casos es evidencia, debe ir más allá de los hechos para ser clara, precisa y comunicable. Además, una disciplina es una categoría en la que se organiza los conocimientos científicos, se establece la división y especialización del trabajo y se responde a la diversidad de áreas que abarcan las ciencias. Aunque una ciencia engloba un todo más grande, una disciplina tiende, naturalmente, a la autonomía y a la delimitación de sus fronteras, el lenguaje y las técnicas que tiene que desarrollar o utilizar y, posiblemente, de las teorías que le son propias (Morin, 1994).

En este sentido, y al analizar la sociedad actual, se observa una serie de problemáticas que definen el siglo XXI, tales como el agua, las migraciones forzadas, la pobreza, las crisis ambientales, la violencia, el armamentismo, el neo-imperialismo y la destrucción del tejido social. Esta complejidad hace que ninguna de ellas se puede abordar de manera adecuada desde la esfera de alguna disciplina individual específica, porque claramente representan desafíos Transdisciplinares. Desde la Ingeniería de Requisitos esta situación no se resuelve, como frecuentemente se intenta, creando equipos multidisciplinares en torno a un problema dado, con especialistas en diferentes áreas. Con un mecanismo de este tipo solamente se puede aspirar a lograr una acumulación de visiones e interpretaciones desde cada una de las disciplinas participantes. Se necesita una visión diferente y, aunque una síntesis de integración no se consigue a través de la acumulación de diferentes cerebros, debe ocurrir dentro de cada uno.

Hay que aceptar que el número de problemas sociales complejos se ha incrementado en este siglo, y que los conocimientos técnicos y la comprensión de la ciencia y la ingeniería, necesarios para abordarlos y mitigarlos, están en rápida transformación. El mundo está cada vez más interconectado y surgen nuevas oportunidades y problemas que, cuando no se resuelven oportuna y correctamente, se convierten en crisis, tales como la escasez de energía, la contaminación, el transporte, el medio ambiente, los desastres naturales, la seguridad, la salud, el hambre y la escases mundial de agua, que amenazan la existencia misma del mundo tal como se conoce. Por eso es que ninguno de estos problemas complejos se puede solucionar desde la sola perspectiva de una disciplina tradicional, algo que la Ingeniería del Software debe atender desde la Ingeniería de Requisitos, y estructurar verdaderos equipos transdisciplinares para comprenderlos, abstraerlos, modelarlos y presentarles solución a través de un producto software.

Asimismo, el diseño de sistemas ingenieriles ha demostrado que los enfoques mono, inter o multidisciplinares no proporcionan un ambiente que promueva la colaboración y la síntesis, necesarias para extenderlos más allá de los límites disciplinares existentes, ni para diseñar amplias soluciones realmente creativas e innovadoras a sus problemas. Ellos exigen más que el

diseño, porque presenta numerosos componentes y subsistemas que interactúan en múltiples e intrincadas formas. También implica el diseño, el rediseño y la interacción de sistemas sociales (comunicaciones), políticos (relaciones internacionales), administrativos (gestión del conocimiento), comerciales (costos), biológicos (medio ambiente) y de salud (control de pandemias), entre otros. Además, estos sistemas son propensos a una naturaleza dinámica y adaptativa, por lo que las soluciones requieren muchas actividades que trascienden las fronteras de las disciplinas tradicionales, es decir, se requiere investigación y trabajo transdisciplinar.

Este tipo de investigación y forma de trabajo hace énfasis en los equipos y reúne a investigadores e ingenieros de diversas disciplinas, además, desarrolla y pone en común conceptos, metodologías, procesos y herramientas con el objetivo de generar ideas estimulantes y frescas que expandan los límites de las posibilidades. La Transdisciplinariedad genera en las personas un deseo por buscar colaboración por fuera de los límites de su experiencia profesional, a hacer nuevos descubrimientos, a explorar diferentes perspectivas, a expresar e intercambiar ideas y a adquirir nuevos conocimientos. Este enfoque es el que se requiere en la Ingeniería de Requisitos, porque de otra forma el software continuará sin satisfacer las necesidades de los clientes y usuarios, especialmente en seguridad y fiabilidad.

La investigación desde la que se desarrolla este libro tiene como objetivo: *Integrar, como principios del Pensamiento Complejo, a la Transdisciplinariedad y la Multidimensionalidad en un modelo para gestionar la Ingeniería de Requisitos para mejorar la fiabilidad de los productos software*.

Esto se debe a que, como se mencionó antes, con la aplicación de las metodologías tradicionales estos productos todavía no ofrecen la fiabilidad que los usuarios esperan de ellos. Entonces, la idea es buscar alternativas desde otras áreas del conocimiento para innovar los procedimientos de desarrollo de software, y para ayudar a mejorar su fiabilidad y seguridad. Si esto es posible, los beneficios serán para los equipos de ingenieros y para la sociedad en general, que podrá satisfacer sus necesidades con un software más fiable y seguro. Con esto en mente, los objetivos específicos son: 1) *determinar la forma en que la Transdisciplinariedad y la Multidimensionalidad se pueden integrar en un modelo para gestionar la Ingeniería de Requisitos*, y 2) *realizar la interrelación lógica de estos principios en las etapas de la Ingeniería de Requisitos*.

Se parte de la hipótesis de que *si la Ingeniería de Requisitos se gestionara involucrando Transdisciplinariedad y Multidimensionalidad se podría mejorar la fiabilidad de los productos software*. Además, en el proceso se busca encontrar respuestas a las siguientes preguntas de investigación: 1) ¿qué relación se puede encontrar entre Transdisciplinariedad, Multidimensionalidad e Ingeniería de Requisitos?, 2) ¿Cómo integrar los principios del Pensamiento Complejo (Transdisciplinariedad y Multidimensionalidad) para gestionar la complejidad del conocimiento en la Ingeniería de Requisitos? y 3) ¿En qué medida se puede verificar que con un modelo de este tipo es posible mejorar la fiabilidad de los productos software?

Para integrar los resultados de la investigación, alcanzar los objetivos y responder las preguntas formuladas, el texto se estructura de la siguiente forma:

- *Capítulo II. Marco referencial de conceptos socio-ingenieriles necesarios para el análisis.* Atendiendo las propuestas sociales y desde la ingeniería, cuando una investigación trata de integrar las llamadas *ciencias blandas* con las *ciencias duras*, primero hay elaborar un marco de entendimiento común en cuanto a conceptos, términos y apreciaciones involucrados. Un proceso el que hay que evitar caer en lo que Snow (1959) llamó *la separación de culturas*, mediante una puesta en común del significado relevante del conocimiento involucrado en la investigación. El objetivo de este capítulo es presentar el resumen del diálogo entre científicos sociales y estudiosos de las ciencias duras (Wallerstein, 1996), particularmente de las Ciencias Computacionales, para llegar a un acuerdo sobre los conceptos necesarios. Se describe y

relaciona los conceptos de: Pensamiento Complejo, sistema, requisitos, Ingeniería de Requisitos, gestión de la Ingeniería de Requisitos, Complejidad, Transdisciplinariedad, Multidimensionalidad, gestión del conocimiento y fiabilidad del software. Esta puesta en común es necesaria para que el lector ubique estos significados en el marco teórico, para comprender lo que se investiga y lo que se publica en la misma línea de la investigación.

- *Capítulo III. Marco teórico como recurso interpretativo.* El marco teórico es una especie de guía de actuación en la investigación y responde a dos cuestiones esenciales: *qué y para qué se investiga*, con la idea de encontrar qué es lo que se conoce teóricamente sobre el tema. En este capítulo se presenta el conjunto de ideas, procedimientos y teorías necesarias para llevar a cabo la investigación y responder a las preguntas y objetivos planteados. De esta forma se establece las coordenadas teóricas y los paradigmas a partir de los cuales se puede establecer el proceso investigativo. Se presenta un análisis a los aportes que diversos autores han hecho para el logro del objetivo central, además de un recorrido por los trabajos y resultados de las investigaciones que han aportado para construir y complementar el cuerpo del conocimiento relacionado. Con el Capítulo I direcciona la lectura del documento hacia el estado del arte, de forma que, con la información recopilada en ellos, el lector comprenda las propuestas que tienen alguna similitud o relación con la que aquí se describe.

- *Capítulo IV. Estado del arte desde las perspectivas de la Ingeniería de Requisitos y el Pensamiento Complejo.* Este capítulo contiene el análisis a las propuestas similares e intentos por integrar principios desde otras áreas a la Ingeniería de Requisitos, incluidos el Pensamiento Complejo y la Complejidad. Es un recorrido por la línea de tiempo de los trabajos recolectados en el que se presenta el nivel de madurez que esta línea de investigación ha alcanzado hasta el momento. El objetivo es presentar un acercamiento formal a lo que se ha hecho en el tema para clarificar las ideas preconcebidas al respecto de la temática en los capítulos anteriores. Esto permite definirla, afinarla, delimitarla y enfocarla desde la perspectiva de poder integrar principios no-ingenieriles en la gestión de la Ingeniería de Requisitos. El conocimiento asimilado del trabajo de otros le permite al investigador no repetir sus errores, no buscar lo que otros han demostrado que no se puede lograr y presentar resultados nuevos, que ellos no han determinado. Para enlazar los capítulos anteriores y llegar a demostrar o negar la hipótesis se necesita una metodología para desarrollar la investigación, lo que constituye un proceso formal de trabajo que se estructura alrededor del logro de objetivos y que se describe a continuación.

- *Capítulo V. Proceso metodológico transdisciplinar.* Contiene la descripción de la metodología utilizada en la investigación y que se seleccionó de corte cualitativo. Se tiene en cuenta que este enfoque se centra en el estudio de realidades, subjetivas e intersubjetivas, como objetos legítimos de conocimiento en los que se reconoce que la realidad es un proceso en construcción, además, que los actores involucrados en la producción y comprensión tienen su propia perspectiva de la misma. En el caso específico de este trabajo se aclara que: 1) el tema de investigación es acerca de la integración de principios del Pensamiento Complejo en la gestión de la Ingeniería de Requisitos, 2) la atención se centra sobre todo en los aportes que esta área del conocimiento puede hacer para mejorar la fiabilidad de los productos software, y 3) el interés es interrelacionar lógicamente los principios de la Transdisciplinariedad y la Multidimensionalidad con los objetivos de las etapas de la Ingeniería de Requisitos. Luego que el lector reconoce el proceso bajo el cual se desarrolló la investigación debería tener acceso a los resultados obtenidos en el desarrollo de la misma. Ese es el objetivo del capítulo siguiente, en el cual se describe lo qué se encontró como respuesta a las preguntas de investigación y el logro de objetivos.

- *Capítulo VI. Resultados.* En este capítulo se presentan los resultados obtenidos en la investigación luego de aplicar el proceso metodológico. En la primera parte se describe los principios de Transdisciplinariedad y Multidimensionalidad desde la perspectiva de utilidad

para la Ingeniería de Requisitos, y como respuesta a la primera pregunta de investigación. En la segunda parte se detalla el modelo estructurado para gestionar la Ingeniería de Requisitos, en el que se involucran los principios seleccionados y constituye la respuesta a la segunda pregunta. Se trata de un modelo iterativo e interactivo que se propone como herramienta para que los ingenieros tengan una guía de trabajo para gestionar la Ingeniería de Requisitos, superpuesto en una metodología meramente ingenieril y utilizada ampliamente en la práctica industrial. Hay que aclarar que la gestión del conocimiento es la base y el objetivo sobre el cual se describe cada una de las etapas del modelo propuesto. La información contenida en este capítulo representa el centro de este libro y determina la validación de la hipótesis formulada. En el capítulo siguiente se describe las conclusiones generales de la investigación realizada y de esta forma concluye la guía a través de una lectura enlazada lógicamente. Al igual que en los anteriores capítulos y para facilitar su comprensión y lectura, al final se presenta un resumen estructurado del mismo.

- *Capítulo VII. Conclusiones.* Contiene los argumentos y afirmaciones finales del trabajo realizado, en los que se concluye que sí es posible incorporar Transdisciplinariedad y Multidimensionalidad en los procedimientos ingenieriles de la gestión de la Ingeniería de Requisitos. En el modelo se puede evidenciar esta integración en cada una de las Etapas (temprana, elicitación, desarrollo, gestión y especificación), donde los objetivos y el producto final de cada una se pueden mejorar con esta forma integral para desarrollar sus procesos. Transdisciplinariedad y Multidimensionalidad, en consonancia con la metodología para desarrollar la Ingeniería de Requisitos seleccionada, podrían ayudar a lograr una mejor especificación de requisitos, donde los indicadores y atributos de fiabilidad, que debería alcanzar el software, se involucran como componentes activos y en paralelo para cada una de sus etapas. En el estado del arte se concluye que los intentos por integrar estos principios del Pensamiento Complejo en el desarrollo de software se han direccionado a la Ingeniería del Software como metodología, pero que, por diversas razones, esos intentos no prosperaron como esperaban los investigadores. Aunque no se pudo establecer claramente cuáles esfuerzos tuvieron como objetivo una integración de estos principios a la fase de la Ingeniería de Requisitos, el hecho de pensar en la metodología completa ofrece una oportunidad de aprender de estas experiencias. Por todo esto, en esta investigación no se encontró un modelo a seguir para definir los principios y la forma de integrarlos en la gestión de requisitos. Otra cuestión que contiene este capítulo es la respuesta a la tercera pregunta de investigación, cuya idea principal es que se necesita comenzar a cambiar la forma en que se piensa, trabaja y desarrolla las actividades ingenieriles. Además, y debido a que todo proceso de investigación presenta limitaciones, en el siguiente capítulo se detalla las que se encontraron, además de una autocrítica al proceso y a los resultados de la misma.

- *Capítulo VIII. Limitaciones y autocrítica.* Aquí se describe las restricciones detectadas o presumidas en una aplicación formal del modelo presentado. Entre otras se determinaron: poco conocimiento del Pensamiento Complejo en la ingeniería, organizaciones sin una gestión del conocimiento establecida, ingenieros con experiencia arraigada en trabajo individual-disciplinar, partes interesadas con preconceptos difíciles de modificar y una sociedad que exige productos software con carácter de inmediatez. En el otro componente del capítulo, la autocrítica, se presenta una visión ajena a la que se podría tener como investigador y a través del modelamiento en un escenario de aplicación del modelo propuesto. Aquí se describe los pro y los contra de la aplicación real del modelo y una prospectiva si el objetivo es lograr que sea aceptado y utilizado por los ingenieros en la gestión de la Ingeniería de Requisitos. Pero, aunque las limitaciones son reales, el escenario de aplicación del modelo puede ser la base para superarlas, aunque se requiere un poco de esfuerzo desde la academia para hacerlo conocer y desde la industria para darle una oportunidad práctica.

II

MARCO REFERENCIAL DE CONCEPTOS SOCIO-INGENIERILES NECESARIOS PARA EL ANÁLISIS

De acuerdo con Charles Snow (1959) existe una dificultad en la comunicación entre el mundo científico y la sociedad, es decir, entre los científicos y el pueblo. Sus argumentos son que la terminología que utilizan los primeros no hace parte del vocabulario y comprensión de los segundos, porque se estructura y expresa solamente para el interior de su comunidad. Esta misma comparación podría hacerse entre la ingeniería y las áreas sociales y humanas, donde persiste incomunicación por falta de trabajo transdisciplinar entre ellas.

En el desarrollo de esta investigación ese fue el primer reto: poner en común una serie de conceptos con el objetivo de que ambas culturas comprendieran los comunicados y los enunciados que aparecerían en el desarrollo de cada una. Además, el desafío consistió en encontrar un punto de entendimiento, en el sentido de que no se incurriera en uno de los problemas de la Ingeniería de Requisitos: la ambigüedad. A continuación, se presenta la puesta en común de los conceptos involucrados a lo largo del libro.

1. PENSAMIENTO COMPLEJO

La noción de Pensamiento Complejo propuesta por Edgar Morin (1994), retomada y fortalecida por diversos autores (Lipman, 1998; Kellogg et al., 2005; Ferrara, 2010), se refiere a la habilidad para interconectar diferentes dimensiones de la realidad. En este siglo y con la aparición de eventos y objetos interactivos y multidimensionales con componentes aleatorios, las personas se ven obligadas a desarrollar una estrategia de pensamiento no-simplista ni totalizadora, sino más bien reflexiva, para convivir con ellos. A esta habilidad es a lo que Morin llama surgimiento del Pensamiento Complejo que, como concepto, se opone a la división disciplinar del conocimiento y promueve un enfoque transdisciplinar y holístico, pero sin abandonar la noción de los elementos constitutivos de la totalidad. Se soporta en los Sistemas, la Cibernética y la Teoría de la Información (Benbya y McKelvey, 2006).

El Pensamiento Complejo se basa en tres principios fundamentales: 1) la *diaphore* (la consistencia del sistema que aparece con la paradoja), 2) la *recursividad* (la habilidad de retroalimentar para cambiar el sistema), y 3) el *hologrammie*, la parte en el todo y el todo en la parte (Bickhard, 2011). Por lo tanto, es una estrategia o una forma de pensar que se refiere a un fenómeno global, pero que, al mismo tiempo, reconoce la especificidad de las partes. La propuesta es re-articular el conocimiento mientras se implementa estos principios. Por otro lado, lo que se refiere al Pensamiento Complejo está relacionado con la epistemología o teoría del conocimiento (la doctrina de los métodos del conocimiento científico), cuyo objeto de estudio es la producción y validación del conocimiento a través del análisis de diferentes criterios (Caseau, Krob y Peyronnet, 2007). También tiene una relación directa con la Transdisciplinariedad, cuyo concepto clave es los niveles de realidad y, mientras generalmente las teorías de la complejidad los ignoran, la propuesta de Edgar Morin es compatible con dichos niveles (Cronin, 2008).

Esta no-aceptación se puede comprender desde las mismas definiciones de realidad, porque son como círculos viciosos en los que se define en términos de lo que es real, aunque en un sentido

restringido se puede precisar como todo lo que tiene efectos sobre otra cosa. Esta definición pone el acento en la causalidad, donde también se tiene que definir el tipo de causalidad involucrado. Es el caso de la Ingeniería de Requisitos, porque en ella se plasman los niveles de realidad a medida que evoluciona la comprensión de las necesidades de los usuarios (Benbya y McKelvey, 2006):

1. El primer nivel de realidad es el *estado de las cosas*, y corresponde al momento en que usuarios y clientes intentan explicar en su lenguaje las necesidades del sistema, algo que sucede por fuera del proceso del conocimiento que el equipo de trabajo pueda tener del problema. Además, se ubica en el *axioma ontológico* de la Transdisciplinariedad, porque en este momento se vivencia diferentes niveles de realidad del objeto y, en consecuencia, diferentes niveles de realidad del sujeto que se intenta comunicar.

2. El segundo nivel corresponde a los *estados de las cosas inseparables del proceso del conocimiento*, y sucede cuando el equipo comienza a identificar el valor y las relaciones de los requisitos individuales en un contexto más amplio, originado porque el conocimiento de cada actor se entremezcla con el de los demás para darle cuerpo a cada parte en la que se divide el problema. Esto se corresponde con el *axioma lógico* de la Transdisciplina, donde el paso de un nivel de realidad a otro está asegurado por la lógica del *tercero excluido*.

3. El tercer nivel corresponde a los *estados de las cosas creadas en relación con el proceso del conocimiento*. En la Ingeniería de Requisitos es el momento en que el equipo puede modelar el problema desde una interpretación abstracta y lógica de las descripciones de los actores y del conocimiento adquirido. Este nivel se ubica en el *axioma epistemológico* transdisciplinar, porque a través del modelo aparece la estructura de la totalidad de los niveles de realidad en la representación del conocimiento. En esta compleja estructura cada nivel y axioma es la razón por la cual existen todos los otros niveles y axiomas, materializados al mismo tiempo en la especificación de requisitos.

Este proceso es una iteración continua entre Pensamiento Complejo, Transdisciplinariedad y Multidimensionalidad, porque de otra manera no se podría construir una realidad desde cada nivel, axioma y dimensión. Además, todos están interconectados a través de la complejidad. Desde un punto de vista transdisciplinar la complejidad es una forma moderna del antiguo principio de la interdependencia universal, cuya simplicidad de la interacción no puede ser capturada por el lenguaje matemático, sino por el lenguaje filosófico-simbólico. Pero, aunque el objetivo de la comunidad de los métodos formales es llegar a matematizar la Ingeniería de Requisitos (Cooke, 2011), lograrlo parece una meta difícil y lejana, porque el nivel de formación en matemáticas de los ingenieros, por fuera de esta comunidad, es deficiente (Serna, 2015a), aunque se necesita para darle continuidad y aplicabilidad a los postulados que ella propone. Por el contrario, en la idea de Morin el lenguaje simbólico se puede desarrollar y aplicar en muchas y diversas formas, la matemática entre ellas, y solamente se requiere desarrollar habilidades lógico-interpretativas y abstractivas para impulsar el conocimiento de los requisitos desde un nivel de realidad a otro.

2. REQUISITOS

Los requisitos son una lista de componentes necesarios para el diseño, desarrollo y funcionamiento de un programa que, generalmente, se conocen como *necesidades de los usuarios*. Los requisitos son descripciones de los servicios que un sistema software debe proporcionar y de las limitaciones con las que debe operar. Pueden variar desde enunciados abstractos de alto nivel hasta especificaciones funcionales matemáticas detalladas. En cualquier caso, se deben expresar en una combinación adecuada de declaraciones textuales relacionadas con eficiencia, efectividad, seguridad, fiabilidad y otras tantas[1]. Para el éxito de cualquier proyecto se necesita una

[1] Estas descripciones son las que conforman el documento de la especificación de requisitos.

representación y comprensión clara y consensuada de ellos, lo mismo que de los criterios de aceptación[2] (Kauppinen et al., 2004).

De acuerdo con ISO (2007) un requisito es una declaración que identifica un atributo, una capacidad o una característica del sistema, que tiene valor y utilidad para el cliente, la organización, el usuario u otra de las partes interesadas[3]. Además, es inequívoca, comprobable, medible y necesaria para la aceptabilidad del producto final[4]. Gestionar requisitos consiste en identificar y sintetizar las funciones requeridas por el sistema, asociadas con el rendimiento y otras medidas de calidad, que proporcionan la base para evaluar las soluciones candidatas y para la verificación del sistema completo[5]. Generalmente, se expresan en un lenguaje técnico útil para estructurar una arquitectura y un diseño inequívocos, consistentes, coherentes, exhaustivos y verificables. Por supuesto, se requiere una estrecha coordinación entre el grupo de trabajo para garantizar una correcta *traducción*, desde el lenguaje natural de las partes interesadas al formal de los ingenieros, lo que puede garantizar que se mantenga su trazabilidad. Todo esto se traduce en un conjunto de funciones y descripciones de requisitos, en el que se especifica las características de la solución propuesta que se pueden verificar y validar.

En el desarrollo de productos y en la optimización de procesos un requisito es una necesidad física y funcional documentada que el diseño, el producto o el proceso debe ser capaz de llevar a cabo. El término se utiliza con mayor frecuencia en los campos de las Ciencias Computacionales, aunque es común en muchas otras áreas tecnológicas, científicas y sociales. Por otro lado, la especificación de requisitos se refiere a la documentación del conjunto explícito que debe cumplir el sistema o producto (ASTM, 2012), luego de desarrollar la fase de la Ingeniería de Requisitos.

De forma general los requisitos se clasifican en *funcionales* y *no-funcionales*: los primeros definen las funciones del sistema y de sus componentes, y los segundos las propiedades generales del mismo. Una función es un conjunto de insumos, comportamientos y resultados necesarios para que el software funcione, tales como esquemas de informes asociados, o consultas en línea, y los detalles de los datos con los que operará el sistema. Los requisitos funcionales pueden ser cálculos, detalles técnicos de manipulación de datos y de procesamiento, y otras funciones específicas que definen lo que se supone que un sistema debe llevar a cabo y que, de manera amplia, se expresan en la forma de: *el sistema deberá hacer <requisito>*. Mientras que los requisitos no-funcionales definen cómo se supone que debe ser el sistema, y se expresan en la forma de: *el sistema será <requisito>.* Es decir, son propiedades generales del sistema, o de un aspecto particular del mismo, y no una función específica, por lo que también son conocidos como *cualidades del sistema*. Otros términos para requisitos no-funcionales son restricciones, atributos de fiabilidad, objetivos de calidad, calidad del servicio y requisitos no-conductuales (Stellman y Greene, 2005). El plan para la implementación de los requisitos funcionales se detalla en el diseño del sistema y el de los no-funcionales en su arquitectura.

Gervasi y Zowgh (2005) opinan que uno de los principales problemas con los requisitos es su inconsistencia, que se genera cuando la especificación presenta conflictos, es decir, cuando contiene descripciones contradictorias del comportamiento esperado del sistema[6]. Esas contradicciones pueden surgir: 1) como resultado de objetivos e interpretaciones discordantes entre las partes interesadas, o 2) como consecuencia de cambios no-coordinados que se introducen en la especificación debido a la evolución de los requisitos. La solución propuesta por estos autores es crear sinergia entre el lenguaje natural y la lógica formal, con el objetivo de lograr una mejor interpretación de las necesidades. Lo que olvidan es que la lógica y el lenguaje natural

[2] Se refieren a acuerdos en el equipo de trabajo acerca de la importancia y necesidad de cada requisito en el desarrollo del sistema.
[3] Con este término se identifican todas las personas que tienen alguna participación en la Ingeniería de Requisitos.
[4] Esto lo determina el proceso de las pruebas que lleva a cabo el equipo de desarrollo.
[5] Aquí es donde cobra importancia la Transdisciplinariedad y la Multidimensionalidad, porque los equipos se integran con personas de diversas disciplinas y porque las fuentes de los requisitos son dimensionales.
[6] Un aspecto muy común en la Ingeniería de Requisitos debido en parte a que los equipos no se integran de forma transdisciplinar.

tienen ventajas y desventajas complementarias, y parece complicado encontrar un escenario en el que los requisitos se expresen en este lenguaje y, automáticamente, traducirlos a la lógica formal. Por eso es necesario tener en cuenta que, para realizar una especificación de requisitos formal, los ingenieros deben aplicar análisis y razonamiento formal, y que para discutir los resultados con los usuarios y clientes deben crear el mismo escenario en lenguaje natural y prescindir del razonamiento formal. La propuesta de Gervasi y Zowgh (2005) tiene varios inconvenientes: 1) difícilmente las partes interesadas dominan la lógica formal y no se puede garantizar que los ingenieros lo hagan, 2) la Multidimensionalidad de los requisitos dificultan ese proceso de traducción simultánea, y 3) además de que los escenarios origen son en sí mismos complejos, esta labor le añadiría otro nivel de complejidad. Con el inconveniente adicional de que los ingenieros de software no están formados en Pensamiento Complejo, ni desarrollan adecuadamente la capacidad lógico-interpretativa y abstractiva (Serna, 2015).

Otro asunto es que los requisitos tienen inmersa la característica de *volatilidad*[7], lo que obstaculiza hacerles seguimiento y mantenerlos actualizados al modelar la solución. Aquí es importante tener en cuenta el desarrollo que propone Edgar Morin (1991): la *auto-eco-re-organización*, tal como se explicó previamente. Este modelo se convertiría en una herramienta básica para modelar soluciones y atender las necesidades de los usuarios, porque se podría traducir las diferentes interpretaciones del problema en un modelo integrador de visiones y contextos, para diseñar y desarrollar una solución[8].

3. INGENIERÍA DE REQUISITOS

En los proyectos software la Ingeniería de Requisitos es una actividad de intervención humana, en la que participan las partes interesadas mediante un proceso orientado a elicitar, comprender y modelar las necesidades del cliente. Los requisitos provienen de diferentes dimensiones cuyas fuentes son especialmente personas, con objetivos organizacionales e individuales diversos y con posiciones laborales diferentes. Además, con diversas formas de comprender, expresar y comunicar la información y el conocimiento. Esto hace que la captura y calidad de esas necesidades varíe en función de esas dimensiones (Benbya y McKelvey, 2006). Debido a ello, los modelos actuales para desarrollarlas y gestionarlas son insuficientes, a la vez que se convierten en uno de los factores que puede conducir al fracaso de los proyectos (Reynolds, 2007).

Con el objetivo de ayudarles a los ingenieros de software durante esta fase se han propuesto varios modelos de gestión, sin embargo, es una tarea difícil debido a la poca comprensión y a la alta variabilidad de las situaciones en que se desarrolla cada problema. En esas circunstancias es común que estos profesionales prevean *proactivamente* los potenciales problemas y que identifiquen y documenten *pasivamente* los requisitos[9]. Por eso deben seleccionar un modelo adecuado que les ayude a disminuir hasta cierto punto esa complejidad (Herrmann y Paech, 2008). Pero, debido a que la Ingeniería de Requisitos es una fase crítica y propensa a errores, la realidad es que los métodos actuales no les permite levantar un documento de especificación adecuado para las demás fases (Gunter et al., 2000).

Las principales actividades de la Ingeniería de Requisitos son la elicitación, el análisis, la especificación, la validación y la documentación. En la primera se desarrolla entrevistas con las partes interesadas (usuarios, clientes, ingenieros), que pueden ser seres humanos o sistemas, tales como entornos físicos, organizacionales o legislativos en los que se utilizará el producto (Kotonya y Sommerville, 1998; de Vries et al., 2003). Debido a estos orígenes multidimensionales es poco probable que una única técnica o metodología sea suficiente para elicitarlos. Además, porque los

[7] Esto se refiere a que los requisitos pueden cambiar en cualquier momento y por cualquier factor: legal, económico, personal, empresarial, etc.
[8] Este un principio podría mejorar la fiabilidad de los productos software.
[9] Es decir, producen el documento de especificación de requisitos sin tener total comprensión de los mismos, lo que genera productos sin la fiabilidad esperada.

actores tienen diferentes formas de almacenar, reconocer y expresar su conocimiento acerca del dominio del problema, y porque en la gestión se presentan cambios inesperados en sus necesidades y niveles de experiencia (Alexander, 2003). Por lo tanto, lo tradicional es llevar a cabo la Ingeniería de Requisitos mediante un conjunto de actividades, en paralelo o en secuencia, que incluye una serie de técnicas en línea con el contexto. Estas actividades se repiten hasta que se comprende y documenta el dominio del problema y del sistema en desarrollo, es decir, hasta lograr la especificación de requisitos (Glinz, 2007). Durante la validación y mediante procesos iterativos e incrementales se analiza los requisitos omitidos, los admitidos, los redundantes y los incompatibles, actividades que, debido a la multidimensionalidad de las fuentes, se deben desarrollar de forma proactiva (Lamsweerde, 2001). Además, el equipo de trabajo no solamente debe comprender bien cada técnica de elicitación, sino también seleccionar la que mejor se ajuste al contexto del problema y a las características de las partes interesadas (Sutcliffe et al., 2003).

Por otro lado, y aunque la Ingeniería de Requisitos es un proceso sintético que requiere comunicación social y minería de la información (Cysneiros et al., 2005), los modelos actuales se centran casi exclusivamente en la primera, es decir, en la perspectiva de la participación de los usuarios, careciendo de una integración extensiva a la amplia gama de disciplinas, áreas y teorías que tienen alguna relación con ella y con el sistema en desarrollo[10]. Además, no se ocupan específicamente de cómo utilizar las técnicas en contextos diferentes y que, para desarrollar productos centrados en los clientes y los usuarios[11], es una cuestión que le adiciona complejidad a esta fase. Otro asunto es que los investigadores y los profesionales reconocen que cuando las actividades relacionadas con los requisitos no se realizan adecuadamente los proyectos software son sumamente vulnerables.

En estas circunstancias la presunción es que la parte más difícil de construir un producto software es precisamente *saber qué construir*. Por eso es que ninguna otra parte del trabajo conceptual es tan espinosa como la de especificar los requisitos; ninguna otra parte, si tiene alguna falla, afecta tanto el desarrollo del sistema; y ninguna es tan difícil de corregir más adelante (Jackson y Dick, 2002). Todo esto hace que la Ingeniería de Requisitos sea la tarea más importante que el ingeniero de software realiza para el cliente. Aun así, los proyectos fracasan porque no se realiza una adecuada definición, especificación y gestión de la misma. Entre las causas se puede encontrar factores tales como falta de participación del usuario[12], requisitos incompletos y un manejo inadecuado de las modificaciones inesperadas (Glinz, 2007). En todo caso, la principal función de esta fase es generar especificaciones correctas, que describan con claridad las necesidades de los usuarios y clientes, sin ambigüedades y de forma consistente y compacta. De esta manera se minimiza los problemas relacionados con la gestión de requisitos en el desarrollo del producto.

En este sentido, muchas organizaciones se interesan en mejorar sus prácticas y procesos, debido a que han descubierto que la confianza en una buena especificación de requisitos puede ser la clave para el desarrollo de sistemas exitosos. Pero, como lo describen Kauppinen et al. (2004), implementar sus ideas y convencer a las personas para que apliquen las buenas prácticas en los proyectos puede ser un reto considerable. Esta no es una cuestión trivial y los experimentos que se ha intentado no suelen conducir al éxito, en parte debido a la volatilidad de los requisitos, a la ausencia de razonamiento lógico estructurado en el equipo, a la falta de una comprensión adecuada del problema y a que los clientes y usuarios no documentan adecuadamente sus necesidades (Wiegers, 1999). Por eso es que una de las lecciones aprendidas a partir de la mejora de los procesos software es el reto de enfrentar la resistencia de las personas al cambio[13]

[10] Tales como la Gestión del Conocimiento, los modelos económicos, la normas y la jurisprudencia, el pensamiento global, entre otras.
[11] Una obligación de la Ingeniería de Requisitos es trabajar de la mano con clientes y usuarios, porque son quienes necesitan el producto y serán los que lo utilicen.
[12] Porque se asume que los ingenieros conocen tanto el problema como los modelos de negocios, aunque eso puede llegar a no ser cierto debido a que no están inmersos en el contexto.
[13] Esto parece ser una cuestión natural, porque las personas no se preparan para el cambio y desean que las cosas sigan igual, aunque no estén funcionando adecuadamente y aunque se puedan mejorar para obtener mayores beneficios.

(McFeeley, 1996; Curtis, 1997; Díaz y Sligo, 1997; Zahran, 1998). Según Kaindl et al. (2002) introducir en la práctica general los resultados de la investigación en Ingeniería de Requisitos, claramente es complicado, porque transferir tecnología con éxito depende de una colaboración de doble vía entre investigadores y profesionales. Esta situación exige, por un lado, aplicar razonamiento lógico para convencer a las personas que necesitan cambiar en el entorno de la Era Digital y, por otro, tener en cuenta el principio de la Teoría de la Complejidad para modelar los escenarios empresariales a través de la descomposición en sus diferentes sistemas. De esta forma se les facilita a las personas ubicar su papel y que apliquen las buenas prácticas en la elicitación de requisitos (Downey, 2012).

Por otro lado, actualmente existe un excesivo énfasis en el modelado de casos de uso[14] como *única técnica* para identificar y analizar requisitos, pero algunos investigadores afirman que parecen haberse convertido más en parte del problema que en su solución (Firesmith, 2007). Porque, aunque hasta ahora es la técnica más utilizada para identificar los requisitos funcionales, no es tan eficiente para los no-funcionales. Además, muchos proyectos desarrollan únicamente diagramas de casos de uso, en lugar de utilizar diagramas de secuencia para capturar los caminos normales y excepcionales a través de los primeros, dejando de utilizar el texto para capturar precondiciones de trayectoria, factores desencadenantes, pasos y condiciones posteriores[15]. Un error subyacente es que a menudo solo se desarrolla el camino principal, dejando por fuera muchos caminos secundarios importantes. En otras palabras, capturan lo que el sistema debe hacer bajo circunstancias normales, pero no lo que hace cuando no puede hacer lo que normalmente debería hacer. En este caso, los ingenieros de software deberían utilizar todos los aspectos del modelado de casos de uso para identificarlos y analizarlos o, por lo menos, la mayoría de los posibles caminos de datos (Serna, 2012).

También deberían recurrir a los casos de uso como una técnica de identificación y de análisis, en lugar de una para especificar requisitos, y utilizar técnicas apropiadas para cada tipo, porque es un error asumir que provienen de una sola dimensión y que sus fuentes son holísticas. Entonces, se deberían apropiar de los principios básicos de áreas como la psicología, la neurociencia y la complejidad (Richardson, 2010), además de las matemáticas. Las dos primeras les permite comprender, desde el razonamiento lógico y desde cómo funciona el cerebro, el lenguaje oculto que utilizan los individuos al momento de expresar sus necesidades o ideas que, posteriormente, se convierten en requisitos funcionales o no-funcionales (Serna, 2013). La tercera es una herramienta de soporte que les ayudará a determinar la Transdisciplinariedad y la Multidimensionalidad de los requisitos y a identificar las dimensiones de origen (Benbya y McKelvey, 2006). Al unir estos principios a la técnica de casos de uso sus modelos podrían acercarse a las soluciones esperadas y cubrir de mejor forma la mayoría de caminos identificados.

3.1 Gestión de la Ingeniería de Requisitos

Como ha quedado claro hasta ahora la Ingeniería de Requisitos es un área del conocimiento relacionada con la comunicación[16] entre las partes interesadas, cuyo objetivo es conocer sus visiones, intenciones y actividades en relación con las necesidades de soporte computacional. Además, para desarrollar y mantener una adecuada especificación de las necesidades del sistema, lo que implícitamente incluye la gestión de aspectos empresariales, organizativos, económicos y sociales, lo mismo que de problemas y cuestiones técnicas. Asimismo, la elicitación de requisitos es una actividad de diseño fuertemente comunicativa, interactiva y creativa[17].

[14] Se refiere a representar cada necesidad (requisito) como un caso que se puede moldear y desarrollar como parte de una solución software. El problema es que existe una alta dependencia entre ellas que esta técnica apenas sí considera.
[15] Precisamente esto es lo que se denomina casos de uso dinámicos, donde se considera todas las alternativas y dependencias.
[16] Solamente con un adecuado proceso de comunicación se logrará que el equipo gestione la Ingeniería de Requisitos y que logre estructurar un documento de especificación acorde con la solución planteada.
[17] Por eso es que se requiere una forma deferente de gestionarla, tomando principios que hasta el momento no se han tenido en cuenta, aunque se acepta que son útiles y necesarios.

En la literatura se puede encontrar diversas propuestas para llevar a cabo la gestión de la Ingeniería de Requisitos, desde las más orientadas a la tecnología y la ingeniería de la información, hasta las que se centran en el modelado de la empresa, el negocio y sus objetivos y reglas. Además, se reconoce la importancia de establecer vínculos explícitos entre todos ellos y se puede encontrar un amplio número de métodos de desarrollo de sistemas y herramientas. Aunque tratan principalmente las fases medias y/o finales del ciclo de vida del desarrollo, prácticamente ninguno aborda estructuradamente los principios y el análisis de objetivos del negocio, ni tiene en cuenta las necesidades provenientes de las fases previas, o las que requieren las posteriores. Esto hace que los ingenieros deban pasar de una *comprensión* vaga e informal a un *modelamiento* formal del dominio[18], aunque explícitamente los métodos no sean adecuados para hacerlo, y a representar de forma estructurada el conocimiento organizacional para utilizarlo en las fases posteriores. Esto se debe a que no mantienen los vínculos entre los modelos de negocios y de la empresa y las especificaciones del sistema, y no facilita el razonamiento acerca de las modificaciones necesarias en el sistema en desarrollo cuando se presenta un cambio. Con la aparición de la Sociedad de la Información y el Conocimiento y la Era Digital, a esta situación técnica de la Ingeniería de Requisitos se ha sumado una serie de desafíos (Serna y Serna, 2015) que necesitan soluciones inmediatas:

1. *Gestión de requisitos*, porque es común que las partes interesadas no tengan conocimiento de los roles estratégicos que juegan y, generalmente, su participación es baja. Debido a esto los procesos no se relacionan adecuadamente con las visiones y objetivos de negocio; tienen débil concordancia con el análisis de inversión; poseen métodos de inversión inadecuados; las alternativas de solución no se documentan ni analizan; falta conocimiento para cuantificar los beneficios y los riesgos de los diferentes diseños y requisitos alternativos; se descuida los beneficios intangibles; no se considera las ventajas competitivas del negocio; no son claros el papel y la responsabilidad del departamento de Tecnologías de la Información TI y no son explícitas las metas para el sistema. Esto conduce a descuidar el control del proyecto y a que los presupuestos y calendarios no tengan claridad.

2. *Las partes interesadas*, porque las organizaciones tienden a pensar que su nivel de madurez en TI es alto y, normalmente, no consideran necesario su capacitación para llevar a cabo la gestión de requisitos. Como resultado surgen problemas en la comunicación; es baja la validez de las especificaciones; los usuarios asumen que los ingenieros conocen sus requisitos tácitos y éstos aceptan requisitos sin comprender plenamente las consecuencias[19]. Esto conlleva a que las partes no participen activamente en el proceso, a que se les consulte poco y a que no se les dé la autoridad ni el tiempo suficiente para participar, por lo que se dejan de validar porciones considerables de la especificación de requisitos. El desafío aquí es idear, estructurar y aplicar nuevas formas de gestionar la Ingeniería de Requisitos, de tal manera que se mejore la participación de las partes interesadas[20].

3. *Métodos y metodologías*, porque hay que mejorar la comunicación ingeniero-usuario. Los modelos actuales no son comunicativos y los usuarios y clientes no comprenden plenamente sus descripciones; son prestados de la Ingeniería del Software y no documentan el razonamiento y la lógica detrás de las soluciones sugeridas; no muestran explícitamente los actores del modelado ni sus visiones y necesidades; son neutrales, aunque los problemas no; son unidimensionales, aunque las fuentes de los requisitos son multidimensionales y su componente intencional es insuficiente[21]. Por lo que es difícil entender y reutilizar estos modelos y, por tanto, se aceptan especificaciones sin entenderlas plenamente. Esto hace que los

[18] Algo para lo que los usuarios y clientes no están preparados, y muchos ingenieros tampoco, porque no trabajan en equipo, no tienen una buena comunicación y no tienen en cuenta principios no-técnicos en el proceso.
[19] Especialmente lo relacionado con la fiabilidad y la seguridad del producto final.
[20] Lo que se podría solucionar al conformar equipos transdisciplinares de ingenieros, usuarios y clientes.
[21] Se puede tener una buena intencionalidad, pero si el equipo no se integra y comunica adecuadamente se queda solamente en deseos. Un equipo transdisciplinar es más adecuado para materializar esa intencionalidad.

ingenieros se enfrenten al problema de analizar y determinar su calidad sin haber pasado por un adecuado proceso de gestión. Considerando estos factores y las posibles alternativas de acción, los modelos no permiten conocer suficientemente los procesos de la Ingeniería de Requisitos y, en muchos casos, hacen que las características del sistema y la percepción de las necesidades del usuario evolucionen más rápido que el proceso en sí, por lo que no es posible aplicar ni documentar los cambios en las fases posteriores.

4. *Las herramientas de apoyo*, porque las actuales no son suficientes para llevar a cabo una adecuada gestión debido a que están orientadas a la Ingeniería del Software y les falta nociones para la Ingeniería de Requisitos, por lo que, normalmente, no se emplean en esta fase. Además, la mayoría se desarrolla para pequeños sistemas y son difíciles de aplicar en los grandes y complejos. El reto aquí es desarrollar y experimentar nuevas iniciativas de visualización y abstracción, porque en las etapas tempranas de esta fase los modelos se utilizan principalmente para documentar las necesidades en lenguaje natural. Esto representa dificultades para llevar a cabo el control, el análisis y la calidad en general. Otra cosa que hace falta es mejorar el apoyo al trabajo distribuido, para comprender las diferentes dimensiones y fuentes de los requisitos[22].

5. *Investigación y formación*, porque en la práctica existe pocas investigaciones sobre el proceso de gestión de la Ingeniería de Requisitos y sus problemas. Es necesario investigar, experimentar, proponer y divulgar iniciativas que permitan responder y clarificar cuestiones tales como qué se hace, cómo se hace, cuáles son los problemas más relevantes que se experimenta, por qué se utiliza una herramienta o un método determinado, o por qué no se utiliza, y qué modificaciones en los requisitos afectan la especificación. Es necesario que la comunidad del software cambie su visión técnica y tome prestados conceptos, por ejemplo, del Pensamiento Complejo, para aplicarlos en la Ingeniería de Requisitos, pero esto requiere investigación y formación permanentes.

6. *Integrar principios y teorías desde otras áreas del conocimiento*, porque los modelos vigentes no son suficientes para comprender la complejidad de los problemas actuales y las dimensiones desde las que se origina los requisitos. Algunos investigadores continúan abordando problemas interesantes de la Ingeniería de Requisitos, aunque a menudo sin conocimiento de los temas y de los problemas relevantes, porque no tienen experiencia en la industria ni se han formado en este sentido. Se necesita mayor investigación para determinar los problemas y las soluciones que se puede llevar a la práctica. Por otro lado, la formación académica en esta área no es la adecuada para el trabajo práctico, porque en el aula se utilizan *ejercicios* alejados de la realidad industrial. Esto hace que se deba mirar de otra manera la formación en Ingeniería de Requisitos y aceptarla como una actividad altamente exigente, que deberían llevar a cabo tanto los científicos computacionales como los ingenieros de software. Porque, aunque es un trabajo que requiere ciencia, también se necesita la visión ingenieril para moldear las soluciones necesarias a los problemas complejos.

4. SISTEMA

La Teoría General de Sistemas (Bertalanffy, 1969) definió la *generalidad* de un sistema, pero no reveló su carácter *totalitario*. Por tal razón, actualmente todo se considera un sistema, aunque su significado no es suficiente por sí solo para determinar el alcance epistemológico del término en toda su complejidad conceptual. Visto desde los planteamientos de las Ciencias Computacionales *sistema* es una estructura organizada, con un propósito determinado, conformada por un conjunto de elementos, interrelacionados e interdependientes, que continuamente se influencian entre sí para mantener su actividad y existencia con el objetivo de alcanzar la finalidad de la arquitectura (Serna, 2013). En otras palabras, es un conjunto complejo de dimensiones e interacciones que

[22] Pero sin una integración transdisciplinar y multidimensional es complicado que se pueda obtener un mayor beneficio de las mismas.

buscan un fin común y no un todo homogéneo y armónico, como generalmente se considera. Para Edgar Morin (1982) este holismo surge desde el mismo principio como una simplificación del reduccionismo al que se opone, es decir, una simplificación de y una reducción para la totalidad. Aquí es donde fallan los modelos para gestionar la Ingeniería de Requisitos, porque no establecen auto-principios para dilucidar el concepto *sistema* que van a gestionar.

Moriello (2002) sostiene que los sistemas no son independientes, sino que se sitúan en un contexto que los circunda y envuelve total y absolutamente. Por lo tanto, de acuerdo con las entradas y salidas que se produce, se encuentran en un proceso de intercambio constante que los modifica, redefine y construye. Esta característica podría asimilarse a lo que sucede en la Ingeniería de Requisitos, porque el sistema que se resuelve está inmerso en un contexto complicado, a la vez que complejo, con el que mantiene intercambios y relaciones permanentes. Esto origina la volatilidad de los requisitos y la necesidad de mantener una actualización constante de sus relaciones y redefiniciones. Este contexto amplio en el que el sistema se origina y del que genera entradas a la vez que proporciona salidas es multidimensional, por lo que sus interrelaciones se deben comprender para presentar una solución que se convierta en parte del mismo contexto en el que se origina el problema[23]. Si los ingenieros no modelan este tipo de relaciones complejas la solución que definan atenderá al sistema fuente como un componente aislado, y no podrán integrarlo al universo ontológico del que se desprende.

Todo esto se debe a que actualmente los productos software se desarrollan mediante un ciclo de vida holístico, en el que los modelos no tienen en cuenta las múltiples fuentes desde las cuales se originan sus requisitos, ni las diversas dimensiones que hacen parte del problema y de la solución. Al poner en tela de juicio la idea clásica del orden y al derrumbar la representación de objetos sustanciales que responden a planes trazados y organizados para alcanzar logros, la Teoría de la Complejidad define la Multidimensionalidad como principio. Además, debido a que el contexto unidimensional clásico no representa ni integra la forma en que vive la sociedad del siglo XXI, se necesita una nueva manera de abordar la dimensión cognoscitiva para comprender y resolver problemas, con la idea de afrontar esa realidad con pensamiento integrador (Xiong, 2011).

5. COMPLEJIDAD

Lo que generalmente se reconoce como *complejo* solamente suele ser *complicado*, enredado, o confuso, porque no se puede describir dado el número de mediciones, operaciones y cálculos que requiere. Sin embargo, desde la Complejidad un sistema se puede explicar en términos de unos cuantos principios simples, que a su vez permiten una combinación casi infinita de otros tantos elementos, igualmente simples (Downey, 2012). Por eso es que la Teoría de la Complejidad es útil para gestionar la Ingeniería de Requisitos, porque en ella se encuentra la dualidad básica entre partes, que a la vez son distintas y conectadas, y que los métodos analíticos actuales por sí solos no logran identificar. Además, debido a que los componentes del sistema se mezclan entre sí es difícil describir su comportamiento mediante los métodos clásicos, porque un cambio en alguno de ellos se propaga a través de un complejo tejido de interacciones con los otros, y es espinoso rastrear su comportamiento global en términos de los elementos (requisitos). En el caso de la Ingeniería de Requisitos es imperioso realizar esta integración, porque la sociedad está inmersa y rodeada de problemas complejos, aunque primero se requiere un cambio en la forma tradicional de pensar y ver el mundo que tienen los ingenieros (Edmonds, 1999). Un modelo para gestionar y administrar esta fase del ciclo de vida del software, en el que se involucren principios como Transdisciplinariedad y Multidimensionalidad, se convierte en una herramienta de amplia ayuda para los profesionales y sería un aporte para que se comience a ver a la Teoría de la Complejidad de otra manera, al menos desde las funciones que llevan a cabo como ingenieros de software.

[23] Esto se refiere al concepto de *sistema de sistemas*, porque los productos software son sistemas inmersos en sistemas y no es posible afirmar que un sistema es independiente por sí solo, debido a que siempre tendrá interrelaciones con otros.

Por su parte, la concepción ontológica de las realidades espaciales de Roux (2002), en la que un territorio siempre estará definido por características objetivables como un fragmento físico del espacio, se puede re-direccionar en la Ingeniería de Requisitos con el objetivo de: 1) identificar y descomponer el problema en sus componentes referidos mediante características unidimensionales, 2) caracterizar las superposiciones en el sistema a través de su Multidimensionalidad, y 3) establecer las múltiples relaciones al interior y el exterior del sistema, para re-conocer la multidependencia de sus componentes. Analizar los requisitos desde esta concepción le permitiría al ingeniero comprender el sistema como una realidad ontológica, en la que cada sub-sistema tiene sus propias dimensiones y relaciones, a la vez que se integra multidimensionalmente con los demás sistemas mediante un proceso continuo de entradas-salidas con y para el contexto.

Por otro lado, la complejidad y el tamaño de los sistemas software continúa aumentando y se genera conjuntos de requisitos cada vez más grandes y complejos (Regnell et al., 2008). Debido a eso estos autores se preguntan: ¿cuántos requisitos puede gestionar una organización de desarrollo de software? Y como ellos mismos lo demuestran, es un asunto difícil de cuantificar porque a menudo la caracterización de la escalabilidad de los métodos se queda corta. Al realizar una investigación de nuevas tecnologías para la Ingeniería de Requisitos surge otro interrogante: ¿qué tan grande y complejo debería ser el conjunto de requisitos que se necesita considerar para solucionar un problema? La misma investigación concluye que tampoco es fácil responder, porque, generalmente y en términos de la complejidad y la multidimensionalidad de los requisitos, los investigadores no tienen una imagen completa de la práctica industrial. Aunque se conoce algunas experiencias en casos industriales complejos (Dag et al., 2004, 2005; Regnell, Olsson y Mossberg, 2006), las tecnologías actuales tienen poca o ninguna utilidad.

Uno de los factores que incrementa la complejidad de la Ingeniería de Requisitos es el alto número de éstos porque, así como el conjunto crece también lo hace el número de clientes, usuarios finales, desarrolladores, sub-contratistas, características del producto, interfaces, dimensiones y fuentes del sistema, por lo que es complicado gestionar ese volumen. Además, en casi todos los escenarios, modelados o reales, los requisitos se documentan y enumeran como si su procedencia fuera unidimensional (Regnell et al., 2008); aunque la experiencia ha demostrado que, en gran medida, la complejidad de un conjunto de este tipo está relacionada con la naturaleza de sus interdependencias, y hacerles seguimiento y documentarlos no es una cuestión trivial, ni mucho menos para tomarla a la ligera (Carlshamre et al., 2001). Por eso, a medida que se incrementan también se complica poderlas verificar y validar. Una respuesta a estos desafíos se puede encontrar en la Teoría de la Complejidad, concretamente en su principio de la Multidimensionalidad (Klein, 2005), porque se puede utilizar para gestionar esa interdependencia mediante agrupamientos de requisitos en particiones, creando así un mayor nivel de abstracción para gestionarlos con un esfuerzo razonable (Reynolds, 2007).

Otros investigadores han demostrado que medir el grado de intrincación inherente en el software es un proceso para el que los ingenieros deben desarrollar habilidades cognitivas y de razonamiento lógico (Serna y Zapata, 2014). La iniciativa de Sharma y Kushwaha (2010) tiene en cuenta estas deficiencias en los enfoques y propone identificar la complejidad del software inmediatamente después de interpretar el problema. La cuestión con esta propuesta es que asume que la mayor parte de esa complejidad se presenta en el código, aunque en la realidad, cuando el equipo logra el desarrollo, es demasiado tarde para analizarla. La complejidad se debe mirar en el producto esperado, no en uno de sus componentes, en este caso el código. Por otro lado, la complejidad se puede definir como el grado en que un sistema o componente tiene un diseño o aplicación difícil de entender y verificar (Kushwaha y Misra, 2006) y, para comprenderla, se puede tener en cuenta aportes como los de Wang y Shao (2003), Kushwaha, Singh y Misra (2006) y el de Kushwaha y Misra (2006a). La recomendación es no intentar comprenderla en el software como producto, porque sería demasiado tarde para solucionar los inconvenientes, sino desde los

requisitos que debe satisfacer, es decir, al momento de comprender el problema y de evaluar las posibles soluciones con las partes interesadas[24].

Según Cooke (2011) el grado de complejidad de los problemas actuales abruma a los profesionales, por lo que han tenido que iniciar procesos de cambio de mentalidad y aplicar una perspectiva proactiva (natural o aprendida). Para muchos la obtención de esta habilidad para pensar y actuar de manera integral se puede acelerar, pero no sustituir por un aprendizaje exógeno, porque muchas personas lo logran solamente a través de años de experiencia (Serna, 2011a). A pesar de que todos los proyectos presentan complejidad en alguno de sus niveles, no se pueden ver como un todo homogéneo, porque no sería posible comprender sus inter y extra-relaciones y porque para la mayoría de individuos la cuestión no es solamente comprender cómo los van a entregar, sino cómo manejarlos y esto es lo que definen como complejo (Edmonds, 1999). Desde esta perspectiva complejo y complejidad son términos que se entienden como aquello a lo que se enfrentan, pero que no entienden, cuando en realidad puede ser únicamente complicado. Por el contrario, aquellas situaciones o problemas en las se evidencia relaciones complicadas que componen el todo, es lo que se debería reconocer como complejo (Cooke, 2011). Este tema le interesa a la Ingeniería de Requisitos, porque los proyectos, esencialmente aquellos cuya solución es un producto software, son por naturaleza complicados a la vez que complejos. Es por esto que los ingenieros necesitan desarrollar habilidades para comprender la multidimensionalidad de las fuentes e integrar un modelo que le permita al cliente observar tempranamente una futura solución.

5.1 Transdisciplinariedad

Este principio se ha definido de diversas maneras, por ejemplo, Gibbons y Nowotny (2002) lo analizan como una visión pragmática centrada en un enfoque para resolver problemas concretos; para Basarab Nicolescu (1996, 2008), desde lo epistemológico se refiere a las cosas que tienen en común al mismo tiempo en medio, a través y más allá todas las disciplinas. Su objetivo es la comprensión del mundo y, para lograrlo, una de sus imperiosas necesidades es la unidad del conocimiento. Es una teoría que coloca al ser humano en el centro de sus preocupaciones, una visión con mayor generalidad y adecuada para discutir temas en educación, ética y otros aspectos sociales y científicos.

Si bien las primeras universidades se estructuraron alrededor de cuatro facultades: Medicina, Filosofía, Teología y Derecho como áreas que contenían la totalidad del conocimiento, de hecho, los académicos eran versátiles y eruditos, precursores legítimos de los pensadores y creadores del Renacimiento (Schulz 1970). Con el tiempo esas facultades se especializaron y así surgieron y se multiplicaron las disciplinas y sub-disciplinas, al punto que para la década de 1950 existían más de 1.100 reconocidas, sin incluir a las humanidades y todo lo social (Schulz 1970). La asociación entre disciplinas, departamentos e institutos es un fenómeno relativamente moderno, que comenzó a consolidarse a finales del siglo XIX. Esta departamentalización ha sido importante para mantener las autonomías disciplinares, para la consecución de fondos para investigación y para la consolidación del prestigio académico. De esta forma los profesores y estudiantes desarrollan y aumentan las legalidades disciplinares y, frecuentemente, sienten que la suya es la más importante de todas. Pero en el siglo XXI la situación es diferente: los problemas son más complejos y se acumulan y exigen soluciones inmediatas. Para comprender la evolución de Transdisciplina como término a continuación se define cada etapa de su proceso (Nicolescu, (2008):

- La *Disciplina* se refiere a mono-disciplina y representa una especialidad en el aislamiento. De hecho, una persona puede estudiar Biología y desempeñarse bien sin la necesidad de conocimientos en Física o Psicología. Es más, si se escribiera una lista de ciencias de izquierda

[24] Aquí es donde fallan los modelos actuales, porque consideran el software como un todo y no comprenden la complejidad de cada fase y tarea.

a derecha: Física, Química, Biología, Matemáticas, Computacionales, …, sería posible percibirlas conectadas lógicamente de manera horizontal, pero no vertical. La mono-disciplina se caracteriza por la especialización, un método en el que una disciplina encoge gradualmente los límites de su objeto de estudio, lejos de los de otras especialidades. De manera que, y como resultado de la fragmentación y del enfoque reduccionista, cada área se convierte en insignificante y sin relación con el resto. Quienes critican las mono-disciplinas consideran este método el más ineficaz para hacerle frente a la realidad compleja del mundo actual.

- En la *Multidisciplina* una persona estudia, simultáneamente o en secuencia, más de un área de conocimiento, pero sin hacer conexión entre ellas. De tal manera que puede, por ejemplo, tener conocimientos en Química, Sociología y Lingüística, pero sin generar ninguna cooperación entre esas áreas. Los equipos multidisciplinares son comunes hoy en día. En ellos, los miembros llevan a cabo su análisis por separado y se ven solamente desde la perspectiva de sus disciplinas individuales. Por lo tanto, el resultado es una serie de reportes sin trabajo de integración. Esto sucede a pesar de la convergencia de integrantes de diferentes disciplinas, porque el enfoque multidisciplinar sigue siendo fragmentado y reduccionista, y porque mantiene la autonomía de los participantes: *no es necesario compartir el mismo idioma y la misma meta, simplemente a alguien que clasifique e incorpore los resultados*.

- La *Pluridisciplina* implica cooperación entre disciplinas, aunque sin coordinación. Normalmente ocurre entre áreas del conocimiento afines y sobre un nivel jerárquico común. Como ejemplo podría citarse la combinación de la Física, la Química y la Geología o la Historia, la Sociología y el Lenguaje. El estudio de cada una de ellas refuerza la comprensión de las otras.

- La *Interdisciplina* se organiza en niveles jerárquicos e implica una coordinación desde un nivel más alto a uno más bajo. En este sentido, un propósito se introduce cuando las axiomáticas comunes de un grupo de disciplinas afines se definen en el siguiente nivel jerárquico superior. Pero ¿qué se entiende por nivel jerárquico? Las disciplinas que se enumeraron antes horizontalmente (además de otras) se puede considerar como el primer piso de una edificación y se identifican como el nivel empírico. Inmediatamente por encima, en el segundo nivel, está otro grupo de disciplinas que constituyen el nivel pragmático y que incluyen, entre otras, a la ingeniería, la arquitectura, la agricultura, la medicina, … El tercero es el nivel normativo, que involucra disciplinas tales como la planificación, la política, el diseño de sistemas sociales, el diseño ambiental, … Por último, el nivel superior corresponde al de valor, y está ocupado por la ética, la filosofía y la teología. Así se construye una imagen jerárquica en la que el propósito de cada nivel se define por el siguiente más alto.

- Por su parte, *Transdisciplina* es el resultado de una coordinación entre todos los niveles jerárquicos y que ahora se pueden describir de manera diferente. Las disciplinas en la base describen el mundo tal como es. Aquí se puede aprender de las leyes físicas de la naturaleza y de los principios que impulsan la vida y las sociedades. Este nivel se pregunta y responde a la cuestión de *¿qué existe?* Por ejemplo, desde las Ciencias Computacionales acerca de la Ingeniería de Requisitos; desde la Física acerca de los cuantos; desde la Química acerca de los elementos de la tabla periódica; desde la Biología acerca de la composición de las células y desde las matemáticas acerca de la resolución de problemas. El lenguaje organizador de este nivel es la *Lógica*.

  El siguiente nivel se compone principalmente de disciplinas tecnológicas y se pregunta y responde a *¿qué somos capaces de hacer?* (con lo que se ha aprendido del nivel empírico). Aquí se aprende cómo construir aeronaves, cómo viajar a los planetas y cómo navegar mar adentro. Una cuestión importante es determinar cómo integrar las capacidades humanas, porque se corre el riesgo de hacer las cosas de manera simple y únicamente porque *se sabe cómo hacerlo* (como cuando los procesos se implementan, porque así han funcionado siempre, y no como

debería ser de acuerdo con una eficiente gestión del conocimiento). El lenguaje organizador de este nivel es la *Cibernética*, que enfatiza solamente en las propiedades mecánicas de la naturaleza y la sociedad.

El nivel normativo pregunta y responde a la cuestión¿*qué queremos hacer*? En las sociedades democráticas normalmente las respuestas se someterán a votación y en la mayoría existe libertad de reunión y manifestación pública. Un ejemplo es el control al impacto ambiental que se origina como consecuencia del movimiento ecologista, o el que se logra a partir de otros movimientos, como el de los indignados. Estos y otros son claros ejemplos de las personas que tienen capacidad de influir directamente en su entorno lo que quieren que suceda. El lenguaje organizador de este nivel es la *Planificación*.

El nivel de valor pregunta y responde a ¿*qué debemos hacer*? O mejor ¿*cómo debemos hacer lo que queremos hacer*? Este nivel va más allá del presente y de la inmediatez. Apunta a las generaciones por venir, al planeta como un todo y a una economía en la que importan las personas. Al hacer explícita una preocupación mundial por la especie humana y la vida en general, el lenguaje organizacional es una especie de *Ecología profunda*. Para el caso de la Ingeniería de Requisitos no hace falta decir que, actualmente, no se piensa ni estructura de forma transdisciplinar. En el mejor de los casos se encuentra algunos esfuerzos interdisciplinares (los cuales se describen en el estado del arte), pero las experiencias son principalmente marginales y no integradas en la macro-estructura.

En términos generales y luego de analizar a diversos autores, Transdisciplina significa algo diferente a Interdisciplina, en primer lugar, respeto del estatus científico y en segundo respeto de la función social. En cuanto al primero, el concepto no es simplemente una combinación de disciplinas, sino una transgresión de sus fronteras tradicionales y con ello su transformación en algo nuevo, que tendrá su propia identidad en la medida que disponga de una terminología general que parta de cada disciplina individual. Por tanto, se espera que la Transdisciplina ayude a cerrar brechas tales como: 1) entre las dos culturas de la ciencia (natural) y lo social y lo humano, 2) entre especialistas y generalistas, y 3) entre la investigación aplicada y la básica. En cuanto a la función social de la ciencia el concepto no se adhiere a la hipótesis, de larga data, donde la ciencia debía estar en una torre de marfil, sino que implica una transgresión desde lo científico a lo social (que se ve afectado por sus resultados) y una transformación en una nueva ciencia centrada en lo humano (democrática y participativa).

La sociedad juega un papel importante en esta nueva perspectiva, porque el conocimiento transdisciplinar se re-contextualiza para una audiencia más amplia de múltiples disciplinas, donde se hace más accesible e interpretable (Hunsinger 2005). Para Charles Kleiber (2002) Transdisciplinariedad significa poner en común el conocimiento y la información disciplinar y, debido al aspecto global de los problemas actuales que no pueden ser resueltos por personas o grupos individuales, en las soluciones deben participar otros sectores de la sociedad (Häberli y Thompson 2002)[25]. Además, debido a que el conocimiento es transgresivo la Transdisciplinariedad no respeta fronteras institucionales (Gibbons y Nowotny 2002). De ahí que en la Ingeniería de Requisitos signifique algo más que la suma de profesionales de diferentes disciplinas trabajando juntos, porque también cruza las fronteras del conocimiento para resolver problemas del mundo real. Los equipos que especifican requisitos tienen que tener mente abierta y estar dispuestos a cooperar con no-académicos y con profesionales de otras disciplinas, porque los requisitos se originan desde diferentes dimensiones y disciplinas, y será el trabajo y la gestión del conocimiento transdisciplinar lo que permitirá especificarlos adecuadamente. En este entendimiento todos los integrantes aprenden unos de otros, porque para la Ingeniería de

[25] Este tipo de problemas son los que se resuelven desde la Ingeniería de Requisitos, tal como lo han demostrado Al-Rawas y Easterbrook (1996), Carlshamre et al. (2001), Daneva y Wieringa (2006) y Birk y Heller (2007), entre otros.

Requisitos la colaboración en el trabajo transdisciplinar necesita actores que abran los horizontes y participantes que aporten nuevos puntos de vista e ideas, con el objetivo de comprender mejor el problema a resolver y las necesidades de las partes interesadas (Häberli y Thompson 2002).

En resumen y mirando en retrospectiva la edificación transdisciplinar, es fácil darse cuenta que la mayoría de acciones en la Ingeniería de Requisitos actual no va más allá de combinaciones en los niveles inferiores. En términos de gestión del conocimiento tampoco pasa la prueba y, en la medida en que no se restaura, los productos software no van a ser capaces de resolver con éxito las grandes problemáticas de este siglo. Entonces, ¿qué hacer? Ciertamente no es fácil, pero hay que pensar primero en modificar la forma como se estructuran los equipos de trabajo y la forma de ver y aplicar la Ingeniería de Requisitos, lo que en la mayoría de casos es complicado. Las resistencias internas pueden llegar a ser insalvables porque, generalmente, la disputa para construir prestigio disciplinar ataca vigorosamente cualquier cambio estructural en la forma de trabajo. Pero aun así el cambio es necesario y, a pesar de todas las dificultades existentes, solamente puede venir de adentro del mismo equipo y a través de la acción y la cooperación entre profesionales cultos.

5.2 Multidimensionalidad

Para Edgar Morin (1994) y otros pensadores (Penrose, 1991; Zemelman, 2003; Giraldo, 2004; De Silva y Carlsson, 2004), la *Multidimensionalidad* permite una comprensión total del mundo, es decir, ayuda a desenmarañar la telaraña en que se confunde el conocimiento a medida que avanza el siglo XXI. Esto es posible porque hace una relación de todo lo que implica ese tejido para lograr que los individuos, a través de procesos interpretativos, desarrollen una definición unívoca del mundo y asimilen al conocimiento como global (Richardson, 2010). Esto es importante, porque en la actualidad se asume que la dificultad de pensar y vivir es una cuestión que no tiene retroceso, porque los desarrollos tecnológicos (el software incluido) no facilitan esa conjunción.

La razón es que, dada la complejidad de los problemas de la sociedad, para las organizaciones es una necesidad validar los requisitos multidimensionales en términos de sus fuentes operacionales. Como lo demuestra la propuesta de Romero y Abelló (2006), es necesario descomponer las consultas de entrada para inferir el conocimiento multidimensional implícito y explícito relevante, por lo que en su estudio proponen un conjunto de esquemas automáticos para lograrlo. Con ellos es posible identificar los hechos, las dimensiones y las jerarquías de las necesidades del cliente para apoyar el proceso de la Ingeniería de Requisitos. Lo que estos autores no demuestran es que su propuesta funcione para requisitos multidimensionales reales, porque su proceso ha sido estrictamente de laboratorio, y la mayoría de modelos de este tipo pueden no ajustarse a las necesidades del cliente o al contexto de un problema real. Los ingenieros de software necesitan manipular el modelo que seleccionan de acuerdo con las características propias de cada problema, aunque no lo puedan prever de antemano. Otro inconveniente en esta propuesta es que asume que todas las situaciones a solucionar son lineales, unidimensionales y homogéneas, mientras que en la vida real son Multidimensionales, Transdisciplinares y complejas.

Aun así, la propuesta es interesante porque es de las pocas que hace alusión a las dimensiones de los requisitos, aunque hace falta integrarla con el principio de la Multidimensionalidad visto como una característica intrínseca en los problemas de la vida real. Con perspectivas de trabajo como esta sería posible estructurar un modelo en el que la Multidimensionalidad de los requisitos sea un principio que implique el todo y que involucre todas las formas posibles, para hacerlas converger en un contexto único de resolución del problema. Además, permitiría examinar el proceso lógico-interpretativo y abstractivo que aplican los ingenieros y descubrir e interpretar los supuestos que lo sostienen, lo mismo que sus implicaciones (Serna y Polo, 2014). Por otro lado, se podrían utilizar otros principios del Pensamiento Complejo para relacionar y generar tejidos de eventos y para establecer relaciones significativas, con el objetivo de estructurar una organización

lógico-abstracta adecuada de los conocimientos involucrados en la fase de la Ingeniería de Requisitos.

6. GESTIÓN DEL CONOCIMIENTO

Según Nikula y Sajaniemi (2005), los esfuerzos por mejorar los procesos software son complejos, costosos y, a largo plazo, requieren muchos recursos y compromisos. Los problemas que surgen en la industria durante su implementación se han reportado en numerosos trabajos, por ejemplo, Fowler et al. (1998) describen que los esfuerzos para gestionar el conocimiento en la Ingeniería de Requisitos no son suficientes debido a la falta de recursos adecuados en los proyectos; Nishiyama, Ikeda y Niwa (2000) afirman que incluso una simple modificación de los procesos se puede convertir en una carga para la Ingeniería de Requisitos; para Kaindl et al. (2002) la presión del entorno competitivo no da lugar a encontrar el tiempo para una adecuada gestión del conocimiento, y Lycett et al. (2002) descubrieron que se dedica más tiempo a procesos genéricos, que requieren más esfuerzo que el que se dedica a los específicos. En estos intentos se presentan dos situaciones recurrentes: 1) la indisponibilidad general de las personas a presentar adecuadamente sus opiniones y sugerencias, debido a que no comprenden totalmente el problema, aunque lo vivan a diario (Richardson, 2002), y 2) las empresas no aprovechan el conocimiento interno para mejorar sustancialmente sus procesos, pero sí pueden *comprar* el que se divulga externamente y que complica sus posibilidades de mejora, aunque esto signifique desconocer el que poseen sus empleados e iniciar continuamente procesos de reingeniería innecesarios (Kuvaja, Palo y Bicego, 1999).

De acuerdo con estas apreciaciones queda la idea que las organizaciones parecen no tener en cuenta a la Gestión del Conocimiento como alternativa para solucionar sus problemas. Pero diversos autores han propuesto categorizaciones para lograrlo, tales como las perspectivas *funcionalista* e *interpretativista*. En la primera, el conocimiento se concibe como un *objeto estático* que existe en un número de formas y localizaciones (Hedlund, 1994; Nonaka y Takeuchi, 1995), y en la segunda se considera que no existe independientemente de la experiencia humana, la práctica social, ni el conocimiento mismo y su uso, donde es moldeado por las prácticas sociales de las comunidades, porque es *dinámico y activo* (Tenkasi y Boland, 1996; Brown y Duguid, 1998; Schultze, 1998; Venters, 2002; Serna, 2012b).

Con base en los enfoques racionalista y empirista, el funcionalismo para la perspectiva interpretativista puede ser pertinente para conocer acerca del mundo racional, pero no tiene en cuenta el papel de los individuos como parte del conocimiento en el mundo social. Esto se basa en los estudios de Kant (Kalisman et al., 2005), quien acepta que la mente no es una *tabula rasa* pasiva o una hoja en blanco, sino que participa activamente en la ordenación de las experiencias sensoriales. El argumento es que el conocimiento directo de las cosas, en sí mismas –*noumena*–, es imposible, porque es un concepto que no es conocimiento en sí mismo, pero cuya comprensión se logra mediante la aplicación de nuestro conocimiento *a priori* para crear fenómenos cognoscibles (Johnson y Duberley, 2000). El punto de vista aquí es que el conocimiento humano se logra a través de la experiencia y que es *intrínsecamente indeterminado* (Tsoukas, 1996; Davenport y Prusak, 1998).

Por eso es que en la Ingeniería de Requisitos los equipos de trabajo deben adoptar un enfoque para la gestión del conocimiento y concebirlo como un objeto activo, porque está embebido en las prácticas humanas recurrentes de las partes interesadas, y transferirlo de un lugar a otro no significa que sea un objeto que se hace móvil cuando se desplaza, codifica u ofrece como servicio básico. Más bien, el conocimiento se hace móvil como un producto permanente embebido en toda actividad humana dentro de un contexto social[26].

[26] Algo que se puede lograr cuando se conforma equipos de trabajo transdisciplinares.

Además, por eso es que desde la perspectiva de la Complejidad se indica que el conocimiento es un bien de las organizaciones y, como tal, se debe gestionar desde las dimensiones en las que se origina. Por otro lado, y debido a que el contexto unidimensional clásico para gestionar la Ingeniería de Requisitos no representa ni integra la forma en que viven los individuos en la sociedad, se necesita una nueva manera de abordar su dimensión cognoscitiva para comprender y resolver problemas, si el objetivo es afrontar la realidad multidimensional con un pensamiento integrador. Porque un razonamiento parcelado le impide al ingeniero de software visualizar la complejidad del contexto, y su comprensión del mundo y del problema no sería adecuada (Xiong, 2011).

Debido a esto, desde hace algunos años y como lo explican Daneva y Wieringa (2006), las empresas vienen adoptando los enfoques de negocios colaborativos, con la perspectiva de compartir cierta información que les permita mantenerse competitivas sin alterar la confidencialidad. En esencia, esto significa restringirse a sus competencias básicas para cooperar con las demás a través de sistemas de apoyo a la coordinación general. Los resultados son redes independientes, o casi independientes, que le entregan valor al cliente, pero que acogen en sus necesidades a los procesos propios que deben mejorar. Otros han encontrado utilidad en los principios de estos enfoques, sobre todo en lo relacionado con una sub-división sin sentido de los procesos en componentes[27]. Pero, como lo sugiere Xiong (2011), es necesario integrar los argumentos detectados que clarifican lo complejo, como elementos de la Ingeniería de Requisitos. De esta manera sería posible comprender a la organización desde su interior para mejorar los procesos que argumentan las necesidades de sus sistemas, que luego son analizadas y discutidas para plasmarlas en el documento de la especificación de requisitos.

7. FIABILIDAD DEL SOFTWARE

Es tentador intentar comprender este concepto a través de la misma lente con la que se mira otros tipos de calidad, por ejemplo, la calidad en un proceso de fabricación, aunque se sabe de antemano que en el caso del software hacerlo de esta manera es engañoso. En la industria manufacturera el objetivo principal es minimizar los defectos en los productos creados a través de procesos repetitivos. Metodologías como Six Sigma (Schroeder y Harry, 2006) se crearon para ayudar a hacerlo y han sido muy eficaces. Pero no es lo mismo con el desarrollo de software, porque los proyectos requieren innovación permanente y no es posible encontrar procesos repetitivos comunes para todos, como si fuera una producción en serie. Debido a esto, el concepto de calidad arraigado en la fabricación no es la mejor manera de pensar acerca de la fiabilidad del software, porque se requiere una perspectiva más amplia.

Es claro que somos una Sociedad Software-Dependiente, porque las empresas, la salud, las comunicaciones, el transporte y todos los servicios dependen de este producto. Sin el software la cultura actual se derrumbaría por lo que, ante esta realidad, es importante prestarle atención a su fiabilidad y seguridad. Pero ¿qué es la fiabilidad del software? No es una pregunta fácil de responder, porque el concepto significa diferentes cosas para diferentes personas. Luego de analizar diversas definiciones, una manera de pensar sobre el tema es dividirlo en tres aspectos: calidad funcional, calidad estructural y calidad de procesos.

1. La *calidad funcional* significa que el software ejecuta correctamente las tareas que está destinado a realizar (Pressman, 2005). Entre los atributos de esta calidad se encuentran:

    - *Cumplimiento de los requisitos especificados*. Debido a que la mayoría proviene de los usuarios y clientes, cumplirlos es una condición *sine qua non* de la calidad funcional.

[27] En esta práctica se toma el problema y se intenta comprender desde sus componentes, pero de esta manera se desintegra la homogeneidad del trabajo en equipo.

Incluso, en algunos casos podría circunscribir el cumplimiento de leyes y reglamentos nacionales e internacionales y, puesto que los requisitos cambian frecuentemente a lo largo del proceso de desarrollo, el logro de este objetivo requiere que el equipo comprenda y aplique correctamente todos los requisitos, no solamente los que inicialmente fueron definidos para el proyecto.

- *Crear software con pocos defectos*. Entre ellos se encuentra los errores que reducen la fiabilidad del software, comprometen su seguridad, o limitan su funcionalidad. El logro de cero defectos es mucho pedir para la mayoría de proyectos, a la vez que es difícil encontrar usuarios conformes con un producto defectuoso. Por eso es imperativo detectar y corregir la mayor cantidad posible antes de liberarlo, o por lo menos los más críticos.

- *Rendimiento adecuado*. Desde el punto de vista de un usuario no se acepta tal cosa como una aplicación buena, pero lenta.

- *Facilidad de aprendizaje y de uso*. Para la mayoría de usuarios la interfaz del software es la aplicación, por lo que esperan que se les proporcione una interfaz eficaz y un flujo de trabajo bien pensado. La estética de la interfaz también puede ser importante, especialmente en aplicaciones de consumo.

2. La *calidad estructural* significa que el código está bien estructurado (OMG, 2012) y, a diferencia de la funcional, es difícil de probar, aunque hay herramientas que ayudan a medirla. Entre sus atributos se incluye:

   - *Capacidad de prueba del código*. ¿El código está organizado de forma que se pueda probar fácilmente?

   - *Mantenibilidad del código*. ¿Es fácil agregar nuevo código o cambiar el existente sin cometer o introducir errores?

   - *Comprensibilidad del código*. ¿El código es legible? ¿Es más complicado de lo que tiene que ser? Esto tiene gran impacto en la rapidez con la que los desarrolladores pueden comenzar a trabajar un nuevo sistema con base en un código existente.

   - *Eficiencia del código*. La escritura de código eficiente puede ser de importancia crítica, especialmente en situaciones con recursos limitados.

   - *Seguridad del código*. ¿El software presenta vulnerabilidades comunes, tales como desbordamientos de memoria y saturaciones de bases de datos? ¿Es inseguro de otras maneras?

3. La *calidad del proceso de desarrollo* afecta significativamente el valor recibido por los usuarios, los equipos de trabajo y los clientes, por lo que todos tienen interés en mejorar este aspecto de la fiabilidad (Chappell, 2012). Entre los atributos más evidentes se incluye:

   - *Cumplimiento del calendario*. ¿El software se entregó a tiempo?

   - *Cumplimiento del presupuesto*. ¿El software se desarrolló dentro del presupuesto?

   - *Un proceso de desarrollo repetible que entrega software de calidad fiable*. Si el proceso cumple los primeros dos atributos, pero por tensiones en el equipo renuncian sus mejores miembros, no es un proceso de calidad. La verdadera calidad significa ser coherentes de un proyecto a otro para poder repetir la experiencia.

Existen muchas conexiones entre estos tres aspectos de la fiabilidad del software, por ejemplo, mejorar la calidad del proceso de desarrollo aumenta las probabilidades de cumplir los requisitos y mejorar la calidad funcional. Pero, de la misma forma tienen ventajas y desventajas, porque mejorar la calidad en un área puede reducir la de otra. Una organización puede acelerar el proceso de desarrollo para cumplir con un plazo de entrega, pero se puede incrementar el número de errores afectando la calidad funcional. Del mismo modo, reducir características puede afectar la calidad funcional, porque los usuarios reciben menos de lo que están buscando, pero puede mejorar la calidad del proceso al incrementar las probabilidades de cumplir con la fecha de entrega. En general, cada proyecto de desarrollo debe sopesar los intereses de los tres grupos y los tres aspectos de la fiabilidad, teniendo en cuenta que cada proyecto tiene ventajas y desventajas diferentes.

En todo caso, es común que todos los implicados en un proyecto software se preocupen más acerca de los aspectos de la fiabilidad que los afecta directamente. Los usuarios se preocupan sobre todo por la calidad funcional, porque eso es lo que ven y también son propensos a preocuparse por algunos aspectos de la calidad del proceso, como la fecha de entrega. Pero, normalmente, no se preocupan mucho por la calidad estructural, a pesar de que su ausencia podría afectarlos durante la vida útil del software. Por su parte, el equipo de desarrollo se preocupa por la calidad estructural, porque son las personas las que se verán afectadas por los problemas causados por una baja calidad en este aspecto. También se preocupan por la calidad funcional, aunque tal vez un poco menos que los usuarios, y por la calidad del proceso, en parte debido a que proporciona muchos de los parámetros por los que se está midiendo. El tercer grupo, los clientes, se preocupa por todo, porque las personas que pagan por el proyecto saben que un problema en cualquier aspecto es una mala estrategia a largo plazo. Al final están tratando de crear valor para el negocio y la mejor manera de hacerlo es mediante la adopción de una visión amplia de la fiabilidad del software. También deben entender la conexión entre calidad y riesgo. El riesgo de una menor calidad del software en un sitio web es mucho menor que el de un sistema de control de vuelos, por lo que realizar la elección apropiada exige compensaciones entre objetivos contrapuestos[28].

De esta manera, en este capítulo se presentaron los conceptos generales necesarios para comprender los resultados de la investigación cualitativa que se presenta en este libro, tanto desde lo ingenieril como desde lo social. Aun así, es necesario describir el conjunto de ideas, procedimientos y teorías que sirvieron para realizar el proceso investigativo y, de esta manera, establecer las coordenadas básicas a partir de las cuales se lleva a cabo. Ese es el objetivo del siguiente capítulo donde se presenta el Marco Teórico de la investigación.

[28] Una cuestión que clara para los equipos transdisciplinares.

III

MARCO TEÓRICO COMO RECURSO DE COMPRENSIÓN

Desde mediados del siglo XX el software se convirtió en un desarrollo tecnológico imprescindible para el desarrollo de la sociedad. Los adelantos tecnológicos y la explosión digital de la última década lo afianzaron como producto de uso obligatorio, y hoy es difícil concebir una actividad humana en la que no esté involucrado. A pesar de esto, su proceso de fabricación todavía presenta falencias y las entregas finales no satisfacen totalmente a clientes y usuarios (Mahaux y Canon, 2012). Por eso se necesita analizar esta situación y buscar formas de mejorar su fiabilidad y seguridad involucrando nuevos conceptos y teorías para lograr el objetivo. Desde hace tiempo diversos investigadores iniciaron procesos desde diversas áreas del saber para alcanzar este propósito y, actualmente, esos resultados se pueden aprovechar de mejor forma. A continuación, se describe las que se considera de mayor utilidad para el propósito de esta investigación.

1. INFORMACIÓN

La conclusión general es que gran parte de la información que se necesita en la Ingeniería de Requisitos está incrustada en los mundos sociales de las partes interesadas, que se extrae mediante la interacción entre personas, por ejemplo, a través de la observación, las entrevistas y los cuestionarios, mediante comunicación en lenguaje natural[1]. Por su origen, esta información tiende a ser informal y muy dependiente del contexto social para la interpretación. Por otro lado, muchas representaciones que aparecen en la construcción y el uso de los sistemas informáticos son formales y definen las reglas sintácticas y semánticas para el funcionamiento de los sistemas, por lo que su interpretación es relativamente independiente del contexto social. Por eso es que las representaciones formales e informales de la Ingeniería de Requisitos son necesarias para el éxito de los proyectos (Goguen, 1992).

Aunque a menudo se critica al lenguaje natural, por ejemplo, por los defensores de los métodos formales, debido a su informalidad, ambigüedad y falta de estructura explícita, en realidad estas características pueden ser ventajosas en la elicitación de requisitos: pueden facilitar su evolución gradual sin forzar muy pronto una resolución de los conflictos y ambigüedades que surgen del contexto del problema. Además, es importante no prejuzgar las muchas concesiones que tendrán que ser exploradas más adelante, tales como el costo en comparación con casi todo lo demás, incluyendo velocidad y funcionalidad del producto. Por otro lado, el lenguaje natural complementado con un modelo gráfico es un medio que prefieren los clientes y usuarios para comprender los pasos iniciales de la construcción del sistema (Sage y Rouse, 2014).

Si se conociera más acerca de qué es y cómo se utiliza la información entonces se podría estar en una mejor posición para mejorar la práctica de la Ingeniería de Requisitos, aunque teorías objetivas y cuantitativas de la información, tales como la *teoría estadística* de Shannon y Weaver (1964), o las objetivas pero cualitativas, tales como la *situación semántica* de Barwise y Perry (1983), no son adecuadas para este propósito. Si bien pueden iluminar ciertos aspectos de la información, en la Ingeniería de Requisitos no consideran los procesos sociales que la crean y mantienen a través de la interpretación y la negociación.

[1] Es la forma común como se comunican las personas, pero las máquinas requieren un lenguaje más formal, más técnico y, en el proceso de traducción, se pierde cohesión de la información.

La teoría de Shannon no tiene en cuenta el significado de la información, incluso en el sentido técnico estricto, y no puede explicar cómo puede variar su grado de dependencia del contexto sin dejar de ser situada a fondo. En este sentido vale la pena analizar brevemente y contrastar la visión de la información y su significado y gestión con la teoría representacional de sentido, que se considera más o menos estándar en las Ciencias Computacionales y más generalmente en la tradición analítica angloamericana de la filosofía. Según Ton Jörg (2011) el significado es el logro continuo de algún grupo social en el que se requiere trabajo para interpretar algunas señales, para que puedan ser vistas como información, y que se produce necesariamente en un tiempo y en un lugar determinados.

Por el contrario, la *teoría representacional* de significado afirma que una configuración significativa de signos representa algo en el mundo real (Goguen, 1991). En las teorías representacionales sofisticadas, como la *situación semántica* (Barwise y Perry, 1983), lo que representa, por ejemplo, una frase en lenguaje natural, puede variar con el contexto y, necesariamente, no tiene que ser un objeto simple, porque puede ser una relación situacional compleja. Esto puede ser adecuado para algunos propósitos, pero no para la Ingeniería de Requisitos, sin embargo, incluso la teoría representacional más sofisticada deja de lado la interpretación y la responsabilidad social que la acompañan[2].

Otro principio básico de la teoría social de la información, necesario en la Ingeniería de Requisitos, puede ser una extensión de la obra de Suchman (1987) sobre una demanda más amplia, en la que solamente nuestras explicaciones *post hoc* para eventos situados parecen alcanzar la relativa estabilidad e independencia del contexto (Goguen, Weiner y Linde, 1983; Linde y Goguen, 1978). En particular, interpretar los requisitos indicando si se han cumplido con éxito o no es el resultado de un proceso social complejo que, típicamente, implica negociación y reconstrucción retrospectiva, e incluso puede implicar acciones legales. Por lo tanto, en muchas situaciones puede ser engañoso pensar en los requisitos como pre-determinados, aunque, por supuesto, esto no es general para todas las situaciones. Más radicalmente, a veces puede ser útil pensar que las secuencias causales, en este sentido incluso el propio tiempo, se determinan *a posteriori* por las interpretaciones que las partes interesadas aportan en los acontecimientos[3].

En la Ingeniería de Requisitos esta información, formal o informal, constituye el centro de la especificación de requisitos y, como se puede apreciar, se basa en el conocimiento del dominio, ya sea técnico, funcional, administrativo o social. Por eso es que lo ideal sería que los miembros del equipo de trabajo se incorporaran de forma selectiva, con el objetivo de que los niveles y la distribución del conocimiento cubran desde las disciplinas todos los aspectos de ese dominio[4]. Sin embargo, rara vez es el caso debido al déficit de la gestión del conocimiento en las organizaciones (Curtis, Krasner e Iscoe, 1988). En general, los miembros individuales (disciplinares) no tienen todo el conocimiento requerido para el proyecto, por lo que deben adquirir información adicional antes de llevar a cabo un trabajo productivo (Walz, Elam y Curtis, 1993).

2. COMUNICACIÓN

Resulta que la adquisición de conocimiento y su intercambio en la Ingeniería de Requisitos solamente se puede lograr a través de una comunicación eficaz entre las distintas partes interesadas. Por eso es que se reconoce que los problemas de comunicación son un factor importante en el retraso y el fracaso de los proyectos software (Curtis, Krasner e Iscoe, 1988), especialmente para los sistemas que deben existir en un entorno organizacional complejo. A menudo, los dominios de la organización en la que se introducen son demasiado intrincados y no

[2] En otras palabras, esto se refiere a que en la Ingeniería de Requisitos la interpretación libre no encaja, porque la información debe ser analizada y discutida en equipos transdisciplinares para que tome el significado necesario para la solución propuesta.
[3] Lo que ratifica la necesidad de mejorar el proceso de comunicación antes de acordar una interpretación.
[4] Es decir, se requiere un equipo transdisciplinar.

se entienden completamente, por lo tanto, las descripciones de los requisitos necesariamente serán inciertas.

Algunas de las causas de la ruptura de la comunicación durante la Ingeniería de Requisitos son: canales de comunicación de una sola vía[5], anotaciones masivas, barreras organizacionales, comunicación informal y pérdida de contactos. Estas dificultades no permiten que el equipo de trabajo utilice eficazmente los restringidos canales de comunicación de la Ingeniería de Requisitos tradicional. Uno de los peligros es que cada miembro interpreta los datos a la luz de sus propias suposiciones de fondo (desde su propia disciplina). Esto es especialmente problemático en la comunicación no-interactiva, como en el documento de la especificación de requisitos, porque no hay oportunidad de comprobar que el lector lo interpreta como debería.

McDermid (1993: 31) señala otro problema en la comunicación de los conceptos abstractos de la especificación de requisitos: *Este documento es el resultado de la visión del (los) analista(s), porque generalmente se adueña(n) del problema y expresa(n) sus propias interpretaciones, no las del grupo*[6]. Estas incertidumbres se propagan y multiplican en cada intercambio de información, porque cada miembro trata de darle su propia interpretación disciplinar. Robinson y Bannon (1991) llaman a esto la *deriva ontológica*, como una descripción al cambio en el significado de los términos abstractos cuando pasan entre las diferentes comunidades y disciplinas del equipo de trabajo. En este sentido, y dado que la Ingeniería de Requisitos implica la comprensión, el análisis y la transformación de requisitos informales, ambiguos y quizás contradictorios, en un modelo formal (por lo menos en lenguaje formalizado) preciso y consistente que constituye la base para intervenir el contexto del problema, se ha incrementado la demanda social (Kushwaha y Misra, 2006; Sharma y Kushwaha, 2010) por enfoques para mejorar la calidad del modelo de requisitos que se aplica y, por tanto, de la fiabilidad del producto software. Por eso es que se requiere una profunda comprensión de la complejidad del modelo de la Ingeniería de Requisitos, y su dinámica es fundamental para mejorar los procesos de gestión.

De acuerdo con Nguyen y Swatman (2000) esa complejidad evoluciona con el tiempo y se clasifica en dos tipos diferentes: la esencial y la incidental. La complejidad *esencial* representa la comprensión inherente del espacio del problema, mientras que la *incidental* surge de un ajuste inadecuado entre la estructura del modelo y la estructura del mundo que tiene como objetivo representar. La evolución del modelo implica tanto el crecimiento de la complejidad esencial a lo largo del descubrimiento y la comprensión del espacio del problema, como el crecimiento y la reducción de la complejidad incidental cuando el modelo se somete a gran número de cambios.

Varios autores han descrito la complejidad de la Ingeniería de Requisitos (Malhotra et al., 1980; Guindon, 1990; Batra y Davis, 1992) y para ellos esta fase es una actividad de comprensión y solución de problemas, pero, debido a que la comunicación se realiza en lenguaje natural, lograrlo es una tarea complicada. Visser (1992) y Curlee y Gordon (2014) afirman que a los ingenieros de software no se les entrega requisitos fijos, porque cambian constantemente y no pueden fijar ni definir el problema para el cual deben construir una solución. Por su parte, Guindon (1989) y Olabiyisi, Adetunji y Olusi (2013) la describen como un proceso de descubrimiento de conocimiento, en el que no es posible disminuir la incompletitud ni la ambigüedad del modelo de requisitos emergente. De otro lado, tradicionalmente el modelo de gestión se organiza jerárquicamente y la complejidad situacional se soluciona descomponiéndola en componentes más simples (Jeffries et al., 1981; Negele et al., 2005; Rebel et al., 2013).

Esto ha hecho que esa complejidad se incremente, porque el proceso es un desarrollo evolutivo sistemático del modelo (Douglas y Kuras, 2006; Haße y Michielsen, 2013). Además, en el proceso

[5] Porque los integrantes de los equipos aplican el principio de que la razón la tiene quién más *grite* y no dialogan entre sí.
[6] Una situación que se complica porque, generalmente, son ingenieros escribiendo para ingenieros.

los requisitos evolucionan y se someten a cambios frecuentes, por lo que la volatilidad es una propiedad inherente al modelo de requisitos que, a menudo, causa enormes dificultades al equipo de trabajo (Caseau, Krob yPeyronnet, 2007; Grunwald & Vitanyi, 2010). La mayoría de los autores reconocen que el conocimiento de por qué y cómo se toman las decisiones subyacentes es fundamental para el control y la gestión de los cambios inevitables, lo mismo que en la promoción de la comprensibilidad y la trazabilidad dentro del modelo (Baldwin y Clark, 2000; Maeda, 2006; Douglas, 2009).

3. PENSAMIENTO SISTÉMICO

Para atender la situación anterior los investigadores proponen la estructuración, el desarrollo y la aplicación del Pensamiento Sistémico en la Ingeniería de Requisitos (Savransky, 2000). Partiendo del hecho de que el sistema es un todo complejo cuyo funcionamiento depende de las partes y sus interacciones, hay otros critican los métodos reduccionistas tradicionales, porque no permiten una comprensión adecuada de los problemas que intentan resolver (Checkland, 1981; Capra, 1996). El reduccionismo les da preponderancia a las partes y busca identificar, entender y trabajar el sistema a partir de la comprensión que se logre de ellas. El problema es que a menudo el todo parece asumir una forma no reconocible por las partes, porque emerge de las interacciones entre ellas a través de redes de relaciones complejas y, una vez que el sistema surge, el todo parece darles sentido a las partes y a sus interacciones (Rosenhead y Mingers, 2001; Maxwell, 2003; Jackson, 2003). La historia del Pensamiento Sistémico aplicado está presentada en términos de los esfuerzos por superar las debilidades de los modelos de la Ingeniería de Requisitos.

Este empeño ha permito alcanzar desarrollos importantes y, como enfoque de sistemas, es valorado por hacer importantes contribuciones a la solución de una amplia gama de problemas complejos (Jackson y Keys, 1984). Por eso es que la Ingeniería de Requisitos es un contexto problemático útil para comprender y aplicar el Pensamiento Sistémico, sobre el que se ha desarrollado durante décadas (Kellogg et al., 2005). La mayoría de experiencias permite comprender la variedad de iniciativas que los equipos de trabajo aplican en sus intentos por superar la complejidad de los requisitos, al mismo tiempo que muchas no pueden hacerles frente a situaciones problemáticas cada vez más complejas (Ferrara, 2010). La realidad es que este viaje hacia la comprensión de los problemas en la Ingeniería de Requisitos implica complejidad e incertidumbre y la interacción con múltiples grupos de interés y perspectivas. Por lo que superar la competencia de valores, la falta de criterios de valoración y una terminología ambigua (Martin et al., 2008), no será posible solamente con Pensamiento Sistémico. En pocas palabras, se trata de evitar el desastre de un sistema con baja fiabilidad, porque la sociedad espera que los productos satisfagan sus necesidades de forma eficiente y eficaz.

Por otro lado, los problemas en este siglo son diferentes de aquellos para los cuales se diseñaron y aplicaron las herramientas existentes y, debido al carácter desordenado de los requisitos y a las contradicciones que enfrentan los equipos de trabajo, no puede haber una receta única o un conjunto definitivo de herramientas para encontrar una solución. Sin embargo, algunas formas de pensar y de hacer las cosas parecen más útiles que otras en este contexto, como en el caso del Pensamiento Sistémico (Haße y Michielsen, 2013). Este enfoque, estructurado para algo más que hallar, explorar, comprender y solucionar un problema se puede convertir en una herramienta útil para atender la complejidad de la Ingeniería de Requisitos. Pero pensar y actuar de forma sistémica no es suficiente, porque es necesario tener en cuenta otras características de la Ingeniería de Requisitos que incrementan su complejidad: los requisitos son multidimensionales y sus fuentes son transdisciplinares (Hass, 2009; Serna, 2015). Algunos enfoques reconocen explícitamente estas tensiones y su impacto en la fiabilidad de los sistemas (Lichtenstein et al., 2006; Curlee y Gordon, 2014a), sin embargo, y a pesar de su relevancia, poco se viene haciendo para solucionarlas efectivamente en el documento de especificación de requisitos.

4. TRANSDISCIPLINARIEDAD

En la Ingeniería de Requisitos se puede aplicar un modelo transdisciplinar en la etapa de elicitación para integrar y gestionar el conocimiento que los ingenieros necesitan para especificar los requisitos. Esto no significa que las disciplinas tradicionales involucradas en esta fase del desarrollo se deban desmontar completamente. Significa que las áreas de conocimiento, generalmente incluidas en cada una de ellas, se deben integrar en una estructura transdisciplinar de gestión del diseño y de los procesos, al mismo tiempo que los límites entre ellas serán más delimitados. Además, que los conceptos y los conocimientos de las áreas tradicionalmente no-ingenieriles involucradas, tales como la gestión del negocio, la economía, la política, las relaciones humanas y muchas otras, serán incluidas en la estructura de forma más natural (Nelson, 2001; Tharp y Zalewski, 2001). Así, los ingenieros responsables de la elicitación y la especificación estarán mejor rodeados y en posición de hacerle frente a la complejidad del problema.

Una característica inherente de un modelo transdisciplinar es que todos los conocimientos, habilidades y destrezas de los ingenieros, tales como el análisis, la experimentación, la síntesis, el ingenio, la abstracción, la simulación y el modelado, se consideran herramientas útiles para elicitar y especificar los requisitos, además de que ofrece lo necesario para comprender y solucionar el problema. En los sistemas las buenas soluciones se deben acoplar con las estructuras y organizaciones existentes, y no integrarse como necesidad de último momento (Bickhard, 2011). La Transdisciplina permite una comprensión de cómo los sistemas se integran a partir de componentes y subsistemas y, de esta forma, se ve al problema y su entorno desde una perspectiva global. La expansión del conocimiento y la complejidad de los problemas generan la necesidad de evaluarlos desde una perspectiva transdisciplinar (Mahaux y Canon, 2012).

Como alternativa a la búsqueda de un individuo renacentista con el conocimiento necesario se requiere personas con experiencia en la temática, y con la capacidad para trabajar con eficacia dentro de un equipo transdisciplinar (Lawrence y Després, 2004). Por lo tanto, los líderes de proyectos deben asegurarse de atraer el mayor número de individuos que trabajen con enfoques conceptuales orientados a la resolución de problemas, con pensamiento crítico y con creatividad. Estas personas deben ser entusiastas, creativas, inteligentes y aportar sus visiones para abordar el estudio del problema con una perspectiva general no-disciplinar (Wicklein y Rojewski, 1995).

Un enfoque transdisciplinar involucra mayor cooperación e integración entre disciplinas, donde mutuamente comparten métodos y temas para gestionar el conocimiento generado en cada paso del proceso, por ejemplo, en el desarrollo de software (Tate et al., 2006). La Ingeniería de Requisitos con visión transdisciplinar presenta ventajas sobre los modelos tradicionales: la experiencia y el conocimiento del problema se comparten desde múltiples disciplinas; se tiene una visión del problema desde diferentes perspectivas; se diluye la ambigüedad del lenguaje natural; se logra más fácilmente la formalidad; el diálogo con las partes interesadas se realiza en su propio lenguaje; la documentación es más explícita y los requisitos se discuten y analizan desde fuentes diversas (Thompson, 2004). Además, la utilización de recursos es más eficiente porque se dirigen directamente al foco de necesidad (Kossiakoff et al., 2011).

Por otro lado, el desarrollo de requisitos es un proceso sistemático de descomposición y seguimiento a través de múltiples niveles de abstracción. Las estructuras resultantes son grafos dirigidos en los que las trayectorias de descomposición divergen y convergen a medida que los requisitos son re-factorizados a través de las capas de diseño[7]. En general, la divergencia se espera porque cada etapa de la Ingeniería de Requisitos le añade detalles al proceso, sin embargo, la convergencia se produce, por ejemplo, cada vez que una única característica del diseño satisface múltiples requisitos. Por eso es que en el desarrollo de sistemas complejos convergen diferentes

[7] En palabras más simples, se trata de *masticar* y *re-masticar* la información hasta lograr un bocado digerible, es decir, un requisito aprobado.

equipos para abordar cada capa del diseño, a la vez que analizan la descomposición de los requisitos asociados. Si el equipo no es transdisciplinar la complejidad de toda la cadena de descomposición para generar un requisito puede no ser evidente para un ingeniero individual. Por lo tanto, si el objetivo es mejorar la fiabilidad del producto final, persiste la necesidad de trabajar de forma transdisciplinar (Jackson y Dick, 2002).

De acuerdo con Cronin (2008) actualmente existe una necesidad latente por equipos de trabajo transdisciplinares en todos los campos, porque el conocimiento sobre los problemas socialmente relevantes es incierto, su naturaleza concreta se mantiene en abierta disputa y, para mantener su supervivencia, la humanidad pone cada vez más en juego. Los modelos transdisciplinares son eficientes para: 1) comprender la complejidad de los problemas, 2) tener en cuenta la diversidad de mundo de la vida y las percepciones científicas de los problemas, 3) resumir el enlace y el caso del conocimiento específico, y 4) constituyen conocimientos y prácticas que promueven lo que está concebido para el bien común. Además, promueve la reorientación teórica, conceptual y metodológica con respecto a los conceptos básicos de las disciplinas participantes (McMichael, 2000). El resultado esperado es un potencial creativo de disciplinas colaborativas y maneras de resolver los problemas desafiantes. Este enfoque les enseña a los participantes a buscar la colaboración por fuera de los límites de su experiencia profesional, para descubrir información, explorar diferentes perspectivas, expresar e intercambiar ideas y obtener nuevo conocimiento.

5. MULTIDIMENSIONALIDAD

Para Morin (1994) y otros pensadores (Penrose, 1991; Zemelman, 2003; Giraldo, 2004; De Silva y Carlsson, 2004) la Multidimensionalidad permite una comprensión total del mundo y en la Ingeniería de Requisitos se basa en el principio de dicotomía realidad-dimensión, en la que se considera que una dimensión está conformada por una jerarquía de niveles que representan diferentes detalles para estudiar los requisitos, y que esos niveles contienen los descriptores[8]. Por otro lado, la realidad contiene la información que describe al requisito desde los niveles de comprensión de las partes. Moody y Kortink (2000) consideran que una realidad puede contener no solo uno, sino varios y diferentes niveles de granularidad de los datos necesarios. Por lo tanto, esa información representa las interpretaciones individuales de la granularidad de los datos sobre el mismo requisito[9], por lo que la realidad de un requisito está relacionada con un nivel de las dimensiones de análisis asociadas. Además, una realidad y varias dimensiones dan lugar a la comprensión unificada que se reflejará en su especificación (Kimball et al., 1998).

Este análisis e interpretación multidimensional de los requisitos contribuye a satisfacerlos y, por lo general, se puede estimar y medir en términos de las múltiples dimensiones desde las que provienen (Gilb, 2005). Las implicaciones son: 1) a fin de comprender el verdadero valor y los impactos de una abstracción se puede cuantificar todas las dimensiones críticas de rendimiento del requisito; 2) el error de no estimar correctamente el impacto de cualquier requisito puede hacer que toda la especificación se invalide, o que la elicitación sea menos fiable de lo previsto (Abelló, Samos y Saltor, 2002); 3) al evaluar una especificación de requisitos sobre la base de una o dos dimensiones, es casi seguro que hará falta información útil para comprender las consecuencias de una abstracción (Luján, Trujillo y Song, 2002); y 4) dada la baja probabilidad de que desde una dimensión se pueda obtener información confiable sobre los impactos de los requisitos en el sistema, el equipo de trabajo está obligado a hacer estimaciones de rendimiento, costos, implicaciones, cobertura, pruebas y seguridad, además de tomar algunos riesgos al momento de la abstracción y el modelado de la solución. En la práctica esto se podría lograr una vez que la elicitación se atienda desde la multidimensionalidad de los requisitos (Pedersen y Jensen, 1999).

[8] Esto se refiere a lo que sucede en los equipos de trabajo en la medida que comprenden las necesidades de los usuarios, porque como necesidad social las personas necesitan ser escuchadas y en ese proceso de intercambio se identifica las dimensiones desde las que surgen esas necesidades.
[9] Porque los requisitos tienen múltiples relaciones con el sistema, los otros requisitos y los otros sistemas, tanto de forma entrante como saliente.

En la literatura revisada no fue posible encontrar investigaciones dirigidas expresamente a gestionar las fuentes multidimensionales de los requisitos, aunque Winter y Strauch (2003) presentan resultados y conclusiones orientadas al diseño de bases de datos, y Giorgini, Rizzi y Garzetti (2005) se centran en determinar requisitos de usuario multidimensionales para las bases de datos y para asignar las fuentes de datos. Por otro lado, Cabibbo y Torlone (1998), Golfarelli, Maio y Rizzi (1998), Böhnlein y vom Ende (1999), Hüsemann, Lechtenbörger y Vossen (2000) y Moody y Kortink (2000) realizan un amplio análisis de las fuentes de datos para determinar los conceptos multidimensionales en un proceso de reingeniería.

La noción del origen multidimensional de requisitos ha sido desarrollada por autores como Cardoso (1998) y Hyvärinen y Hoyer (2000), pero apoyados en la hipótesis de que no todas las fuentes pueden ser razonablemente modeladas como procesos de una sola dimensión y con independencia mutua (Ziehe y Müller, 1998). Por el contrario, algunas pueden generar señales que llenan un sub-espacio multidimensional que se resiste a la descomposición en fuentes independientes unidimensionales. Esto puede ocurrir en situaciones donde se desconoce el origen del requisito, donde los ingenieros deben buscar un modelo plausible de los datos observados que les permita, a través de fuentes y situaciones subyacentes, encontrar el carácter multidimensional buscado[10].

El trabajo de Cowling (1998), aunque orientado a la parte curricular de la ingeniería, aporta conceptos y principios que podrían ser útiles para comprender la multidimensionalidad de los requisitos. En el modelo que este autor propone las dimensiones son: 1) los diferentes niveles de abstracción que definen los componentes hardware-software, 2) el equilibrio en el conocimiento computacional y otras ramas de la ciencia o la ingeniería, 3) el equilibrio entre la teoría, la modelización y las aplicaciones prácticas de las mismas, y 4) el equilibrio entre el personal técnico y no-técnico. Cada una se compone de un número de puntos discretos en lugar de un rango continuo, y las relaciones entre ellos necesariamente no son lineales.

Así, en los términos que se utilizan en la teoría de las mediciones estas dimensiones forman escalas ordinales en lugar de intervalos[11] (Fenton, 1991), y cada una se puede utilizar como un eje de ordenadas a lo largo del cual se puede trazar la proporción de conocimiento que se asignará a cada punto discreto (requisito). La otra propiedad importante de estas dimensiones es que son independientes entre sí y los vínculos entre los puntos se derivan principalmente de las relaciones a lo largo y entre las diferentes dimensiones. Estos vínculos son los que determinan principalmente la estructura y el orden la información necesarios para la especificación de requisitos.

En la investigación en bases de datos los reportes indican que los modelos de datos tradicionales no proporcionan un soporte adecuado para las nuevas aplicaciones y, como resultado, ha surgido nuevos modelos con base en una visión multidimensional de los datos. Estos modelos suelen clasificarlos como realidades medibles del negocio o dimensiones que, en su mayoría, son descripciones textuales de esas realidades. Además, deben tener ciertas características con el fin de apoyar los datos complejos que se encuentran en muchos sistemas del mundo real. Pedersen y Jensen (1999) presentan nueve requisitos avanzados que un modelo de datos multidimensional debe satisfacer y los ilustran utilizando un estudio de caso real. La cuestión con este tipo de modelos es que solamente observan la fuente dimensional del requisito o dato, pero no realizan abstracciones ni interpretaciones transdisciplinares para verificar la información colectada. Esto presupone una falta de comprensión de la necesidad e importancia del requisito para la solución buscada (Annoni et al., 2005).

[10] Esto genera el problema antes mencionado de la interpretación personal disciplinar, porque esas señales solamente se pueden interpretar desde una perspectiva y, cuando se entrelazan con otras, generan conflictos de intereses.
[11] Porque la interpretación es jerárquica y no mediada por espacios de separación.

6. GESTIÓN DEL CONOCIMIENTO

A partir de los elementos abordados anteriormente, los aportes e investigaciones desde la teoría de la información, el conocimiento, el pensamiento sistémico, la transdisciplina y la multidimensionalidad, relacionados con la Ingeniería de Requisitos, básicamente son principios que se deben tener en cuenta para alcanzar el objetivo de mejorar la fiabilidad de los productos software. Pero debe existir un *catalizador* que permita unificar criterios, principios y prácticas en pro de alcanzar esta meta y, en esta investigación, se propone a la *Gestión del conocimiento* como agente integrador.

Debido a que el conocimiento es una categoría mental que se refiere a la relación entre la creencia de un individuo y el mundo exterior, hay que evitar ciertas cuestiones epistemológicas y alejarse de la naturaleza o certeza de dicha relación, porque el conocimiento así se caracteriza por la creencia de que es verdadero y justificado (Plato, 1921). En lugar de ello, en la Ingeniería de Requisitos lo mejor es considerar las representaciones externas del mismo, es decir, las expresiones lingüísticas producidas por las personas que lo poseen (Carnap, 1938)[12]. Además, una cualidad característica del conocimiento es que se puede representar y comunicar de forma oral, visual, o escrita. Para esto sirve los metalenguajes que se desarrollan sobre una base transdisciplinar y para estudiarlo explícitamente con su representación e interpretación.

Con base en estos metalenguajes en la Ingeniería de Requisitos hay que darle significado a la integración del conocimiento transdisciplinar y multidimensional. Para lograrlo, y debido a que hay que respetar su origen disciplinar, es necesario elaborar procesos compartidos y diseñar una metodología que conduzca a su integración, comprensión y comunicación. En este proceso hay que tener en cuenta que en la integración del conocimiento transdisciplinar ya se han desarrollado intentos fallidos, como el del empirismo lógico (Neurath, 1938) y el de la Teoría General de Sistemas (Bertalanffy, 1969), porque el objetivo es re-conocer la necesidad de los requisitos[13]. La dificultad que enfrentaron estos y otros intentos fue tratar de encajar los lenguajes, las teorías y los métodos disciplinares como piezas de un rompecabezas (Kitcher, 1999).

La realidad es que, en relación con los requisitos, cada disciplina abstrae el mundo físico de manera diferente, porque solamente selecciona los aspectos necesarios y abandona los demás (Jaeger, 2003). De esta forma se concentra en un pequeño número de aspectos para solucionar un problema en el mundo complejo. Además, por su propia naturaleza, los lenguajes y las teorías disciplinares tienen como objetivo simplificar el mundo real para un propósito específico, por lo que su alcance es limitado. En contraste con esta visión, los problemas complejos integran el conocimiento, pragmático y específico, además de las dimensiones en que se origina y de las variables que lo definen.

En la Ingeniería de Requisitos el conocimiento no existe en la ausencia de diálogo, ya sea a través de interacciones socio-culturales, el intercambio con pares, o porque se extrae de la naturaleza y se acumula a través del intercambio bidireccional de información (Kitchenham y Pfleeger, 1996). Sin este diálogo no hay transmisión, revisión, gestión, o expansión, porque son procesos inherentemente reflexivos, dinámicos y transdisciplinares. Si no se comparte, todo conocimiento será estéril, estará condenado al olvido y no mutará ni evolucionará. Por eso es que, como transdisciplinar, se potencializa en el intercambio constructivo entre disciplinas y mediante la gestión asertiva y estructurada de la elicitación de requisitos[14].

En este sentido, las bases teóricas que soportan los principios de la Transdisciplina y que han puesto de manifiesto una posible crisis en la Gestión del conocimiento, ponen de relieve la

[12] Esto se debe hacer inicialmente en lenguaje natural, aunque para la Ingeniería de Requisitos sería preferible de manera formal.
[13] Es decir, no asumir la necesidad de un requisito sin haberlo analizado y sometido a escrutinio, porque sería como trabajar tradicionalmente.
[14] Claro que solamente será posible si los equipos se integran de forma transdisciplinar.

necesidad de modelos que interrelacionen información, disciplinas múltiples y equipos de trabajo, para comprender y solucionar los problemas complejos de este siglo (King et al., 2009). No es solamente la cantidad de conocimiento que se produce alrededor de los requisitos, sino quiénes tienen acceso a él, cuáles son sus relaciones de poder en el equipo de trabajo, cómo se adapta e incorpora el trabajo transdisciplinar a un documento de especificación de requisitos y cómo se puede gestionar para hacerlo útil en el diseño de la solución.

De esta manera las etapas de la Ingeniería de Requisitos implicarían el descubrimiento de nueva información, porque existe un equipo transdisciplinar que induce una exploración de conocimientos, actitudes, interpretaciones y prácticas relacionadas con cada requisito en particular. Este equipo trabaja de manera conjunta y, a través del diálogo, descubre y potencializa información que convierte en conocimiento, para gestionarlo adecuadamente y definir las abstracciones y las interpretaciones de cada uno de los requisitos elicitados. El problema es la falta de capital humano preparado para realizar este tipo de trabajo (Levison et al., 2011). En todo caso se necesita modelos para gestionar el conocimiento colectado sobre los requisitos, que permitan conocer su nivel de abstracción y que, mediante actividades de creación, diálogo y socialización, permitan integrarlo en el documento de la especificación (Serna, 2015).

Este marco teórico presenta el resumen de lo que se investiga, para qué se hace y cómo se difunde. En él se tomó la postura de que es necesario gestionar de mejor forma la Ingeniería de Requisitos e innovar los modelos actuales teniendo en cuenta principios de áreas no-ingenieriles. Además, se presentó el conjunto de ideas, procedimientos y teorías necesarias para llevar a término la investigación. De esta forma se establecen las coordenadas a partir de las cuales se desarrolla el proceso investigativo que se describe más adelante. Pero antes de presentar este desarrollo, es necesario analizar las propuestas e intentos similares por integrar los principios propuestos en este trabajo a la Ingeniería de Requisitos, lo cual se describe en el siguiente capítulo.

IV

ESTADO DEL ARTE DESDE LA INGENIERÍA DE REQUISITOS Y EL PENSAMIENTO COMPLEJO

La necesidad de mejorar la gestión de la Ingeniería de Requisitos es apremiante, porque actualmente los proyectos software tienen una vergonzosamente alta tasa de fracasos. Por otra parte, esta fase es la más propensa a errores y corregirlos en otro momento del desarrollo es costoso e improcedente (Boehm, 1981; Davis, 1990), por lo que tiene gran influencia en el costo del producto lo mismo que en su calidad. Además, la fiabilidad del software depende en gran medida de que la Ingeniería de Requisitos se gestione eficientemente, porque esta fase involucra a la especificación de requisitos que, a su vez, es la base para el diseño y las demás fases del ciclo de vida del producto. Por desgracia, también es un área poco explorada y tiene los fundamentos intelectuales menos satisfactorios en la Ingeniería del Software, aunque es un campo en el que se debería invertir mejores esfuerzos de investigación. El hecho de que los requisitos surjan de interacciones sociales, que son la base de muchas de sus dificultades, obstaculiza el hecho de modelarlos o simularlos con las técnicas habituales, lo que sugiere que se requiere nuevos enfoques para gestionarlos. A continuación, se describe el estado del arte relacionado con la gestión de la Ingeniería de Requisitos hallado en la revisión de la literatura de esta investigación.

Joseph Goguen (1994) explora modelos de gestión desde una variedad de campos, especialmente desde la etnometodología y el análisis del discurso, para analizar textos escritos, el lenguaje hablado y las interacciones naturales en un contexto social. Además, utiliza ideas de las Ciencias Computacionales y de la sociología de la ciencia para ayudar a comprender el aspecto formal de los requisitos. Uno de los resultados de su investigación se orienta hacia una teoría social de la información.

Desafortunadamente, la etnometodología puede ser difícil de entender, aunque se ha presentado exposiciones relativamente comprensibles de ciertos puntos relacionados (Levinson, 1983; Suchman, 1987; Jirotka, 1991; Goguen y Linde, 1993). Por su parte, al análisis del discurso, surgido a partir del trabajo de Harvey Sacks (1974, 1992), hace aportes sobre cómo los oradores organizan detalles tales como sincronización, superposición, respuesta, interrupción y reparación en una conversación ordinaria. Todos estos campos son fuertemente empíricos por lo que, normalmente, no aplican el principio básico de la rendición de cuentas y no se puede estar seguro de que los eventos en los datos tengan algún significado social natural. Por ejemplo, los datos obtenidos mediante entrevistas en la elicitación de requisitos no se pueden utilizar, a menos que sea posible estudiar el contexto de lo que sucede en la entrevista. Estas dificultades han imposibilitado una amplia utilización de estos modelos en la Ingeniería de Requisitos y no se encontraron estadísticas acerca de que mejoren la fiabilidad del software.

Por otro lado, se ha desarrollado modelos conceptuales para aspectos individuales de la Ingeniería del Software, tales como el *modelado de procesos* (Ambriola y Montagnero, 1992; Bandinelli, Fuggetta y Grigolli, 1993; Tankoano, Derniame y Kaba, 1994), la *especificación orientada a objetos* (Rumbaugh et al., 1991; Fernström, Närfelt y Ohlsson, 1992; Booch, 2007), la *arquitectura del software* (Fernström, Närfelt y Ohlsson, 1992; Perry y Wolf, 1992; Garlan, Allen y Ockerbloom, 1994) y la *Ingeniería de Requisitos* (Pohl, 1992). El problema de estas propuestas es que se centran en áreas aisladas del asunto central, lo que ha llevado a modelos teóricos sofisticados y, por lo

tanto, a lograr información y conocimiento más profundos sobre cada área (diseño, pruebas, codificación, arquitectura, implementación), pero de forma aislada. En la realidad de los proyectos de desarrollo de software el uso práctico de este tipo de modelos individualizados es limitado, porque no permiten comprender la amplitud del proyecto cuando interfieren diferentes áreas de interés y cuando interactúan personas con diferentes roles y funciones. Como consecuencia de esta deficiencia su aceptación entre los profesionales y la industria del software suele ser bastante baja (Potts, 1993) y no logran el objetivo central de mejorar la fiabilidad del producto final.

Para atender a este reto Andreas Zamperoni (1996) afirma que la investigación en el campo de la Ingeniería del Software suele tratar su complejidad a través del estudio de diferentes áreas de interés (como las mencionadas previamente), pero de forma separada. El objetivo de su trabajo es desarrollar un modelo formal multidimensional, que integre a otros modelos y áreas de interés del desarrollo de software, como base para describir (y prescribir) consistentemente diferentes tareas de proyectos software reales y desde diferentes puntos de vista. Con este objetivo propone construir una herramienta que apoye ese tipo de modelos, pero que les esconda a los usuarios la complejidad y la formalidad. Este investigador llama a su modelo Three-Dimensional Model of Software Engineering 3DM y gestiona la Ingeniería del Software desde tres dimensiones: 1) los procesos de desarrollo, 2) los componentes del sistema, y 3) los puntos de vista sobre el sistema. El inconveniente con su propuesta es que trata de gestionar el proceso de desarrollo del software como un todo, desconociendo la importancia de cada uno de sus componentes, especialmente de la Ingeniería de Requisitos como base del ciclo de vida. Por lo tanto, no ofrece una posibilidad real para que un equipo transdisciplinar gestione la Ingeniería de Requisitos.

Al-Rawas y Easterbrook (1996) investigan la falta de calidad en las soluciones software y concluyen que, en parte, se debe a que la especificación de requisitos se basa en el conocimiento del dominio del problema que tienen las partes interesadas, ya sea técnico, funcional, administrativo o social. Aunque, en general, los miembros individuales no tienen todo el conocimiento requerido para el proyecto y deben adquirir información adicional antes de llevar a cabo un trabajo productivo (Walz, Elam y Curtis, 1993). Para estos autores lo ideal sería que el equipo de trabajo se conformara de forma selectiva y desde diversas disciplinas, para que los niveles y la distribución del conocimiento en su interior cubran todos los aspectos de ese dominio (Curtis, Krasner e Iscoe, 1988). Además, la adquisición del conocimiento y el intercambio solamente puede lograrse a través de una comunicación eficaz entre las partes interesadas, porque los problemas de comunicación son un factor importante en el retraso, la calidad y el fracaso de los proyectos software (Curtis, Krasner e Iscoe, 1988). Para esto se puede aprovechar los principios de la Transdisciplinariedad y la Multidimensionalidad, porque su aporte podría permitirle al equipo y a las partes interesadas mejorar los procesos de comunicación y encontrar un lenguaje común, antes de elicitar los requisitos.

A esto se suma el hecho de que los sistemas existen en un entorno organizacional complejo y que, a menudo, el dominio en el que se introducen es intrincado y el equipo no logra entenderlo completamente. Por lo tanto, la elicitación de requisitos y sus descripciones necesariamente son inciertas, además, la documentación de esa especificación puede ser tan amplia que ningún miembro del equipo llega a comprenderla completamente. En esta situación las equivocaciones, las interpretaciones individuales y las opiniones personales prevalecen en la Ingeniería de Requisitos, aunque se haya presentado estudios en la Ingeniería del Software (Curtis, 1990, 1992).

Buglione y Abran (1999) presentan un modelo multidimensional de rendimiento abierto, llamado Quality factor + Economic, Social and Technical dimensions QEST, en el que definen rendimiento como la productividad ajustada por la calidad, que se puede representar desde múltiples puntos de vista, pero que no tiene en cuenta las distintas ópticas disciplinares. El modelo integra tres dimensiones representadas por un valor de medición de la productividad, derivado de un proceso de medición instrumental, que se ajusta mediante una comprobación basada en la percepción de

la calidad del producto alcanzada. Ambos componentes de rendimiento, la productividad y la calidad, tienen en cuenta las mismas tres dimensiones para el cálculo: la económica, la social y la técnica. Estos autores afirman que la medición es cada vez más reconocida como fundamental para evaluar adecuadamente las actuales prácticas del desarrollo de software y para el establecimiento de objetivos realistas en el diseño de programas.

Aunque el objetivo de su trabajo son los productos software, los cuales miden y evalúan a través del modelo tridimensional, esta estructura también les permite a las organizaciones elegir los componentes de cada dimensión de acuerdo con sus propias necesidades, además, les da la posibilidad de seleccionar las medidas pertinentes y ponerlas en práctica. El asunto es que lo enfocan a la Ingeniería del Software como ciclo de vida completo, sin delimitar los alcances para cada una de sus fases. El resultado podría ser una serie de medidas interesantes, pero que no brindan lineamientos para mejorar la gestión de la Ingeniería de Requisitos para alcanzar la fiabilidad del producto. El problema con esta propuesta es que se focaliza en el proceso como un todo, es decir, observa al desarrollo de software como un proceso completo sin tener en cuenta que cada sub-proceso (Ingeniería de Requisitos, diseño, arquitectura, programación, pruebas, implementación, mantenimiento) necesita trabajarse como un componente integral. Por eso es que al final no se logra conformar una solución fiable.

Roelof van Staden (2004) afirma que la industria del software se enfrenta a desafíos específicos, como la calidad de los sistemas, y sostiene que actualmente el grado de satisfacción puede hacer la diferencia entre su permanencia o cierre; por lo tanto, se necesita un enfoque organizado para alcanzarla, es decir, una estrategia de calidad. El autor encuentra y describe cómo identificar y medir la calidad del software, pero afirma que: *Las estrategias organizacionales en estas industrias no se enfocan en gestionar el conocimiento, no tienen una orientación práctica general en la aplicación de los enfoques de calidad y se limitan a aplicar estándares que, en muchas ocasiones, no son adecuados para el tamaño de sus proyectos* (Staden, 2004: 118). También investiga la creación de un marco de procesos de mejora y medición continua, necesario para alinear los objetivos estratégicos y de calidad de una organización que desarrolla software. De acuerdo con los resultados, una vista general de la fiabilidad es el conjunto de rasgos y características de un producto o servicio que satisface sus necesidades específicas o implícitas. Otros autores la definen como la aptitud para el uso o la conformidad con los requisitos (Stephen, 1995; Pressman, 2005).

Por su parte, Gillies (1992) ofrece algunas ideas sobre los problemas de fiabilidad del software en general y afirma que: 1) la calidad del software puede tener muchos significados diferentes en situaciones diferentes; 2) es multidimensional y con muchos factores que contribuyen, siendo difícil resumirlos de manera sencilla y cuantitativa; 3) está sujeta a restricciones tales como el costo y los recursos; y 4) los criterios de calidad dependen generalmente de otros criterios de calidad, que a menudo interactúan entre sí provocando conflictos. Aunque estos autores se refieren a la fiabilidad como parte integral de la Ingeniería del Software, en realidad es algo más que la conformidad con los requisitos de los usuarios: se trata de compromisos y compensaciones para crear sistemas que proporcionen beneficios tangibles a las organizaciones y personas que los utilizan (el nivel superior de la fiabilidad). Por eso es que sus trabajos se limitan a describir y no a proponer modelos que permitan alcanzar realmente esa descripción de calidad del software.

Para Tarr y sus colegas (1999) la arbitrariedad dominante en la descomposición de los sistemas se refiere a mecanismos limitados, utilizados por los métodos tradicionales para descomponer sistemas complejos. Y, aunque los enfoques modernos proponen nuevos mecanismos, en su mayoría utilizan solamente la dimensión dominante como descomposición base, sin tener en cuenta otras relacionadas con el sistema. Por ejemplo, el marco NFR (Chung et al., 2000) utiliza los requisitos no-funcionales como dimensión dominante, mientras que para el enfoque de puntos de vista (Finkelstein y Sommerville, 1996; Sommerville y Sawyer, 1997) y el de casos de uso (Jacobson, 1992), la dimensión dominante es los requisitos funcionales.

Por su parte, Moreira, Araújo y Rashid (2005) proponen un modelo para descomponer los requisitos de manera uniforme, independientemente de su naturaleza funcional o no-funcional. Con esta propuesta es posible proyectar cualquier conjunto particular de requisitos en una gama de otros requisitos, apoyados en una separación multidimensional, de tal manera que las proyecciones subsecuentes y diversas los orientan para que reflejen el interés individual. Este enfoque es compatible con compensaciones transversales y superpuestas de la calidad de la Ingeniería del Software, y a la vez facilita la negociación y la toma de decisiones entre las partes interesadas. Lo que le hace falta es atender la composición del equipo de trabajo con una visión transdisciplinar y dejar de tratar al producto como un todo que se fabrica por etapas. Porque, como lo afirma García (2006: 32-33): *Los sistemas complejos están constituidos por elementos heterogéneos en interacción -y de allí su denominación de complejos-, lo cual significa que sus subsistemas pertenecen a los 'dominios materiales' de muy diversas disciplinas*.

Para Souheil Khaddaj y Gerard Horgan (2005) la calidad del software es un constructo multidimensional que se refleja en un modelo de calidad para la fabricación, en el que cada parámetro define una dimensión de calidad. Por lo tanto, para la producción de software eficaz y fiable es requisito contar con un marco basado en métricas vinculado a un modelo de calidad (Offen y Jeffery, 1997). Muchos de los primeros diseños de modelos de calidad siguieron un enfoque jerárquico, en el que se define un conjunto de factores que afectan la calidad con poco margen para la expansión. Entre ellos se incluye Factor Criteria Metric Model (Khaddaj y Horgan, 2005; Offen y Jeffery, 1997) y Hierarchical Quality Model (Boehm, 1978). Lo que ocurre con estos modelos es que tienen sesgo hacia la mantenibilidad y fiabilidad como factores no-independientes y de poco reconocimiento como requisitos de calidad en diferentes proyectos (Matsumoto y Ohno, 1989; Gillies, 1992; Manns y Coleman, 1996).

Para hacerle frente a estos problemas se ha desarrollado modelos que siguen la propuesta de Finkelstein y Sommerville (1996), en el que se toma una decisión colectiva entre los desarrolladores y los usuarios en cuanto a qué reconocer como atributo de fiabilidad. Debido a su naturaleza disciplinar y a la falta de entender las fuentes multidimensionales de los requisitos, el principal problema con estos enfoques es la falta de directrices generales para producir una visión de consenso de los atributos y la incapacidad de establecer criterios comunes de calidad.

La Ingeniería de Requisitos ha sido definida como la rama de la Ingeniería del Software relacionada con los objetivos del mundo real, en cuanto a las funciones y restricciones de un sistema software (Zave y Jackson, 1997). Pero cuando se pregunta a los profesionales acerca de las causas de los fallos y de la baja fiabilidad de los sistemas, la mayoría las identifican con esta fase del ciclo de vida: mediocridad de los requisitos, especificaciones incorrectas y una gestión inadecuada e ineficaz de las necesidades de las partes (Ibanez, 1996; Chaos, 2000). Por eso es que cuanto más tarde se detecte y corrija un error, más costoso será modificar el producto y su fiabilidad tiende a ser inferior (Boehm, 1976; Daly, 1977; Davis, Jordan y Nakajima, 1997). Esto demuestra que la gestión, validación y corrección temprana de los requisitos puede aliviar muchos de los problemas asociados con el desarrollo del software y con la fiabilidad del producto.

Una de las principales causas de estas dificultades es la misma inconsistencia de los requisitos, que se produce cuando la especificación contiene descripciones conflictivas y contradictorias del comportamiento esperado del sistema que se construye (Ghezzi y Nuseibeh, 1998). Como aporte en la búsqueda de una solución, Gervasi y Zowghi (2005) proponen un método para descubrir automáticamente inconsistencias en los requisitos. Utilizando técnicas de demostración de teoremas y un modelo de comprobación, muestran además cómo tratar con ellos de manera formal, pero se encontraron con que no estaban orientadas a solucionar las inconsistencias de los requisitos y del trabajo entre partes interesadas provenientes de diversas disciplinas. En estos casos la formalización impide una comunicación fluida en el equipo de trabajo, porque no todos los integrantes tienen la formación ni el conocimiento para comprenderla.

Queda claro entonces que la Ingeniería de Requisitos es el proceso más crítico y complejo en el desarrollo de sistemas (Siddiqi, 1996; Juristo, Moreno y Silva, 2002; Komi y Tihinen, 2003), porque como proceso tiene alto impacto en las capacidades del producto resultante. Además, porque en esta fase hay que considerar el conjunto diverso de necesidades proveniente desde las diferentes disciplinas y dimensiones de las partes interesadas, lo que la convierte en un proceso complejo y crítico. En esta fase se realiza una serie de actividades orientadas a descubrir, analizar, documentar, validar y mantener el conjunto de requisitos para el sistema (Sommerville y Sawyer, 1997), mediante un modelo que generalmente concentra estas actividades en dos grandes grupos: de desarrollo y de gestión de requisitos.

El primero se refiere al descubrimiento, el análisis, la documentación y la validación, en tanto que el segundo incluye el mantenimiento, es decir, la identificación, la trazabilidad y la gestión de cambios. En todo el proceso el equipo elicita los requisitos para identificar el funcionamiento del sistema, decide qué funcionalidad implementar y dónde y, mediante interpretaciones, les da significado para tomar decisiones técnicas en cuanto a factibilidad, fiabilidad, calidad y economía del producto a desarrollar (Parviainen, Tihinen y Solingen, 2005), por lo que se requiere un proceso de Ingeniería de Requisitos estructurado y orientado a responder estas demandas. El asunto es que los modelos actuales no están diseñados para atender las exigencias y objetivos de esta fase del ciclo de vida, por lo que la especificación de requisitos es deficiente y, generalmente, no es adecuada para lograr la fiabilidad que esperan las partes (Dobrica y Niemelä, 2002).

De acuerdo con Camp (2006) en el pasado la tecnología impulsó cambios en los procesos de diseño y desarrollo de sistemas, pero, actualmente, los productos software son cada vez más complejos y su solución implica tener en cuenta y gestionar múltiples puntos de vista. Debido a esto para el diseño y el desarrollo de sistemas se debe integrar conocimientos desde diversas disciplinas y, a medida que el ritmo del desarrollo de nuevos sistemas se acelera, el trabajo de los equipos pasa de ser interdisciplinar o multidisciplinar a trans-organizacional e incluso, a trans-nacional y trans-continental (Tate et al., 2006).

En la Ingeniería de Requisitos los enfoques multidisciplinares utilizan métodos de dos o más disciplinas para examinar las necesidades del sistema, y es común que los integrantes del equipo trabajen de forma independiente dentro de su propia disciplina. Los equipos interdisciplinares aplican un método único desde cierta disciplina para intentar comprender los requisitos que, en muchas ocasiones, se deberían estudiar en otras disciplinas. Ambas actividades, multidisciplinares e interdisciplinares, cruzan las fronteras de las disciplinas, pero no son suficientes para satisfacer los desafíos que plantean los problemas actuales (Tate et al., 2006). Debido a que los enfoques transdisciplinares incluyen una mayor cooperación e integración entre las disciplinas, son más convenientes para realizar la fase de la Ingeniería de Requisitos, porque comparten métodos, conocimiento y temáticas entre las disciplinas.

Pero a pesar de que el trabajo ingenieril transdisciplinar se referencia desde la década de 1970, los métodos para especificar requisitos todavía no lo aplican ampliamente, lo que genera problemas de costos, tiempos y baja fiabilidad en los productos software (Do Ru y Dogru, 2011). La Ingeniería de Requisitos transdisciplinar presenta varias ventajas sobre la disciplinar: se comparte la experiencia desde múltiples disciplinas y, para la fiabilidad del producto, es útil analizar los requisitos desde diversas perspectivas; los recursos son menos propensos a ser desperdiciados y la duplicación de esfuerzos es mínima, porque el equipo dialoga y analiza cada requisito con la interpretación propia de cada disciplina, lo que al final genera un documento de especificación de requisitos fiable (Häberli et al., 2001; Carew, 2006).

Romero y Abelló (2006) proponen un método para validar requisitos multidimensionales. Su propuesta consiste en descomponer una consulta de entrada para inferir el potencial conocimiento multidimensional implícito y explícito relevante que contiene el requisito y, en

consecuencia, generar automáticamente el conjunto de esquemas multidimensionales que debe satisfacer. De esta manera se identifica los hechos, las dimensiones y las jerarquías dimensionales que dan apoyo al proceso de la elicitación de requisitos. Los autores concluyen que, si la consulta no genera un esquema multidimensional, se puede afirmar que el requisito tampoco lo es. El aporte más importante de este trabajo es una manera para automatizar la identificación de los conceptos multidimensionales en los requisitos, en relación con las necesidades del usuario final. Pero presenta la falencia de que solamente identifica este tipo de requisitos y no tiene en cuenta los demás, aunque necesariamente se tienen que gestionar en la Ingeniería de Requisitos. Esta limitante no le ha permito al modelo una mayor aceptación en la comunidad.

Jay Xiong (2011) afirma que la mayoría de los productos software de hoy todavía se desarrollan con el tradicional paradigma de la Ingeniería del Software, lo que genera problemas críticos tales como baja productividad y calidad, además, que se incrementan los costos y la posibilidad de riesgo es mayor. Igualmente, expone que, aunque se ha propuesto diversos modelos de procesos para mejorar el paradigma y superar estos problemas, están basados en las mejores prácticas, con el inconveniente de que este concepto todavía no tiene un significado propio y ampliamente aceptado (Serna, 2015). Por lo tanto, surge el interrogante de si en el desarrollo de software con el paradigma existente realmente se aplican las *mejores prácticas* y, si es así, qué se entiende como como tal (Ambler, Nalbone y Vizdos, 2004).

En este sentido llama la atención que esas buenas prácticas no se mencionen específicamente para el desarrollo y la gestión de la Ingeniería de Requisitos. Porque sería agradable desarrollar un producto de forma iterativa debido a que en la mayoría de casos los requisitos no se conocen por completo desde el principio, y los clientes necesitan revisar el resultado con anticipación (Kimball, Weinstein y Silber, 2004). Pero esto no es posible con los modelos analizados, porque son lineales y totalitarios. Por eso es que la mayoría se orienta a la Ingeniería del Software como proceso y no a las fases como componentes. Esto hace que todavía y con el paradigma actual no se pueda solucionar los problemas críticos de la Ingeniería del Software, especialmente el de la fiabilidad de su producto.

La gestión de requisitos es un proceso sistemático de descomposición y trazado a través de múltiples niveles de abstracción (Zhou y Leung, 2006); las estructuras resultantes son grafos dirigidos en los que las trayectorias de descomposición divergen y convergen a medida que los requisitos son re-factorizados a través de las capas de diseño; y en el desarrollo de sistemas complejos es común que diferentes equipos aborden cada una de sus capas, lo mismo que la descomposición de requisitos asociada. Pero, debido a la complejidad de la cadena de descomposición que surge de cada requisito multidimensional, se requiere que esos equipos sean transdisciplinares, porque los niveles de abstracción pueden no ser evidentes para cada integrante individual (Serrano, Calero y Piattini, 2003), y no sería posible satisfacer la calidad del producto.

Por lo tanto, se requiere trabajo transdisciplinar para poner de relieve la complejidad del problema y, para responder a esto, Hull, Jackson y Dick (2011) recomiendan el uso de estadísticas de flujo descendente para identificar los requisitos cuya descomposición sea significativamente diferente de los demás, para luego aplicarles una consideración especial durante las revisiones. Dick y Jones (2012) amplían esta sugerencia y proponen el uso de una serie de medidas de la complejidad como medio para evaluarla en ese flujo descendente. Aunque el modelo propuesto por estos investigadores aborda la multidimensionalidad de los requisitos y la necesidad de equipos transdisciplinares en la Ingeniería de Requisitos, el hecho de tratar la abstracción de los requisitos como capas acumulativas genera más complejidad al proceso, porque en muchas ocasiones el equipo de trabajo tiene que aplicar re-ingeniería para llegar a acuerdos en las interpretaciones disciplinares en cada capa, lo que genera pérdida de tiempo e incrementa los costos del desarrollo de esta fase del ciclo de vida.

Dubey, Ghosh y Rana (2012) presentan una comparación de varios modelos de calidad del software y llegan a la conclusión de que, para mejorarla, no importa qué modelo se aplique en la Ingeniería del Software, porque mientras no se atienda adecuadamente la fase de la Ingeniería de Requisitos este desarrollo tecnológico seguirá con falencias. Para estos investigadores la calidad del software tiene un contenido multidimensional, que se puede distinguir y medir fácilmente, pero los modelos que intentan determinarlo se orientan a diferentes aspectos de este asunto y no realizan un proceso adecuado de investigación.

Como resultado utilizan expresiones diversas para la comprensión de contenidos básicos que son difíciles de aplicar y de seguir, y aunque se ha hecho intentos para introducir modelos para el análisis de las características cualitativas y cuantitativas de la calidad del software, son muy generales y tratan el proceso del desarrollo como una fase única, sin tener en cuenta que es la suma de diversas fases con objetivos específicos. La fiabilidad del software es el grado en que se le incorpora un conjunto de características deseables a fin de mejorar su rendimiento de por vida (Firesmith, 2003), pero para lograrlo hay que tener en cuenta que cualquier sistema software debe respetar tres especificaciones: 1) funcional (*qué hace el sistema*), 2) de calidad (*qué también lo hace*), y 3) de recursos (*qué se requiere para lograrlo*). Además, el equipo de trabajo debe abordarlas desde la Ingeniería de Requisitos, porque es la fase en la que se traza las bases para que el producto alcance la fiabilidad deseada.

Para Maadawy y Salah (2012) la complejidad es un tema importante para la industria del software, porque afecta la planificación y el costo estimado del proyecto; además, porque genera inconvenientes en productividad y en el logro de la fiabilidad esperada. Por eso es difícil desarrollar un sistema que no necesite cambios, sea en el proceso o luego de ser liberado, por lo que medir la complejidad es importante desde el inicio del proyecto, es decir, desde la Ingeniería de Requisitos. Si esto no se hace adecuadamente el costo relativo para el mantenimiento y la gestión se incrementa continuamente, al punto de que actualmente se estima que representa más del 90% del esfuerzo total (Erlikh, 2000).

Por su parte, Maadawy y Salah (2012) afirman que gestionar requisitos es difícil, no solo porque significa invertir tiempo, sino porque un cambio en cualquiera de ellos puede tener un impacto directo o indirecto sobre los otros. Para gestionar de mejor forma el proceso de los requisitos, estos autores recomiendan integrar equipos transdisciplinares: *porque su visión es más amplia y las interpretaciones se pueden discutir con mejores argumentos.* Lo que no proponen es cómo hacerlo, ni estructuran un modelo que permita gestionar una Ingeniería de Requisitos orientada a salvar su complejidad y a alcanzar la calidad esperada del producto.

Debido a que esta fase depende en cierta medida de la plataforma de despliegue[1], de la elección arquitectónica y de la experiencia y experticia del equipo de trabajo, se debe atender con máxima prioridad e importancia (Gal, 2013). Además, porque su producto, la especificación de requisitos, es la base para la elección del lenguaje de programación y de las herramientas para las demás fases del ciclo de vida. Por eso es necesario desarrollarla por medio de un modelo estructurado y de una metodología transdisciplinar y multidimensional, para clasificar y agrupar los requisitos de aplicación y orientarlos al logro de la fiabilidad presupuestada para el producto. De esta manera se puede: 1) asegurar que los requisitos se ajustan específicamente a la descripción de las características y necesidades de las partes interesadas, 2) asignar los requisitos a un método de implementación adecuado y a cierto tipo de desarrollo, y 3) estructurar un documento de especificación de requisitos apropiado (Moreira, Rashid y Arajo, 2005). En este sentido, Gal (2013) propone un enfoque basado en la separación multidimensional de las necesidades del sistema para guiar el proceso de especificación, pero no describe la forma para lograrlo, ni qué hacer cuando el equipo de trabajo está conformado por actores de una sola disciplina.

[1] Es decir, el Sistema de Información sobre el que se implementará el producto.

Klaus Pohl y Chris Rupp (2015) sostienen que el objetivo de la Ingeniería de Requisitos es documentar las necesidades de las partes de la mejor manera, pensando en alcanzar la fiabilidad establecida y en identificar y resolver los problemas tan pronto como sea posible. Para ser exitosa debe ser incluyente, es decir, tener en cuenta a todos los actores y desarrollar adecuadamente las etapas básicas: elicitación, desarrollo, gestión y especificación (Serna y Serna, 2015). En el centro del proceso se encuentra el equipo de trabajo, que a la vez es el enlace entre las partes interesadas y el equipo que se encargará del diseño y el desarrollo, por lo que debe poseer un buen conocimiento del dominio, de la organización, de los procesos y los sistemas involucrados.

Pohl y Rupp (2015) declaran que, actualmente, los requisitos se especifican utilizando el lenguaje natural y que esto conlleva a interpretaciones erróneas por parte del equipo de trabajo. Sin embargo, y aunque se han propuesto diversos modelos orientados a alcanzar la fiabilidad del producto, el equipo todavía es el responsable de materializarlos en el documento de especificación. Pero dado que generalmente los integrantes provienen de una misma disciplina, esto puede provocar que las interpretaciones de los requisitos no se refinen desde diferentes puntos de vista y que la gestión no sea la adecuada para el problema. Además, dado que la calidad del producto suele estar relacionada con esa interpretación y que el objetivo es desarrollar productos software de buena de calidad, es necesario conformar equipos transdisciplinares (Batorowicz y Shepherd, 2008; King et al., 2009; SPAAL, 2009) para llevar a cabo la Ingeniería de Requisitos y diseñar modelos que permitan una gestión orientada a analizarlos desde la Multidimensionalidad.

En relación con la Ingeniería de Requisitos Bohem (1981) y Davis (1990) reportan que es la fase del desarrollo de software más propensa a errores de todo tipo y que, en gran medida, la fiabilidad del producto final depende de una gestión eficiente de la misma. Por eso diversos autores han presentado propuestas de solución integrando principios y prácticas de diversas disciplinas, con resultados que todavía no convencen completamente a la industria y a la academia. Entre ellas se puede citar a Sacks (1992), quien intenta involucrar a la sociología para sincronizar, superponer y reparar los diálogos entre las partes. Algunos utilizan el modelado de procesos (Ambriola y Montagnero, 1992), la especificación orientada a objetos (Fernström, Närfelt y Ohlsson, 1992) o la arquitectura del software (Perry y Wolf, 1992), como modelos teóricos de gestión.

Tomando como base esos aportes, otros autores han intentado estructurar modelos de gestión de requisitos, por ejemplo, Jacobson (1992), Finkelstein y Sommerville (1996) y Chung et al. (2000) se concentran en aprovechar los tipos de requisitos (funcionales y no-funcionales) para presentar ideas de solución. Goguen (1994) utiliza la etnometodología y el análisis del discurso como medios de interacción social, con el objetivo de encontrar una teoría social de la información. Roelof van Staden (2004) llama la atención sobre la necesidad que tiene la industria de mejorar la fiabilidad y seguridad de los sistemas y presenta un modelo para medirla y seguirla, aunque poco se ha utilizado en la industria. Mientras que Moreira, Araújo y Rashid (2005) proponen un modelo para descomponer cualquiera de esos tipos de requisitos y Gervasi y Zowghi (2005) un método para descubrir inconsistencias en ellos. Por su parte, Zhou y Leung (2006) sostienen que la gestión de requisitos es un proceso sistemático de descomposición, trazado a través de múltiples niveles de abstracción, por lo que es extremadamente difícil (Maadawy y Salah, 2012), pero que se debe atender con máxima prioridad e importancia (Gal, 2013). Mientras tanto, Pohl y Rupp (2015) declaran que los requisitos se especifican en lenguaje natural y que eso genera ambigüedades que incrementan la complejidad de comprenderlos.

En relación con la perspectiva del Pensamiento Complejo, Gillies (1992) opina que la fiabilidad del software tiene diversos significados, que es multidimensional y que eso genera conflictos difíciles de resolver. Walz, Elam y Curtis (1993) sostienen que el problema se reduce a la conformación de los equipos de trabajo, porque deberían estar integrados desde diversas disciplinas, de tal forma que la Ingeniería de Requisitos se lleve a cabo de forma transdisciplinar. Por su parte, Al-Rawas y

Easterbrook (1996) concluyen que la fiabilidad del software depende del nivel de conocimiento acerca del dominio en las primeras fases de desarrollo, lo que exige trabajo multidimensional y transdisciplinar.

Zamperoni (1996) parte de la complejidad de los requisitos para desarrollar un modelo formal multidimensional, integrando otras áreas involucradas en la Ingeniería de Requisitos, pero trata de gestionar el software como un producto completo, ignorando la complejidad propia de cada una de las fases de su desarrollo. En este mismo sentido el modelo multidimensional de rendimiento abierto, de Buglione y Abran (1999), define que la fiabilidad es una cuestión que se puede representar desde múltiples puntos de vista, pero no tiene en cuenta las distintas ópticas disciplinares. Komi y Tihinen (2003) afirman que gestionar requisitos es un proceso crítico y complejo, porque el resultado impacta la fiabilidad del producto, y para Serrano, Calero y Piattini (2003) se requiere un trabajo transdisciplinar para poner de relieve esa complejidad.

Dick y Jones (2012) proponen utilizar una serie de medidas de la complejidad para evaluar la fiabilidad desde las primeras fases del desarrollo, y Souheil Khaddaj y Gerard Horgan (2005) reportan que la fiabilidad del software es una construcción multidimensional en todo el proceso del desarrollo. Por su parte, Camp (2006) sostiene que todo lo relacionado con el software es complejo por lo que involucra múltiples puntos de vista. Tate et al. (2006) demuestran que el trabajo multidisciplinar en la Ingeniería de Requisitos es una práctica que debe migrar al transdisciplinar, porque de otra manera no se podrá superar la complejidad de esta fase. Esta afirmación también la comparten Carew (2006) y Do Ru y Dogru (2011).

Romero y Abelló (2006) proponen un método para validar requisitos de usuario multidimensionales mediante descomposición y composición; para Firesmith (2003) y Dubey, Ghosh y Rana (2012) la fiabilidad tiene un contenido multidimensional que se puede distinguir y medir fácilmente, y para Maadawy y Salah (2012) es una cuestión importante que se debe hacer desde el inicio del proyecto. En respuesta a esto, Gal (2013) propone la separación multidimensional de las necesidades del sistema para guiar el proceso de especificación.

Como se observa en este estado del arte y como se afirma en el análisis de los trabajos encontrados, la Ingeniería de Requisitos es una fase que involucra actividades desde diversas disciplinas y dimensiones, pero los modelos actuales para gestionarla no tienen en cuenta principios del Pensamiento Complejo, tales como la Transdisciplinariedad y la Multidimensionalidad, que podrían ayudar a gestionarla de mejor forma. Los modelos propuestos y analizados tienen falencias que se reflejan en una baja fiabilidad del producto, en parte porque observan el desarrollo de software como un proceso completo. Esto hace que no pongan atención al hecho de que las distintas etapas inmersas en él tienen sus propias actividades, que de una u otra manera también son complejas.

El estado del arte presentado reafirma la presunción de esta investigación, en el sentido de que tal como se gestiona actualmente la Ingeniería de Requisitos no es posible mejorar la fiabilidad del software como producto de desarrollo. Además, que integrar en sus procesos principios del Pensamiento Complejo, tales como Transdisciplinariedad y Multidimensionalidad, que han sido propuestos por diversos autores, podría ser una alternativa para mejorarla.

V
DISEÑO METODOLÓGICO

Una metodología es el marco y la razón de ser en torno a los que se estructura un proyecto de investigación y que, entre otras cosas, determina la relación entre el marco teórico y la recolección de datos y su posterior análisis (Kleiber, 2002). La metodología contiene información acerca de cómo se elige los métodos y cómo se analiza los datos, además, sustenta cada aspecto del proceso investigativo. Para esta investigación la selección del enfoque se determinó a partir de: 1) los puntos de vista acerca de la naturaleza de la realidad y del ser (ontología), y 2) lo que cuenta como conocimiento (epistemología). Porter (1996) describe este tipo de investigación como fundamentada en cuatro niveles de comprensión:

1. Ontología: ¿cuál es la naturaleza de la realidad?
2. Epistemología: ¿qué cuenta como conocimiento del mundo real?
3. Metodología: ¿cómo obtener una comprensión de la realidad?
4. Métodos: ¿cómo reunir pruebas acerca de la realidad?

La Ontología es el estudio del ser y tiene que ver con la pregunta ¿cuál es la naturaleza de su existencia y realidad? Tiene fuertes vínculos con la Epistemología y, en ocasiones, sus límites pueden llegar a ser borrosos. Sin embargo, en aras de clarificar su comprensión, en esta investigación se tratan como diferentes. Para Stainton (2003) Ontología se refiere a la naturaleza de lo que son las cosas y su razón de estar en el mundo, en qué consisten, qué entidades operan dentro de ellas y cómo se relacionan entre sí. Blaikie (2000) añade que los supuestos ontológicos tienen que ver con lo que se cree que constituye la realidad, es decir, las afirmaciones acerca de lo que existe, lo que se ve, lo que se hace y cómo se relacionan entre sí sus componentes. Hay (2002) afirma que la posición ontológica del científico depende de la naturaleza de la realidad que desea investigar. Desde la perspectiva social una posición ontológica implica que los fenómenos son hechos externos del alcance o la influencia de las personas (Bryman, 2008), es decir, todos los fenómenos y categorías sociales existen por fuera del control de las personas y sus acciones, por lo que se encuentran separadas de cualquier significado: *la verdad está ahí a pesar de todo*.

Una forma alternativa de ver la naturaleza del ser es a través de la posición constructivista, que sostiene que no existe una realidad objetiva sino modelos de la misma y que el conocimiento no se descubre, sino que se construye. Mientras que en el objetivismo el investigador busca cosas que aún no han sido descubiertas, este paradigma argumenta que simplemente encontrar cosas no es suficiente para explicar lo que hay en el mundo; no niega la existencia del mundo real, pero cuestiona la idea de que todo lo que las personas deben hacer es quitarse el velo de la ignorancia con el fin de revelar datos sobre cómo es el mundo realmente (Stainton, 2003). Desde esta perspectiva ontológica los significados son construidos por personas que participan del mundo (Crotty, 2003) y que, para efectos de la investigación, no se considera al mundo como un lugar donde las cosas siempre han estado allí y se pueden medir a través de la observación o la experimentación, sino como un lugar donde se influencia la construcción de la realidad y se necesita interpretar lo que se va a construir. Por tal razón hay que describir la visión de ese mundo antes de comenzar a pensar cómo influenciar e interpretar su realidad.

Por otro lado, puede ser difícil separar al ser del saber, por lo que la Epistemología se refiere a la forma en que las personas conocen las cosas y a lo que cuenta como conocimiento (Crotty, 2003).

En otras palabras: ¿qué saben? y ¿cómo saben que lo saben? Pero, al igual que con los debates sobre la Ontología, existe más de un punto de vista epistemológico. Una de las formas en que se podría tratar de determinar qué tradición epistemológica seleccionar es considerar si el mundo social puede ser estudiado exactamente de la misma manera que se estudia el de las Ciencias Naturales (Bryman, 2008). Si la respuesta es *sí*, se podría decir que se toma una posición *positivista*, que no es diferente de la forma objetivista-realista de ver la cuestión y que confirma lo difícil que puede ser separar la Ontología de la Epistemología. El positivismo se basa en una serie de principios fundamentales (Bryman, 2008):

1. El conocimiento tiene que ser adquirido a través de y confirmado por los sentidos.
2. La investigación debe generar hipótesis que se puedan probar.
3. El conocimiento se construye a través de la reunión de hechos que hagan posibles leyes para establecer y realizar predicciones.
4. La investigación debe llevarse a cabo de manera objetiva.

Los resultados de este enfoque estructuran una posición en la que los investigadores tratan de explicar el comportamiento humano a través de la búsqueda de las fuerzas que actúan sobre las personas, y de la identificación de las causas de sus acciones. Por eso es fácil ver por qué un enfoque positivista puede ser sensible para todas las investigaciones, porque parece ofrecer explicaciones *científicas* claras y confiables. Sin embargo, las cosas pueden no ser tan sencillas como parece la primera vez y se ha sugerido que puede haber más de un enfoque positivista (Crotty, 2003). Este enfoque de conocimiento implica que cada investigador que observa algo siempre lo verá de la misma manera, aunque no sea el caso. Si las cosas fueran tan simples y hubiera leyes universales que rigen las situaciones sociales, entonces las personas no tendrían que discutir sobre la importancia de los acontecimientos, pero los seres humanos claramente argumentan y discrepan sobre todo tipo de situaciones. El enfoque positivista puede incluso encontrar dificultades en relación con las Ciencias Naturales, porque lo que una persona ve como bien, puede ser muy diferente de lo que otra ve, porque están influenciadas por sus propias ideas, motivaciones y visión del mundo.

Este tipo de situaciones se puede ver como *interpretativismo*, porque describe acercamientos al conocimiento que obliga a entender lo que significan determinadas acciones. Inevitablemente, y con el fin de escribir sobre ellas, los investigadores tienen que anteponer sus propios significados frente a esas acciones. De hecho, aquí es donde el término interpretativismo encaja, porque ellos ven la acción humana como algo significativo y hacen hincapié en su contribución al conocimiento, sin sacrificar su objetividad. Además, se argumenta que es posible entender el significado subjetivo de las acciones de manera objetiva (Schwandt, 2001). Aunque tal vez el uso del término *objetivo* sea engañoso, porque podría sugerir una creencia positivista en las predicciones científicas, y claramente el interpretativismo no hace grandes afirmaciones sobre la causalidad (Thomas, 2009). En su lugar, sería más apropiado hablar de justicia, honradez y rigor.

En cualquier caso, la metodología es un plan de acción (Crotty, 2003) y una visión del mundo y de la naturaleza del conocimiento, por lo tanto, debe influir en las decisiones acerca de cómo responder las preguntas que plantea la investigación. Este punto de vista ayuda a determinar cuál enfoque metodológico es el más apropiado (Crotty, 2003; Creswell, 2007), dependiendo de si lo que se busca es describir, explicar, cuantificar y/o cambiar algo. En este sentido, la metodología seleccionada se arraiga en los puntos de vista epistemológico y ontológico para establecer: 1) qué métodos de investigación elegir, 2) cómo hacerlo, 3) cómo utilizarlos, y 4) de qué forma analizar los datos que se produzcan.

Para llevarla a cabo se ha propuesto muchos y diferentes modelos, la mayoría subdivididos en una serie de etapas. Por ejemplo, Cohen y Manion (1994) identifican ocho etapas en la investigación-

acción que parecen demasiado científicas para todos los planteamientos: hipótesis, diseño de experimentos, correlaciones observadas, hipótesis formadas para explicar las regularidades, explicaciones y predicciones probadas, leyes desarrolladas o hipótesis negadas, generalizaciones hechas y nuevas teorías. Blaxter et al. (1999) lo dividen en cinco etapas que parecen ser una simplificación excesiva de un proceso que es largo y complicado: leer para investigar, organizar el proyecto, recolectar datos, analizar datos y redactar y concluir el trabajo. Por su parte, Johnson (1994) identifica 11 etapas de actividad que, para la mayoría de las investigaciones, es extensa y demasiado lineales: pensar, redactar, buscar, identificar, contrastar, analizar, proyectar, aplicar, validar, proponer y publicar.

Una cuestión que hay que reconocer es que el proceso de la investigación es todo menos lineal, porque aferrarse a un esquema es como aceptar que la ciencia no es dinámica, y que cada representación solamente es una simplificación o idealización del mismo. Sin embargo, para el proceso de esta investigación se aplica un enfoque estructurado a partir de ellas (Cohen y Manion, 1994; Blaxter et al., 1999; Johnson, 1994), pero complementado con las necesidades específicas planteadas para la misma, porque lo que se busca es comprender percepciones y realidades particulares del mundo (Bell, 1999) para articularlas en procesos ingenieriles. En este caso específico hay que aclarar: 1) el tema de investigación es acerca de la integración de principios del Pensamiento Complejo en la gestión de la Ingeniería de Requisitos, 2) la atención se centra sobre todo en los aportes que esta área del conocimiento puede hacer para mejorar la fiabilidad de los productos software, y 3) el interés es descubrir principios que se puedan interrelacionar lógicamente con los objetivos de las etapas de la Ingeniería de Requisitos.

Considerando estas características y teniendo en cuenta el análisis y la evaluación que se necesita para lograr los objetivos y responder las preguntas de investigación, la conclusión es que la metodología más apropiada consiste en aplicar una estrategia de carácter *cualitativo*, pero construyendo un diálogo horizontal entre lo social, la Ingeniería de Requisitos y el Pensamiento Complejo. Los resultados sustentarán la propuesta en datos concretos que derivarán de los principios seleccionados de las áreas objeto, a la vez que del análisis de diferentes formas para identificar las interrelaciones lógicas que integran el modelo propuesto. En este sentido, el término metodología se refiere a la forma como se enfocará el problema y se le buscará solución.

Los intereses, supuestos y propósitos de la investigación son la base para elegir una u otra metodología (Taylor y Bogdan, 1998), y Sabariego y Bisquerra (2004) afirman que las aproximaciones a la investigación cualitativa van desde la lógica de la distinción (solamente una de las metodologías es válida) hasta la lógica de la convivencia (valora igualmente las posibilidades de cada metodología respetando sus aportaciones). Para la presente investigación se asume que estos criterios no son excluyentes, porque en realidad no existen métodos puros, sino que presentan características compartidas por lo que se utilizará una metodología mixta en todas las etapas del proceso (Tashakkori y Teddlie, 2003). La razón es que los métodos analizados no son suficientes por sí mismos para captar las tendencias, los detalles y los principios buscados, mientras que una combinación permite complementar los datos entre las fases y sería posible lograr análisis más completos (Green, Caracelli, y Graham, 1989; Tashakkori y Teddlie, 1998).

Para definir esta metodología se sigue los elementos que Massot, Dorio y Sabariego (2004) consideran determinantes en un proceso de investigación cualitativa, y que se sintetizan en la elección de técnicas y estrategias para recolectar, analizar y estructurar datos que sean adecuadas para las características de la investigación, la temática a investigar y el contexto físico y temporal en el que se desarrolla. Todo esto teniendo en cuenta que el enfoque cualitativo se centra en el estudio de realidades, subjetivas e intersubjetivas, como objetos legítimos de conocimiento en las que se reconoce que la realidad es un proceso en construcción, y que los actores involucrados en la producción y comprensión tienen su propia perspectiva de las mismas. Por otra parte, la investigación cualitativa es un proceso de indagación por la comprensión, donde el investigador

desarrolla una imagen global compleja, analiza contenidos, estructura vistas detalladas de los datos y lleva a cabo el estudio en un entorno natural-real (Creswell, 2007). En este enfoque el investigador hace afirmaciones de conocimiento con base en la perspectiva constructivista (Guba y Lincoln, 1982) o en la participativa (Mertens, 2003). Para esta mezcla de enfoques los datos se encuentran inmersos en la vida cotidiana del entorno en el que se enmarca el estudio, el análisis de datos se basa en los valores que se perciben del mundo y, en última instancia, se obtiene una comprensión del problema con base en múltiples factores contextuales (Miller, 2000).

Es resumen, seleccionar un enfoque metodológico para una investigación puede ser una tarea simple o complicada. Una metodología es un plan de acción (Crotty, 2003) y una visión del mundo y de la naturaleza del conocimiento, por lo tanto, debe influir en las decisiones acerca de cómo responder las preguntas planteadas en la investigación. En este caso se estudiaron algunos modelos metodológicos: Cohen y Manion (1994), Johnson (1994), Blaxter et al. (1999), y se tomaron algunos de sus principios. Luego se complementaron con las necesidades específicas planteadas, porque lo que se busca es comprender percepciones y realidades particulares del mundo (Bell, 1999) para articularlas en procesos ingenieriles.

Además, desde la perspectiva social se toma una posición ontológica porque los fenómenos que involucra la Ingeniería de Requisitos son hechos externos al alcance o la influencia de las personas (Bryman, 2008), pero sus efectos tienen influencia directa en ellas. Por otro lado, no se considera al mundo como un lugar donde las cosas siempre han estado allí, para medirlas a través de la observación o la experimentación, sino un lugar donde se va a influenciar la construcción de la realidad, por lo que se necesita interpretar lo que se va a construir. Por tal razón, hay que describir la visión de ese mundo antes de comenzar a pensar cómo influenciar e interpretar su realidad, en este caso, a través de la especificación de requisitos. Para eso es necesario responder interrogantes tales como *qué saben las personas* y *cómo saben que lo saben*, porque ese conocimiento debe servir como insumo para estructurar los resultados de la investigación.

Considerando estas características y teniendo en cuenta el análisis y la evaluación necesarias para alcanzar los objetivos y responder las preguntas de investigación, se estructura una metodología de carácter *cualitativo*, construyendo un diálogo horizontal entre las ciencias sociales, la Ingeniería de Requisitos y el Pensamiento Complejo. En este sentido, el término metodología se refiere a la forma cómo se enfoca el problema y cómo se busca una solución. También se centra en el estudio de realidades (subjetivas e intersubjetivas) como objetos legítimos de conocimiento, en las que se reconoce que la realidad es un proceso en construcción y que los actores involucrados tienen su propia perspectiva de la misma. Por otra parte, la investigación cualitativa es un proceso de indagación por la comprensión, donde el investigador desarrolla una imagen global compleja del problema, analiza contenidos, estructura vistas detalladas de los datos y lleva a cabo el estudio en un entorno natural-real (Creswell, 2007).

VI
RESULTADOS

El contenido de este capítulo es el resultado de aplicar la metodología seleccionada y de comparar y analizar cómo integrar los principios del Pensamiento Complejo con los objetivos de las etapas de la Ingeniería de Requisitos. A partir del estado del arte y el marco teórico se tomaron algunos autores representativos de la *Complejidad* (Nicolescu, 1996, 2006, 2008; Edmonds, 1999; Benbya y McKelvey, 2006; Douglas y Kuras, 2006; Batorowicz y Shepherd, 2008; Morin, 2008; Douglas, 2009; Hofkirchner, 2009; Cooke, 2011; Jörg, 2011; Lemberger y Morel, 2012; Maadawy y Salah, 2012; Mahaux y Canon, 2012), del *Pensamiento Complejo* (Checkland, 1981; Morin, 1994; Gibbons y Nowotny, 2002; Häberli y Thompson, 2002; Häberli et al., 2001; Lawrence y Després, 2004; Hunsinger, 2005; Moreira, Rashid y Arajo, 2005; Tate et al., 2006; Hass, 2009), de la *Ingeniería de Requisitos* (Boehm et al., 1978; Pohl, 1992, 1994; Al-Rawas y Easterbrook, 1996; Cardoso, 1998; Buglione y Abran, 1999; Annoni, et al. 2006; Camp, 2006; Carew, 2006; Castillo et al., 2010; Dick y Jones, 2012; Serna y Serna, 2015), y de la *Gestión del Conocimiento* (Loucopoulos y Champion, 1989; Guindon, 1989; Ramesh, 1993; Davenport y Prusak, 1998; Brown y Duguid, 1998; Guindon, 1990; Cysneiros et al., 2005; Caseau, Krob y Peyronnet, 2007; Serna, 2015;). Luego, y sin perder de vista el objetivo de mejorar la fiabilidad del software, se hizo un análisis de sus aportes para encontrar los principios que se podrían integrar y complementar para diseñar el modelo buscado.

1. PRINCIPIOS DEL PENSAMIENTO COMPLEJO PARA GESTIONAR LA INGENIERÍA DE REQUISITOS

La investigación cualitativa consiste en recoger, analizar e interpretar datos que no se reducen fácilmente a números, porque se refieren al mundo social y a los conceptos y comportamientos de las personas que lo integran. Esas interacciones humanas son complejas y rara vez se pueden estudiar o explicar en términos simples, por lo que estas situaciones exigen comprensión compleja y el reconocimiento de que los datos se entienden en relación con su contexto de producción. Para esta investigación se aplica un proceso deductivo a partir de las premisas aportadas por los autores analizados. Además, se tiene en cuenta los temas emergentes que plantean y a los que, en el marco teórico y el estado del arte, se les presentó el análisis respectivo.

Los métodos tradicionales para gestionar la Ingeniería de Requisitos se centran en atender factores como requisitos funcionales y no-funcionales, aislando problemas tales como la fiabilidad del software. Por eso se necesita nuevas formas de gestionarla como un sub-sistema complejo que aporta a la comprensión del funcionamiento del sistema-solución en su conjunto. Por otro lado, la lógica subyacente en la ingeniería es que para determinar el efecto de la especificación de requisitos sobre el sistema se requiere la comprensión de su funcionamiento completo. Por lo tanto, del efecto de la Ingeniería de Requisitos depende los estados de las demás fases, porque se verán afectadas por los procesos de retroalimentación y por las dependencias de los requisitos. Estas relaciones incluyen interacciones entre individuos y entre los individuos, el contexto y el dominio a través del tiempo. Además, entre los procesos sociales y los tecnológicos esenciales para distinguir los factores que se verán afectados por el sistema.

Debido a que los problemas han aumentado en complejidad con el tiempo, solucionarlos requiere un marco más amplio del que tradicionalmente se ha utilizado. En estos sistemas complejos se reconoce explícitamente la presencia de múltiples niveles, ciclos de retroalimentación y

dependencias que dan lugar a requisitos multidimensionales provenientes de diversas disciplinas. Por eso es que una comprensión más profunda de estas relaciones dinámicas en el sistema puede ayudar a anticipar y monitorear la efectividad y la eficiencia de una solución específica en el tiempo. La noción de Pensamiento Complejo de Edgar Morin se refiere a la *capacidad de interconectar diferentes dimensiones de la realidad*, precisamente la característica que se necesita en la gestión de la Ingeniería de Requisitos, porque en ella se conjuga eventos y objetos interactivos y multidimensionales con componentes aleatorios y, para comprenderlos, es necesario desarrollar estrategias reflexivas de pensamiento.

Como concepto que se opone a la división disciplinar y que promueve un enfoque transdisciplinar y holístico, sin abandonar el concepto de los elementos constitutivos de la totalidad, el Pensamiento Complejo es una herramienta poderosa cuyos principios se pueden incorporar en la gestión de la Ingeniería de Requisitos. Además, porque está soportado en las teorías de Sistemas, de la Cibernética y de la Información, que en conjunto son conceptos ingenieriles necesarios en el desarrollo de software. Por lo tanto, el Pensamiento Complejo es una estrategia, a la vez que una forma de pensar, que se propone para interpretar fenómenos globales, pero que, al mismo tiempo, reconoce la especificidad de sus partes constitutivas. Esta cuestión se necesita en la Ingeniería de Requisitos, porque en esta fase se dan los primeros pasos para comprender el fenómeno y para modelar una solución. Otro asunto es que el Pensamiento Complejo se relaciona con la Epistemología, por lo que en esta investigación se toma como objeto de estudio del conocimiento en pro de producirlo y validarlo a través del análisis de los diferentes criterios.

El objetivo es encontrar los principios del Pensamiento Complejo que se pueden integrar a la Ingeniería de Requisitos para ayudar a mejorar la fiabilidad del software y, luego de analizar y confrontar los aportes y resultados de la revisión de la literatura, se llegó a la conclusión que existían dos principios relacionados que podrían hacerlo, pero que hasta el momento no se habían tenido en cuenta en los métodos y metodologías para gestionar la Ingeniería de Requisitos: la Multidimensionalidad y la Transdisciplinariedad.

1.1 Multidimensionalidad

Todas las etapas de la Ingeniería de Requisitos comparten como objetivo básico común: determinar las necesidades que debe satisfacer el sistema, con las diferencias y separaciones consecuentes del diálogo entre personas. Un equipo de trabajo bien conformado debe ser capaz de detectar no solamente la presencia o la ausencia de esas diferencias, sino también las separaciones y magnitudes más sensibles y precisas entre ellas. De esta manera las pueden solucionar rápidamente antes que tomen formas y caminos que incrementen su complejidad implícita.

En la práctica todos los sistemas se desarrollan a lo largo de un continuo entre dos puntos extremos: lo que necesitan o desean clientes y usuarios, y lo que deben respetar diseñadores y desarrolladores. Por lo que, para evitar que estas diferencias y separaciones en los requisitos lleguen a conformar distancias, es necesario analizarlas desde su escalamiento. Este principio permite construir un espacio multidimensional, o mapa, en el que los requisitos a escala se disponen de tal manera que las distancias entre las interpretaciones sean funciones del trayecto medido desde la visión de los usuarios a la visión de los ingenieros. En la medida que el equipo de trabajo utilice este sistema de medición, la escala de separación entre ambas visiones será una variable fácil de analizar, interpretar y controlar, antes que tome dimensiones que lo imposibiliten, pero para eso se debe desarrollar primero el pensamiento multidimensional.

Este tipo de pensamiento se puede describir como una conciencia y comprensión expandida que proporciona una profunda visión de la realidad, del mundo y de cada situación. Además, permite alcanzar simultáneamente un punto de vista en el que se integren diferentes tipos y niveles de

información. Esto es necesario porque las oportunidades desafiantes del siglo XXI requieren herramientas y métodos que vayan más allá de las limitaciones del lenguaje tradicional para comunicar las ideas. Se trata es de pensar desde muchos y diferentes puntos de vista sobre un tema, una habilidad esencial para el desarrollo de la capacidad de pensamiento crítico requerido en la Ingeniería de Requisitos, donde el equipo necesita: 1) realizar una evaluación crítica de la situación mediante la comprensión de los múltiples puntos de vista; 2) conjugar múltiples puntos de vista conflictivos y contradictorios para el logro del objetivo de la fase; 3) averiguar las interdependencias entre los actores (partes interesadas) y las variables (requisitos); y 4) desarrollar una visión completa de la situación a partir de las múltiples dimensiones intervinientes. Una vez hecho esto puede dedicarse a la búsqueda de una adecuada, y muchas veces creativa, solución para el problema en cuestión.

El problema que surge es que este tipo de pensamiento es difícil de desarrollar, porque los ingenieros no han sido formados para ello y porque el aislamiento disciplinar de los planes de estudios es un problema mundial. En la Ingeniería de Requisitos cada integrante del equipo de trabajo se acerca a la interpretación desde una perspectiva diferente del sistema, pero con una especie de pensamiento reduccionista que no le facilita alcanzar una idea completa del contexto. Puede ser que cada uno tenga una parte de la interpretación y solución, pero la estructura solamente se podrá poner en común cuando todos piensen desde lo multidimensional. Aquí es donde encaja la crítica del Pensamiento Complejo al pensamiento lineal, porque la complejidad de los sistemas no permite que con él se logre una imagen completa de la situación. La *criatura* grande que el equipo trata de entender es el problema y la Ingeniería de Requisitos es el medio para lograrlo, por lo que se requiere el enfoque innovador y multidimensional de pensamiento no-lineal. El pensamiento multidimensional se debe considerar como un requisito previo para la conformación del equipo de trabajo.

Puede que un enfoque estrecho ayude a la especialización productiva de cada disciplina, pero también puede reducir el campo de visión y ocultar cuestiones comunes más amplias. Este es el caso de la Ingeniería de Requisitos, donde el sentido de realidad de las necesidades del sistema se obtiene de la combinación entre lo que se ve, se escucha, se palpa y se comparte al elicitar requisitos. Pero en este camino el equipo se encuentra con muchos obstáculos, tales como las limitaciones tecnológicas, la in-comunicación, la experiencia individual, la organización y las disciplinas involucradas, entre otras. En todo caso, la Ingeniería de Requisitos necesita una gestión con pensamiento multidimensional, es decir, que el equipo sea capaz de ver el problema completo, así como el sistema en el que está embebido.

Por otro lado, tradicionalmente se reconoce dos modos de pensamiento que individualmente aplican diferentes formas para ordenar la experiencia multidimensional y construir realidad. Ambos son complementarios, aunque irreducibles entre sí, y los esfuerzos para hacerlo inevitablemente caen en el desperdicio de la rica diversidad del pensamiento. Esos modos son el *lógico-científico* (paradigmático) y el *narrativo*. Uno conduce a la búsqueda de condiciones de verdad universales y el otro a posibles conexiones particulares entre eventos. Para compararlos primero hay que entender la diferencia entre un argumento sólido y una buena historia, un proceso en el que es necesario contrastar los modos lógico-científico y narrativo en la misma variedad de dimensiones desde las que se originan (Jones, Ertas y Parten, 1995).

En este sentido un supuesto central en la ciencia ha sido que la organización es una característica intrínseca del mundo social. Además, se cree que los sistemas en general y el software en particular están organizados de una manera u otra, y que la tarea de los ingenieros consiste en averiguar cómo y por qué. Con este fin se ha desarrollado dos escuelas de pensamiento: la primera con orientación *sociológica-histórica-antropológica*, que busca comprender y explicar las características específicas de la organización, de la que actualmente existe una amplia variedad de

metodologías y teorías diversas en ella, pero la multidimensionalidad apenas es tenida en cuenta. La segunda escuela es la *cibernético-sistémica*, donde se concibe a la organización de manera mucho más amplia, se piensa que es una característica del cosmos en general, no solamente de las colectividades sociales; además, tiene en cuenta a las formas vivas como a las no-vivas, y la sugerencia es que observando el mundo no-social hay mucho que aprender de la organización.

De hecho, la cibernética y la teoría de sistemas se construyeron sobre esta premisa y por eso es que la multidimensionalidad cobra importancia y tiene sentido en esta escuela, porque la complejidad es inherente en la misma. Sus defensores sostienen que la comprensión de una organización se puede mejorar a través de las modelizaciones y la búsqueda de analogías con otros sistemas naturales y biológicos. Porque ese conocimiento es multidimensional, permite una visión amplia de las relaciones y facilita las interpretaciones, tal como se experimenta en esta fase.

Este enunciado se puede validar con algunos de sus postulados: que los sistemas narran, que recogen y transmiten historias, y que siempre tienen una narrativa. El primero se refiere a que el sistema narra lo que la organización es, es decir, cuenta sus procesos, relaciones y conflictos, lo cual es necesario para que la Ingeniería de Requisitos pueda abstraer y comprender el problema a solucionar. El segundo está relacionado con el hecho de que todo sistema genera y tiene documentación escrita, en la oralidad de las personas o en los diagramas de otros sistemas. Mientras que la narrativa se materializa en la producción del sistema, es decir, no es necesario tener una descripción escrita cuando el producto final puede expresarse de cuenta propia. Algo que es común a todos es que el sistema recibe información y entrega conocimiento o productos, a partir de fuentes multidimensionales que solamente él puede identificar y describir mediante una narrativa particular. Los estudiosos de la Complejidad (Cardoso, 1998; Daneva y Wieringa, 2006; Cooke, 2011) resaltan algunas de esas dimensiones comunes a los sistemas naturales, artificiales, biológicos y sociales que pueden aplicarse para gestionar la Ingeniería de Requisitos:

- *La no-linealidad*. Los sistemas no son lineales y no tienen proporcionalidad entre causas y efectos. En ellos las pequeñas causas pueden generar grandes efectos. Por eso la no-linealidad es la regla mientras que la linealidad es la excepción. Como dimensión, los ingenieros pueden aprovecharla para analizar el origen y la interacción de los requisitos, de tal manera que puedan aprovechar las descripciones intrínsecas como material para llegar a los acuerdos de interpretación de los mismos.

- *Son fractales*. En todos los sistemas las formas irregulares dependen de la escala y no tienen una medida que les dé respuestas verdaderas, porque en todos depende del dispositivo de medición. En la Ingeniería de Requisitos esto es útil para definir la escala con la que se analizará los requisitos y sus interacciones, porque al ser fractales su comprensión depende de la escala elegida para medirlos: cuanto menor sea la escala, mayor será la medida obtenida.

- *Exhiben simetrías recursivas entre niveles de escala*. Porque tienden a repetir una estructura básica sobre varios niveles. Los requisitos reflejan esta dimensión porque siempre existe la posibilidad que hagan parte de otro más complejo, que a su vez está inmerso en otro. Por eso es necesario tener una visión amplia del sistema, para gestionar los requisitos en el sentido de crear mapas mentales recursivos de las relaciones e interrelaciones involucradas en la abstracción de las necesidades de los usuarios.

- *Son sensibles a las condiciones iniciales*. Incluso la más pequeña perturbación los podría enviar en una dirección diferente a la proyectada. Esto es particularmente importante en la Ingeniería de Requisitos, porque las condiciones iniciales del sistema no se pueden especificar con precisión y que a la vez se mantenga, porque las fuentes son personas u otros sistemas que tienen tendencia a ser impredecibles.

- *Son retroalimentados*. Porque la dinámica de los sistemas es el resultado emergente de múltiples cadenas de interacción, y a medida que aumenta la complejidad de las mismas tienen tendencia a cambiar a nuevos modos de comportamiento. Estas novedades emergentes representan puntos de bifurcación, una situación que se vive permanentemente en la Ingeniería de Requisitos donde las necesidades del sistema siempre están en evaluación y actualización. El equipo debe tener en cuenta esta dimensión para estructurar cadenas de seguimiento a los cambios y para adelantarse a uno inminente o a un desastre para el sistema, originado por una retroalimentación interna o externa a los requisitos.

Para comprender la complejidad de estas dimensiones desde la perspectiva del equipo de trabajo, habría que ponerse en la situación de una persona que intenta describir un sistema en los términos que cada integrante manifiesta. Aunque puede encontrar alguna relación entre ellas, siempre lo hará desde su conocimiento disciplinar, en sus términos descriptivos y de acuerdo a su experiencia. Esto remite inmediatamente a la necesidad de involucrar en paralelo los principios transdisciplinares que se describen más adelante.

Por otro lado, ya se ha referido el hecho de que el sistema no puede hablar por sí mismo y, para saber lo que es en realidad, hay que recurrir a las fuentes primarias, conocidas como partes interesadas, para moldear una interpretación que posteriormente se analiza y discute para llegar a acuerdos básicos. La cuestión es que el lenguaje natural no se puede separar de los intereses, las culturas y las creencias de cada actor involucrado en la Ingeniería de Requisitos, lo que hace incierta una adecuada comprensión del sistema. Pero teniendo en cuenta las mismas dimensiones, es posible que los ingenieros puedan controlar la narrativa, las observaciones y las creencias de los involucrados, para abstraer lo que realmente importa como requisitos del sistema. Es una apuesta riesgosa, porque ellos mismos no están preparados para interpretar lo que puedan llegar a moldear.

1.2 Transdisciplinariedad

La Transdisciplinariedad trata de explorar directamente la naturaleza de las conexiones entre las que existen similitudes subyacentes cuando las cosas se perciben desde diversas temáticas. Esta afirmación se sustenta en las palabras de Basarab Nicolescu (1996: 114): *Volvemos entonces a la imperiosa necesidad de proponer, vivir, aprender y enseñar un Pensamiento Complejo, que vuelva a tejer las disciplinas como posibilidad de humanidad en completitud; y que solo de esta manera se vencería la eterna limitación y fragmentación del sujeto separado de sí mismo en la búsqueda del conocimiento*. En un sentido amplio todos los esfuerzos y actividades humanas siempre tratarán de hacer avanzar su comprensión de las interconexiones entre los diversos campos del saber. Esto genera una serie de esfuerzos que pueden ser ordenados jerárquicamente de acuerdo con el grado de interconexión que se persigue.

En este sentido, la Transdisciplinariedad se construye sobre la premisa de que hay principios y propósitos discernibles que subyacen a todo el sistema conocimiento/conocedor. En la opinión de pensadores como Piaget la epistemología de las relaciones debe dedicar sus mejores esfuerzos a este principio. Por su parte, Mather (1967) captura la esencia de la justificación de un enfoque transdisciplinar para la integración, entre otros, de los planes de estudios (Jantsch, 1972). Pero este enfoque busca no solamente una axiomática común subyacente a todo el conocimiento, sino también la coordinación de todo el sistema hacia un propósito general de significado y valor. Un aporte que es sustancialmente importante para la Ingeniería de Requisitos de esta época.

El objetivo de un esfuerzo de este tipo es unir el conocimiento proveniente de cada disciplina involucrada: 1) sobre el nivel de los principios subyacentes comunes necesarios para comprender las necesidades que debe satisfacer el sistema, y 2) sobre el nivel de su misma comunalidad como expresiones de la conducta humana con un propósito definido. Entonces, surge el interrogante

sobre qué características de la Transdisciplinariedad son útiles en la Ingeniería de Requisitos. Una manera de responder sería mediante la recopilación de una larga lista de postulados extraídos desde la Teoría de Sistemas, el estructuralismo, las matemáticas, la física, la psicología y cualquiera otra área del conocimiento con fundamentos teóricos generalizables más allá de su propio dominio. Esta manera puede ser viable debido a que el tema del creciente paralelismo entre los postulados de la ciencia moderna y la clásica cosmovisión religioso/filosófica está bien establecido en la cultura. Los filósofos/científicos han sugerido que las ideas de la ciencia moderna son tan avanzadas, que se podrían utilizar para restaurar el reducido paradigma del conocimiento contemporáneo en una visión más acorde con sus raíces holísticas (Eddington, 1935; Kunz, 1967; Schrodinger, 1967). Por lo tanto, un esfuerzo exploratorio para interrelacionar temas avanzados de las ciencias, las artes y las humanidades con diversas perspectivas de integración, podría ayudar a definir la forma de esta síntesis y responder la pregunta planteada. La preocupación es que un conjunto no-integrado de principios candidatos, tomados por fuera del contexto de la Ingeniería de Requisitos, no contribuiría mucho a la resolución del interrogante.

Otra alternativa sería identificar algunas afirmaciones que se puedan hacer sobre el estado epistemológico de estos principios. De esta forma se pondrían al descubierto los supuestos que subyacen a la filosofía de la Transdisciplinariedad, además de avivar la discusión sobre temas relacionados que merecen ser explorados con más cuidado. Para iniciar este proceso es necesario plantear algunas carcaterísticas que pueden ofrecer claridad para responder la pregunta original:

1. *Averiguar si las características de la Transdisciplinariedad para la Ingeniería de Requisitos pueden describir los patrones de conocimiento que expresan los usuarios en sus necesidades*. En la literatura se encuentra propuestas sobre la necesidad de *inventar* principios interdisciplinares para *crear* conexiones a través de los diversos campos del conocimiento (Jantsch, 1972; Golfarelli, Maio y Rizzi, 1998). Pero se expresan en una terminología un tanto confusa y a veces se utilizan en un contexto que sugiere que la perspectiva transdisciplinar es una herramienta con valor puramente pragmático, por lo que sus principios no pueden asumirse como descripciones de lo que se expresa en lenguaje natural. Nada podría estar más lejos del espíritu de la investigación científica, porque la ciencia supone que existe un orden inherente en la naturaleza que puede ser cognoscible por el hombre. Tradicionalmente, la labor del científico consiste en describir, en términos cada vez más completos y precisos, esos patrones de conocimiento y la relación causal en la que son inmutables y eternos. En este sentido y al hablar de las matemáticas como el lenguaje de la ciencia, Bertrand Russell captura la esencia de este espíritu racionalista (Polanyim, 2009).

   Algo parecido se puede atribuir a las artes y las humanidades bajo la premisa de que todo lo relacionado con la humanidad en estas áreas tiene que ver con una forma superior de conocimiento, desde la que se puede acceder a la ciencia. En consecuencia, sus métodos se diferencian de la ciencia en que son más evocadores que descriptivos y más metafóricos que analíticos. Sin embargo, su objetivo es similar a comunicar las experiencias de la vida y, cuanto más universales y perdurables sean, tradicionalmente serán más valoradas. Esta cuestión es fundamental para que los ingenieros entablen diálogos constructivos con los usuarios y clientes, con el objetivo de comprender su conocimiento del problema cuando describen las necesidades del sistema. Un principio transdisciplinar de este tipo le daría al equipo la libertad para comprender los requisitos desde diferentes puntos de vista; además, para explorar, describir, evocar y verificar, por cualquier medio que estime conveniente, los patrones del conocimiento recopilado en la Ingeniería de Requisitos.

2. Otra cuestión a precisar es *si las características de la Transdisciplina son menos precisas en la Ingeniería de Requisitos que los principios ingenieriles tradicionales*. Si esto es cierto se debe identificar y justificar esa imprecisión antes de seleccionarlos. En la literatura es común encontrar que, cuando se formula algo relacionado con Transdisciplina, los autores se ven

forzados a aceptar compensaciones entre precisión y exhaustividad (McMichael, 2000; Häberli y Thompson, 2002; Lawrence y Després, 2004; Thompson, 2004). Los que están a favor de la precisión argumentan que se incrementa la capacidad descriptiva y manipuladora, a costa de reducir el intervalo sobre el que se aplica el principio. Quienes están a favor de la exhaustividad obtienen una visión en un rango más amplio de interacción, a expensas de la precisión.

Pensar de esta forma en la Ingeniería de Requisitos genera mayor confusión e interpretaciones imprecisas del conocimiento de las partes interesadas. La falta de un principio unificador de los diferentes puntos de vista no le permite al equipo interpretar el conocimiento circulante sobre el contexto del problema, es decir, puede surgir interpretaciones solamente por imposición y no por una adecuada interpretación transdisciplinar. Lo que se debe hacer es estimular la comprensión del conocimiento común discernido y preguntarse colectivamente qué otras cosas se pueden aprovechar desde las disciplinas de los integrantes, para añadirle solidez a la discusión y al análisis. En la Ingeniería de Requisitos hay momentos en los que vale la pena ceder especificidad a favor de la integralidad, y a este propósito son útiles características como:

- *La interdependencia de las partes y el todo*. Porque a medida que se avanza en la comprensión del problema y se comparte conocimiento entre las partes interesadas y el equipo de ingenieros, la integración y las relaciones resultantes le dan forma al sistema-solución. Porque conocer y comprender cada interpretación de forma aislada debe forzosamente conducir, mediante interrelaciones disciplinares, al todo que se desea solucionar.

- *La adaptabilidad sobre la base de la estabilidad subyacente*. Esto quiere decir que el equipo, luego de unificar y comprender el conocimiento de las partes, debe tomar lo que ya se ha hecho para solucionar este tipo de problemas y adaptarlo y reutilizarlo. Porque uno de los principios de la ingeniería es aprovechar lo existente para no repetir los procesos que lo originaron, y de esa forma conseguir tiempo para otras actividades. Si algo funciona desde algún tiempo es porque tiene estabilidad y solamente requiere un poco de adaptabilidad.

- *Aprovechar el conocimiento individual disciplinar como base para estructurar la comprensión grupal transdisciplinar*. El conocimiento que cada individuo tiene del problema desde su disciplina o rol, se integra en un conocimiento común, contextual y compartido en el grupo. Esto genera una visión del problema en la que la Transdisciplina es la base sobre la que se estructura la interpretación individual de los requisitos, al tiempo que sus relaciones van creando el mapa de comprensión del problema.

- *Homeostasis*. En la Ingeniería de Requisitos se genera diálogos, fricciones e interpretaciones y se presenta puntos de vista conflictivos, que conducen a conjuntos de fenómenos que deben ser aprovechados como autorregulación del proceso. El objetivo es mantener la composición y las características al interior de la relación entre el equipo de trabajo y las partes interesadas. Estos dos grupos conforma una especie de organismo cuyas propiedades deben permanecer estables valorando las relaciones que se establezca entre los individuos.

- *Que los cambios externos no afecten la estabilidad interna*. De la misma forma que los seres vivos tienen mecanismo para mantener estable su temperatura interna, sin importar las condiciones climáticas externas, el equipo de la Ingeniería de Requisitos debe crear un sistema de protección que le permita *ser equipo* en todo momento y bajo toda circunstancia. Cuando se resquebraja esta característica aparecen los puntos de vista, las interpretaciones y las imposiciones individuales.

- *Permitir cambios necesarios a través de agentes no involucrados*. Cuando los equipos logran la estabilidad interna todavía necesitan intervenciones de agentes no involucrados, para comprender situaciones que se escapan de su conocimiento o experticia. En este caso es necesario saber seleccionar al agente colaborador y analizar, comprender y ubicar sus aportes en el contexto del problema. Así sería posible salir de los baches en los se cae por falta de conocimiento en alguna disciplina necesaria, pero que no amerita su integración a las ya involucradas.

- *Cada componente estructural tiene funciones que complementan el objetivo del organismo*. Reconocer las capacidades y conocimientos de cada integrante del equipo de trabajo y de las partes interesadas, es importante para mantener la estructura en pro de lograr el objetivo de la Ingeniería de Requisitos. La labor del líder del proceso es encontrar y aprovechar la potencialidad de cada uno para el logro de ese objetivo, y para que como equipo se logre la armonía y consistencia necesaria para alcanzarlo.

Diversas empresas de software argumentan que solamente con el lenguaje y los aportes disciplinares es suficiente para gestionar la Ingeniería de Requisitos, pero si esto fuera completamente cierto, la calidad, y particularmente la fiabilidad del software actual, tendría mayores niveles de aceptación en los usuarios. Lo que sucede es que una visión disciplinar es parcial y no permite trazar un mapa completo del problema, porque su lenguaje y área de aplicación son propios y los profesionales no tienen acceso a agentes externos que complementen su interpretación de los requisitos en un contexto más amplio. Para que las características de la Transdisciplinariedad sean realmente integrales, deben ser en general comprensibles. En la Ingeniería de Requisitos esto está en consonancia con el hecho de que los actores que intervienen poseen el lenguaje y la experiencia de la vida individual diaria en su disciplina, pero todavía no son comprensibles para todos como equipo. Por eso es necesario encontrar el lenguaje, la interpretación, la oralidad, la opinión y el análisis integral que los lleve de lo disciplinar a lo transdisciplinar.

3. *Determinar las implicaciones para la Transdisciplinariedad de la capacidad humana para experimentar la trascendencia*. El término trascendencia se refiere a la inclinación por ir más allá de los límites de cualquier estado o realización. No es simplemente una apertura porque sí, sino la búsqueda del conocimiento y las características de ser: verdad, bondad, belleza, integridad, vitalidad, perfección, justicia, entre muchas otras. No es una experiencia absoluta en el sentido que implica la dicotomía sujeto-objeto, sino un estado de auto-conciencia inmanente sin limitaciones por los objetos de la experiencia. Las personas trascendentes tienen la tendencia a unificar o totalizar la comprensión y la experiencia de su entorno y, en comunidad, son como una conciencia de unidad generalizada que se sumerge en la búsqueda del conocimiento por conocimiento.

   En la Ingeniería de Requisitos la transcendencia puede hacer que, individualmente, las personas se sumerjan en una extensión lógica con la idea de profundizar su perspectiva en la mecánica de la comprensión del problema. Aunque generalmente esto se puede ver como un periodo de creatividad, solamente requiere una simple incubación pasiva para que su análisis desborde las fronteras del problema, y se pierda la idea transdisciplinar de la integración dinámica de las interpretaciones. En este punto es necesario conceptualizar la relación entre trascendencia y Transdisciplinariedad para definir la cuestión de cómo la primera puede mejorar la capacidad del equipo para formular y aplicar las características transdisciplinares. Aunque no es fácil responder a esta cuestión, porque tiene más relación con asuntos de la psicología del ser, en la Ingeniería de Requisitos se puede visualizar sobre la base de la observación y la interacción entre las personas involucradas. En este sentido, la transcendencia le ayuda a tener una visión holística del problema, aunque el reto sería evitar que las extensiones lógicas disciplinares desborden sus fronteras.

Aquí es importante que el equipo administre el conjunto de principios transdisciplinares a un nivel de penetración que satisfaga tanto a los elementos de la comprensión y la interpretación, como a las relaciones intrínsecas de las fronteras del problema. Por ejemplo, se podrían utilizar para interconectar todos los campos del saber involucrados o para unir las disciplinas en un propósito común, materializados en un sistema-solución. Así, las características transdisciplinares holísticas podrían sugerir sutilmente una dirección hacia ese sistema. Por supuesto, no se puede caer en la tentación de involucrar agentes y disciplinas adicionales no necesarios, porque en la Ingeniería de Requisitos hay un número limitado de tipos y categorías de conocimiento disciplinar, que el mismo problema involucra con especificidad.

VII

MODELO PARA GESTIONAR LA INGENIERÍA DE REQUISITOS INTEGRANDO PRINCIPIOS DEL PENSAMIENTO COMPLEJO

La Ingeniería de Requisitos es un enfoque sistemático a través del cual se especifica las necesidades de las diferentes partes interesadas y se documentan para el resto del ciclo de vida. Es un proceso complejo, interactivo e iterativo en el que la gestión es un aspecto de alta importancia (Zave, 1997; Stevens et al., 1998; Kotonya y Sommerville, 1998), porque los productos de esta fase son esenciales para las demás actividades del desarrollo. En la Ingeniería del Software tradicional se lleva a cabo como una fase inicial (Jureta, Faulkner y Schobbens, 2006), sin embargo, en la práctica, especificar un conjunto preciso de requisitos y mantenerlo estable a lo largo de la línea de tiempo, es una tarea que podríamos denominar *imposible* (Serna, 2012).

Por lo tanto, la Ingeniería de Requisitos se debe concebir como un proceso incremental y de *vista atrás*, que se ejecuta y enriquece con el objetivo de entregar una especificación suficientemente estable para las otras fases del desarrollo. Como tal, es el proceso más crítico y complejo en el desarrollo de productos software (Andriole, 1996; Hull, Jackson y Dick, 2011; Juristo, Moreno y Silva, 2002), porque el documento de especificación es la guía para el ciclo de vida y para describir la Transdisciplinariedad de sus procesos y patrones, además de la Multidimensionalidad en las diversas interacciones sociales en las que tiene lugar.

También tiene un impacto predominante en la funcionalidad y en la fiabilidad del producto final, porque en ella confluye un conjunto de solicitudes diverso, proveniente desde diferentes actores que el equipo de desarrollo debe considerar. Por estas razones es una fase compleja, a la vez que crítica, y requiere un modelo adecuado de gestión.

En la Figura 1 se presenta el modelo para gestionar la Ingeniería de Requisitos, en el que se involucra los principios: Transdisciplinariedad, Multidimensionalidad y Gestión del Conocimiento, integrados sistemáticamente para lograr una adecuada especificación. Se estructura en etapas y abarca desde la comprensión del problema y la conformación del equipo, hasta la entrega del documento de especificación de requisitos. Se describe las actividades necesarias para lograr el producto de cada una, incluyendo los principios desde las áreas del conocimiento descritas previamente. El modelo se compone de cinco etapas:

1. Etapa temprana
2. Etapa de elicitación
3. Etapa de desarrollo
4. Etapa de gestión
5. Etapa de especificación

Además, se inscribe bajo la concepción de que: 1) la Ingeniería de Requisitos hace parte integral del proceso de desarrollo de software, y 2) las pruebas son una actividad paralela a todo el ciclo de vida del software (Serna, 2012a). Cuestiones que se debe tener en cuenta si el objetivo es mejorar la fiabilidad del producto final.

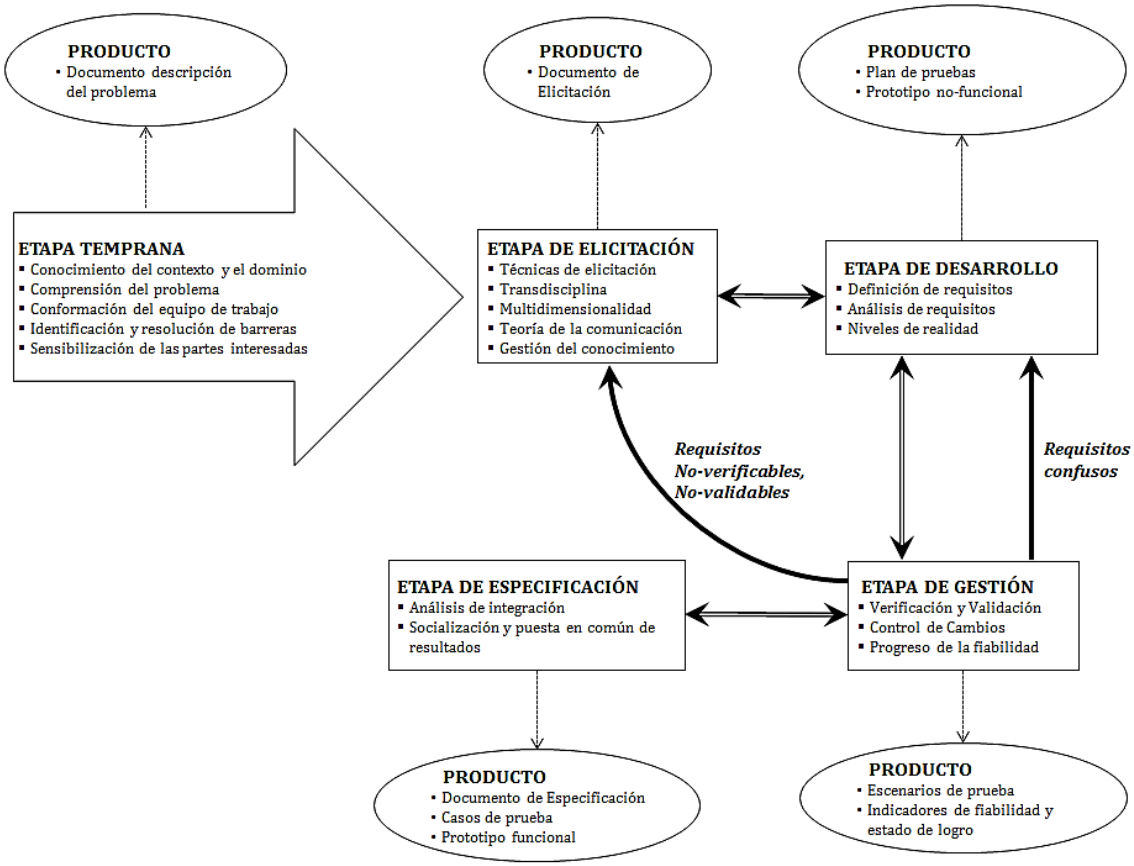


Figura 1. Modelo para gestionar la Ingeniería de Requisitos integrando principios del Pensamiento Complejo

1. ETAPA TEMPRANA

Los problemas relacionados con la Ingeniería del Software, especialmente los que afectan la fiabilidad esperada del producto, el cumplimiento del presupuesto y los retrasos en la entrega, se generan en una gestión inadecuada de la Ingeniería de Requisitos. En la práctica se observa una enorme brecha entre la realidad de las operaciones en la organización y el conocimiento que las partes interesadas tienen de ese contexto. Esto se debe a una insuficiente gestión del conocimiento y a la libre interpretación que se hace del dominio específico. A partir de estas premisas la Etapa Temprana del modelo se estructura alrededor de una serie de actividades básicas, cuya interrelación permitirá comprender el problema y su contexto, a la vez que analizar y reconocer el dominio y el modelo de negocios de la organización. Este objetivo responde a la necesidad de ejecutar procesos repetibles donde se analice, entienda y comprenda adecuadamente el problema, a la vez que se modela el contexto y el dominio del mismo.

La propuesta consiste en articular en esta Etapa actividades ingenieriles, lógicas y de gestión con las que, sin importar el contexto, se pueda alcanzar ese objetivo. En ella se hace una integración y puesta en común de principios ingenieriles y matemáticos, mediante la mediación del Pensamiento Complejo, para identificar la Transdisciplina, la Complejidad y la Multidimensionalidad involucradas en la comprensión de la situación a solucionar. Además, el éxito en la ejecución de toda actividad basada en la gestión del conocimiento requiere un amplio entendimiento del dominio en el que circula. A continuación, se describe el procedimiento de la Etapa Temprana.

1.1 Conocimiento del dominio y el contexto

En esta actividad el líder del equipo, que no se ha conformado estructuralmente porque en este momento solamente tiene integrantes desde la ingeniería, inicia procesos de reconocimiento y de

relaciones orientadas a saber dónde está ubicado, primero como ingeniero y luego como parte interesada. Esto implica encontrar las relaciones transdisciplinares entre lo que cree saber y lo que su lógica e ingenio le indican que va a aprender. Hace uso de su formación disciplinar para ubicarse en el domino de la organización, además de su experiencia y conocimiento para ubicarse en el problema. A medida que comprende y determina las coordenadas de su ubicación, asesora a sus compañeros, generalmente de su misma disciplina, para que compartan el conocimiento en relación con la situación en curso. Esto se realiza a través del diálogo y en medio de escenarios complicados de gestión del conocimiento, porque todos no tienen los mismos puntos de vista ni las mismas experiencias sobre el hecho. Aunque todavía no podría hablarse de que es un proceso complejo, debido a que el lenguaje en que se desarrolla es común, la situación se podría complicar si las fuentes del conocimiento no se gestionan adecuadamente.

En este modelo y para esta Etapa se sugiere utilizar el modelo de madurez para gestionar el conocimiento transdisciplinar propuesto en Serna (2015), porque debido a su principio evolutivo permite gestionar el conocimiento como un elemento intensamente activo y dinámico, que evoluciona en madurez desde los primeros momentos de intercambio. Para eso es necesario aplicar un proceso limpio, claro y conjunto de intercambios disciplinares y de integración transdisciplinar para integrar el conocimiento creado, descubierto y/o almacenado y compartido.

El modelo se basa en que la Transdisciplina es el enfoque de investigación que mejor se adapta para hacerles frente a los complejos problemas que el propio desarrollo científico genera continuamente, de hecho, se refiere principalmente a la relación ciencia-sociedad. Además, es intervencionista porque metódicamente enmarca, estructura y organiza el discurso social sobre una problemática determinada. Al conocimiento se le asigna un papel especial, porque desde la Ingeniería de Requisitos lo transdisciplinar tiene la obligación de gestionarlo y distinguir sus diferentes formas de representación y orígenes, debido a que esclarecen la forma en que se produce y cómo se relaciona en la red de interconexiones complejas entre las partes interesadas y los sistemas. En esencia, la Transdisciplina en el modelo es tanto crítica como auto-reflexiva, porque examina no solamente la forma sistemática en que se produce y utiliza el conocimiento, sino también los diferentes actores que lo soportan. Además, desafía metódicamente la búsqueda de la verdad y la cada vez mayor demanda por resultados de utilidad de la especificación de requisitos.

En este momento se puede decir que el *proto-equipo* conoce con certeza la ubicación y el papel de cada integrante, desde la disciplina y como parte interesada, y por lo tanto pueden modelar el dominio y el contexto, aunque todavía sin comprender completamente el problema. El objetivo de esta actividad es crear los mapas mentales y los modelos lógicos y abstractos de la organización, además del contexto de intervención del problema, asimilando las reglas del negocio de forma general. Al lograr esto pueden orientar sus esfuerzos a comprender el problema y a definir las relaciones que permitirán integrarlo, primero en el contexto y luego en el dominio. El peligro en este momento es que pierdan el control y sigan ejecutando cada actividad de forma lineal. Para evitarlo, deben estructurar y proyectar metas relacionales y en conjunto para las demás actividades de la Etapa. Es decir, desde ahora ya se debe tener claridad sobre qué tipo de problema esperar, qué disciplinas se podrían implicar (contables, Ciencias Computacionales, sociología, derecho, medio ambiente, entre otras), qué dimensiones se involucrarán y cuáles podrían ser los perfiles de las partes interesadas.

1.2 Comprensión del problema

Utilizando el conocimiento de los ingenieros el objetivo ahora es hacer una lectura a la situación problema, a la que deben presentar una solución. A partir de las fuentes de datos, de información y conocimiento que la organización comparte sobre la situación, hacen un bosquejo en el que relacionan las disciplinas conocidas y analizan cuáles deberían incorporar para comprenderlo

completamente. Desde este bosquejo inicial los ingenieros tienen el primer nivel de comprensión transdisciplinar (*empírico*), porque reconocen el problema tal como es y pueden responder a la pregunta ¿*qué existe*? Para ello hacen uso del lenguaje de la lógica[1].

Para pasar el siguiente nivel, el *pragmático*, utilizan el lenguaje de la cibernética[2] para responder a ¿*qué somos capaces de hacer*? Entonces evalúan su conocimiento disciplinar y el adquirido de la documentación para trazar un plan sobre cómo iniciar el proceso de solución. Muy importante que no dejen primar los valores, presunciones e interpretaciones de su disciplina, porque de esa manera estarán direccionando automáticamente el plan de acción para el resto de la Etapa, lo que podría ocasionar que no logren conformar el equipo de trabajo necesario y terminarían aplicando los modelos tradicionales. En este momento tienen una comprensión inicial del problema y una base de conocimiento sólido, aunque todavía su visión es dirigida por la ingeniería. Necesitan ascender a los siguientes niveles de la Transdisciplinariedad, por lo que primero deben terminar de conformar el equipo de trabajo.

1.3 Conformación del equipo de trabajo

Hasta este momento los ingenieros saben *qué existe* y *qué son capaces de hacer* con el conocimiento que tienen del problema, pero, para pasar al nivel *normativo* y responder a ¿*qué queremos hacer*? deben aplicar el lenguaje de la planificación[3], seleccionar los demás miembros del equipo e identificar todas las posibles partes interesadas. Esta es una tarea conjunta en la que, a partir de un mapa mental, construyen el diagrama de procesos que interrelaciona al problema con el contexto y con el dominio, tanto internos como externos de la organización. El objetivo es identificar las áreas del conocimiento involucradas y necesarias en el problema, para luego convocar los representantes disciplinares respectivos e integrar el equipo primario.

Con el equipo de trabajo conformado y con las partes interesadas identificadas, es el momento de avanzar al nivel superior de la Transdisciplinariedad: *de valor*, por lo que deben utilizar el lenguaje de *ecología profunda*[4] para responder transdisciplinarmente a ¿*cómo debemos hacer lo que queremos hacer*? Es un nivel en el que el equipo aplica visión prospectiva para estructurar el plan de acción de la Ingeniería de Requisitos.

1.4 Identificación y resolución de barreras

Una característica común al trabajo en equipo es la persistencia de sus integrantes a hacer prevalecer su disciplina, experiencia y conocimiento sobre los demás. El líder debe identificar todo tipo de barreras que puedan surgir, y plantear mediante el diálogo y la comunicación las normas internas bajo las cuales se desarrollará el trabajo. Debido a la diversidad de las partes interesadas involucradas es común que inicialmente carezcan del conocimiento del dominio y del contexto del problema, lo que le añade complejidad a esta actividad. Por eso se requiere gestión del conocimiento y planeación estratégica, además de lograr que todos confíen en la base de conocimientos que se ha acumulado hasta el momento. Claro que, en la medida que se avanza en la siguiente Etapa del modelo, puede ser que ese conocimiento se tenga que debatir, porque la comprensión inicial partió desde una sola disciplina.

El líder debe integrar lo que saben las partes interesadas y el conocimiento desde las disciplinas como medio para desarrollar una comprensión más amplia del problema, y resolver las barreras que identifique en el proceso. Es una puesta en común de los lenguajes, medios, visiones, interpretaciones y deseos de los integrantes del equipo y de las partes interesadas. De esta manera

[1] Se refiere al lenguaje de notaciones precisas que no dejan espacio a la ambigüedad.
[2] Es decir, una forma de hacer cierta y precisa la información para convertirla en conocimiento.
[3] Es un esfuerzo deliberado para influir en la función, estructura, o la adquisición de un lenguaje común dentro de una comunidad.
[4] Haciendo énfasis en la integridad, el equilibrio, la armonía y las formas tradicionales al interior del equipo.

se mejora la respuesta a la pregunta del primer nivel de la Transdisciplinariedad: *¿qué existe?* porque esta multidimensionalidad de las fuentes del conocimiento les permite una mejor comprensión total del problema.

Una de las barreras tradicionales es que los usuarios o el cliente (partes interesadas) no compartan ampliamente su conocimiento, en parte porque se han apoderado de él a título personal y no quieren que otros sepan cómo realizan sus funciones. Esto es un problema común en las organizaciones que no han implementado la Gestión del Conocimiento. Demostrar lealtad y lograr la confianza de estos actores es fundamental para resolver esta barrera.

1.5 Sensibilización de las partes interesadas

A medida que se descubre e integra el conocimiento con la participación de las partes interesadas, surge un aprendizaje práctico del contexto y del dominio que se desarrolla a través de actualizaciones permanentes, porque las partes no confían sino a medida que sienten que son importantes en el proceso. De esta manera van desarrollando mayor familiaridad con el equipo, el problema, la organización y las tecnologías, por lo que las barreas y prejuicios iniciales también disminuyen. Cuando la organización ha desarrollado un modelo de Gestión del conocimiento la internalización en el equipo no tiene muchas complicaciones, y el funcionamiento como tal se fortalece.

Conocer el contexto y el dominio desde la experiencia del equipo y las partes interesadas es en sí mismo una forma de adquirir conocimiento, pero existe una fuerte tendencia a facilitar su consecución sin una verificación previa. El objetivo de sensibilizar a las partes interesadas es resolver estas dificultades, particularmente en la apropiación y credibilidad del conocimiento compartido al interior del equipo. La participación activa y permanente de todos los actores en las actividades de esta etapa proporciona en gran medida, y determina, el nivel de calidad del producto final. Por eso es importante encontrar y romper las barreras de comunicación y de participación, y trazar planes de sensibilización para que entiendan el rol que desempeñan y su importancia en el proceso.

La cuestión es que se necesita gestionar el conocimiento adecuadamente y aunque se podría pensar que desde los diferentes modelos y técnicas para identificarlo, obtenerlo y representarlo esta situación está resuelta, al considerar el asunto de la adquisición estos enfoques reconocen su importancia, pero no proporcionan un proceso sistemático para realizarla. La propuesta esquemática para gestionar el conocimiento en la Etapa Temprana se estructura mediante una perspectiva simplificada, como se muestra en la Figura 2.

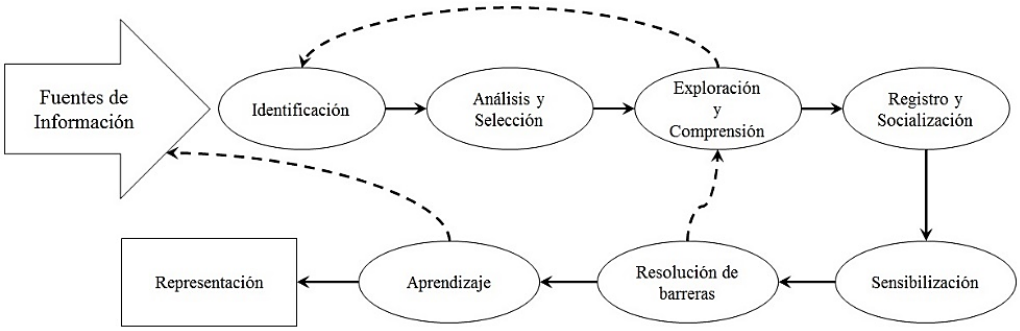


Figura 2. Gestión del conocimiento en la Etapa Temprana

2. ETAPA DE ELICITACIÓN

El objetivo de la elicitación es capturar las necesidades de los usurarios desde las diferentes dimensiones en las que se originan, visto como algo más que un ejercicio de descubrimiento,

porque se centra en examinar y reunir los requisitos necesarios y deseados para el sistema desde dimensiones como clientes, usuarios, negocio, comercialización, estándares, pruebas, seguridad y otros sistemas.

Cuando el equipo no reúne el conocimiento suficiente para comprender los requisitos desde el contexto y el dominio del problema, pueden ser mal interpretados, por lo que es importante aplicar adecuadamente la Etapa Temprana. Hay que tener en cuenta que las normas, los estándares y las restricciones juegan un papel importante en el desarrollo del sistema, por lo que la elicitación debe ser contextual, iterativa e incremental. En el modelo se involucra principios tradicionales, tales como buenas prácticas, lógica, abstracción y métodos formales, además de Transdisciplinariedad y Multidimensionalidad.

Se ha publicado diversas técnicas para llevar a cabo la elicitación, pero lo más importante es tener claro el plan de acción y las metas establecidas para esta Etapa. Dichas técnicas utilizan diversos conceptos de la Teoría de la Comunicación para acercarse a las partes interesadas, con el objetivo de recolectar sus necesidades, apreciaciones e interpretaciones alrededor del problema. En este caso, lo primero que el equipo debe tener en cuenta es que esas necesidades provienen de la gestión del conocimiento en disciplinas específicas, por lo que persiste la prerrogativa de integrarlas en una visión única, pero transparente a todos los integrantes.

Usualmente los encargados de la elicitación tienen solamente la visión del problema desde la ingeniería, por lo que en el proceso tienden a interpretar y modelar los requisitos de forma lineal y disciplinar. Pero para lograr la imagen completa de las necesidades y comenzar a estructurar las bases de una posible solución, hay que aplicar pensamiento multidimensional. Por eso es necesario que el equipo busque el sentido de realidad de las necesidades del sistema, combinando lo que se ve, se escucha, se palpa y se comparte al elicitar los requisitos. En el desarrollo de su trabajo debe vencer las limitaciones en la comunicación, el lenguaje, la falta de experiencia, la organización, la inadecuada Gestión del conocimiento y las disciplinas que reúne cada una de las partes interesadas.

En este momento son útiles los dos modos de pensamiento descritos anteriormente, ya que el *paradigmático* conduce a la búsqueda de condiciones de verdad universales, es decir, a una puesta en común de definiciones, interpretaciones y visiones acerca de cada requisito elicitado; mientras que el *narrativo* permite encontrar las posibles conexiones particulares entre los eventos que describen las partes en sus respuestas a la técnica de elicitación seleccionada. Como el equipo está integrado de forma transdisciplinar, al momento de dialogar con las partes interesadas esto le permite entender la diferencia entre un argumento sólido y una buena historia. Además, podrá contrastar el origen multidimensional de los requisitos, porque los integrantes aplican tanto el pensamiento paradigmático como el narrativo en la elicitación.

Otra potencialidad de un equipo transdisciplinar es que se le facilita concebir a la organización como un todo, no solamente desde las visiones independientes de usuarios y clientes. Además, está preparado para tener en cuenta las formas no-vivas que interviene en las dimensiones de los requisitos, porque el lenguaje de la cibernética y de la teoría de sistemas faculta a los integrantes para realizar esta integración. Como etapa altamente compleja la elicitación requiere una forma diferente de acercarse a la comprensión no-ambigua de los requisitos, esto consiste en elicitarlos teniendo en cuenta que son fractales, no-lineales, recursivos, sensibles a condiciones diversas y que necesitan retroalimentación.

Por otro lado, la Transdisciplinariedad le permite al equipo construir la elicitación sobre la premisa de que los principios y propósitos discernibles, que subyacen al conocimiento compartido, se pueden explorar directamente desde la naturaleza de las conexiones multidimensionales entre las opiniones, experiencias, posturas y visiones de las partes

interesadas. Este enfoque ayuda a encontrar la axiomática común subyacente al conocimiento disciplinar, a la vez que a coordinar todo lo aprendido hacia el propósito general de darle significado y valor en el documento de la elicitación.

En este proceso es importante que el equipo administre y aplique los principios transdisciplinares y multidimensionales a un nivel de comprensión y penetración, orientado a interpretar las relaciones intrínsecas de las fronteras del problema. Es decir, aprovecharlos para interconectar el saber involucrado y para unir las disciplinas en una elicitación precisa, no-ambigua y puesta en común en el documento final. El proceso de la Gestión del conocimiento en la Etapa de elicitación se ilustra en la Figura 3.

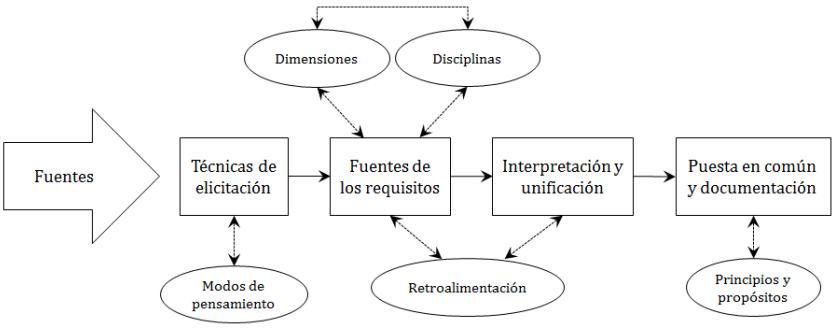


Figura 3. Gestión del conocimiento en la Etapa de Elicitación

3. ETAPA DE DESARROLLO

El objetivo de esta Etapa es realizar un análisis detallado y retroalimentar el documento de la elicitación, para luego comparar los resultados con el carácter técnico del sistema y generar un informe con los requisitos definidos y analizados. La definición de requisitos[5] es una parte crucial de esta Etapa, porque una definición incorrecta, inexacta o excesiva necesariamente dará lugar a retrasos en el plan de trabajo y desperdicio de recursos. Por otro lado, el análisis debe comenzar con los requisitos del negocio, porque es necesario traducirlos en requisitos del proyecto. Cuando no se aplica adecuadamente la Etapa Temprana, elicitar y desarrollar los requisitos puede ser injustificadamente costoso y tomar demasiado tiempo. Cuando se hace bien se puede llegar a los requisitos del proyecto de forma negociada en las conversaciones con las partes interesadas.

Las discusiones sobre qué métodos de definición o de análisis de requisitos son necesarios, deben ser específicas para el tipo de problema y el dominio y el contexto identificados en la Etapa Temprana, porque muchas organizaciones tienen técnicas específicas y probadas para hacerlo, pero pueden no brindar una definición completa y precisa de las necesidades para ese proyecto específico. Mientras que los métodos pueden ser diferentes, los principios siguen siendo los mismos en todo tipo y tamaño de proyecto, deben cubrir todo el alcance del mismo, ser amplios y exhaustivos y tener en cuenta las opiniones y necesidades de todos los interesados. Debido a que los requisitos confusos o no-verificables ni validables se pueden dejar por fuera del ámbito de aplicación de un análisis, es necesario que en esta Etapa se defina con claridad la ambigüedad de los mismos. Por eso se recomienda que los términos para la definición y el análisis de requisitos sean revisados y aprobados por el equipo antes de continuar con el trabajo.

Esto implica comunicación frecuente con los usuarios para determinar las expectativas y características específicas, además de la resolución de conflictos y de la misma ambigüedad en los requisitos. Hay que evitar descripciones excesivas y documentar todos los aspectos específicos del proceso de principio a fin. El objetivo es garantizar que la solución planeada se ajuste a las necesidades del problema, en lugar de intentar moldear las expectativas para adaptar el problema

[5] No significa que hay que definir el término, sino comprender si cada requisito elicitado realmente hace parte del problema y decidir si debe incluirse en la especificación.

a los requisitos. Aquí vuelve a cobrar relevancia la Etapa Temprana, porque la definición y el análisis que se concibe en la Etapa de Desarrollo es un trabajo de equipo, a través de un esfuerzo combinado entre personas y tecnologías en el que los factores multidimensionales y transdisciplinares de los requisitos, identificados en la etapa anterior, se aplican para gestionar el conocimiento acumulado.

Mediante un proceso iterativo se define y analiza los requisitos para determinar sus relaciones e impacto en los procesos del negocio, los sistemas relacionados y las modificaciones posteriores. Con esa información se puede aplicar esfuerzos futuros de diseño para satisfacer la evolución del conocimiento, la integración de sistemas, responder adecuadamente a las necesidades y retos del negocio y ajustar el producto a la fiabilidad esperada. También se debe tener en cuenta que los cambios en el software se producen para fijar fallas, agregar nuevas características o para reestructurarlo y adaptarlo a cambios futuros, pero los cambios en los requisitos es una de las motivaciones más importantes para los mismos. Los requisitos cambian desde el momento en que se elicitan hasta que el sistema se vuelve obsoleto o se retira, lo que refleja que debe ser dinámico para mantenerse útil y seguir siendo competitivo en el contexto de la organización. Al mismo tiempo, estos cambios suponen un riesgo debido a que pueden causar deterioro del software, por lo tanto, los cambios en los requisitos deben ser capturados, gestionados y controlados cuidadosamente para asegurar la supervivencia del sistema. Entonces, el equipo deberá tener en cuenta que suceden por diversas razones:

- Al suponer que el sistema auto-resuelve los cambios sin importar el origen de los requisitos.
- Debido a que al trabajar con el equipo los usuarios entienden mejor sus necesidades, entonces cambian de opinión acerca de lo que quieren que haga el sistema.
- Cuando cambia el entorno del sistema, por ejemplo, el aumento de la velocidad y la capacidad de los computadores afecta sus expectativas.
- Los usuarios descubren nuevas necesidades cuando se libera un sistema que tiene relación con el que se está diseñando.

Si en la Etapa de Elicitación no se superaron las barreras de comunicación entre las partes interesadas y el equipo, podría surgir un problema para definir y analizar adecuadamente los requisitos. Por eso se recomienda tener en cuenta el principio de la *dualidad básica entre partes*, que son a la vez distintas pero interconectadas, y que los métodos analíticos actuales por sí solos no logran identificar. Además, debido a que los requisitos se mezclan entre sí, en la elicitación es difícil describir su comportamiento con los métodos tradicionales, porque un cambio en alguno de ellos se propaga a través de un complejo tejido de interacciones con los otros, lo que no permite rastrear el comportamiento global del sistema en términos de los requisitos aislados. La función del equipo es identificar y analizar lo que se ha elicitado.

Otra herramienta que podría ser de utilidad es la concepción ontológica de las *realidades espaciales*, porque si el equipo logra definir las características objetivables de los requisitos podría identificar y descomponer la elicitación en sus componentes multidimensionales, además, establecer las múltiples relaciones al interior y el exterior de la elicitación, a la vez que re-conocer la multi-dependencia de los requisitos. Analizar los requisitos desde esta concepción permite comprender el sistema como una realidad ontológica, en la que cada requisito tiene dimensiones y relaciones propias, a la vez que se integra multidimensionalmente con los demás, con el contexto y con el dominio.

Esta comprensión total del mundo (sistema) a partir de la definición y el análisis de requisitos le permite al equipo identificar el conocimiento necesario para comenzar a estructurar una de las actividades más importantes en el ciclo de vida del software: las pruebas. Por eso, la

recomendación en el modelo es que luego de elicitar los requisitos e identificar los que podrían hacer parte de una solución, se estructure un plan de pruebas para los mismos. No se puede esperar a hacerlo como en otros modelos, donde se planifica luego que la solución está materializada en un producto. Esto podría generar, y de hecho lo hace, reprocesos en el futuro, que generalmente retroceden hasta la elicitación. Para evitarlo es necesario descomponer la información del documento de elicitación para inferir el conocimiento multidimensional implícito y explícito relevante para la definición de requisitos.

Estos principios permiten construir un espacio multidimensional en el que las distancias entre los requisitos se convierten en funciones del trayecto medido desde la visión de las partes interesadas y la del equipo. En la medida en que se utilice este sistema de medición, la escala de separación entre ambas visiones se convierte en una variable fácil de analizar, interpretar y controlar, antes que los requisitos pasen a la Etapa de Gestión.

Alcanzar esto no debería ser complicado, porque el equipo se conformó de forma transdisciplinar y sus integrantes son capaces de desarrollar y aplicar pensamiento multidimensional. Por su parte, el principio de la eco-organización también ayudaría a comprender las dependencias internas y externas entre los requisitos, además de entre ellos y el contexto y el dominio. Porque en la definición se necesita conocer las posibles entradas y salidas de cada uno, un conocimiento necesario para estructurar el plan de pruebas.

De acuerdo con el Pensamiento Complejo, el pensamiento multidimensional ayuda a ubicar lo impensado, sobre los requisitos en este caso, para analizar y encontrar el conocimiento de su definición. Esto implica que el equipo debe aplicar las formas disciplinares de ver y conocer, a la vez que como transdisciplinar lo entiende y comprende, de tal forma que pueda re-conocer cada requisito con base en lo que sabe de él. De esta forma puede concientizar a las partes interesadas y derrumbar muchas de las interpretaciones erróneas, que puedan haber surgido hasta el momento, sobre los requisitos y una posible solución del problema. Esta habilidad para interconectar las diferentes dimensiones de la realidad es necesaria en la Ingeniería de Requisitos para crear los mapas mentales y realizar las abstracciones, que le permiten al equipo construir un prototipo no-funcional.

Este prototipo constituye el primer acercamiento a una propuesta de solución al problema y existen diversas formas de representarlo, pero la más cercana al conocimiento adquirido hasta este momento es a través de interfaces de usuario[6]. Todavía no hay diseño ni código, por lo que estas interfaces son las que utiliza el equipo para demostrarles a las partes interesadas que han comprendido su problema y que podrán presentarle una solución. Para este propósito la recomendación es aplicar el principio de los niveles de realidad que describe el Pensamiento Complejo.

El primero de estos niveles es *el estado de las cosas*, que se alcanzó en la Etapa anterior cuando el equipo y las partes interesadas lograron un acuerdo en cuanto al lenguaje y la elicitación de los requisitos. Ahora se continúa con el segundo nivel, *los estados de las cosas inseparables del proceso del conocimiento*. En este momento, y a partir del conocimiento que se tiene del estado de la definición y análisis de los requisitos, es posible identificar su valor y relaciones en un contexto de solución, para darle cuerpo al prototipo no-funcional. El tercer nivel, *los estados de las cosas creadas en relación con el proceso del conocimie*nto, se alcanza cuando el equipo modela el prototipo a partir de una interpretación abstracta y lógica de la definición de los requisitos y el conocimiento adquirido. Esta representación es una imagen que refleja la totalidad de los niveles de realidad y el conocimiento acumulado hasta el momento, cuya gestión en la Etapa de Desarrollo se presenta en la Figura 4.

[6] Se refiere a lo que comúnmente se denominan *pantallas* del sistema y sobre las que los usuarios desarrollan sus actividades.

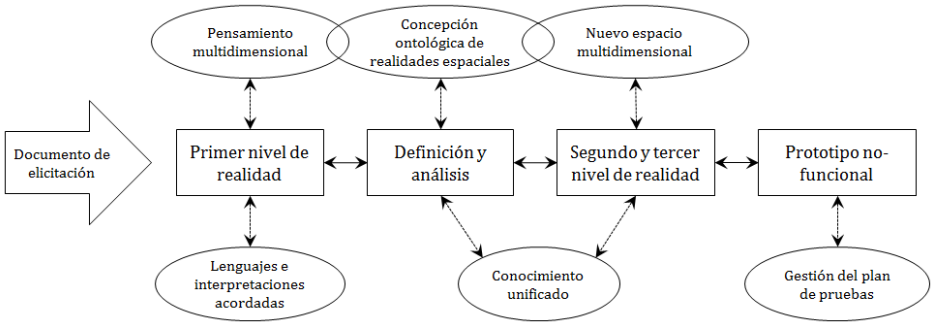


Figura 4. Gestión del conocimiento en la Etapa de Desarrollo

4. ETAPA DE GESTIÓN

Debido a la complejidad y volatilidad de los requisitos todavía pueden ser confusos y no se pueden verificar[7] ni validar[8]. Esto se debe a que aún se encuentran requisitos indefinidos en la documentación y hay que gestionarlos para tomar decisiones. La falta de definición y gestión del cambio de requisitos es la causa más frecuente de fracasos en el desarrollo de software, pero, aunque la industria, los profesionales y los investigadores son conscientes de ello, gestionarlos adecuadamente es difícil. La causa más común es el manejo inadecuado e ineficiente de los cambios, en parte porque la elicitación ocurre en ambientes informales y su documentación permanece estática durante el ciclo de vida de desarrollo, incluso cuando los requisitos cambian. Esto conduce a confusión, planificaciones perdidas, sobrecostos y usuarios insatisfechos, por lo tanto, el objetivo debe ser adoptar y gestionar el cambio, no evitarlo.

El propósito de la gestión de requisitos es garantizar que la solución en desarrollo valida, verifica y responde con fiabilidad a las necesidades y expectativas de las partes interesadas. De ahí que la propuesta en el modelo es diseñar, planificar y ejecutar procesos desde el análisis de requisitos y la gestión del cambio, a través del trabajo colaborativo y la aplicación de una Gestión del conocimiento eficiente y eficaz. Con esto se busca encontrar trazabilidad para reportar los avances del proyecto en términos de cumplimiento, progreso, cobertura y consistencia de los requisitos. La trazabilidad[9] también apoya la gestión del cambio como parte de la gestión del conocimiento, en la comprensión de su impacto en los requisitos u otros elementos relacionados, y para facilitar la introducción de los mismos.

En esta Etapa el equipo debe volver a hacerse las preguntas *¿qué queremos hacer?* y *¿cómo podemos hacerlo?* que respondió antes en la elicitación. Si en este momento sus respuestas no coinciden con las que encontró previamente, muy seguramente es porque el requisito es confuso, por lo tanto, debe regresar a la elicitación, ya que no puede ser verificado ni validado. Utilizar adecuadamente el lenguaje de la planificación del nivel normativo es fundamental para encontrar estas discrepancias, porque los cambios en la planificación inicial son tan pequeños que pueden pasar inadvertidos, pero sus efectos pueden ser desastrosos en fases posteriores del ciclo de vida.

Por otro lado, la Gestión del conocimiento se necesita para verificar y validar que los requisitos desarrollados son los que realmente apuntan a una solución fiable del problema. Además, permite identificar requisitos faltantes y asegurar que cada uno de los seleccionados reúnen las características de fiabilidad necesarias:

- *Correcto*: establece con precisión una necesidad.
- *Claro*: solamente tiene un significado posible.
- *Factible*: puede ser implementado dentro de las restricciones conocidas.

[7] Consiste en responder a la pregunta: *¿el requisito es necesario para solucionar el problema?*
[8] Es responder a la pregunta: *¿el requisito responde a una necesidad del cliente o del sistema?*
[9] La trazabilidad se ocupa de documentar la vida de cada requisito.

- *Modificable*: se puede cambiar fácilmente.
- *Necesario*: representa algo que realmente se necesita.
- *Priorizable*: clasificable de acuerdo con la importancia de su inclusión.
- *Trazable*: se le puede hacer seguimiento a su vida útil.
- *Consistente*: no presenta inconsistencia internas ni externas.
- *Verificable*: su correcta implementación se puede determinar por medio de pruebas, análisis, inspecciones o demostraciones.

El equipo debe comprender que ese conocimiento se adquirió a través de la experiencia del trabajo en las etapas previas y que es intrínsecamente indeterminado, por lo que ya debe haber declarado cómo gestionarlo. Además, debe concebirlo como un objeto activo que está embebido en las prácticas, experiencias, disciplinas e interpretaciones de las partes interesadas y sus interrelaciones.

La fiabilidad del software se analiza desde tres aspectos: 1) calidad funcional, 2) calidad estructural, y 3) calidad de procesos, pero, aunque ya se viene estructurando desde la elicitación, en esta Etapa debe originar los primeros indicadores de logro. Por eso es que el equipo debe lograr el cumplimiento de la elicitación, una cantidad mínima de errores, un prototipo fácil de aprender y usar (que luego será el sistema) y requisitos que cumplan todas las características necesarias. Asimismo, que la Gestión de conocimiento y de los requisitos cumple el calendario, está dentro del presupuesto y es repetible. Con esto en mente el equipo diseña y presenta los escenarios de prueba[10], que se convertirán en un componente sustancial del documento de la especificación.

Identificar estos primeros indicadores de la fiabilidad del software es una actividad que estimula la comprensión del conocimiento común discernido y, para añadirle solidez a la gestión de requisitos, el equipo se pregunta colectivamente *¿qué otras cosas se pueden aprovechar desde las disciplinas involucradas?* Esto implica adicionarle a la Gestión del conocimiento los principios de interdependencia de las partes y el todo, de adaptabilidad sobre la base de la estabilidad subyacente, identificar y aprovechar el conocimiento disciplinar oculto hasta el momento, permitir solamente cambios necesarios, blindar la gestión a los perjudiciales y encontrar los componentes estructurales cuyas funciones complementen la gestión. Aceptando que en la Ingeniería de Requisitos hay momentos en los que vale la pena ceder especificidad a favor de la integralidad de los requisitos, pero sin que al hacerlo se pierda control sobre los indicadores de fiabilidad y la gestión del conocimiento, cuyo proceso en la Etapa de Gestión se presenta en la Figura 5.

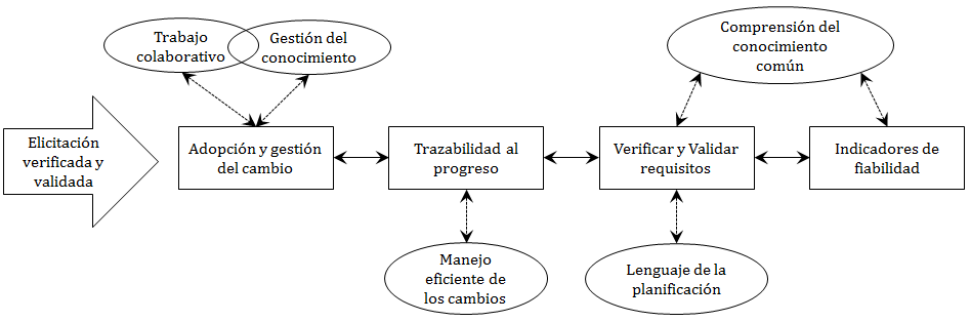


Figura 5. Gestión del conocimiento en la Etapa de Gestión

5. ETAPA DE ESPECIFICACIÓN

Luego de desarrollar las etapas anteriores del modelo se puede preparar el documento en el que se describe cada uno de los requisitos definidos, verificados y validados. Esto no quiere decir que

[10] Son estructuras en las que, utilizando los posibles valores de entrada y de salida en el cumplimiento de los requisitos, se simula posibles escenarios de respuesta para detectar posibles errores de concepción del requisito, antes que evolucionen a fallos en el software.

esos sean los requisitos definitivos, porque su volatilidad y dinámica no permiten llegar a esta conclusión. Una actualización normativa, un cambio de administración, nuevas leyes, cambios estructurales en el modelo de negocios, nuevas disposiciones nacionales (reformas tributarias) o internacionales (tratados de libre comercio), o nuevas propuestas en la Ingeniería de Requisitos, pueden ocasionar que se tengan que eliminar, modificar o incluir nuevas necesidades en la especificación. Pero, siguiendo el modelo que se describe aquí, el equipo podrá estar seguro que los inconvenientes serán mínimos.

La especificación de requisitos es la actividad encargada de determinar qué funcionalidades del sistema se pueden desarrollar con los requisitos aprobados, además, describe completamente qué hará el software y cómo se espera que lo lleve a cabo. Su objetivo es minimizar el tiempo y el esfuerzo en las demás fases del desarrollo, porque entrega una base sólida sobre la comprensión del problema. Un buen documento de especificación define cómo interactuará la aplicación con la arquitectura del sistema, y con otros programas y usuarios, en una amplia variedad de situaciones del mundo real. También se utiliza para evaluar parámetros tales como velocidad de operación, tiempo de respuesta, disponibilidad, portabilidad, facilidad de mantenimiento, trazabilidad, seguridad y velocidad de recuperación de eventos adversos.

En el contenido de este libro se describe los patrones de conocimiento que expresan las partes interesadas y el equipo de trabajo sobre sus necesidades, experiencias y puntos de vista sobre los requisitos. Nada se ha inventado, todo está sustentado y dialogado desde la Etapa Temprana. Se ha roto las barreras disciplinares y a través de la Transdisciplinariedad se ha gestionado el conocimiento que se materializa en el documento, pero que, igualmente, deja un conjunto de prácticas y vivencias que nutren la experiencia de cada individuo participante.

También se valida el hecho de que los principios del Pensamiento Complejo le aportan precisión a la gestión de la Ingeniería de Requisitos, porque el documento de la especificación es un reflejo de los niveles de realidad y Multidimensionalidad de los requisitos. Esto se refleja en el hecho de que las personas que intervienen desarrollan pensamiento trascendente, en el sentido de que ahora pueden mirar un antes, un ahora y un después del problema y viven un estado de auto-conciencia inmanente. El equipo establece un patrón de gestión del conocimiento en el que lo más importante es buscar *conocimiento por conocimiento*, de tal manera que cuando se presente modificaciones en las necesidades de las partes interesadas, incluirlas en este documento solamente generará complicaciones menores.

El documento de especificación reúne los productos de las demás etapas, es decir, desde la Etapa Temprana se inició su edición con el documento de descripción del problema, luego con el de la elicitación, el plan de pruebas, la definición y el análisis de requisitos, la verificación y validación de requisitos, y los escenarios y casos de prueba. Este análisis de integración se convierte en el insumo de la gestión de la Ingeniería de Requisitos para la Ingeniería del Software. En este modelo la especificación de requisitos se alcanza con la integración de principios del Pensamiento Complejo en las etapas que conforman la base. De esta manera se puede aportar al logro de la fiabilidad del software, porque, como proceso iterativo e interactivo, le ha permitido al equipo tener una visión diferente de lo que necesitan las partes interesadas y de cómo solucionar su problema.

Este capítulo contiene las respuestas a las preguntas de investigación y el cumplimiento de los objetivos planteados. Se parte de un análisis a los métodos que tradicionalmente se aplican en esta fase del desarrollo del software. Llegando a la conclusión de que no son suficientes para atender la complejidad de los problemas actuales, porque presentan múltiples niveles, ciclos de retroalimentación y dependencias que dan lugar a requisitos multidimensionales provenientes de diversas disciplinas. Por eso se necesita nuevos principios con los que se puedan interconectar diferentes dimensiones de la realidad, desde las cuales provienen los requisitos. En ese sentido, el

Pensamiento Complejo es tanto una estrategia como una forma de pensar, que se propone para interpretar fenómenos globales a la vez que reconoce la especificidad de sus partes constitutivas.

Para el cumplimiento del primer objetivo de la investigación: *determinar la forma en que la Transdisciplina y la Multidimensionalidad se pueden integrar en un modelo para gestionar la Ingeniería de Requisitos*, se construyó una definición de cada uno de estos principios desde una perspectiva ingenieril. Por lo tanto, para esta disertación, Multidimensionalidad se acepta como una conciencia y comprensión expandida que proporciona una profunda visión de la realidad, del mundo y de cada situación, que permite alcanzar simultáneamente un punto de vista en el que se integra los diferentes tipos y niveles de información presentes en los requisitos. Por otro lado, se parte de que en la Ingeniería de Requisitos todos los esfuerzos y actividades humanas tratan siempre de llegar a la comprensión de las interconexiones entre los diversos campos del saber, para especificar los requisitos. Por lo tanto, Transdisciplinariedad se construye sobre la premisa de que hay principios y propósitos discernibles que subyacen a todo el sistema conocimiento/conocedor involucrado. De esta forma se acepta como un puente para unir: 1) el conocimiento proveniente de cada disciplina involucrada, sobre el nivel de los principios subyacentes comunes necesarios para comprender las necesidades que debe satisfacer el sistema, y 2) sobre el nivel de su misma comunalidad, como expresiones de la conducta humana con un propósito definido.

En el caso del segundo objetivo: *encontrar las interrelaciones lógicas entre estos principios y los objetivos de las etapas de la Ingeniería de Requisitos*, la Multidimensionalidad se observa como herramienta epistemológica para: 1) realizar una evaluación crítica de la situación mediante la comprensión de los múltiples puntos de vista; 2) juntar múltiples puntos de vista conflictivos y contradictorios para el logro del objetivo de cada etapa; 3) averiguar las interdependencias entre los actores (partes interesadas) y las variables (requisitos); y 4) desarrollar una visión completa de la situación a partir de las múltiples dimensiones intervinientes. Esto crea una necesidad que se debe satisfacer obligatoriamente: el equipo debe desarrollar y aplicar pensamiento multidimensional, es decir, deber ser capaza de ver el problema completo, así como el sistema en el que está inmerso. Además, los ingenieros deben ser capaces de involucrar y aprovechar los modos de pensamiento: el lógico-científico (paradigmático) y el narrativo. Desde la visión de la Transdisciplinariedad lo que el equipo debe hacer es estimular la comprensión del conocimiento común discernido, y preguntarse colectivamente *qué otras cosas se pueden aprovechar* desde las disciplinas de los integrantes, para añadirle solidez a la discusión y el análisis de los requisitos.

La respuesta a la primera pregunta de investigación: *¿qué relación se puede encontrar entre Transdisciplinariedad, Multidimensionalidad e Ingeniería de Requisitos?* se estructura a partir de las características dimensionales y disciplinares de los requisitos. Desde las primeras se aprovecha cualidades como que son no-lineales, fractales, simétricos, sensibles y retroalimentados. Mientras que desde las disciplinares son interdependientes, adaptables, individuales, homeostáticos, empotrados, sociables y complementarios. Entonces y mediante el conocimiento del estado epistemológico de los principios de la Ingeniería de Requisitos, la Transdisciplinariedad y la Multidimensionalidad se encuentra una relación entre ellos. La premisa es que como principios pueden: 1) describir los patrones de conocimiento que expresan los usuarios, 2) complementarse para darle mayor precisión a los principios ingenieriles, 3) ofrecer una manera de que la capacidad humana experimente la trascendencia.

La respuesta a la segunda pregunta: *¿cómo integrar los principios del Pensamiento Complejo (Transdisciplinariedad y Multidimensionalidad) para gestionar la complejidad del conocimiento en la Ingeniería de Requisitos?* se encuentra inmersa en cada una de las etapas del modelo de gestión estructurado. En ellas se presenta una descripción detallada de cómo se integran para alcanzar los objetivos de cada una. El resultado es un producto más consolidado en pro de ayudar a mejorar la fiabilidad del producto final de la Ingeniería de Requisitos: la especificación de requisitos.

Para responder a la tercera pregunta: *¿en qué medida se puede verificar que con un modelo de este tipo sería posible mejorar la fiabilidad de los productos software?* en términos generales el examen de los resultados para verificar su fiabilidad y validez pasa por el proceso de la objetividad y la credibilidad de la investigación realizada, en el que validez se refiere a honestidad y autenticidad de lo logrado, mientras que la fiabilidad a la reproducibilidad y a la estabilidad de lo propuesto. Para verificar la integración se requiere una virtud: paciencia, porque la evidencia muestra que aplicar un modelo en el que se integre principios no-ingenieriles requiere tiempo y compromiso de la academia, la industria y la investigación. Además, porque asimilar, comprender y aplicar el lenguaje de las disciplinas ingenieriles tampoco es fácil. Otra cuestión es que ninguna ciencia ha tenido éxito haciendo contención del conocimiento, porque se filtra en todas direcciones a través de las instituciones, las estructuras y las personas. Lo que se necesita es mente abierta para darle una oportunidad a las propuestas innovadoras y diferentes a las tradicionales en un campo de estudio. Por lo tanto, en modelos como el que aquí se propone los principios de Transdisciplinariedad y Multidimensionalidad transgreden los límites disciplinares, por lo que la Ingeniería de Requisitos debe integrarlos para mejorar la comprensión de lo que conocen las partes interesadas.

6. VALIDACIÓN Y VERIFICACIÓN DEL MODELO

La Verificación y Validación V&V de modelos es una metodología que le permite al creador encontrar las certezas y los errores de su propuesta. La cuantificación de la confianza y la exactitud del modelo proporciona la información necesaria para la toma de decisiones sobre su aplicabilidad y seguimiento. Pero, cualquiera que sea el paradigma de modelado o la técnica de solución que se aplique, las medidas de rendimiento extraídas solamente tendrán que ver con el sistema real si el modelo es una buena representación del mismo. Por supuesto, lo que constituye un buen modelo es subjetivo, pero desde un punto de vista lógico los criterios para juzgar sus bondades se deben basar en que los resultados de V&V corresponden a los esperados de su aplicación real.

En este sentido la exactitud del modelo se observa como la ausencia sistemática y aleatoria de errores, lo que se denomina fidelidad y precisión. Algunos autores han presentado trabajos interesantes sobre la verificación y validación de modelos, tales como Doucet y Sloep (1992), quienes concluyen que la verificación debe demostrar que el modelo es verdadero y la validación que es digno de crédito. McCullagh y Nelder (1989) destacan algunos principios básicos que se debe tener en cuenta: 1) todos los modelos son erróneos, aunque algunos son más útiles que otros, 2) no se debe considerar al modelo como el único en su género, y 3) es necesario comprobar cuidadosamente que se ajusta a lo esperado. Por su parte, Law y Kelton (2000) examinan algunas técnicas para incrementar la validez y la credibilidad de los modelos y señalan que muchos no se pueden verificar y validar como un todo, en parte porque su contexto de aplicación es complejo y el proceso requiere un tiempo prolongado. Por eso recomiendan hacerlo individualmente por cada componente, de forma incremental y aplicando un seguimiento *top-down*.

Para el caso del modelo presentado en esta investigación, y debido a que clasifica como aquellos que no se recomienda verificar y validar como conjunto, el procedimiento consta de varios momentos:

1. *Validez aparente*. Consiste en solicitarle a personas con conocimientos sobre el contenido si el modelo y/o su comportamiento son razonables. Para el caso del modelo presentado este momento ya se cumplió, y se recibieron aportes y sugerencias de varios especialistas en la temática del mismo. La conclusión es que es lógico y viable, pero que el proceso de aplicación y contrastación requiere varios semestres en la academia y por lo menos dos años en la industria.
2. *Validación predictiva*. Mediante aplicación simulada el modelo se utiliza para predecir (pronosticar) su comportamiento y los valores de respuesta. Con estos resultados se realiza los ajustes necesarios al modelo y se pone a punto para el siguiente momento. Para el caso del

modelo propuesto, y debido a la dificultad de verificarlo y validarlo completamente, se dividió en los componentes que se describen en la Figura 1.

El proceso de V&V del modelo se debe llevar a cabo en paralelo con los análisis de los resultados y las modificaciones necesarias, además, es necesario contar con los recursos humanos y de calendario suficientes para realizar varias iteraciones. En la Figura 6 se muestra el proceso estructurado para verificar y validar el modelo propuesto en esta investigación. El centro de estas actividades es la recopilación y análisis de los datos hallados en la aplicación de cada Etapa, porque influyen directamente en la toma de alguna de estas decisiones: 1) ajustar los parámetros asociados para alcanzar mayor comprensión del comportamiento observado; 2) rediseñar el componente con base en la información obtenida, a partir del análisis de los datos de verificación y validación; o 3) acreditar la aceptabilidad del modelo con respecto al componente analizado.

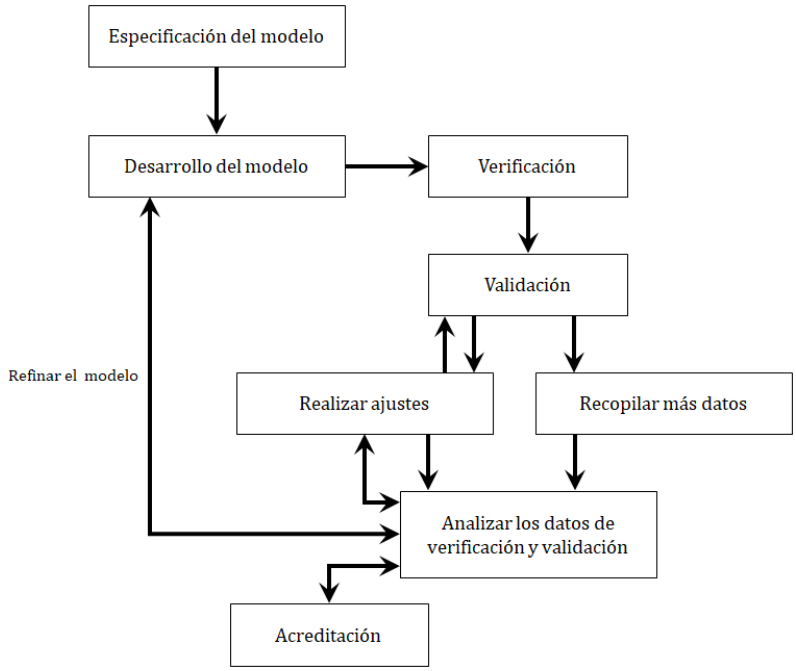


Figura 6. Proceso de V&V para el modelo propuesto (Adaptado de Law, 2007)

VIII
CONCLUSIONES

El principal papel crítico de la ciencia ha sido desafiar las concepciones arraigadas de la razón humana e introducir nuevos métodos de investigación para comprender la naturaleza de las cosas. Una vez más, y con mayor necesidad que nunca antes, los científicos, pensadores críticos e ingenieros que estructuran e introducen nuevos conceptos en el ámbito científico de este siglo, tienen a su alcance el material y los principios para iniciar procesos innovadores. Los principios del Pensamiento Complejo, tales como Transdisciplinariedad y Multidimensionalidad, producirán una nueva generación de análisis intelectual y físico, además de herramientas y técnicas de síntesis, que llevarán a la ciencia a nuevos horizontes. Estas herramientas y técnicas serán la forma en que la ciencia, la ingeniería y la Gestión del conocimiento se constituyan en una entidad transdisciplinar unificada al servicio de la humanidad. El desarrollo de las Ciencias Computacionales y la Era Digital están ayudando a marcar ese comienzo, porque facilitan la introducción de la gota que le falta a la copa del conocimiento para derramarse, de tal forma que una vez más cambie permanentemente nuestra forma de pensar sobre la naturaleza.

Para mantener un control intelectual sobre la expansión de la información y la gestión del conocimiento, desde un punto de vista pragmático se necesita estudios sistemáticos de los procesos científicos. Además, el estudio sistemático de los procesos ingenieriles permitirá superar las deficiencias de las bases y de los métodos de la ingeniería tradicional clásica cartesiano-mecanicista, facilitando la integración entre disciplinas. Actualmente, los ingenieros son integradores de sistemas y desarrollan artefactos y servicios funcionales, teniendo en cuenta aspectos económicos, ambientales y éticos en la resolución de problemas. Por eso requieren un carácter transdisciplinar de diseño y utilización de procesos, y que para el caso del software podría ser valioso para mejorar su fiabilidad general, productividad y seguridad.

Un área de estudio particular se puede llamar disciplina siempre que haya unificado sus herramientas, técnicas y métodos, además de poseer una terminología bien desarrollada. Tradicionalmente, las disciplinas se han desarrollado en *conchas* independientes y tienden a minimizar la interacción con entidades externas o con otras disciplinas. Cuanto más tiempo evoluciona más resistente será su caparazón, y sus practicantes desarrollarán un nivel efectivo de comunicación interdisciplinar. En este proceso lineal tienden a minimizar la comunicación y a desarrollar territorios que defienden con fiereza. Obviamente estos territorios y conchas disciplinares no están en sincronía con el trabajo que realizan los ingenieros para especificar los requisitos del software, y lograr una adecuada integración de los sistemas, porque los problemas que deben resolver no están restringidos a esos límites artificiales.

En la era de la mentalidad cartesiano-mecanicista este enfoque sirvió para solucionar los principales problemas, sin embargo, los tiempos han cambiado y los problemas que se pueden resolver con software son complicados y complejos, por lo que las soluciones requieren masivas infraestructuras de comunicación y de computación, que utilizan equipos de trabajo cada vez más grandes. Pero una inadecuada comunicación interdisciplinar y las preocupaciones territoriales de los integrantes limitan la eficacia de su trabajo y, por lo tanto, de la calidad de sus soluciones.

La Ingeniería de Requisitos es la fase con la que comienza el ciclo de vida del proceso de la Ingeniería del Software. En ella se elicitan, comprenden y especifican las necesidades de las partes

interesadas para desarrollar productos software que las satisfagan. Las actividades de esta fase merecen mayor atención en las prácticas industriales y académicas, por lo que en esta investigación se propone un modelo para gestionar la Ingeniería de Requisitos, con la hipótesis de que trabajando de esta manera se podría ayudar a mejorar la fiabilidad del software.

En la propuesta se reúne las buenas prácticas de la Ingeniería de Requisitos, los métodos formales, la lógica, la abstracción y otras que tradicionalmente se ha utilizado para llevar a cabo esta fase del ciclo de vida. Pero para atender la complejidad de los problemas actuales se debe integrar principios nuevos y visiones diferentes para gestionar los requisitos como un proceso de creación, descubrimiento y Gestión del conocimiento. Esto se debe a que los problemas actuales son diferentes, la seguridad se ha convertido en un asunto clave, las modificaciones son más cotidianas y los tiempos de entrega se acortan drásticamente, a la vez que los presupuestos se deben ajustar a una vida útil muy corta.

Por otro lado, la investigación cualitativa a menudo es criticada como sesgada, a pequeña escala, anecdótica y/o carente de rigor. Sin embargo, cuando se lleva a cabo correctamente es imparcial, profunda, válida, confiable, creíble y rigurosa. Por eso, en este tipo de investigación es necesario que haya una forma de verificar el grado en el que se interviene una realidad, o la veracidad de lo que se propone como resultado de la misma. En esta investigación el examen de los resultados para verificar su fiabilidad y validez pasa por un proceso de objetividad y credibilidad del trabajo realizado, en el que la validez se refiere a la honestidad y autenticidad de lo logrado, mientras que la fiabilidad se refiere a la reproducibilidad y a la estabilidad de lo propuesto.

En este caso, la validez se toma como la medida en que los resultados son una representación de los fenómenos que están destinados a representar, mientras que la fiabilidad está motivada por la triangulación de datos contradictorios, la validación demandada y la comparación constante entre lo propuesto y lo existente. Las pruebas contradictorias, es decir, los casos divergentes, se buscaron, examinaron y analizaron para asegurar que el sesgo del investigador no interfiriera o alterara su percepción y conocimientos aportados en el proceso.

Por eso es que la verificación demandada y que permite la lectura y el análisis del modelo, se refleja en la interpretación de las respuestas a las preguntas. Además, proporciona un método de comprobación de inconsistencias, desafía supuestos y proporciona una oportunidad para volver a analizar los datos. El uso de la comparación constante en el análisis ofrece solidez y no se considera en sí mismo como integradora, permitiéndole al investigador tratar los datos como un todo, en lugar de fragmentarlos. Además, permite identificar temas no previstos o emergentes dentro del proyecto, que constituyen trabajos futuros.

A continuación, se da respuesta a la tercera pregunta de investigación: ¿*En qué medida se puede verificar que con un modelo de este tipo sería posible mejorar la fiabilidad de los productos software*? Pensamiento Complejo, Transdisciplinariedad y Multidimensionalidad son temas recurrentes en muchas publicaciones científicas. Actualmente, responden a una necesidad subyacente y a una creencia inherente, orientadas básicamente a los sistemas de educación. Conforman una expectativa que contribuye a buscar soluciones conjuntas a la complejidad de los problemas, en los que se yuxtaponen diferentes disciplinas, dimensiones e interpretaciones. Pero sabiendo que ninguna disciplina sabe más que todas juntas y que si el objetivo es solucionar conjuntamente esos problemas, se necesita propuestas, metodologías y modelos que integren diversas perspectivas en la identificación, formulación y resolución compartidas.

Para verificar esa integración se requiere una virtud: paciencia. La evidencia muestra que aplicar un modelo en el que se integre principios no involucrados tradicionalmente en la ingeniería, toma tiempo y compromiso por parte de la academia, la industria y los investigadores. El hecho es que asimilar, comprender y aplicar el lenguaje de otras disciplinas ingenieriles toma tiempo, aunque

también hay que entender que, en este caso, es apremiante mejorar la fiabilidad del software. Otra cuestión es que no se ha tenido éxito en ninguna ciencia haciendo contención del conocimiento, porque se filtra en todas direcciones a través de las instituciones, las estructuras y las personas. Lo que se necesita es mente abierta para darle una oportunidad a las propuestas innovadoras y diferentes a las tradicionales en un campo de estudio. Por lo tanto, en modelos como el que aquí se propone los principios de la Transdisciplinariedad y la Multidimensionalidad tratan de transgredir los límites disciplinares, mientras que el Pensamiento Complejo busca integrarlos para mejorar la comprensión de lo que se conoce.

Otro atributo de la investigación contemporánea en ingeniería es que se lleva a cabo en el contexto de la aplicación, es decir, los problemas se formulan desde el inicio a través del diálogo entre un amplio número de actores y perspectivas disciplinares. Esa comprensión también requiere paciencia, porque el análisis y el establecimiento del problema no se formulan por fuera del grupo, y hasta no acordar en cómo se llevará a cabo no se tiene un proyecto que requiera recursos y actividades de investigación. Entonces se inicia un proceso en el que las partes interesadas aportan una heterogeneidad esencial de habilidades y conocimientos necesarios para encontrar una solución. Además, también tiende a aparecer estructuras organizativas libres, jerarquías planas y cadenas de mando abiertas.

En contraste, las universidades son precisamente el polo opuesto y en general son altamente jerárquicas, fijas y con estructuras disciplinares. Por eso es que no es común encontrar investigación práctica entre universidades e industrias en el área del software, porque el objetivo de las primeras es incrementar el acervo del conocimiento, mientras que las segundas necesitan producción inmediata. Por lo tanto, la validación del modelo propuesto en esta investigación necesita paciencia y tiempo para alcanzar a los resultados esperados.

También hay que tener en cuenta que esta investigación es transdisciplinar, aunque muy pocas se cataloguen como tal, y en el proceso se estructuraron foros y plataformas que proporcionaron un espacio para armonizar y potencializar el esfuerzo intelectual, pero de forma muy diferente a las estructuras disciplinares tradicionales. Por otro lado, en la Ingeniería de Requisitos es común que los equipos desarrollen esfuerzos intelectuales para comprender las necesidades del sistema, pero en gran medida surgen de dentro de las disciplinas. En este modelo todavía se puede empezar a trabajar de esa forma, aunque en el desarrollo de las etapas la Multidimensionalidad y la Transdisciplinariedad surgirán casi por derecho propio. Además, al ser un proceso de investigación práctica ese esfuerzo intelectual surge y se desarrolla naturalmente, donde una aplicación lleva a la otra, y así sucesivamente. Pero lo que no puede suceder es lo que tradicionalmente ocurre cuando se habla de Transdisciplinariedad en ingeniería, donde los protagonistas son los obstáculos que se tiene que afrontar en su utilización, sobre todo en lo relacionado con la fiabilidad de los productos.

A este respecto tampoco se puede olvidar que los requisitos son dinámicos y volátiles, y que el lenguaje natural es ambiguo. Por eso se requiere Pensamiento Complejo para aplicar los principios de la Multidimensionalidad y la Transdisciplinariedad para gestionar el conocimiento transgresivo, que no respeta fronteras institucionales ni disciplinares. En la Ingeniería de Requisitos se presenta una especie de convergencia o co-evolución entre lo que ocurre en el ámbito de la producción de conocimiento y cómo lo gestionan los equipos de trabajo. En esta fase del ciclo de vida no hay de antemano una estructura clara de orden, por lo que la gestión de los requisitos se debe realizar mediante modelos adaptativos, con diferentes actores y funciones tomadas desde su experiencia y disciplina, porque el proceso es en sí mismo un Sistema Adaptativo Complejo. Así se ha pensado la estructura del modelo que aquí se describe, porque tiene en cuenta principios ajenos a la ingeniería, pero que, por el mismo desarrollo tecnológico y la complejidad de los problemas, es necesario involucrarlos para gestionar de mejor forma la Ingeniería de Requisitos.

De esta manera se podrían atender dos necesidades apremiantes del desarrollo de software: la seguridad y la fiabilidad. La sociedad actual tiene una alta dependencia de este producto tecnológico, y la Ingeniería del Software, como metodología que orienta su producción, tiene la obligatoriedad de rendir cuentas y ser responsable al desplegarlo. En el equipo debe estar claramente definido que, aunque cada integrante tiene una ética de la responsabilidad individual y desde su disciplina, es necesario unificar un tipo de responsabilidad grupal para atender la necesidad social de un producto fiable.

Por otro lado, la fiabilidad del software es un criterio difícil de alcanzar, porque el software es un producto intangible de la ingeniería que no se fabrica, sino que se desarrolla. Esto incrementa su complejidad y no se puede aplicar los mismos criterios de control de calidad que para otros productos ingenieriles tangibles. En el entorno de la Ingeniería de Requisitos surgen demandas especiales y únicas en cuanto a la fiabilidad, por lo que el conocimiento que se genera se debe gestionar en pro de alcanzar el mejor nivel.

Por eso se requiere los principios del Pensamiento Complejo, para que el equipo tenga herramientas nuevas para gestionar los requisitos y aportar en gran medida al logro de la fiabilidad buscada. Además, porque la gestión de requisitos tiene ingredientes que van más allá de la excelencia del equipo de trabajo y que se presentan en formas difíciles de entender, porque el contexto varía a todo momento. Esto le permite al equipo llegar a conclusiones sobre qué hay que aplicar y respetar para cada una, pero no en términos del proceso global, sino de forma parcial. Ese es el papel del Pensamiento Complejo, porque a través de sus principios el equipo puede concretar estos criterios como valor agregado para alcanzar la fiabilidad.

En la práctica el proceso debe ir más allá de un simple valor añadido a la gestión, y alcanzar, aplicar y proyectar un valor integral. El potencial de los principios de la Multidimensionalidad y la Transdisciplinariedad reside precisamente ahí, porque al integrarlos en un modelo de gestión de requisitos se abre la posibilidad para que el equipo de trabajo pueda atender responsablemente los requisitos de fiabilidad del producto.

Por otro lado, la Multidimensionalidad de los requisitos no puede abordarse como un asunto más en la comprensión del problema. Además de las personas y su conocimiento, todas las dimensiones involucradas se deben valorar y ubicar con certeza para entender el contexto y las posibles disciplinas involucradas. Es decir, hay que saber que los integrantes del equipo no se desenvuelven solamente en el contexto del problema, sino que también tienen implicación su experiencia, las normas a las que se ajustan, su rol en el sistema y sus preconcepciones.

En este caso hay que pensar con prospectiva para reconocer las implicaciones de lo que se está haciendo, y proyectarlo en la formulación y comprensión del problema. No basta con analizar el contexto del problema, compararlo con soluciones previas y luego trazar los objetivos y metas a alcanzar en su solución. Se requiere algo mucho más radical, como la aplicación que se sugiere de los principios seleccionados en la Ingeniería de Requisitos, y combinarlos de acuerdo con las disciplinas necesarias para comprender el problema y el sistema, reconociendo al mismo tiempo que existe una red de relaciones complejas entre ellos, y que son personas quienes intentan darles solución. En términos generales se puede decir que:

- Un modelo para gestionar la Ingeniería de Requisitos debe definir actividades que tengan como objetivo identificar requisitos, proporcionando actividades de soporte que contribuyan al éxito y a la calidad de los mismos.

- Para los ingenieros la flexibilidad en la implementación y la integración de los métodos existentes influye en el éxito del enfoque que adoptan, por lo que la presencia de asociación entre notaciones y métodos particulares es un factor clave en la aceptación del enfoque.

- Al principio la introducción de nuevos conceptos puede causar problemas, pero una sesión introductoria, para que los usuarios se familiaricen con los conceptos innovadores, puede contribuir al éxito de la implementación del modelo.

- Se necesita un enfoque iterativo para que los requisitos evolucionen, sin embargo, se requiere una guía para determinar cuándo comienza o termina cada etapa de la Ingeniería de Requisitos.

- Los ingenieros del software deben buscar activamente los requisitos, en lugar de enfatizar en la elicitación desde fuentes conocidas, porque ocurre cuando el equipo identifica la fuente y la dimensión de cada requisito.

IX

LIMITACIONES A LA INVESTIGACIÓN

A continuación, se describe las limitaciones para la aplicación del modelo; se trata de una reflexión sustentada en la experiencia del investigador al trabajar con ingenieros la gestión de la Ingeniería de Requisitos en proyectos reales. Se parte de la premisa de que para el desarrollo de software es muy importante utilizar el tiempo de la forma más eficiente posible, en el sentido de que esta variable incrementa el costo del producto final. Entonces, aplicar un modelo que requiera una visión diferente a los que la industria aplica tradicionalmente, podría encontrar reacciones desfavorables para su aplicación. En este sentido, algunas de las limitaciones son:

- La formación de los ingenieros relacionados con sistemas o software se sustenta en planes de estudios y principios establecidos desde hace mucho tiempo, cuya innovación es relativamente lenta (Serna, 2019). Las instituciones, los profesores, las didácticas y los mismos estudiantes, no siempre están de acuerdo en trabajar con modelos que todavía no han demostrado algún grado de eficiencia y efectividad en la industria. Esto genera un círculo vicioso en el que la academia espera que sus propuestas sean experimente la industria, mientras que ésta espera que la academia demuestre su utilidad. Por eso es que el uso y, en cierta medida, la masificación de este modelo, estarán sujetos a que los profesores lo comprendan y se apropien de él en sus cursos de Ingeniería de Requisitos, pero, asimismo, ellos estarán restringidos por la siguiente limitación.

- Los profesores que asesoran cursos de Ingeniería del Software han sido formados bajo los mismos conceptos y principios que siguen los planes de estudios desde hace mucho tiempo. Por eso es que pocos, o casi ninguno, ha tenido contacto y trabajado con conceptos como el Pensamiento Complejo y la Complejidad. Esta limitante les genera muchos interrogantes, e inclusive temores, para intentar acercarse a un modelo del que solamente pueden reconocer los principios ingenieriles.

- Como se mencionó previamente, la base del modelo se sitúa en la Gestión del Conocimiento, lo que quiere decir que los principios seleccionados se estructuran para lograr una gestión efectiva del mismo. Conociendo que el objetivo de la Ingeniería de Requisitos es comprender el problema y especificar las necesidades de las partes interesadas, se debería prestar mayor atención al conocimiento que se crea y descubre en el proceso. Esta limitación se presenta porque muchas organizaciones, ya sean clientes o la industria del software, no ha establecido en sus modelos de negocios procesos de Gestión del conocimiento, lo que no permitiría un adecuado trabajo en la Ingeniería de Requisitos, como el que aquí se propone.

- Una cuestión muy importante para gestionar la Ingeniería de Requisitos, con el modelo que se describe en este libro, es que los ingenieros entiendan que se deben apartar de la forma tradicional de hacer las cosas desde su disciplina. Por eso el modelo describe que el trabajo en equipos transdisciplinares debe ser sólido y permanente. El problema es que los ingenieros han desarrollado por años un trabajo individual y en pocas ocasiones comparten con otras disciplinas para integrar sus conceptos en lo que hacen. Esto genera reacciones negativas cuando son llamados a conformar equipos en los que estarán acompañados de profesionales de otras áreas del conocimiento.

- En relación con lo anterior, existe el asunto de que muchos programas se han acuñado como ingenierías, sin haber realizado un verdadero estudio de pertinencia, por lo que muchos simplemente tienen un nombre relativo, pero poco de ingeniería. Es el caso de la Ingeniería del Software, que en muchos países se estructura con los principios de las ingenierías tradicionales, pero que en la práctica parecen no demostrar que lo sea. El objetivo de esta ingeniería es capacitar profesionales para que desarrollen software fiable, seguro y de buena calidad, pero a la vez, los estudiantes deben pasar por cursos en ciencias básicas y otras áreas que no aportan al logro de este objetivo. Es decir, el término Ingeniería del Software se refiere a un *proceso ingenieril para desarrollar un producto*, *no a una ingeniería para fabricarlo*. Por eso es que desarrollar software se debe concebir más como un arte. Estas formas de educación limitan la aceptación y uso del modelo propuesto, porque esa formación ingenieril no capacita a los estudiantes para buscar formas alternativas de lograr el objetivo de su capacitación profesional.

- Las partes interesadas son un actor muy importante en la Ingeniería de Requisitos, porque son las que tienen los datos, la información, la experiencia y las necesidades que los ingenieros deben elicitar para especificarlas como requisitos. El problema es que muchas de ellas se niegan, consciente o inconscientemente, a facilitar lo que saben a los ingenieros. Porque tienen preconceptos en contra del desarrollo del sistema o porque consideran que no les traerá ningún beneficio. Si les es difícil responder a modelos rígidos y estructurados como los ingenieriles, es posible que recurran a una oposición pasiva ante este modelo. En parte porque pueden no comprender las ideas de trabajo que se involucran, o porque se sienten en inferioridad ante situaciones que se salen de su control.

- En medio de la Era Digital y con una Sociedad Software-Dependiente estamos viviendo una época en la que espera que todas las necesidades se resuelvan con celeridad. Se tiene poca paciencia y se exige respuestas y soluciones inmediatas ante cualquier situación. Esto hace que, en el caso del software, la industria se deje arrastrar por las circunstancias y se limite a aplicar los procesos conocidos para cumplir con los tiempos y presupuestos de entrega, por eso todo lo nuevo y desconocido se relega, porque está de por medio su permanencia en el mercado y su competitividad. Esto limita la aceptación y uso de este modelo para gestionar la Ingeniería de Requisitos, porque les exige a los ingenieros y a las partes interesadas mucho más de lo que hacen, pero, además, porque involucra principios casi desconocidos para la mayoría de ellos. Por lo que es posible no le den una oportunidad y se podrían estar perdiendo de implementar un proceso que les ayudaría a disminuir tiempos, a bajar costos y a entregar productos software más fiables.

Estas limitaciones llevan al investigador a hacerse una autocrítica en relación con el modelo que propone para gestionar la Ingeniería de Requisitos. La impresión es que se trata de una propuesta sin futuro inmediato y sin un público objetivo receptivo para apropiarse del modelo. Puede que esto sea cierto, pero también lo es el hecho de que solamente con visiones innovadoras y diferentes se impacta en la ciencia y se logra beneficios sólidos para la sociedad.

La crisis del software tiene más de medio siglo y todavía no se visualiza una solución a corto plazo, por el contrario, parece que en cada nueva época muta y se complica aún más. En el marco teórico y en estado del arte se sustenta que si bien mucho de lo que se ha propuesto hasta hora busca una solución, el progreso para encontrarla todavía no convence a los usuarios del software. Esto debe ser un aliciente para atreverse a proponer innovaciones a lo tradicional y a ayudar para alcanzar el objetivo. Posiblemente este modelo encuentre más detractores que seguidores, pero queda como obligación demostrar su eficiencia y efectividad para mejorar la fiabilidad del software. En este sentido, se ha venido incorporando paulatinamente en los cursos de Ingeniería de Requisitos que el investigador imparte, de la misma manera que ha hecho con el modelo que estructuró alrededor de los métodos formales. Dicho modelo fue descalificado por muchos profesores,

porque les exigía a ellos, y a los estudiantes, mejorar su formación en matemáticas y eso les parecía muy difícil. Pero con el tiempo se pudo demostrar que se trataba solamente de apreciaciones sin sustento y que la idea era integrar esa formación en el mismo modelo. Hoy se está utilizando en varios países de Latinoamérica.

El objetivo del investigador es demostrar que los principios y conceptos que involucra en el modelo, que no son tradicionales en la formación y capacitación en ingeniería, se pueden aprender en la aplicación del mismo modelo, porque al experimentarlo se van haciendo importantes y su aprendizaje se dará de forma escalonada. Conociendo el pensamiento ingenieril acerca de que lo que no se puede probar no existe o no es útil, la tarea es demostrar que estos principios sí son probables y útiles, cuando se realicen comparaciones entre proyectos desarrollados con los métodos tradicionales y los que se desarrollan siguiendo este modelo. Puede que al comienzo lleve un poco más de tiempo asimilarlo y ponerlo en práctica, porque él mismo se va adecuando a cada problema, pero con el tiempo, digamos en el segundo o tercer proyecto, seguramente se logrará una amplia mejoría en la fiabilidad del software.

Los ingenieros están acostumbrados a trazar diseños y a estructurarlos de acuerdo con las normas y recomendaciones que la ingeniería misma propone. Diversas organizaciones nacionales e internacionales siempre les han dicho cómo hacer las cosas y puede que muchas de ellas funcionen. Pero lo que todavía no se demuestra es que sean eficientes y eficaces para mejorar la fiabilidad del software. Esta es una premisa que el investigador piensa seguir para defender el hecho de que este objetivo no se ha podido lograr con lo tradicional, entonces es el momento de pensar diferente y permitir que otras ideas sean experimentadas con el mismo propósito.

No todo lo nuevo, y posiblemente descabellado, tiene que ser olvidado. Si esto se hiciera siempre entonces la humanidad no disfrutaría de los grandes desarrollos que se han materializado en las últimas décadas. Lo común a todos ellos es que nacieron de innovaciones contrarias y muchas veces por fuera de lo que se suponía que era el denominador común para construirlas. Por eso es que la Gestión del Conocimiento es tan importante en este modelo, porque el convencimiento es que solamente a través de él se podrá demostrar que el Pensamiento Complejo sí tiene ruta de ser Paradigma, y que su aplicación tiene fronteras más allá del campo de la educación.

REFERENCIAS

Abelló, A., Samos, J. y Saltor, F. (2002). Yam2 (yet another multidimensional model): An extension of UML. En M. Nascimento, M. Özsu y O. Zäıane (Eds.), IDEAS (pp. 172–181). IEEE Computer Society.

Alexander, I. (2003). Misuse cases help to elicit non-functional requirements. Computing & Control Engineering Journal 14(1), 40-45.

Al-Rawas, A. y Easterbrook, S. (1996). Communication problems in requirements engineering: A field study. En First Westminster Conference on Professional Awareness in Software Engineering. London, England.

Ambler, S., Nalbone, J. y Vizdos, M. (2004). Enterprise unified process: Extending the rational unified process. Prentice.

Ambriola, V. y Montagnero, C. (1992). Oikos at the age of three. Lectures Notes on Computer Science 635, 84-93.

Andriole, S. (1996). Managing Systems Requirements: Methods, tools and cases. McGraw-Hill.

Annoni, E. et al. (2005). Une approche d'analyse et de conception de SID à base de patrons. Ingénierie des systèmes d'information décisionnels. Revue des sciences et technologies de l'information 10(6), 1-28.

Annoni, E. et al. (2006). Towards multidimensional requirement design. Lecture Notes in Comp. Science 4081, 75-84.

ASTM (2012). Form and style of standards - ASTM Blue Book. ASTM International.

Baldwin, C. y Clark, K. (2000). Design Rules: The power of modularity. MIT Press.

Bandinelli, S., Fuggetta, A. y Grigolli, S. (1993). Process modeling-in-the-large with SLANG. En Second International Conference on the Software Process. Berlin, Germany.

Barwise, J. y Perry, J. (1983). Situations and attitudes. Bradford Books.

Batorowicz, B. y Shepherd, T. (2008). Measuring the quality of transdisciplinary teams. Journal of Interprofessional Care 22(6), 612-20

Batra, D. y Davis, J. (1992). Conceptual data modelling in database design: similarities and differences between expert and novice designers. International Journal Man-Machine Studies 37(1), 83-101.

Bell, J. (1999). Doing your research project. Open University Press.

Benbya, H. y McKelvey, B. (2006). Toward a complexity theory of information systems development. Information Technology & People 19(1), 12-35.

Bertalanffy, L. (1969). General System Theory: Foundations, development, applications. George Braziller Inc.

Bhat, J., Gupta, M. y Murthy, S. (2006). Overcoming Requirements Engineering Challenges - Lessons from Offshore Outsourcing. IEEE Software 23(5), 38-44.

Bickhard, M. (2011). Systems and process metaphysics. En C. Hooker (Ed.), Philosophy of Complex Systems (pp. 91-104). Elsevier.

Birk, A. y Heller, G. (2007). Challenges for Requirements Engineering and management in software product line development. En Sawyer, P., Paech, B. y Heymans, P. (Eds.), Requirements Engineering: Foundation for Software Quality (pp. 300-305). Springer.

Blaikie, N. (2000). Designing social research: The logic of anticipation. Polity Press.

Blaxter, M. et al. (1999). Parasitic helminth genomics. Parasitology 118, S39-S51.

Boehm, B. (1976). Software engineering. IEEE Transactions on Computers 25(12), 1226-1241.

Boehm, B. (1981). Software Engineering Economics. Prentice-Hall.

Boehm, B. et al. (1978). Characteristics of software quality. North Holland.

Böhnlein, M. y vom Ende, U. (1999). Deriving initial data warehouse structures from the conceptual data models of the underlying operational information systems. En 2nd ACM international workshop on Data warehousing and OLAP. Kansas City, USA.

Booch, G. (2007). Object-oriented design with applications. Addison-Wesley.

Brown, J. y Duguid, P. (1998). Organizing knowledge. California Management Review 40(3), 90-112.

Bryman, A. (2008). Social research methods. Oxford University Press.

Bubenko, J. (1995). Challenges in Requirements Engineering. En Second IEEE International Symposium on Requirements Engineering. York, United Kingdom.

Buglione, L. y Abran, A. (1999). Multidimensional software performance measurement models: A tetrahedron-based design. En Dumke, R. y Abran, A. (Eds.), Software measurement - Current trends in research and practice (pp. 93-107). Springer.

Cabibbo, L. y Torlone, R. (1998). A logical approach to multidimensional databases. Lecture Notes in Computer Science 1377, 183-197.

Camp, C. (2006). Applying transdisciplinary engineering to home inventory systems. A master of engineering report. Texas Tech University.

Canclini, N. (1995). Consumidores y ciudadanos - Conflictos culturales de la globalización. Grijalbo.

Canclini, N. (2004). Diferentes, desiguales y desconectados – Mapas de la Interculturalidad. Gedisa.

Canclini, N. (2006). La nueva escena sociocultural. En Castlini, N. y Piedras, E. (Eds.), Las industrias culturales y el desarrollo de México (pp. 123-145). Siglo XX editores.

Canclini, N. et al. (1996). Culturas en globalización. Nueva Sociedad.

Capra, F. (1996). The web of life: A new synthesis of mind and matter. Flamingo.
Cardoso, J. (1998). Multidimensional independent component analysis. En IEEE International Conference on Acoustics, Speech, and Signal Processing. Seattle, USA.
Carew, A. (2006). Enabling transdisciplinary engineering practice through targeted problem-based learning. University of Wollongong.
Carlshamre, P. et al. (2001). An industrial survey of requirements interdependencies in software product release planning. En IEEE Int. Conf. on Requirements Engineering. Toronto, Canada.
Carnap, R. (1938). Logical foundations of the unity of science. En Neurath, O., Carnap, R. y Morris, C. (Eds.), International Encyclopedia of Unified Science, Vol. 1 (pp. 56-87). Chicago University Press.
Carroll, J. y Swatman, P. (1998). The process of deriving requirements: Learning from practice. En ninth annual Australian Conference on Information Systems. Sydney, Australia.
Caseau, Y., Krob, D. y Peyronnet, S. (2007). Complexité des systèmes d'information: Une famille de mesures de la complexité scalaire d'un schéma d'architecture. Génie Logiciel 82, 23-30.
Castillo, I. et al. (2010). Requirements, aspects and software quality: The REASQ model. J. of Object Tec. 9(4), 69-91.
Chaos (2000). The software development report. Standish group.
Chappell, D. (2012). The three aspects of software quality: Functional, structural, and process. Chappell & Associates.
Checkland, P. (1981). Systems thinking - Systems practice. Wiley.
Christel, M. y Kang, K. (1992). Issues in requirements elicitation. Technical Report ESC-TR-92-012CMU/SEI-92-TR-12. Carnegie Mellon University.
Chung, L. et al. (2000). Non-Functional Requirements in Software Engineering. Kluwer.
Cohen, L. y Manion, L. (1994). Research methods in education. Routledge.
Cooke, T. (2011). Aspects of complexity: Managing projects in a complex world. Project Management Institute.
Cowling, A. (1998). A multi-dimensional model of the software engineering curriculum. En 11th Conference on Software Engineering Education. Atlanta, USA.
Creswell, J. (2002). Educational research: Planning, conducting, and evaluating quantitative and qualitative approaches to research. Pearson Education.
Creswell, J. (2003). Research design: Qualitative, quantitative, and mixed methods approaches. Sage.
Creswell, J. (2007). Qualitative inquiry and research design: Choosing among five traditions. Sage.
Creswell, J. y Maitta, R. (2002). Qualitative research. En Salkind, N. (Ed.), Handbook of research design and social measurement (pp. 143-184). Sage Publications.
Cronin, K. (2008). Transdisciplinary Research (TDR) and Sustainability. Environmental Science and Research.
Crotty, M. (2003). The foundations of social research: Meaning and perspective in the research process. Sage.
Curlee, W. y Gordon, R. (2014). Best practices and advances in program management series. CRC Press.
Curlee, W. y Gordon, R. (2014a). Successful program management - Complexity Theory, communication, and leadership. CRC Press.
Curtis, B. (1990). Empirical studies of the software design process. En 3rd IFIP International Conference on Human-Computer Interaction. Cambridge, UK.
Curtis, B. (1992). Insights from empirical studies of the software design process. Future Generation Computer Systems 7(2–3), 139-149.
Curtis, B. (1997). Software process improvement: Methods and lessons learned. En 19th International Conference on Software Engineering. Boston, USA.
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.
Cysneiros, L. et al. (2005). Reusable knowledge for satisficing usability requirements. En 13th IEEE International Requirements Engineering Conference. Paris, France.
Dag, N. et al. (2004). Speeding up requirements management in a product software company. Linking customer wishes to product requirements through linguistic engineering. En 12th International Requirements Engineering Conference. Kyoto, Japan.
Dag, N. et al. (2005). A linguistic engineering approach to large-scale requirements management. IEEE Software 22(1), 32-39.
Daly, E. (1977). Management of software development. IEEE Transactions on Software Engineering 3(3), 229-242.
Daneva, M. y Wieringa, R. (2006). A coordination complexity model to support Requirements Engineering for cross-organizational ERP. En 14th IEEE International Requirements Engineering Conference. Karlskrona, Sweden.
Davenport, T. y Prusak, L. (1998). Working Knowledge: How organisations manage what they know. Harvard Business School Press.
Davis, A. (1990). Software Requirements: Analysis & Specification. Prentice-Hall.
Davis, A., Jordan, K. y Nakajima, T. (1997). Elements underlying the specification of requirements. Annals of Software Engineering 3(1), 63-100.
De Silva, V. y Carlsson, G. (2004). Topological estimation using witness complexes. En Symposium on Point-Based Graphics. Zurich, Switzerland.
De Vries, H. et al. (2003). Stakeholder identification in IT standardization processes. En MIS Quarterly Special Issue Workshop on Standard Making: A Critical Research Frontier for Information Systems. Seattle, USA.
Diaz, M. y Sligo, J. (1997). How software process improvement helped Motorola. IEEE Software 14 (5), 75-81.

Dick, J. y Jones, B. (2012). On the complexity of requirements flow-down structures. INCOSE International Symposium 22(1), 1197-1206.
Do Ru, A. y Dogru, A. (2011). Modern software engineering concepts and practices: Advanced approaches. Information Science Reference.
Dobrica, L. y Niemelä, E. (2002). A survey on software architecture analysis methods. IEEE Transactions on Software Engineering 28(7), 638-653.
Doucet, P. y Sloep, P. (1992). Mathematical modelling in the life sciences. Ellis Horwood Limited.
Douglas, N. (2009). Engineering complex systems: Challenges in the theory and practice. CSIS Press.
Douglas, N. y Kuras, M. (2006). Engineering complex systems. En D. Braha, A. Minai y Y. Yam (Eds.), Complex Engineered Systems (pp. 206-245). Springer.
Downey, A. (2012). Think Complexity. Green Tea Press.
Dubey, S., Ghosh, S. y Rana, A. (2012). Comparison of software quality models: An analytical approach. International Journal of Emerging Technology and Advanced Engineering 2(2), 111-119.
Eddington, A. (1935). New pathways in science. MacMillan Company.
Edmonds, B. (1999). What is Complexity? - The philosophy of complexity per se with application to some examples in evolution. En Heylighen, F. y Aerts, D. (Eds.), The evolution of Complexity (pp. 1-18). Kluwer.
Eisner, E. (1991). The enlightened eye: Qualitative inquiry and the enhancement of educational practice. Macmillan.
Erlikh, L. (2000). Leveraging legacy system dollars for E-business. IT Professional 2(3), 17-23.
Fenton, N. (1991). Software metrics: A rigorous approach. Chapman & Hall.
Fernström, C., Närfelt, K. y Ohlsson, L. (1992). Software factory principles, architectures, and experiments. IEEE Software 9(2), 36-44.
Ferrara, N. (2010). The Complex Thinking: Building of a new paradigm. V!RUS 3, 1-21.
Finkelstein, A. y Sommerville, I. (1996). The Viewpoints FAQ. Software Engineering Journal 11(1), 2-4.
Firesmith, D. (2003). Common concepts underlying safety, security, and survivability engineering. Technical Note CMU/SEI-2003-TN-033. Carnegie Mellon University.
Firesmith, D. (2007). Common requirements problems, their negative consequences, and the industry best practices to help solve them. Journal of object technology 6(1), 17-33.
Fowler, P. et al. (1998). Transition packages: An experiment in expediting the introduction of requirements management. En Third IEEE International Conf. on RE. Colorado Springs, USA.
Freire, P. (2012). Professora Sim, Tia Nao. Civilizacao Brasileira.
Gal, C. (2013). A multi-dimensional separation of concerns of the web application requirements. Studia Universitatis, Informatica 58(3), 29-40.
García, R. (2006). Sistemas Complejos. Gedisa.
Garlan, D., Allen, R. y Ockerbloom, J. (1994). Exploiting style in architectural design environments. Software Engineering Notes 19(5), 175-188.
Gervasi, V. y Zowghi, A. (2005). Reasoning about inconsistencies in natural language requirements. ACM Transactions on Software Engineering and Methodology 14(3), 277-330.
Ghezzi, C. y Nuseibeh, B. (1998). Introduction to the special section on managing inconsistency in software development. IEEE Transactions on Software Engineering 24(11), 906-907.
Gibbons, M. y Nowotny, H. (2002). The potential of transdisciplinarity. En A. Bill et al. (Eds.), Transdisciplinarity: Joint problem solving among science, technology and society. An effective way for managing complexity (pp. 67-80). Springer.
Gilb, T. (2005). Competitive Engineering: A Handbook for systems engineering, requirements engineering, and software engineering using planguage. Elsevier.
Gillies, A. (1992). Software quality: Theory and management. Chapman & Hall.
Giorgini, P., Rizzi, S. y Garzetti, M. (2005). Goal-oriented requirement analysis for data warehouse design. En 8th ACM international workshop on Data warehousing and OLAP. Bremen, Germany.
Giraldo, G. (2004). Teoría de la complejidad y premisas de legitimidad en las políticas de educación superior. Cinta de Moebio. Recuperado: http://www.facso.uchile.cl/publicaciones/moebio/22/giraldo.htm.
Glinz, M. (2007). On non-functional requirements. En 15th IEEE Inte. Requirements Engineering Conf. Delhi, India.
Goguen, J. (1991). Types as theories. En G. Reed, A. Roscoe y R. Wachter (Eds.), Topology and Category Theory in Computer Science (pp. 357-390). Oxford University Press.
Goguen, J. (1992). The dry and the wet. En E. Falkenberg et al. (Eds.), Information Systems Concepts (pp. 1-17). Elsevier.
Goguen, J. (1994). Requirements engineering as the reconciliation of social and technical issues. En Jirotka, M. y Coguen, J. (Eds.), Requirements engineering - Social and technical issues (pp. 165-199). Academic Press Professional.
Goguen, J. y Linde, C. (1993). Techniques for requirements elicitation. En Fickas, S. y Finkelstein, A. (Eds.), Requirements Engineering '93 (pp. 152-164). IEEE.
Goguen, J., Weiner, J. y Linde, C. (1983). Reasoning and natural explanation. International Journal of Man-Machine Studies 19, 521-559.
Golfarelli, M., Maio, D. y Rizzi, S. (1998). The dimensional fact model: A conceptual model for data warehouses. Journals of Cooperative Information Systems 7(2-3), 215–247.

Green, J., Caracelli, V. y Graham, W. (1989). Toward a conceptual framework for mixed-method evaluation designs. Educational Evaluation and Policy Analysis 11(3), 255-274.
Grunwald, P. y Vitanyi, P. (2010). Shannon information and Kolmogorov complexity. IEEE Transaction Information Theory 34, 1-51.
Guba, E. y Lincoln, Y. (1982). Epistemological and methodological bases of naturalistic inquiry. Educational Communications and Technology Journal 30(4), 232-252.
Guindon, R. (1989). The process of knowledge discovery in system design. En Salvendy, G. y Smith, M. (Eds.), Designing and Using Human-Computer Interfaces and Knowledge Based Systems (pp. 727-734). Elsevier.
Guindon, R. (1990). Knowledge exploited by experts during software system design. International Journal Man-Machine Studies 33(3), 279-304.
Gunter, C. et al. (2000). A reference model for requirements and specifications. IEEE Software 17(3), 37-43.
Häberli, R. y Thompson, K. (2002). Summary. En A. Bill et al. (Eds.), Transdisciplinarity: Joint problem solving among science, technology and society. An effective way for managing complexity (pp. 3-5). Verlag.
Häberli, W. et al. (2001). Transdisciplinarity: Joint problem solving among science, technology and society. Birkhauser Verlag.
Hass, K. (2009). Managing complex projects: A new model. Management Concepts.
Haße, A. y Michielsen, C. (2013). The missing link – between requirements and design. En Complex Systems Design & Management Conf. Paris, France.
Hay, C. (2002). Political analysis - A critical introduction. Palgrave.
Hedlund, G. (1994). A model of knowledge management and the N-form Corporation. Strategic Management Journal 15, 73-90.
Herrmann, A. y Paech, B. (2008). MOQARE: Misuse-oriented quality requirements engineering. Requirements Engineering 13(1), 73-86.
Hofkirchner, W. (2009). The challenge of complexity: Social and human sciences in the information age. Discussion Paper CSGP D4/09. Pyden.
Hull, E., Jackson, K. y Dick, J. (2011). Requirements Engineering. Springer.
Hunsinger, J. (2005). Toward a transdisciplinary Internet research. The Information Society 21(4), 277-279.
Hüsemann, B., Lechtenbörger, J. y Vossen, G. (2000). Conceptual data warehouse modeling. En 2nd International Workshop on design and management of data warehouses. Stockholm, Sweden.
Hyvärinen, A. y Hoyer, P. (2000). Emergence of phase- and shift-invariant features by decomposition of natural images into independent feature subspaces. Neural Computation 12(7), 1705-1720.
Ibanez, M. (1996). European user survey analysis. Technical Report TR95104. European Software Institute.
ISO/IEC (2007). Systems and Software Engineering - Recommended practice for architectural description of software-intensive systems. ISO/IEC 42010:2007. IEEE.
Jackson, H. y Dick, J. (2002). Requirements Engineering. Springer
Jackson, M. (2003). Systems Thinking. Wiley.
Jackson, M. y Keys, P. (1984). Towards a system of systems methodologies. Journal of the Operational Research Society 35, 473-486.
Jacobson, I. (1992). Object-Oriented Software Engineering - A use case driven approach. Addison-Wesley.
Jaeger, C. (2003). A note on domains of discourse - Logical know-how for integrated environmental modelling. PIK Report 86. Potsdam Institute for Climate Impact Research.
Jantsch, E. (1972). Interdisciplinarity: Problems of teaching and research in universities. Center for Research and Innovation.
Jeffries, R. et al. (1981). The processes involved in designing software. En J. Anderson (Ed.), Cognitive Skills and Their Acquisition (pp. 255-283). Lawrence Erlbaum.
Jirotka, M. (1991). Ethnomethodology and Requirements Engineering. Technical Report PRGTR-92-27. Centre for Requirements and Foundations.
Johnson, D. (1994). Research methods in educational management. Longman Group.
Johnson, P. y Duberley, J. (2000). Understanding Management Research: An Introduction to Epistemology. Sage.
Jones, J., Ertas, A. y Parten, M. (1995). Multidisciplinary engineering design program at Texas Tech University. En First World Conference on Integrated Design and Process Technology. Austin, USA.
Jörg, T. (2011). New thinking in complexity for the social sciences and humanities - A generative, transdisciplinary approach. Springer.
Jureta, I., Faulkner, S. y Schobbens, P. (2006). A more expressive soft goal conceptualization for quality requirements analysis. En Embley, D., Olivé, A. y Ram, S. (Eds.), ER 2006 (pp. 281-295). Springer.
Juristo, N., Moreno A. y Silva, A. (2002). Is the European industry moving toward solving requirements engineering problems? IEEE Software 19(6), 70-77.
Kaindl, H. et al. (2002). Requirements engineering and technology transfer: Obstacles, incentives and improvement agenda. Requirements Engineering 7(3), 113-123.
Kalisman, N. et al. (2005). The neocortical microcircuit as a tabula rasa. Proceedings of the National Academy of Science of the United States of America 102(3), 880-885.
Kauppinen, M. et al. (2004). Implementing requirements engineering processes throughout organizations: Success factors and challenges. Information and Software Technology 46(14), 937-953.

Kellogg, S. et al. (2005). Developing the complex thinking skills required in today's global economy. En 35th ASEE/IEEE Frontiers in Education Conference. Indianapolis, USA.
Khaddaj, S. y Horgan, G. (2005). A proposed adaptable quality model for software quality assurance. Journal of Computer Sciences 1(4), 482-487.
Khushalani, A. (1997). Modelling and supporting opportunistic design problem solving. Doctoral Dissertation. Swinburne University of Technology.
Kimball, L., Weinstein, N. y Silber, T. (2004). Maximizing facilitation skills using principles of complexity science. En OD Network Conference. San Juan, Puerto Rico.
Kimball, R. et al. (1998). Data warehouse lifecycle toolkit: Expert methods for designing, developing and deploying data warehouses. Wiley.
King, G. et al. (2009). The Application of a Transdisciplinary Model for Early Intervention Services. Infants & Young Children 22(3), 211–223.
Kitchenham, B. y Pfleeger, S. (1996). Software quality: The elusive target. IEEE Software 13(1), 12-21.
Kitcher, P. (1999). Unification as a regulative ideal. Perspectives on Science 7(3), 337-348.
Kleiber, C. (2002). What kind of sciences does our world need today and tomorrow? A new contract between science and society. En Bill, A. et al. (Eds.), Transdisciplinarity: Joint problem solving among science, technology and society. An effective way for managing complexity (pp. 47-58). Verlag.
Klein, J. (1985). The interdisciplinary concept: Past, present, and future. En Levin, L. y Lind, I (Eds.), Inter-Disciplinarity Revisited: Re-assessing the Concept in Light of Institutional Experience (pp. l04-l36). OECD.
Klein, J. (2005). Interdisciplinary teamwork: The dynamics of collaboration and integration. En Derry, S. et al (Eds.), Interdisciplinary Collaboration: An Emerging Cognitive Science (pp. 23-50). Erlbaum.
Komi, S. y Tihinen, M. (2003). Great challenges and opportunities of distributed software development - An industrial survey. En Fifteenth International Conference on Software Engineering and Knowledge Engineering. San Francisco, USA.
Köppen, E., Mansilla, R. y Miramontes, P. (2005). La interdisciplina desde la teoría de los sistemas complejos. Ciencias 79, 4-12.
Kossiakoff, A. et al. (2011). Systems Engineering principles and practice. Wiley.
Kotonya, G. y Sommerville, I. (1998). Requirements Engineering: Process and techniques. John Wiley.
Kuhn, T. (1996). The Structure of Scientific Revolutions. University of Chicago Press.
Kunz, F. (1967). The search for conceptual unity. Main Currents.
Kushwaha, D. y Misra, A. (2006). Cognitive information complexity measure: A metric based on information contained in the software. WSEAS Transactions on Computers 3(5), 1109-2750.
Kushwaha, D. y Misra, A. (2006a). Evaluating cognitive information complexity measure. En 13th Annual IEEE International Symposium and Workshop on Engineering of Computer Based Systems. Berlin, Germany.
Kushwaha, D., Singh, R. y Misra, A. (2006). Cognitive complexity of procedural and object oriented programs. En 10th WSEAS international conference on Computers. Athens, Greece.
Kuvaja, P., Palo, J. y Bicego, A. (1999). TAPISTRY - A Software process improvement approach tailored for small enterprises. Software Quality Journal 8(2), 149-156.
Lamsweerde, A. (2001). Goal-oriented Requirements Engineering - A guided tour. En 5th IEEE international Symposium on Requirements Engineering. Toronto, Canada.
Law, A. (2007). Simulation Modeling and Analysis. McGraw-Hill.
Law, A. y Kelton, W. (2000). Simulation Modeling and analysis. McGraw-Hill.
Lawrence, R. y Després, C. (2004). Introduction: Futures of Transdisciplinarity. Futures 36(4), 397-405.
LeCompte, M. y Schensul, J. (1999). Designing and conducting ethnographic research. AltaMira.
Lemberger, P. y Morel, M. (2012). Managing complexity of information systems - The value of simplicity. John Wiley.
Levinson, S. (1983). Pragmatics. Cambridge University Press.
Lichtenstein, B. et al. (2006). Complexity leadership theory: An interactive perspective on leading in complex adaptive systems. Emergence: Complexity and Organization 8(4), 2–12.
Lincoln, Y. y Guba, E. (1985). Naturalistic inquiry. Sage.
Linde, C. y Goguen, J. (1978). Structure of planning discourse. Social and Biological Structures 1, 219-251.
Lipman, M. (1998). Pensamiento Complejo y educación. Ediciones de la Torre.
Loucopoulos, P. y Champion, R. (1989). Knowledge-based support for Requirements Engineering. Information and Software Technology 31(3), 123-135.
Loucopoulos, P. y Karakostas, V. (1995). System requirements engineering. McGraw-Hill.
Lubars, M., Potts, C. y Richter, C. (1993). A review of the state of the practice in requirements modelling. En IEEE International Symposium on Requirements Engineering. San Diego, USA.
Luján, S., Trujillo, J. y Song, I. (2002). Extending the UML for multidimensional modeling. Lecture Notes in Computer Science 2460, 290–304.
Lycett, M. et al. (2002). Migrating agile methods to standardized development practice. Computer 36(6), 79-85.
Maadawy, S. y Salah, A. (2012). Measuring change complexity from requirements: A proposed methodology. International Magazine on Advances in Computer Science and Telecommunications 3(1), 1-11.
Macaulay, L. (1996). Requirements Engineering. Springer.
Maeda, J. (2006). The Laws of Simplicity. MIT Press.

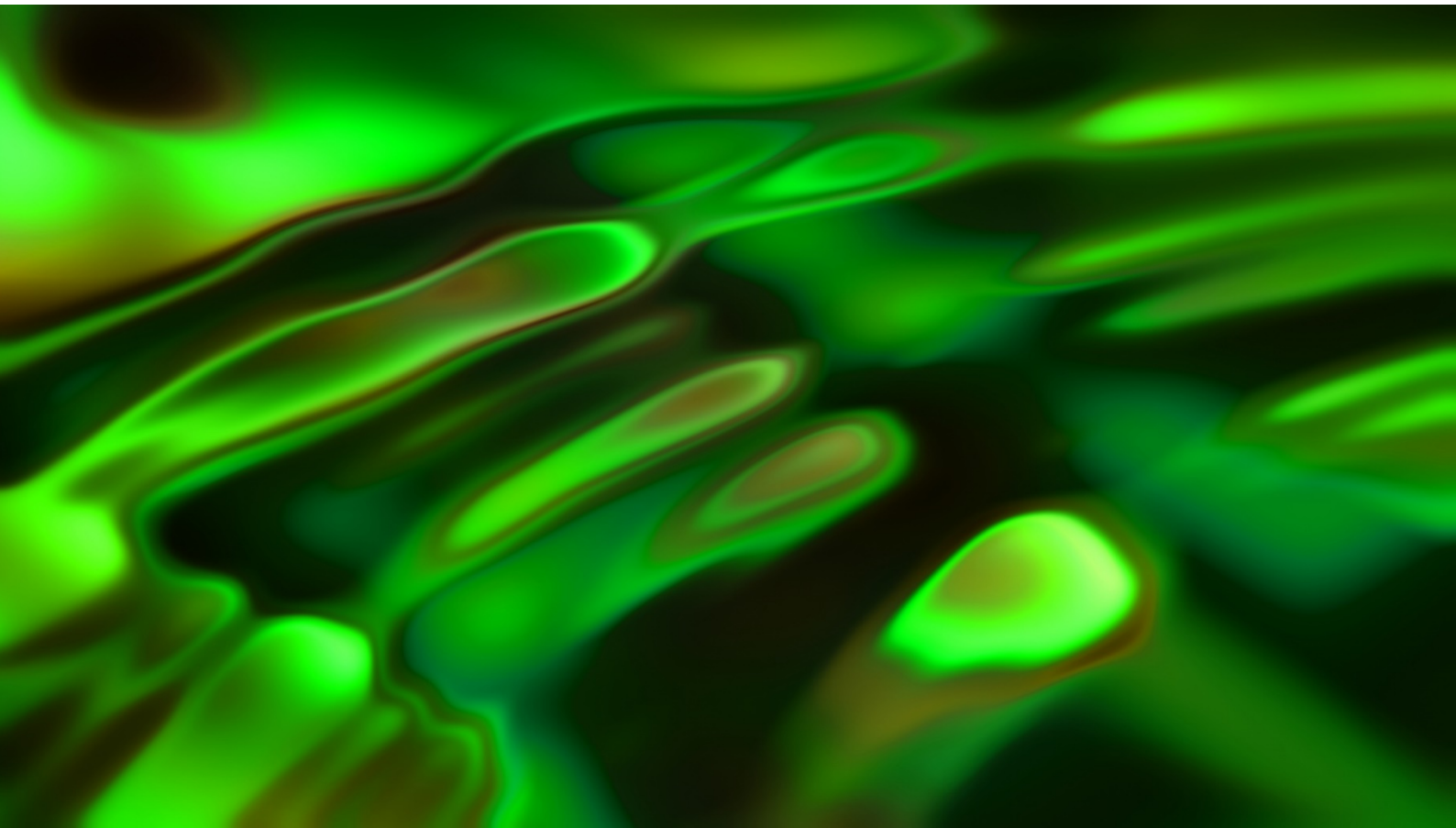
Mahaux, M. y Canon, C. (2012). Integrating the complexity of sustainability in requirements engineering. En First international workshop on Requirements for Sustainable Systems. Toronto, Canada.
Malhotra, A. et al. (1980). Cognitive processes in design. International journal Man-Machine Studies 12(2), 119-140.
Manns, T. y Coleman, M. (1996). Software quality assurance. MacMillan.
Martin, S. et al. (2008). Education and sustainable development – Learning to last? En J. Larkley y V. Maynhard (Eds.), Innovation in education (pp. 234-259). Nova Science.
Massot, I., Dorio, I. y Sabariego, M. (2004). Estrategias de recogida y análisis de la información. En Bisquerra, R. (Ed.), Metodología de la investigación educativa (pp. 329-366). La Muralla.
Mather, M. (1967). Objectives and Nature of Integrative Studies. Main Currents.
Matsumoto, Y. y Ohno, Y. (1989). Japanese perspectives in software engineering. Addison-Wesley.
Maxwell, J. (2003). Thinking for a Change. Warner Books.
McCullagh, P. y Nelder, J. (1989). Generalized linear models. Chapman & Hall.
McDermid, J. (1993). Requirements analysis: Orthodoxy, fundamentalism and heresy. En M. Bickerton y M. Jirotka (Eds.), Requirements Engineering (pp. 17-40). Academic Press.
McFeeley, B. (1996). IDEAL: A user's guide for software process improvement. Handbook CMU/SEI-96-HB-001. Pittsburgh.
McMichael, A. (2000). What makes transdisciplinarity succeed or fail? First Report. En A. Somerville y D. Rapport (Eds.), Transdisciplinarity: Recreating integrated knowledge (pp. 45-76). EOLSS Publishers.
Merriam, S. (1998). Qualitative research and case study applications in education: Revised and expanded from case study research in education. Jossey-Bass Publishers.
Mertens, D. (2003). Mixed methods and the politics of human research: The transformative-emancipatory perspective. En Tashakkori, A. y Teddlie, C. (Eds.), Handbook on mixed methods in the behavioral and social sciences (pp. 135-164). Sage.
Miles, M. y Huberman, A. (1994). Qualitative data analysis: A sourcebook. Sage.
Miller, D. (2000). Qualitative research course packet. University of Nebraska.
Moody, D. y Kortink, M. (2000). From enterprise models to dimensional models: A methodology for datawarehouse and data mart design. En 2nd Inter. Work. on Design and Management of Data Warehouses. Stockholm, Sweden.
Moreira, A., Araújo, J. y Rashid, A. (2005). A concern-oriented requirements engineering model. Lecture Notes in Computer Science 3520, 293-308.
Moreira, A., Rashid, A. y Arajo, J. (2005). Multi-Dimensional separation of concerns in Requirements Engineering. En 13th IEEE International Conference on Requirements Engineering. La Sorbonne, France.
Moriello, S. (2002). Dinámica de los Sistemas complejos. Recuperado: http://www.pensamientocomplejo.com.ar/docs/files/Moriello_Dinamica de los Sistemas Complejos.pdf.
Morin, E. (1980). El método II: La vida de la vida. Cátedra.
Morin, E. (1982). Ciencia con conciencia. Anthropos Editorial del Hombre.
Morin, E. (1991). Un nouveau commencement. Editions du Seuil.
Morin, E. (1994). Introducción al pensamiento complejo. Gedisa.
Morin, E. (2008). On Complexity. Hampton Press.
Morin, E. (2008a). Reform of Thought. En Nicolescu, B. (Ed.), Transdisciplinarity: Theory and Practice (pp. 67-90). Hampton Press.
Negele, H. et al. (2005). Successful implementation and application of continuous risk management to complex systems development in the automotive industry. INCOSE 15(1), 1331-1344.
Nelson, S. (2001). Transdisciplinary team training. En Workshop on Global Transdisciplinary Education, and Research, Society for Design and Process Science. Pasadena, USA.
Neurath, O. (1938). Unified science as encyclopedic integration. En Neurath, O., Carnap, R. y Morris, C. (Eds.), International Encyclopedia of Unified Science. Vol. 1 (pp. 345-360). Chicago University Press.
Nguyen, L. y Swatman, P. (2000). Essential and incidental complexity in requirements models. En 4th International Conference on Requirements Engineering. Schaumburg, USA.
Nicolescu, B. (1996). La transdisciplinaritè: Manifeste. Editions du Rocher.
Nicolescu, B. (2006). Transdisciplinarity - Past, present and future. En Haverkort, B. y Reijntjes, C. (Eds.), Moving Worldviews – Reshaping sciences, policies and practices for endogenous sustainable development (pp. 142-166). COMPAS Editions.
Nicolescu, B. (2008). Transdisciplinarity: Theory and Practice. Hampton Press.
Nikula, U. y Sajaniemi, J. (2005). Tackling the complexity of requirements engineering process improvement by partitioning the improvement task. En Australian Software Engineering Conference. Brisbane, Australia.
Nishiyama, T., Ikeda, K. y Niwa, T. (2000). Technology transfer macro-process: A practical guide for the effective introduction of technology. En 22nd International Conference on Software Engineering. Limerick, Ireland.
Nonaka, I. y Takeuchi, H. (1995). The knowledge-creating company: How Japanese companies create the dynamics of innovation. Oxford University Press.
Offen, R. y Jeffery, R. (1997). Establishing software measurement. IEEE Software 14(2), 45-53.
Olabiyisi S., Adetunji, A. y Olusi, T. (2013). Using Software requirement specification as complexity metric for multi-paradigm programming languages. International Journal of Emerging Technology and Advanced Engineering 3(3), 562-569.

OMG. (2012). How to deliver resilient, secure, efficient, and easily changed it systems in line with CISQ recommendations. Object Management Group.
Parviainen, P., Tihinen, M. y Solingen, V. (2005). Requirements Engineering - Dealing with the Complexity of Sociotechnical Systems Development. En Maté, J. y Silva, A. (Eds.), Requirements Engineering for sociotechnical systems (pp. 1-20). Information Science Publishing.
Pedersen, T. y Jensen, C. (1999). Multidimensional data modeling for complex data. En 15th International Conference on Data Engineering. Sydney, Australia.
Penrose, R. (1991). La nueva mente del emperador. Teoría de la complejidad. Grijalbo Mondoval.
Perry, D. y Wolf, A. (1992). Foundations for the study of software architecture. Software Engineering Notes 17(4), 40-52.
Plato (1921). Theaetetus. Plato in Twelve Volumes, Vol. 12. Harvard University Press.
Pohl, K. (1992). The three dimensions of requirements engineering. Technical Report 92-11, RWTH-Aachen. Informatik V.
Pohl, K. (1994). Three dimensions of requirements engineering - A framework and its application. Information Systems 19(3), 243-258.
Pohl, K. y Rupp, C. (2015). Requirements Engineering fundamentals. Rocky Nook.
Polanyim, M. (2009). The tacit dimension. University of Chicago Press.
Porter, S. (1996). Qualitative research in the research process in nursing. Blackwell Science.
Potts, C. (1993). Software engineering research revisited. IEEE Software 10(5), 19-28.
Pressman, R. (2005). Software Engineering: A practitioner's approach. McGraw-Hill.
Radcliffe, D. (2006). Shaping the discipline of engineering education. Journal of Engineering Education 95(4), 5-6.
Ramesh, B. (1993). Process knowledge based rapid prototyping for Requirements Engineering. En IEEE International Symposium on Requirements Engineering. San Diego, California.
Ramesh, B. y Edwards, M. (1993). Issues in the development of a requirements traceability model. En IEEE Int. Symposium on Requirements Engineering. San Diego, California.
Ramingwong, R. (2011). A review of requirements engineering processes, problems and models. International Journal of Engineering Science and Technology 4(6), 2997-3002.
Rebel, M. et al. (2013). Enhancing interpretation-quality of requirements using PLM integrated requirements-communication in cross company development processes. En M. Abramovici y R. Stark (Eds.), Smart Product Engineering (pp. 43–50). Springer.
Regnell, B. et al. (2008). Can we beat the complexity of very large-scale requirements engineering? Requirements Engineering: Foundation for software quality. Lecture Notes in Computer Science 5025, 123-128.
Regnell, B., Olsson, H. y Mossberg, S. (2006). Assessing requirements compliance scenarios in system platform subcontracting. En 7th International Conference on Product Focused Software Process Improvement. Amsterdam, The Netherlands.
Reynolds, P. (2007). Managing requirements for a US$1bn IT-based business transformation: New approaches and challenges. Journal of Systems and Software 80(3), 285-293.
Richardson, I. (2002). SPI Models: What characteristics are required for small software development companies? Software Quality Journal 10(2), 101-114.
Richardson, K. (2010). Thinking About Complexity - Grasping the Continuum through Criticism and Pluralism. Emergent.
Robinson, M. y Bannon, L. (1991). Questioning representations. En Second European Conference on Computer-Supported Co-operative Work. Amsterdam, The Netherlands.
Romero, O. y Abelló, A. (2006). Multidimensional design by examples. Lecture Notes in Comp. Science 4081, 85-94.
Ropohl, G. (1999). Philosophy of socio-technical systems. Society for philosophy and technology 4(3), 59-71.
Rosenhead, J. y Mingers, J. (2001). Rational analysis for a problematic world revisited. Wiley.
Roux, M. (2002). Inventer un nouvel art d'habiter - Le ré-enchantement de l'Espace. Edit. L'Harmattan.
Rumbaugh, J. et al. (1991). Object-oriented modeling and design. Prentice-Hall.
Sabariego, M. y Bisquerra, R. (2004). Fundamentos metodológicos de la investigación educativa. En Bisquerra, R. (Ed.), Metodología de la investigación educativa (pp. 20-49). La Muralla.
Sacks, H. (1992). Lectures on Conversation. Blackwell.
Sacks, H., Schegloff, E. y Jefferson, G. (1974). A simplest systematics of the organization of turn-taking in conversation. Language, 50(4), 696-735.
Sage, A. y Rouse, W. (2014). Handbook of Systems Engineering and Management. Willey.
Satria, R. (2003). Analyzing requirements engineering problems. En IECI Japan Workshop. Chofu Bunka, Japan
Savransky, S. (2000). Engineering of creativity: TRIZ. CRC Press.
Scharnhorst, A., Börner, K. y Besselaar, P. (2012). Models of science dynamics - Encounters between complexity theory and information sciences. Springer.
Schrodinger, E. (1967). What is life? The physical aspect of the living cell and mind and matter. Cambridge University.
Schroeder, R. y Harry, M. (2006). Six Sigma: The Breakthrough Management Strategy Revolutionizing the World's Top Corporations. Currency.
Schultze, U. (1998). Investigating the contradictions in knowledge management. En IFIP Working Groups 8.2 and 8.6 Joint working conference on information systems: Current issues and future changes. Helsinki, Finland.

Schulz, A. (1970). Ecosystemology. University of California Press.
Schwandt, T. (2001). Dictionary of qualitative enquiry. Sage.
Serna, E. (2011). Analysis and selection to requirements elicitation techniques. En 7th Colombian Computing Congress. Medellín, Colombia.
Serna, E. (2011a). The importance of abstraction in the informatics. Revista Scientia et Técnica 2(48), 122-136.
Serna, E. (2012). Current state of research on non-functional requirements. Revista Ingeniería y Universidad, 16(1), 225-246.
Serna, E. (2012a). Software testing: More than a stage in the life cycle. Revista de Ingeniería 35, 34-40.
Serna, E. (2012b). Maturity model of Knowledge Management in the interpretativist perspective. International Journal of Information Management 32(4), 365-371.
Serna, E. (2013). Logic in Computer Science. Revista Educación en Ingeniería 8(15), 62-68.
Serna, E. (2013a). Manifiesto por la profesionalización del desarrollo de software. Editorial Instituto Antioqueño de Investigación.
Serna, E. (2015). Maturity model of transdisciplinary knowledge management. International Journal of Information Management 35(6), 647-654.
Serna, E. (2015a). Ciencia y pensamiento complejo - Desarrollo transdisciplinar de un paradigma. Editorial Instituto Antioqueño de Investigación.
Serna, E. (2019). Educación para un nuevo orden mundial: Retos de un escenario emergente para la formación y la capacitación de una nueva categoría de estudiantes. Editorial Instituto Antioqueño de Investigación.
Serna, E. y Polo, J. (2014). Logic and abstraction in engineering education: A necessary relationship. Ingeniería, Investigación y Tecnología XV(2), 299-310.
Serna, E. y Serna, A. (2013). A review processes for science, technology and innovation. Entramado 9(1), 172-187.
Serna, E. y Serna, A. (2020). MoDeMaRE: Model to Development and Management Requirements Engineering. Advances in Engineering Software. *In press*.
Serna, E. y Suaza, J. (2020). REDOC: A Model for Documenting Requirements Elicitation. Computer Science - Research and Development. *In press.*
Serna, E. y Zapata, L. (2014). Approach to logic and abstraction in the engineering training. Revista Internacional de Educación y Aprendizaje 2(1), 35-47.
Serrano, M., Calero, C. y Piattini, M. (2003). Experimental validation of multidimensional data models metrics. En 36th Hawaii International Conference on Sys-tem Sciences. Hawaii, USA.
Shannon, C. y Weaver, W. (1964). The mathematical theory of communication. University of Illinois press.
Sharma, A. y Kushwaha, D. (2010). A complexity measure based on Requirement Engineering document. Journal of Computer Science and engineering 1(1), 112-117.
Shelley, M. (2014). Frankenstein. Grupo Editorial Tomo.
Siddiqi, J. (1996). Requirement Engineering: The emerging wisdom. IEEE Software 13(2), 15-19.
Snow, C. (1959). The two cultures and the cientific revolution. Cambridge University Press.
Sommerville, I. y Sawyer, P. (1997). Requirements Engineering: A good practice guide. John Wiley.
Sommerville, I. y Sawyer, P. (1998). Viewpoints: Principles, problems and a practical approach to requirements engineering. Annals of Software Engineering 3, 101-130.
Sotolongo, P. y Delgado, C. (2006). La revolución contemporánea del saber y la complejidad social: Hacia unas ciencias sociales de nuevo tipo. CLACSO.
SPAAL. (2009). Transdisciplinary Practice - Position Statement. ABN 17 008 393 440. The Speech Pathology Association of Australia Limited.
Staden, R. (2004). Strategic quality – A software engineering approach. Rand Afrikaans University Press.
Stainton, W. (2003). Social psychology: Experimental and critical approaches. Open University Press.
Stake, R. (1995). The art of case study research. Sage.
Stellman, A. y Greene, J. (2005). Applied software project management. O'Reilly Media.
Stephen, K. (1995). Metrics and models in software quality engineering. Addison-Wesley.
Stevens, R. et al. (1998). Systems Engineering - Coping with complexity. Prentice Hall.
Suchman, L. (1987). Plans and situated actions: The problem of human-machine communication. Cambridge University Press.
Sutcliffe, A. et al. (2003). Evolutionary requirements analysis. En 11th IEEE International Conference on Requirements Engineering Monterey Bay, USA.
Tanik, M. y Ertas, A. (1997). Interdisciplinary design and process science: A discourse on scientific method for the integration age. Journal of Integrated Design & Process Science 1(1), 76-94.
Tankoano, J., Derniame, J. y Kaba, A. (1994). Software process design based on products and the object oriented paradigm. Lectures Notes on Computer Science 772, 177-185.
Tarr, P. et al. (1999). N Degrees of separation: Multi-dimensional separation of concerns. En 21st international conference on Software Engineering. Los Angeles, USA.
Tashakkori, A. y Teddlie, C. (1998). Mixed methodology: Combining qualitative and quantitative approaches. Sage.
Tate, D. et al. (2006). Foundations for a Transdisciplinary approach to engineering systems research based on design & process. En D. Tate et al. (Eds.), The ATLAS Module (pp. 45-65). The ATLAS Publishing.
Taylor, S. y Bogdan, R. (1998). Introducción a los métodos cualitativos de investigación. Paidós.

Tenkasi, R. y Boland, R. (1996). Exploring Knowledge diversity in knowledge intensive firms: A new role for information systems. Journal of Organizational Change Management 9(1), 79-91.
Tharp, T. y Zalewski, J. (2001). Economics of software engineering: Transdisciplinary issues in research and education. Integrated Design & Process Technology Proceedings 1, 6-11.
Thomas, G. (2009). How to do your research project: A guide for students in education and applied social sciences. Sage.
Thompson, J. (2004). Prospects for Transdisciplinarity. Futures 36(4), 515-526.
Tsoukas, H. (1996). The firm as a distributed knowledge system: A constructionist approach. Strategic Management Journal 17, 11-25.
Venters, W. (2002). Literature for C-Sand: Knowledge management. C-SandD/WP/1002/2.
Vijayan, J. y Raju, G. (2011). A new approach to requirements elicitation using paper prototype. International Journal of Advanced Science and Technology 28, 9-16.
Visser, W. (1992). Designer's activities examined at three levels: Organisation, strategies and problem-solving processes. Knowledge-Based Systems 5(1), 92-104.
Wallerstein, I. (1996). Open the Social Sciences: Report of the Gulbenkian Commission on the Restructuring of the Social Sciences. Stanford University Press.
Walz, D., Elam, J. y Curtis, B. (1993). Inside a software design team: Knowledge acquisition, sharing, and integration. Communications of the ACM 36(10), 63-77.
Wang, Y. y Shao, J. (2003). Measurement of the cognitive functional complexity of software. En Second IEEE International Conference on Cognitive Informatics. London, UK.
Wicklein, R. y Rojewski, J. (1995). The relationship between psychological type and professional orientation among technology education teachers. Technical Education 7(1), 57-74.
Wiegers, K. (1999). Software Requirements. Microsoft Press.
Winter, R. y Strauch, B. (2003). A method for demand-driven information requirements analysis in Data Warehousing Projects. En 36th Annual Hawaii International Conference on System Sciences. Hawaii, USA.
Xiong, J. (2011). New software engineering paradigm based on complexity science - An introduction to NSE. Springer.
Zahran, S. (1998). Software process improvement: Practical guidelines for business success. Addison-Wesley.
Zamperoni, A. (1996). GRaph-based, integrated development of software: Integrating different perspectives of Software Engineering. En 18th International Conference on Software Engineering. Berlin, Germany.
Zave, P. (1997). Classification of research efforts in Requirements Engineering. ACM Computing Surveys 29(4), 315-321.
Zemelman, H. (2003). Conocimiento y ciencias sociales: algunas lecciones sobre problemas epistemológicos. Editorial Universidad Ciudad de México.
Zhou, Y. y Leung, H. (2006). Empirical analysis of object-oriented design metrics for predicting high and low severity faults. IEEE transactions on software engineering 32(10), 771-789.
Ziehe, A. y Müller, K. (1998). Tdsep – An efficient algorithm for blind separation using time structure. En L. Niklasson, M. Boden y T. Ziemke (Eds.), ICANN'98 (pp. 675-680). Springer.

Gestión de la Ingeniería de Requisitos Integrando Principios del Pensamiento Complejo

En este libro se identifica la problemática de la fiabilidad del software y se presenta un método para gestionar la Ingeniería de Requisitos dividido en etapas, al que se incorpora los principios de la Transdisciplinariedad y la Multidimensionalidad del Pensamiento Complejo, para estructurar una propuesta inter-relacional lógica, orientada a gestionar esta fase del desarrollo de software. Se trata de una innovación a la forma como se lleva a cabo esta fase del ciclo de vida, con el objetivo de ayudar a mejorar la fiabilidad y seguridad del producto final. Cada uno de estos principios se analiza desde los aportes de diversos autores y se incorpora las posibilidades de conjugación a lo establecido para la Ingeniería de Requisitos. De esta manera se integran a las etapas del método de forma complementaria a los objetivos que cada una busca en la gestión de los requisitos del sistema.

© 2021 Editorial Instituto Antioqueño de Investigación