Universidad Rey Juan Carlos

Máster Oficial en
Ingeniería de Sistemas de Información
Curso académico 2018 / 2019

Trabajo Fin de Máster

Patrones y anti-patrones en equipos ágiles: una revisión sistemática

Autor: Cristina Ramos Vega
Tutor: Javier Garzás Parra

Agradecimientos

A Javier Garzás, tutor de este trabajo fin de máster, por su confianza y por su tiempo.

A José Manuel Vara, director del máster, por su paciencia infinita y por su tiempo.

A Alfonso, mi marido, por su apoyo incondicional en todo lo que emprendo.

A Rocío y Marta, mis hijas, mis dos inspiraciones para seguir adaptándome a los tiempos.

A Pilar, mi madre, por alentarme siempre a no dejar de aprender. Su “hija, el saber no ocupa lugar” me acompaña desde que tengo uso de razón.

Resumen

En el contexto de la ingeniería del software, los patrones resuelven problemas recurrentes en contextos específicos y están ampliamente aceptados por los profesionales del software. Dado que los patrones son soluciones reutilizables y probadas repetidas veces, su uso minimiza tanto el riesgo de cometer errores como el tiempo dedicado a resolver problemas a los que otros ya se han enfrentado en situaciones similares. De igual modo, los patrones se formulan con un lenguaje común que facilita la comunicación entre los miembros del equipo. Todo ello redunda directamente en la calidad del producto y en su coste.

La llegada de la agilidad al desarrollo de software tuvo como objetivo reducir el riesgo que provoca la incertidumbre de los posibles cambios de alcance en los proyectos sin perder el foco, ni en la entrega en plazo ni en el coste del proyecto. En el caso de la agilidad, una de las claves para conseguirlo es el equipo. Un equipo auto-organizado que colabore en aportar el máximo valor al negocio en periodos cortos de tiempo y minimizando el esfuerzo.

Ambos conceptos, patrón y agilidad, persiguen un propósito común: minimizar el esfuerzo y garantizar la calidad del producto. De hecho, el uso de patrones en un equipo ágil es inherente a los frameworks ágiles que practican, ya que tanto XP como Scrum son considerados lenguajes de patrones en algunos foros, aunque no estén formulados con el formato estándar de los patrones de software. Por el contrario, hay quienes, por esa circunstancia los consideran simplemente un conjunto de buenas prácticas testadas que resuelven problemas recurrentes en la gestión y el desarrollo de proyectos software.

Para las empresas es una necesidad adaptar sus modelos de negocio a los nuevos escenarios que la tecnología está provocando en los hábitos de consumo. Por tanto, su prioridad pasa por la satisfacción del cliente para garantizar su cifra de negocios y por la optimización de cualquier esfuerzo para minimizar los costes. La potencia del concepto de patrón, unido a la potencia de la agilidad en forma de un equipo auto-organizado, multiplica exponencialmente el efecto de ambos.

En este trabajo fin de máster se va a realizar un estudio del arte para identificar si existen propuestas de patrones en torno a equipos ágiles y analizar el contexto en el que se proponen. Para revisar y analizar la literatura mediante una metodología rigurosa, formal y que proporcione valor científico a los resultados, se lleva a cabo un proceso de revisión sistemática.

A partir de los resultados obtenidos se propone una línea de investigación futura para mejorar el estado del arte. Dicha contribución se centra en generar un marco de trabajo de patrones en formato ágil, orientados a facilitar la auto-organización de cualquier equipo ágil desde su inicio hasta su madurez.

Índice

Índice de tablas ..... **vi**

Índice de figuras ..... **vii**

1 Introducción **8**

1.1 Presentación del problema 8

1.2 Objetivos 9

1.3 Estructura de la memoria 10

2 Estudios previos **11**

2.1 Revisión sistemática 11

2.2 Agilidad 12

2.3 Patrones 14

3 El proceso de la revisión sistemática **17**

3.1 Metodología 17

3.2 Objetivo de la revisión sistemática 18

3.3 Estrategia de búsqueda 18

3.3.1 Palabras clave 18

3.3.2 Selección de fuentes 19

3.4 Selección de los estudios 19

3.1 Extracción de la información 23

4 Ejecución de la revisión sistemática **24**

4.1 Criterios de la Evaluación 24

4.2 Estudios analizados 25

4.2.1 EP01 (2004): AntiPractices: AntiPatterns for XP Practices 26

4.2.2 EP02 (2006): Value based Extreme Programming 27

4.2.3 EP03 (2006): Patterns for Agile Development Practice part 3* 28

4.2.4 EP04 (2006): Programmers are from Mars, Customers are from Venus 30

4.2.5 EP05 (2007): A Process Pattern Language for Agile Methods 31

4.2.6 EP06 (2008): Agile Learning & Teaching with Wikis 33

4.2.7 EP07 (2010): How Much is Just Enough? 35

4.2.8 EP08 (2011): Communication Patterns of Agile Requirements 37

4.2.8 EP09 (2001): A Pattern Language for Release and Deployment 38

4.2.9 EP10 (2011): Extending Patterns for Fearless Change* 40

4.2.10 EP11 (2011): The Art of Software Estimation Patterns Language 42
4.2.11 EP12 (2012): A pattern Language for Agile Software Estimation 43
4.2.12 EP13 (2013): Is there an Agile Handover? 45
4.2.13 EP14 (2014): Teams that Finish Early Accelerate Faster.....46
4.2.14 EP15 (2015): Animating Organizational Patterns 48
4.2.15 EP16 (2015): Patterns for Integrating Agile Development and UCD 49
4.2.16 EP17 (2015): Distributed Agile Patterns for Offshore Software 50
4.2.17 EP18 (2016): Agile base Patterns in the Agile Canon 52
4.2.18 EP19 (2016): Exploring ScrumBut – Scrum anti-patterns 54
4.2.19 EP20 (2017): Patterns of Collaboration Driven by Requirements 56
4.2.20 EP21 (2017): Anti-patterns in Agile Adoption: A Grounded Theory.....57
4.2.21 EP22 (2018): A Program Backlog Story with Patterns* 58
5 Análisis de los resultados 62
5.1 Extracción y síntesis de los resultados 62
5.1.1 Escalas de valores para los criterios de evaluación 62
5.1.2 Resumen de criterios analizados en cada estudio primario 64
5.1.3 Resumen de formatos de patrón identificados en cada estudio primario 66
5.2 Discusión 68
5.3 Tendencia de los estudios 70
5.3.1 Año de publicación 71
5.3.2 Medios de publicación de los trabajos 72
5.4 Propuestas de mejora 74
6 Conclusiones y trabajos futuros.....75
6.1 Análisis de la consecución de los objetivos 75
6.2 Estimación de esfuerzos 76
6.3 Trabajos futuros 78
Referencias.....80
Anexo A. Referencias de los estudios primarios 83
Anexo B. Resumen de patrones por estudio primario 86
Anexo C. Formato de patrón por estudio primario 96
Anexo D. Scrum Pattern Language 101

Índice de tablas

Tabla 3.1 - Cadena Básica de Búsqueda 19
Tabla 3.2 - Lista inicial de fuentes de información 19
Tabla 3.3 - Resultados tras adaptar la cadena de búsqueda a las fuentes 20
Tabla 3.4 - Total de Estudios Primarios seleccionados 21
Tabla 3.5 - Listado de Estudios Primarios por orden cronológico de aparición 22
Tabla 3.6 - Formato de extracción de los Estudios Primarios 23
Tabla 4.1 - Ficha de evaluación EP01 27
Tabla 4.2 - Ficha de evaluación EP02 28
Tabla 4.3 - Ficha de evaluación EP03 29
Tabla 4.4 - Ficha de evaluación EP04 31
Tabla 4.5 - Ficha de evaluación EP05 33
Tabla 4.6 - Ficha de evaluación EP06 34
Tabla 4.7 - Ficha de evaluación EP07 36
Tabla 4.8 - Ficha de evaluación EP08 37
Tabla 4.9 - Ficha de evaluación EP09 39
Tabla 4.10 - Ficha de evaluación EP10 41
Tabla 4.11 - Ficha de evaluación EP11 43
Tabla 4.12 - Ficha de evaluación EP12 44
Tabla 4.13 - Ficha de evaluación EP13 46
Tabla 4.14 - Ficha de evaluación EP14 47
Tabla 4.15 - Ficha de evaluación EP15 48
Tabla 4.16 - Ficha de evaluación EP16 49
Tabla 4.17 - Ficha de evaluación EP17 51
Tabla 4.18 - Ficha de evaluación EP18 53
Tabla 4.19 - Ficha de evaluación EP19 55
Tabla 4.20 - Ficha de evaluación EP20 57
Tabla 4.21 - Ficha de evaluación EP21 58
Tabla 4.22 - Ficha de evaluación EP22 60
Tabla 5.1 - Valores para los criterios de evaluación 63
Tabla 5.2 – Resumen de criterios analizados por estudio primario 64
Tabla 5.3 - Resumen de formatos de patrón por estudio primario 67

Tabla Anexo B.0.1 - Resumen de patrones por estudio primario 95

Índice de figuras

Figura 2.1 – Uso de frameworks ágiles en 2018 (VersionOne Survey [21]) 13
Figura 3.1 - Proceso de Revisión Sistemática (Bolchini) 17
Figura 4.1 – Lenguaje de patrones de proceso para frameworks ágiles [EP05].....32
Figura 4.2 – Proceso Agile Learning and Teaching basado en Scrum [EP06] 34
Figura 4.3 – Relación entre patrones [EP07] 36
Figura 4.4 - Relación entre patrones [EP09].....39
Figura 4.5 - Relación de patrones [EP12] 44
Figura 4.6 - Puntos en común y diferencias entre procesos ágiles y DCU [EP16] 50
Figura 4.7 - Los eventos Scrum como patrones básicos Agile [EP18].....53
Figura 4.8 - Relación entre anti-patrones Scrum y algunos de sus conceptos [EP19]....55
Figura 4.9 - Árbol de decisión patrones product backlog de programa [EP22].....59
Figura 4.10- The Magic Backlog Patterns [52].....61
Figura 5.1 - Número de estudios primarios publicados por año 71
Figura 5.2 - Número de estudios primarios por publicación 72
Figura 5.3 - Número de estudios primarios por editor.....73
Figura 6.1 – Estimación de esfuerzo por actividad.....76
Figura Anexo C.0.1 - Relación entre las secciones del formato de patrón [EP05].....97
Figura Anexo D.0.1 – Lenguaje de patrones Scrum (ScrumPLoP) [53] 101

1 Introducción

En este primer capítulo, se presenta el problema al que se va a dar respuesta en este trabajo fin de máster y los objetivos identificados para su resolución. También se incluye una breve descripción del contenido del resto de capítulos de este documento.

1.1 Presentación del problema

En este apartado, se presenta el problema que se pretende resolver contextualizado en el momento actual y se apunta tanto la solución como la metodología que se plantea en este trabajo fin de máster.

En el contexto de la revolución digital que se esta viviendo en los últimos años, la tecnología y los hábitos de consumo exigen a las empresas nuevas formas de hacer negocio y de relacionarse. Sus modelos de negocio necesitan responder a la demanda de un cliente, cada vez, más acostumbrado a consumir productos y servicios a través de medios digitales. Un cliente que manda más que nunca, por la capacidad que le ha dado la tecnología para elegir y, además, para elegir de manera inmediata. Por ello y con el fin de garantizar su presencia en el mercado y la rentabilidad de sus operaciones, las empresas tienen la **necesidad de adaptarse a los cambios** de manera rápida, continua, eficiente y sostenible en el tiempo.

Cuando se habla de este tipo de adaptación al cambio, se habla de agilidad. Una agilidad fundamentada en los valores y principios del manifiesto ágil [1] y experimentada en primera instancia en el mundo del desarrollo de software, pero que se puede extrapolar a cualquier actividad, sector o industria que necesite operar su negocio de manera ágil de principio a fin, es decir, desde la identificación de la necesidad del cliente o la generación de una nueva idea hasta la comercialización y operación del nuevo producto o servicio, pasando por cualquiera de las fases de su ciclo de vida.

La agilidad aplica siempre que exista la necesidad de entregar valor al negocio de manera continua, en periodos cortos de tiempo y optimizando los esfuerzos para conseguirlo [2]. Para que la aplicación de la agilidad sea un éxito, una de las claves está en las personas y sus interacciones, es decir, en el equipo. Un equipo auto-organizado y focalizado en aportar valor de manera iterativa e incremental para garantizar la satisfacción del cliente y la calidad del producto. Un equipo ágil que incluye al cliente tanto en la solución propuesta como en el proceso ágil llevado a cabo.

En agilidad no existe una fórmula mágica [3], ni una solución única que resuelva todas y cada una de las situaciones que puedan surgir en torno al negocio y a los equipos de cada una de las empresas. Cada empresa es un mundo, no solo por el producto que comercializa o por la gestión que pueda hacer de su negocio en particular, lo es también por la idiosincrasia de las personas que forman sus equipos. Esto es lo que las hace realmente diferentes, lo que define su comportamiento, sus valores y en definitiva su

cultura [4]. Lo que sí existe son patrones que resuelven problemas recurrentes en determinados contextos y que cada empresa tendrá que aplicar en función de sus necesidades [5], [6], [7].

Para ayudar a las empresas a adaptarse a los nuevos y continuos cambios de escenario que la tecnología esta propiciando, se plantea la posibilidad de identificar los patrones que facilitan o impiden que **sus equipos se desenvuelvan de manera eficaz y eficiente** en este nuevo contexto. En este trabajo fin de máster se realiza un estado del arte para identificar las propuestas existentes en torno a patrones y anti-patrones en equipos ágiles, así como para analizar el contexto en el que los presentan.

En la actualidad, la evolución de las tecnologías de la información, en especial Internet, ha facilitado enormemente el acceso a todo tipo de documentación. Disponer de tanta información facilita la recopilación de datos, pero a la vez también es fácil pasar por alto alguna información que pueda parecer poco interesante para el objetivo de la investigación. Por ello, este trabajo fin de máster se ha realizado siguiendo una **revisión sistemática** adaptada a la ingeniería de software [8], [9], [10] como metodología rigurosa y repetible de búsqueda, recopilación, análisis y evaluación de la información relevante para la investigación.

1.2 Objetivos

En este apartado, se presentan el objetivo principal y los objetivos parciales que guían el desarrollo de este trabajo fin de máster.

Teniendo en cuenta la bondad de la agilidad y las ventajas de utilizar patrones sería interesante disponer de un marco de trabajo ágil en ingeniería del software que permita a los equipos garantizar la calidad de su producto, la satisfacción del cliente y la rentabilidad del negocio. Por ello, **el objetivo principal** de este trabajo fin de máster es **identificar si existen propuestas de patrones en torno a equipos ágiles y analizar el contexto en el que se proponen**.

Para alcanzar este objetivo, se define una serie de objetivos parciales que se describen a continuación:

Objetivo 1.- Establecer de un proceso de revisión sistemática para identificar y analizar trabajos existentes en la literatura.

Objetivo 2.- Definir un conjunto de características que permitan evaluar y comparar las diferentes propuestas encontradas en el proceso de Revisión Sistemática.

Objetivo 3.- Sintetizar los resultados obtenidos en la Revisión Sistemática a partir del análisis de los trabajos y de su comparación.

Objetivo 4.- Identificar posibles puntos de mejora en el estado del arte.

1.3 Estructura de la memoria

En este apartado, se presenta como se organizan el resto de los capítulos de este documento:

- En el **capítulo 2** se presentan los conceptos teóricos básicos necesarios para entender el contexto de trabajo fin de máster. En este capítulo se profundiza en conceptos como revisión sistemática, agilidad y patrones.

- En el **capítulo 3** se presenta el proceso de revisión sistemática. Dicho proceso establece el tema objetivo de la revisión sistemática y la estrategia de búsqueda de información que incluye la identificación de la cadena de búsqueda y las fuentes de información. Además, se presenta cómo se lleva a cabo la selección de los estudios primarios y la extracción de su información.

- En el **capítulo 4** se presenta el análisis de los trabajos seleccionados de acuerdo con los criterios de evaluación que son definidos también en este capítulo. El análisis de los estudios se presenta en términos de un estado del arte del tema objetivo de la investigación.

- En el **capítulo 5** se presentan los valores establecidos para cada uno de los criterios de evaluación. A partir de ellos, se presenta una comparativa de los estudios analizados en el capítulo anterior. El resultado del análisis de esta comparativa da pie a proponer una línea de investigación futura que permitirá mejorar el estado del arte.

- En el **capítulo 6** se presentan las conclusiones obtenidas tras realizar la revisión sistemática y también se presentan los trabajos futuros derivados de este trabajo fin de máster.

- En **Referencias** se incluyen todas las referencias bibliográficas citadas y consultadas durante la realización de este trabajo fin de máster.

- En el **Anexo A** se presentan las referencias específicas de los estudios primarios seleccionados y analizados durante el proceso de revisión sistemática.

- En el **Anexo B** se presentan la relación de todos los patrones identificados en este trabajo fin de máster, clasificados por el estudio primario que lo ha presentado.

- En el **Anexo C** se presenta toda la información relativa a formato de patrones que se ha considerado de interés para el presente o futuros trabajos.

2 Estudios previos

En este capítulo, se exponen los conceptos que se deben conocer para entender el contexto del trabajo realizado.

2.1 Revisión sistemática

Una revisión sistemática es una forma de recopilar, evaluar, sintetizar e interpretar todo el trabajo de investigación que hay en relación con un área concreta o fenómeno de interés. Para ello, se buscan y revisan los trabajos existentes en el campo investigado para dar respuesta a una serie de cuestiones concretas y específicas que se formulan y plantean antes de abordar la revisión.

A diferencia de una revisión de la literatura tradicional, la revisión sistemática minimiza la posibilidad de introducir sesgo en los resultados de la investigación siguiendo un método confiable, riguroso y auditable. Por tanto, el proceso de revisiones sistemáticas obliga a seguir una estrategia de búsqueda predefinida que asegure que esa búsqueda ha sido completa y que se han contemplado todos o gran parte de los trabajos que hay en el área investigada.

Para ello, se define un **protocolo que garantice el valor científico** de los resultados y que se pueda reproducir en un momento dado, obteniendo el mismo resultado [9]. Este proceso se construye en torno a un tema principal que representa el núcleo de la investigación.

El origen de la revisión sistemática está en el proceso Evidence-Based Medicine (EBM), un proceso que pone en valor la toma de decisiones en el cuidado de la salud en base a evidencias empíricas y no solo en función de las experiencias y criterio de los médicos. El mejor ejemplo de esta disciplina es la librería Cochrane que, desde 1972, recoge evidencias empíricas sobre las cuales cualquier médico puede fundamentar sus decisiones.

Extrapolar la buena práctica de tomar decisiones en función de la evidencia empírica al campo de la ingeniería del software fue una propuesta de Bárbara Kitchenham que ha ido evolucionando en el tiempo [8], [10]–[13]. Su aplicación permite dotar de valor científico a las revisiones tradicionales de literatura, definir estrategias de búsqueda óptimas, evaluar los estudios seleccionados y realizar una hipótesis a favor o en contra de dicha literatura.

Existen varios estudios en los que los investigadores han utilizado la propuesta de Kitchenham para realizar revisiones sistemáticas de literatura en torno a distintos campos de área de la ingeniería del software en general [14] y del desarrollo ágil en particular [15].

2.2 Agilidad

El término de agilidad en ingeniería del software surge a partir de la publicación del **manifiesto ágil** en 2001 [1], en el que 17 profesionales del desarrollo de software descubrieron mejores formas de hacer su trabajo en base a su experiencia. Estos descubrimientos se plasmaron en 4 postulados que resumen los 12 principios del manifiesto.

Los postulados ponen en valor los elementos enunciados en la parte izquierda frente a los enunciados en la parte derecha, sin que ello quiera decir que estos últimos no tengan su importancia en un desarrollo de software ágil. Eso sí, en su justa medida, siempre y cuando tengan sentido, es decir, aporten valor y no sean un desperdicio.

A continuación, se enuncian los postulados del manifestó ágil en su idioma original, en inglés, para evitar el posible sesgo en la interpretación de su traducción:

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

Estos postulados resumen los principios del manifiesto ágil que se enuncian a continuación[1] [16]:

1. Nuestra máxima prioridad es satisfacer al cliente a través de la entrega temprana y continua de software valioso.
2. Bienvenidos los requisitos cambiantes, incluso con el desarrollo avanzado. Los procesos ágiles aprovechan el cambio para la ventaja competitiva del cliente.
3. Ofrezca software funcionando con frecuencia, desde un par de semanas hasta un par de meses, preferiblemente en el plazo más corto.
4. Las personas de negocios y los desarrolladores deben trabajar juntos todos los días a lo largo del proyecto.
5. Construye proyectos alrededor de individuos motivados. Deles el entorno y el apoyo que necesitan, y confíe en ellos para hacer el trabajo.
6. El método más eficiente y efectivo de transmitir información hacia y dentro de un equipo de desarrollo es la conversación cara a cara.
7. El software funcionando es la principal medida del progreso.

[1] Traducción literal, lo más ajustada posible, para garantizar la esencia de cada principio.

8. Los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, los desarrolladores y los usuarios deberían poder mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y el buen diseño mejoran la agilidad.
10. La simplicidad, el arte de maximizar la cantidad de trabajo no hecho, es esencial.
11. Las mejores arquitecturas, requisitos y diseños surgen de los equipos **auto-organizados.**
12. A intervalos regulares, el equipo reflexiona sobre cómo ser más efectivo, luego ajusta su comportamiento en consecuencia.

El primer postulado del manifiesto ágil refleja la importancia del equipo (personas e interacciones) por encima de cualquier proceso o herramienta. De hecho, 6 de los 12 principios del manifiesto hacen referencia expresa a las personas que forman parte de un equipo [17]. Una de las características más determinantes y que mejor describen a un equipo ágil es la auto-organización, entendiendo la auto-organización como autonomía, adaptación y responsabilidad [18].

Dos de los frameworks ágiles más reconocidos en ingeniería de software son **XP** (Extreme Programming) [19] y **Scrum** [20]. Aunque en la actualidad, Scrum es el framework ágil más usado [21], en 2009 lo era XP [EP10].

La Figura 2.1 muestra los resultados del último informe sobre el estado de la agilidad publicado por VersiónOne[2] en 2019 [21], relativo a datos recogidos y analizados de 2018.

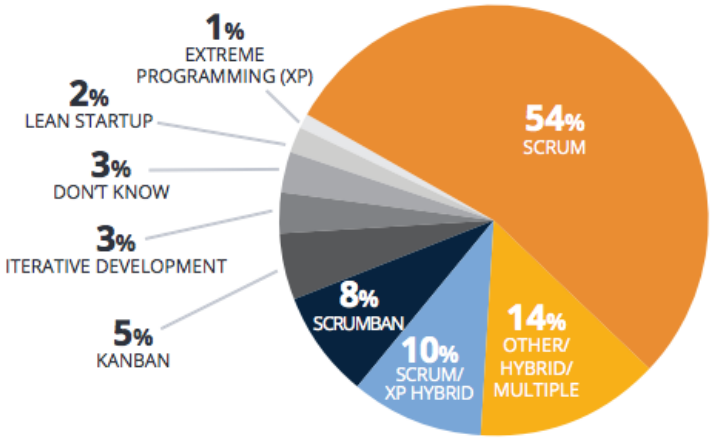

Figura 2.1 – Uso de frameworks ágiles en 2018 (VersionOne Survey [21]*)*

[2] VersionOne publica encuestas sobre el estado de la agilidad desde 2007 [54].

2.3 Patrones

Los conceptos de patrón y de lenguaje de patrones, muy consolidados en ingeniería del software, tienen su origen en las ideas del arquitecto Christopher Alexander en la década de los 70 para mejorar la calidad de la construcción de los edificios [22], [23]. Alexander recopila patrones sobre arquitectura, diseño urbano y habitabilidad de edificios, **describiendo para cada patrón un problema y ofreciendo una solución**.

> *"Cada patrón describe un problema que ocurre infinidad de veces en nuestro entorno, así como la solución al mismo, de tal modo que podemos utilizar esta solución un millón de veces más adelante sin tener que volver a pensarla otra vez" – Christopher Alexander –*

En 1987, Ward Cunningham y Kent Beck escriben el primer lenguaje de patrones en el ámbito del diseño de aplicaciones software, inspirados en las ideas de Alexander. Los autores recogen 5 patrones HCI[3] para diseñar ventanas en Smalltalk y los presentan en OOPSLA-87[4] [24], [25].

En 1991, James Coplien cataloga un lenguaje especifico de patrones para C++, al que llama Idioms, constituyendo uno de los tipos de patrones de referencia en desarrollo de software [26].

En 1992, Peter Coad escribe el primer artículo sobre patrones software en el que explica como los métodos para la creación de modelos OO[5] se enfocan a la construcción de bloques de bajo nivel, existiendo una similitud clara entre estos bloques y los elementos base descritos por Alexander [27], [28].

En 1993, Kent Beck y Grady Booch promueven un retiro con colegas interesados por la mejora continua del diseño y desarrollo de software. De esta reunión surge la formación de la asociación sin ánimo de lucro Hillside Group que fomenta la escritura de patrones a través de las conferencias PLoP[6] [29].

Pero será en 1995, cuando el concepto de patrón se consolida en el ámbito de la ingeniería de software a raíz de la publicación de 23 patrones de diseño [6] por parte de un grupo de trabajo autodenominado Gang of Four, formado por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides.

Un año después, en 1996, el concepto de patrón se extiende a otras áreas de la industria del software. Destacan dos autores, Frank Buschmann que establece patrones de arquitectura [30] y Martin Fowler que define patrones de análisis [31].

[3] Human Centered Infertace.
[4] Conferencia Anual ACM sobre Programación Orientada a Objetos, Sistemas, Lenguajes y Aplicaciones.
[5] Orientado a Objeto.
[6] Pattern Language of Programming, conferencia anual que promueve la creación y mejora de patrones.

En 1998, Scott Ambler define patrones de procesos para escalar aplicaciones [32]. También en este año, como contrapartida natural al estudio de patrones de diseño, William Brown desarrolla el concepto de anti-patrón [33]. Un anti-patrón es un patrón que invariablemente lleva a una mala solución o a no solucionar un problema.

En el año 2000, Linda Rising recoge y publica un inventario completo de patrones software publicados hasta esa fecha en conferencias, revistas y libros [34]. A partir de 2008 y en años sucesivos, Linda ha extendido el uso de patrones al desarrollo sostenible [35], [36] y al cambio organizacional [37]–[41].

En la actualidad, el término patrón está ampliamente extendido y aceptado entre los profesionales de ingeniería del software, que consideran como definición clásica de patrón "una buena solución a un problema recurrente en un contexto específico" [42].

El formato de patrón más común y reconocido en ingeniería de software es el formato de GoF[7] [6], en el que la mayoría de los estudios se han inspirado para proponer el suyo adaptándolo a su propósito. A continuación, se describen cada una de las secciones de este formato de patrón[8]:

- **Name**: Nombre único y descriptivo estándar del patrón por el cual será reconocido en la comunidad (normalmente se expresa en inglés).
- **Clasification**: Creacional, estructural o de comportamiento.
- **Intent**: ¿Qué problema pretende resolver el patrón? Se trata de una descripción del objetivo que hay detrás del patrón y la razón para usarlo.
- **Also Known As**: Otros nombres de uso común para el patrón.
- **Motivation (Forces)**: Escenario que consiste en un problema y un contexto en el que se puede utilizar el patrón.
- **Applicability**: Situaciones en las cuales el patrón aplica. Usos comunes y criterios de aplicabilidad del patrón.
- **Structure**: Representación gráfica del patrón. Diagramas de clases y diagramas de interacción pueden ser usados para este propósito.
- **Participants**: Enumeración de las entidades y roles que participan en el patrón.
- **Collaboration**: Explicación de las interrelaciones que se dan entre los participantes.
- **Consequences**: Descripción de los resultados, efectos secundarios y compensaciones causados por el uso del patrón. Consecuencias positivas y negativas derivadas de la aplicación del patrón.
- **Implementation**: La solución que plantea el patrón. Técnicas o comentarios oportunos de cara a la implementación del patrón.
- **Sample Code**: Ilustración de como el patrón puede ser usado en un lenguaje de programación. Código fuente ejemplo de la implementación del patrón.
- **Known Uses**: Ejemplos de usos reales del patrón.

[7] Gang of Four, grupo de trabajo que define y publica los patrones de diseño [6].
[8] El nombre de la sección se indica en el lenguaje original, en inglés.

- **Related Patterns**: Referencias cruzadas con otros patrones. Otros patrones que tienen alguna relación con el patrón. Discusión de las diferencias entre el patrón y patrones similares.

3 El proceso de la revisión sistemática

En este capítulo se presenta el proceso de revisión sistemática llevado a cabo en este trabajo fin de máster. A continuación, se explican aspectos como la metodología usada para su desarrollo, el principal objetivo planteado y el protocolo seguido para la búsqueda y extracción de la información.

3.1 Metodología

Una revisión sistemática es una metodología específica de investigación que recopila y evalúa toda la información disponible en torno a un tema de interés de un modo formal, sistemático y repetible. Dada la gran cantidad de información que se recopila, es necesario emplear un método riguroso que permita identificar los resultados reales del tema investigado. Por tanto, para realizar una revisión sistemática es necesario seguir una secuencia de pasos bien definida y estricta, es decir, un protocolo que permita evaluar críticamente los estudios para posteriormente recopilar y analizar los datos.

Atendiendo a la propuesta de Kitchenham [8] para realizar este tipo de análisis en el ámbito de la Ingeniería del Software, el proceso de revisión sistemática consta de tres fases principales: **planificación, ejecución y reporte de la revisión**. Sin embargo, Biolchini [9] presenta una nueva aproximación del proceso de revisión sistemática basada en la propuesta de Kitchenham. En esta aproximación se propone un proceso de conducción de revisiones sistemáticas en cuatro fases: planificación, ejecución, análisis y almacenamiento de los resultados. La Figura 3.1 muestra este proceso.

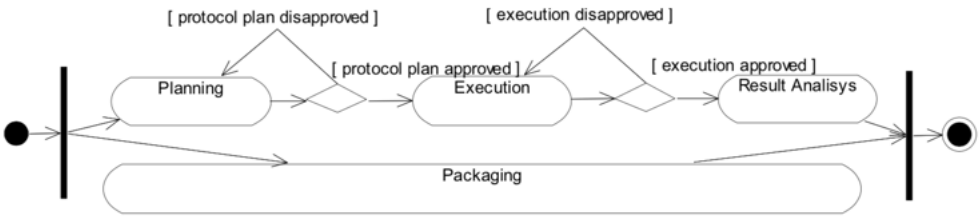

Figura 3.1 - Proceso de Revisión Sistemática (Bolchini)

En la fase de planificación (**Planning**), se definen los objetivos de la investigación y el protocolo a seguir durante la revisión, concretando en el protocolo la pregunta de investigación y los métodos que se van a utilizar para ejecutar la revisión. En la fase de ejecución (**Execution**), se identifican, seleccionan y evalúan los estudios primarios según

los criterios de inclusión y exclusión indicados en el protocolo. En la fase de análisis de resultados (**Result Analisys**), se extraen y sintetizan los datos de cada estudio primario seleccionado. En paralelo a la ejecución de estas tres fases, los resultados se van almacenado en lo que Biolchini [9] denomina fase de empaquetado (**Packaging**). Durante el proceso de conducción de la revisión, se indican dos puntos de control: el primero para evaluar el protocolo en la fase de planificación, justo antes de ejecutar la revisión; y el segundo para chequear los motores de búsqueda de información en la fase de ejecución.

Dentro de cada fase pueden darse secuencias iterativas que permiten refinar el protocolo definido en la fase de planificación, en el momento de ejecutarlo. Por ejemplo, al aplicar los criterios de inclusión y exclusión en la selección de los estudios o al aplicar los criterios de evaluación en el análisis de los estudios.

En los siguientes apartados de este capítulo se presenta el objetivo de la revisión sistemática, la estrategia de la búsqueda, la selección de los estudios y la extracción de datos.

3.2 Objetivo de la revisión sistemática

El objetivo de la investigación al que se pretende dar respuesta con esta revisión sistemática es el siguiente:

¿Qué propuestas de patrones existen en torno a los equipos ágiles?

Como resultado de este trabajo se espera obtener, además de las propuestas identificadas y analizadas en su contexto, una **recopilación patrones** que faciliten el desempeño de un equipo ágil en cualquier actividad del ciclo de vida de un producto software.

3.3 Estrategia de búsqueda

Una vez definido el objetivo, se plantea la estrategia de búsqueda identificando las palabras clave y seleccionando las fuentes de información más apropiadas para esta investigación.

3.3.1 Palabras clave

Para crear la cadena de búsqueda de información qué de respuesta al objetivo planteado, se identifican las siguientes palabras clave principales: **agile**, **team** y **pattern**, y las siguientes relacionadas: **scrum, kanban, XP, "extreme programming", people, pattern, model, "good practice"** y **PLoP**.

Estos términos combinados adecuadamente mediante diferentes conectores lógicos dan lugar a la cadena básica de búsqueda. La Tabla 3.1 muestra la cadena básica de búsqueda para esta investigación.

Cadena Básica de Búsqueda
((agil OR scrum OR kanban OR XP OR "extreme programming") AND (team* OR people*) AND (pattern OR model OR "good practice" OR PLoP))*

Tabla 3.1 - Cadena Básica de Búsqueda

3.3.2 Selección de fuentes

Las fuentes de información seleccionadas inicialmente son las mostradas en la Tabla 3.2. La cadena básica de búsqueda se adapta, de forma individualizada, a las características del motor de búsqueda de cada fuente de información para realizar la búsqueda inicial y analizar los resultados.

Fuente de Información	Web
ACM Digital Library	https://dl.acm.org
IEEE Xplore	https://ieeexplore.ieee.org
Scient Direct	https://www.sciencedirect.com
Scopus	https://www.scopus.com
Google Scholar	https://scholar.google.es

Tabla 3.2 - Lista inicial de fuentes de información

3.4 Selección de los estudios

Los resultados de las búsquedas en cada una de las fuentes seleccionadas, previa adaptación de la cadena de búsqueda en las fuentes seleccionadas se refleja en la Tabla 3.3.

Conviene señalar que, aunque la primera búsqueda para la investigación se realizo hace casi un año (en agosto de 2018), se ha repetido hace poco más de un mes (en mayo de

2019) arrojando los mismos resultados en cuanto a estudios primarios seleccionados. Es decir, en este último año no se ha publicado ningún estudio nuevo relacionado con patrones en equipos ágiles.

Fuente	Cadena de búsqueda	Ámbito	F.Consulta	Resultados
ACM Library	*((agil* OR scrum OR kanban OR XP OR "extreme programming") AND (team* OR people*) AND (pattern OR model OR "good practice" OR PLoP))*	Metadatos	13-may-19	303
IEEE Xplore	*((agil* OR scrum OR kanban OR XP OR "extreme programming") AND (team* OR people*) AND (pattern OR model OR "good practice" OR PLoP))*	Metadatos	13-may-19	556
Scient Direct	*((agil* OR scrum OR kanban OR XP OR "extreme programming") AND (team* OR people*) AND (pattern OR model OR "good practice" OR PLoP))*	TAK	13-may-19	39
Scopus	*((agil* OR scrum OR kanban OR XP OR "extreme programming") AND (team* OR people*) AND (pattern OR model OR "good practice" OR PLoP))*	TAK	17-may-19	1.717
Google Scholar	*((agil* OR scrum OR kanban OR XP OR "extreme programming") AND (team* OR people*) AND (pattern OR model OR "good practice" OR PLoP))*	All	17-may-19	2.960
				5.575

Tabla 3.3 - Resultados tras adaptar la cadena de búsqueda a las fuentes

Finalmente se descarta Google Scholar por ser un contenedor de referencias con menos reconocimiento dentro de la comunidad científica que Scopus, quedando **2.615 estudios** resultantes de la búsqueda para empezar un proceso iterativo de selección de estudios en función de los siguientes criterios de inclusión y exclusión:

- **Criterios de inclusión**: estudios de cualquier fecha que contemplen alguna de las palabras clave de la búsqueda en el título, que estén basados en alguna actividad relacionada con el ciclo de vida de un producto software y que estén escritos en ingles.

- **Criterios de exclusión**: estudios que estén duplicados o que no cumplan los criterios de inclusión. No se considerarán los estudios relativos a un campo diferente a la ingeniería del software, como, por ejemplo, los de medicina, veterinaria, psicología, electrónica o deporte. Se excluirán los estudios que no se centren explícitamente en el concepto “agile” derivado del manifiesto ágil [1], es decir, que se refieran a “ágil” como un adjetivo. Tampoco se contemplarán estudios relativos a prácticas, modelos, marcos o herramientas ágiles que no hagan ninguna referencia expresa a patrones en alguno de sus metadatos (título, resumen y palabras clave).

Aplicando estos criterios de inclusión y exclusión se realiza un proceso depurativo de los estudios en 4 iteraciones. En la Tabla 3.4, se muestra el número de estudios resultante de cada iteración.

Estudios seleccionados en búsqueda inicial	2615
Estudios no duplicados	2030
Estudios contemplados según título	873
Estudios relevantes según resumen	302
Estudios primarios	**22**

Tabla 3.4 - Total de Estudios Primarios seleccionados

En la primera iteración, se eliminan los estudios repetidos, resultando **2.030 estudios no duplicados**. En la segunda iteración, se revisa el título de los estudios no duplicados para depurar la selección en función del cumplimiento de los criterios de inclusión y exclusión, resultando **873 estudios a contemplar en la siguiente iteración**. En la tercera iteración, se depuran los estudios restantes revisando el resumen del estudio en función del cumplimiento de los criterios, resultando un total de **302 estudios relevantes**. En la cuarta y última iteración, se revisa el texto completo de cada estudio relevante descartando los estudios según los criterios. Los estudios relevantes que finalmente no cumplen los criterios de exclusión son considerados como estudios primarios.

El resultado del proceso de selección es de **22 estudios primarios** a analizar en la revisión sistemática. La Tabla 3.5 muestra la relación de estos estudios primarios ordenados por fecha de publicación,

EP	Año	Título
EP01	2004	AntiPractices: AntiPatterns for XP practices
EP02	2006	Value based extreme programming
EP03	2006	Patterns for agile development practice part 3 (version 4)
EP04	2006	Programmers are from Mars, customers are from Venus: A practical guide for customers on XP projects
EP05	2007	A process pattern language for agile methods
EP06	2008	Agile Learning & Teaching with Wikis: Building a Pattern
EP07	2010	How Much is Just Enough?: Some Documentation Patterns on Agile Projects
EP08	2011	Communication patterns of agile requirements engineering
EP09	2011	A Pattern Language for Release and Deployment Management
EP10	2011	Extending patterns for fearless change
EP11	2011	The Art of Software Estimation Pattern Language
EP12	2012	A pattern language for agile software estimation
EP13	2013	Is there an agile handover? An empirical study of documentation and project handover practices across agile software teams
EP14	2014	Teams that finish early accelerate faster: A pattern language for high performing scrum teams
EP15	2015	Animating Organizational Patterns - KO
EP16	2015	Patterns for integrating agile development processes and user centred design
EP17	2015	Distributed Agile Patterns for Offshore Software Development
EP18	2016	Agile base patterns in the agile canon
EP19	2016	Exploring ScrumBut - An empirical study of Scrum anti-patterns
EP20	2017	Patterns of collaboration driven by requirements in agile software development teams: Findings from a multiple case study
EP21	2017	Anti-patterns in Agile Adoption: A Grounded Theory Case Study of One Irish IT Organisation
EP22	2018	A program backlog story with patterns: Expanding the magic backlog pattern collection

Tabla 3.5 - Listado de Estudios Primarios por orden cronológico de aparición

3.1 Extracción de la información

Para extraer los datos más relevantes de identificación de cada estudio que ayuden a dar respuesta al objetivo de la investigación, se utiliza una adaptación de la plantilla de Pino [43]. La Tabla 3.6 muestra la plantilla utilizada en esta investigación.

Título	Título del estudio
Autores	Autores del estudio
Resumen	Resumen del estudio (Abstract)
Año	Año de Publicación
Publicado en …	Revista, Libro, Actas de Congreso, etc. donde se ha publicado el estudio
Inf. relevante	Información relevante para la investigación de la revisión sistemática

Tabla 3.6 - Formato de extracción de los Estudios Primarios

4 Ejecución de la revisión sistemática

Después de extraer los estudios primarios y su información relevante, se analiza en detalle cada estudio con el fin de dar respuesta al objetivo planteado. Este análisis se presenta en términos de un estado del arte sobre los patrones y anti-patrones en equipos ágiles en un contexto de Desarrollo de Software Ágil.

4.1 Criterios de la Evaluación

Se han identificado un conjunto de criterios de evaluación, de modo qué, cada uno de ellos se evaluará de forma individual para cada uno de los estudios propuestos. El objetivo de los criterios es dar respuesta a cuestiones relacionadas con:

- La cantidad y calidad de los patrones.
- El entorno ágil en el que se presentan.
- La orientación al proceso o al equipo.
- La aplicación en el ciclo de vida software.

Para ello, se definen los siguientes criterios de evaluación:

1. **Identificación del patrón**. Valora la naturaleza del patrón tratado en el estudio, es decir, lo tipifica en función de que sea un patrón o de un anti-patrón.

2. **Número de patrones**. Este criterio permite conocer el número de patrones claramente identificados, al menos con nombre, en el estudio.

3. **Marco ágil referenciado**. Establece el entorno ágil en el que está enmarcado el patrón en ese estudio en concreto: XP, SCRUM, ambos o AGILE en general.

4. **Explicación del patrón**. Evalúa en que medida se describe el patrón identificado en el estudio ya sea con una descripción (corta o larga) o según las secciones contempladas en su formato.

5. **Formulación del patrón**. Este criterio determina si el patrón se presenta con un formato determinado, ya sea este un formato estándar reconocido o adaptado dependiendo de las circunstancias de su aplicación.

6. **Descripción del formato**. Confirma si en el estudio se explica lo que significan cada una de las secciones del formato en el que se va a presentar el patrón en el estudio.

7. **Relación entre patrones**. Indica si los patrones recogidos en el estudio están relacionados entre si, y si lo están, si se percibe la relación a primera vista, es decir, a través de un gráfico o de una tabla.

8. **Orientación al proceso y/o al equipo**. Confirma si el foco de los patrones contenidos en el estudio esta relacionado fundamentalmente con algún aspecto del proceso de software o fundamentalmente con algún aspecto relativo a la parte humana de un equipo o con ambos.

9. **Ámbito en ciclo software**. Este criterio determina los aspectos relacionados con el ámbito de aplicación del patrón dentro del ciclo de vida de un producto software.

Una vez analizados los estudios primarios de acuerdo con los criterios de evaluación descritos, en el apartado 5.1.1 se presentan los valores que pueden tomar cada uno de los criterios y se establece una comparativa de criterios de evaluación entre los estudios.

4.2 Estudios analizados

En este apartado se realiza un análisis de todos los estudios primarios que han sido seleccionados y recogidos en la Tabla 3.5, aplicando los criterios de evaluación propuestos en el apartado 4.1 a cada estudio.

Cada estudio primario se resumirá, **a modo de patrón**, siguiendo el siguiente formato: propósito del estudio, formato del patrón, enumeración de patrones, relación gráfica entre patrones del estudio, orientación del patrón, aplicación del patrón en el ciclo de vida software y ficha resumen de criterios de evaluación del estudio. De manera individual, el análisis de cada estudio puede ofrecer información adicional y específica que se aportará inmediatamente antes o después de su ficha de evaluación.

Los estudios se presentan por **orden cronológico** para entender como ha ido evolucionando el concepto de patrón en los equipos ágiles, tanto en su formato como en su enfoque, y como ha ido creciendo el número de patrones en cada ámbito de aplicación del ciclo de vida de un producto software ágil. Por temas estéticos, cada estudio se nombrará con su título completo o con un título reducido que lo identifique sin ninguna duda y, en la medida de lo posible, sin penalizar su esencia.

Los estudios primarios que referencian colecciones de patrones, de los propios autores o de otros, aparecerán marcados con un “*” en su título

Los nombres de los estudios, así como todos los nombres y formatos de patrones aparecerán en el **idioma de la publicación original** para evitar el posible sesgo derivado de la interpretación de su traducción.

4.2.1 EP01 (2004): AntiPractices: AntiPatterns for XP Practices

Autores: Y. Kuranuki y K. Hiranabe

Se trata de un informe de experiencia que recopila 10 anti-prácticas, en formato patrón, detectadas al introducir la programación extrema (XP) en un proyecto.

El formato patrón utilizado en el estudio atiende a la siguiente estructura:
Name-Background-Sympton-Cause-Ideal-Prescription.
La descripción de cada una de las secciones del formato del patrón se puede consultar en el Anexo C de este documento.

En el estudio se presentan 3 anti-patrones que están claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado:

1. Brownie´s Works – "The boss refactored our code!"
2. Anybody Syndrome – "I'm not necessary"
3. Pairing Prison – "I'm always under observation!"

Adicionalmente, se identifican otros 7 anti-patrones que los autores pretenden explicar en detalle en futuros trabajos.

1. Powerful Owner – "We are under his thumb!"
2. Outnumbers View – "We are a minority group"
3. Imperial XP – "Why do I do such a thing?"
4. Loose Control – "Sure. XP is free, you know"
5. Frozen Pair – "The two are pairing … again!"
6. Formal Format – "Do we have to do all the practices just as in the book?"
7. Misleading influenza – "I heard you are doing THE XP"

En total, el estudio recoge 10 anti-patrones XP, todos ellos orientados fundamentalmente al equipo y de aplicación general a cualquier ámbito del ciclo de vida de un proyecto software. En el estudio no se relacionan gráficamente los anti-patrones entre si.

Para los autores, una anti-practica XP es un "error cometido con frecuencia en proyectos XP". En su opinión y como conclusión del informe, los problemas que surgen al aplicar XP no son debidos a la tecnología ni a la aparente saturación de prácticas del framework, sino a las personas que forman parte del equipo y a su organización. Por tanto, recomiendan que el coach esté muy pendiente de las "sutilezas" de las personas y no solo de ajustar y añadir prácticas. Todos los anti-patrones que se presentan en el estudio han sido verificados por los autores en más de un proyecto en Japón.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.1.

Criterios Evaluación EP1	Valor	Lectura
Identificación del patrón	A	Anti-patrón identificado con nombre
Número de patrones	3+7	3 formulados + 7 nombrados
Marco ágil referenciado	XP	Extreme Programming
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	3	Usa formato adaptado de uno reconocido
Descripción del formato	1	Se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	EQU	Anti-patrones orientados al equipo
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.1 - Ficha de evaluación EP01

4.2.2 EP02 (2006): Value based Extreme Programming

Autor: M. Yap

Se trata de un informe de experiencia que recopila 4 prácticas, en formato patrón, identificadas a la hora de entregar el máximo valor en el plazo comprometido en proyectos XP.

El formato del patrón utilizado en el estudio sigue la siguiente estructura:

Name-Category-Problem-Context-Solution

El estudio no describe explícitamente cada una de las secciones del formato del patrón planteado.

En el estudio se presentan 4 patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Value-based Investment Decisions
2. High Confidence Stories First
3. Incremental Story Delivery
4. Story Ownership

En su conjunto, los patrones del estudio están orientados tanto al equipo como al proceso y aplican a la entrega del software poniendo el foco en maximizar el valor entregado y optimizar los costes. En este estudio no se relacionan gráficamente los patrones entre si.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.2.

Criterios Evaluación EP2	Valor	Lectura
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	4	4 formulados
Marco ágil referenciado	XP	Prácticas XP y principios Lean
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	ENT	Foco en la entrega de valor

Tabla 4.2 - Ficha de evaluación EP02

Los 4 patrones están dirigidos tanto a ayudar a los clientes a seleccionar y priorizar las funciones de alto valor en relación con el presupuesto y con el coste, como a garantizar su entrega en el plazo comprometido, fomentando la responsabilidad compartida entre el cliente y el equipo y facilitando el empoderamiento de cada uno de los miembros del equipo.

Todos los patrones del estudio están basados en la aplicación de prácticas XP (como “highest value features first” o “don’t do anything extra”) y principios Lean (como “eliminate waste”).

4.2.3 EP03 (2006): Patterns for Agile Development Practice part 3*

Autor: Joseph Bergin

Se trata de una propuesta de 10 patrones, basados en prácticas XP y dirigidos a quienes vayan a empezar un proyecto con XP o con otro marco ágil.

El formato de patrón utilizado en el estudio atiende a la siguiente estructura:
Name-Context-Problem-Forces-Therefore (Solution)-Commentary-Consequences.
La descripción de cada una de las secciones del formato del patrón se puede consultar en el Anexo C de este documento.

En el estudio se presentan 10 patrones que están claramente identificados por su nombre y explicados en detalle atendiendo a un formato concreto.

1. Stand Up Meeting
2. Test First
3. Once and Only Once
4. Executable Tests
5. Continuous Integration
6. Social Tracker
7. Project Diary (Record of Velocity)
8. Customer Checks off Tasks
9. Customer Obtains Consensus
10. Individual Stakeholder Budgets

Adicionalmente, se enuncian otros 50 patrones más que forman parte de una recopilación de patrones recogidos en anteriores estudios del autor [44], [45]. El total de los 60 patrones incluidos en el estudio EP03 puede consultarse en el Anexo B de este documento.

En su conjunto, los 10 patrones del estudio están orientados tanto al equipo como al proceso y aplican a la entrega del software poniendo el foco en maximizar el valor y optimizar los costes. Este estudio no relaciona gráficamente los patrones entre si.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.3.

Criterios Evaluación EP3*	Valor	Lectura
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	10+50	10 formulados + 50 nombrados
Marco ágil referenciado	XP	Extreme Programming
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	1	Se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.3 - Ficha de evaluación EP03

El autor considera que XP es un lenguaje de patrones en el que las propias prácticas de XP son patrones base sinérgicos y generativos. Por ello, las prácticas XP también están incluidas en la recopilación.

4.2.4 EP04 (2006): Programmers are from Mars, Customers are from Venus

Autores: Angela Martin, James Noble y Robert Biddle

Se trata de una investigación, basada en experiencia práctica, en la que se proponen 18 patrones dirigidos a aumentar la efectividad del rol de cliente en proyectos XP.

El formato de patrón utilizado en el estudio atiende a la siguiente estructura:
Name-Problem-Forces-Solution-Forces Resolved
El estudio no describe explícitamente cada una de las secciones del formato del patrón planteado.

En el estudio se presentan 18 patrones que están claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado. De los 18 patrones, 9 son relativos a los roles claves del cliente y los otros 9 a las prácticas que permitiría el éxito del proyecto.

Foco en los roles del cliente:

1. Geek Interpreter
2. Technical Liaison
3. Political Advisor
4. Acceptance Tester
5. User Interface Designer
6. Technical Writer
7. Diplomat
8. Super-Secretary
9. Negotiator

Foco en las prácticas del cliente:

1. Programmer On-Site
2. Customer's Apprentice
3. Programmer Holiday
4. Story Standards
5. Show & Tell
6. Customer Pairing
7. Customer Counselor
8. Look Before You Leap
9. Three-month Calibration

En su conjunto, los 18 patrones del estudio están orientados tanto al equipo como al proceso y aplican a la entrega del software poniendo el foco en maximizar el valor y optimizar los costes. Este estudio no relaciona gráficamente los patrones entre si.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.4.

Criterios Evaluación EP4	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	9+9	18 formulados
Marco ágil referenciado	XP	Extreme Programming
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.4 - Ficha de evaluación EP04

El autor realizó la investigación a través de entrevistas a clientes de proyectos por todo el mundo durante 3 años, valorando sobre todo las circunstancias de tareas complejas y difíciles.

La idea de aplicar estos los patrones es incrementar la velocidad y confiabilidad del proyecto garantizando un ritmo de trabajo sostenible de un equipo en cualquier entorno ágil, poniendo el foco en el rol del cliente dentro del equipo.

4.2.5 EP05 (2007): A Process Pattern Language for Agile Methods

Autor: Xiang-xi Meng, Ya-sha wang, Lei Shi y Feng-jian Wang

Se trata de la propuesta de un lenguaje de patrones de proceso para generar diferentes modelos de procesos software dependiendo de las circunstancias concretas del proyecto en cuanto, entre otros, al equipo, a las prácticas o a la experiencia en frameworks ágiles.

El formato de patrón utilizado en el estudio atiende a la siguiente estructura:

Name-Context-Problem-Solution-Consequence-Related Pattern

La descripción de cada una de las secciones del formato del patrón se puede consultar en el Anexo C de este documento.

Aunque en la Figura 4.1 se muestra el lenguaje de patrones de proceso propuesto por los autores, el estudio sólo detalla el del "**Sprint**", estando este claramente identificado por su nombre y explicado en detalle atendiendo al formato indicado.

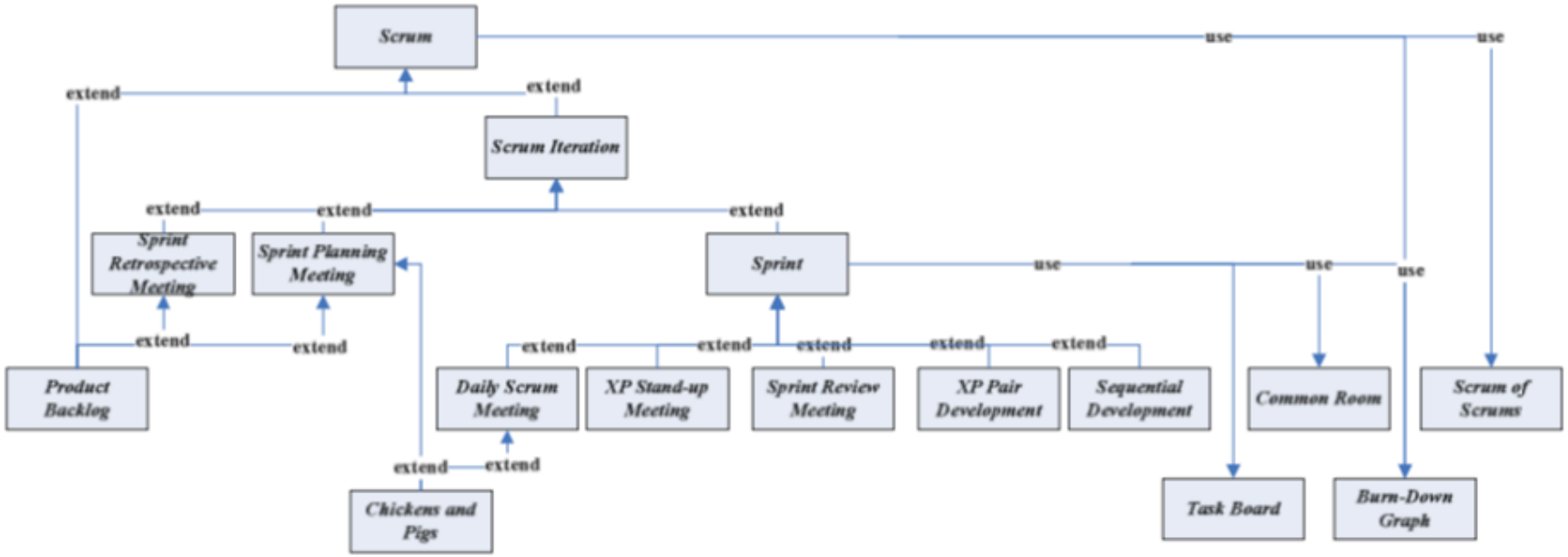

Figura 4.1 – Lenguaje de patrones de proceso para frameworks ágiles [EP05]

El patrón “Sprint” está orientado fundamentalmente al proceso y es de aplicación general en cualquier ámbito del ciclo de vida de un proyecto software.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.5.

Criterios Evaluación EP5	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	1	1 formulado
Marco ágil referenciado	AGILE	Agile en general
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	3	Usa formato adaptado de uno reconocido
Descripción del formato	1	Se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	PRC	Lenguaje de patrones orientado al proceso
Ámbito en ciclo software	GEN	De aplicación general al proceso ágil de software

Tabla 4.5 - Ficha de evaluación EP05

El autor pretende demostrar que el lenguaje de patrones propuesto puede integrar modelos y generar diferentes modelos de procesos software dependiendo de las circunstancias concretas de los proyectos relativas al equipo, a las prácticas o a su experiencia en frameworks ágiles.

El estudio ejemplifica el uso del patrón Sprint para dos tipos de organizaciones diferentes: una sin experiencia en frameworks ágiles pero que quiere empezar a usar Scrum, y otra habituada a usar el framework XP pero que quiere introducir Scrum para complementar sus prácticas de gestión.

4.2.6 EP06 (2008): Agile Learning & Teaching with Wikis

Autor: Marija Cubric

Se trata de una propuesta de patrón de proceso para el aprendizaje de conceptos ágiles basado en principios [1] y técnicas [46] de desarrollo ágil.

El formato de patrón utilizado en el estudio atiende a la siguiente estructura:

Name-Intent-Motivation (Forces)-Applicability-Structure-Participants-Collaboration-Consequences-Implementacion-Known Uses-Related

El estudio en sí es la descripción explicita de las secciones del formato de patrón planteado.

En el estudio se presenta un único patrón relativo a un proceso ágil de formación en Agilidad, "**Agile Learning and Teaching**", claramente identificado por su nombre y explicado en detalle atendiendo al formato indicado.

La Figura 4.2 muestra el patrón de proceso atendiendo al proceso Scrum ya que el autor lo considera uno de los procesos de gestión de proyecto y desarrollo ágil más populares.

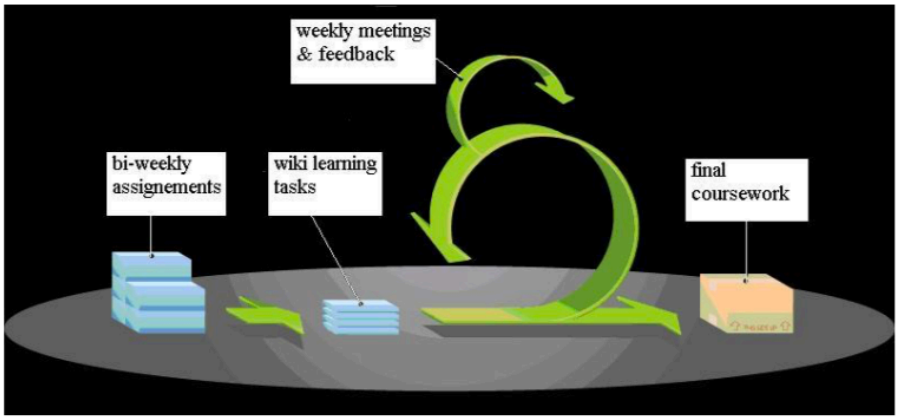

Figura 4.2 – Proceso Agile Learning and Teaching basado en Scrum [EP06]

El autor considera que el entender los principios ágiles y poder aplicar técnicas ágiles a cualquier actividad es fundamental en la capacitación de los estudiantes de cara a su empleabilidad.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.6.

Criterios Evaluación EP6	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	1	1 formulado
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	3	Usa formato adaptado de uno reconocido
Descripción del formato	1	Se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	PRC	Lenguaje de patrones orientado al proceso
Ámbito en ciclo software	GEN	De aplicación general a los procesos de software

Tabla 4.6 - Ficha de evaluación EP06

Aunque este patrón de proceso ágil está definido inicialmente para el aprendizaje y enseñanza de conceptos ágiles, podría extrapolarse a la formación de cualquier materia orientada al desarrollo de habilidades no cognitivas como por ejemplo el trabajo en equipo, dar y recibir retroalimentación, la escritura colaborativa o el desarrollo de un aprendizaje incremental e iterativo.

4.2.7 EP07 (2010): How Much is Just Enough?

Autores: Rashina Hoda, James Noble y Stuart Marshall

Se trata de una propuesta de 5 patrones para ayudar a concretar la documentación justa y necesaria que debería producir un equipo ágil.

El formato de patrón utilizado en el estudio responde a la siguiente estructura:

Name-Context-Example-Problem-Forces-Solution-Consequences-Example Resolved- Known Uses- Related Practices

La descripción de cada una de las secciones del formato del patrón se puede consultar en el Anexo C de este documento.

En el estudio se presentan 5 patrones que están claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Fake Documentacion
2. Time Stamp
3. e-Backup
4. Project Dictionary
5. Advantage Agile

Los 5 patrones en su conjunto están orientados fundamentalmente al proceso y focalizados en la generación de documentación en los proyectos. La Figura 4.3 muestra la relación existente entre los patrones.

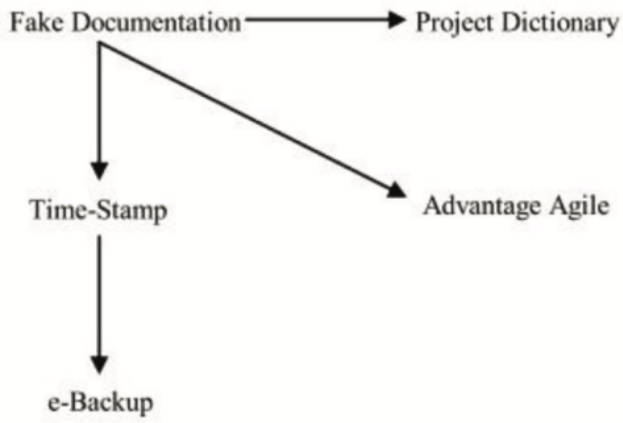

Figura 4.3 – Relación entre patrones [EP07]

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.7.

Criterios Evaluación EP7	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	5	5 formulados
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	1	Se describe el formato del patrón en el estudio
Relación entre patrones	1	Si presenta relación visual entre patrones
Orientación proceso/equipo	PRC	Patrones orientados al proceso
Ámbito en ciclo software	DOC	Foco en la documentación de los proyectos

Tabla 4.7 - Ficha de evaluación EP07

Los 5 patrones están dirigidos a los equipos ágiles para ayudarle a determinar “just enough” de documentación en su contexto especifico y a usar esos patrones como estrategias para afrontar algunos de los retos comunes que supone el adoptar frameworks ágiles.

4.2.8 EP08 (2011): Communication Patterns of Agile Requirements

Autores: Nik Nailah Binti Abdullah, Shinichi Honiden, Helen Sharp, Bashar Nuseibeh y David Notkin

Se trata de un estudio empírico sobre e impacto de los patrones de comunicación en las actividades clave de la ingeniería de requisitos como son la recopilación, la evolución y la aclaración de estos.

El estudio **no identifica** ningún patrón en concreto, aunque sí pone de manifiesto dos evidencias en torno a los patrones de comunicación:

1. La ingeniería de requisitos se basa en la comunicación y colaboración del equipo más que en herramientas de análisis y documentación especificas.

2. El uso disciplinado de dos artefactos físicos, tan simples como son las tarjetas de historia y el muro, son claves en el desarrollo de requisitos siempre y cuando estén complementados por una actividad social adecuada.

Los patrones de comunicación están orientados fundamentalmente al equipo y, en este caso, están focalizados en la ingeniería de requisitos del ciclo de vida de un proyecto software.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.8.

Criterios Evaluación EP8	Valor	Leyenda
Identificación del patrón	P	Patrones de comunicación
Número de patrones	0	No identifica patrones concretos
Marco ágil referenciado	XP	Extreme Programming
Explicación del patrón	0	No explica el patrón
Formulación del patrón	0	No usa formato
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	EQU	Patrones orientados al equipo
Ámbito en ciclo software	REQ	Foco en la ingeniería de requisitos

Tabla 4.8 - Ficha de evaluación EP08

El autor considera que los patrones de comunicación sugieren que compartir la conceptualización de los requisitos es una idea más profunda que la comprensión

compartida que reconocen algunos frameworks ágiles, en su caso XP, y más sofisticada que el entendimiento común necesario para trabajar en equipo.

4.2.8 EP09 (2001): A Pattern Language for Release and Deployment

Autor: Rick Rodin, Jon Leet, Maria Azua y Dwight Bygrave

Se trata de una propuesta de 6 patrones para gestionar el lanzamiento y despliegue de versiones de aplicaciones en un entorno de desarrollo de software ágil.

El formato de patrón utilizado en el estudio responde a la siguiente estructura:
Name-Context-Problem-Forces-Solution-Resulting Context-Rationale
El estudio no describe explícitamente cada una de las secciones del formato del patrón planteado.

En el estudio se presentan 6 patrones, relacionados entre si, que están claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Obtain certificate originality
2. Release Scheduler
3. Release Acceptance
4. Emergency Release
5. Deployment Preparation
6. Application Retirement

Los 6 patrones en su conjunto están orientados fundamentalmente al proceso y aplican a la entrega, poniendo foco en la liberación y despliegue del. La Figura 4.4 muestra la relación existente entre los patrones.

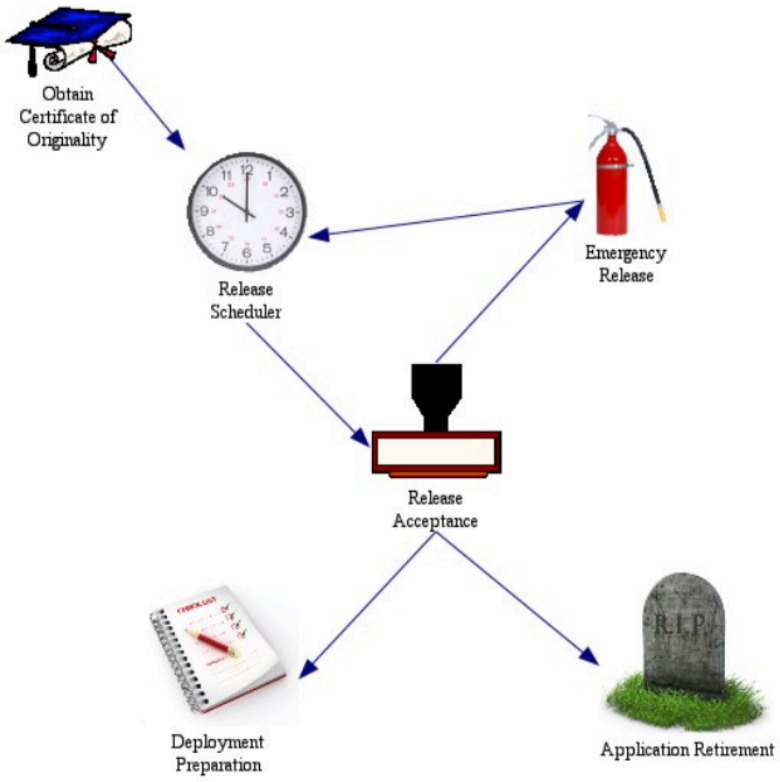

Figura 4.4 - Relación entre patrones [EP09]

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.9.

Criterios Evaluación EP9	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	6	6 formulados
Marco ágil referenciado	AGILE	Agile en general
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	1	Si presenta relación visual entre patrones
Orientación proceso/equipo	PRC	Patrones orientados al proceso
Ámbito en ciclo software	ENT	Foco en la liberación y despliegue de la entrega

Tabla 4.9 - Ficha de evaluación EP09

El autor considera que una versión es un conjunto de cambios autorizados en una aplicación que ya está en producción. En contextos ágiles se liberan y despliegan muchas versiones pequeñas cada poco tiempo. Esta entrega continua de versiones debe seguir un riguroso proceso que gestione el despliegue adecuadamente evitando problemas típicos como cuellos de botella en los recursos del sistema y garantizando el despliegue de versiones de emergencia real. En caso de no existir control sobre los cambios liberados, la calidad del sistema se verá deteriorada.

Los 6 patrones están orientados a crear un proceso ordenado de gestión de versiones que ayude a desplegar versiones de software de forma disciplinada y controlado dentro de un entorno ágil.

4.2.9 EP10 (2011): Extending Patterns for Fearless Change*

Autores: Daniel Cukier y Fabio Kon

Se trata de una propuesta de 4 patrones de comportamiento para introducir nuevas ideas en las organizaciones.

El formato de patrón utilizado en el estudio atiende a la siguiente estructura:

Name-Alias-Abstract-Opening Story-Context-Problem-Description of the Problem and Forces-Therefore-Essence of the Solution-Resulting Context-Known Uses

La descripción de cada una de las secciones del formato del patrón se puede consultar en el Anexo C de este documento.

En el estudio se presentan 4 patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Let Them Play
2. Deep Dive
3. Pair to Share
4. Do Art

En su conjunto, los 4 patrones están fundamentalmente orientados al equipo y son de aplicación general a cualquier ámbito del ciclo de vida de un proyecto software. En el estudio no se relacionan gráficamente los patrones entre si.

Los autores proponen añadir estos 4 patrones al catalogo resultante del trabajo realizado por de Mary Lynn Manns y Linda Rising relativos a patrones para un cambio sin miedo[37]–[41]. El total de patrones de este catalogo, así como los patrones para garantizar un desarrollo sostenible [35], [36], se puede consultar en el Anexo B de este documento.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.10.

Criterios Evaluación EP10*	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	4+10	4 formulados + 10 nombrados (de MLynn y LRising)
Marco ágil referenciado	XP	Extreme Programming
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	2	Usa un formato reconocido tipo historia
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	EQU	Patrones orientados al equipo
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.10 - Ficha de evaluación EP10

Para los autores, las nuevas ideas que surgen en torno a la industria del software son dinámicas y constantes. Su adopción es complicada porque la mayoría de las veces la solución implica que las personas cambien su forma de pensar y de comportarse. A diferencia de las máquinas que reciben comandos bien definidos y ejecutan con precisión, los seres humanos somos independientes e impredecibles. No obstante, pese a esta imprevisibilidad, existen patrones de comportamiento que pueden ayudar a asumir el cambio y lograr los objetivos.

Uno de los patrones presentados, “Do Arts”, muestra la importancia de combinar las actividades artísticas con las actividades de desarrollo de software ya que existen elementos de conexión entre el lado puramente matemático y el creativo/artístico de la mente humana. En este punto, los autores concluyen que el desarrollo de software debe abordarse como una “actividad humana”, no exclusivamente como una actividad técnica o lógica.

4.2.10 EP11 (2011): The Art of Software Estimation Patterns Language

Autor: Dmitry Nikelshpur

Se trata de una propuesta de 9 patrones, basados en las mejores prácticas de estimación de software, para ayudar a entregar estimaciones rápidas, precisas, confiables y útiles.

El formato de patrón utilizado en este estudio responde a la siguiente estructura:
Name-Context-Problem-Forces-Solution-Discussion-Resulting Context
La descripción de cada una de las secciones del formato del patrón puede consultarse en el Anexo C de este documento.

En el estudio se presentan 9 patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Cone of Uncertainty
2. One Bite at a Time
3. Time is Relative
4. [Method of] Succesisive Estimations
5. Experts Estimate
6. Safe Estimating
7. Find Something to Count
8. Déjà Vu
9. All Aboard

En su conjunto los 9 patrones están orientados fundamentalmente al proceso y se aplican al ámbito de estimación del desarrollo software. En el estudio no se relacionan gráficamente los patrones entre sí.

Para el autor, el riesgo de sobreestimar o subestimar un proyecto tiene su impacto en la planificación y presupuesto de alto nivel. El problema ocurre cuando se piden estimaciones tempranas al principio del ciclo de vida del proyecto y se asumen como estimaciones lo suficientemente precisas para valorar los riesgos del proyecto, para negociar plazos, priorizar actividades, asignar recursos e incluso evaluar el desempeño de los desarrolladores. Por esta razón, el autor considera la estimación de software un “arte negro” [47] más que una ciencia.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.11.

Criterios Evaluación EP11	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	9	9 formulados
Marco ágil referenciado	AGILE	Agile en general
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	1	Si se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	PRC	Patrones orientados al proceso
Ámbito en ciclo software	EST	Foco en la estimación de software

Tabla 4.11 - Ficha de evaluación EP11

4.2.11 EP12 (2012): A pattern Language for Agile Software Estimation

Autores: Mirla RR Braga, Carla IM Bezerra, Jose maria s. Monteiro y Rossana MC Andrade

Se trata de una propuesta de lenguaje de patrones, con 8 patrones relacionados, para facilitar la estimación de software a los equipos que adoptan metodologías ágiles.

El formato de patrón utilizado en este estudio responde a la siguiente estructura:

Name-Context-Problem-Forces-Solution-Example-Resulting Context-Known Uses-Related Patterns

La descripción de cada una de las secciones del formato del patrón se puede consultar en el Anexo C de este documento.

En el estudio se presentan 8 patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Estimating size with Story Points
2. Estimating Effort Based on Tasks
3. Estimating Velocity
4. Estimating Duration Based on Team Velocity and Project Size
5. Financial Cost Estimation
6. Negotiating Estimates
7. Collecting Software Measures
8. Historical Data Adquisition

En su conjunto los 8 patrones están orientados fundamentalmente al proceso y se aplican al ámbito de estimación del desarrollo software. La Figura 4.5 muestra la relación entre los patrones de este estudio.

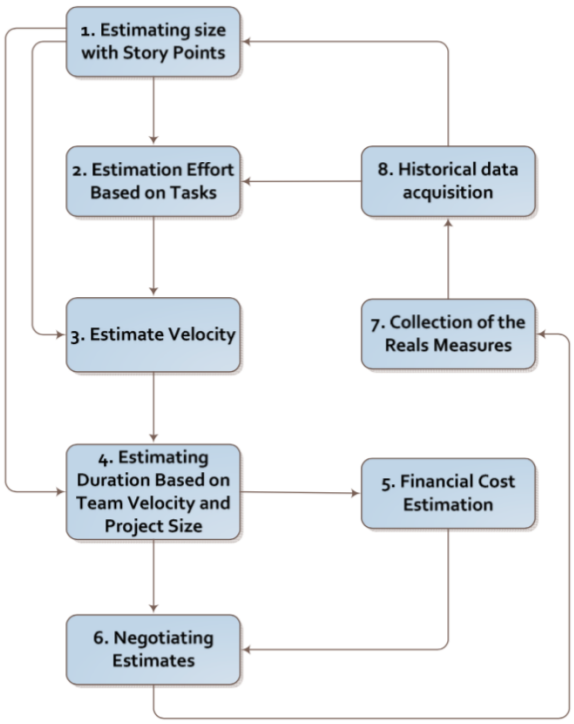

Figura 4.5 - Relación de patrones [EP12]

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.12.

Criterios Evaluación EP12	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	8	8 formulados
Marco ágil referenciado	AGILE	Agile en general
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	1	Se describe el formato del patrón en el estudio
Relación entre patrones	1	Si presenta relación visual entre patrones
Orientación proceso/equipo	PRC	Patrones orientados al proceso
Ámbito en ciclo software	EST	Foco en la estimación de software

Tabla 4.12 - Ficha de evaluación EP12

El estudio está basado en entrevistas a varias compañías que usan frameworks ágiles y fundamentado en bibliografía relevante sobre patrones software [7], [32], sobre agilidad y sobre estimaciones en entornos ágiles [46].

Para los autores, aunque las estimaciones son importantes para la gestión de los plazos y de los presupuestos de los proyectos, estás nunca serán perfectas. Por tanto, el objetivo de los equipos ágiles no es tanto acertar con las estimaciones como aprender de su experiencia en iteraciones cortas.

4.2.12 EP13 (2013): Is there an Agile Handover?

Autores: Christoph Johann Stettina y Egbert Kroon

Se trata de un estudio empírico sobre patrones de acción en equipos agiles para el traspaso de proyectos software.

El estudio **no identifica** ningún patrón de acción concreto, pero pone de manifiesto dos evidencias que facilitan el traspaso de un proyecto:

1. La comunicación directa y los artefactos de documentación útiles aceleran el proceso

2. Incluir al equipo de mantenimiento durante el desarrollo del proyecto facilita la transferencia de conocimiento.

En general, los patrones de acción expuestos en el estudio están orientados fundamentalmente al equipo y aplican a la entrega del proyecto, poniendo el foco en el impacto que la comunicación y la documentación tienen en el traspaso de conocimiento del proyecto.

Los autores consideran que el traspaso de un proyecto es un proceso de aprendizaje basado en la adopción del proyecto y la transferencia de conocimiento, más que en la entrega pura y dura de documentación. La creación de conocimiento está estrechamente relacionada a la acción humana y los artefactos de documentación que la soporta [48].

Derivado de a la dificultad de mantener la documentación de un proyecto, surge el problema de la percepción de la utilidad de la documentación, ya que si no es útil se considera un desperdicio a nivel económico y un elemento desmotivador para el equipo.

El estudio está basado la investigación realizada en 30 equipos de 7 proyectos de diferente tamaño y complejidad.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.13.

Criterios Evaluación EP13	Valor	Leyenda
Identificación del patrón	P	Patrones de acción humana
Número de patrones	0	No identifica patrones concretos
Marco ágil referenciado	AGILE	Agile en general
Explicación del patrón	0	No explica el patrón
Formulación del patrón	0	No usa formato
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	EQU	Patrones orientados al equipo
Ámbito en ciclo software	ENT	Foco en el traspaso del proyecto

Tabla 4.13 - Ficha de evaluación EP13

4.2.13 EP14 (2014): Teams that Finish Early Accelerate Faster

Autores: Jeff Sutherland, Neil Harrison y Joel Riddle

Se trata de una selección de 9 patrones Scrum que forman un lenguaje de patrones que facilitan la hiper-productividad de equipos ágiles garantizando la mejora continua de su rendimiento, su felicidad y la calidad de los entregables.

Aunque los patrones identificados en el estudio no se formulan como un patrón al uso[9], se explican con una descripción que recoge su esencia.

En el estudio se presentan 9 patrones claramente identificados por su nombre y descritos en detalle para facilitar su comprensión.

1. Stable Teams
2. Yesterday's Weather
3. Swarming One Piece Continuous Flow
4. Interrupt Pattern: Illigitimus Non Interruptus
5. Daily Clean Code
6. Emergency Procedure
7. Scrumming the Scrum
8. Happiness Metric
9. Team that Finish Early Accelerate Faster

[9] Formulación del patrón atendiendo a un formato reconocido (estándar o tipo historia).

Los dos primeros patrones ayudan al equipo a prepararse para un sprint exitoso. Los patrones del 3 al 6 ayudan al equipo a lidiar con los problemas disruptivos más comunes en un sprint. Los patrones 7 y 8 llevarán a un equipo al estado de alto rendimiento, haciendo que el patrón 9 emerja como un efecto secundario.

En su conjunto, los 9 patrones están orientados al proceso y al equipo y aplican en general a cualquier ámbito del ciclo de vida de un proyecto software. En el estudio no se relacionan gráficamente los patrones entre si.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.14.

Criterios Evaluación EP14*	**Valor**	**Leyenda**
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	9	9 descritos, no formulados
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	1	Explica el patrón con descripción
Formulación del patrón	0	No usa formato
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.14 - Ficha de evaluación EP14

Los autores consideran que el uso combianado de estos patrones aumenta la capacidad del equipo para terminar el Sprint antes, lo que permite aumentar la velocidad de cara el próximo Sprint (en cuanto a abordar más elementos del backlog) y también permite aumentar la métrica de la felicidad del equipo porque se sienten seguros de su capacidad para completar el Sprint. Esta situación arranca una espiral de mejora continua que lleva a la hiper-productividad del equipo.

El estudio incluye un ejemplo de implementación de los patrones en una empresa durante 2010 con datos concretos de métricas que justifican su valor.

4.2.14 EP15 (2015): Animating Organizational Patterns

Autores: Tomás Frt'ala y Valentino Vranic

Se trata de una propuesta para adoptar un enfoque ágil, adquiriendo experiencia en patrones organizacionales a través del juego.
El estudio **no identifica** ningún patrón en concreto, aunque sí pone de manifiesto la importancia de los patrones organizacionales para una adopción ágil y analiza las posibilidades de ganar experiencia, sin fallar en un entorno real, a través de juegos con el fin de llegar a dominar los patrones organizacionales.

En general los patrones organizacionales están orientados al proceso y al equipo y aplican en general a cualquier ámbito del ciclo de vida de un proyecto software.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.15.

Criterios Evaluación EP15	Valor	Leyenda
Identificación del patrón	P	Patrones organizacionales
Número de patrones	0	No identifica patrones concretos
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	0	No explica el patrón
Formulación del patrón	0	No usa formato
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al equipo y al proceso
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.15 - Ficha de evaluación EP15

Para los autores las dificultades para adoptar un enfoque ágil son recurrentes. Estas dificultades suelen venir dadas, en parte, porque la naturaleza humana requiere experiencia y a los profesionales no se les da la oportunidad de practicar lo que es un enfoque ágil una vez que han sido capacitados para ello.

Su propuesta gira en torno a los patrones organizacionales como un juego de aventuras en el que la importancia del escenario y el poder de la imaginación “juegan” un papel importante.

4.2.15 EP16 (2015): Patterns for Integrating Agile Development and UCD

Autores: Dina Salah, Richard Paige y Paul Cairns

Se trata de una propuesta de 5 patrones que facilitan la integración de procesos de diseño centrado en el usuario con procesos de desarrollo ágil.

El formato de patrón utilizado en este estudio responde a la siguiente estructura:
Name-Context-Problem-Forces-Solution
El estudio no describe explícitamente cada una de las secciones del formato del patrón planteado.

En el estudio se presentan 5 patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Less is More!
2. Usability Testing Sessions Alongside Agile Development Tests
3. Do it RITE! (Rapid Iterative Testing and Evaluation)
4. Developers as UCD Practitioners
5. Websites as Lightweight Documentation

En su conjunto, los 5 patrones están orientados tanto al proceso como al equipo y aunque son de aplicación general a cualquier ámbito del ciclo de vida de un proyecto, se focalizan en el diseño y desarrollo software centrado en el usuario.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.16.

Criterios Evaluación EP16	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	5	5 formulados
Marco ágil referenciado	XP	Extreme Programming
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	D&D	Foco en el diseño y desarrollo software

Tabla 4.16 - Ficha de evaluación EP16

Los patrones presentados por los autores para la integración de procesos de desarrollo ágil y diseño centrado en el usuario surgieron a partir de dos estudios: una revisión sistemática de la literatura sobre los desafíos, estrategias y factores de éxito de la integración, y un estudio basado en entrevistas sobre los intentos de la integración en la industria.

La Figura 4.6 muestra uno de los resultados de este estudio, las semejanzas y diferencias entre el proceso Agile y el Diseño Centrado en el Usuario.

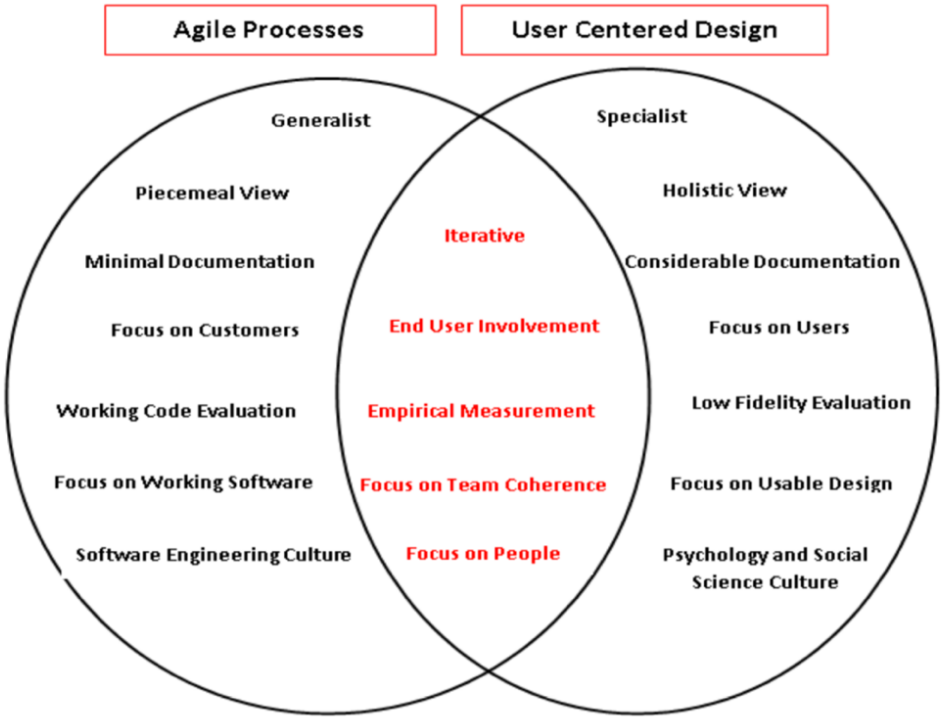

Figura 4.6 - Puntos en común y diferencias entre procesos ágiles y DCU [EP16]

4.2.16 EP17 (2015): Distributed Agile Patterns for Offshore Software

Autores: Maryam Kausar y Adil Al-Yasiri

Se trata de una recopilación de buenas prácticas ágiles, en formato patrón, para abordar proyectos con equipos distribuidos que trabajan en entornos ágiles.

El formato de patrón usado en el estudio responde a la siguiente estructura:

Name-Intent-Also Known-Category-Motivation-Applicability-Participants-Collaboration-Consequences-Known Uses-Related Patterns

El estudio no describe explícitamente cada una de las secciones del formato del patrón planteado.

En el estudio se presentan 15 patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Scrum of Scrum Pattern
2. Local Stand-up Meetings Pattern
3. Follow the Sun Pattern
4. Onshore Review Meeting
5. Collective Project Planning
6. Project Charter Pattern
7. Collaborative Planning Poker
8. Global Scrum Board
9. Local Sprint Planning
10. Local Pair Programing
11. Central Code Repository
12. Asynchronous Retrospective Meetings
13. Asynchronous Information Transfer
14. Synchronous Communication
15. Visit Onshore-Offshore Teams

En su conjunto, los 15 patrones están orientados tanto al proceso como al equipo y son de aplicación general a cualquier ámbito del ciclo de vida de un proyecto software. En el estudio no se relacionan gráficamente los anti-patrones entre si.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.17.

Criterios Evaluación EP17	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	2+13	2 formulados + 13 nombrados
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.17 - Ficha de evaluación EP17

Para los autores, los desafíos a los que se enfrentan los equipos distribuidos que desarrollan software surgen en torno a cuestiones relacionadas con la confianza, con aspectos socio-culturales, con la comunicación y la coordinación de equipos remotos, y con el traspaso de conocimiento entre proyectos.
El resultado de la investigación son los 15 patrones recopilados (clasificados por los autores como patrones ágiles distribuidos) para resolver los problemas repetitivos que surgen en los proyectos de software abordados en entornos ágiles con equipos remotos.

El estudio esta basado en el estudio de más de 200 casos reflejados en la literatura y las entrevistas realizadas a profesionales que participan en equipos distribuidos.

4.2.17 EP18 (2016): Agile base Patterns in the Agile Canon

Autor: Daniel R Greening

Se trata de una propuesta de 5 patrones básicos Agile que garantizan la agilidad en las organizaciones más allá del desarrollo de software e independientemente del framework ágil que se utilice.

El formato de patrón utilizado en el estudio responde a la siguiente estructura:
Name-Context-Problem-Forces-Solution (Therefore)-Examples-ResultingContext
La descripción de cada una de las secciones del formato del patrón se puede consultar en el Anexo C de este documento.

En el estudio se presentan 5 patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Measure Economic Progress
2. Proactiely Experiment
3. Limit Work-in-process
4. Collective Responsibiity
5. Solve Systemic Problems

Los tres primeros patrones son necesarios y suficientes para ser ágiles. El cuarto resuelve la resistencia ágil. El quinto favorece el extrapolar la agilidad más allá de un individuo o equipo de software concreto.

En su conjunto, los 5 patrones básicos Agile están orientados al proceso y al equipo y aplican en general a cualquier ámbito del ciclo de vida de software. La Figura 4.7 muestra como el marco Scrum relaciona estos patrones.

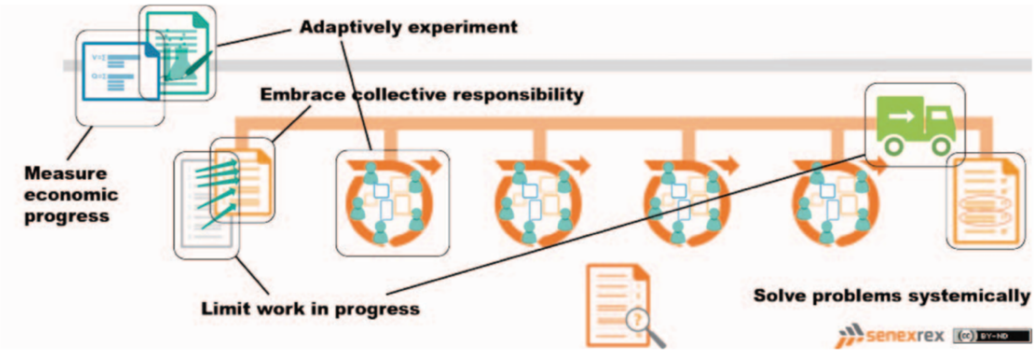

Figura 4.7 - Los eventos Scrum como patrones básicos Agile [EP18]

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.18.

Criterios Evaluación EP18	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	3+2	5 formulados (3 básicos + 2 de extensión)
Marco ágil referenciado	AGILE	Agile en general. Ejemplifica con Scrum
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	1	Usa formato estándar reconocido
Descripción del formato	1	Se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.18 - Ficha de evaluación EP18

Para el autor, los frameworks ágiles ayudan a detectar rápidamente los cambios de entorno, a adaptar las capacidades y a crear soluciones. Los frameworks ágiles se adaptan al caos para obtener una ventaja competitiva.

Los patrones básicos de Agile son prescriptivos y surgen de agrupar roles, artefactos y eventos de frameworks ágiles reconocidos. Por el contrario, el manifiesto ágil es descriptivo y surge del consenso de los líderes ágiles.

4.2.18 EP19 (2016): Exploring ScrumBut – Scrum anti-patterns

Autores: Veli-Pekka Eloranta, Kai Koskimies y Tommi Mikkonen

Se trata de un estudio empírico que presenta 14 anti-patrones Scrum basados en las desviaciones asumidas por las empresas en la práctica de Scrum.

El formato utilizado en el estudio responde a la siguiente estructura:

Name-Scrum Recommendation-Context-Solution-Consequences-Exceptions-Company Recommendations

La descripción de cada una de las secciones del formato del anti-patrón se puede consultar en el Anexo C de este documento.

En el estudio se presentan 14 anti-patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Big Requirements Documentation
2. Customer Product Owner
3. Producto Owner without Authority
4. Long or Non-existent Feedback Loops
5. Unordered Product Backlog
6. Work Estimates Given to Teams
7. Hour in Progress Monitoring
8. Semi-functional Teams
9. Customer Caused Disruption
10. Business as Usal (No Sprint Retrospective)
11. Invisible Progress
12. Varying Sprint Length
13. Too Long Sprint
14. Testing in Next Sprint

En su conjunto, los 14 anti-patrones están orientados tanto al proceso como al equipo, y aplican en general a cualquier ámbito del ciclo de vida de software. La Figura 4.8 muestra la relación de estos patrones entre si mediante conceptos de Scrum (representados gráficamente en rectángulos).

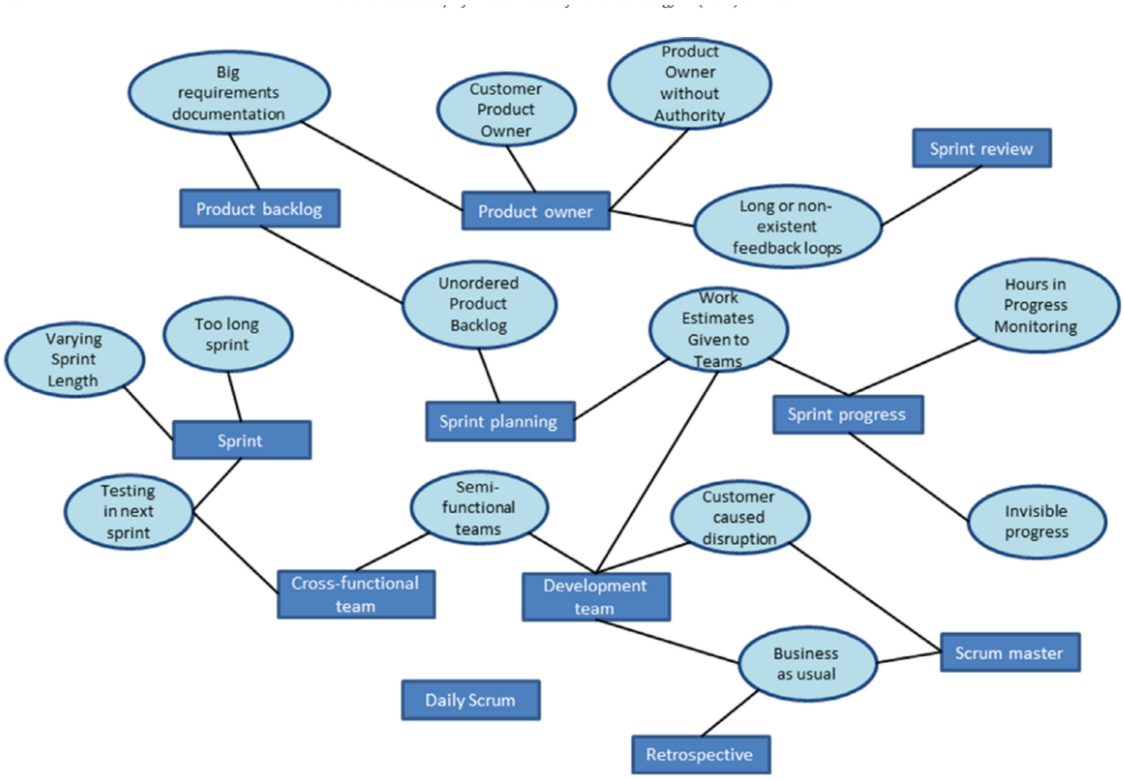

Figura 4.8 - Relación entre anti-patrones Scrum y algunos de sus conceptos [EP19]

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.19.

Criterios Evaluación EP19	Valor	Leyenda
Identificación del patrón	A	Anti-patrón identificado con nombre
Número de patrones	14	14 formulados
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	3	Usa formato adaptado de uno reconocido
Descripción del formato	1	Se describe el formato del patrón en el estudio
Relación entre patrones	1	Si presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	GEN	Patrones de aplicación general en el ciclo software

Tabla 4.19 - Ficha de evaluación EP19

El estudio recopila un buen conjunto de prácticas Scrum que a las empresas les cuesta seguir por diferentes motivos. Los resultados indican que las empresas más pequeñas siguen las recomendaciones Scrum evitando los anti-patrones y que las empresas que tienen más experiencia en Scrum tienden a usar más anti-patrones. Pero hay dos anti-patrones que se repiten independientemente del tamaño y experiencia de la empresa: dejar las pruebas para el próximo Sprint y permitir que el cliente interrumpa el ritmo del sprint.

4.2.19 EP20 (2017): Patterns of Collaboration Driven by Requirements

Autores: Irum Inayat, Sabrina marczak, Siti Salwah Salim y Daniela Damian

Se trata de una propuesta de 7 patrones de colaboración dirigida por requisitos en equipos ágiles de desarrollo software.

Aunque los patrones identificados en el estudio no se formulan como un patrón al uso[10], se explican con una descripción que recoge su esencia.

En el estudio se presentan 7 patrones claramente identificados por su nombre y descritos en detalle para facilitar su comprensión.

1. Team members involvement was different than assigned
2. Agile teams are sefl-contained
3. Project managers are key players in communication
4. Distance does not (seem to) matter
5. Bugs discussion, communication of changes, and user story clarification as the most discussed topics
6. Familiarity from past projecs facilitaes awareness
7. Communication is still an important source of awarenesss

En su conjunto, los 7 patrones están orientados fundamentalmente al equipo y están focalizados a la gestión de requisitos. En el estudio no se relacionan gráficamente los patrones entre si.

Para los autores, la comunicación juega un papel importante en el desarrollo de software y la gestión de requisitos es una actividad compleja que implica colaboración. Los resultados del estudio indican que la distancia no impacta en la gestión del conocimiento, que la experiencia en proyectos similares facilita la toma de conciencia y que la comunicación es un factor clave en la toma de conciencia.

[10] Formulación del patrón atendiendo a un formato reconocido (estándar o tipo historia).

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.20.

Criterios Evaluación EP20	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	7	7 descritos, no formulados
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	1	Explica el patrón con descripción
Formulación del patrón	0	No usa formato
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	EQU	Patrones orientados al equipo
Ámbito en ciclo software	REQ	Foco en la ingeniería de requisitos

Tabla 4.20 - Ficha de evaluación EP20

El estudio se realizó en 4 empresas multinacionales ubicadas en distintas partes del mundo y que utilizan Scrum para desarrollar sus productos.

4.2.20 EP21 (2017): Anti-patterns in Agile Adoption: A Grounded Theory

Autores: Peter J. Carew y David Glynn

Se trata de un estudio empírico realizado para recopilar los anti-patrones detectados en la adopción ágil por parte de una empresa tradicionalmente no ágil.

El estudio **no identifica** ningún patrón de adaptación en concreto, aunque sí pone de manifiesto dos evidencias en torno a los patrones de adaptación:

1. Las peticiones recurrentes no pertenecientes al Sprint, interrumpen significativamente el trabajo priorizado, dificulta conseguir los objetivos del Sprint y afecta al rendimiento del equipo Scrum.

2. Todas las partes interesadas en el proyecto con capacidad de decisión, tienden a la gestión tradicional no ágil que no permite la flexibilidad de auto-organización necesarias en un enfoque ágil.

En general los patrones de adopción están orientados tanto al proceso como al equipo y aplican a cualquier ámbito del ciclo de vida de un proyecto software poniendo el foco en la adopción del enfoque ágil.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.21.

Criterios Evaluación EP21	Valor	Leyenda
Identificación del patrón	A	Anti-patrones de la adopción ágil
Número de patrones	0	No identifica patrones concretos
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	0	No explica el patrón
Formulación del patrón	0	No usa formato
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	0	No presenta relación visual entre patrones
Orientación proceso/equipo	2	Patrones orientados al proceso y al equipo
Ámbito en ciclo software	GEN	Foco en la adopción de un enfoque ágil

Tabla 4.21 - Ficha de evaluación EP21

Para los autores, la cultura ágil tiene que instalarse dentro de las organizaciones tradicionales no ágiles, adoptando un enfoque sociotécnico humano y cultural. Es necesario que los estudios ágiles no se centren exclusivamente en artefactos técnicos y entregables.

4.2.21 EP22 (2018): A Program Backlog Story with Patterns*

Autores: Lise B. Hvatum y Rebecca Wirfs-Brock

Se trata de una propuesta de 3 patrones para gestionar el backlog de programa y que complementan la colección mágica de patrones de backlog.

El formato de patrón usado en el estudio responde a la siguiente estructura:

Name-Context-Problem-Therefore

El estudio no describe explícitamente cada una de las secciones del formato del patrón planteado.

En el estudio se presentan 3 patrones claramente identificados por su nombre y explicados en detalle atendiendo al formato indicado.

1. Unified Product Backlog
2. Linked Product Backogs
3. Pragamatic Product Backlogs

Las autoras también identifican en el estudo otros 3 patrones de backlog que pretenden explicar en detalle en futuros trabajos.

1. Remodel
2. Rules
3. Definitions

En general, los patrones de backlog de programa están orientados fundamentalmente al proceso y son de aplicación directa a la gestión de requisitos del ciclo de vida de un proyecto software. La Figura 4.9 muestra, a modo de árbol de decisión, la relación entre los patrones de backlog de programa.

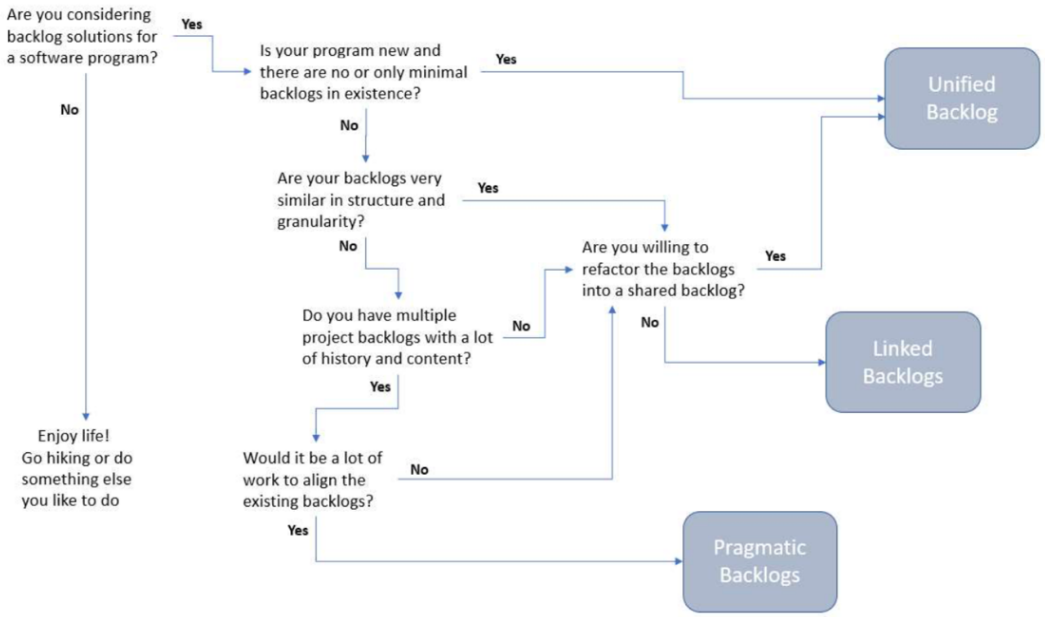

Figura 4.9 - Árbol de decisión patrones product backlog de programa [EP22]

Para las autoras, los proyectos que forman parte de un programa pueden tener dependencias entre ellos y un despliegue compartido. Por ello, aunque sus equipos puedan trabajar de manera independiente, para producir un producto es necesario coordinara su trabajo.

Los criterios utilizados para la evaluación del estudio se resumen, en formato ficha, en la Tabla 4.22.

Criterios Evaluación EP22*	Valor	Leyenda
Identificación del patrón	P	Patrón identificado con nombre
Número de patrones	3+15	3 formulados + 15 nombrados
Marco ágil referenciado	SCRUM	Scrum
Explicación del patrón	2	Explica el patrón según formato
Formulación del patrón	3	Usa formato adaptado de uno reconocido
Descripción del formato	0	No se describe el formato del patrón en el estudio
Relación entre patrones	1	Si presenta relación visual entre patrones
Orientación proceso/equipo	PRC	Patrones orientados al proceso
Ámbito en ciclo software	REQ	Foco en la ingeniería de requisitos

Tabla 4.22 - Ficha de evaluación EP22

Los 3 patrones descritos en este estudio suponen estrategias alternativas para estructurar el backlog del programa y se presentan a través de la historia de un analista de negocios en la que su hipotético programa se va moviendo a través de las diferentes estrategias de gestión del backlog. Adicionalmente, se enuncian en formato corto los 12 patrones de la colección recogidos en anteriores estudios de las autoras [51], [52]. El total de los 18 patrones incluidos en el estudio EP22 puede consultarse en el Anexo B de este documento y cuya relación se puede observar en la Figura 4.10.

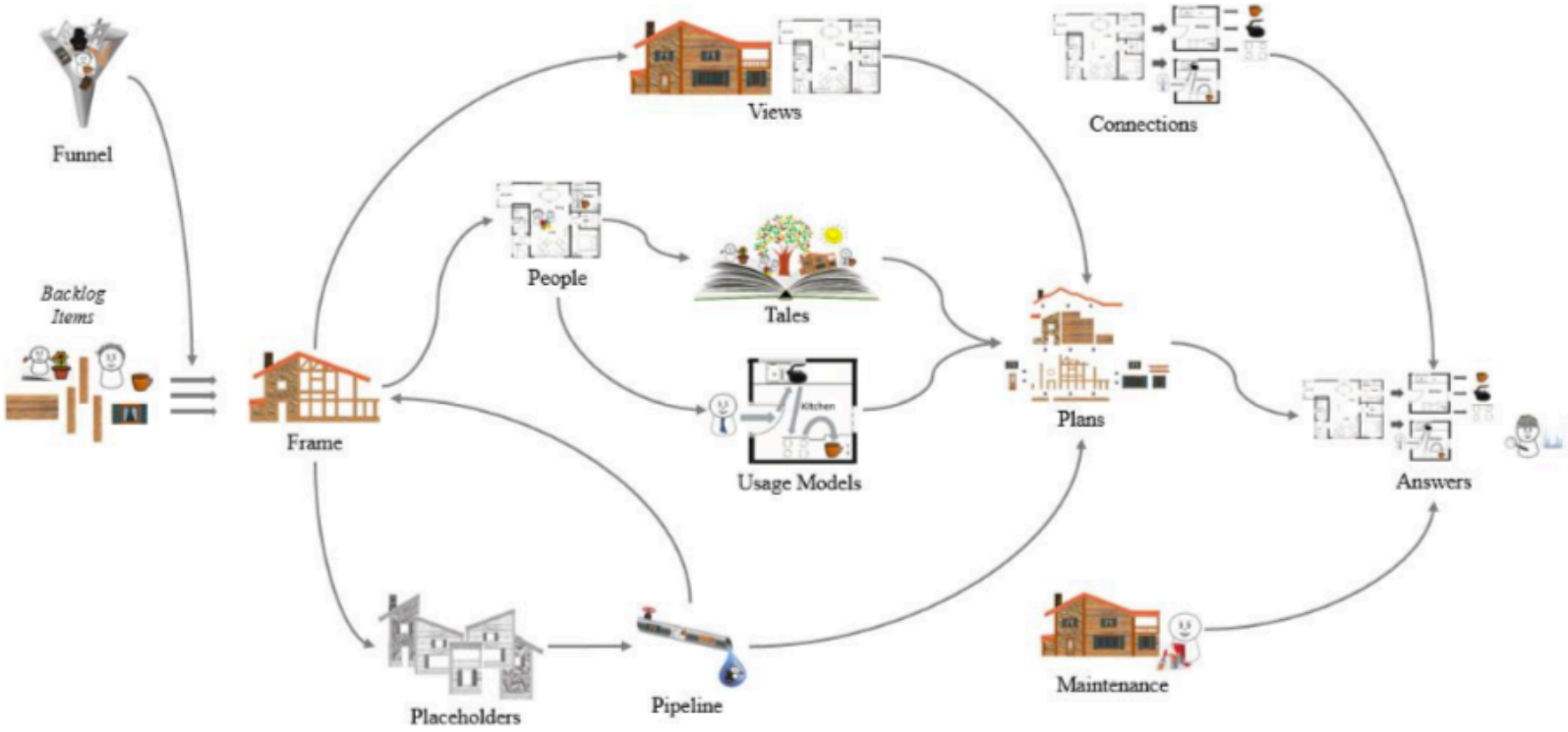

Figura 4.10- The Magic Backlog Patterns [52]

5 Análisis de los resultados

Una vez presentado el estado del arte, en este capitulo se analizan los resultados y se presentan las principales conclusiones derivadas de la evaluación de los estudios de acuerdo con los criterios indicados en el apartado 4.1. Partiendo de este análisis, se valora como ofrecer una mejora al estado del arte y se propone una línea de investigación para su desarrollo.

5.1 Extracción y síntesis de los resultados

En el capítulo anterior se han analizado todos los estudios primarios seleccionados. El análisis se ha realizado atendiendo a los criterios de evaluación presentados en el apartado 4.1. Esta evaluación no ha presentado datos cuantitativos, algo que dificulta la comparativa de las propuestas analizadas.

A continuación, se ofrece una escala de valores para cada uno de los criterios con los que se han evaluado los estudios. Estás escalas aportan datos cuantitativos que facilitan la comparación de las diferentes aproximaciones de los estudios. La Tabla 5.1 muestra de manera resumida la evaluación de cada uno de los estudios, según estás escalas.

5.1.1 Escalas de valores para los criterios de evaluación

Atendiendo al análisis de los criterios de evaluación que se ha realizado sobre cada uno de los estudios, se determinan los valores para cada uno de los criterios:

Criterio de Evaluación	Valor
1. Identificación del patrón	P - Patrón con nombre A - Anti-patrón con nombre
2. Número de patrones	0 - No identifica ningún patrón concreto n - Número de patrones concretos identificados
3. Marco ágil referenciado	XP - Extreme Programming SCRUM - Marco Scrum AGILE - Agile en general
4. Explicación del patrón	0 - No describe patrón (solo nombre) 1 - Si describe patrón (con descripción) 2 - Si describe patrón (según formato de patrón)
5. Formulación del patrón	0 - Sin formato 1 - Con formato estándar reconocido 2 - Con formato tipo historia 3 - Con formato adaptado
6. Descripción del formato	0 - No explica las secciones del formato 1 - Si explica las secciones del formato
7. Relación entre patrones	0 - No muestra relación gráfica entre patrones 1 - Si muestra relación gráfica entre patrones
8. Orientación proceso/equipo	PRC - Orientado al proceso EQU - Orientado al equipo 2 - Orientado a ambos
9. Ámbito en ciclo software	GEN - General DCU - Diseño Centrado en Usuario REQ - Requisitos EST - Estimación D&D - Diseño y Desarrollo DOC - Documentación ENT - Entrega

Tabla 5.1 - Valores para los criterios de evaluación

5.1.2 Resumen de criterios analizados en cada estudio primario

EP	Identificación de patrón	Número de patrones	Marco ágil referenciado	Explicación del patrón	Formulación del patrón	Descripción del formato	Relación entre patrones	Proceso y/o Equipo	Ciclo de Software
EP01	A	3+7[1]	XP	2	3	1	0	EQU	GEN
EP02	P	4	XP	2	1	0	0	2	ENT
EP03*	P	10+50[29]	XP	2	1	1	0	2	GEN
EP04	P	9+9[3]	XP	2	1	0	0	2	GEN
EP05	P	1	AGILE	2	3	1	0	PRC	GEN
EP06	P	1	SCRUM	2	3	1	0	PRC	GEN
EP07	P	5	SCRUM	2	1	1	1	PRC	DOC
EP08	P	0	XP	0	0	0	0	EQU	REQ
EP09	P	6	AGILE	2	1	0	1	PRC	ENT
EP10*	P	4+10*[4]	XP	2	2	0	0	EQU	GEN
EP11	P	9	AGILE	2	1	1	0	PRC	EST
EP12	P	8	AGILE	2	1	1	1	PRC	EST
EP13	P	0	AGILE	0	0	0	0	EQU	ENT
EP14	P	9	SCRUM	1	0	0	0	2	GEN
EP15	P	0	SCRUM	0	0	0	0	2	GEN
EP16	P	5	XP	2	1	0	0	2	D&D
EP17	P	2+13[5]	SCRUM	2	1	0	0	2	GEN
EP18	P	3+2[6]	AGILE	2	1	1	0	2	GEN
EP19	A	14	SCRUM	2	3	1	1	2	GEN
EP20	P	7	SCRUM	1	0	0	0	EQU	REQ
EP21	A	0	SCRUM	0	0	0	0	2	GEN
EP22*	P	3+15[7]	SCRUM	2	3	0	1	PRC	REQ

Tabla 5.2 – Resumen de criterios analizados por estudio primario

(1) EP01: En total, el estudio refiere 10 anti-patrones, de los cuales 3 están detallados según el formato propuesto y 7 anti-patrones más están identificados, pero solo nombrados, para detallarlos en trabajos futuros.

(2) EP03: En total, el estudio refiere 60 patrones, de los cuales 10 están detallados según el formato propuesto y 50 patrones más están recopilados, pero solo enunciados, de estudios anteriores del autor.

(3) EP04: En total, el estudio refiere 18 patrones, de los cuales 9 están focalizados en los distintos roles del cliente y los otros 9 patrones en las prácticas del equipo cliente. Todos los patrones están detallados según el formato propuesto en el estudio.

(4) EP10: En total, el estudio refiere 14 patrones, de los cuales 4 son propios del estudio y están detallados según el formato propuesto y 10 patrones más seleccionados, pero solo enunciados, de la colección original de Mary Lynn Mann y Linda Rising [37]. No obstante, en la relación de patrones recogida en el Anexo B, aparecen 76 patrones asociados al estudio EP10: los 4 de este estudio, más las dos colecciones completas de patrones publicadas por Linda Rising: la relativa al cambio sin miedo [37]–[41] (48 patrones iniciales + 17 en ampliaciones sucesivas); y la relativa al desarrollo sostenible [35], [36] (7 patrones en dos entregas).

(5) EP17: En total, el estudio refiere 15 patrones, de los cuales tan solo 2 están detallados según el formato propuesto y los otros 13 patrones solo están enunciados.

(6) EP18: En total, el estudio refiere 5 patrones todos ellos detallados según el formato propuesto. De los 5 patrones, 3 son considerados por el autor básicos para ser ágiles y los otros 2 añaden resiliencia y expansión a la agilidad.

(7) EP22: En total, el estudio refiere 18 patrones, de los cuales 3 están detallados según el formato propuesto y 15 patrones más están recopilados, pero solo enunciados, de estudios anteriores de las autoras.

El total de los **262 patrones identificados** en la elaboración de este trabajo fin de máster, se recogen en el Anexo B de este documento asociados al estudio de referencia.

5.1.3 Resumen de formatos de patrón identificados en cada estudio primario

EP	Nombre corto	Formato de patrón
EP01	AntiPractices: AntiPatterns for XP practices	**Name**-Background-Sympton-Cause-Ideal-Prescription [(1)(2)]
EP02	Value based extreme programming	**Nam**e-Category-Problem-Context-Solution
EP03	Patterns for agile development practice part 3 (version4)	**Name**-Context-Problem-Forces-Therefore (Solution)-Commentary-Consequences [(1)]
EP04	Programmers are from Mars, customers are from Venus …	**Name**-Problem-Forces-Solution-Forces Resolved
EP05	A process pattern language for agile methods	**Name**-Context-Problem-Solution-Consequence-Related Pattern [(1)]
EP06	Agile Learning & Teaching with Wikis: Building a Pattern	**Name**-Intent-Motivation (Forces)-Applicability-Structure-Participants-Collaboration-Consequences-Implementacion-Known Uses-Related [(1)(3)]
EP07	How Much is Just Enough?: Some Documentation Patterns …	**Name**-Context-Example-Problem-Forces-Solution-Consequences-Example Resolved- Known Uses- Related Practices [(1)]
EP08	Communication patterns of agile requirements engineering	El estudio no identifica ningún patrón en concreto
EP09	A Pattern Language for Release and Deployment Management	**Name**-Context-Problem-Forces-Solution-Resulting Context-Rationale
EP10	Extending patterns for fearless change	**Name**-Alias-Abstract-Opening Story-Context-Problem-Description of the Problem and Forces-Therefore-Essence of the Solution-Resulting Context-Known Uses [(1)(4)]
EP11	The Art of Software Estimation Pattern Language	**Name**-Context-Problem-Forces-Solution-Discussion-Resulting Context [(1)]
EP12	A pattern language for agile software estimation	**Name**-Context-Problem-Forces-Solution-Example-Resulting Context-Known Uses-Related Patterns [(1)]
EP13	Is there an agile handover? … documentation and handover …	El estudio no identifica ningún patrón en concreto
EP14	Teams that finish early accelerate faster … high performing …	El patrón no tiene formato concreto. Se explica con una **descripción**
EP15	Animating Organizational Patterns	El estudio no identifica ningún patrón en concreto
EP16	Patterns for integrating agile development processes and UCD	**Name**-Context-Problem-Forces-Solution
EP17	Distributed Agile Patterns for Offshore Software Development	**Name**-Intent-Also Known-Category-Motivation-Applicability-Participants-Collaboration-Consequences-Known Uses-Related Patterns [(1)(3)]
EP18	Agile base patterns in the agile canon	**Name**-Context-Problem-Forces-Solution (Therefore)-Examples-Resulting Context [(1)(5)]

EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns	**Name**-Scrum Recommendation-Context-Solution-Consequences-Exceptions-Company Recommendations [(1)(2)]
EP20	Patterns of collaboration driven by requirements …	El patrón no tiene formato concreto. Se explica con una **descripción**
EP21	Anti-patterns in Agile Adoption: A Grounded Theory …	El estudio no identifica ningún patrón en concreto
EP22	A program backlog story with patterns … magic ... collection	**Name**-Context-Problem-Therefore [(4)]

Tabla 5.3 - Resumen de formatos de patrón por estudio primario

(1) El autor describe explícitamente cada una de las secciones del formato del patrón utilizado en el estudio. En el Anexo C, se puede consultar el detalle de las secciones de cada formato de patrón propuesto.

(2) EP01 y EP19: El autor indica explícitamente que el formato propuesto es una adaptación de Brown [33].

(3) EP06 y EP17: El autor indica explícitamente que el formato propuesto es una adaptación de Gamma [6]. En el caso concreto del estudio EP06, el autor indica que no se incluyen las secciones “tittle and classification”, “also known as” y “code examples”.

(4) EP10 y EP22: Aunque el autor no lo explicita, el formato propuesto esta inspirado en el de Rising [37], que a su vez está basado en el de Alexander [22].

(5) EP18: El autor indica explícitamente que el formato propuesto es una adaptación de Alexander [22].

En el Anexo C, se pueden consultar los formatos de patrón que describen explícitamente sus secciones en el estudio.

5.2 Discusión

En este apartado se presentan las conclusiones obtenidas tras realizar el análisis del resumen de la evaluación de los criterios de los estudios primarios (Tabla 5.2). Un análisis que se complementa con lo datos del formato de patrón propuesto en cada estudio (Tabla 5.3) y la fecha de publicación de cada estudio (Tabla 3.5).

Como se puede observar en la Tabla 5.2, las principales diferencias entre los estudios analizados residen en los siguientes aspectos claves: la identificación del patrón y la calidad de su formulación, la cantidad de patrones que recogen, el entorno ágil en el que se presentan, su orientación al proceso/equipo y su aplicación en el ciclo de vida software.

En cuanto a la **identificación y formato del patrón**, se observa:

- De los 22 estudios, se han detectado 4 estudios que no identifican ningún patrón en concreto, aunque si investigan sobre patrones en general relacionados con la comunicación [EP08], el traspaso de conocimiento [EP13], modelos organizacionales [EP15] y adopción de frameworks ágiles [EP21].

- De los 18 estudios que identifican algún patrón:

    - 2 explican los patrones con una descripción larga [EP14] y [EP20], sin un formato de patrón al uso.
    - 16 describen los patrones con un formato reconocido [22], [6], [33], [37], de los cuales 10 describen en detalle cada una de las secciones del formato de patrón utilizado [EP01], [EP03], [EP05], [EP06], [EP07], [EP10], [EP11], [EP12], [EP18] y [EP19]. El detalle de las secciones de los formatos de cada uno de estos estudios se puede consultar en el Anexo C de este documento.

- De los 18 estudios que identifican algún patrón, 2 identifican un único patrón cada uno, en forma de patrón de procesos: [EP05], para procesos basados en frameworks ágiles en general (típicamente XP y Scrum) y [EP06] para procesos ágiles de formación, en principio en conceptos ágiles pero extrapolable a cualquier formación cuyo objetivo sea, además del aprendizaje de un área de conocimiento específica, el desarrollo de aptitudes para el trabajo en equipo de forma colaborativa, iterativa e incremental.

- De los 22 estudios, 3 refieren colecciones de patrones: sobre desarrollo ágil [EP01], sobre el cambio sin miedo [EP10] y sobre la gestión del product backlog [EP22].

- De los 22 estudios, 5 presentan relación gráfica entre los patrones que proponen, haciendo más fácil su comprensión como lenguaje de patrones [EP07], [EP09], [EP12], [EP19] y [EP22].

En cuanto al **número de patrones identificados**, se observa:

- En los 22 estudios, se han identificado 262 patrones aplicables a equipos que desempeñan alguna actividad del ciclo de vida software en entornos ágiles. La relación completa de patrones se puede consultar en el Anexo B de este documento.

- El 99% de los patrones se recopilan a partir del 73% de los estudios. Es decir, 260 de los 262 patrones se refieren en 16 de los 22 estudios analizados.

- El 52% de los patrones se concentran en el 9% de los estudios. Es decir, 136 de los 262 patrones se refieren en 2 de los 22 estudios analizados. Cada uno de estos estudios hace referencia a más de 60 patrones [EP03] y [EP10].

- El 59% de los patrones se concentran en el 14% de los estudios, que se corresponden con lo estudios catalogados como colecciones en este trabajo (marcados con "*" en su numeración o en su nombre). Es decir, 154 de los 262 patrones se refieren en 3 de los 22 estudios analizados [EP03], [EP10] y [EP22].

- El 9% de los patrones identificados son anti-patrones. Es decir, 24 de los 262 patrones son anti-patrones [EP01] y [EP19].

- El 42% de los anti-patrones identificados están relacionado con XP y el 58% con Scrum. Es decir, 10 de los 24 son anti-patrones XP [EP01] y 14 de los 24 son anti-patrones Scrum [EP19].

En cuanto al **marco ágil referenciado**, se observa:

- El 41% de los estudios, es decir, 9 estudios hacen referencia fundamentalmente a patrones Scrum.

- El 32% de los estudios, es decir, 7 estudios hacen referencia fundamentalmente a patrones XP.

- El 27% de los estudios, es decir, 6 estudios hacen referencia a entornos ágiles en general, independientemente del framework ágil utilizado.

- Los 4 primeros estudios seleccionados están publicados en los años 2004 y 2006 y sus patrones están basados en prácticas XP. Por el contrario, los 4 últimos están publicados entre 2015 y 2018 y están basados en el marco Scrum.

- El primer estudio publicado en torno al marco Scrum fue en 2008 [EP06].

- El último estudio publicado en torno a las prácticas XP fue en 2015 [EP16].

- La publicación de patrones en entornos ágiles concretos (XP o Scrum) coincide con el uso de frameworks ágiles en la industria del software [21].

En cuanto a la **orientación a proceso/equipo**, se observa:

- El 45% de los estudios, es decir, 10 estudios, orientan sus patrones tanto a proceso como a equipo.

- El 32% de los estudios, es decir, 7 estudios orientan sus patrones fundamentalmente al proceso.

- El 23% de los estudios, es decir, 5 estudios orientan sus patrones fundamentalmente al equipo. De los 5 estudios, en 3 sus patrones hacen referencia a equipos XP, en 1 a equipos Scrum y en 1 a equipos ágiles en general.

En cuanto al **ámbito de aplicación en el ciclo de vida software**, se observa:

- El 55% de los estudios, es decir, 12 estudios proponen patrones que aplican en general a un enfoque ágil del ciclo de vida software.

- Del 45% restante, es decir, de los otros 10 estudios:

    - 3 estudios proponen patrones que aplican a la ingeniería de requisitos.
    - 2 estudios proponen patrones que aplican a la estimación de software.
    - 1 estudio propone patrones que aplican al diseño y desarrollo de software, focalizando en la integración de un proceso de diseño centrado en usuario con los procesos ágiles.
    - 3 estudios proponen patrones que aplican a la entrega de software.
    - 1 estudio propone patrones que aplican específicamente a la documentación de los proyectos.

5.3 Tendencia de los estudios

En este apartado se analizan, desde el punto de vista estadístico, algunos aspectos relativos a la publicación de los estudios seleccionados. Si bien, se podría incluir el idioma en el que están escritos los estudios, en este caso, todos los estudios primarios se encuentran en inglés.

5.3.1 Año de publicación

El concepto de patrón lleva ligado a la ingeniería del software desde finales del siglo pasado [24] y es un concepto ampliamente reconocido por los profesionales dedicados al desarrollo de software [6]. Por otro lado, los frameworks ágiles se dieron a conocer a primeros de este siglo a partir de la publicación del manifiesto ágil [1] y su adopción por parte de las empresas sigue creciendo significativamente en los últimos años [21]. Sin embargo, aunque tan solo tres años después de la publicación manifiesto ágil ya apareciese el primer estudio sobe patrones en entornos ágiles, lo cierto es que hasta el día de hoy las publicaciones en torno a patrones en equipos ágiles han sido escasas e intermitentes.

La Figura 5.1 muestra el número de estudios relacionados con patrones ágiles que han ido apareciendo a lo largo de los años desde la publicación del primer articulo en 2004 hasta el año pasado incluido, 2018.

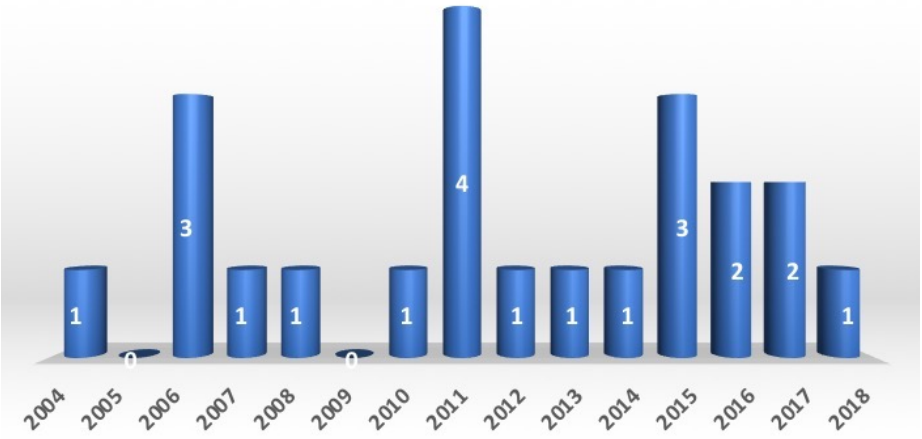

Figura 5.1 - Número de estudios primarios publicados por año

Como se observa en la Figura 5.1, desde que apareció el primer estudio en torno a patrones en equipos ágiles en 2004, **las publicaciones han sido escasas e intermitentes**. Cabe destacar que en tres años no continuos se han publicado el 45% de los estudios analizados, es decir 6 de los 22 estudios primarios. El año 2011 ha sido en el mas prolífico, publicándose 4 de los estudios, seguido de 2006 y de 2015 con 3 estudios cada uno. Por otro lado, en dos años, 2005 y 2009, no se publicó ningún estudio relativo al tema de la investigación.

5.3.2 Medios de publicación de los trabajos

En este apartado se analiza que medio de difusión se ha utilizado para publicar cada estudio. Para el caso concreto de los estudios que ocupan esta investigación, se indica en que revista o conferencia se han publicado. La
Figura 5.2 muestra el número de estudios primarios por publicación.

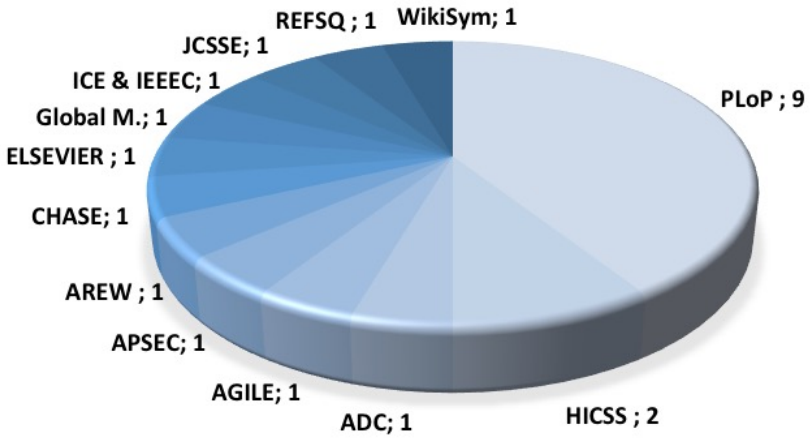

Figura 5.2 - Número de estudios primarios por publicación

En la Figura 5.2 se observa que el **41%** de los estudios, es decir 9 de los 22 estudios primarios, se han publicado en **conferencias PLoP** y el resto en distintos medios sin destacar en número ninguno de ellos:

- **PLoP** - Pattern Languahes of Programs Conference
- **ADC** - Agile Development Conference
- **AGILE** - Agile Conference
- **APSEC** - Asia-Pacific Software Engineering Conference
- **WikiSym** - Wikis International Symposium
- **AREW** - Agile Requirements Engineering Workshop
- **ICE & IEEEC** - International Conference on Engineering, Technology and Innovation & International Technology Management Conference
- **CHASE** - International Workshop on Cooperative and Human Aspects of Software Engineering
- **JCSSE** - Joint International Conference on Computer Science and Software Engineering IEEE
- **HICSS** - International Conference of Sciences of the Hawaiian System
- **ELSEVIER** - Information Technology and Software ELSEVIER
- **REFSQ** - Requirements Engineering Conference - Foundation for Software Quality
- **Global M.** - Global Magazine of Flexible Systems Management

Para completar el análisis de la publicación, se analiza también el editor de la publicación. La Figura 5.3 muestra el número de estudios primarios publicado por editor.

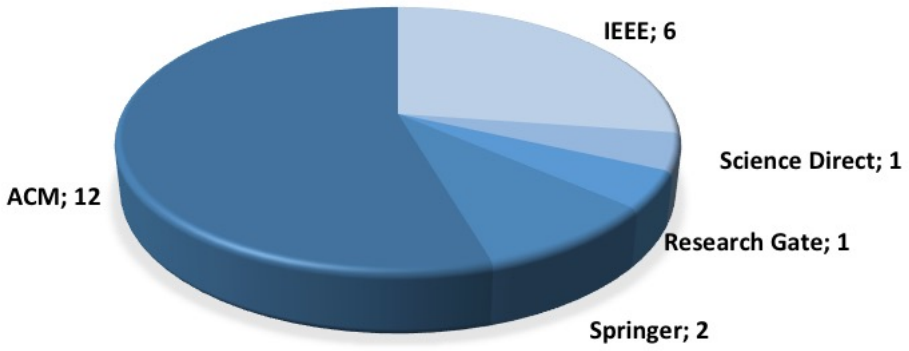

Figura 5.3 - Número de estudios primarios por editor

En la Figura 5.3 se observa que los editores que más estudios primarios han publicado son: ACM con un 55% de los estudios, es decir 9 de los 22 estudios primarios; e IEEE con un 25% de los estudios, es decir con 6 estudios primarios. Entre los dos editores publican el **80% de los estudios seleccionados en esta investigación**. Algo nada extraño si se tiene en cuenta que ambos son asociaciones muy reconocidas en el ámbito de la ingeniería y, en el caso que nos ocupa, en la ingeniería informática.

ACM responde a las siglas *Association for Computing Machinery*. Es **la primera asociación** de carácter científico y educativo dedicada específicamente a las ciencias de la computación. Fundada en 1947 por Edmund C. Berkeley, promueve la libre difusión del conocimiento e intercambio de información entre los investigadores de las ciencias de la computación

IEEE responde a las siglas *Institute of Electrical and Electronics Engineers*. Es **la mayor asociación** internacional sin ánimo de lucro formada por profesionales de las nuevas tecnologías. Se fundó en 1884, siendo algunos de sus fundadores Thomas Alva Edison o Alexander Graham Bell. En 1963, tras la fusión de varias asociaciones, adoptó el nombre actual. Es una asociación mundial que reúne a distintos tipos de profesionales como ingenieros eléctricos, ingenieros en telecomunicación, en biomedicina, además de ingenieros en informática.

5.4 Propuestas de mejora

De todos los estudios analizados en esta revisión sistemática, resulta extraño que siendo la auto-organización una de las características de los equipos ágiles según uno de los principios del manifiesto ágil [1], en ninguno de ellos se propongan patrones que aborden de manera clara y específica problemas derivados de la auto-organización de equipos. Por otra parte, los estudios que apuntan a aspectos relacionados con la comunicación y la colaboración no identifican ningún patrón concreto. De este modo, **una posible mejora sería** profundizar en el estudio de **prácticas ágiles desde el enfoque más humano** del equipo para proponer nuevos patrones de comportamiento y de relación que faciliten la auto-organización de los equipos ágiles.

Pese a que la mayoría de los autores se inspiran en un formato estándar a la hora de formular los patrones que proponen, existen diferentes criterios entre ellos a la hora de formular y describir el patrón. Esta diversidad no permite la fácil comparación entre patrones ni su catalogación. Por otro lado, la densa descripción que algunos autores hacen de los patrones que proponen no facilita identificar de manera clara y rápida el problema que resuelven y como lo resuelven. Por tanto, **otra contribución** al estudio realizado en esta investigación sería realizar una **ontología de patrones** con el objetivo de diseñar un formato de patrón ágil, a modo de canvas, que permita, de un solo vistazo, el entendimiento común del problema que resuelve el patrón, el contexto y su solución.

La recopilación de patrones realizada en esta investigación (Tabla Anexo B.0.1) está clasificada por el estudio que lo refiere y por tanto por el ámbito de aplicación en el ciclo de software. No obstante, **se podría mejorar su catalogación** estudiando en profundidad cada uno de los patrones para identificar cuales de ellos están orientados específicamente al comportamiento del equipo.

Las mejoras propuestas, que **formarán parte la tesis doctoral** de la autora de este trabajo, tienen como propósito final la creación de un framework de patrones, en formato ágil, para facilitar la auto-organización de los equipos ágiles.

6 Conclusiones y trabajos futuros

En este capítulo se presentan los resultados de este trabajo fin de máster en forma de conclusiones. Para ello, se analiza la consecución de objetivos planteada al comienzo de este documento, se muestra la estimación de esfuerzos y se enumeran los trabajos futuros identificados.

6.1 Análisis de la consecución de los objetivos

En el apartado 1.2 de este documento, se propusieron un conjunto de objetivos parciales para facilitar la consecución del objetivo principal: **identificar si existen propuestas de patrones en torno a equipos ágiles y analizar el contexto en el que lo hacen**.

A continuación, se analiza en qué medida se ha conseguido cubrir cada uno de estos objetivos parciales:

Objetivo 1.- Establecer de un proceso de revisión sistemática para identificar y analizar los trabajos encontrados en la literatura.

> En este trabajo se ha presentado un método para realizar un proceso de revisión y ha sido aplicado al objetivo principal de este trabajo fin de máster. Gracias a la consecución de este objetivo parcial, se han conseguido identificar diferentes propuestas de patrones en equipos ágiles, concretamente en el ámbito de ingeniería del software. Una vez identificadas las propuestas, se han analizado dando lugar a un estado del arte.

Objetivo 2.- Definir un conjunto de características que permitan evaluar y comparar las diferentes propuestas encontradas en el proceso de revisión sistemática.

> Para analizar cada una de las propuestas que forman parte del estado del arte presentado y facilitar la consecución del objetivo 1, en el apartado 4.1 se han establecido un conjunto de criterios de evaluación. En el apartado 4.2 se ha analizado el comportamiento de cada propuesta respecto de estos criterios y se ha resumido en forma de ficha de evaluación.
>
> Para clasificar las propuestas según los criterios de evaluación y establecer una comparación entre dichas propuestas, en el apartado 5.1.1 se ha establecido una escala de posibles valores para cada uno de los criterios. A partir de la aplicación de estos valores, en el apartado 5.1.2 de este documento se ha presentado una tabla que permite comparar las propuestas analizadas.

Objetivo 3.- Sintetizar los resultados obtenidos en la revisión sistemática a partir del análisis de los trabajos y de su comparación.

En el apartado 5.2, a partir de los datos de la tabla comparativa de los estudios, se analizan las bondades y limitaciones de las propuestas que forman el estado del arte. La consecución del objetivo 3 ha permitido el cumplimiento del objetivo 4.

Objetivo 4.- Identificar posibles puntos de mejora en el estado del arte.

Como consecuencia de los resultados obtenidos durante la realización del estado del arte presentado en este documento y de mi experiencia en entornos ágiles, en el apartado 5.4 se propone el desarrollo de un marco de trabajo ágil que incorpore el uso de patrones ágiles aplicables a todo el ciclo de vida de un producto software. Este framework, que se abordará en el contexto de la tesis doctoral de la autora de este trabajo, tendrá por objetivo facilitar la adaptación de las empresas para desarrollar modelos de negocio ágiles en torno a un producto software.

6.2 Estimación de esfuerzos

En este apartado se presenta la estimación del esfuerzo realizado en el periodo de elaboración de este trabajo fin de máster. La duración del periodo ha sido de 12 meses, de julio de 2018 a junio de 2019. El esfuerzo realizado ha sido el equivalente a **1.100 horas de trabajo**. La Figura 6.1 muestra el esfuerzo dedicado a las principales actividades realizadas en el trabajo.

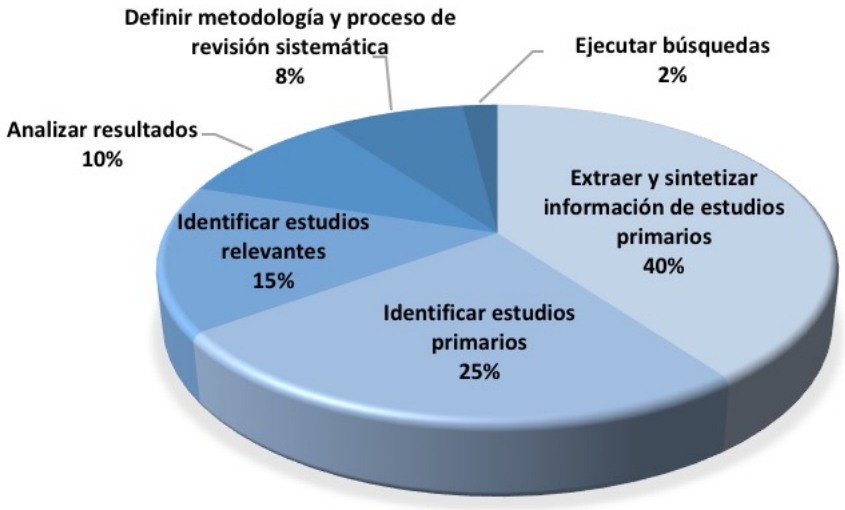

Figura 6.1 – Estimación de esfuerzo por actividad

Como se puede observar, la actividad que ha requerido mayor esfuerzo ha sido la **extracción y síntesis de la información de los 22 estudios primarios identificados (40%)**. Para comprender el trabajo propuesto en cada estudio primario y extraer la información relevante para su análisis ha sido necesario realizar diversas iteraciones de esta actividad. Además, se debe tener en cuenta que todos los estudios primarios están escritos en inglés, por tanto, ya que no es el idioma nativo, el esfuerzo para la comprensión de los trabajos es mayor.

Otra actividad que ha supuesto un gran esfuerzo ha sido la **identificación de los estudios primarios (25%)**. Para llevar a cabo esta actividad ha sido necesario revisar todos los trabajos identificados previamente como relevante y evaluar si cumplen con el criterio de exclusión. En este caso el esfuerzo proviene de la gran cantidad de trabajos relevantes, 302 artículos tal y como se indica en el apartado 3.4.

En tercer lugar, se encuentra la **identificación de los estudios relevantes (15%)**. Esta actividad comprende dos subtareas: en primer lugar, ha sido leído el título y el resumen de todos los estudios obtenidos como resultado de las búsquedas en cada fuente de información. Esta lectura tenía por objetivo evaluar si los trabajos cumplían el criterio de inclusión; y a continuación, se ha realizado un filtrado para eliminar la repetición de aquellos trabajos que aparecían en distintas fuentes de información. De nuevo, el esfuerzo es proporcional al número de estudios que se deben leer y analizar.

La siguiente actividad en cuanto al esfuerzo dedicado es el **análisis de los resultados (10%)**. A partir de los criterios de evaluación y de la tabla comparativa de las propuestas (Tabla 5.2) se han analizado los aspectos más significativos estado del arte. Con estos datos se ha analizado qué tipo de contribución se podía aportar para mejorar el estado del arte existente.

En quinto lugar, se encuentra la **definición de la metodología y el proceso de la revisión sistemática (8%)**. Poder consultar trabajos como [9], [8], [15], [14] y haber realizado previamente una práctica, relacionada con la asignatura “Métodología y Documentación Científico Técnica” del máster, en la que se realizaban los primeros pasos de una revisión sistemática hasta la selección de estudios primarios, ha facilitado el desarrollo de esta tarea minimizando su esfuerzo.

En último lugar a nivel de esfuerzo está la **ejecución de las búsquedas (2%)**. Esta tarea sólo implica adecuar la cadena de búsqueda al motor de la fuente de información y llevar a cabo la búsqueda. Dado que la mayoría de las fuentes de información seleccionadas tratan las cadenas de forma similar, una misma adaptación ha podido ser empleada en distintas fuentes, tal y como se muestra en la Tabla 3.3. Por tanto, la adaptación de la cadena de búsqueda a los motores de las fuentes de información no ha supuesto un gran esfuerzo adicional.

6.3 Trabajos futuros

Para dar continuidad a la labor de investigación realizada en este trabajo fin de máster, se proponen las siguientes líneas de trabajo:

- Por una parte, dado que la aplicación de patrones específicos para entornos ágiles es un tema de investigación reciente, intermitente y escaso, se propone seguir realizando nuevas **revisiones de la literatura periódicas** para identificar los trabajos que hayan sido publicados después de finalizar este trabajo fin de máster.

- Dado que no existen revisiones sistemáticas de patrones en entorno ágiles, se propone **elaborar artículos a partir de los datos de esta revisión** para publicarlo en algún editor de impacto y/o presentarlo en algún congreso de interés. Algunos de los artículos sobre los que se podría trabajar serían:

    - Revisión sistemática de patrones en equipos ágiles.
    - Revisión sistemática de anti-patrones en equipos ágiles.
    - Relación de patrones existentes en entornos ágiles.
    - Formulación de patrones en entornos ágiles.

- En relación con la línea de investigación futura planteada en el apartado 5.4 de este documento, se propone realizar un estudio ontológico que permita estructurar y unificar el conocimiento acumulado de patrones en equipos ágiles. Para ello se prevé la realización de los siguientes trabajos (algunos actualmente en curso):

    - Profundizar en el análisis de los patrones recopilados en este trabajo para mejorar su catalogación.
    - Explorar prácticas ágiles relacionadas con el comportamiento humano para complementar a los patrones ya identificados en torno a equipos ágiles.
    - Clasificar los patrones recopilados en función de la dimensión en la que se apliquen dentro de un equipo ágil: cultura (dimensión relacionada con el comportamiento), proceso (dimensión relacionada con el método) o estructura (dimensión relacionada con la organización).
    - Diseñar un formato de patrón ágil (unificado) para mejorar la comprensión del patrón y garantizar su entendimiento común.
    - Crear un framework ágil de patrones que resuelvan problemas derivados de la adaptación de las empresas tradicionales a entornos ágiles, y que están relacionados con la cultura, el proceso y la estructura.
    - Extrapolar los patrones a cualquier actividad, sector o industria en la que un equipo auto-organizado tenga que aportar valor de manera iterativa e incremental, en periodos cortos de tiempo, maximizando ese valor y

optimizando al máximo su esfuerzo, independientemente del tipo de proyecto, tamaño de la empresa o estructura organizativa. En definitiva, extender este framework a cualquier actividad del ciclo de vida de un producto software más allá de lo que estrictamente sería la ingeniería del software, es decir, desde la generación de la idea hasta la comercialización del producto.

- Aunque no forma parte de la literatura científica, también se propone **analizar el lenguaje de patrones basado en el marco Scrum** y valorar el incluirlo en el catalogo resultante de la tesis doctoral. En el Anexo D se puede consultar el mapa de patrones de este lenguaje publicado en SrumPLoP [53].

Todas las líneas propuestas, que se pretenden llevar a cabo como **parte de la tesis doctoral** de la autora de este trabajo, tienen el objetivo de facilitar la auto-organización en los equipos ágiles y el propósito final de ayudar a las empresas a ser ágiles, es decir, ayudarlas a adaptar su negocio, sin que su rentabilidad se resienta, a los cambios continuos que la tecnología está provocando en los hábitos de consumo de los clientes.

Referencias

[1] K. Beck *et al.*, “Manifesto for Agile Software Development.,” 2001.

[2] C. Ramos, “Primera parada: #AGILIDAD,” 2016. [Online]. Available: https://cristinaramosvega.com/primera-parada-agilidad/.

[3] C. Ramos, “La #Agilidad, tu compañera de viaje en #TransformacionDigital,” 2017. [Online]. Available: https://cristinaramosvega.com/la-agilidad-tu-companera-de-viaje-en-transformaciondigital/.

[4] C. Ramos, “#CulturaAgil, un must, en #AgilidadCorporativa,” 2016. [Online]. Available: https://cristinaramosvega.com/culturaagil-must-agilidadcorporativa/.

[5] K. Beck, *Smalltalk best practice patterns*. Prentice Hall, 1997.

[6] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns : elements of reusable object-oriented software*. Addison-Wesley, 1995.

[7] J. Coplien and N. Harrison, *Organizational patterns of agile software development*. Pearson Prentice Hall, 2004.

[8] B. Kitchenham, “Procedures for Performing Systematic Reviews,” 2004.

[9] J. Biolchini, P. Gomes Mian, A. Candida Cruz Natali, and G. Horta Travassos, “Systematic Review in Software Engineering,” 2005.

[10] B. Kitchenham, S. Charters, K. BA, and S. Charters, “Guidelines for performing Systematic Literature Reviews in Software Engineering,” *Engineering*, vol. 2, 2007.

[11] B. A. Kitchenham *et al.*, “Preliminary guidelines for empirical research in software engineering,” *IEEE Trans. Softw. Eng.*, vol. 28, no. 8, pp. 721–734, Aug. 2002.

[12] T. Dyba, B. A. Kitchenham, and M. Jorgensen, “Evidence-based software engineering for practitioners,” *IEEE Softw.*, vol. 22, no. 1, pp. 58–65, Jan. 2005.

[13] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering – A systematic literature review,” *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, Jan. 2009.

[14] I. Santiago, Á. Jiménez, J. M. Vara, V. De Castro, V. A. Bollati, and E. Marcos, “Model-Driven Engineering as a new landscape for traceability management: A systematic literature review,” *Inf. Softw. Technol.*, vol. 54, no. 12, pp. 1340–1356, Dec. 2012.

[15] T. Dybå and T. Dingsøyr, “Empirical studies of agile software development: A systematic review,” *Inf. Softw. Technol.*, vol. 50, no. 9–10, pp. 833–859, Aug. 2008.

[16] K. Beck *et al.*, “Principles behind the Agile Manifesto,” 2001. [Online]. Available: https://agilemanifesto.org/principles.html.

[17] C. Ramos, “#Agilidad: Personas sobre Procesos!,” 2016. [Online]. Available: https://cristinaramosvega.com/agilidad-personas-procesos/.

[18] J. Garzás, “El equipo ágil auto-organizado,” 2012. [Online]. Available: https://www.javiergarzas.com/2012/07/equipo-agil-auto-organizado.html.

[19] K. Beck, *Extreme programming eXplained: embrace change*. Addison-Wesley, 2000.
[20] K. Schwaber and J. Sutherland, "The Scrum Guide™ The Definitive Guide to Scrum: The Rules of the Game," 2017.
[21] Versionone, "State of Agile Survey (2019)," 2019. [Online]. Available: https://www.stateofagile.com/#ufh-i-521251909-13th-annual-state-of-agile-report/473508.
[22] C. Alexander, "A pattern language: towns, buildings, construction," 1977.
[23] C. Alexander, "The timeless way of building," 1979.
[24] K. Beck and W. Cunningham, "Using Pattern Languages for Object-Oriented Programs," 1987. [Online]. Available: http://c2.com/doc/oopsla87.html.
[25] Wikipedia, "OOPSLA." [Online]. Available: https://en.wikipedia.org/wiki/OOPSLA.
[26] James O. Coplien, *Advanced C++ Programming Styles and Idioms*. Addison-Wesley, 1991.
[27] P. Coad, "Object-oriented patterns," *Commun. ACM*, vol. 35, no. 9, pp. 152–159, Sep. 1992.
[28] J. Garzás, "De dónde viene la palabra 'patrón' aplicada al software (sí, otra herencia de la arquitectura) - Javier Garzas." [Online]. Available: https://www.javiergarzas.com/2014/05/de-donde-viene-la-palabra-patron-aplicada-al-software.html. [Accessed: 16-Jun-2019].
[29] G. Meszaros and Doble, "A Pattern Language for Pattern Writing." [Online]. Available: https://hillside.net/index.php/a-pattern-language-for-pattern-writing.
[30] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-oriented software architecture: a system of patterns*. Wiley, 1996.
[31] M. Fowler, *Analysis patterns: reusable object models*. Addison Wesley, 1997.
[32] S. Ambler, *Process patterns: building large-scale systems using object technology*. Cambridge University Press, 1998.
[33] W. Brown, R. Malveau, H. McCormick, and T. Mowbray, *AntiPatterns: refactoring software, architectures, and projects in crisis*. 1998.
[34] L. Rising, *The pattern almanac 2000*. Addison-Wesley, 2000.
[35] L. Rising and K. Rehmer, "Patterns for sustainable development (2009)," in *Proceedings of the 16th Conference on Pattern Languages of Programs - PLoP '09*, 2009, p. 1.
[36] L. Rising and K. Rehmer, "Patterns for sustainable development (2010)," in *Proceedings of the 17th Conference on Pattern Languages of Programs - PLOP '10*, 2010, pp. 1–11.
[37] M. Manns and L. Rising, *Fearless change : patterns for introducing new ideas*. Addison-Wesley, 2005.
[38] M. Manns and L. Rising, "Additional patterns for fearless change," in *Proceedings of the 15th Conference on Pattern Languages of Programs - PLoP '08*, 2008, p. 1.
[39] M. Manns and L. Rising, "Additional patterns for fearless change II," in *Proceedings of the 16th Conference on Pattern Languages of Programs - PLoP '09*, 2010, p. 1.

[40] M. Manns and L. Rising, “Additional patterns for fearless change III,” in *Proceedings of the 17th Conference on Pattern Languages of Programs - PLOP ’10*, 2010, pp. 1–14.
[41] M. Manns and L. Rising, “Additional patterns for fearless change, part IV,” in *Proceedings of the 9th Latin-American Conference on Pattern Languages of Programming - SugarLoafPLoP ’12*, 2012, pp. 1–11.
[42] J. Garzás, “¿Qué tipos de patrones hay?,” 2014. [Online]. Available: https://www.javiergarzas.com/2014/08/tipos-patrones-software.html.
[43] F. J. Pino, F. García, and M. Piattini, “Software process improvement in small and medium software enterprises: a systematic review,” *Softw. Qual. J.*, vol. 16, no. 2, pp. 237–261, Jun. 2008.
[44] J. Bergin, “Patterns for Agile Development Practice Part 2,” 2006.
[45] J. Bergin, “Patterns for Agile Development Practice Part 1 (Version 7),” 2005. [Online]. Available: https://csis.pace.edu/~bergin/patterns/xpPatternsEuroV7.html.
[46] M. Cohn, *Agile estimating and planning*. Prentice Hall Professional Technical Reference, 2006.
[47] S. McConnell, *Software estimation: demystifying the black art*. Microsoft Press, 2006.
[48] I. Nonaka and H. Takeuchi, *The knowledge-creating company : how Japanese companies create the dynamics of innovation*. Oxford University Press, 1995.
[49] D. Greening, “Agile Base Patterns in the Agile Canon,” in *2016 49th Hawaii International Conference on System Sciences (HICSS)*, 2016, pp. 5368–5377.
[50] V. Eloranta, K. Koskimies, and T. Mikkonen, “Exploring ScrumBut—An empirical study of Scrum anti-patterns,” *Inf. Softw. Technol.*, vol. 74, pp. 194–203, Jun. 2016.
[51] L. Hvatum and R. Wirfs-Brock, “Patterns to build the magic backlog (2015),” in *Proceedings of the 20th European Conference on Pattern Languages of Programs - EuroPLoP ’15*, 2015, pp. 1–36.
[52] R. Wirfs-Brock and L. Hvatum, “More patterns for the magic backlog (2016),” *Proceedings of the 23rd Conference on Pattern Languages of Programs*. The Hillside Group, p. 22, 2016.
[53] ScrumPLoP, “History of the Patterns,” 2019. [Online]. Available: https://sites.google.com/a/scrumplop.org/published-patterns/book-outline/history-of-the-patterns.
[54] VersionOne, “State of Agile Survey.” [Online]. Available: https://www.stateofagile.com/#ufh-c-473508-state-of-agile-report.

Anexo A. Referencias de los estudios primarios

En este anexo se recogen las referencias especificas de los estudios primarios analizados en el presente trabajo fin de máster.

[EP01] Y. Kuranuki and K. Hiranabe, “AntiPractices: AntiPatterns for XP practices,” in *Agile Development Conference*, 2004, pp. 83–86.

[EP02] M. Yap, “Value based extreme programming,” in AGILE 2006 (AGILE’06), 2006, pp. 8 pp. – 184. [EP05] X. Meng, Y. Wang, L. Shi, and F. Wang, “A Process Pattern Language for Agile Methods,” in 14th Asia-Pacific Software Engineering Conference (APSEC’07), 2007, pp. 374–381.

[EP03] J. Bergin and Joseph, “Patterns for agile development practice part 3 (version 4),” in Proceedings of the 2006 conference on Pattern languages of programs - PLoP ’06, 2006, p. 1.

[EP04] A. Martin, J. Noble, and R. Biddle, “Programmers are from Mars, customers are from Venus,” in Proceedings of the 2006 conference on Pattern languages of programs - PLoP ’06, 2006, p. 1.

[EP05] X. Meng, Y. Wang, L. Shi, and F. Wang, “A Process Pattern Language for Agile Methods,” in 14th Asia-Pacific Software Engineering Conference (APSEC’07), 2007, pp. 374–381.

[EP06] M. Cubric, “Agile learning and teaching with wikis: Building a pattern,” in WikiSym 2008 - The 4th International Symposium on Wikis, Proceedings, 2008. [EP15] T. Frt’ala and V. Vranić, “Animating organizational patterns,” Proceedings of the Eighth International Workshop on Cooperative and Human Aspects of Software Engineering. IEEE Press, pp. 8–14, 2015.

[EP07] R. Hoda, J. Noble, and S. Marshall, “How much is just enough?,” in Proceedings of the 15th European Conference on Pattern Languages of Programs - EuroPLoP ’10, 2010, p. 1.

[EP08] N. Abdullah, H. Shinichi, H. Sharp, N. Bashar and D. Notkin, “Communication patterns of agile requirements engineering,” in Proceedings of the 1st Workshop on Agile Requirements Engineering - AREW ’11, 2011, pp. 1–4.

[EP09] R. Rodin, J. Leet, M. Azua, and D. Bygrave, A pattern language for release and deployment management. New York, New York, USA: ACM Press, 2011.

[EP10] D. Cukier and F. Kon, “Extending patterns for fearless change,” in Proceedings of the 18th Conference on Pattern Languages of Programs - PLoP ’11, 2011, pp. 1–10.

[EP11] D. Nikelshpur, “The Art of Software Estimation Pattern Language ,” 2011. [EP19] V.-P. Eloranta, K. Koskimies, and T. Mikkonen, “Exploring ScrumBut—An empirical study of Scrum anti-patterns,” Inf. Softw. Technol., vol. 74, pp. 194–203, Jun. 2016.

[EP12] M. Braga et al., “A pattern language for agile software estimation,” in Proceedings of the 9th Latin-American Conference on Pattern Languages of Programming - SugarLoafPLoP ’12, 2012, vol. 2012-Septe, pp. 1–15.

[EP13] C. Stettina and E. Kroon, “Is there an agile handover? An empirical study of documentation and project handover practices across agile software teams,” in 2013 International Conference on Engineering, Technology and Innovation (ICE) & IEEE International Technology Management Conference, 2013, pp. 1–12.

[EP14] J. Sutherland, N. Harrison, and J. Riddle, “Teams that finish early accelerate faster: A pattern language for high performing scrum teams,” in Proceedings of the Annual Hawaii International Conference on System Sciences, 2014, pp. 4722–4728.

[EP15] T. Frt’ala and V. Vranić, “Animating organizational patterns,” Proceedings of the Eighth International Workshop on Cooperative and Human Aspects of Software Engineering. IEEE Press, pp. 8–14, 2015.

[EP16] A. Capiluppi et al., “Patterns for integrating agile development processes and user centred design,” in 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, 2015, vol. 2, pp. 59–68.

[EP17] M. Kausar and A. Al-Yasiri, Distributed Agile Patterns for Offshore Software Development CBSE for Large Scale Reuse View project Global Identity Management in the Internet of things View project Distributed Agile Patterns for Offshore Software Development. 2015.

[EP18] D. Greening, “Agile Base Patterns in the Agile Canon,” 2016, vol. 2016-March, pp. 5368–5377.

[EP19] V. Eloranta, K. Koskimies, and T. Mikkonen, “Exploring ScrumBut—An empirical study of Scrum anti-patterns,” Inf. Softw. Technol., vol. 74, pp. 194–203, Jun. 2016.

[EP20] I. Inayat, S. Marczak, S. S. Salim, and D. Damian, Patterns of collaboration driven by requirements in agile software development teams: Findings from a multiple case study, vol. 10153 LNCS. 2017.

[EP21] P. Carew and D. Glynn, “Anti-patterns in Agile Adoption: A Grounded Theory Case Study of One Irish IT Organisation,” Glob. J. Flex. Syst. Manag., vol. 18, no. 4, pp. 275–289, Dec. 2017.

[EP22] L. Hvatum and R. Wirfs-Brock, “A Program Backlog Story with Patterns,” in Proceedings of the 23rd European Conference on Pattern Languages of Programs - EuroPLoP ’18, 2018, pp. 1–18.

Anexo B. Resumen de patrones por estudio primario

En este anexo se registran los 262 patrones identificados en el transcurso de la revisión sistemática, clasificados por el estudio primario en el que se ha localizado. Esta relación incluye las colecciones referidas en alguno de los estudios. Los estudios primarios que referencian colecciones de patrones están marcados con un "*" en su numeración.

Para cada patrón se informa, su número de orden secuencial (del 1 al 262), su número de orden dentro del estudio primario, su nombre, el identificador del estudio primario y el nombre corto del estudio. Para los patrones pertenecientes a una colección, en el nombre corto se indica el nombre del estudio original dónde apareció por primera vez.

Los anti-patrones aparecen en la tabla con color azul.

		Patrón	EP	Nombre corto del estudio
1	1	Brownie's Works - "The boss refactored our code!"	EP01	AntiPractices: AntiPatterns for XP practices
2	2	Anybody Syndrome - "I'm not necessary"	EP01	AntiPractices: AntiPatterns for XP practices
3	3	Pairing Prison - "I'm always under observation!"	EP01	AntiPractices: AntiPatterns for XP practices
4	1	Powerful Owner - "We are under his thumb!"	EP01	AntiPractices: AntiPatterns for XP practices
5	2	Outnumbers View - "We are a minority group"	EP01	AntiPractices: AntiPatterns for XP practices
6	3	Imperial XP - "Why do I do such a thing?"	EP01	AntiPractices: AntiPatterns for XP practices
7	4	Loose Control - "Sure. XP is free, you know"	EP01	AntiPractices: AntiPatterns for XP practices
8	5	Frozen Pair - "The two are pairing ... again!"	EP01	AntiPractices: AntiPatterns for XP practices
9	6	Formal Format - "Do we have to do all the practices just as in the book?"	EP01	AntiPractices: AntiPatterns for XP practices
10	7	Misleading Influenza - "I heard you are doing THE XP"	EP01	AntiPractices: AntiPatterns for XP practices
11	1	Value-based Investment Decisions	EP02	Value based extreme programming
12	2	High Confidence Stories First	EP02	Value based extreme programming

13	3	Incremental Story Delivery	EP02	Value based extreme programming
14	4	Story Ownership	EP02	Value based extreme programming
15	1	Acceptance Test	EP03*	Patterns for agile development practice part 3 (version4)
16	2	Ask for More	EP03*	Patterns for agile development practice part 3 (version4)
17	3	Be Human	EP03*	Patterns for agile development practice part 3 (version4)
18	4	Best Effort	EP03*	Patterns for agile development practice part 3 (version4)
19	5	Bug Generates Test	EP03*	Patterns for agile development practice part 3 (version4)
20	6	Cards and Whiteboards	EP03*	Patterns for agile development practice part 3 (version4)
21	7	Coding Standards	EP03*	Patterns for agile development practice part 3 (version4)
22	8	Collective Responsibiity	EP03*	Patterns for agile development practice part 3 (version4)
23	9	Collective Ownership	EP03*	Patterns for agile development practice part 3 (version4)
24	10	Constant Refactoring	EP03*	Patterns for agile development practice part 3 (version4)
25	11	Continuous Integration*	EP03*	Patterns for agile development practice part 3 (version4)
26	12	Customer Checks off Tasks*	EP03*	Patterns for agile development practice part 3 (version4)
27	13	Customer Obtains Consensus*	EP03*	Patterns for agile development practice part 3 (version4)
28	14	Deliver (Customer) Value	EP03*	Patterns for agile development practice part 3 (version4)
29	15	Documentation is Just Another Task	EP03*	Patterns for agile development practice part 3 (version4)
30	16	DTSTTCPW-Do the Simplest Thing that Could Possibly Work	EP03*	Patterns for agile development practice part 3 (version4)
31	17	Easy Does It	EP03*	Patterns for agile development practice part 3 (version4)
32	18	Effective Coach	EP03*	Patterns for agile development practice part 3 (version4)
33	19	End to End	EP03*	Patterns for agile development practice part 3 (version4)
34	20	Estimate Whole Task	EP03*	Patterns for agile development practice part 3 (version4)
35	21	Everything Visible	EP03*	Patterns for agile development practice part 3 (version4)
36	22	Executable Tests*	EP03*	Patterns for agile development practice part 3 (version4)
37	23	Flexible Velocity	EP03*	Patterns for agile development practice part 3 (version4)
38	24	Full Communication	EP03*	Patterns for agile development practice part 3 (version4)
39	25	Grow Up	EP03*	Patterns for agile development practice part 3 (version4)
40	26	Guiding Metaphor (Topos)	EP03*	Patterns for agile development practice part 3 (version4)
41	27	High Discipline	EP03*	Patterns for agile development practice part 3 (version4)
42	28	High Value First	EP03*	Patterns for agile development practice part 3 (version4)

43	29	Implementer Re-estimates Tasks	EP03*	Patterns for agile development practice part 3 (version4)
44	30	Individual Stakeholder Budgets*	EP03*	Patterns for agile development practice part 3 (version4)
45	31	Infrastructure	EP03*	Patterns for agile development practice part 3 (version4)
46	32	Negotiated Scope Contract	EP03*	Patterns for agile development practice part 3 (version4)
47	33	Offer Alternatives	EP03*	Patterns for agile development practice part 3 (version4)
48	34	Once and Only Once*	EP03*	Patterns for agile development practice part 3 (version4)
49	35	Onsite Customer	EP03*	Patterns for agile development practice part 3 (version4)
50	36	Our Space	EP03*	Patterns for agile development practice part 3 (version4)
51	37	Pair Programming	EP03*	Patterns for agile development practice part 3 (version4)
52	38	Personal Velocity	EP03*	Patterns for agile development practice part 3 (version4)
53	39	Planning Game	EP03*	Patterns for agile development practice part 3 (version4)
54	40	Promiscuuous Programming	EP03*	Patterns for agile development practice part 3 (version4)
55	41	Project Diary*	EP03*	Patterns for agile development practice part 3 (version4)
56	42	Question Implies Acceptance Test	EP03*	Patterns for agile development practice part 3 (version4)
57	43	Re-estimate Periodically	EP03*	Patterns for agile development practice part 3 (version4)
58	44	Retrospective	EP03*	Patterns for agile development practice part 3 (version4)
59	45	Sacred Schedule	EP03*	Patterns for agile development practice part 3 (version4)
60	46	Scrum Master	EP03*	Patterns for agile development practice part 3 (version4)
61	47	Sheitering Manager	EP03*	Patterns for agile development practice part 3 (version4)
62	48	Simple Design	EP03*	Patterns for agile development practice part 3 (version4)
63	49	Small Releases	EP03*	Patterns for agile development practice part 3 (version4)
64	50	Social Tracker*	EP03*	Patterns for agile development practice part 3 (version4)
65	51	Spike	EP03*	Patterns for agile development practice part 3 (version4)
66	52	Stand up Meeting*	EP03*	Patterns for agile development practice part 3 (version4)
67	53	Sustainable Pace	EP03*	Patterns for agile development practice part 3 (version4)
68	54	Team Continuity	EP03*	Patterns for agile development practice part 3 (version4)
69	55	Test Card	EP03*	Patterns for agile development practice part 3 (version4)
70	56	Test First*	EP03*	Patterns for agile development practice part 3 (version4)
71	57	Train Everyone	EP03*	Patterns for agile development practice part 3 (version4)
72	58	Whole Team	EP03*	Patterns for agile development practice part 3 (version4)

73	59	YAGNI-You Ain't Gonna Need it	EP03*	Patterns for agile development practice part 3 (version4)
74	60	Yesterday's Weather	EP03*	Patterns for agile development practice part 3 (version4)
75	1	Geek Interpreter	EP04	Programmers are from Mars, customers are from Venus …
76	2	Technical Liaison	EP04	Programmers are from Mars, customers are from Venus …
77	3	Political Advisor	EP04	Programmers are from Mars, customers are from Venus …
78	4	Acceptance Tester	EP04	Programmers are from Mars, customers are from Venus …
79	5	User Interface Designer	EP04	Programmers are from Mars, customers are from Venus …
80	6	Technical Writer	EP04	Programmers are from Mars, customers are from Venus …
81	7	Diplomat	EP04	Programmers are from Mars, customers are from Venus …
82	8	Super-Secretary	EP04	Programmers are from Mars, customers are from Venus …
83	9	Negotiator	EP04	Programmers are from Mars, customers are from Venus …
84	1	Sprint	EP05	A process pattern language for agile methods
85	1	Agile Learning and Teaching	EP06	Agile Learning & Teaching with Wikis: Building a Pattern
86	1	Fake Documentacion	EP07	Extending patterns for fearless change
87	2	Time Stamp	EP07	Extending patterns for fearless change
88	3	e-Backup	EP07	Extending patterns for fearless change
89	4	Project Dictionary	EP07	Extending patterns for fearless change
90	5	Advantage Agile	EP07	Extending patterns for fearless change
91	1	Obtain certificate originality	EP09	A Pattern Language for Release and Deployment Management
92	2	Release Scheduler	EP09	A Pattern Language for Release and Deployment Management
93	3	Release Acceptance	EP09	A Pattern Language for Release and Deployment Management
94	4	Emergency Release	EP09	A Pattern Language for Release and Deployment Management
95	5	Deployment Preparation	EP09	A Pattern Language for Release and Deployment Management
96	6	Application Retirement	EP09	A Pattern Language for Release and Deployment Management
97	1	Let Them Play	EP10*	Extending patterns for fearless change
98	2	Deep Dive	EP10*	Extending patterns for fearless change
99	3	Pair to Share	EP10*	Extending patterns for fearless change
100	4	Do Art	EP10*	Extending patterns for fearless change

101	1	Passing on the Gift	EP10*	Patterns for Sustainable Development-2009
102	2	Small Support Group	EP10*	Patterns for Sustainable Development-2009
103	3	Empowering Women	EP10*	Patterns for Sustainable Development-2009
104	4	Organizational Conrnerstones	EP10*	Patterns for Sustainable Development-2010
105	5	Cultural and Environmental Awareness	EP10*	Patterns for Sustainable Development-2010
106	6	Early Wins	EP10*	Patterns for Sustainable Development-2010
107	7	Constructive Participation	EP10*	Patterns for Sustainable Development-2010
108	1	Ask for Help	EP10*	Patterns for Fearless Change
109	2	Big Jolt	EP10*	Patterns for Fearless Change
110	3	Bridge-Builder	EP10*	Patterns for Fearless Change
111	4	Brown Bag	EP10*	Patterns for Fearless Change
112	5	Champion Skeptic	EP10*	Patterns for Fearless Change
113	6	Connector	EP10*	Patterns for Fearless Change
114	7	Corporate Angel	EP10*	Patterns for Fearless Change
115	8	Corridor Politics	EP10*	Patterns for Fearless Change
116	9	Delicated Champion	EP10*	Patterns for Fearless Change
117	10	Do Food	EP10*	Patterns for Fearless Change
118	11	e-Forum	EP10*	Patterns for Fearless Change
119	12	Early Adopter	EP10*	Patterns for Fearless Change
120	13	Early Majority	EP10*	Patterns for Fearless Change
121	14	Evangelist	EP10*	Patterns for Fearless Change
122	15	External Validation	EP10*	Patterns for Fearless Change
123	16	Fear Less	EP10*	Patterns for Fearless Change
124	17	Group Identity	EP10*	Patterns for Fearless Change
125	18	Guru on Your Side	EP10*	Patterns for Fearless Change
126	19	Guru Review	EP10*	Patterns for Fearless Change
127	20	Hometown Story	EP10*	Patterns for Fearless Change
128	21	In Your Space	EP10*	Patterns for Fearless Change
129	22	Innovator	EP10*	Patterns for Fearless Change

130	23	Involve Everyone	EP10*	Patterns for Fearless Change
131	24	Just Do It	EP10*	Patterns for Fearless Change
132	25	Just Enough	EP10*	Patterns for Fearless Change
133	26	Just Say Thanks	EP10*	Patterns for Fearless Change
134	27	Local Sponsor	EP10*	Patterns for Fearless Change
135	28	Location, Location, Location	EP10*	Patterns for Fearless Change
136	29	Mentor	EP10*	Patterns for Fearless Change
137	30	Next Steps	EP10*	Patterns for Fearless Change
138	31	Personal Touch	EP10*	Patterns for Fearless Change
139	32	Piggyback	EP10*	Patterns for Fearless Change
140	33	Plant the Seeds	EP10*	Patterns for Fearless Change
141	34	Royal Audience	EP10*	Patterns for Fearless Change
142	35	Shoulder to Cry On	EP10*	Patterns for Fearless Change
143	36	Small Successes	EP10*	Patterns for Fearless Change
144	37	Smell of Success	EP10*	Patterns for Fearless Change
145	38	Stay in Touch	EP10*	Patterns for Fearless Change
146	39	Step by Step	EP10*	Patterns for Fearless Change
147	40	Study Group	EP10*	Patterns for Fearless Change
148	41	Sustained Momentum	EP10*	Patterns for Fearless Change
149	42	Tailor Made	EP10*	Patterns for Fearless Change
150	43	Test the Waters	EP10*	Patterns for Fearless Change
151	44	The Right Time	EP10*	Patterns for Fearless Change
152	45	Time For Reflection	EP10*	Patterns for Fearless Change
153	46	Token	EP10*	Patterns for Fearless Change
154	47	Trial Run	EP10*	Patterns for Fearless Change
155	48	Whisper in the General's Ear	EP10*	Patterns for Fearless Change
156	1	Wake-up Call	EP10*	Additional Patterns for Fearless Change
157	2	Town meeting	EP10*	Additional Patterns for Fearless Change
158	3	Emotional connection	EP10*	Additional Patterns for Fearless Change

159	4	Elevator speech	EP10*	Additional Patterns for Fearless Change
160	5	Pick your battles	EP10*	Additional Patterns for Fearless Change
161	6	Concrete Action Plan	EP10*	Additional Patterns for Fearless Change II
162	7	Small Concession	EP10*	Additional Patterns for Fearless Change II
163	8	Myth Buster	EP10*	Additional Patterns for Fearless Change II
164	9	Go to Person	EP10*	Additional Patterns for Fearless Change II
165	10	Imagine That!	EP10*	Additional Patterns for Fearless Change II
166	11	Know Yourself	EP10*	Additional Patterns for Fearless Change III
167	12	Accentuate the Positive	EP10*	Additional Patterns for Fearless Change III
168	13	Low Hanging Fruit	EP10*	Additional Patterns for Fearless Change III
169	14	Easier Path	EP10*	Additional Patterns for Fearless Change IV
170	15	Evolving Vision	EP10*	Additional Patterns for Fearless Change IV
171	16	Low Hanging Fruit (rep-mejorado?)	EP10*	Additional Patterns for Fearless Change IV
172	17	Futre Commitment	EP10*	Additional Patterns for Fearless Change IV
173	1	Cone of Uncertainty	EP11	The Art of Software Estimation Pattern Language
174	2	One Bite at a Time	EP11	The Art of Software Estimation Pattern Language
175	3	Time is Relative	EP11	The Art of Software Estimation Pattern Language
176	4	[Method of] Succesisive Estimations	EP11	The Art of Software Estimation Pattern Language
177	5	Experts Estimate	EP11	The Art of Software Estimation Pattern Language
178	6	Safe Estimating	EP11	The Art of Software Estimation Pattern Language
179	7	Find Something to Count	EP11	The Art of Software Estimation Pattern Language
180	8	Déjà Vu	EP11	The Art of Software Estimation Pattern Language
181	9	All Aboard	EP11	The Art of Software Estimation Pattern Language
182	1	Estimating size with Story Points	EP12	A pattern language for agile software estimation
183	2	Estimating Effort Based on Tasks	EP12	A pattern language for agile software estimation
184	3	Estimating Velocity	EP12	A pattern language for agile software estimation
185	4	Estimating Duration Based on Team Velocity and Project Size	EP12	A pattern language for agile software estimation
186	5	Financial Cost Estimation	EP12	A pattern language for agile software estimation

187	6	Negotiating Estimates	EP12	A pattern language for agile software estimation
188	7	Collecting Software Measures	EP12	A pattern language for agile software estimation
189	8	Historical Data Adquisition	EP12	A pattern language for agile software estimation
190	1	Stable Teams	EP14	Teams that finish early accelerate faster … high performing …
191	2	Yesterday's Weather	EP14	Teams that finish early accelerate faster … high performing …
192	3	Swarming_ One Piece Continuous Flow	EP14	Teams that finish early accelerate faster … high performing …
193	4	Interrupt Pattern: Illigitimus Non Interruptus	EP14	Teams that finish early accelerate faster … high performing …
194	5	Daily Clean Code	EP14	Teams that finish early accelerate faster … high performing …
195	6	Emergency Procedure	EP14	Teams that finish early accelerate faster … high performing …
196	7	Scrumming the Scrum	EP14	Teams that finish early accelerate faster … high performing …
197	8	Happiness Metric	EP14	Teams that finish early accelerate faster … high performing …
198	9	Team that Finish Early Accelerate Faster	EP14	Teams that finish early accelerate faster … high performing …
199	1	Less is More!	EP16	Patterns for integrating agile development processes and UCD
200	2	Usability Testing Sessions Alongside Agile Development Tests	EP16	Patterns for integrating agile development processes and UCD
201	3	Do it RITE! (Rapid Iterative Testing and Evaluation)	EP16	Patterns for integrating agile development processes and UCD
202	4	Developers as UCD Practitioners	EP16	Patterns for integrating agile development processes and UCD
203	5	Websites as Lightweight Documentation	EP16	Patterns for integrating agile development processes and UCD
204	1	Scrum of Scrum Pattern	EP17	Distributed Agile Patterns for Offshore Software Development
205	2	Local Stand-up Meetings Pattern	EP17	Distributed Agile Patterns for Offshore Software Development
206	3	Follow the Sun Pattern	EP17	Distributed Agile Patterns for Offshore Software Development
207	4	Onshore Review Meeting	EP17	Distributed Agile Patterns for Offshore Software Development
208	5	Collective Project Planning	EP17	Distributed Agile Patterns for Offshore Software Development
209	6	Project Charter Pattern	EP17	Distributed Agile Patterns for Offshore Software Development
210	7	Collaborative Planning Poker	EP17	Distributed Agile Patterns for Offshore Software Development
211	8	Global Scrum Board	EP17	Distributed Agile Patterns for Offshore Software Development
212	9	Local Sprint Planning	EP17	Distributed Agile Patterns for Offshore Software Development
213	10	Local Pair Programing	EP17	Distributed Agile Patterns for Offshore Software Development
214	11	Central Code Repository	EP17	Distributed Agile Patterns for Offshore Software Development
215	12	Asynchronous Retrospective Meetings	EP17	Distributed Agile Patterns for Offshore Software Development

216	13	Asynchronous Information Transfer	EP17	Distributed Agile Patterns for Offshore Software Development
217	14	Synchronous Communication	EP17	Distributed Agile Patterns for Offshore Software Development
218	15	Visit Onshore-Offshore Teams	EP17	Distributed Agile Patterns for Offshore Software Development
219	1	Measure Economic Progress	EP18	Agile base patterns in the agile canon
220	2	Proactiely Experiment	EP18	Agile base patterns in the agile canon
221	3	Limit Work-in-process	EP18	Agile base patterns in the agile canon
222	4	Collective Responsibiity	EP18	Agile base patterns in the agile canon
223	5	Solve Systemic Problems	EP18	Agile base patterns in the agile canon
224	1	Big Requirements Documentation	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
225	2	Customer Product Owner	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
226	3	Producto Owner without Authority	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
227	4	Long or Non-existent Feedback Loops	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
228	5	Unordered Product Backlog	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
229	6	Work Estimates Given to Teams	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
230	7	Hour in Progress Monitoring	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
231	8	Semi-functional Teams	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
232	9	Customer Caused Disruption	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
233	10	Business as Usal (No Sprint Retrospective)	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
234	11	Invisible Progress	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
235	12	Varying Sprint Length	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
236	13	Too Long Sprint	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
237	14	Testing in Next Sprint	EP19	Exploring ScrumBut - An empirical study of Scrum anti-patterns
238	1	Team members involvement was different than assigned	EP20	Patterns of collaboration driven by requirements …
239	2	Agile teams are sefl-contained	EP20	Patterns of collaboration driven by requirements …
240	3	Project managers are key players in communication	EP20	Patterns of collaboration driven by requirements …
241	4	Distance does not (seem to) matter	EP20	Patterns of collaboration driven by requirements …
242	5	Bugs discussion, communication of changes, and user story clarification as the most discussed topics	EP20	Patterns of collaboration driven by requirements …
243	6	Familiarity from past projecs facilitaes awareness	EP20	Patterns of collaboration driven by requirements …

244	7	Communication is still an important source of awarenesss	EP20	Patterns of collaboration driven by requirements …
245	1	Unified Product Backlog	EP22*	A program backlog story with patterns … magic ... collection
246	2	Linked Product Backogs	EP22*	A program backlog story with patterns … magic ... collection
247	3	Pragamatic Product Backlogs	EP22*	A program backlog story with patterns … magic ... collection
248	1	Remodel	EP22*	A program backlog story with patterns … magic ... collection
249	2	Rules	EP22*	A program backlog story with patterns … magic ... collection
250	3	Definitions	EP22*	A program backlog story with patterns … magic ... collection
251	1	Answers	EP22*	Patterns to Build the Magic Backlog
252	2	Connections	EP22*	Patterns to Build the Magic Backlog
253	3	Frame	EP22*	Patterns to Build the Magic Backlog
254	4	People	EP22*	Patterns to Build the Magic Backlog
255	5	Placeholders	EP22*	Patterns to Build the Magic Backlog
256	6	Plans	EP22*	Patterns to Build the Magic Backlog
257	7	Tales	EP22*	Patterns to Build the Magic Backlog
258	8	Usage Models	EP22*	Patterns to Build the Magic Backlog
259	9	Views	EP22*	Patterns to Build the Magic Backlog
260	10	Funnel	EP22*	More Patterns for the Magic Backlog
261	11	Maintenance	EP22*	More Patterns for the Magic Backlog
262	12	Pipeline	EP22*	More Patterns for the Magic Backlog

Tabla Anexo B.0.1 - Resumen de patrones por estudio primario

Anexo C. Formato de patrón por estudio primario

En este anexo se recogen los formatos de patrón de los estudios en los que se describen las secciones del formato explícitamente. Se ha realizado una traducción literal de la descripción de las secciones para respetar lo transmitido por los autores y evitar el posible sesgo derivado de la interpretación de la traducción.

EP01 (2004): AntiPractices: AntiPatterns for XP Practices

Name-Background-Sympton-Cause-Ideal-Prescription

- **Name**: Nombre del anti-patrón.
- **Background**: Contexto donde ocurre el síntoma.
- **Symptom**: Fenómenos malos visibles.
- **Cause**: Por qué se produce el síntoma.
- **Ideal**: Lo que realmente debería ser.
- **Prescription**: Contramedidas que detienen el síntoma atacando la causa y acercándola al ideal, incluyendo la prevención.

EP03 (2006): Patterns for Agile Development Practice Part 3 (version4)*

Name-Context-Problem-Forces-Therefore (Solution)-Commentary-Consequences

- **Name**: Nombre del patron.
- **Context**: Oración que indica a quién se dirige el patrón y cuando se puede aplicar en el ciclo.
- **Problem**: Párrafo en el que la oración clave está en cursiva.
- **Forces**: Párrafo en el que se indica qué necesitas considerar para aplicar este patrón. En esta versión pondremos las fuerzas en listas con viñetas.
- **Therefore**: Solución. La oración clave (normalmente la primera) está en cursiva.
- **Consequences**: Aparecen en viñetas.
- **Commentary**: Aparece en párrafos.

EP05 (2007): A Process Pattern Panguage for Agile Methods

Name-Context-Problem-Solution-Consequence-Related Pattern

- **Name**: En general, una palabra o frase significativa y fácil de recordar utilizada como referencia. El nombre del patrón describe la esencia del patrón de manera concisa. Por lo general, consiste en el nombre de la tarea correspondiente que admite un patrón.

- **Context**: Describe el fondo de un problema en un proceso de software. Ayuda al usuario a comprender mejor el problema y la solución, y a elegir el patrón adecuado.
- **Problem**: Describe el problema que necesita ser resuelto en cierto contexto. Comprende Intención y Fuerzas. La intención es un resumen de la función y el objeto del patrón. Las fuerzas son factores objetivos que influyen en una organización o en un proceso de desarrollo. Normalmente hay conflictos entre un grupo de fuerzas. La intención y el contexto componen las instrucciones para que el usuario seleccione patrones: en cierto contexto, creamos una solución para equilibrar las fuerzas con el fin de lograr funciones y objetos en la intención.
- **Solution**: Da instrucciones y sugerencias para resolver el problema y equilibrar las fuerzas. La solución tiene cuatro partes: Artefacto, Rol, Actividad y Regla.
- Los artefactos proporcionan información sobre qué actividades deben realizarse y / o qué producen. El rol se incluye para tomar parte en la solución. La actividad proporciona la especificación para una actividad en la solución. La regla presenta una restricción o directriz que debe seguirse en la solución o actividad
- **Consequence**: Describe los efectos después de aplicar el patrón actual.
- **Related Pattern**: Lista de patrones relacionados con el patrón actual.

La Figura Anexo C.0.1, muestra la relación entre las secciones del patrón mediante un diagrama UML:

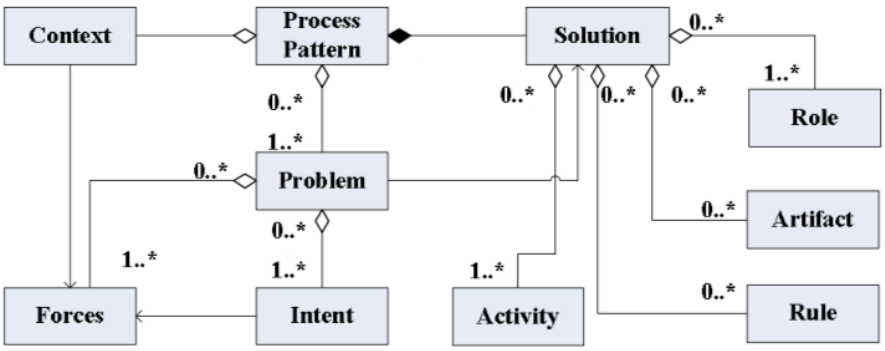

Figura Anexo C.0.1 - Relación entre las secciones del formato de patrón [EP05]

EP06 (2008): Agile Learning & Teaching with Wikis: Building a Pattern

Intent-Motivation (Forces)-Applicability-Structure-Participants-Collaboration-Consequences-Implementacion-Known Uses-Related

El estudio [EP06] en sí es la descripción explicita del formato de patrón.

EP07 (2010): How Much is Just Enough?: Some Documentation Patterns on Agile Projects

Name-Context-Example-Problem-Forces-Solution-Consequences-Example Resolved-Known Uses- Related Practices

Name: El nombre del patrón.
Context: El contexto de uso del patrón.
Example: Un ejemplo de un equipo Agile como introducción de la siguiente sección.
Problem: El problema al que se dirige el patrón.
Forces: Las fuerzas en el juego.
Solution: La solución.
Consequences: Las consecuencias de la solución que resuelve las fuerzas (mostradas con un símbolo positivo) y las que no resuelve (mostradas con un símbolo negativo)
Example Resolved: Un ejemplo resuelto.
Known Uses: Usos conocidos en la investigación.
Related Practices: Prácticas relacionadas.

EP10 (2011): Extending Patterns for Fearless Change*

Name-Alias-Abstract-Opening Story-Context-Problem-Description of the Problem and Forces-Therefore-Essence of the Solution-Resulting Context-Known Uses

Name: El nombre del patrón, en negrita.
Alias: Solo si aplica.
Abstract: El resumen, en negrita.
Opening Story: La historia de apertura que transmite la esencia del patrón, en cursiva.
Context: El contexto de aplicación del patrón.
Problem: El problema que resuelve el patrón, una oración en negrita.
Descriptión of the Problem and Forces: La descripción del problema y las fuerzas.
Therefore: La solución al problema, en cursiva.
Essence of the Solution: La esencia de la solución, en negrita
Elaboration of the Solution: La explicación de la solución.
Resulting Context: El contexto resultante.
Known Uses: Los usos conocidos, en cursiva

EP11 (2011): The Art of Software Estimation Patterns Language

Name-Context-Problem-Forces-Solution-Discussion-Resulting Context

- **Name**: Nombre del patrón
- **Context**: Describe la situación o el entorno donde se puede usar el patrón. Actualmente estás aquí ...

- **Problem**: Describe el problema que intenta resolver o el objetivo que intenta alcanzar el patrón. Estás tratando de llegar a ...
- **Forces**: Describe las influencias e intereses, a menudo en conflicto, que existen dentro del contexto dado el problema.
- **Solution**: Esta es la mejor práctica o la solución recomendada para el problema. Tenga en cuenta que, aunque la solución intenta optimizar las fuerzas, es raro que un solo patrón mitigue todas las fuerzas negativas. Los mejores resultados se logran cuando los patrones se usan de forma sinérgica.
- **Discussion**: Este es un breve resumen y sugerencias para usar el patrón. En esta sección, a menudo encontrará apuntes a patrones que funcionan bien cuando se implementan junto con este patrón (por ejemplo, patrones que resuelven fuerzas que no se abordan en este patrón).
- **Resulting Context**: Este es el entorno o situación en la que se encuentra después de usar el patrón. Las cosas están mejor ahora, pero generalmente no son perfectas y pueden mejorarse aún más (conviene prestar atención a los patrones complementarios en esta sección).

EP12 (2012): A Pattern Language for Agile Software Estimation

Name-Context-Problem-Forces-Solution-Example-Resulting Context-Known Uses-Related Patterns

- **Name**: Nombre descriptivo y único que ayuda a identificar y referirse al patrón.
- **Contexto**: Situación a la que se aplica la solución de patrón y, si corresponde, las condiciones de entrada que deben cumplirse antes de que comience el proceso.
- **Problema**: La declaración del problema es una expresión de lo que el usuario debe hacer, percibir o entender.
- **Fuerzas**: Aspectos que influyen en la elección del patrón.
- **Solución**: Explica cómo se resuelve el problema del usuario a través de un conjunto claro y comprensible de pasos y percepciones. Describa en detalle cómo realizar los pasos / actividades del patrón de proceso.
- **Ejemplo**: Un ejemplo de la aplicación de patrón.
- **Contexto resultante**: Indica la situación / contexto que resultará de realizar la solución del patrón de proceso, incluidas, si corresponde, las condiciones que deben cumplirse para que el proceso se considere completo.
- **Usos conocidos**: Ejemplos de usos reales del patrón.
- **Patrones relacionados**: Otros patrones que tienen alguna relación con el patrón; discusión de las diferencias entre el patrón y patrones similares.

EP18 (2016): Agile Base Patterns in the Agile Canon

Name-Context-Problem-Forces-Solution (Therefore)-Examples-Resulting Context

- **Name**: Nombre del patrón.
- **Context**: Contexto.
- **Problem**: Problema.
- **Forces**: Fuerzas que restringen las soluciones.
- **Solution**: Solución que concilia las fuerzas y mitiga el problema.
- **Examples**: Ejemplos que usan la solución.
- **Resulting Context**: Contexto resultante.

EP19 (2016) Exploring ScrumBut – An Empirical Study of Scrum Anti-Patterns

Name-Scrum Recommendation-Context-Solution-Consequences-Exceptions-Company Recommendations

Nombre: Nombre del patrón.
Scrum Recommendation: La recomendación relacionada con Scrum. Se toma de la guía Scrum [20].
Context: Condiciones especificas bajo las cuales el anti-patrón ocurre típicamente. El contexto se extrae de los resultados de la entrevista, reflejando las explicaciones que los entrevistados dieron de las desviaciones de Scrum.
Solution: La solución anti-patrón (generalmente dañina) o desviación Scrum.
Consequences: Las consecuencias de usar el anti-patrón. Las consecuencias se deben fundamentalmente a los problemas observados por los entrevistados o a los indicados en la guía Scrum [20].
Exceptions: Posibles situaciones en las que el uso del anti-patrón podría estar justificado.
Company Recommendations: Posibles soluciones recomendadas por las empresas que han participado en la encuesta.

Anexo D. Scrum Pattern Language

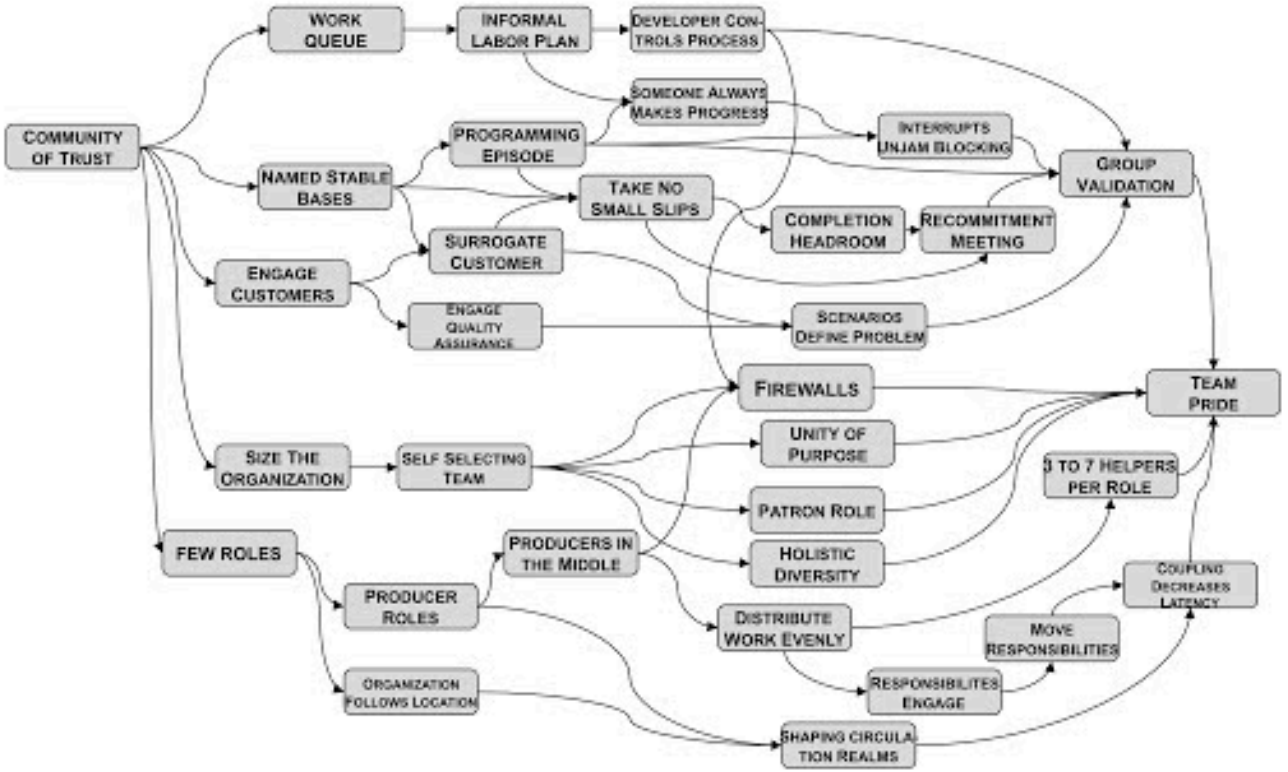

Figura Anexo D.0.1 – Lenguaje de patrones Scrum (ScrumPLoP) [53]