Universidad de Alcalá

Trabajo Fin de Máster

Desarrollo y aplicación de una herramienta de filtro de adecuación ágil

Máster Universitario en Dirección de Proyectos Informáticos

Presentado por:

Dª. ANTONIO HONORATO DE LA PEÑA

Dirigido por:

D. MARIA DEL CARMEN PAGÉS ARÉVALO

Alcalá de Henares, a 16 de septiembre de 2022

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

MÁSTER EN DIRECCIÓN DE PROYECTOS INFORMÁTICOS

TRABAJO FIN DE MÁSTER

DESARROLLO Y APLICACIÓN DE UNA HERRAMIENTA DE FILTRO DE ADECUACIÓN ÁGIL

AUTOR: ANTONIO HONORATO DE LA PEÑA

DIRECTOR: MARIA DEL CARMEN PAGÉS ARÉVALO

TRIBUNAL:

PRESIDENTE:

VOCAL 1º:

VOCAL 2º:

CALIFICACIÓN: **FECHA:**

RESUMEN

Las metodologías Agile son cada vez más predominantes en el mundo empresarial. Todas las empresas y equipos dedicados al desarrollo del software están migrando sus organizaciones a este tipo de metodologías debido a la evolución del mundo tecnológico, cada vez más rápido.

Por ello, este trabajo de fin de máster (TFM) desarrolla una aplicación que sirve para medir la adaptación a los procesos instaurados por las metodologías agiles. La aplicación consiste en un cuestionario de 27 preguntas, de varias categorías, que permite observar de una manera cómoda para el usuario que puntos definidos por las metodologías agiles tiene instaurados en su organización y cuales necesitan mejorar en caso de querer migrar la forma de organización a metodologías Agiles. Estos puntos de acierto y error se miden gracias a la recopilación de varias teorías agiles ya conocidas y extendidas por todo el mundo.

Palabras clave:
Agile, filtros de adecuación, aplicación web

ABSTRACT

Nowadays, technology world is changing exponentially, bringing with those improvements in almost every product. To fit in this competitive and challenging world, companies need to adapt the organization of his developer team, with the objective of take-out products faster than the other companies. In this context, the best organization method for business are the Agile methodologies, which leads to the business to migrate his organization to these methodologies.

To facilitate this migration, this project develops an application that measure the Agile's adaptation grade. The app is formed by a 27-question quiz that allows the user to see which categories of the Agile methodologies are established in his company and which needs to improve, due to a compilation of some of the most known Agile theories.

Keywords:
Agile, suitability filters, web application

ÍNDICE DE CONTENIDO

RESUMEN 4
ABSTRACT 5
ÍNDICE DE CONTENIDO 6
ÍNDICE DE FIGURAS 9
ÍNDICE DE TABLAS 11
1. INTRODUCCIÓN 13
1.1 Marco socioeconómico 13
1.2 Objetivos 14
1.3 Estructura 15
2. ESTADO DEL ARTE 17
2.1 Metodologías Tradicionales 17
2.1.1 Historia 17
2.1.2 Modelo en cascada 18
2.1.3 Ventajas e inconvenientes 19
2.1.4 Cuando utilizar metodologías tradicionales 21
2.2 Metodologías Agile 22
2.2.1 Historia 22
2.2.2 Manifiesto Agile 23
2.2.3 Ventajas y desventajas 26
2.2.4 Cuando utilizar metodologías Agile 27
2.3 Filtros de adecuación Agile 28
2.3.1 Filtros existentes 28
2.3.1.1 DSDM Suitability Filter 28
2.3.1.2 Crystal Team Size Factors 30
2.3.1.3 Boehm and Turner – Radar Chart 32

2.3.1.4 Dave Cohen's Agile Factors 34
2.3.1.5 Organizational Suitability Filter 34
2.3.1.6 PMI: Guía de práctica ágil 35
3. DESARROLLO DEL FILTRO DE ADECUACIÓN 37
3.1 Herramientas utilizadas en el desarrollo. Manual de Usuario 37
3.1.1 Guía de instalación 38
3.1.1.1 Programas que instalar 38
3.1.1.2 Ejecución con IDE 39
3.1.1.3 Ejecución sin IDE 41
3.2 Desarrollo del cuestionario 41
3.2.1 Asignación de resultados 43
3.2.2 Asignación teoría-cuestión y cálculo de porcentajes 45
3.2.3 Creación de figuras 45
3.3 Estructura de la aplicación 47
3.3.1 Vista de cuestionario 47
3.3.2 Vista general 49
3.3.3 Vista de teoría 50
4. RESULTADOS DEL FILTRO 54
4.1 Empresa 1 54
4.2 Empresa 2 57
4.3 Empresa 3 60
4.4 Conclusiones después de la utilización del filtro 63
5. CONCLUSIONES 64
5.1 Posibles usos de la aplicación y mejoras 65
5.2 Planificación y presupuesto 65
6. BIBLIOGRAFÍA 70
ANEXO I: TABLAS DE PROYECTO 72

ÍNDICE DE FIGURAS

Fig 1 Metodología de desarrollo utilizada por las organizaciones [1] 14
Fig 2 Modelo en cascada [3] 18
Fig 3 Evolución de las metodologías Agile 23
Fig 4 Crystal Families [8] 31
Fig 5 Crystal families adaptadas al Agile 32
Fig 6 Grafico de Boehm y Turner [12] [11] 33
Fig 7 Gráfico de Organizational Suitability Filter 35
Fig 8 Logo de Apache Tomcat 39
Fig 9 Ventana para añadir servidor en Eclipse 40
Fig 10 Ventana de Eclipse para importar archivos .war 40
Fig 11 Sección Desplegar de Tomcat 41
Fig 12 Código de creación de gráfico 46
Fig 13 Gráfico radar creado 46
Fig 14 Cabecera y título del cuestionario 47
Fig 15 Preguntas del cuestionario 48
Fig 16 Última pregunta y botón "Ver Madurez Agile" 48
Fig 17 Representación del porcentaje general 49
Fig 18 Representación del resumen de las teorías 49
Fig 19 Recuadrado de porcentaje de cumplimiento 51
Fig 20 Gráfico de Crystal Systems [11] 51
Fig 21 Gráfico de categoría 52
Fig 22 Categorías y principios por categoría 53
Fig 23 Vista de Crystal Systems al 100% con puntuación 53
Fig 24 Vista general de los resultados de la empresa 1 54
Fig 25 Detalle de Crystal Systems en la empresa 1 55
Fig 26 Radar Boehm y Chart para la empresa 1 56
Fig 27 Gráfico de Organizational Filter para la empresa 1 56
Fig 28 Vista general para la empresa 2 57
Fig 29 Ejemplo de DSDM para empresa 2 58
Fig 30 Crystal Clear para empresa 2 59
Fig 31 Radar de Boehm y Turner para empresa 2 59
Fig 32 Gráfico de Organizational Suitability para la empresa 2 60
Fig 33 Vista general de la empresa 3 61
Fig 34 Gráfico radar de Boehm y Turner para la empresa 3 62

Fig 35 Diagrama de Gantt (1/3) 66
Fig 36 Diagrama de Gantt (2/3) 67
Fig 37 Diagrama de Gantt (3/3) 67

ÍNDICE DE TABLAS

TABLA 1 Doce principios del Manifiesto Agile 25
TABLA 2 Preguntas del cuestionario 43
TABLA 3 Preguntas con opciones diferentes 44
TABLA 4 Teoría y descripción en la aplicación 50
TABLA 5 Planificación del proyecto 66
TABLA 6 Costes de mano de obra 68
TABLA 7 Coste del material 68
TABLA 8 Coste indirecto 69
TABLA 9 Coste total del proyecto 69
Anexo I Puntuaciones de las empresas 74
Anexo II Relación cuestión - filtro de adecuación 77
Anexo III Relación cuestión - principio DSDM 80
Anexo IV Relación cuestión - principio Boehm-Turner 82
Anexo V Relación cuestión - principio de Dave Cohen Agile Factors 84
Anexo VI Relación cuestión - principio de Organizational Suitability Filter 86
Anexo VII Relación cuestión - principio de la guía ágil PMI 88

1. INTRODUCCIÓN

1.1 Marco socioeconómico

Desde finales del Siglo XX y sobre todo principios del Siglo XXI, estamos experimentando como sociedad una evolución exponencialmente rápida. Sirve como ejemplo el caso del teléfono, que ha pasado de ser un aparato meramente útil para hacer llamadas de voz a poder integrar aplicaciones de todos los tipos en únicamente 15 años (si cogemos como referencia el lanzamiento del primer *iPhone*).

Uno de los inventos que ha llevado a esta evolución tan veloz es internet, que, entre otras muchas cosas, ha hecho que todo el mundo pueda estar conectado con cualquier otra persona sin importar su localización (el término acuñado para esto es globalización), por lo que las noticias de todo el mundo pueden llegar a cualquier punto casi en el mismo momento en el que se han producido.

Esta globalización ha supuesto también que el mercado capitalista que define la sociedad sea mucho más voraz, ya que marcas de todo el mundo compiten entre ellas sin importar su origen, y, por consiguiente, los clientes disponen de una gran cantidad de catálogo de un mismo producto. Debido a esto, las empresas buscan seducir a estos clientes con productos mejorados y fácilmente identificables de diferentes maneras, como pueden ser la gran calidad del producto, su bajo precio, o sus mejoras con respecto a la competencia. Por lo tanto, las empresas necesitarán que sus productos sean siempre punteros, y necesitarán que la puesta en producción de los desarrollos de los productos sea lo más rápida posible.

Para ayudar en estas tareas, nacieron las llamadas metodologías agiles, que exponían una manera de organizar los equipos de desarrollo de software para que estos pudieran ser más rápidos en puestas a producción y flexibles que las metodologías tradicionales.

En estos momentos no hay casi ninguna empresa dedicada al desarrollo del software en la que no se escuche la palabra “Scrum”, “Kanban” o “*daily*” (términos usados por las principales metodologías ágiles). Según el estudio de Hewlett Packard Enterprise “Agile is the new normal” [1], dos tercios de las empresas encuestadas en el propio estudio utilizan agile puro o una aproximación a estas metodologías:

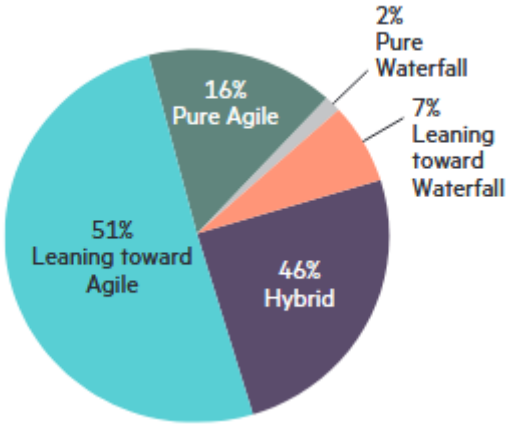


Fig 1 Metodología de desarrollo utilizada por las organizaciones [1]

Aun así, al contrario de lo que mucha gente piensa, estas metodologías ágiles no encajan en todas las organizaciones ni en todos los equipos de desarrollo, ya sea por las propias características del proyecto, por la propia cultura de la organización o más parámetros. Muchas empresas enuncian que utilizan el agile por sentirse puntera en el ámbito del desarrollo del software (ya que las empresas más importantes del mundo utilizan Agile) y no por la propia práctica de la metodología.

1.2 Objetivos

Por ello, este TFM tratará de proporcionar una visión a las empresas de si están implementando bien las metodologías ágiles, o, por el contrario, no es aconsejable que se utilicen estos tipos de metodología.

En conclusión, el objetivo de este trabajo es crear un filtro de adecuación al Agile para equipos de desarrollo funcionales, en el que se desarrollen y justifiquen distintas evidencias para poder emitir un juicio.

Para conseguir este objetivo, se han planteado objetivos intermedios, que ayudarán a su consecución:

- Desarrollo de aplicación del filtro de adecuación: El filtro de adecuación se hará en una aplicación web, que contendrá varias vistas y lógicas, por medio del patrón Modelo-Vista-Controlador.

- Definición y concreción de las cuestiones del filtro de adecuación: Para poder emitir un juicio con respecto a lo expuesto por las diferentes empresas, dentro de la aplicación se creará un cuestionario que contenga enunciados con contenidos clave de los distintos principios Agile.

- Definición de los filtros de adaptación Agile ya existentes: Para refutar y justificar las respuestas obtenidas en el cuestionario, se expondrán los principios de algunas de las principales teorías y filtros de adecuación agiles ya existentes.

1.3 Estructura

La memoria de este proyecto estará dividida en 3 grandes apartados:

- Introducción: En este capítulo se dará información general sobre todo lo que se va a contar a continuación, poniendo en contexto la realización del proyecto y ahondando en los objetivos que se marcan. En este capítulo también se incluye un pequeño resumen sobre esta memoria, explicando los puntos más significativos.

- Estado del Arte: En este apartado se hablará sobre las tecnologías utilizadas durante el proyecto y se justificará su misión en él. Principalmente, este apartado irá destinado a las metodologías Agile, haciendo un análisis de sus ventajas y desventajas en comparación con las metodologías tradicionales, además de distinguir los casos en los que elegir una metodología y otra. Por otra parte, se describirán los filtros de adecuación que ya existen y son utilizados por el nuestro para hacer el cuestionario.

- Desarrollo del proyecto: Se explicará cómo se ha desarrollado el filtro de adaptación, la estructura de la aplicación web y se dará justificación a las distintas soluciones que se han ido aplicando para la consecución de los diferentes hitos marcados.

- Resultados del filtro: En este apartado se analizarán resultados reales logrados después de que tres empresas de diferentes características rellenaran el filtro de adecuación creado.

- Conclusiones: Al final, se establece un apartado a modo de conclusión, donde, además de los posibles usos y las mejoras futuras de este proyecto, se detallan la planificación y los costes que ha derivado dicho estudio.

2. ESTADO DEL ARTE

2.1 Metodologías Tradicionales

2.1.1 Historia

Se conoce como metodologías tradicionales a aquellas que son más primigenias en el desarrollo de software, apareciendo a la vez que este, y aportando técnicas de organización a lo que era el desarrollo del software, aún muy primitivo, en la segunda mitad del siglo XX. Existen varias etapas en lo que es el desarrollo tradicional, según Rogerio y Egas [2]:

- Una primera etapa en los años 50, en la que no existían metodologías como tal, sino que los desarrolladores eran los encargados de saber las necesidades de los usuarios y recogerlas en el código.

- En la década de los 60, empezaron las primeras técnicas de organización de equipos, aunque mucho más enfocados en los procesos que en la propia organización. Todavía el desarrollo del software era un trabajo de una persona, y cada uno era responsable de su código.

- En los años 70 ya se presenció que esta manera utilizada en la década de los 60 no era la correcta. Existió lo que se definió como la “Crisis del Software”, marcada por el tremendo coste y la poca fiabilidad que tenían los desarrollos de software de aquella época, derivados de la forma de organización que se tenía. Por esto, la manera de pensar en cuanto a estructura cambió radicalmente y se empezaron a originar conceptos aún existentes como el ciclo de vida del software.
  Además, surge el primer concepto de metodología, llamada por Winston Royce modelo “en cascada” o” waterfall”, técnica que ha llegado hasta estos días y aún define lo que es el modelo tradicional.

2.1.2 Modelo en cascada

El nombre de modelo en cascada viene debido a que la metodología se divide en pasos indivisibles y secuenciales, haciendo un esquema similar a una cascada, como el que se muestra a continuación:

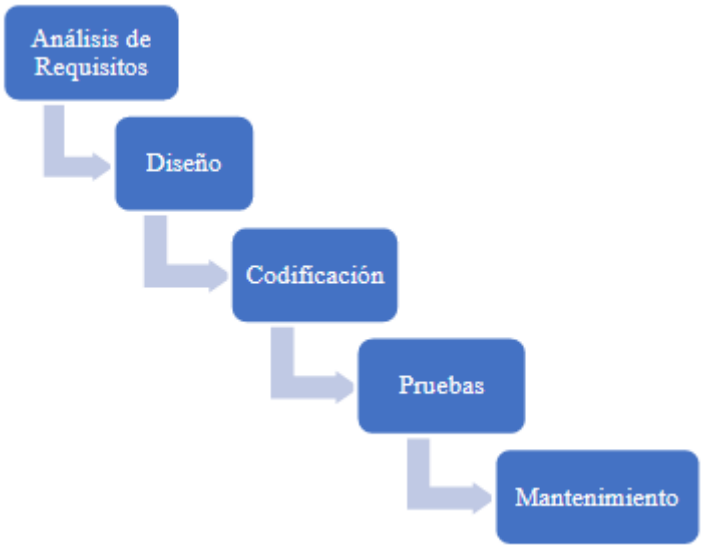


Fig 2 Modelo en cascada [3]

Como se observa en la figura, las fases van de uno en uno y según la metodología, no se pueden saltar. En el caso en el que falle alguno de estos pasos, se debería parar el proceso de desarrollo hasta su resolución. Los detalles de los pasos de la metodología son [4]:

- **Análisis de requerimientos**: Esta fase consiste en que los desarrolladores entiendan las exigencias de los clientes, sabiendo que es lo que se necesita, además de su función en el proyecto. Si estos requerimientos son entendidos por parte del equipo de desarrollo, se pasa a la siguiente fase y se genera un documento de requisitos en los que estos se listan.

- **Diseño**: En esta fase el equipo de desarrollo se reúne para planificar cómo hará la funcionalidad pedida por los clientes, teniendo los requisitos salientes del análisis de requerimientos. No es raro que durante esta etapa surjan más requerimientos que no se hayan recogido durante la primera etapa, por lo que

en este caso se podrá volver a la fase de análisis de requisitos y empezar de nuevo el proceso.

- **Creación de código**: El equipo comienza a desarrollar la funcionalidad en esta fase. La dificultad de la creación de la funcionalidad dependerá de lo logrados y acotados que estén los requisitos propuestos en el diseño de las anteriores etapas, haciendo así que la interpretación del desarrollador sea la mínima posible, factor que suele reducir al máximo el número de posibles fallos.

- **Pruebas**: Una vez terminado la creación del código se pasa a la fase de testeo de la funcionalidad. Las pruebas de esta funcionalidad estarán delimitadas a los requisitos propuestos en las primeras fases, y si existe algún bug, se volverá a la fase de diseño para poder llevar a cabo la resolución de dicho error. Hasta que no estén todos los errores solucionados, no se podrá salir de esta fase a la última de mantenimiento.

- **Mantenimiento**: Esta última fase del modelo en cascada. En esta fase el cliente verá la nueva funcionalidad y dará su visto bueno. Será tarea de los desarrolladores chequear que esta funcionalidad siga funcionando tal y como se especificó. Puede pasar que el cliente sugiera añadir mejoras a la funcionalidad, que se tratará como un proceso independiente y, por tanto, se volverá a empezar de nuevo todo desde el análisis de requisitos.

2.1.3 Ventajas e inconvenientes

Una vez explicado en que consiste la metodología en cascada, podemos analizar sus ventajas y sus desventajas. A continuación, se listan las ventajas del modelo en cascada [5]:

- **Estructura sencilla y clara**: Como se ha comentado anteriormente, el modelo en cascada tiene unas características tan herméticas que hace que la estructura del proyecto sea clara desde el primer momento. Por otra parte, las fases están tan diferenciadas que es muy fácil saber en qué fase del proyecto se está.

- **El objetivo final es claro**: Determinar el objetivo final del proyecto es sencillo ya que se determina al inicio. Esto permite al desarrollador estar centrado en el objetivo final durante todas las fases del proyecto.

- **Documentación en todos los procesos de desarrollo**: Tener las fases tan especificadas requiere tener documentos de todo el proceso de desarrollo para constatar el paso de fase en fase. Por ejemplo, esta característica puede hacer que ordenar cronológicamente el orden de las fases sea muy fácil de sacar.

- **Estimación sencilla**: Al ser un proyecto cerrado y en el que no se permite que haya cambios, es bastante sencillo estimar los costes del proyecto y la carga de los trabajadores, así como hacer la planificación al inicio del proyecto.

Por otra parte, también existen ciertos inconvenientes a la hora de plantear este tipo de metodologías, estas son:

- **No da lugar a cambios**: El principal problema del modelo en cascada es que no permite que existan cambios en las funcionalidades ya definidas. Esto es un gran inconveniente ya que, como se ha comentado en anteriores apartados, la evolución rápida de los productos puede hacer que cuando se termine el producto este haya quedado ya desactualizado.

- **El cliente final no tiene impacto en la metodología**: Por esta falta de reacción ante cambios, el cliente final queda en un lugar residual con respecto al desarrollo del producto, interviniendo solo en el análisis de los requisitos y viendo el producto final. Esto suele ser un inconveniente ya que, si hay algo en el producto que no le guste, se pierde demasiado tiempo cuando, habiéndolo involucrado antes, se podría haber evitado.

- **Las pruebas se hacen sobre el producto final**: Hacer pruebas sobre el producto final, a finales del proceso de desarrollo es arriesgado, ya que un error en la base del desarrollo puede hacer que todo lo posterior desarrollado sea inútil.

- **Resultados para el cliente a muy largo plazo y muy difíciles de comprender**: Para el cliente los resultados se ven al final de la última etapa, que puede

alargarse a meses, lo que puede hacer que se sientan incomodos con su inversión y pierdan la paciencia. Además, al cliente se le da el resultado final del producto, perdiéndose así este todo el proceso de desarrollo y, por tanto, teniendo muchas dudas en el que va a ser su producto.

2.1.4 Cuando utilizar metodologías tradicionales

Aunque sean modelos ya antiguos, la metodología en cascada tiene una estructura que hace que sea idónea para tipos de proyectos con características determinadas. En este apartado, se va a intentar detallar los tipos de rasgos que deben tener los proyectos para que el modelo en cascada sea el más idóneo de implementar:

- **Proyectos con pocos cambios**: El hecho de que los cambios no sean posibles en la metodología acota el uso de esta metodología a proyectos ya muy instalados, en la que la proporción de cambios durante las fases del proyecto sea mínima.

- **Proyectos en el que el usuario no quiera estar involucrado**: Los proyectos creados para usuarios que, por características de su empresa, sean más invisibles en cuanto al proyecto. La no interacción del cliente hace que la metodología en cascada funcione ya que no necesita de esta dentro de sus estándares, solo en la fase inicial y final.

- **Proyectos simples**: Los proyectos con pocas herramientas harán que la definición de fases sea más fácil y por tanto ayudará a la implementación de esta metodología. Hay que tener en cuenta que, si el proyecto es multidisciplinar, todas las disciplinas deben estar en la misma fase, lo que puede llevar a retardos indirectos en las fases. Además, cuanto más sencillo sea el proyecto menos errores se encontrarán, lo que hará que el flujo termine mucho antes que en los proyectos más difíciles de afrontar.

Cabe destacar que, aunque esta metodología en cascada parece ya acabada y antigua, existen empresas que han seguido utilizando este método. Un ejemplo es la compañía Toyota, que empezó su transición a empresa IT con esta metodología (como es conocido, antes únicamente era una compañía únicamente dedicada a la

automoción). Esta metodología sirvió para sellar las bases de la organización IT y a partir de ahí poder crecer hacia metodologías más adecuadas para el mercado actual, como son las metodologías Agile [6] .

2.2 Metodologías Agile

Como se comentó en los anteriores apartados, la evolución de la sociedad y del mercado hizo que las empresas empezaran a necesitar otro tipo de metodología diferente a los modelos tradicionales existentes. Esta necesidad hace que nazcan lo conocido como metodologías agiles.

2.2.1 Historia

El término “Metodología Agile” fue propuesto en 2001 por diecisiete académicos en las montañas de Utah (EE. UU.) que crearon la “Agile Alliance”, organización sin ánimo de lucro que quería seguir evolucionando el desarrollo de software, proponiendo nuevas formas de trabajar. Para esto, crearon el “Manifiesto Agile” [7], una serie de cuatro valores y doce principios que resumen las bases de lo que ellos creían que era el mejor modelo de desarrollo de software. Se explicará mejor que es el manifiesto Agile en el próximo apartado.

Aun así, y al contrario de lo pensado, existían teorías dentro de lo conocido como metodología Agile que existían antes de que se acuñara este término. Estas teorías empiezan a aparecer a principio de la década de los 90 con la familia de Crystal Clear creada por Alistair Blackburn, empezando así la ruptura con las metodologías antiguas [8].

Una de las metodologías más antiguas consideradas como Agile es a la vez una de las más utilizadas, que es la metodología Scrum, en 1991. Scrum se ha convertido en una metodología tan extendida dada a su flexibilidad, que hace que sea muy fácil para las organizaciones estructurarse ante cambios no previstos dentro de los proyectos. Muchas de las palabras conocidas del Agile como son “Sprint”, “Historia de Usuario” o “Product Owner” vienen de esta metodología.

Ese mismo año aparece otra metodología que se utilizará como es DSDM (Dynamic Systems Development Method).

Al año siguiente aparece la metodología eXtreme Programming (XP) formulada por Kent Beck (posteriormente, uno de los firmantes del Manifiesto Agile). Esta metodología se caracteriza sobre todo por adaptarse al máximo a los requerimientos de los clientes y al desarrollo del proyecto, más que cualquier otra metodología Agile, lo que le hace, hoy en día, una de las metodologías más utilizadas junto a Scrum.

La más moderna de las metodologías Agile más populares que forman este marco es la conocida como Lean, del 2003, ya posterior a la creación del manifiesto Agile. Del Lean viene relacionada otro de los métodos más utilizados por todas las empresas, el Kanban. La idea del Kanban es tener una visión general de cómo es el estado las pequeñas funcionalidades y tareas en las que se puede dividir un proyecto.

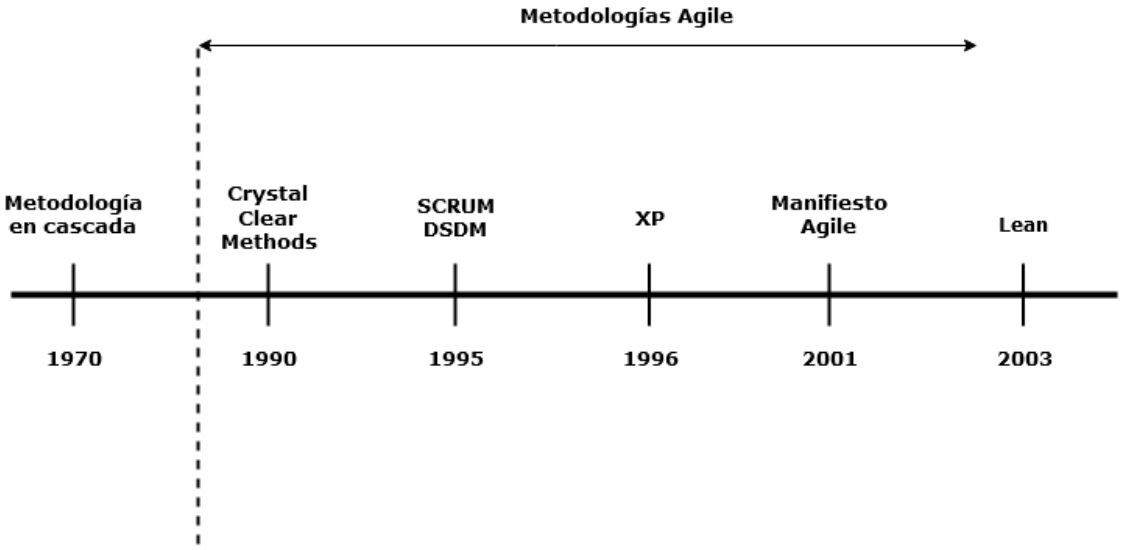


Fig 3 Evolución de las metodologías Agile

2.2.2 Manifiesto Agile

La creación del manifiesto Agile en 2001 supuso una revolución en el mundo del desarrollo del software, ya que puso en el panorama un enfoque totalmente contrario a las metodologías tradicionales que utilizaban las empresas. Los firmantes del manifiesto pensaban que estas metodologías tradicionales ya no cumplían las necesidades que tenían las empresas, por lo que era necesaria un cambio en la mentalidad de las organizaciones de las empresas.

Para ello, definieron 4 valores sobre los que las empresas deberían estructurar su organización. Estas son:

- **Individuos e interacciones sobre procesos y herramientas**: La organización debe estar centrado en el equipo de trabajo más que en una herramienta.
- **Software funcionando sobre documentación extensiva**: Las metodologías en cascada necesitaban de elaborar documentos entre fase y fase, lo que para los firmantes del manifiesto es excesivo, y es preferible trabajar en el desarrollo del software.
- **Colaboración con el cliente sobre negociación contractual**: El cliente forma parte esencial del proyecto. En las metodologías tradicionales, este tiene un papel residual.
- **Respuesta ante el cambio sobre seguir un plan**: Los proyectos deben poder responder ante cambios.

Además, de estos cuatro valores surgen 12 principios, definidos en la siguiente tabla:

12 principios del Manifiesto Agile	
I.	Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
II.	Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
III.	Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
IV.	Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.

V.	Los proyectos se desarrollan en torno a individuos motivados. Hay que darles el entorno y el apoyo que necesitan, y confiarles la ejecución del trabajo.
VI.	El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
VII.	El software funcionando es la medida principal de progreso.
VIII.	Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
IX.	La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
X.	La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
XI.	Las mejores arquitecturas, requisitos y diseños emergen de equipos autoorganizados.
XII.	A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia .

TABLA 1 Doce principios del Manifiesto Agile

2.2.3 Ventajas y desventajas

El manifiesto Agile supuso que muchas de las empresas empezaran a considerar las metodologías que hacían gala de los valores y principios que el manifiesto enunciaba. Los valedores recomiendan el uso de las metodologías Agile debido a que:

- Permite que el empleado pueda centrarse en tareas pequeñas por su divisibilidad y, por tanto, hacer más rápido el trabajo. En las metodologías tradicionales, únicamente se tiene una fase de desarrollo en la que no se sabe que están haciendo los trabajadores.

- La división por etapas pequeñas de desarrollo hace que el proyecto sea totalmente adaptable a cambios y acomoda la organización de las personas del equipo. En la metodología en cascada no se admitían cambios y en caso de haber, había que volver a empezar el ciclo.

- La división en tareas más pequeñas permite identificar las tareas más importantes o que más tiempo van a conllevar, y estimar su riesgo y su planificación.

- Quizá el punto más fuerte del Agile es la estrecha relación entre equipo y cliente. Al dividir por etapas pequeñas, la puesta en producción tiende a ser bastante menor que en organizaciones en cascada. Además, hace que el cliente pueda hacer que su inversión se vea reflejada antes que, en otras organizaciones, así como sugerir mejoras y cambios sin esperar a que el producto esté finalizado.

Sin embargo, siguen existiendo detractores de estas metodologías Agile, que siguen pensando que las metodologías Agile no sirven para todos los tipos de proyectos por sus características. Las desventajas que ven de las metodologías Agile son [9]:

- La evaluación de riesgos es compleja, ya que no se tiene una versión global del proyecto y los posibles cambios pedidos por cliente, por lo que la planificación del cierre del proyecto en etapas tempranas es muy difícil.

- Existen proyectos que no tienen muchos cambios en largos plazos. Por ello, es una pérdida de tiempo para los trabajadores la realización de todas las reuniones que conllevan las metodologías Agile (por ejemplo, en Scrum existen las reuniones de la *daily*, *planning*, *sprint review*, etc.).

- Depende mucho de la relación con el cliente. Pueden existir situaciones en las que el cliente quiera desentenderse del desarrollo del producto, debido a no conocer el alcance exacto de este. En este punto, este método de organización puede hacer que el cliente se sienta en una situación comprometida o sin embargo crea que la empresa no está capacitada para hacer el trabajo asignado.

2.2.4 Cuando utilizar metodologías Agile

Vistas las características de las metodologías Agile, queda claro que proponen cosas totalmente diferente a lo que enunciaban las metodologías tradicionales. Por tanto, en contraposición, es conveniente utilizar metodologías Agile en proyectos con las siguientes características:

- Proyectos muy cambiantes: El Agile es perfecto para proyectos donde haya constantes cambios de alcance dentro de las funcionalidades o los requisitos no sean muy específicos. Agile permite planificar sobre la marcha.

- Proyectos con mucha relación con el cliente: Con Agile, el cliente tiene la posibilidad de estar enterado en todo momento de la situación del proyecto, así como la potestad de sugerir cambios en medio del desarrollo.

- Proyectos complejos: Al implementar la división de las funcionalidades en pequeñas tareas, las metodologías ágiles son perfectas para proyectos complejos, ya que se acaban convirtiendo en una acumulación de tareas simples.

En conclusión, con ambas metodologías se entiende la evolución que ha tenido el mundo del desarrollo del software hasta, interpolando la sociedad del trabajo en el mundo. Pasando de una sociedad que entendía el trabajo como una serie de procesos

definidos (como las metodologías tradicionales) a una sociedad en la que el trabajo se entiende como una adaptación del trabajo a los recursos que se tienen.

2.3 Filtros de adecuación Agile

Una vez definida cuales son los tipos de metodologías existentes, y vistos cuales son las características de los posibles proyectos que se relacionan con cada tipo, la siguiente pregunta que se deberían hacer los responsables de los proyectos sería: Si mi proyecto no cumple a rajatabla estas características, ¿cómo puedo saber qué tipo de metodología puedo establecer en mi empresa? Para ello, se utilizan los conocidos como filtros de adecuación, concepto que se desarrollara en este capítulo.

En resumen, un filtro de adecuación Agile no es más que una herramienta que sirve para estimar como de adaptada está una organización a las características propias de las metodologías ágiles. Estos pueden estar definidos de diferentes formas, aunque la principal es en forma de cuestionarios que las empresas rellenan para ver su nivel de adecuación.

2.3.1 Filtros existentes

A lo largo de la existencia de las metodologías ágiles, varios teóricos han creado filtros de adaptación para tratar de ayudar a las empresas en la transformación de su organización desde las metodologías tradicionales a las ágiles. Unos ejemplos existentes y mayormente utilizados por las empresas son:

2.3.1.1 DSDM Suitability Filter

Como se vio en el apartado de historia del Agile, DSDM es en realidad una metodología Agile existente, aunque dentro de ella se han construido filtros de adaptación.

DSDM (Método de desarrollo de sistemas dinámicos) fue desarrollada en el año 1995 por un consorcio de empresas y expertos en desarrollo de software ingleses a partir del RAD (desarrollo rápido de aplicaciones), un método de desarrollo de software a medias entre las metodologías Agile y tradicionales, que abogaba por un desarrollo de software basado en prototipos que entendieran 3 bases de todo desarrollo software: Usabilidad, Utilidad y velocidad de ejecución. Así pues, el principio más básico de la metodología DSDM es que el tiempo y los recursos deben ser ajustados al máximo [10] para poder satisfacer los principios del manifiesto Agile.

El consorcio DSDM, además de crear las bases, creó dos filtros de adecuación para que a las empresas interesadas en poner en práctica la metodología les fuera más fácil la integración. Estos dos tipos de filtros fueron creados en forma de cuestionario: el *Project Suitability Filter* y el *Organizational Suitability Filter.* Como sus propios nombres indican, estos dos filtros están desarrollados para adaptar el proyecto y la organización de la empresa al DSDM, aunque esto no excluye estos dos parámetros de los filtros contrarios.

En este primer apartado detallaremos el *Project Suitability Filter*, que trata de chequear que el proyecto cumpla las siguientes características propias de DSDM [11]:

- **Aceptación de la filosofía Agile antes de empezar el proyecto**: Los principios que surgen del manifiesto Agile deben estar definidos en la organización antes del comienzo del proyecto.

- **La toma de decisiones está en los usuarios y en los desarrolladores**: Es decir, la tecnología se basa en el empoderamiento de los equipos de desarrollo.

- **Compromiso de la alta dirección de hacer partícipes a los usuarios finales**: Los usuarios finales son parte fundamental del desarrollo del proyecto.

- **Acceso de los desarrolladores a entornos de producción**: Dentro del empoderamiento de los equipos de desarrollo, tendrán acceso a entornos de producción, donde antes solo tenían acceso la alta dirección y los usuarios.

- **Estabilidad del equipo**: Mantener el equipo de desarrollo lo más estable posible hace que se tenga mayor conocimiento de la aplicación y, por tanto, no sea necesaria ninguna documentación.

- **Mayores conocimientos en el equipo**: Se prefiere poca gente con más conocimiento del proyecto que mucha gente con poco conocimiento. Esto ayuda a la comunicación del equipo.

- **Menor tamaño del equipo**: Como hemos comentado en el punto anterior, DSDM propone ajustar al máximo los integrantes del equipo al proyecto, ya que esto ayudará a la comunicación entre desarrolladores y la fluidez del trasvase de conocimientos.

- **Tener una buena relación comercial con el cliente**, por encima de los contratos pactados y las negociaciones.

- **Tecnología avanzada en el desarrollo**: Se prefieren usar tecnologías que ayuden al desarrollo incremental, a la rápida puesta en producción o a la refactorización.

2.3.1.2 Crystal Team Size Factors

La metodología Crystal fue desarrollada por Alistair Cockburn, encargado por la empresa IBM a principios de la década de los 90. En el proceso de creación, esta metodología tenía el objetivo de ser orientada para proyectos basados en objetos, pero en esta construcción Alistair se dio cuenta de que todos los proyectos que eran exitosos seguían los mismos patrones.

En su análisis, concluyó que el tamaño del proyecto será más grande cuanta más complejidad tenga el proyecto. Por ejemplo, concluyó que existían equipos de trabajo con menos integrantes, que no necesitaban de documentación, mientras que los proyectos más grandes necesitaban mucha documentación.

Por esto, resumió que se podían agrupar las categorías de los proyectos dependiendo de tres factores: el tamaño del equipo, y la criticidad y prioridad del proyecto. Dependiendo de estas tres características permitió categorizar los proyectos por lo que se conoció como familias Crystal. Cabe destacar que los parámetros criticidad y prioridad del producto se resumen en la metodología en una única variable que es impacto ante fallos. Los distintos tipos de familias se resumen en el siguiente gráfico:

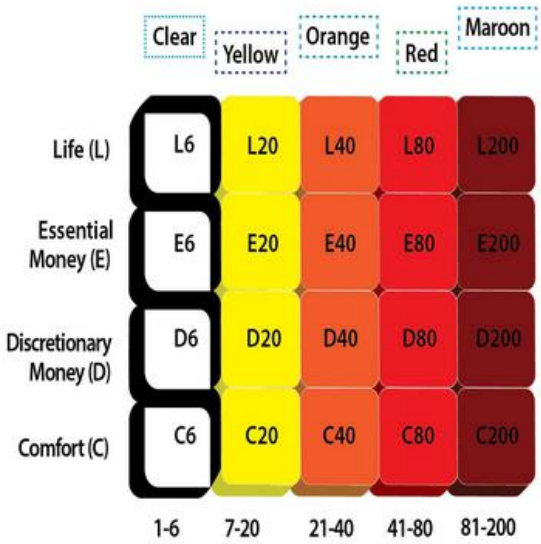


Fig 4 Crystal Families [8]

Haciendo la representación de la anterior figura como si fuera una gráfica, se puede observar que el eje X representa el tamaño del equipo de desarrollo. Dependiendo de este parámetro, las familias cambian de color, siendo el más pequeño el color más claro (Clear) y el más extenso el más oscuro (Maroon).

El eje Y, por tanto, muestra el impacto ante el fallo, distinguido en varias categorías dependiendo del impacto: si algo falla, lo único que ocurre es que se pierde algo de tiempo (C), se pierde dinero aumentando los costes (D), dinero del presupuesto del proyecto, lo que afecta a las funcionalidades (E), o directamente, si falla algo es bloqueante y puede hacer que la vida del proyecto se resienta (L).

Dependiendo de estos, se puede hacer la caracterización de los proyectos. Si, por ejemplo, el proyecto es un proyecto de una gran multinacional de desarrollo del software de un coche, este puede ser considerado un Crystal Maroon > L200, ya que el proyecto de desarrollo lo compone un equipo de desarrollo extenso y un fallo en producción puede crear la quiebra del proyecto. Sin embargo, si el equipo de desarrollo es pequeño y se ocupa de una pequeña funcionalidad de un proyecto, este puede ser un Crystal Clear > C6.

Desde el punto de vista de las metodologías ágiles, no todas las familias pueden seguir las directrices que el manifiesto prevalece. Como ya se vio en apartados anteriores, es preferible que los equipos sean más pequeños, y sobre todo en equipos

que no tengan proyectos que dependan de sus fallos en producción. Uno de los fundamentos de Agile es la mejora continua y evolución del producto, lo que puede llevar a fallos en producción, no deseables para estos proyectos.

Con estas características, el filtro de adecuación de Crystal existente ha elegido los tipos de familias Crystal que pueden ser factibles de usar las metodologías ágiles, marcados en la siguiente gráfica. Como se ve, estos están caracterizados por ser los que menos tamaño de equipo de desarrollo tienen, y los que el impacto por fallo supone menos [11]. Esto no quiere decir que las familias no incluidas no puedan utilizarlas, solo que es menos aconsejable dadas sus características. El único caso peculiar es que el Crystal Clear > E6 no sea recomendable para Agile. Esto es que pérdidas en el presupuesto, asumiblemente tan pequeño, pueden hacer que directamente la vida de este se vea perjudicado.

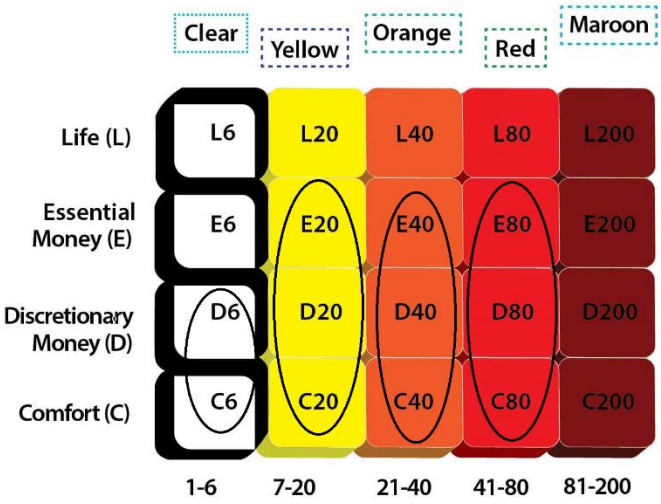


Fig 5 Crystal families adaptadas al Agile

2.3.1.3 Boehm and Turner – Radar Chart

Este filtro fue creado por Barry Boehm y Richard Turner en el libro “Balancing Agility and Discipline: A Guide for the Perplexed“ [12]. En él, se muestra una forma muy visual de comprobar el nivel de adecuación de una empresa al Agile.

Este gráfico será un gráfico de tipo radar. Al ser de este tipo, se tendrán cinco tipos de propiedades, que harán función de ejes, que marcarán el nivel de adecuación. Estos serán:

- **Skills del equipo**: Será mejor tener mayor proporción de desarrolladores con un nivel avanzado e intermedio que de desarrolladores principiantes.

- **Dinamismo del proyecto**: Dependiendo del número de requerimientos cambiados durante el transcurso del proyecto o número de puestas en producción durante un cierto periodo, será mejor o no utilizar metodologías ágiles o tradicionales.

- **Cultura de la empresa**: Este apartado tiene que ver con el comportamiento de la empresa acerca de los cambios. Existen empresas que no son favorables a cambios, o que directamente pueden afectarse mucho con las propiedades del agile (dejar de hacer documentación, métodos de organización…).

- **Tamaño del equipo y Criticidad**: Al igual que en el método de Crystal Families, estas propiedades se tendrán en cuenta de la misma manera para ver el nivel de adecuación ágil.

De esta manera, se crea un gráfico radar de este tipo:

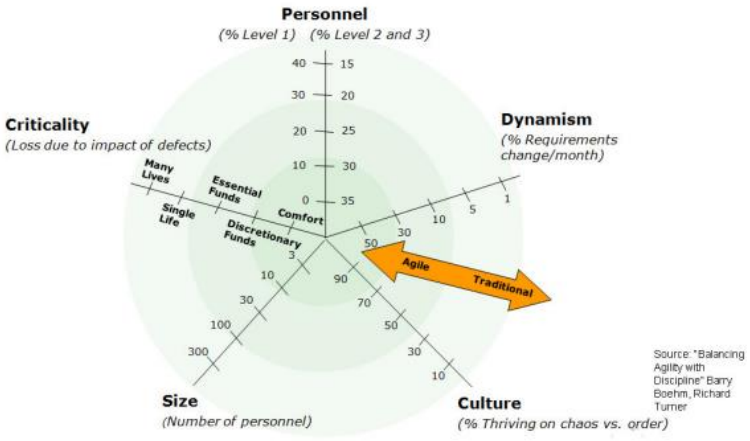


Fig 6 Grafico de Boehm y Turner [12] [11]

2.3.1.4 Dave Cohen's Agile Factors

Este filtro de adaptación fue publicado por Dave Cohen en el artículo "Advances in Computers" en 2004 [13]. En él, se identifican varias características del proyecto que sirven para saber el nivel de adecuación Agile:

- **La cultura de la organización debe ser basada en la negociación**: Es decir, como también se comenta en el filtro de Boehm y Turner, el impacto en la organización debe de ser el menor posible.

- **Se debe confiar en los equipos**: Este punto va en relación con que los usuarios y la alta dirección de la empresa deben confiar en el trabajo de sus desarrolladores y no poner trabas.

- **Poca gente, pero más competente**: Es preferible menos gente, pero más cualificada.

- **La organización debe existir con las decisiones de los desarrolladores**: Repite un poco la idea del segundo punto, aunque en este punto se hace más hincapié en el poder de decisión del equipo de desarrolladores.

- **El entorno de la organización debe ayudar a la rápida comunicación de los miembros del equipo.**

2.3.1.5 Organizational Suitability Filter

Como comentamos en el primer apartado, el consorcio de empresas DSDM publicó dos tipos de filtros de adecuación, uno relacionado con los proyectos (Project Suitability Filter), ya desarrollado en el primer apartado de este capítulo, y un segundo, basado en la organización, que es el que se va a analizar en este apartado.

Este filtro de adecuación tiene forma de cuestionario, dividido en 46 preguntas, de estas categorías:

- Usuarios: ¿Pueden hacer decisiones en el proceso de desarrollo?

- Alta dirección: ¿Entienden lo que es el desarrollo Agile?
- Organización: ¿Están dispuestos a cambios en la estructura
- Cultura de la organización: ¿Está dispuesto a integrar la metodología Agile?
- Equipo de desarrollo: ¿Están preparados para trabajar en Agile?
- Managers del equipo de desarrollo: ¿Entienden los procedimientos Agile?
- Técnicas de desarrollo: ¿Se utilizan técnicas para trabajar en Agile?
- Organización de la alta dirección: ¿Serán capaces de hacer a los stakeholders participes del desarrollo? ¿Les convencerán del desarrollo incremental?

Las ventajas de este cuestionario son que es fácil de rellenar al ser 46 preguntas y que analiza las áreas peligrosas en cuanto a las metodologías ágiles. Además, al final, es posible crear un gráfico de tela de araña con los resultados de la prueba.

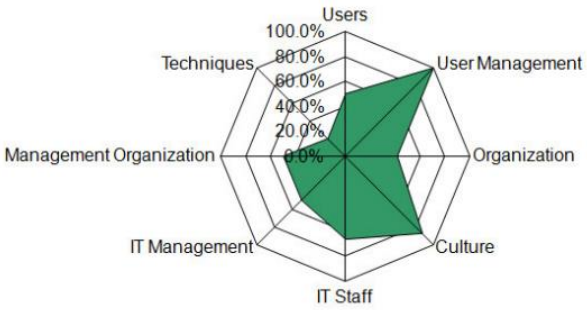


Fig 7 Gráfico de Organizational Suitability Filter

2.3.1.6 PMI: Guía de práctica ágil

Otro filtro de adecuación existente fue el creado por el Project Managment Institute (PMI) y la Agile Alliance en el libro “Guía práctica de ágil” [14]. En este libro se desarrollan herramientas, situaciones y enfoques para hacer que los equipos de desarrollo les sea más fácil la transición hacia metodologías Agile. En el apéndice de este libro, propone un cuestionario resumiendo algunos de los filtros de adaptación anteriormente vistos. Para ello, desarrolla un cuestionario con los siguientes apartados:

- Tamaño del equipo
- Experiencia del equipo

- Relación equipo – clientes
- Confianza de los clientes
- Confianza de la organización
- Capacidad de decisión de los desarrolladores
- Cambios intrínsecos del proyecto
- Criticidad del proyecto
- Entrega del proyecto

Como se puede comprobar, todos estos apartados han sido descritos implícita o explícitamente en los demás filtros de adecuación desarrollados a lo largo de este apartado.

3. DESARROLLO DEL FILTRO DE ADECUACIÓN

En este capítulo se mostrará como se ha hecho la implementación de una aplicación que tiene como objetivo mostrar un filtro de adecuación Agile. Este filtro de adecuación, formado por un cuestionario de 27, pretende agrupar los resultados obtenidos de cada uno de los filtros de adecuación desarrollados en el apartado 2.3, de manera que el usuario reciba en la misma aplicación toda la información.

3.1 Herramientas utilizadas en el desarrollo. Manual de Usuario

La aplicación desarrollada es una aplicación web basada en las librerías de JAVA con un servidor en local Tomcat que aloja la solución. Para implementar la aplicación se ha decidido seguir el patrón de desarrollo de software Modelo-Vista-Controlador. A continuación, se desarrollan las librerías y aplicaciones más importantes a la hora de hacer el desarrollo.

- **JAVA**: Como se ha comentado, la aplicación web está diseñada sobre JAVA. Por tanto, la aplicación reúne clases JAVA que soportan la lógica de la aplicación. Estas clases JAVA se podrían dividir en varios tipos:
    - **Modelos**: Servirán para definir los diferentes objetos utilizados durante el desarrollo. Por ejemplo, existirán las clases Pregunta, para referirse a cada cuestión implementada dentro del cuestionario, la clase Categoría, para referirse a los filtros de adaptación recogidos en la aplicación, o incluso la clase Gráfico, que recogerá las características de los gráficos de cada tipo.
    - **Controladores**: Utilizados para la lógica entre las vistas. Estos utilizan las clases de JAVA llamadas Servlets, especializadas en hacer las tareas de asignadas a los servidores web. En este tipo de clases se recogen los objetos de request y session, encargados de transportar datos entre las vistas para que la información sean uniformes.
    - Existen otro tipo de clases JAVA complementarias que tienen como objetivo recoger todas las funcionalidades requeridas por cada modelo,

desdoblándolas de lo que es el controlador. En este caso se ha referido a ellas como el nombre del modelo más “Actions”. Por ejemplo, existe la clase CategoriaActions, en la que se recogen todas las funcionalidades que tienen las categorías, por ejemplo, inicializar cada categoría dependiendo de sus características (si requiere o no gráfico, asignarle las preguntas del cuestionario que se corresponden con el filtro, etc.).

- **JSP**: Para las vistas se ha utilizado la clase JSP de JAVA. Este tipo de clases te permiten mezclar el lenguaje HTML, necesario para crear las vistas, con las lógicas de JAVA, útiles para crear condiciones y bucles que ahorran líneas de código. Como veremos más adelante, existen tres vistas generadas: una diseñada para la representación del cuestionario, la segunda siendo una vista general, en la que se representará un resumen de lo concluido a partir de las respuestas del cuestionario; y la tercera y última siendo una vista desarrollada para los detalles de todos los filtros de adecuación que son agrupadas dentro de la aplicación.

- **Apache Tomcat**: Aplicación que funciona como contenedor de servlets, y encargada de que la aplicación desarrollada pueda ejecutarse. Esta aplicación establece una base sobre la que los servlets y las vistas (JSPs) que incluye la aplicación pueden hacer su función. Esto hace que la aplicación pueda verse a través de un navegador en la dirección de localhost.

3.1.1 Guía de instalación

Para poder lanzar la aplicación en el ordenador, existen dos maneras que se explicarán a continuación. Primeramente, se explicarán los programas a instalar y su función.

3.1.1.1 Programas que instalar

A continuación, se detallan los archivos necesarios para la ejecución de la aplicación [12]:

- Archivo .war, contenedor de la solución, adjuntado con esta memoria.
- Programa Eclipse IDE: Disponible en el enlace https://www.eclipse.org/downloads/. En la propia descarga ya viene la instalación del Java RunTime Enviroment (JRE), necesario para que funcione. Solo es necesario si se va a ejecutar el programa con un IDE.
- Apache Tomcat: Servidor de páginas web, disponible en el enlace https://tomcat.apache.org/download-10.cgi. Para que funcione, es necesario tener instalado Java Development Kit, disponible en el enlace https://www.oracle.com/java/technologies/downloads/.

El primer paso será instalarse el Eclipse (si se quiere ejecutar a través de un IDE), para lo que será necesario antes haberse descargado el JDK. Para esta aplicación, con aplicar la instalación recomendada bastaría.

El siguiente paso será instalar Tomcat. Una vez descargado los archivos, habrá que entrar en la carpeta descargada, y dentro, entrar en la ruta */bin* para lanzar el fichero *startup.bat* para iniciarlo y *shutdown.bat* para pararlo. Hay que tener en cuenta que Tomcat va también sobre Java, por lo que será necesario indicarle la variable de entorno JAVA_HOME al JDK anteriormente descargado. Una vez hecho, si se abre un navegador y se pone la dirección http://localhost:8080, aparecerá la página principal de Tomcat, con el logo de la compañía:

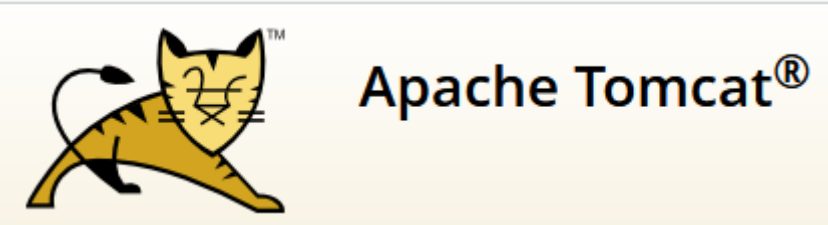


Fig 8 Logo de Apache Tomcat

3.1.1.2 Ejecución con IDE

La primera y más sencilla, es hacerlo a través de un IDE. En este caso, se explicará la guía de instalación por medio del IDE de Eclipse, que es el que se ha utilizado para este desarrollo.

Una vez instalado todo, podremos abrir Eclipse. En este punto tocará relacionar el servidor Tomcat instalado con el IDE. Esto se hará dentro de la pestaña Window > Preferences > Server > Runtime Environments. En este cuadro aparecerá un pop-up para añadir el servidor. En este caso, necesitaremos pinchar en la versión de Tomcat instalada y, meter las direcciones de instalación. Una vez hecho, ya se podrán lanzar aplicaciones web desde Eclipse.

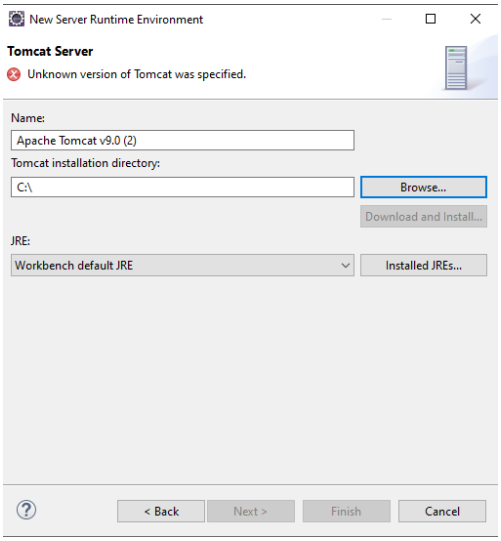


Fig 9 Ventana para añadir servidor en Eclipse

El último paso es importar el archivo .war que contiene el proyecto a Eclipse. Esto se hará, en la barra de herramientas de Eclipse, pinchando en File > Import > Web > WAR File. En la ventana para importar que aparecerá, habrá que meter la ruta al fichero .war (donde al rellenarlo aparecerá el nombre del WebProject, y después, añadir el servidor Tomcat anteriormente configurado:

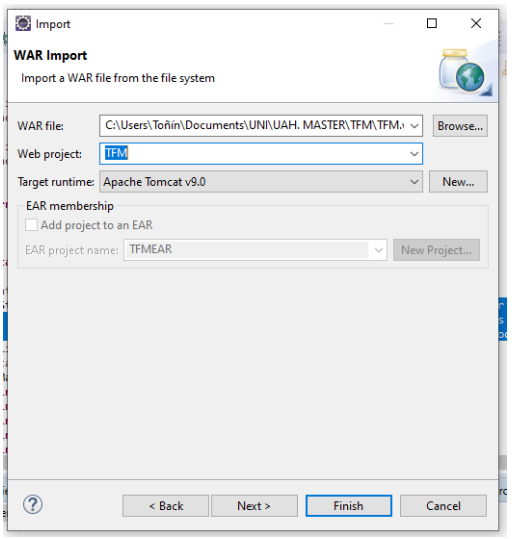


Fig 10 Ventana de Eclipse para importar archivos .war

Una vez importado, el proyecto aparecerá en el IDE y podremos ejecutarlo pinchando en el botón Run.

3.1.1.3 Ejecución sin IDE

Existe otra manera de ejecutar la aplicación, aunque esta es algo más complicada, ya que está a expensas de usuarios administradores.

Una vez instalado Tomcat, se va a iniciar el servidor en la dirección http://localhost:8080, para que aparezca la página principal de Tomcat. Una vez aparezca, pincharemos en Manager App, que nos permitirá acceder, previa identificación, a las opciones para manejar el servidor.

Una vez se abra esta página, aparecerá el gestor de aplicaciones de Tomcat. En ella, podremos cargar las aplicaciones para que las lance el servidor, ver estadísticas, etc. Para subir la aplicación, se tendrá que adjuntar en la sección “Desplegar > Archivo WAR a desplegar”, y seguidamente pinchar en el botón “Desplegar”. Una vez cargue, aparecerá en la sección de Aplicaciones, comprobando que está bien cargada.

Fig 11 Sección Desplegar de Tomcat

Ya con esto, solo hará falta escribir en el navegador la URL correcta (en este caso, http://localhost:8080/TFM) y comprobar que el cuestionario aparece correctamente.

3.2 Desarrollo del cuestionario

La fase inicial del desarrollo de la aplicación fue buscar información acerca de los filtros de adecuación Agiles existentes, eligiendo para la aplicación los filtros desarrollados en el apartado 2.3.

Una vez elegidos los filtros, la tarea principal fue desarrollar un cuestionario para realizar nuestro propio filtro de adecuación, ateniéndose a un objetivo principal: reducir al máximo el número preguntas del cuestionario, pero sin perder las características de cada uno de los filtros incluidos. Como resultado de esto, se integró un cuestionario de 27 afirmaciones, listado a continuación, que será el elegido para hacer todo el desarrollo posterior:

N.º	Preguntas
1	Los clientes de la aplicación están dispuestos a cambios de organización,
2	La organización vería bien un cambio de metodología,
3	Hay una separación de tiempos en la organización,
4	Existe confianza en el equipo de desarrollo por parte de los clientes,
5	El equipo de desarrollo tiene total poder de decisión,
6	La alta dirección estaría dispuesta a dar más poder de decisión al equipo de desarrollo,
7	Los usuarios finales/clientes participan en los procesos de desarrollo (Haciendo pruebas antes de subidas a producción, sugiriendo mejoras, etc.),
8	El equipo de desarrollo sabe en todo momento que es lo que quiere el cliente,
9	El equipo de desarrollo tiene acceso a los entornos de producción,
10	El rango de rotación del equipo es bajo,
11	¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?,
12	¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?,
13	¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?,
14	¿De cuantas personas se compone el equipo de desarrollo?,
15	Dentro de la organización se trabaja en iteraciones,
16	El porcentaje de requerimientos cambiados al mes es bajo,
17	Se utilizan tecnologías que apuesten por la pronta entrega a cliente (empaquetados rápidos del producto, subidas a producción de pocas horas…),
18	El producto entregado se puede construir y evaluar en porciones,
19	Si existe un cambio, este se puede revertir rápidamente de producción,
20	¿Si existe un cambio de alcance dentro del desarrollo de una funcionalidad, que supondría?,

21	Se hacen pruebas durante el proceso de desarrollo,
22	El periodo de pruebas una vez subido a producción es acorde a las funcionalidades creadas,
23	El tiempo de puesta a producción asumido es acorde a las funcionalidades creadas,
24	¿Si existe un fallo en la subida a producción, que supondría?,
25	¿Si existe un fallo en producción que supondría?,
26	La comunicación entre personas del mismo equipo de desarrollo es fluida,
27	La comunicación entre desarrolladores y clientes/alta dirección es fluida

TABLA 2 Preguntas del cuestionario

Se decidió que las cuestiones tuvieran formato de afirmación debido a la decisión de asignación de resultados, que veremos a continuación.

3.2.1 Asignación de resultados

Una vez creado el cuestionario y añadido en la aplicación, hace falta asignar los resultados pertinentes a cada cuestión. Para intentar resumir cada filtro de adecuación, se decidió que, por regla general, el usuario pueda elegir entre cinco opciones, del 1 al 5, siendo el número 1 que se está en desacuerdo de que la afirmación mostrada y el 5, que se está totalmente de acuerdo con que la afirmación se cumple en su organización. A efectos de resultados, cada cuestión se ha dividido para que el resultado optimo equivalga el 100% de la puntuación de la pregunta, la siguiente 80%, y así sucesivamente.

Por otra parte, existen cuestiones que, dada su naturaleza, no pueden tener estos tipos de resultados, A continuación, se adjunta el listado de las cuestiones con resultados diferentes:

Pregunta	**Opción 1**	**Opción 2**	**Opción 3**	**Opción 4**	**Opción 5**
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?	<25%	<50%	<60%	<75%	>75%

¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?	<25%	<50%	<60%	<75%	>75%
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?	<25%	<50%	<60%	<75%	>75%
¿De cuantas personas se compone el equipo de desarrollo?	1-6	7-20	21-40	40-80	81-200
¿Si existe un cambio de alcance dentro del desarrollo de una funcionalidad, que supondría?	Nada	Pérdida de tiempo	Dinero del proyecto, reduciendo el presupuesto	Dinero, incrementando los costes	Todas las anteriores
¿Si existe un fallo en la subida a producción, que supondría?	Nada	Pérdida de tiempo	Dinero del proyecto, reduciendo el presupuesto	Dinero, incrementando los costes	Todas las anteriores
¿Si existe un fallo en producción que supondría?	Nada	Pérdida de tiempo	Dinero del proyecto, reduciendo el presupuesto	Dinero, incrementando los costes	Todas las anteriores

TABLA 3 Preguntas con opciones diferentes

A efectos de los porcentajes totales, se ha realizado una parametrización de las puntuaciones de estas cuestiones de manera que equivalgan igual que las cuestiones con opciones comunes. Las puntuaciones de las cuestiones se mostrarán en la aplicación al lado de las cuestiones, divididas en cada teoría, en la vista de la categoría. Además, al lado, aparecerán distintos iconos dependiendo de la respuesta:

- Si la puntuación de la respuesta es superior a 80%, aparecerá un icono de un check verde, que haciendo *hover* (dejando el ratón encima) aparecerá el texto “Se cumple el principio agile”.

- Si la puntuación está entre el 20% y el 80%, aparecerá un icono de 50% con el texto: “No se cumple el principio agile, pero necesitaría de pequeños cambios para cumplirlo”.

- Si la puntuación es menor del 20%, el icono será una cruz roja con el texto “No se cumple el principio agile y costaría llevarlo a cabo”.

3.2.2 Asignación teoría-cuestión y cálculo de porcentajes

Como se ha comentado anteriormente, cada pregunta que conforma el cuestionario está relacionada directamente con una o varios filtros de adecuación creados por expertos en Agile. A su vez, cada filtro de adecuación Agile se constituye a partir de varios principios o categorías, por lo que las preguntas se han dividido también en los principios de cada filtro. En el anexo se adhiere una tabla que relaciona la pregunta del cuestionario con el o los filtros y dentro de él, y el principio asociado:

Estas relaciones de cada pregunta en su teoría y su principio asociado servirán para calcular los porcentajes de adecuación que marcará la aplicación para mostrar, de una manera cuantitativa, si la organización está adaptada al Agile en base a las respuestas del cuestionario. Se van a calcular 3 tipos de porcentajes:

- **Porcentaje de principio del filtro**: Este porcentaje será calculado con las puntuaciones de las cuestiones que conforman el principio de dentro del filtro de adecuación existente. El valor del porcentaje se verá al lado del enunciado del principio.
- **Porcentaje de categoría**: Este porcentaje es el general por filtro existente. Se verá en la vista detallada de la categoría en la parte superior y en la vista general, al lado del nombre del filtro.
- **Porcentaje general**: Valor que concentra la media de los porcentajes de categorías. Este porcentaje se verá en la parte superior de la vista general.

3.2.3 Creación de figuras

Existen filtros de adaptación existentes incluidos en la aplicación que están basados en gráficas. Por ejemplo, una de las ventajas del filtro de adecuación de Boehm y Turner es que simplifican de una manera gráfica las características del proyecto.

Los filtros de Boehm y Turner y el *Organizational Suitability Filter* necesitan de estos gráficos, en este caso en tipo radar, por lo que, para garantizar la buena inclusión de estos en la aplicación, se necesitó introducir desarrollos para la creación de los gráficos con los resultados obtenidos del cuestionario.

Para ello, en el desarrollo se ha incluido la librería ChartDirector, que permite crear el gráfico radar pasando las categorías y las puntuaciones en dos arrays diferentes. A continuación, se incluye el código específico:

```
//Creamos el gráfico radar
PolarChart c = new PolarChart(600, 500);

// Poner el centro en estas coordenadas
c.setPlotArea(300, 250, 150);

// Insertar el Area de las layers
c.addAreaLayer(data, 0x9999ff);

// Insertar las labels
c.angularAxis().setLabels(labels);

// Especifica el formato de las labels
c.radialAxis().setLabelFormat("{value}%");
```

Fig 12 Código de creación de gráfico

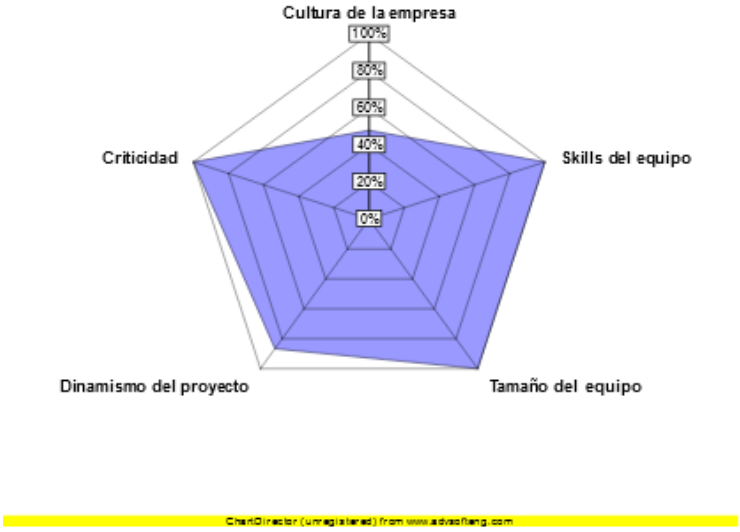


Fig 13 Gráfico radar creado

3.3 Estructura de la aplicación

En este apartado se muestra el flujo de las vistas de la aplicación. La aplicación se divide en tres vistas diferenciadas: En la primera se muestra el cuestionario con los resultados, en la segunda se hace un resumen general de las puntuaciones obtenidas en el cuestionario de la primera vista y en la tercera, los detalles del filtro recogido incluido en la aplicación. La aplicación en todo momento tendrá un fondo azul marino, con una cabecera roja. Sobre estas, existirán contenedores con el fondo blanco para que el texto, de fuente Comic Sans de color negro, se vea correctamente.

3.3.1 Vista de cuestionario

La aplicación arranca con una vista que muestra el cuestionario a rellenar por el usuario. En la cabecera de esta vista se verá el título del cuestionario “Cuestionario Agile”, y unas instrucciones para poder rellenarlo tal y como se ha explicado.

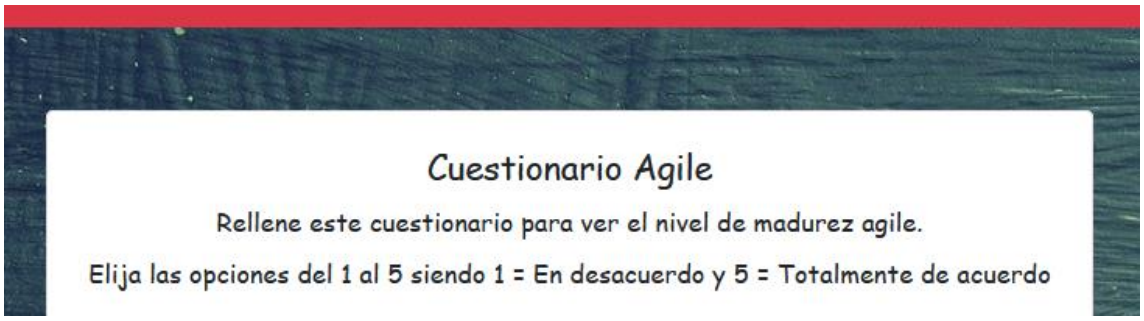


Fig 14 Cabecera y título del cuestionario

Después, aparecen las preguntas del cuestionario con sus respectivas puntuaciones. Como se ha comentado anteriormente, todas las cuestiones tienen una puntuación del 1 al 5 dependiendo del grado de conformidad con la cuestión, salvo excepciones ya detalladas. Además, solo se puede pulsar una opción por pregunta.

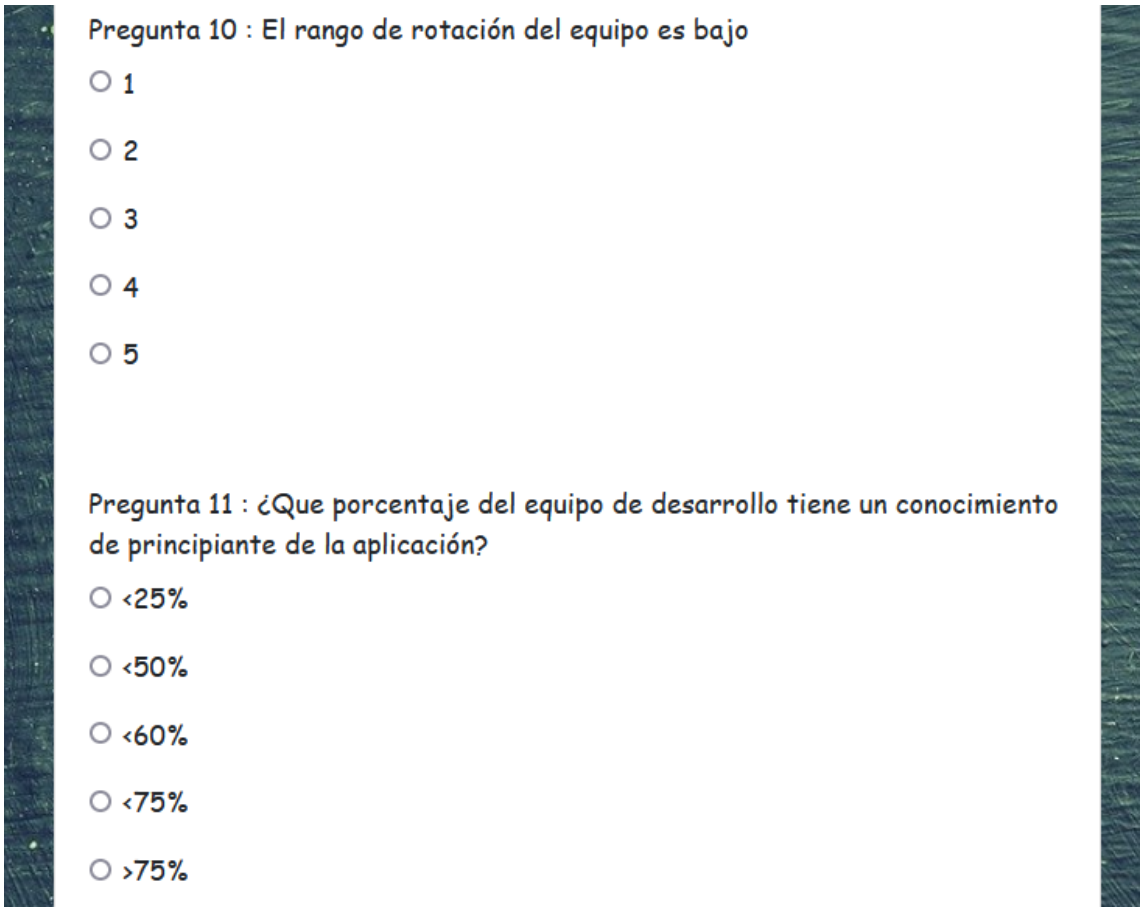


Fig 15 Preguntas del cuestionario

Cuando se termina de rellenar el cuestionario, existe un botón con el texto “VER MADUREZ AGILE” que dará paso al análisis de resultados. Cuando este botón sea pulsado, se añaden las puntuaciones, se hacen las lógicas correspondientes y se pasa a la vista general, donde se empiezan a mostrar resultados acordes a lo rellenado en el cuestionario.

Fig 16 Última pregunta y botón "Ver Madurez Agile"

3.3.2 Vista general

Esta vista está diseñada para mostrar los resultados principales obtenidos con las respuestas del cuestionario. En la parte superior de la vista se podrá encontrar el porcentaje general del cuestionario Agile, que como comentamos en capítulos anteriores, se componía de la media de los porcentajes de puntuación de cada uno de los filtros incluidos en la aplicación. Este se da con un título que lo precede:

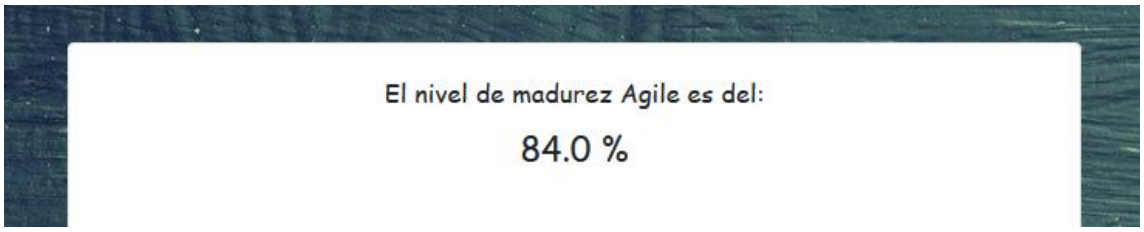
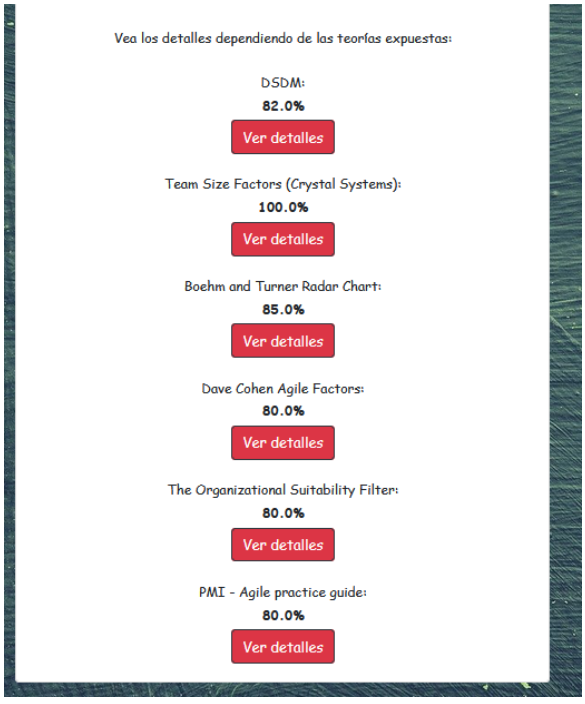


Fig 17 Representación del porcentaje general

Posteriormente, se observa la parte de acceso a los detalles de los filtros conocidos de autores que están incluidos en la aplicación. Para cada uno de ellos, se muestran tres características:



- Nombre del filtro conocido
- Porcentaje de cumplimiento basado en el filtro conocido en negrita para que resalte.
- Botón Ver Detalles.

En caso de pulsar el botón Ver Detalles de una de estas categorías, la aplicación redirigirá a lo conocido como Vista de teoría, en la que se desarrollarán las características de cada filtro existente.

Fig 18 Representación del resumen de las teorías

3.3.3 Vista de teoría

En esta vista se desarrolla todo lo expuesto en las otras vistas. El propósito de esta pantalla es saber, dependiendo de la teoría seleccionada, si la respuesta dada se adecúa a lo que exponen las metodologías Agiles en cuestión.

Para ello, la parte superior de la vista recoge el título del filtro seleccionado con una pequeña frase introductoria de la teoría, en esta tabla se recogen los títulos con las frases introductorias.

FILTRO EXISTENTE	DESCRIPCIÓN
DSDM Suitability Filter	Cuestionario más primitivo para medir la adaptación Agile
Team Size Factors (Crystal Systems):	Cuestionario desarrollado por Crystal Systems, mayormente enfocado en los tamaños de equipo y tipo de daño ocasionado dependiendo de la perdida
Boehm and Turner Radar Chart:	Gráfico creado para medir el agile dependiendo de ciertas características del proyecto
Dave Cohen Agile Factors:	Estudio en el que se discute la adaptación Agile en varias categorías consideradas predecesoras
The Organizational Suitability Filter:	Cuestionario de DSDM para medir el Agile en clave organización
PMI - Agile practice guide:	Guía Agile sacada del libro del PMI, quizá la más seguida del mundo

TABLA 4 Teoría y descripción en la aplicación

Después de esto, aparece el porcentaje de categoría que también aparece en la vista de resumen. Se ha incorporado que el porcentaje de cumplimiento se recuadre a modo de semáforo. Es decir, si es mayor que el 66%, este aparecerá recuadrado de un color verde. Si está entre el 66% y el 33% aparecerá recuadrado de amarillo, y si es menor que el 33% aparecerá recuadrado de rojo:

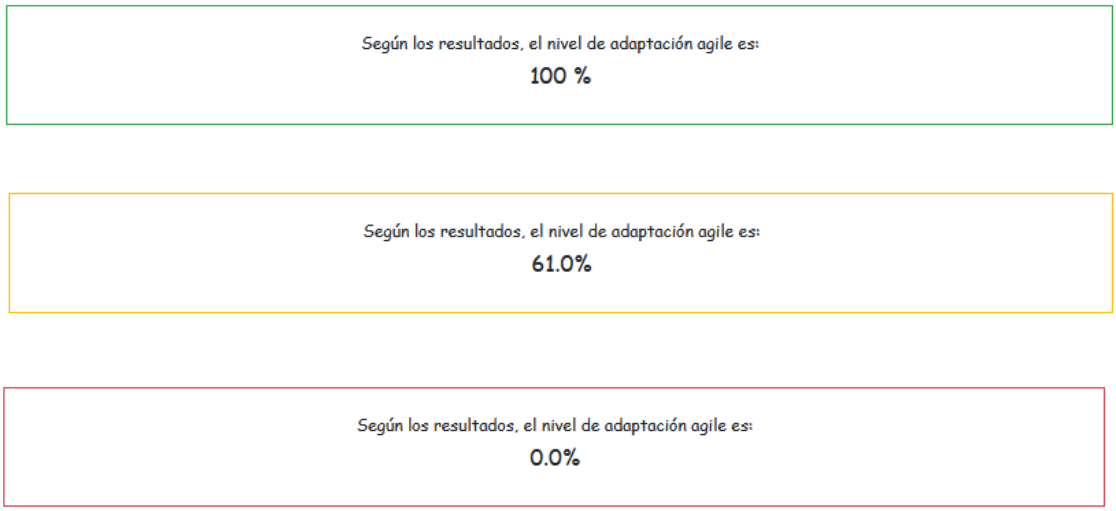


Fig 19 Recuadrado de porcentaje de cumplimiento

Debajo de estos contenedores, se mostrarán los gráficos creados en caso de que sea necesario representarlos. Como hemos comentado anteriormente, se necesitará gráfico para estos tres tipos de filtros:

- Crystal Systems: El filtro diseñado por Crystal Systems desarrolla su teoría por medio de varios factores que determinan si la organización está llevando a cabo metodologías o no. Todo esto lo resume en un gráfico como el que se muestra a continuación. Esta figura también se incluye en la aplicación:

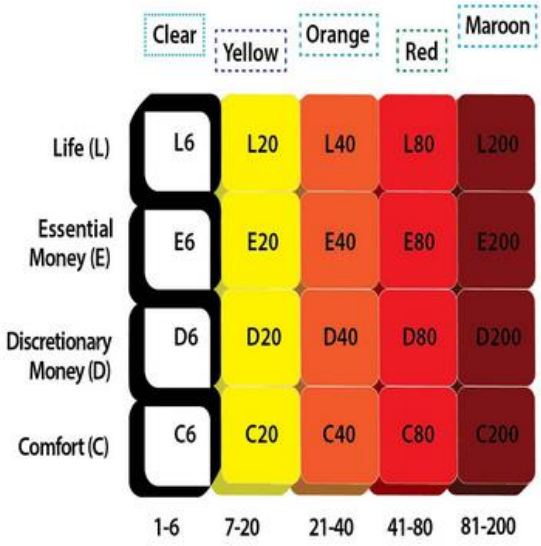


Fig 20 Gráfico de Crystal Systems [11]

- Boehm y Turner y The *Organizational Suitability Filter*: Estas categorías se basan en crear gráficos representativos tipo radar. Estos gráficos se representan en esta parte, con un texto indicativo de como analizar el gráfico:

Gráfico tipo radar del modelo. Cuánto más area tenga el polígono creado en la gráfica, más Agile será la organización

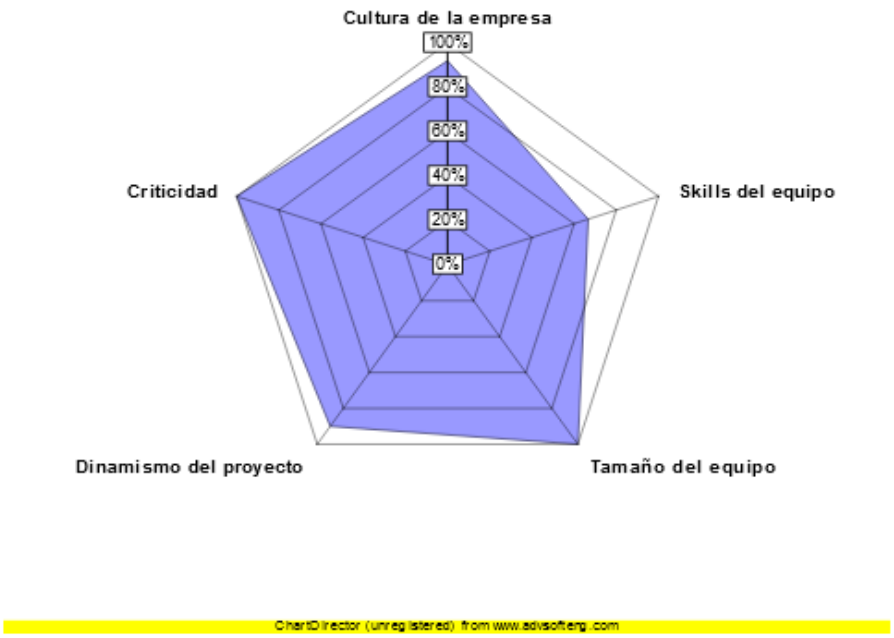


Fig 21 Gráfico de categoría

La última parte de la categoría será el desglose de los principios de cada una de ellas. En este apartado se verá el enunciado del principio junto a el porcentaje de cumplimiento del principio. Después del principio, aparecerá un listado en el que se incluyen los enunciados de las preguntas del cuestionario que se relacionan con dicho principio. Dependiendo de la puntuación de cada pregunta, aparecerá el icono del *tick* verde, cruz roja, o icono de mitad amarillo.

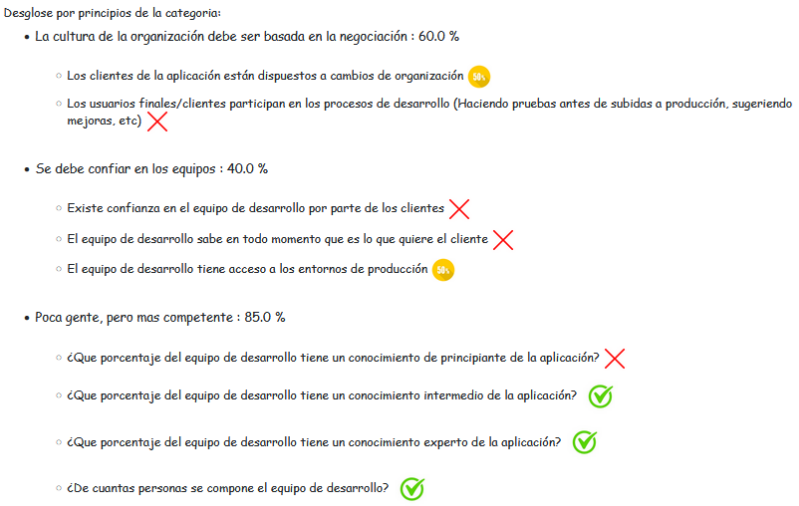


Fig 22 Categorías y principios por categoría

Un caso especial es el caso de la categoría de Crystal Systems. Como vimos en apartados anteriores, este filtro de adaptación no se ajusta a principios, sino que hace una clasificación con respecto a varias características y asigna la familia Crystal a la que pertenece la organización. La aplicación permite hacer esta clasificación y, en base a la familia a la que pertenece, justifica si es Agile o no y en que característica falla. Por tanto, la puntuación Agile de esta teoría será o 0% o 100%.

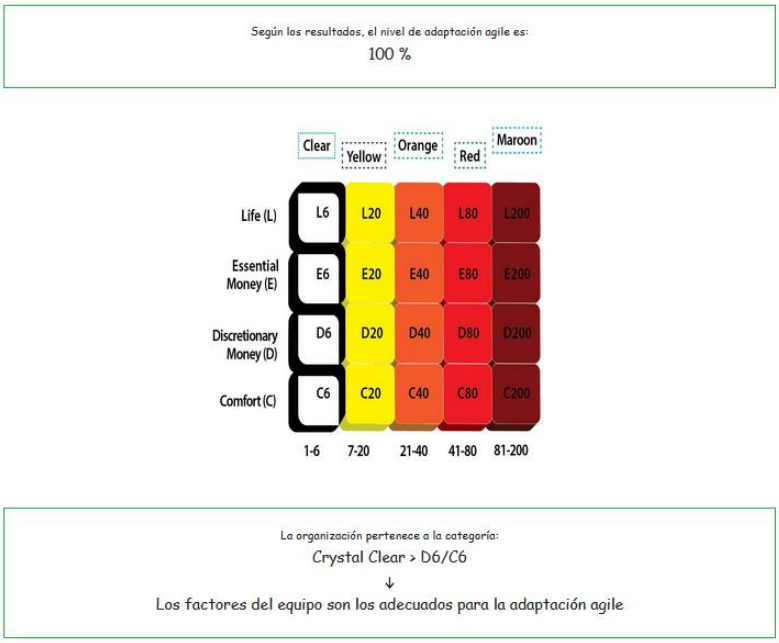


Fig 23 Vista de Crystal Systems al 100% con puntuación

4. RESULTADOS DEL FILTRO

En este capítulo se explicarán los análisis realizados tras que varios tipos de empresas hayan utilizado la aplicación realizada y explicada durante este proyecto. Cada empresa tiene sus características, que serán analizadas en los siguientes apartados. Por motivos de privacidad, estas empresas han sido anonimizadas, aunque se expondrán varios datos verídicos sobre ellas en el cuestionario. Para ello, se ha pedido el consentimiento del tratamiento de los datos incluidos en esta aplicación, y únicamente para este proyecto.

4.1 Empresa 1

La primera empresa que realizó un estudio de su adecuación Agile fue una empresa considerada como una gran corporación.

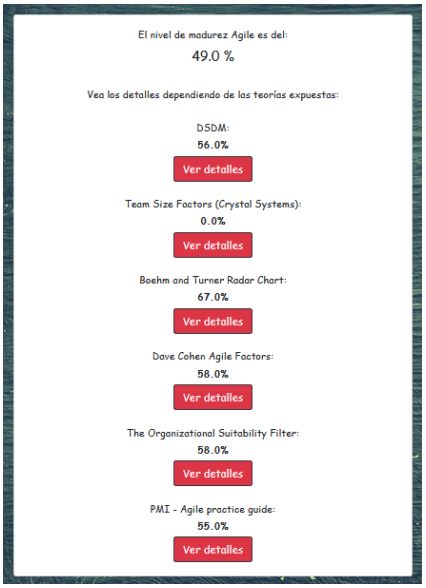


Fig 24 Vista general de los resultados de la empresa 1

Tras ver los resultados del cuestionario (reflejados en la tabla del anexo), se puede ver que esta empresa tiene un equipo de desarrollo de un gran tamaño, y con cierto poder de responsabilidad. Por otra parte, los clientes no tienen relación prácticamente con el equipo de desarrollo, lo que hace que no se integren en el desarrollo del producto hasta el final. Como podemos ver en la vista general (Fig 24), tiene un 49% de adecuación Agile.

Seguidamente, se muestran los resultados más importantes detallados de cada categoría, mostrados en la vista de categoría:

- **DSDM Suitability Filter**: Según este filtro, la organización tendría un 56% de adecuación a las metodologías ágiles. Analizando los resultados por los principios expuestos por la categoría, podemos ver que existen principios que la organización cumple a su 100% (por ejemplo, en todo lo relacionado con la pronta puesta en producción y el desarrollo incremental), pero otros que no cumplen y tienen la mínima puntuación del 20% como en el tamaño del equipo o en hacer los cambios reversibles.

- **Crystal Systems**: Según la metodología de Crystal Systems, la organización no puede ser Agile por su gran número de integrantes en el equipo de desarrollo, ya que pertenece a la familia de Crystal Maroon. Por ello, según esta metodología, tiene una adecuación de un 0%.

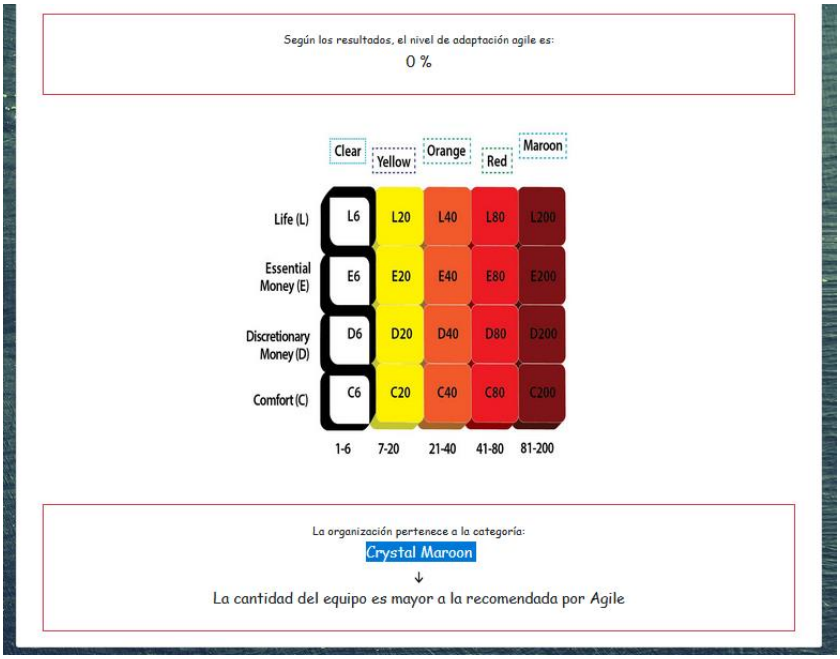


Fig 25 Detalle de Crystal Systems en la empresa 1

- **Boehm y Turner**: Derivado de este filtro salen los resultados más favorables en cuanto a nivel de adecuación Agile con un 69%. Al igual que en el filtro de DSDM, tiene el valor mínimo en tamaño del equipo (un 20%), pero lo compensa con el 90% y 70% en otros apartados como en la criticidad y el dinamismo, respectivamente. El gráfico radar resultante queda de esta manera:

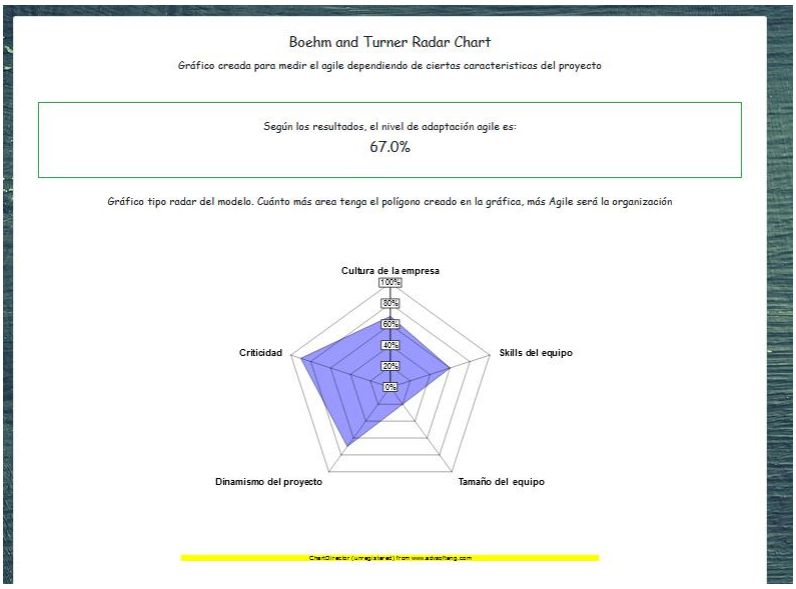


Fig 26 Radar Boehm y Chart para la empresa 1

- **Dave Cohen Agile Factors**: El filtro de adaptación de Dave Cohen muestra aproximadamente los mismos resultados que el filtro DSDM, con un valor cercano de 58%. Cabe destacar que, en esta categoría, no existe ningún principio con la nota mínima del 20%, únicamente la más baja es el principio relacionado con el poder de decisión del equipo de desarrollo del 30%. Existe un principio que se cumplen al 100%, el relacionado con la cultura con respecto al cambio de la organización.

- **Organizational Filter**: Con respecto a dicho filtro de adaptación, los principios están bastante equilibrados con lo que es la media general que es el 58%. El gráfico radar generado por este filtro quedaría de la siguiente manera:

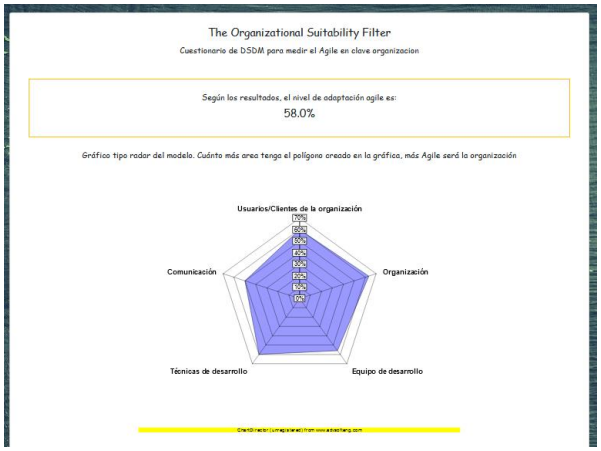


Fig 27 Gráfico de Organizational Filter para la empresa 1

- **PMI**: El filtro definido en el libro del PMI no aporta mayor diferencia a los demás filtros aplicados. El principio relacionado con el tamaño del equipo sigue con la puntuación más baja, mientras que las demás están un poco por encima del 60%, lo que hace que la media general quede con un 55%.

4.2 Empresa 2

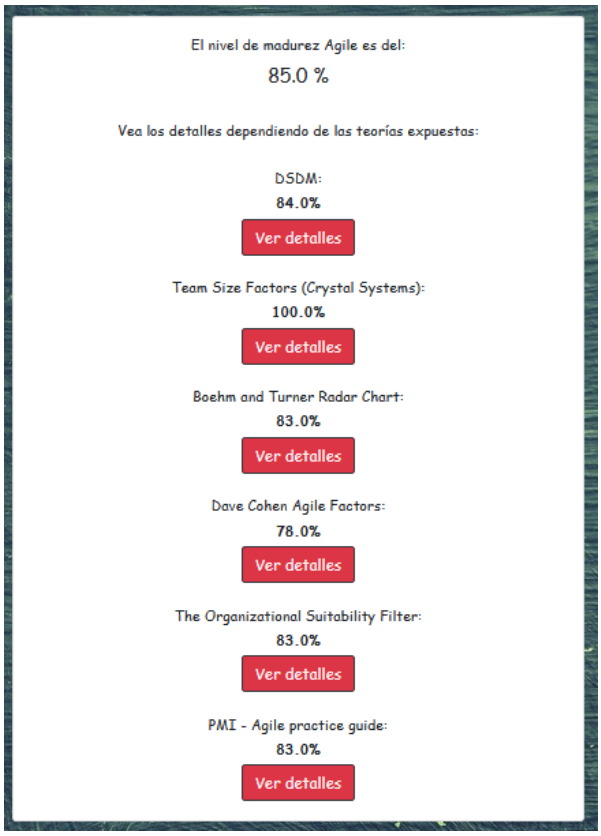


Fig 28 Vista general para la empresa 2

La segunda empresa que realizó el cuestionario es, al contrario de la primera empresa, una empresa mucho más pequeña, lo que reduce el número de desarrolladores participantes en el equipo. Además, hace que la cadena de mando sea mucho más reducida, lo que hace que la comunicación con la alta dirección sea mucho más accesible.

Por otra parte, los clientes están muy implicados en el desarrollo del producto, lo que hace que haya reuniones con ellos casi todas las semanas y haya un dialogo continuo

mediante correos. Además, tienen plena confianza en el equipo de desarrollo y les han dado todo lo que han pedido.

Al ser una compañía pequeña, muchas veces no puede competir con la competencia del sector, lo que hace que los desarrolladores no tengan muchos conocimientos y la estabilidad del equipo sea menor que en otras.

Por ello, una vez rellenado el cuestionario (cuyas respuestas aparecen en la tabla del anexo), los resultados de la figura anterior han sido hallados. El detalle por cada filtro ha resultado:

- **DSDM**: Como se puede ver, esta empresa cumple en un 84% la metodología. Todos los principios tienen más de un 90%, salvo el principio relacionado a las skills del equipo y la estabilidad del equipo, que hace que la media caiga más.

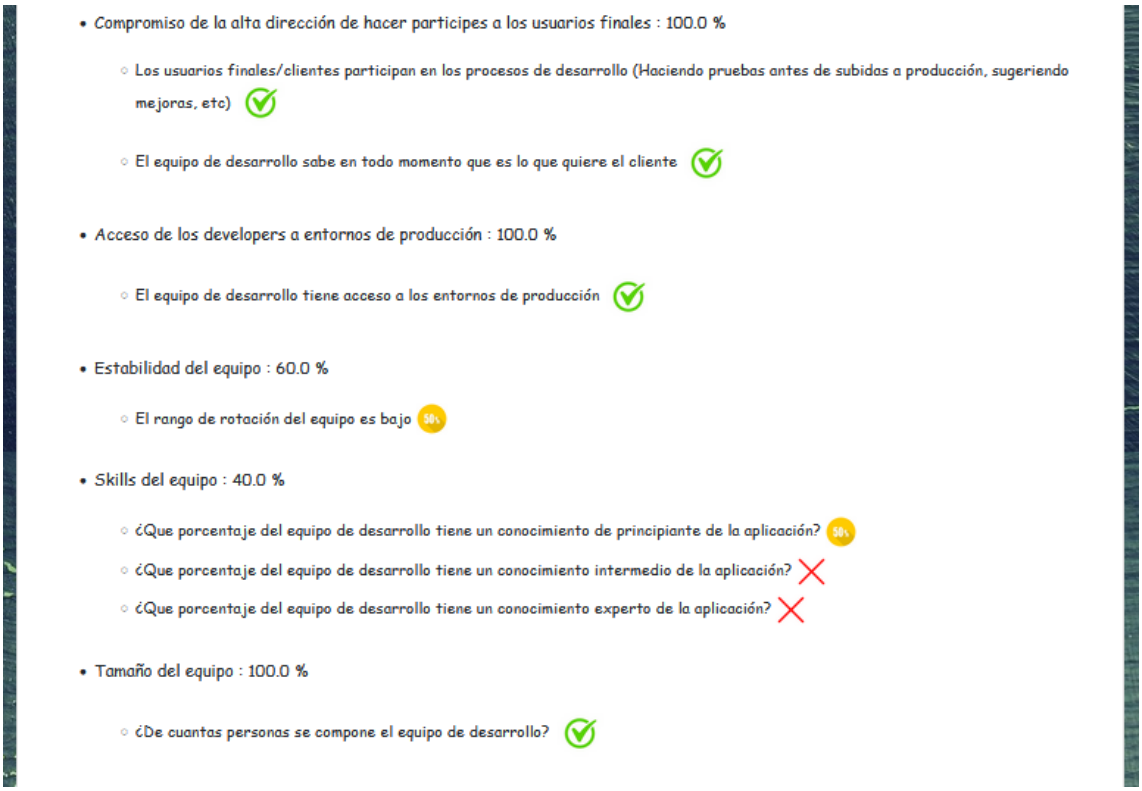


Fig 29 Ejemplo de DSDM para empresa 2

- **Crystal Systems**: Según lo rellenado, por el tamaño del equipo de desarrollo y la criticidad que tienen los posibles distintos fallos, la empresa pertenece a la familia Crystal Clear > D6/C6. Por lo tanto, tiene propiedades que hacen que la empresa esté adaptada al Agile

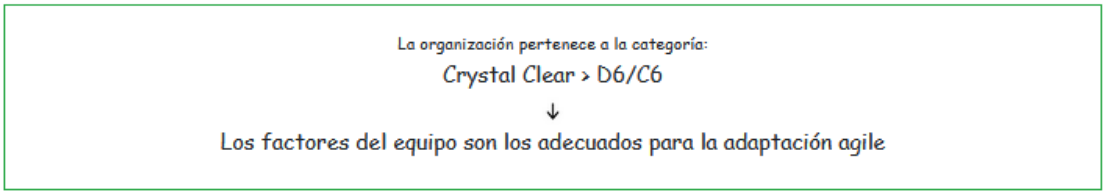


Fig 30 Crystal Clear para empresa 2

- **Boehm y Turner**: La metodología tiene un 83% de adaptación. Cabe resaltar que prácticamente todos los principios tienen una adecuación del 100%, menos las skills del equipo, que bajan considerablemente la media, al tener únicamente la puntuación del 40%.

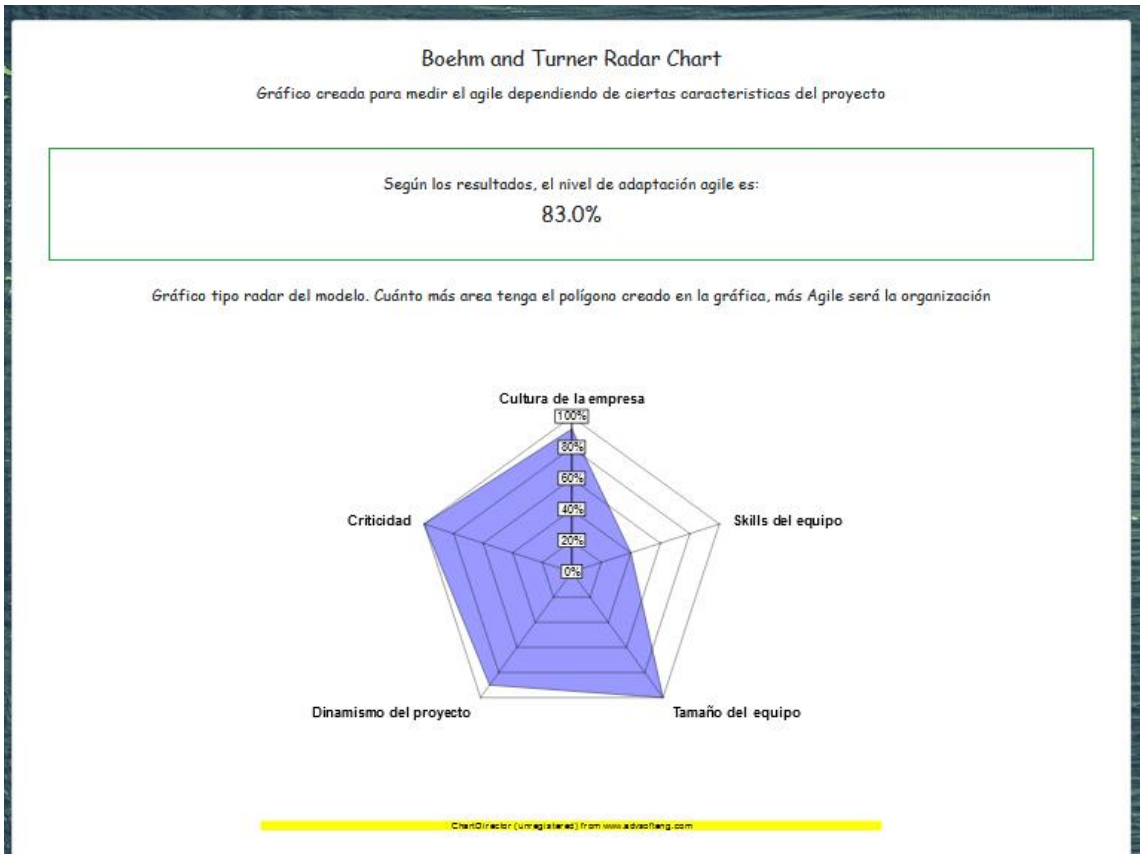


Fig 31 Radar de Boehm y Turner para empresa 2

- **Dave Cohen**: La puntuación de este filtro de adecuación es del 78%. A diferencia de las demás categorías, existen menos principios con el 100% de puntuación (de los 5 principios, solo 2 tienen una puntuación del 100%), y el principio relacionado con las skills del equipo tiene una puntuación del 55% al incluir el tamaño del equipo.

- **Organizational Filter**: Como en las demás categorías, la mayoría de las categorías tienen la puntuación completa. En este caso, existen dos principios con menos del 100%: el principio relacionado con la organización por tema de requerimientos bajos; y las skills del equipo de desarrollo. Por tanto, la puntuación general es del 83%.

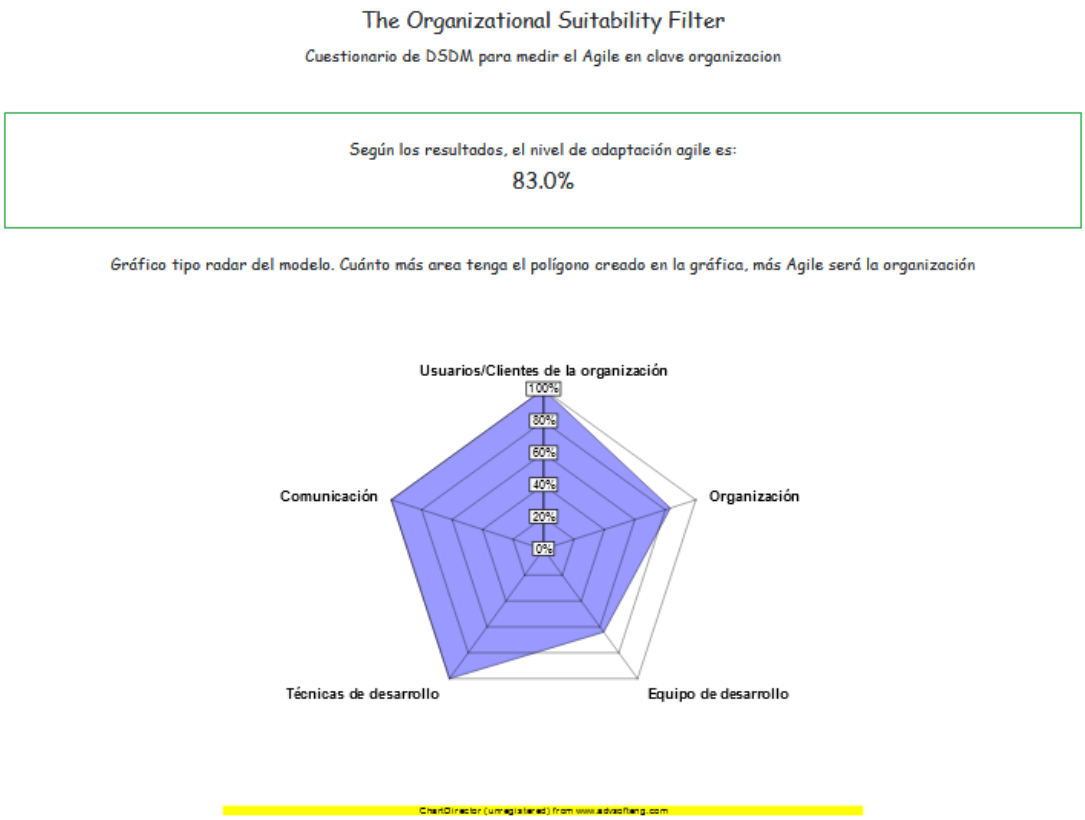


Fig 32 Gráfico de Organizational Suitability para la empresa 2

- **PMI**: Al igual que el *Organizational Filter*, esta categoría tiene una puntuación del 83% y, al igual que en las demás, la mayoría de los principios están por encima del 90% menos el principio relacionado con las skills.

4.3 Empresa 3

La tercera empresa está caracterizada por desarrollar un producto dedicado a la ciberseguridad. Por lo tanto, las características del entorno de utilidad del producto hacen que el producto sea muy sensible a fallos, haciendo que un fallo produzca perdida de dinero y márketing de la empresa, ya que podría provocar ciberataques que

podrían ser fatales para los clientes. Por ello, la alta dirección de la empresa y los clientes no son muy partidarios de cambios a nivel organizacional, aun estando muy involucrados en el desarrollo del producto, introduciendo mejoras y probando antes de la implantación.

Por otra parte, la empresa divide a los equipos de desarrollo en sub-equipos de como máximo 5 trabajadores que se dividen las distintas funcionalidades del producto. Los desarrolladores son gente muy cualificada, gastando mucha parte del presupuesto en el capital humano.

Por ello, una vez relleno el cuestionario, se han obtenido los siguientes resultados:

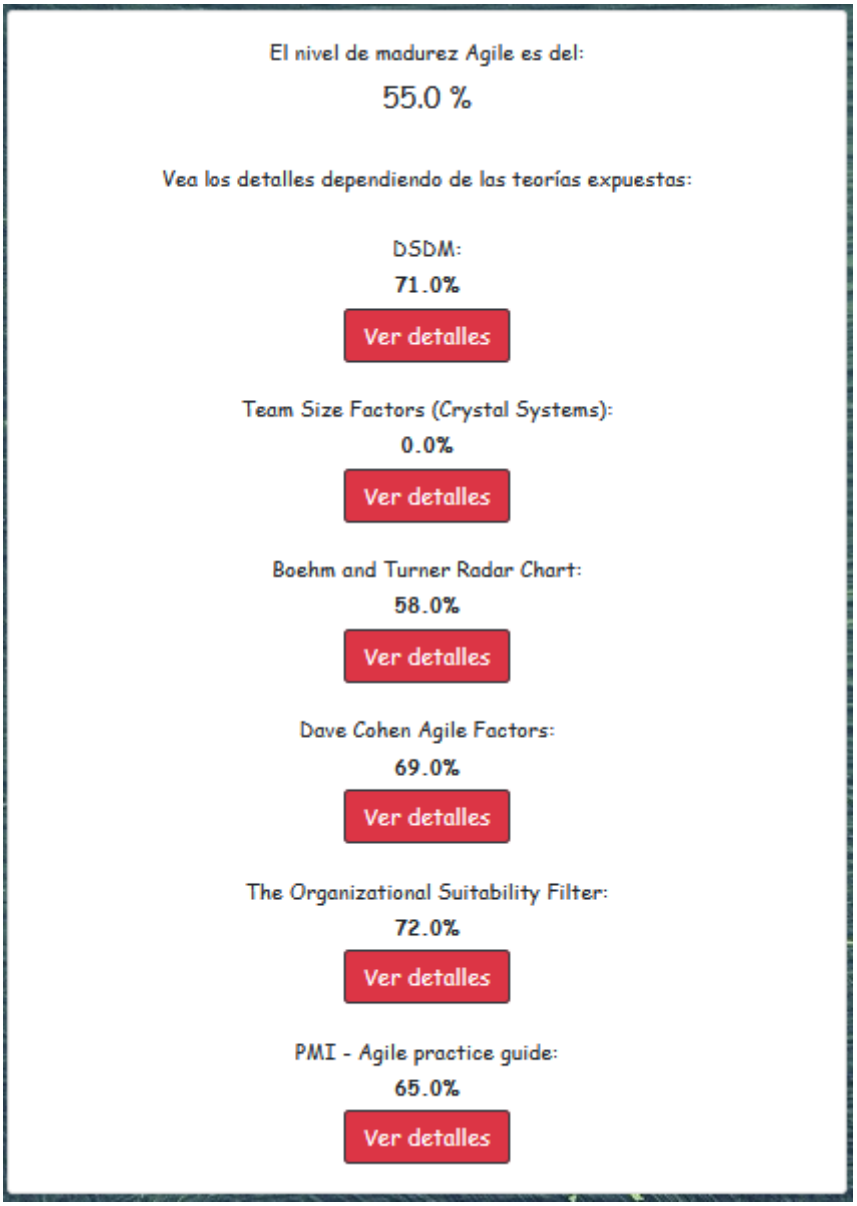


Fig 33 Vista general de la empresa 3

- **DSDM**: Para este caso, la empresa 3 obtiene una puntuación de un 71%. Existen categorías con una puntuación dispar, debido a las características de la empresa:
    - Entre el 90 y 100% existen las categorías relacionadas con la alta dirección, estabilidad, skills y tamaño del equipo. Además, otros principios como el desarrollo incremental y la puesta pronta en producción, o el hacer partícipe a los clientes finales en el desarrollo.
    - Entre 50 y 80, aparecen principios como los derivados de la toma de decisión de los desarrolladores, cambios reversibles.
    - Menor del 50%, la aceptación de la metodología. Esta tiene una puntuación del 46%, ya que ni los clientes ni los usuarios quieren el cambio de metodología, aunque tienen una separación de tiempos hecha.

- **Crystal Systems**: En consecuencia, de las características del proyecto, la organización pertenece a la familia de Crystal Clear L6. Esto se explica debido a que el tamaño de los equipos de desarrollo está entre 1 y 6 integrantes, pero las pérdidas por fallos pueden constituir la pérdida de la vida del proyecto. Según la metodología, esta familia no tiene las características propias del Agile, por lo que tiene una puntuación del 0%

- **Boehm y Turner**: Según este filtro de adaptación, la puntuación del filtro de adecuación es de un 58%. La puntuación en este caso, baja al tener mantener el principio de la criticidad, que tiene la puntuación mínima del 20%. En los demás casos, se mantiene la media vista en DSDM.

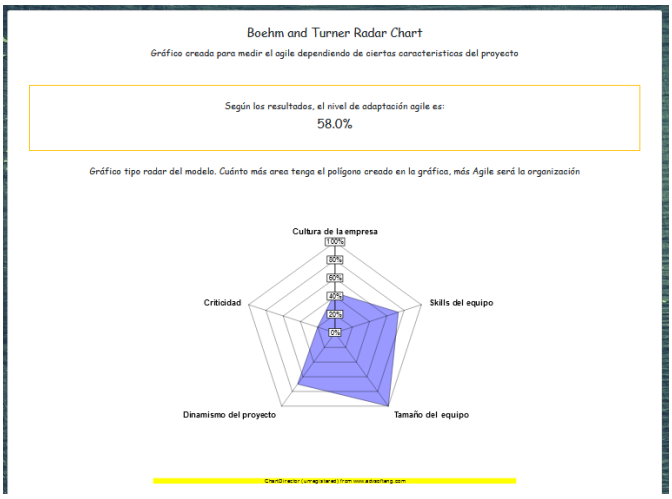


Fig 34 Gráfico radar de Boehm y Turner para la empresa 3

- **Dave Cohen**: El filtro de Dave Cohen marca una puntuación del 69% para la empresa. Esto se debe a que la mayoría de los principios de esta teoría tienen entre el 80 y el 100% de puntuación. Sin embargo, la puntuación baja incrementalmente debido a la puntuación mínima del 20 en el principio relacionado los cambios de organización, y las decisiones de los desarrolladores.

- **Organizational Filter**: En este caso el porcentaje de adecuación será del 72%. En esta ocasión, todas las categorías tienen unos porcentajes entre el 60 y 80%, pero sube sosteniblemente debido al apartado que define la comunicación del equipo de desarrollo, que sube un 100%.

- **PMI**: Sin embargo, la guía ágil de PMI obtiene puntuaciones en sus principios que son diferentes entre ellas, al igual que DSDM. Existen principios con el máximo de puntuación, como el tamaño del equipo o la relación entre el equipo y los clientes (un 93%), pero también existen principios con los puntuajes más bajos, como la confianza de la organización en los cambios.

4.4 Conclusiones después de la utilización del filtro

Después de la utilización de los filtros, podríamos concluir en que la empresa 2 es la más adecuada al Agile. La utilización de la aplicación no hace más que constatar lo que se ve desde la descripción de la empresa, que es una empresa adaptada en la totalidad a lo que es son las metodologías Agile.

Por otra parte, la empresa 1 aparece como la peor adecuada al Agile, debido a sus características. Aun así, veo a esta más adaptable al agile con cambios de organización que la empresa 3. Esto es porque la empresa 3, aunque por lo sacado por la aplicación está más adaptada al Agile y debería hacer menos cambios, tiene un obstáculo bastante más grande que todos los que tiene la empresa 1, que es la reticencia de los usuarios y la alta dirección al cambio de metodología. Por lo tanto, la empresa 1 debería hacer más cambios para adaptarse al agile, pero la dificultad de realizar estos cambios parece mayor para la empresa 3.

Esto quiere decir que, aunque la puntuación cuantitativa pueda ser un termómetro de como de adecuado se está con respecto al Agile, esto también depende del ambiente empresarial y organizativo del proyecto.

5. CONCLUSIONES

El desarrollo de la aplicación y la inclusión de los filtros de autores en esta ha permitido ver las semejanzas y diferencias entre diferentes tipos de metodologías ágiles.

Se ve que prácticamente todos los filtros de adecuación contienen apartados muy similares, más o menos extensos, pero buscando principalmente que las empresas contesten a apartados similares de una u otra forma. Esto es debido a que como era de esperar, todos los filtros contienen, implícita o explícitamente, cuestiones referentes a los 4 valores y 12 principios expuestos en el Manifiesto Agile, base de todas estas metodologías y filtros.

Sin embargo, se ve que cada uno de estos filtros da prioridad a algún valor o principio con respecto a otros, dando más importancia a ciertas características. Un ejemplo es el filtro de adecuación de Crystal Systems, en el que da prioridad únicamente a 3 características (tamaño del equipo, criticidad y prioridad) y ya con ello da un resultado, sin tener en cuenta otros parámetros importantes subrayados en el manifiesto Agile como pueden ser la relación con el cliente o el proceso del desarrollo del software.

Debido a esto, al incluirlos todos en la aplicación, hemos podido hacer una aplicación que se podría determinar como un “gran filtro de adecuación Agile”, ya que, cada uno de los filtros ágiles se complementa con los demás, subrayando y detallando valores y principios del Manifiesto Agile que otros no analizan. Además, ayuda que la aplicación valore de manera detallada tanto los casos en los que la empresa se adecúa al Agile tanto los casos que no lo hace. De una manera visual, permite ver los detalles de los puntos a mejorar por parte de estas empresas gracias a la señalización del resultado de las propias cuestiones. Esto ayuda a la empresa a saber específicamente que puntos tiene que mejorar para poder ser más adecuado al Agile.

Por otra parte, el feedback de las personas que realizaron el cuestionario reveló que la aplicación recogía perfectamente los principios de los filtros de adecuación en muy pocas preguntas. Creo que, simplificar estos cuestionarios, hace su aplicación más útil y fácil de usar que los cuestionarios construidos por los expertos.

5.1 Posibles usos de la aplicación y mejoras

El principal uso de esta aplicación es para la que se ha utilizado en este proyecto, que es servir como guía para las empresas a la hora de saber qué nivel de adecuación Agile tienen. Aun así, esta aplicación puede servir como herramienta para verificar datos a la hora de hacer consultorías o para obtener cualificaciones a nivel empresarial, por ejemplo, del PMI.

Aun así, tiene áreas de mejora. Por ejemplo, añadiendo más filtros de adecuación, como puede ser un filtro de adecuación para las metodologías SCRUM o XP, más utilizadas actualmente. Además, se podría mejorar la puntuación de las cuestiones dentro del cuestionario, por ejemplo, asignando prioridades a las preguntas o añadiendo más cuestiones sin perder la principal misión de la aplicación.

5.2 Planificación y presupuesto

La planificación que se hizo para el desarrollo del proyecto ha sido acorde linealmente a lo dispuesto a lo largo de esta memoria. Primeramente, se buscó información sobre lo que son las metodologías tradicionales y ágiles para tener un enfoque general de la situación. Seguidamente, se investigó sobre los filtros de adecuación existentes, para observar y concluir cuales de estos se iban a incluir en la aplicación. Una vez se sabían los filtros a incluir, se resumieron en el cuestionario, del que salieron las 27 preguntas. A continuación, se empezó con todo lo relacionado con el desarrollo de la aplicación, tanto las clases, la lógica y las vistas que conforman el esquema Modelo-Vista-Controlador. Cuando esta aplicación estuvo desarrollada, se siguió con la fase de recogida de resultados, haciendo que las tres empresas nombradas en esta memoria rellenaran el cuestionario, para así seguir con el análisis de los datos mostrados. Y, por último, se recopilaron todos los datos para escribir esta memoria y el resumen. Todo esto se recoge en este diagrama de Gantt:

Fecha de inicio	Fecha de finalización	Hito o actividad
01/06/2022	11/06/2022	Búsqueda de información general
12/06/2022	27/06/2022	Búsqueda de información sobre metodologías agiles

27/06/2022	04/07/2022	Búsqueda de información sobre filtros de adaptación
27/06/2022	17/07/2022	Elaboración del cuestionario
01/06/2022	17/07/2022	Esquematización de las vistas en la aplicación
01/07/2022	10/07/2022	Desarrollo de las clases de la aplicación
10/07/2022	10/08/2022	Desarrollo de la lógica de la aplicación
10/07/2022	10/08/2022	Desarrollo de las vistas de la aplicación
12/08/2022	19/08/2022	Entrega y resolución de cuestionarios a las empresas
19/08/2022	26/08/2022	Análisis de resultados
15/08/2022	09/09/2022	Elaboración de la memoria
22/08/2022	09/09/2022	Elaboración del resumen

TABLA 5 Planificación del proyecto

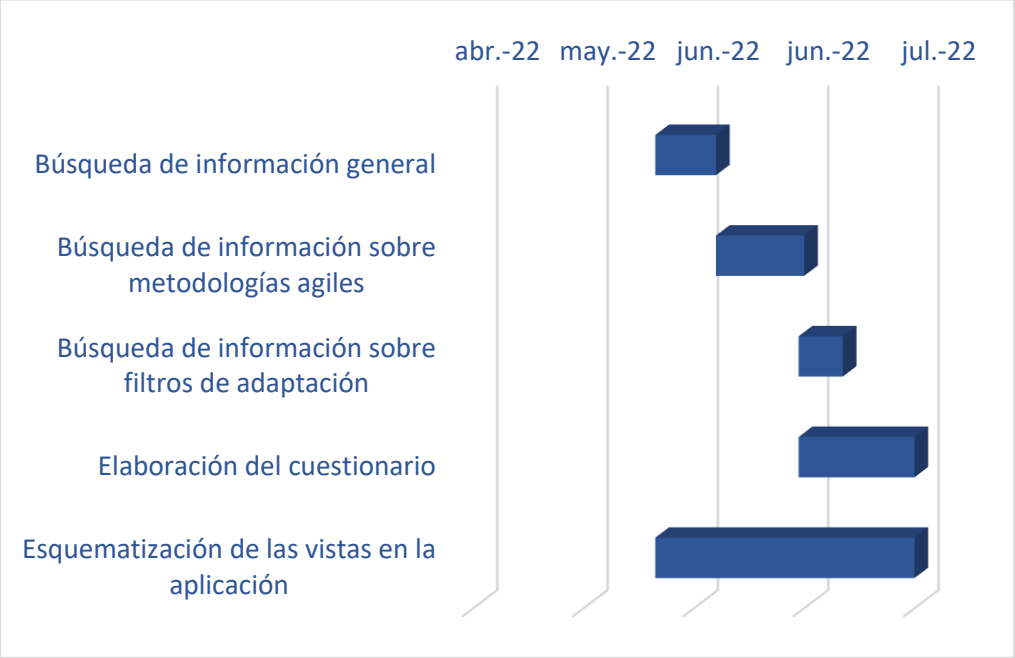


Fig 35 Diagrama de Gantt (1/3)

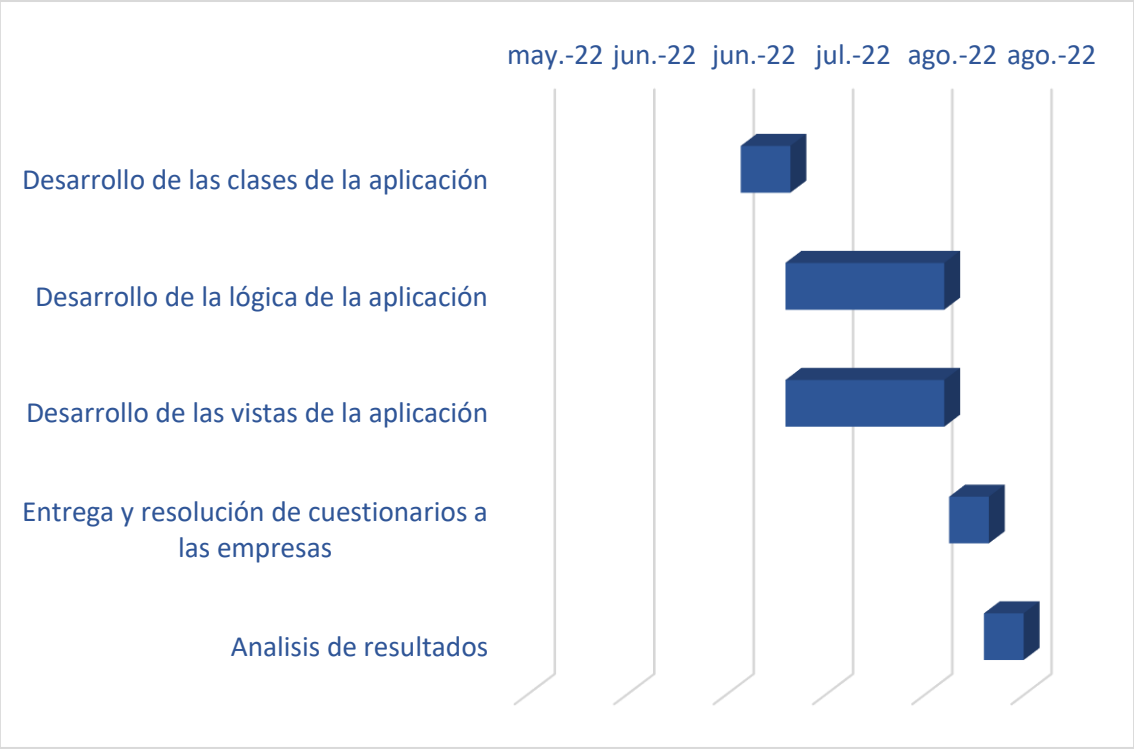


Fig 36 Diagrama de Gantt (2/3)

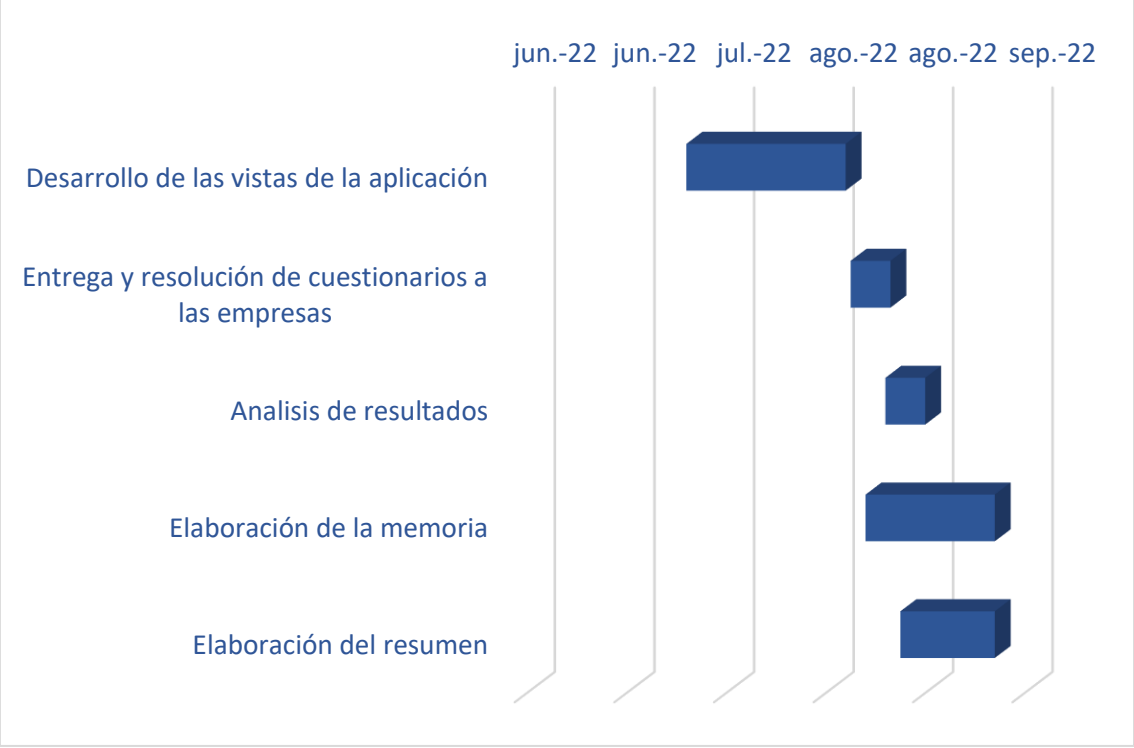


Fig 37 Diagrama de Gantt (3/3)

En cuanto al presupuesto del proyecto, el grueso del coste recae en la mano de obra, que consiste en el propio autor y diseñador del proyecto, y de un tutor-supervisor que sirve de ayuda para dudas.

Por otra parte, el coste del material únicamente cubrirá el coste del ordenador portátil para desarrollar la aplicación. Todos los softwares utilizados son open-sources y no hace falta desembolso en ellos

Por último, existen otros costes indirectos como la luz o la conexión a internet que se necesitan para el desarrollo de este proyecto. Todos los costes están desglosados en las siguientes tablas:

COSTE DE LA MANO DE OBRA			
Descripción	**Coste por hora (€ / h)**	**Total de horas (h)**	**Coste total (€)**
Ingeniero Junior – Autor del proyecto	**10**	**400**	**4000**
Ingeniero Senior – Supervisor del proyecto	**25**	**40**	**1000**
Total			**5000**

TABLA 6 Costes de mano de obra

COSTE DEL MATERIAL			
Descripción	**Coste por unidad (€ / u)**	**Total de unidades (u)**	**Coste total (€)**
Ordenador portátil (utilizado la mitad de la jornada)	**250**	**1**	**250**
Total			**250**

TABLA 7 Coste del material

COSTE INDIRECTO			
Descripción	**Coste al mes (€ / mes)**	**Total de horas de uso (h)**	**Coste total (€)**
Conexión a internet	**24**	**375**	**12,5**
Luz y corriente eléctrica	**60**	**440**	**36,6**
Total			**49,1**

TABLA 8 Coste indirecto

COSTE TOTAL	
Descripción	**Precio (€)**
Mano de obra	**5000**
Material	**250**
Coste indirecto	**49,1**
Total	**5299,1**

TABLA 9 Coste total del proyecto

6. BIBLIOGRAFÍA

[1] Hewlett Packard Enterprise, «Agile is the new normal,» 2015. [En línea]. Available: https://softwaretestinggenius.com/docs/4aa5-7619.pdf. [Último acceso: 17 08 2021].

[2] L. M. E. Garcés Guayta Lucas Rogerio, «EVOLUCIÓN DE LAS METODOLOGÍAS DE DESARROLLO DE LA INGENIERÍA DE SOFTWARE EN EL PROCESO LA INGENIERÍA DE SISTEMAS SOFTWARE,» 12 Diciembre 2013. [En línea]. Available: https://incyt.upse.edu.ec/ciencia/revistas/index.php/rctu/article/view/29/28. [Último acceso: 26 Agosto 2022].

[3] H. V. C. J. D. C. Bryan Molina Montero, «Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software,» 2018. [En línea]. Available: https://www.researchgate.net/profile/Harry-Vite-Cevallos/publication/327537074_Metodologias_agiles_frente_a_las_tradicionales_en_el_proceso_de_desarrollo_de_software/links/5b942061a6fdccfd542a2b13/Metodologias-agiles-frente-a-las-tradicionales-en-el-proce. [Último acceso: 11 08 2022].

[4] M. McCormick, «Waterfall vs. Agile Methodology,» 2012. [En línea]. Available: http://www.mccormickpcs.com/images/Waterfall_vs_Agile_Methodology.pdf. [Último acceso: 16 08 2022].

[5] IONOS, «El modelo en cascada: desarrollo secuencial de software,» 11 03 2019. [En línea]. Available: https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/el-modelo-en-cascada/. [Último acceso: 07 08 2022].

[6] H. Kniberg, «Toyota’s journey from Waterfall to Lean software development,» 16 03 2010. [En línea]. Available: https://blog.crisp.se/2010/03/16/henrikkniberg/1268757660000. [Último acceso: 15 08 2022].

[7] Agile Society, «Manifiesto por el Desarrollo Ágil de Software,» [En línea]. Available: https://agilemanifesto.org/iso/es/manifesto.html.

[8] V. Singh, «Crystal Method in Agile,» 2021. [En línea]. Available: https://www.toolsqa.com/agile/crystal-method/.

[9] «Figueroa, R. G., Solís, C. J., & Cabrera, A. A.,» *Metodologías tradicionales vs. metodologías ágiles.Universidad Técnica Particular de Loja, Escuela de Ciencias de la Computación, 9(1), 1-10.,* 2008.

[10] S. R. J. I. G. Abdullahi Sani Adila Firdaus, «A Review on Software Development Security Engineering using Dynamic System Method (DSDM),» *International Journal of Computer Applications,* vol. 69, nº 25, pp. 0975-8887, 25 05 2013.

[11] M. Griffith, «Agile Suitability Filters,» 2007.

[12] R. T. Barry Boehm, Balancing Agility and Discipline: A Guide for the Perplexed.

[13] D. Cohen, Advanced in Computers, 2004.

[14] I. Project Management Institute, GUÍA PRÁCTICA DE ÁGIL, Newtown Square, Pennsylvania 19073-3299 EE.UU., 2017.

[15] «Instalar Eclipse e incluir Tomcat en Eclipse,» 9 2012. [En línea]. Available: https://jesusfernandeztoledo.com/instalar-eclipse-e-incluir-tomcat-en-eclipse/.

ANEXO I: TABLAS DE PROYECTO

N.º	Pregunta	Puntuación Empresa 1	Puntuación Empresa 2	Puntuación Empresa 3
1	Los clientes de la aplicación están dispuestos a cambios de organización	5	5	1
2	La organización vería bien un cambio de metodología	4	4	1
3	Hay una separación de tiempos en la organización	4	5	5
4	Existe confianza en el equipo de desarrollo por parte de los clientes	5	5	4
5	El equipo de desarrollo tiene total poder de decisión	2	5	4
6	La alta dirección estaría dispuesta a dar más poder de decisión al equipo de desarrollo	1	4	1
7	Los usuarios finales/clientes participan en los procesos de desarrollo (Haciendo pruebas antes de subidas a producción, sugiriendo mejoras, etc.)	1	5	5
8	El equipo de desarrollo sabe en todo momento que es lo que quiere el cliente	1	5	4
9	El equipo de desarrollo tiene acceso a los entornos de producción	4	5	4
10	El rango de rotación del equipo es bajo	2	3	5
11	¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?	<50%	<60%	<25%

12	¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?	<75%	<50%	<50%
13	¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?	<25%	<25%	<75%
14	¿De cuantas personas se compone el equipo de desarrollo?	81-200	1-6	1-6
15	Dentro de la organización se trabaja en iteraciones	5	5	5
16	El porcentaje de requerimientos cambiados al mes es bajo	3	4	3
17	Se utilizan tecnologías que apuesten por la pronta entrega a cliente (empaquetados rápidos del producto, subidas a producción de pocas horas…)	5	5	4
18	El producto entregado se puede construir y evaluar en porciones	2	4	5
19	Si existe un cambio, este se puede revertir rápidamente de producción	1	4	1
20	¿Si existe un cambio de alcance dentro del desarrollo de una funcionalidad, que supondría?	Pérdida de tiempo	Nada	Todas las anteriores
21	Se hacen pruebas durante el proceso de desarrollo	4	4	3

22	El periodo de pruebas una vez subido a producción es acorde a las funcionalidades creadas	1	5	4
23	El tiempo de puesta a producción asumido es acorde a las funcionalidades creadas	3	4	2
24	¿Si existe un fallo en la subida a producción, que supondría?	Nada	Nada	Todas las anteriores
25	¿Si existe un fallo en producción que supondría?	Pérdida de tiempo	Nada	Todas las anteriores
26	La comunicación entre personas del mismo equipo de desarrollo es fluida	3	5	5

Anexo I Puntuaciones de las empresas

PREGUNTA\FILTRO	DSDM Suitability	Team Size Factors (Crystal Systems)	Boehm and Turner – Radar Chart	Dave Cohen's Agile Factors	The Organizational Suitability Filter	PMI - Agile practice guide - Apéndice
Los clientes de la aplicación están dispuestos a cambios de organización,	X		X	X	X	X
La organización vería bien un cambio de metodología,	X		X		X	
Hay una separación de tiempos en la organización,	X				X	
Existe confianza en el equipo de desarrollo por parte de los clientes,	X	X	X	X		X
El equipo de desarrollo tiene total poder de decisión,	X		X	X	X	X
La alta dirección estaría dispuesta a dar más poder de decisión al equipo de desarrollo,	X		X	X	X	X
Los usuarios finales/clientes participan en los procesos de desarrollo (Haciendo pruebas antes de subidas a producción, sugiriendo mejoras, etc.),	X			X	X	X
El equipo de desarrollo sabe en todo momento que es lo que quiere el cliente,	X			X	X	X
El equipo de desarrollo tiene acceso a los entornos de producción,	X			X	X	X

El rango de rotación del equipo es bajo,	X					X
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?,	X		X	X	X	X
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?,	X		X	X	X	X
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?,	X		X	X	X	X
¿De cuantas personas se compone el equipo de desarrollo?,	X	X	X	X		X
Dentro de la organización se trabaja en iteraciones,	X		X		X	
El porcentaje de requerimientos cambiados al mes es bajo,			X		X	X
Se utilizan tecnologías que apuesten por la pronta entrega a cliente (empaquetados rápidos del producto, subidas a producción de pocas horas…),	X		X		X	X
El producto entregado se puede construir y evaluar en porciones,						X
Si existe un cambio, este se puede revertir rápidamente de producción,	X					X
¿Si existe un cambio de alcance dentro del desarrollo de una funcionalidad, que supondría?,		X				X
Se hacen pruebas durante el proceso de desarrollo,		X	X		X	

El periodo de pruebas una vez subido a producción es acorde a las funcionalidades creadas,	X		X			X
El tiempo de puesta a producción asumido es acorde a las funcionalidades creadas,			X			
¿Si existe un fallo en la subida a producción, que supondría?,		X	X			X
¿Si existe un fallo en producción que supondría?,		X	X			X
La comunicación entre personas del mismo equipo de desarrollo es fluida,				X	X	
La comunicación entre desarrolladores y clientes/alta dirección es fluida				X	X	X

Anexo II Relación cuestión - filtro de adecuación

Pregunta\Principio	Aceptación de la metodología agile antes de empezar a trabajar	La toma de decisiones está en los usuarios y en los developers	Compromiso de la alta dirección de hacer partícipes a los usuarios finales	Acceso de los developers a entornos de producción	Estabilidad del equipo	Skills del equipo	Tamaño del equipo	Desarrollo incremental y pronta puesta en producción	Cambios reversibles
Los clientes de la aplicación están dispuestos a cambios de organización	X								
La organización vería bien un cambio de metodología	X								
Hay una separación de tiempos en la organización	X								
Existe confianza en el equipo de desarrollo por parte de los clientes		X							
El equipo de desarrollo tiene total poder de decisión		X							
La alta dirección estaría dispuesta a dar más poder de decisión al equipo de desarrollo		X							
Los usuarios finales/clientes participan en los procesos de			X						

desarrollo (Haciendo pruebas antes de subidas a producción, sugiriendo mejoras, etc.)									
El equipo de desarrollo sabe en todo momento que es lo que quiere el cliente			X						
El equipo de desarrollo tiene acceso a los entornos de producción				X					
El rango de rotación del equipo es bajo					X				
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?						X			
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?						X			
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?						X			

¿De cuantas personas se compone el equipo de desarrollo?							X		
Dentro de la organización se trabaja en iteraciones								X	
Se utilizan tecnologías que apuesten por la pronta entrega a cliente (empaquetados rápidos del producto, subidas a producción de pocas horas…)								X	
Si existe un cambio, este se puede revertir rápidamente de producción									X
El periodo de pruebas una vez subido a producción es acorde a las funcionalidades creadas									X

Anexo III Relación cuestión - principio DSDM

Pregunta\Principio	Skills del equipo	Dinamismo del proyecto	Tamaño del equipo	Criticidad	Cultura de la empresa
Los clientes de la aplicación están dispuestos a cambios de organización					X
La organización vería bien un cambio de metodología					X
Existe confianza en el equipo de desarrollo por parte de los clientes					X
El equipo de desarrollo tiene total poder de decisión					X
La alta dirección estaría dispuesta a dar más poder de decisión al equipo de desarrollo					X
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?	X				
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?	X				
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?	X				
¿De cuantas personas se compone el equipo de desarrollo?			X		
Dentro de la organización se trabaja en iteraciones		X			
El porcentaje de requerimientos cambiados al mes es bajo		X			
Se utilizan tecnologías que apuesten por la pronta entrega a cliente (empaquetados rápidos del producto, subidas a producción de pocas horas)		X			
Se hacen pruebas durante el proceso de desarrollo		X			

El periodo de pruebas una vez subido a producción es acorde a las funcionalidades creadas		X			
El tiempo de puesta a producción asumido es acorde a las funcionalidades creadas		X			
¿Si existe un fallo en la subida a producción, que supondría?				X	
¿Si existe un fallo en producción que supondría?				X	

Anexo IV Relación cuestión - principio Boehm-Turner

Pregunta\Principio	**La cultura de la organización debe ser basada en la negociación**	**Se debe confiar en los equipos**	**Poca gente, pero más competente**	**La organización debe existir con las decisiones de los desarrolladores**	**El entorno de la organización debe ayudar a la rápida comunicación de los miembros del equipo**
Los clientes de la aplicación están dispuestos a cambios de organización	X				
Existe confianza en el equipo de desarrollo por parte de los clientes		X			
El equipo de desarrollo tiene total poder de decisión				X	

La alta dirección estaría dispuesta a dar más poder de decisión al equipo de desarrollo				X	
Los usuarios finales/clientes participan en los procesos de desarrollo (Haciendo pruebas antes de subidas a producción, sugiriendo mejoras, etc.)	X				
El equipo de desarrollo sabe en todo momento que es lo que quiere el cliente		X			
El equipo de desarrollo tiene acceso a los entornos de producción		X			
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?			X		
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?			X		
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?			X		
¿De cuantas personas se compone el equipo de desarrollo?			X		

La comunicación entre personas del mismo equipo de desarrollo es fluida					X
La comunicación entre desarrolladores y clientes/alta dirección es fluida					X

Anexo V Relación cuestión - principio de Dave Cohen Agile Factors

Pregunta\Principio	**Usuarios/Clientes de la organización -> ¿Entienden lo que es el desarrollo Agile?**	**Organización -> ¿Está dispuesto a integrar la metodología Agile?**	**Equipo de desarrollo -> ¿Están preparados para trabajar en Agile?**	**Técnicas de desarrollo -> ¿Se utilizan técnicas para trabajar en Agile?**	**Comunicación -> ¿La comunicación dentro de la organización es apropiada?**
Los clientes de la aplicación están dispuestos a cambios de organización	X				
La organización vería bien un cambio de metodología		X			
Hay una separación de tiempos en la organización		X			

El equipo de desarrollo tiene total poder de decisión		X			
La alta dirección estaría dispuesta a dar más poder de decisión al equipo de desarrollo		X			
Los usuarios finales/clientes participan en los procesos de desarrollo (Haciendo pruebas antes de subidas a producción, sugiriendo mejoras, etc.)	X				
El equipo de desarrollo sabe en todo momento que es lo que quiere el cliente			X		
El equipo de desarrollo tiene acceso a los entornos de producción			X		
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?			X		
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?			X		

¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?			X		
Dentro de la organización se trabaja en iteraciones		X			
El porcentaje de requerimientos cambiados al mes es bajo		X			
Se utilizan tecnologías que apuesten por la pronta entrega a cliente (empaquetados rápidos del producto, subidas a producción de pocas horas)				X	
El periodo de pruebas una vez subido a producción es acorde a las funcionalidades creadas				X	
La comunicación entre personas del mismo equipo de desarrollo es fluida					X
La comunicación entre desarrolladores y clientes/alta dirección es fluida					X

Anexo VI Relación cuestión - principio de Organizational Suitability Filter

Pregunta\Cuestión	Tamaño del equipo	Experiencia del equipo	Relación equipo - clientes	Confianza de clientes	Confianza de la organización	Capacidad de decisión	Cambios intrínsecos del proyecto	Criticidad del proyecto	Entrega del proyecto
El equipo de desarrollo tiene total poder de decisión						X			
La alta dirección estaría dispuesta a dar más poder de decisión al equipo de desarrollo					X				
Los usuarios finales/clientes participan en los procesos de desarrollo (Haciendo pruebas antes de subidas a producción, sugiriendo mejoras, etc.)			X						
El equipo de desarrollo sabe en todo momento que es lo que quiere el cliente			X						
El equipo de desarrollo tiene acceso a los entornos de producción				X	X				
El rango de rotación del equipo es bajo		X							
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento de principiante de la aplicación?		X							
¿Qué porcentaje del equipo de desarrollo tiene un conocimiento intermedio de la aplicación?		X							

¿Qué porcentaje del equipo de desarrollo tiene un conocimiento experto de la aplicación?		X							
¿De cuantas personas se compone el equipo de desarrollo?	X								
El porcentaje de requerimientos cambiados al mes es bajo									
Se utilizan tecnologías que apuesten por la pronta entrega a cliente (empaquetados rápidos del producto, subidas a producción de pocas horas?)									X
Si existe un cambio, este se puede revertir rápidamente de producción							X		
Si existe un cambio de alcance dentro del desarrollo de una funcionalidad, que supondría?							X		
El periodo de pruebas una vez subido a producción es acorde a las funcionalidades creadas								X	X
¿Si existe un fallo en la subida a producción, que supondría?								X	
¿Si existe un fallo en producción que supondría?								X	
La comunicación entre desarrolladores y clientes/alta dirección es fluida			X						

Anexo VII Relación cuestión - principio de la guía ágil PMI