ESTUDIO COMPARATIVO DE METODOLOGÍAS, HERRAMIENTAS Y WIKI DE SOPORTE PARA LA GESTIÓN DE PROYECTOS DE DESARROLLO DE SOFTWARE

Jesús García Navarro
Grado en Ingeniería Informática
Gestión de proyectos

Consultor: Xavier Martínez Munné
Responsable: Atanasi Daradoumis Haralabus

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Estudio comparativo de metodologías, herramientas y wiki de soporte para la gestión de proyectos de desarrollo software*
Nombre del autor:	*Jesús García Navarro*
Nombre del consultor/a:	*Xavier Martínez Munné*
Nombre del PRA:	*Atanasi Daradoumis Haralabus*
Fecha de entrega (mm/aaaa):	*01/2018*
Titulación:	*Grado en Ingeniería Informática*
Área del Trabajo Final:	*Gestión de proyectos*
Idioma del trabajo:	*Castellano*
Palabras clave	*Gestión* *Ágil* *Wiki*

Resumen del Trabajo:

En la actualidad, las empresas apuestan cada vez más por la aplicación de las metodologías ágiles que por el uso de las metodologías tradicionales para la gestión de proyectos de desarrollo de software. Por ello, el presente trabajo pretende cubrir la necesidad de analizar qué modelo de gestión es más recomendable dependiendo del tipo de proyecto y, a su vez, mostrar sus ventajas e inconvenientes.

La idea es hacer un estudio de las principales herramientas que existen en el mercado para la gestión de proyectos y así poder proporcionar un análisis comparativo sobre cuáles de ellas se adapta mejor a las metodologías ágiles. También, se valorará si es suficiente el uso de un sitio wiki para la gestión de este tipo de proyectos.

Así pues, el objetivo final del presente Trabajo Fin de Grado es el de realizar una investigación sobre las distintas herramientas de gestión de proyectos software, incluyendo el uso de sitios wiki, focalizando en las herramientas con soporte para metodología ágiles, y así proporcionar una guía de recomendaciones antes de elegir la herramienta a usar para la administración del proyecto.

Por último, se realiza una simulación empírica sobre una iteración de un proyecto aplicando la metodología SCRUM en un sitio wiki de soporte.

Abstract:

Nowadays, more and more companies are choosing Agile Methodologies over Traditional for Software project management. For this reason, the current work is intended to cover the needing of analyze which of this management method is better to use in a project and show its advantages.

The idea is to make an analysis of the main project management tools available in market and get to know and to get a comparative study to know which tools is more focused for Agile Methodology. Also, we figure out if a wiki site is enough to management an Agile project.

In summary, the main target is to research the main project management tools focused in Agile Methodologies even a wiki site, to provide a recommendation guide to make the better choice of the management tool.

Finally, we make a simulation of a SCRUM iteration managed by a wiki site.

Índice

1. Introducción 1
1.1 Contexto y justificación del Trabajo 1
1.2 Objetivos del Trabajo 1
Objetivo principal 1
Objetivos secundarios 2
1.3 Enfoque y método seguido 2
1.4 Planificación del Trabajo 2
Distribución de horas 2
Fechas claves 3
Diagrama de Gantt 3
1.5 Breve sumario de productos obtenidos 4
1.6 Breve descripción de los otros capítulos de la memoria 5

2. Gestión de proyectos de desarrollo software 6
2.1. Influencia de las metodologías en la gestión de proyectos 7
2.2. Factores de éxito 8
2.3. Gestión de proyectos predictiva 9
Metodologías de desarrollo tradicionales 10
2.3.1 RUP 11
2.3.2 MSF 11
2.3.3 Iconix 12
2.4. Gestión de proyectos evolutiva 12
Manifiesto ágil 13
Metodologías de desarrollo ágiles 13
2.4.1 SCRUM 14
Roles 15
Artefactos 15
Eventos 16
Workflow de Scrum 17
2.4.2 Metodología Lean 18
2.4.3 eXtreme Programming 19
Roles 19
Valores 19
Fases 20
Reglas 20
Pair Programming 20
2.4.4 Otros métodos y técnicas 21
a) Dynamic Systems Development Method (DSDM) 21
Fases 21
Roles 21
Principios 22
b) Técnica visual: Kanban 22
2.5. Mapa conceptual de la gestión de proyectos de desarrollo software 23
2.6. Estudio comparativo de las distintas metodologías 24
Características principales de cada metodología. 24
Ventajas y Desventajas 25
¿Qué tipo de metodología usar: tradicional o ágil? 25
¿Qué metodología ágil emplear? 27

3. Herramientas de gestión de proyectos ágiles 30
3.1. Introducción 30
Características mínimas necesarias de un APMS 31
Beneficios de un AMPS 31
3.2. Herramientas mejor valoradas 31
3.2.1. Atlassian / Jira 34
Características principales 34
3.2.2. Microsoft: Team Foundation Server (TFS) 35
Características principales 36
3.2.3. VersionOne 37
Características principales 37
3.2.4. Rally (CA Technologies) 38
3.2.5. Otras herramientas ágiles open source 39
3.3. Cuadro comparativo de las distintas herramientas 40

4. Creación de sitio wiki de soporte 43
4.1 Estructura del sitio wiki 43
4.2 Creación de plantillas 47
4.2.1. Product Backlog 48
a) Historias de usuario 49
b) Fechas de entrega 49
4.2.2. Sprints 50
a) Sprint Retrospective 50
b) Sprint Backlog 51
c) Sprint review 51
4.2.3. Gráfico Burndown 52
4.2.4. Otras pantallas 53
4.3. Crear plantilla completa del proyecto 53
4.4. Añadir comentarios 54
4.5. Programar eventos 55
4.6. Agregar miembros 56
4.6.1. Agregar usuarios de forma masiva 57
4.7. Asignar permisos y roles 57

5. Simulación de la gestión de un proyecto 58
5.1. Descripción del proyecto 58
5.2. Selección de metodología 59
5.3. Selección de la herramienta de gestión 61
5.4. Creación del wiki de soporte del proyecto 63
5.5. Simulación Kick-off 64
5.6. Simulación Product Backlog 65
5.5. Simulación Sprint 01 (Día 7) 67

6. Conclusiones 68
6.1. Conclusión general 68
6.2. Logro de objetivos 69
6.3. Seguimiento de la planificación y metodología 69
6.4. Líneas de trabajo futuro 69

7. Glosario 70

8. Bibliografía 71

9. Webgrafía 74

10. Anexos 75
ANEXO 1. Técnicas de estimación de tareas 75
ANEXO 2. Manual Wikispace 76
ANEXO 3. Pliego de prescripciones técnicas 77

Lista de figuras

Figura 1 Estimated project-oriented job openings, 2010-2020. 1
Figura 2 Distribución de horas..... 2
Figura 3 Diagrama de Gantt..... 4
Figura 4 Ciclo de vida de la gestión de proyectos..... 6
Figura 5 Porcentaje de éxito según Informe del Caos 2015..... 6
Figura 6 Prácticas y metodologías 7
Figura 7 Desarrollo en cascada..... 10
Figura 8 Incremento iterativo/continuo 14
Figura 9 Ciclo iterativo scrum..... 15
Figura 10 Ficha sinóptica scrum..... 16
Figura 11 Fases XP..... 20
Figura 12 Roles en DSDM 21
Figura 13 Tablero kanban 22
Figura 14 Mapa conceptual de la gestión de proyectos 23
Figura 15 Características SaaS y On-premise 30
Figura 16 Cuadrante mágico para ADLM. 32
Figura 17 Puntuación de herramientas por usuarios..... 33
Figura 18 Uso y recomendación de herramientas ágiles 33
Figura 19 Tablero scrum en Jira..... 35
Figura 20 Interfaz TSF 36
Figura 21 TSF backlog board. 36
Figura 22 VersionOne backlog board..... 38
Figura 23 CA Rally Board..... 39
Figura 24 Cambiar estructura del sitio wiki..... 43
Figura 25 Panel de administración de wikispace – Estilo Classroom..... 44
Figura 26 Estructura básica..... 44
Figura 27 Crear un nuevo proyecto..... 44
Figura 28 Crear artefactos..... 45
Figura 29 Asociar miembros..... 45
Figura 30 Kick-off 46
Figura 31 Product Backlog 46
Figura 32 Página inicial de un Sprint..... 46
Figura 33 Estructura de pantallas de la plantilla wiki..... 47
Figura 34 Nueva plantilla..... 47
Figura 35 Popup de creación de plantilla 48
Figura 36 Plantilla creada..... 48
Figura 37 Plantilla para Product backlog..... 48
Figura 38 Plantilla para las historias de usuario..... 49
Figura 39 Plantilla para las fechas de entrega 49
Figura 40 Pantalla principal de un Sprint..... 50
Figura 41 Pantalla de la Retrospective..... 50
Figura 42 Plantilla para los Sprints..... 51
Figura 43 Pantalla del Sprint Review 51
Figura 44 Insertar hoja de cálculo 52
Figura 45 Embeber el código HTML..... 52
Figura 46 Pantalla Burndown Chart 53

Figura 47 Exportar wiki 54
Figura 48 Estructura de carpetas del sitio wiki 54
Figura 49 Iniciar una discusión 54
Figura 50 Escribir el comentario 55
Figura 51 Iniciar evento 55
Figura 52 Confirmar evento 55
Figura 53 Agregar miembros al sitio wiki 56
Figura 54 Añadir miembros al proyecto 56
Figura 55 Creación masiva de usuarios 57
Figura 56 Asignar permisos 57
Figura 57 Creación del proyecto MiFeria 64
Figura 58 Página de Kick Off 64
Figura 60 Página de historias de usuario 65
Figura 61 Página de tareas y estimación 66
Figura 62 Página de fechas de entregas 66
Figura 63 Página del Sprint Backlog en el 7º día 67
Figura 64 Burndown Chart del Sprint-1 en el séptimo día 67

1. Introducción

1.1 Contexto y justificación del Trabajo

Los recientes estudios llevados a cabo por el **PMI**[1] predicen que durante la actual década se crearán 15,7 millones de puestos de trabajo orientados a la gestión de proyectos (PMI, 2013). Según la firma de investigación en mercados tecnológicos, **Gartner Inc.**: "los métodos ágiles de desarrollo serán utilizados en el 80% de todos los proyectos de desarrollo software" (Gartner Inc., 2017). También, los estudios del **PMI** han demostrado que el uso de la agilidad se ha triplicado en los últimos años.

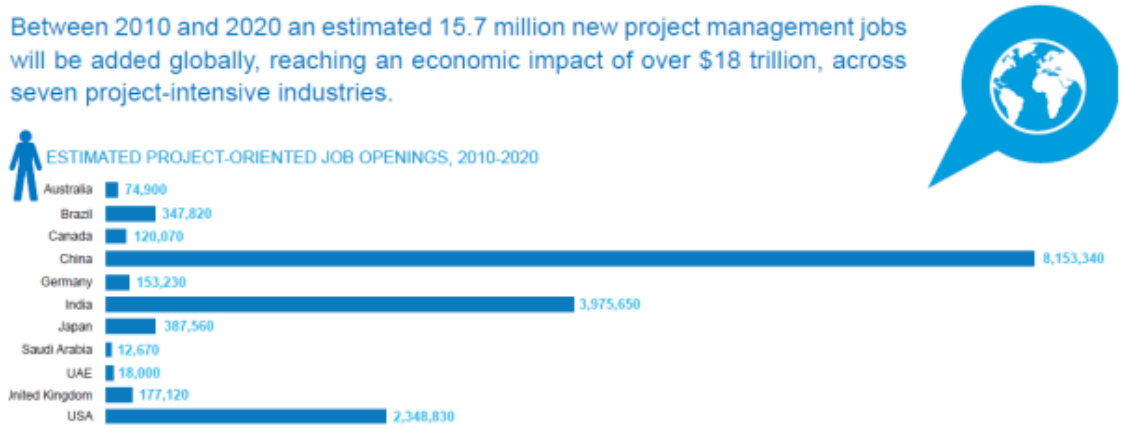


Figura 1 Estimated project-oriented job openings, 2010-2020. (PMI, 2013)

Siguiendo estas predicciones, el presente **TFG**[2] pretende analizar las principales herramientas de gestión de proyectos, centrándose en los marcos de desarrollo ágiles más usados en la actualidad, para proporcionar a la comunidad científica una guía para la correcta elección y aplicación de dichas herramientas, incluyendo una simulación en un sitio wiki. En los primeros capítulos se realizará una introducción a la gestión de proyectos, así como a sus distintas metodologías tradicionales y ágiles, haciendo hincapié en el modelo **Scrum** y posteriormente, se valorarán las distintas herramientas de gestión de proyectos, así como si es posible llevar el control de un proyecto solamente con un sitio wiki.

1.2 Objetivos del Trabajo

Objetivo principal

- Crear una guía para facilitar la correcta elección entre las principales herramientas de gestión de proyectos existentes en el mercado acorde a los proyectos de desarrollo software.
- Desarrollar un sitio wiki para la gestión de un proyecto y valorar si puede ser usado con garantías como una herramienta de gestión de proyectos.

[1] Project Management Institute
[2] Trabajo fin de Grado

Objetivos secundarios

- Realizar una tabla o gráfico comparativo entre las metodologías tradicionales y las metodologías ágiles.

1.3 Enfoque y método seguido

El presente **TFG**, al tener un gran contenido de investigación, la estrategia seguida es la de recopilar la mayor cantidad de información de información posible acerca de los distintos productos a estudiar, para obtener el máximo conocimiento posible. Estos datos se recogen tras evaluar lo que ofrecen las distintas herramientas y comprobar las opiniones de expertos en el sector, así como de los usuarios.

Por último, se realiza un análisis exhaustivo y una pequeña simulación para llegar a una conclusión lo más acertada posible, y de este modo adecuar la mejor herramienta y metodología de gestión según el proyecto. Se descarta una prueba empírica con las distintas herramientas de administración de proyectos, ya que consumiría bastante coste y tiempo, debido a que la mayoría de software es de pago y se tendrían que aplicar a una gran variedad de proyectos.

1.4 Planificación del Trabajo

Distribución de horas

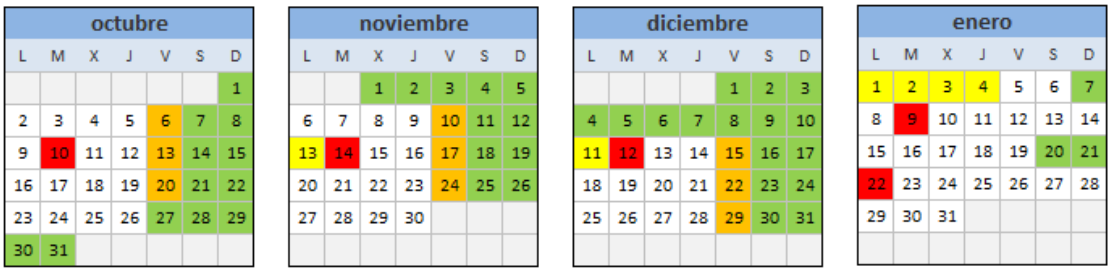


Figura 2 Distribución de horas

	HORAS	JORNADAS	TOTAL
	6	42	252
	4	9	36
	2	6	12
	FECHAS CLAVES		
			300 HORAS

Fechas claves

HITO	FECHA	HORAS DEDICADAS
PEC 1: Plan de trabajo	10/10/2017	22
PEC 2	14/11/2017	110
PEC 3	12/12/2017	94
ENTREGA FINAL	09/01/2018	62
DEFENSA VIRTUAL	29/01/2018	12

Diagrama de Gantt

Id	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
1	**PEC 1**	**10 días**	**mié 27/09/17**	**mar 10/10/17**		
2	Descripción del TFG	2 días	mié 27/09/17	jue 28/09/17		
3	Objetivos	2 días	vie 29/09/17	lun 02/10/17	2	
4	Enfoque y método a seguir	1 día	mar 03/10/17	mar 03/10/17	3	
5	Planificación del trabajo	3 días	mié 04/10/17	vie 06/10/17	4	
6	Sumario y descripción de capítulos	1 día	lun 09/10/17	lun 09/10/17	5	
7	Glosario, Bibliografía y Tabla de figuras	1 día	mar 10/10/17	mar 10/10/17	6	
8	Hito 1 - Entrega PEC 1	1 día	mar 10/10/17	mar 10/10/17	7	
9	**PEC 2**	**31 días**	**mié 11/10/17**	**mar 14/11/17**		**Documentación y selección de herramientas**
10	Introducción a la Gestión de proyectos	4 días	mié 11/10/17	lun 16/10/17	8	
11	Metodología Tradicional	4 días	mar 17/10/17	vie 20/10/17	10	
12	Metodologías Ágiles	4 días	lun 23/10/17	jue 26/10/17	11	
13	Elección de herramientas a estudiar	2 días	vie 27/10/17	sáb 28/10/17	12	
14	Investigación en la red	4 días	dom 29/10/17	mié 01/11/17	13	
15	Búsqueda de opiniones	3 días	jue 02/11/17	sáb 04/11/17	14	
16	Estudio comparativo	8 días	dom 05/11/17	dom 12/11/17	15	
17	Glosario, Bibliografía y Tabla de figuras	1 día	lun 13/11/17	lun 13/11/17	16	
18	Hito 2 - Entrega PEC 2	1 día	mar 14/11/17	mar 14/11/17	17	
19	**PEC 3**	**28 días**	**mié 15/11/17**	**mar 12/12/17**		**Análisis de Herramientas y wiki**
20	Tabla resultado final	6 días	mié 15/11/17	lun 20/11/17	18	
21	Creación de sitio wiki de soporte	3 días	mar 21/11/17	jue 23/11/17	20	
22	Configuración sitio wiki	6 días	vie 24/11/17	mié 29/11/17	21	
23	Simulación de Sprint	8 días	jue 30/11/17	jue 07/12/17	22	
24	Conclusión final	2 días	vie 08/12/17	sáb 09/12/17	23	
25	Anexos	1 día	dom 10/12/17	dom 10/12/17	24	
26	Glosario, Bibliografía y Tabla de figuras	1 día	lun 11/12/17	lun 11/12/17	25	
27	Hito 3 - Entrega PEC 3	1 día	mar 12/12/17	mar 12/12/17	26	
28	**ENTREGA FINAL**	**28 días**	**mié 13/12/17**	**mar 09/01/18**	**19**	**Memoria y Defensa**
29	Revisión final de la memoria	10 días	mié 13/12/17	vie 22/12/17	27	
30	Preparación de la presentación virtual	10 días	sáb 23/12/17	lun 01/01/18	29	
31	Autoinforme de competencias	7 días	mar 02/01/18	lun 08/01/18	30	
32	Hito 4 - Entrega Final	1 día	mar 09/01/18	mar 09/01/18	31	
33	**DEFENSA VIRTUAL**	**3 días**	**sáb 20/01/18**	**lun 22/01/18**		
34	Contestar preguntas	3 días	sáb 20/01/18	lun 22/01/18		

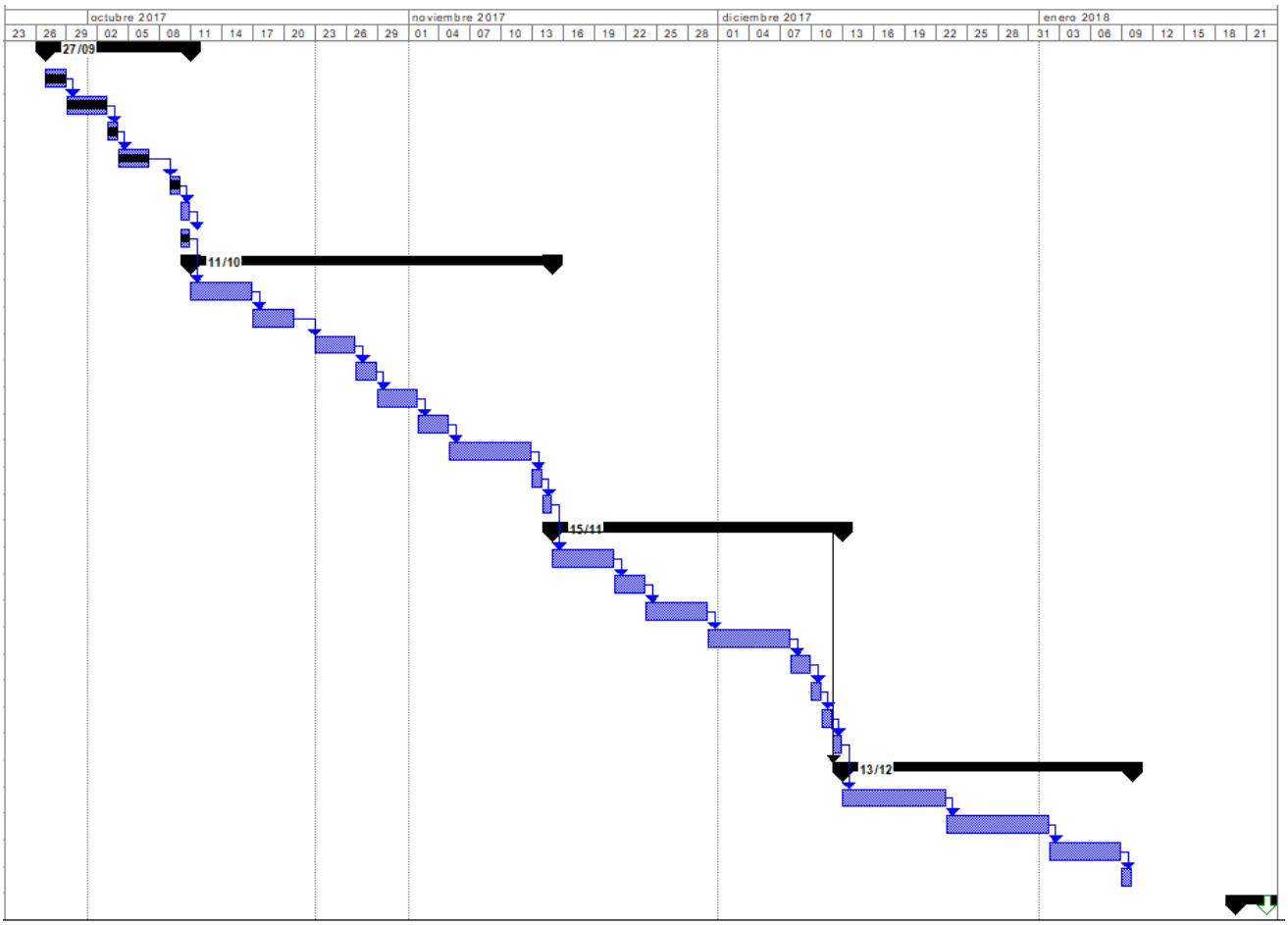


Figura 3 Diagrama de Gantt

1.5 Breve sumario de productos obtenidos

- Formulario y método para conocer la mejor metodología de gestión de proyectos que se puede aplicar a un proyecto en concreto.
- Formulario y método para conocer de entre las principales herramientas de gestión ágil cuál se adapta mejor a un proyecto concreto.
- Plantilla de sitio wiki con la estructura básica para iniciar la gestión de un proyecto con Scrum.
- Sitio wiki con la simulación de un estadio de un proyecto.

1.6 Breve descripción de los otros capítulos de la memoria

- Gestión de proyectos
    - Metodología predictiva
        - RUP (Rational Unified Procces)
        - MSF (Microsoft Solution Framework)
        - Iconix
    - Metodología ágil
        - Scrum
        - Xtreme programming
        - Lean Software Development
        - Kanban
    - Estudio comparativo de las distintas metodologías
- Análisis comparativo de herramientas de gestión ágiles
    - Fiabilidad
    - Coste
    - Garantía
    - Opiniones
- Creación de sitio wiki para la gestión de un proyecto de desarrollo software:
    - WikiSpace
    - Simulación de la gestión de un sprint de un proyecto
- Conclusión final

2. Gestión de proyectos de desarrollo software

Según la guía **PMBOK®**[3], la gestión de proyectos es la aplicación de técnicas, habilidades, conocimientos y herramientas a las actividades de un proyecto para alcanzar sus requisitos. **Robert K. Wysocki** expone que, independientemente del tipo de proyecto a tratar, los cincos **procesos** que forman el ciclo de vida de la gestión de proyectos son (Wysocki, 2014):

Figura 4 Ciclo de vida de la gestión de proyectos

Y que los conocimientos que implica la gestión de proyectos se basan en diez áreas:

- Alcance.
- Tiempo.
- Integración.
- Coste.
- Calidad.
- Gestión de accionistas.
- Comunicaciones.
- Gestión del riesgo.
- Recursos humanos.
- Abastecimiento.

La correcta gestión del proyecto es de vital importancia para llegar a una consecución exitosa, sin embargo, llevarla a cabo no es una tarea fácil. Si se estudia el **Informe del Caos 2015** realizado por **Standish Group International** (firma independiente de investigación y asesoramiento TI), donde se analizaron 50.000 proyectos de TI en todo el mundo, se puede observar cómo un 19% de los proyectos fracasaron, o dicho de otra manera, solamente un 29% tuvieron éxito (The Standish Group, 2015).

Figura 5 Porcentaje de éxito según Informe del Caos 2015 (InfoQ, 2015)

[3] Project Management Body of Knowledge

El resultado es bastante esclarecedor, solamente algo más de la cuarta parte de los proyectos tuvieron un éxito rotundo. Este hecho indica que todavía hay trabajo por hacer para lograr mejores resultados en la gestión de proyectos de desarrollo de software. Si se revisan los datos de años atrás proporcionados por **Standish Group**[4], se puede observar como no ha habido una mejora significativa en los porcentajes (InfoQ, 2015):

	2011	2012	2013	2014	2015
Exitosos	29%	27%	31%	28%	29%
Discutidos	49%	56%	50%	55%	52%
Fracasos	22%	17%	19%	17%	19%

2.1. Influencia de las metodologías en la gestión de proyectos

Al principio de la década de los 80, surgió una nueva forma de gestionar proyectos, así que, se hizo necesaria una distinción entre la metodología que se llevaba aplicando hasta entonces y la nueva. Esta última se autodenominó ágil, por lo tanto, obligó a dar un nombre al modelo de gestión de proyectos que hasta entonces era único.

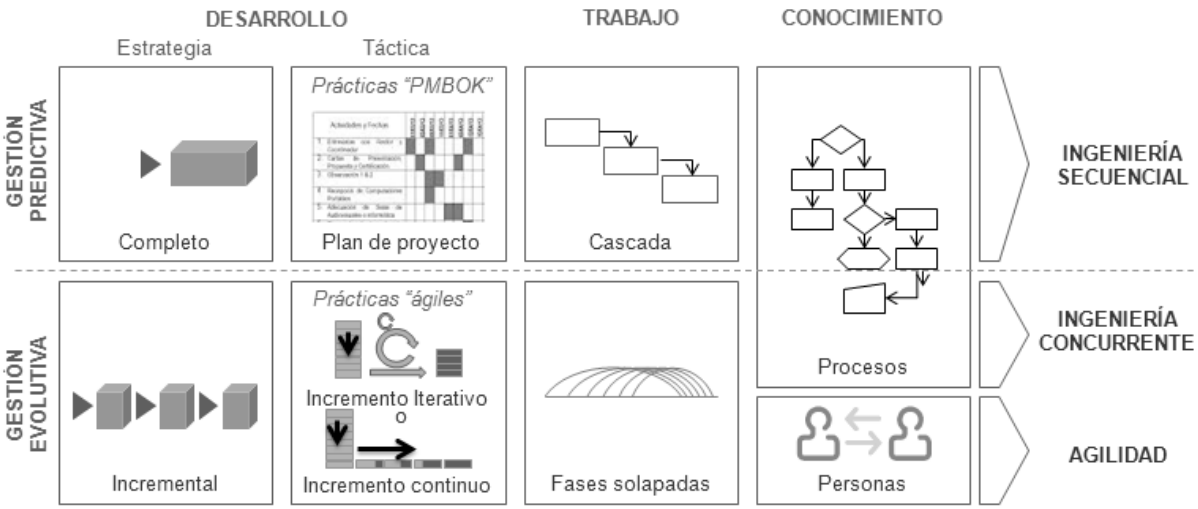


Figura 6 Prácticas y metodologías (Menzinsky, López, & Palacios, 2016)

Un dato llamativo del **Informe del Caos 2015** es que dependiendo de la metodología aplicada a la gestión y del tamaño del proyecto, los porcentajes de éxito varían significativamente (InfoQ, 2015):

[4] http://www.standishgroup.com

TAMAÑO	METODOLOGÍA	EXITOSO	DISCUTIDO	FRACASO
Grande	Ágil	18%	59%	23%
	Tradicional	3%	55%	42%
Mediano	Ágil	27%	62%	11%
	Tradicional	7%	68%	25%
Pequeño	Ágil	58%	38%	4%
	Tradicional	44%	45%	11%
TOTAL	Ágil	39%	52%	9%
	Tradicional	11%	60%	29%

Este hecho demuestra que la gestión de proyectos que emplean metodologías ágiles proporciona una mayor probabilidad de éxito, independientemente del tamaño del proyecto. De forma global, las metodologías ágiles consiguen más del triple del porcentaje de éxitos respecto a las metodologías tradicionales, usadas en la gestión de proyectos tradicional o predictiva.

2.2. Factores de éxito

Siguiendo con el **Informe del Caos 2015**, los principales factores de éxito en la gestión de proyectos son los siguientes:

- Apoyo a la dirección.
- Madurez emocional.
- Optimización.
- Personal calificado.
- Arquitecturas y marcos estándar.
- Procesos ágiles.
- Ejecución moderada y limitada.
- Experiencia en la gestión de proyectos.
- Objetivos claros del negocio.

La inmensa mayoría de estos factores están mejor cubiertos con las metodologías ágiles, sobre todo con **SCRUM**, que con las metodologías tradicionales o predictivas. Otro factor que puede influir en el éxito de la gestión es la correcta selección y uso de la herramienta software elegida para la administración del proyecto. Las principales ventajas que podemos obtener de usar una herramienta software de gestión de proyectos son:

- Recogida automática de información.
- Control total sobre el proyecto.
- Análisis de desviaciones.
- Mejora la comunicación interna.

Existen numerosas herramientas de gestión, pero el principal problema no es encontrarlas sino identificar y decidir cuál es la que mejor se adapta al proyecto. Dependiendo de la metodología seguida a la hora de proponer el ciclo de vida del proyecto, se puede clasificar la gestión como **tradicional**, también conocida como predictiva o pesada, o como **evolutiva**, conocida como adaptativa o ágil.

2.3. Gestión de proyectos predictiva

La gestión de proyectos predictiva es una disciplina formal que trata la planificación, ejecución, seguimiento y control de los aspectos de un proyecto a través de **procesos** sistemáticos y repetibles, para alcanzar los objetivos del mismo de forma segura y satisfaciendo las especificaciones definidas en la triple restricción: alcance, plazo y coste. El objetivo de la gestión de proyectos predictiva es resolver los planeamientos iniciales cumpliendo con el plazo y coste preestablecido (Menzinsky, López, & Palacios, 2016).

Este tipo de gestión usa metodologías y herramientas basadas en la definición de un alcance muy definido para obtener un producto concreto, luego se realiza la planificación del desarrollo en una serie de tareas, agrupadas en fases. Cada fase tiene asociado recursos y suele contener varios entregables muy claros. El conjunto de estas fases es lo que se conoce como "**ciclo de vida**".

Como se puede observar, este modelo está más orientado a la planificación. Los posibles contratiempos se pueden prever y arreglar con cierta normalidad a través del plan de desarrollo y el proyecto puede tomar forma de manera más rápida si los plazos y los costes están estimados de manera precisa. Este tipo de gestión tiende a valorar más a los procesos, en contraposición de la gestión ágil, que otorga una mayor importancia a las personas.

Dentro de la de gestión predictiva cabe destacar dos grandes pilares:

- **PMI** que otorga la certificación **PMP**[5] basada en la guía de gestión de proyectos **PMBOK®**.
- **PRINCE2**[6] certificado otorgado por la compañía británica **Axelos,** donde se estudia un modelo de referencia de gestión de proyectos de todo tipo.

En el escenario planteado, habitualmente el modelo de ciclo de vida de los proyectos de desarrollo de software se basan en un "desarrollo en cascada" (**Waterfall**), y se utiliza cuando el problema y su solución se conocen. Propone la creación de etapas ordenadas, de forma que el inicio de cada una debe esperar a la finalización de la etapa anterior.

[5] Project Management Profesional
[6] Projects in Controlled Enviroments

El desarrollo fluye secuencialmente desde el punto inicial hasta el punto final siguiendo un proceso descendiente, lo que significa, que no se puede volver a una etapa ya cerrada, de ahí que se le conozca como desarrollo en cascada.

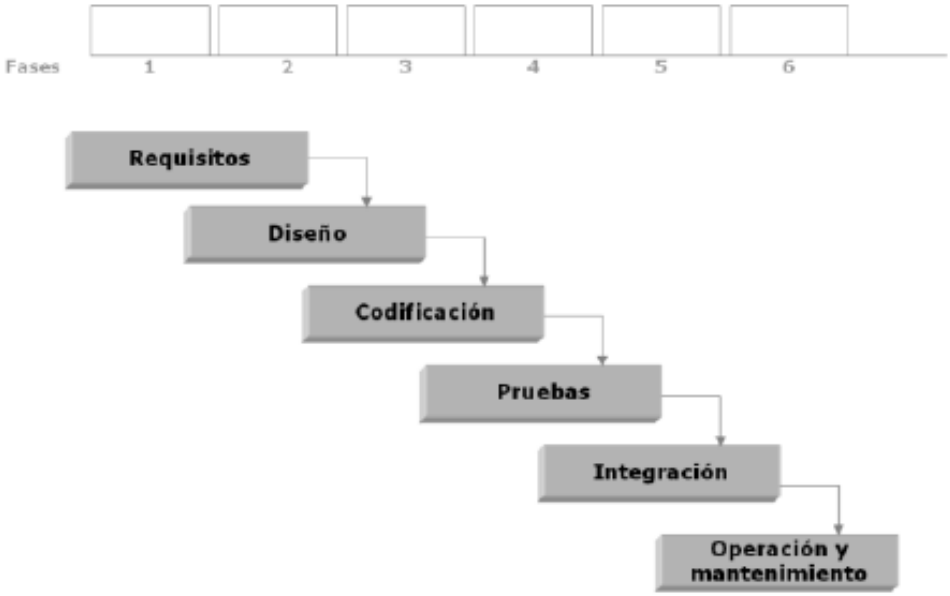


Figura 7 Desarrollo en cascada (Palacio & Ruata, Scrum Manager, 2009)

Cada fase la realiza un departamento, personas o equipos diferentes, profesionales especializados en los conocimientos necesarios. El método en cascada ofrece un mayor control de la acción, aunque no es impermeable al cambio, puesto que, como se ha dicho, dedica esfuerzos a prever los problemas. Existe un enfoque más flexible y realista en el desarrollo de software que permite realizar arreglos y cambios en etapas ya completadas e incluso solapar actividades de fases consecutivas para evitar la rigidez del flujo de trabajo (Wikiversidad Contributors, 2017). Como son:

- Desarrollo mediante prototipos.
- Desarrollo en espiral.
- Desarrollo iterativo e incremental.

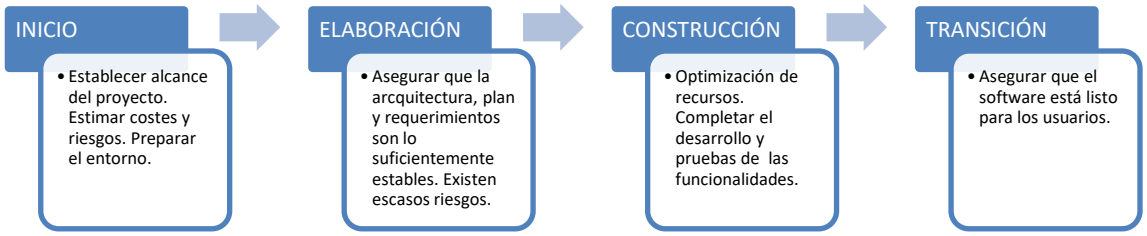
Metodologías de desarrollo tradicionales

Las metodologías tradicionales se consideran muy estrictas, por este motivo se le suelen denominar con el término de “pesadas”, ya que proponen la creación de una documentación exhaustiva, así como una planificación y seguimiento riguroso de las múltiples actividades a realizar durante el proyecto. Entre las más populares se encuentran: **RUP**[7] y **MSF**[8].

[7] Rational Unified Process
[8] Microsoft Solutions Framework

2.3.1 RUP

El proceso unificado de desarrollo es un tipo de metodología tradicional empleada en el desarrollo de software que no lleva a cabo un desarrollo estrictamente secuencial, sino que asume que dentro de cada fase se podrán realizar pequeñas iteraciones según los comentarios recibidos por parte de los usuarios. **RUP** se repite a lo largo de una serie de ciclos que constituyen la vida de un producto. Cada ciclo concluye con una generación del producto para los clientes. Cada ciclo consta de cuatro fases (International Technical Support Organization IBM, 2007):



Cada fase se subdivide a la vez en iteraciones cuya cantidad puede ser variable.

2.3.2 MSF

MSF es una guía de desarrollo de software flexible que puede adaptarse a cualquier proyecto de TI. Tal como expone la propia Microsoft en su web:

> "**Microsoft Solutions Framework (MSF)** es un enfoque personalizable para entregar con éxito soluciones tecnológicas de manera más rápida, con menos recursos humanos y menos riesgos, pero con resultados de más calidad. MSF ayuda a los equipos a enfrentarse directamente a las causas más habituales de fracaso de los proyectos tecnológicos y mejorar así las tasas de éxito, la calidad de las soluciones y el impacto comercial (Microsoft, 2017)."

La metodología **MSF** está basada en un conjunto de principios, modelos, disciplinas, conceptos, directrices y practicas aprobadas por Microsoft, que asegura resultados con menor riesgo y de mayor calidad, centrándose en el proceso y las personas.

Fases principales (Uniminuto, 2011):

Éste Framework está basado en los modelos espiral y cascada, lo cual indica que toma elementos de los métodos tradicionales que aún son referentes importantes para procesos de software. Es adaptable, flexible y escalable, e independiente de tecnologías, lo cual significa que no se cierra a un sólo modelo de programación sino más bien queda abierto según la naturaleza del proyecto (Uniminuto, 2011).

2.3.3 Iconix

Mención especial requiere esta metodología pesada-ligera de desarrollo de Software ya que se halla entre **RUP** y **XP**[9]. Unifica un conjunto de métodos de orientación a objetos con el objetivo de tener un control estricto sobre todo el ciclo de vida del producto a realizar (Palacio & Ruata, Scrum Manager, 2009).

2.4. Gestión de proyectos evolutiva

Se denomina gestión de proyectos evolutiva al conjunto de técnicas y prácticas empleadas para conducir la ejecución progresiva de un proyecto, de forma que genere un mínimo producto viable en el menor tiempo posible, y a partir de esa primera entrega, lo mantenga en mejora e incremento continuo.

Este tipo de gestión, está basado en la aplicación de metodologías ágiles. Según el **Scrum Manager Bok**:

> "La gestión de proyectos ágil no se formula sobre la necesidad de anticipación, sino sobre la de adaptación continua. (Menzinsky, López, & Palacios, 2016)"

Las ideas fundamentales de este estilo de gestión son según (Wysocki, 2014):

- Valorar a las personas y su interacción, por encima de los procesos
- Valorar el software que funciona, por encima de la documentación exhaustiva
- Valorar la colaboración con el cliente, por encima de la negociación contractual
- Valorar la respuesta al cambio, por encima del seguimiento de un plan

[9] eXtreme Programming

Manifiesto ágil

El manifestó ágil sigue estos doce principios:

1. “Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.
2. Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos Ágiles aprovechan el cambio para proporcionar ventaja competitiva al cliente.
3. Entregamos software funcional frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto posible.
4. Los responsables de negocio y los desarrolladores trabajamos juntos de forma cotidiana durante todo el proyecto.
5. 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.
6. El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.
7. El software funcionando es la medida principal de progreso.
8. Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida.
9. La atención continua a la excelencia técnica y al buen diseño mejora la Agilidad.
10. La simplicidad, o el arte de maximizar la cantidad de trabajo no realizado, es esencial.
11. Las mejores arquitecturas, requisitos y diseños emergen de equipos auto-organizados.
12. A intervalos regulares el equipo reflexiona sobre cómo ser más efectivo para a continuación ajustar y perfeccionar su comportamiento en consecuencia”. (Agile Manifesto Org, 2001).

Metodologías de desarrollo ágiles

El objetivo principal de esta metodología consiste en agilizar el desarrollo, enfocándose más en la interacción entre personas y en la entrega de un producto funcional en cada iteración que en la elaboración de documentación o el seguimiento estricto de protocolos (Wikiversidad Contributors, 2017).

Las metodologías ágiles más usadas son:

- Scrum
- Lean Software Development
- XP - eXtremme Programming

Cabe mencionar que estas metodologías junto con otras, como por ejemplo: Agile Database Techniques (ADT), Agile Modeling (AM), Adaptive Software Development (ASM), Agile Unified Process (AUP) o Crystal, se recogen en la organización **Agile Alliance**[10] (Agile Alliance, 2001)**.**

2.4.1 SCRUM

Scrum es una metodología ágil, basada en un modelo de desarrollo iterativo e incremental, estando orientado más hacia las personas que hacia los procesos. Cada iteración, **Sprints**, finaliza con la entrega de una parte operativa del producto, **incremento**, y este ciclo se va repitiendo hasta que el cliente da por completado el producto. Este incremento puede adoptar dos tácticas diferentes para mantener un avance continuo en el proyecto, incremento iterativo o incremento continuo.

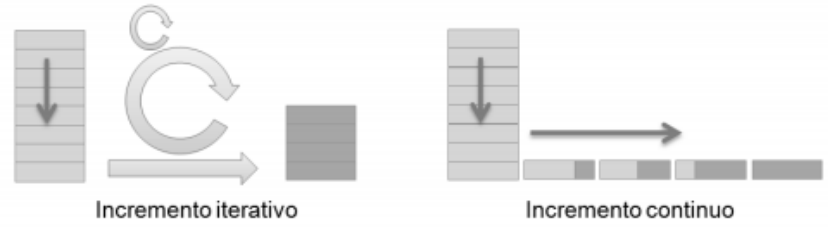


Figura 8 Incremento iterativo/continuo (Menzinsky, López, & Palacios, 2016)

Los **Sprints** son la base del desarrollo ágil, y **Scrum** gestiona su evolución en reuniones breves diarias donde todo el equipo revisa el trabajo realizado del día anterior y el previsto para el siguiente. Se trata de un método muy ligero que minimiza el conjunto de prácticas y artefactos, así como también las tareas y los roles (Wysocki, 2014).

Sus principios se basan en:

- "Mantener equipos de trabajo bien organizados en los que se maximice la comunicación.
- Utilizar un proceso flexible susceptible a cambios para asegurar una máxima calidad del producto.
- Dividir el trabajo en paquetes poco acoplados." (Wikiversidad Contributors, 2017)

[10] Organización global sin fines de lucro dedicada a promover los conceptos de desarrollo de software Agile como se describe en el Manifiesto Ágil.

Scrum es adecuado para aquellas empresas en las que el desarrollo de los productos se realiza en entornos caracterizados por tener: (Palacio & Ruata, Scrum Manager, 2009):

- Incertidumbre.
- Auto-organización.
- Control moderado.
- Transmisión del conocimiento.

Tal como se indica en (Menzinsky, López, & Palacios, 2016) y en (Pradel Miquel & Raya Martos, 2015), el marco técnico de scrum está formado por:

Roles

Scrum agrupa los roles de las personas en dos grandes bloques: personas comprometidas en el desarrollo, equipo, y el resto de personas involucradas en el proyecto, como los usurarios o **stakeholders**. Los roles que existen dentro del equipo son:

- **Product Owner:** Persona que toma las decisiones y que conoce el negocio del cliente y su visión (dueño del producto).
- **Scrum Master:** Encargado de comprobar que la metodología funciona.
- **Equipo Scrum:** pequeño grupo de desarrolladores que realizarán las tareas específicas de cada sprint.

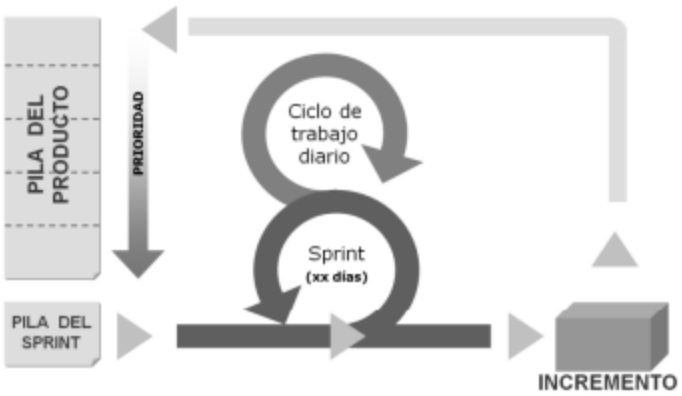


Figura 9 Ciclo iterativo scrum (Menzinsky, López, & Palacios, 2016)

Artefactos

- **Product Backlog.** Lista de requisitos de usuario (historias de usuario) que tiene asociada una estimación del valor y también su coste. El producto va creciendo durante su desarrollo.
- **Sprint Backlog.** Lista de los trabajos que debe realizar el equipo durante el sprint para generar el incremento previsto.
- **Release Burndown Chart.** Gráfico que muestra el progreso actual del equipo en función del número de historias de usuario que faltan por implementar.
- **Sprint Burndown**. Resultado de cada sprint.

Eventos

Los eventos se basan en dos ciclos de iteraciones: una iteración más larga y una iteración diaria.

- **Sprint Planning Meeting.** Reunión de trabajo que marca el inicio de cada sprint y en la que se deciden qué historias de usuario se implementarán. Por lo tanto, se crea el sprint backlog.
- **Daily Scrum.** Reunión diaria en la que todos los miembros del equipo responden a tres preguntas: qué hicieron ayer, qué piensan hacer hoy y qué impedimentos se han encontrado y que les han impedido avanzar.
- **Sprint Review Meeting.** Al finalizar un sprint se revisa el trabajo realizado y se enseña, a quien esté interesado, el resultado implementado (la demo).
- **Sprint Retrospective.** Sirve para reflexionar sobre lo que haya pasado durante el sprint y para identificar oportunidades de mejora en el proceso de desarrollo.

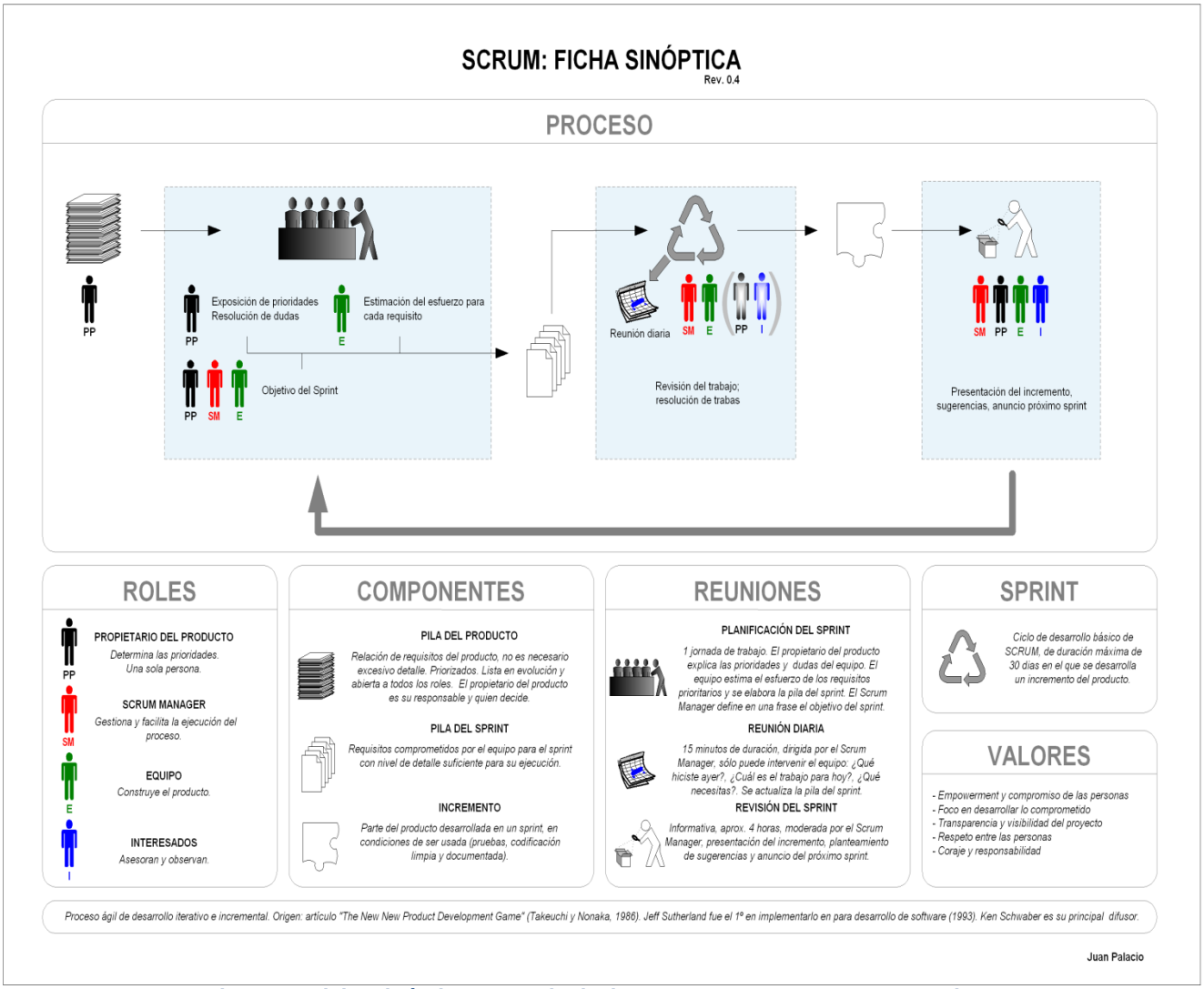


Figura 10 Ficha sinóptica scrum (Palacio & Ruata, Scrum Manager, 2009)

Workflow de Scrum

Como se puede estudiar en **PMBOK**, todo proyecto debe iniciarse con una reunión de ***Kick off***, o reunión de arranque, entre todos los interesados en el proyecto. En ella se reunirán los responsables de la empresa ejecutora del proyecto y el cliente, para hablar de todo lo que tenga que ver con el proyecto. Esta reunión establece su arranque formal.

Por parte del cliente, el **Product Owner** junto con los usuarios del producto, definen los **requisitos**[11] de todo el proyecto. Según la granularidad de los requisitos, se pueden dividir en:

- **Epics**, están presente durante prácticamente todo el proyecto.
- **Features**: historia de usuario que está presente en más de un sprint.
- **Stories**: Son las historias de usuario propiamente dicha, que abarcan un sprint. Se suelen agrupar en ***Themes***

Se comenzará por las épicas y se irá desgranando en requisitos más concretos hasta llegar a las **Stories**. Cada requisito funcional[12] se conoce como **historia de usuario** y tiene la siguiente estructura:

"Como **<rol>** necesito **<requisito>** para **<propósito>**"

El conjunto de todas las historias de usuario se conoce como **Product Backlog**. A continuación, se estiman[13] todas estas historias, por ejemplo con la técnica de **Planning Poker**, y se agrupan para ir desarrollándolas en cada sprint, y se propone el plan de entrega, **Release Plan**. El proyecto se divide en **Sprints**, de duración inferior a 30 días.

El **Scrum Master** será el encargado de asegurarse que se cumplan todas las pautas de la metodología y que los componentes del **equipo Scrum** puedan desarrollar su trabajo. Se realiza una reunión de planificación, **Planning Meeting**, el primer día de cada Sprint y se selecciona el conjunto de historias de usuario que entran en ese sprint, a esas historias se les conoce como **Sprint Backlog**.

También se realizará una reunión diaria no superior a 15 minutos, que se denomina **Scrum Meeting** o **Daily Scrum**, para sincronizar las tareas de todos los miembros e identificar los problemas que surjan. Cada componente debe indicar, qué hizo el día anterior, en qué va a trabajar ese día y qué inconvenientes tiene. Entre todos se intentará resolver esos problemas.

Al terminar cada Sprint se realizará una demostración del estado actual del producto, **Sprint Review Meeting**, y una reunión de retrospectiva, **Sprint retrospective,** con todos los miembros involucrados, cuyo fin es buscar la mejora continua del equipo. Este ciclo se repite hasta terminar el proyecto.

[11] Un requisito es un factor de calidad que un sistema software necesita para tener valor y utilidad para un usuario.
[12] Los requisitos pueden ser funcionales o no funcionales.
[13] Existen distintas técnicas de estimación (Anexo 1).

2.4.2 Metodología Lean

Lean Software Development (LSD) toma los principios del *Lean manufacturing* y del *Lean IT* y los aplica al desarrollo de software. El término **LSD** tiene origen en un libro del mismo nombre, escrito por Mary Poppendieck y Tom Poppendieck. Esta metodología muestra una serie de claves, siete principios para ser exactos, que su aplicación es condición necesaria para la existencia del ajuste. Como se nos presenta en el libro (Poppendieck & Poppendieck, 2003) estos principios son los siguientes:

- Eliminación de desperdicios:
  - Desarrollos parciales correctos pero erróneos al unificar el desarrollo.
  - Tareas Extras. Generación de documentación.
  - Añadir características extras. Pueden no ser nunca necesarias.
  - Requisitos poco claros o sujetos a constantes cambios.
  - Tiempos de espera.
  - La aparición tardía de defectos.
  - Cambios de tarea. Cambiar los recursos de tareas conlleva un tiempo de adaptación.

- Ampliación del conocimiento:
  - Las revisiones y notas sobre los códigos.
  - La creación de documentación.
  - La elaboración de una base de datos informativa sujeta a actualizaciones.
  - Las puestas en común de conocimiento.
  - Las iniciativas formativas.

- Decidir lo más tarde posible:
  - Frente a la incertidumbre, lo más sensato es retrasar la toma de decisiones lo máximo posible. Así se mantiene todas las opciones disponibles durante más tiempo y no se compromete con algo que no se sabe con certeza si se va a poder cumplir.

- Entregar lo más pronto posible:
  - En desarrollo, el ciclo de descubrimiento es fundamental para el aprendizaje: diseñar, implementar, retroalimentar, mejorar. Cuantos más cortos son estos ciclos, más se puede aprender.

- Mejorar el equipo:
  - Involucrar a los desarrolladores en los detalles de las decisiones técnicas es fundamental para lograr la excelencia.

- Aseguramiento de la integridad:
  - El software con integridad tiene una arquitectura coherente, puntajes altos en usabilidad y idoneidad para el propósito, y es mantenible, adaptable y extensible.

- Integrar el todo:
    - Cuando los individuos u organizaciones se miden en su contribución especializada en lugar del rendimiento general, es probable que se produzca una suboptimización

2.4.3 eXtreme Programming

El **eXtreme Programming**, es una metodología ágil propuesta por Kent Beck, autor del primer libro sobre la materia, *Extreme Programming Explained: Embrace Change*.

> "**XP** es un estilo de desarrollo de software que se enfoca en una excelente aplicación de técnicas de programación, comunicación clara y trabajo en equipo lo que nos permite lograr cosas que antes ni siquiera podíamos imaginar" (Beck, 2004).

Tal y como describe Kent Beck, (Beck, 2004), esta metodología define cuatro variables independientes ante cualquier proyecto software, que son: costo, tiempo, calidad y alcance. Además, propone que sólo tres podrán ser fijadas por actores externos al grupo, por lo tanto, el valor de la variable que falta deberá ser establecido por el equipo, en función de los valores de las otras tres.

Como se recoge en extremeprogramming.org, sitio dedicado a la divulgación de la metodología **XP**, ésta plantea un conjunto de roles, reglas, valores y prácticas bien definidas (Wells, 2009):

Roles

- Programador
- Cliente
- Tester
- Entrenador
- Consultor
- Gestor (Big boss)

Valores

- Simplicidad
- Comunicación
- Respeto
- Coraje
- Retroalimentación

Fases

El ciclo de vida de un proyecto **XP** se puede separar en estas fases:

Figura 11 Fases XP

Reglas

- Planificación:
  - Se recogen las historias de usuario.
  - Se especifica qué historias de usuarios se implementarán para cada versión del sistema y las fechas de esas versiones.
  - Se divide el proyecto en iteraciones.
  - Se realizan frecuentes entregas.

- Diseño:
  - Simplicidad.
  - Estructura que ayude a las personas nuevas a comenzar a contribuir rápidamente
  - Refactorización de código.
  - Se obvian posibles requisitos futuros.
  - Intentar reducir el riesgo.
  - Emplear sistema *CRC card*.

- Desarrollo
  - Se integra al cliente en el equipo.
  - Se plantean al principio las pruebas unitarias.
  - Se emplea la programación por pares.
  - Se realiza una integración continúa.
  - Se aplican reglas de codificación estándar.

- Pruebas
  - Todo el código debe pasar pruebas unitarias.
  - Si se encuentra un *bug* (error) se crea un test para comprobar que no vuelve a aparecer.

Pair Programming

Otra característica importante es que todo el código desarrollado es creado por dos personas, técnica conocida como ***Pair Programming***[14]. Se sentarán dos programadores juntos y mientras uno programa, su compañero irá revisando el código y aportando ideas. Cada cierto tiempo cambiarán sus roles. Aunque parezca que esta técnica sea una pérdida de tiempo, no es así, porque a la vez que se programa se va depurando el código, dando un resultado mucho más óptimo.

[14] Programación por parejas

2.4.4 Otros métodos y técnicas

Otros métodos que requieren atención serían:

a) Dynamic Systems Development Method (DSDM)

Mención especial requiere el framework de desarrollo ágil **DSDM.** Como comenta (Palacio & Ruata, Scrum Manager, 2009) se fundó en Inglaterra en 1994 por un consorcio de compañías británicas conocido como **Consorcio DSDM**, y ya sentaba las bases de las futuras metodologías ágiles. Este método de desarrollo ágil se apoya en la continua interacción con el usuario convirtiéndolo en un sistema adaptable a las necesidades de la empresa.

Fases

1. Pre-proyecto
2. Estudio de viabilidad
3. Estudio de negocio
4. Iteración de modelado funcional
5. Iteración de diseño y desarrollo
6. Implementación
7. Post-desarrollo

Roles

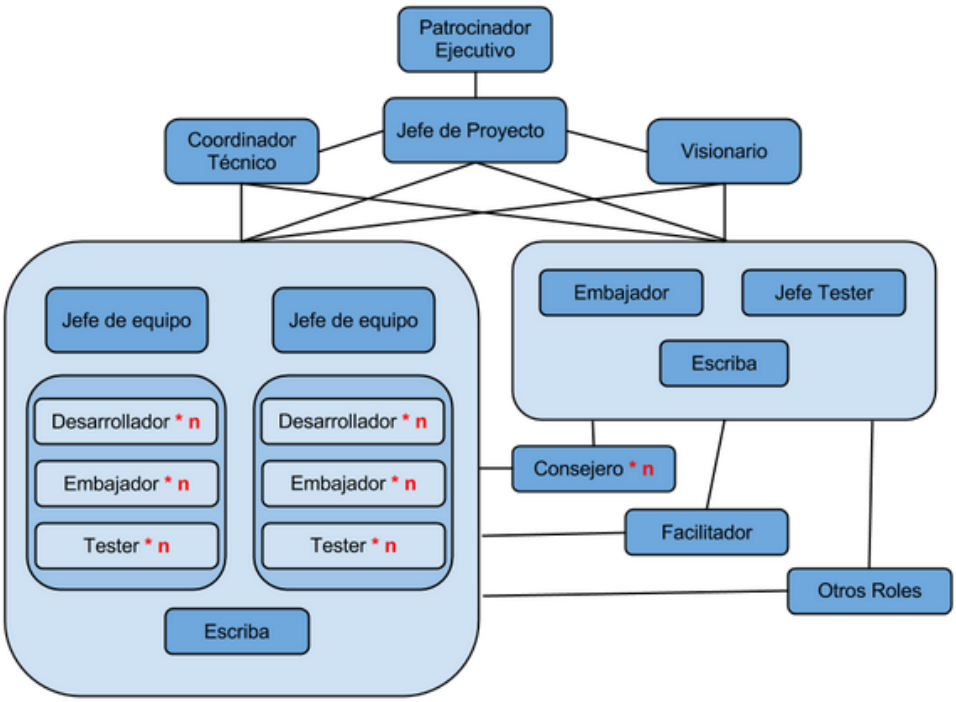


Figura 12 Roles en DSDM (Wikiversidad Contributors, 2015)

Principios

El método **DSDM** utiliza 9 principios (Wikiversidad Contributors, 2015):

1. La obligación del usuario de implicarse.
2. Los equipos deben de tener el poder de tomar decisiones.
3. Realización de entregas tempranas.
4. Las entregas que se acepten han de ser adaptadas a la empresa.
5. El desarrollo es Iterativo e Incremental.
6. Todos los cambios que se hagan durante el desarrollo han de ser reversibles.
7. Los requisitos se describen en alto nivel y tendrán una línea base.
8. Las pruebas se integran a lo largo del ciclo de vida.
9. Enfoque cooperativo y colaborativo.

b) Técnica visual: Kanban

Según se describe en **Scrum Manager Bok** (Palacio & Ruata, Scrum Manager, 2012), **Kanban** es un sistema de señalización para comunicar información relativa y necesaria en la ejecución o monitorización de un trabajo, muy útil para la gestión ágil. El desarrollo ágil de software emplea prácticas de gestión visual, por ser las que mejor sirven a los principios de comunicación directa y simplicidad en la documentación y gestión.

Kanban no es un modelo o marco de gestión, sino una herramienta de señalización. No hay por tanto un formato cerrado de tarjetas o tableros **Kanban**. La gestión de un proyecto puede utilizar señalización **Kanban** para:

- Ofrecer información de seguimiento del proyecto.
- Tener una herramienta de gestión y control del ritmo del proyecto.

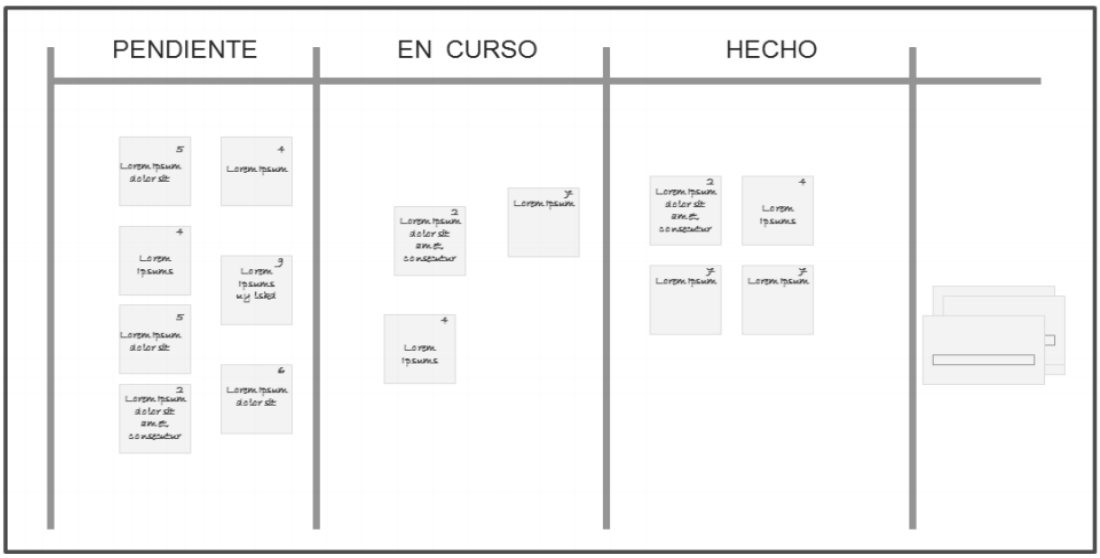


Figura 13 Tablero Kanban (Menzinsky, López, & Palacios, 2016)

2.5. Mapa conceptual de la gestión de proyectos de desarrollo software

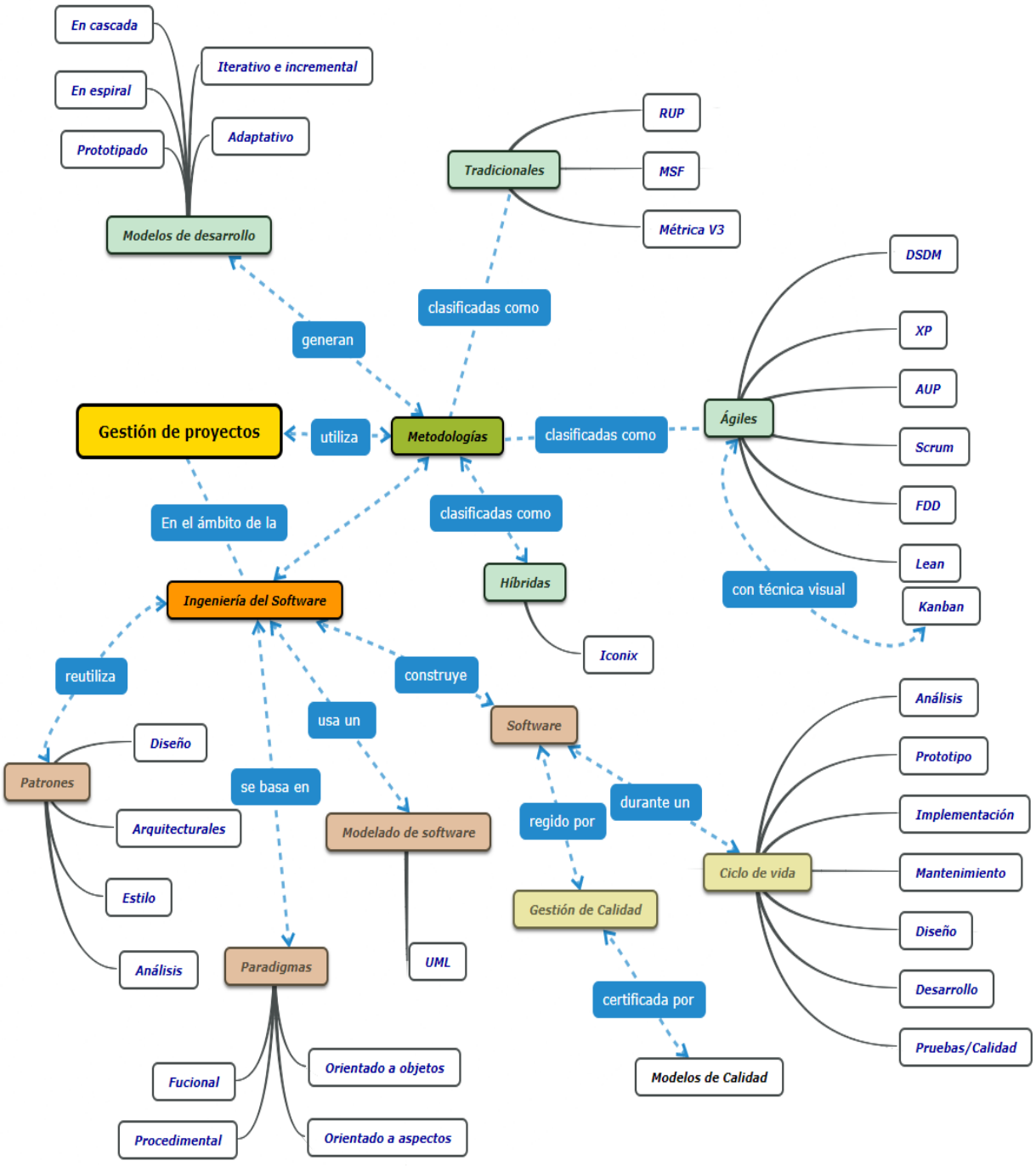


Figura 14 Mapa conceptual de la gestión de proyectos – Elaboración propia.

2.6. Estudio comparativo de las distintas metodologías

A continuación, según todas las fuentes consultadas y estudiadas en los apartados anteriores, se exponen distintos cuadros de estudio comparativo de las dos principales metodologías ya explicadas, las tradicionales y las ágiles.

Características principales de cada metodología.

TRADICIONAL	ÁGIL
• Orientada a proyectos de cualquier tamaño	• Orientada a proyectos pequeños
• Equipos grandes y dispersos	• Equipos pequeños, sobre 10 personas
• Proyectos de media / larga duración	• Proyectos de corta duración
• Proyecto cerrado	• Proyecto abierto a cambios
• El cliente mantiene reuniones con la dirección	• El cliente está integrado en el equipo
• Arquitectura prefijada	• Arquitectura se va mejorando
• Documentación rigurosa	• Poca documentación
• Roles específicos	• Roles genéricos
• Roles no intercambiables	• Roles flexibles
• Centrada en los procesos	• Centrada en las personas
• Gestión dirigida	• Gestión colaborativa
• Alto coste de prototipado	• Bajo coste de prototipado
• Planificación inicial alta	• Planificación inicial baja
• Basada en estándares de desarrollo	• Basadas en heurísticas
• Poco *feedback*	• Continuo *feedback*
• Proceso lineal	• Proceso iterativo
• El coste se acerca a lo estimado	• El coste puede dispararse

Ventajas y Desventajas

TRADICIONAL		ÁGIL	
VENTAJAS	**DESVENTAJAS**	**VENTAJAS**	**DESVENTAJAS**
• Registra toda la información • El cliente sabe exactamente cómo será su producto final • Se conoce el tamaño, costo y fechas del proyecto • La rotación de empleados tiene un impacto mínimo en el proyecto, al estar todo rigurosamente documentado	• Tras cada fase, no se puede volver hacia atrás • Dependencia de los requisitos iniciales. Si están mal planteados, fracasará el proyecto • Si aparece un error de requisito o si se necesita hacer un cambio, el proyecto debe comenzar de nuevo • Las pruebas se retrasan hasta el final • La planificación no tiene en cuenta las necesidades cambiantes del cliente	• Cambios sobre la marcha • Añadir nuevas funcionalidades • Se realizan pruebas tras cada sprint • Producto estable tras cada entregable • Cliente integrado en el equipo	• El producto final puede ser muy diferente de lo que inicialmente se pretendía • Existe el riesgo de entrar en un ciclo de entrega de prototipos y nunca cerrar el proyecto • El desarrollo depende del equipo, un cambio de empleado supone necesitar tiempo para su adaptación • Debido a las modificaciones el coste del proyecto puede dispararse.

¿Qué tipo de metodología usar: tradicional o ágil?

Según todo lo expuesto hasta el momento en el presente **TFG**, no se puede afirmar categóricamente que una metodología sea mejor que otra, ya que los resultados del éxito o fracaso del proyecto dependerán de si se ha elegido una metodología adecuada para su desarrollo, si se han empleado correctamente las herramientas de gestión, si se ha realizado una buena gestión por parte del jefe de proyectos o si el equipo ha estado motivado y concentrado en el trabajo.

Lo que sí se puede hacer es proponer una guía, la cual recoja la mayoría de características asociadas al desarrollo del proyecto, que sirva de referencia a la hora de decidir que metodología emplear y así aumentar la probabilidad de acierto.

Se identifican los factores más importantes que intervienen en el desarrollo del proyecto, que según lo estudiado serían:

FACTOR	TRADICIONAL	ÁGIL
Prioridad de negocio	Cumplimiento	Valor
Tiempo del proyecto	Largo	Corto
Tamaño del equipo	Grande	Pequeño
Interacción con el cliente	Mínima	Máxima
Ambiente del desarrollo	Controlado	Incertidumbre
Dirección	Organizativa	Colaborativa
Rigidez del producto	Cerrado	Ampliable
Requisitos	Claros	Ambiguos
Riesgo de fallos	Bajo	Alto
Enfoque de desarrollo	Procesos	Personas
Necesidad de documentación	Alta	Baja
Probabilidad de cambios en el equipo	Alta	Baja
Roles intercambiables	No flexible	Flexible
VALORES	1 2	3 4

Ahora se plante al siguiente formulario y se rellena según las necesidades del proyecto, (solo se puede marcar una opción por factor):

FACTOR	VALOR
Prioridad de negocio	Cumplimiento - Valor ○ ○ ○ ○
Tiempo del proyecto	Largo - Corto ○ ○ ○ ○
Tamaño del equipo	Grande - Pequeño ○ ○ ○ ○
Interacción con el cliente	Mínima - Máxima ○ ○ ○ ○
Ambiente del desarrollo	Controlado - Incertidumbre ○ ○ ○ ○
Dirección	Organizativa - Colaborativa ○ ○ ○ ○
Rigidez del producto	Cerrado - Ampliable ○ ○ ○ ○
Requisitos	Claros - Ambiguos ○ ○ ○ ○

Riesgo de fallos	Bajos - Altos ○	○	○	○
Enfoque de desarrollo	Procesos - Personas ○	○	○	○
Necesidad de documentación	Alta - Baja ○	○	○	○
Probabilidad de cambios en el equipo	Alta - Baja ○	○	○	○
Roles intercambiables	No flexible - Flexible ○	○	○	○
Frecuencia de entregables	Baja - Alta ○	○	○	○
VALORES	**1**	**2**	**3**	**4**
TOTAL POR COLUMNAS	**A**	**B**	**C**	**D**
VALORACIÓN FINAL	**(A+B+C+D) / 14 = VALORACIÓN**			

Los valores corresponden a:

1. Acorde total con enfoque tradicional
2. Acorde parcial con enfoque tradicional
3. Acorde parcial con enfoque ágil
4. Acorde total con enfoque ágil

Se rellena, se suman los valores y se divide entre el número de factores. Si el resultado está más cerca del 1, sería más óptima una metodología tradicional, y si es más cercano a 4, sería más recomendable una metodología ágil. De una forma visual se colocaría el resultado en la siguiente recta:

¿Qué metodología ágil emplear?

Se ha estudiado varios *frameworks* para la evaluación de las distintas metodologías ágiles y de este modo elegir la más adecuadas (Mendes Calo, Estevez, & Fillottrani) y (Iacovelli & Souveyet, 2008) y se ha creado la siguiente tabla para realizar la evaluación:

FACTOR	DSDM	XP	LEAN	KANBAN	SCRUM
Actividades de puesta en marcha	1	1	0	0	1
Adaptación a Incertidumbre	1	1	0	1	1
Cambiar plan de trabajo	0	1	1	1	0
Descripción de procesos	1	1	1	1	1
Documentación de usuario	1	0	1	0	1
Gran impacto al cambiar el equipo	1	0	1	0	1
Informe de calidad	0	0	1	0	0
Integración de cambios	1	1	1	1	1
Interacción con cliente	1	1	1	1	1
Iteraciones cortas	1	1	1	0	1
Modelado	1	1	0	0	1
Necesidad de gestionar el proyecto	1	0	0	0	1
Necesidad de interacción con usuarios finales	1	0	0	0	0
Offshoring	1	0	1	1	0
Peso ligero	0	1	1	1	1
Política de refactoring	1	1	1	0	0
Pruebas	1	1	1	1	1
Requisito funcional pueden cambiar	1	1	1	1	1
Respetar alto nivel de calidad	0	0	1	0	0
Rigurosidad en fechas de entrega	1	0	0	0	1
Satisfacción del usuario	1	1	1	0	1
Se pueden cambiar indicadores	0	1	1	0	0
Uso del sistema	1	0	0	0	0

Se plantea el siguiente formulario a rellenar donde se le dará una puntuación según la necesidad del factor o funcionalidad del 0 al 5, siendo 0 que no se necesita y 5 que tiene la importancia más alta:

FACTOR	VALOR (0 al 5)
Actividades de puesta en marcha	
Adaptación a Incertidumbre	
Cambiar plan de trabajo	
Descripción de procesos	
Documentación de usuario	
Gran impacto al cambiar a miembros del equipo	
Informe de calidad	
Integración de cambios	
Interacción con cliente	
Iteraciones cortas	
Modelado	
Necesidad de gestionar el proyecto	
Necesidad de interacción con usuarios finales	
Offshoring	
Peso ligero	
Política de refactoring	
Pruebas	
Requisito funcional pueden cambiar	
Respetar alto nivel de calidad	
Rigurosidad en fechas de entrega	
Satisfacción del usuario	
Se pueden cambiar indicadores	
Uso del sistema	

Una vez rellenado el formulario, las cantidades se multiplican por el valor correspondiente a su misma posición en la tabla base, haciéndolo por cada columna, es decir una vez por cada metodología. Luego se sumarán las columnas y aquella opción que tenga una puntuación más alta será la metodología más apropiada para usar en el proyecto.

3. Herramientas de gestión de proyectos ágiles

Una vez que ya se ha decidido la metodología a usar en la gestión del proyecto, ahora queda otra tarea no menos importante, la elección de la herramienta software de gestión ágil (**APMS**[15]) que se va a utilizar durante todo el desarrollo del proyecto (si es que se elige alguna). En el mercado existen numerosas soluciones software, tanto privativas como libres, de pago o gratuitas, web o locales, que ofrecen numerosas funcionalidades para la administración del proyecto.

El objetivo del presente capítulo es realizar una comparación de las herramientas más valoradas por el cuadrante mágico de **Gartner**, entre otros, y otras más habituales del mercado y así obtener la máxima información para poder elegir la que mejor se adapte tanto a la tipología de la empresa como a los recursos que participan en el proyecto. No hay que dejar de lado, que aparte de la elección más óptima, también tiene un gran peso el correcto uso que se haga de ella.

3.1. Introducción

Según la famosa web *FinancesOnline*, portal independiente de análisis para plataformas **B2B**[16], **SaaS** y soluciones financieras (FinancesOnline, 2017), una herramienta software de gestión de proyectos ágiles ayuda a los usuarios a ejecutar proyectos complejos, a identificar problemas, a reestructurar procedimientos y recursos de acuerdo con las situaciones cambiantes. En definitiva, este software, ayuda en gran medida al equipo a lograr los objetivos finales del proyecto.

Este tipo de herramientas se vende mediante dos modelos de distribución, **SaaS** y/o **on-premise**, siendo sus respectivas características:

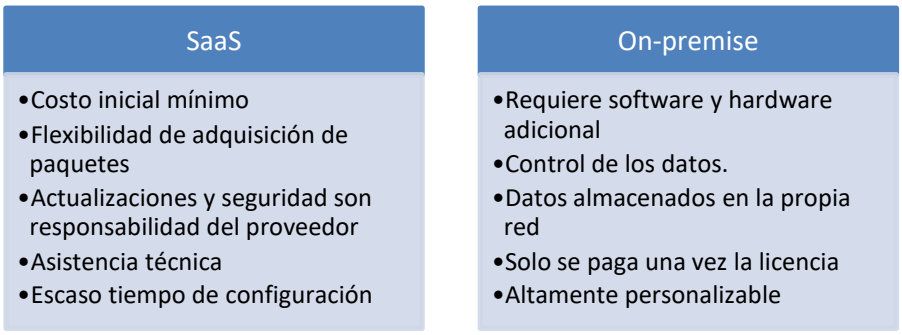


Figura 15 Características SaaS y On-premise

También, *FinancesOnline*, explica las características y beneficios de usar un **APMS**.

[15] Agile Project Management Software
[16] Business to Business

Características mínimas necesarias de un APMS

- Visualización del progreso del proyecto.
- Tablero kanban.
- Seguimiento de tareas.
- Seguimientos de fallos y problemas.
- Wikis.
- Estimación.
- Gestión de requisitos.
- Portal de clientes.
- Cartera de proyectos.

Según el informe (VersionOne, 2017) a estas características, es aconsejable sumarle otras herramientas como:

- Herramienta de prueba de unidad.
- Herramienta de construcción automatizada
- Herramienta de integración continua
- Herramienta de automatización de lanzamiento / implementación
- Herramienta de aceptación automatizada
- Herramienta de control de versiones.

Beneficios de un AMPS

- Mejor calidad del producto.
- Mayor satisfacción del cliente.
- Mayor propiedad y colaboración.
- Estructuras de equipo personalizadas.
- Métricas relevantes.
- Mejor visibilidad del rendimiento.
- Control mejorado del proyecto.
- Mejora la previsibilidad.
- Riesgo reducido.

3.2. Herramientas mejor valoradas

Tal como explica **Gartner, Inc.** en su web:

> "El mercado de herramientas de administración de ciclo de vida (**ADLM**) de desarrollo de aplicaciones se centra en las actividades de planificación y gobierno del ciclo de vida de desarrollo de software (**SDLC**).
>
> Los productos **ADLM** se enfocan en la parte de "desarrollo" de la vida de una aplicación. Los elementos clave de una solución **ADLM** incluyen: definición y gestión de requisitos de software, cambio de software y gestión de configuración, planificación de proyectos de software, con un enfoque actual en la planificación ágil, gestión de elementos de trabajo, gestión de calidad, incluida la gestión de defectos.

Otras capacidades clave incluyen: informes, flujo de trabajo, integración con la gestión de versiones, soporte para wikis y colaboración, instalaciones sólidas para la integración con otras herramientas **ADLM**" (Gartner Inc., 2017)

El siguiente cuadrante mágico de **Gartner** muestra las herramientas líderes del sector:

Figura 16 Cuadrante mágico para ADLM (Gartner Inc., 2015).

Se puede observar como las herramientas que destacan sobre las otras son *Atlassian, Microsoft, IBM, VersionOne* y *Rally*. En la misma web **Gartner Peer Insights,** se puede comprobar cómo según una ponderación entre la puntuación otorgada a las distintas herramientas por usuarios refutados y el número de *reviews*[17], las tres primeras que encabezan la lista son: *Atlassian, Microsoft, VersionOne y Rally*[18]*.*

[17] Análisis
[18] Rally fue adquirido por CA Technologies en 2015.

Proveedores y productos	+ Mostrar productos	Comentarios	Calificación general
			1 2 3 4 5
Atlassian		434	4.2
Microsoft		95	4.2
Rally (ahora CA Technologies)		46	3.9
Hewlett Packard Enterprise (HPE)		37	3.9
VersionOne (CollabNet)		35	4.3
IBM		25	3.8

Figura 17 Puntuación de herramientas por usuarios (Gartner Inc., 2017)

Consultando también el informe anual proporcionado por (VersionOne, 2017) sobre qué herramientas ágiles son más usadas y cuáles son las más recomendadas, se puede observar los siguientes resultados:

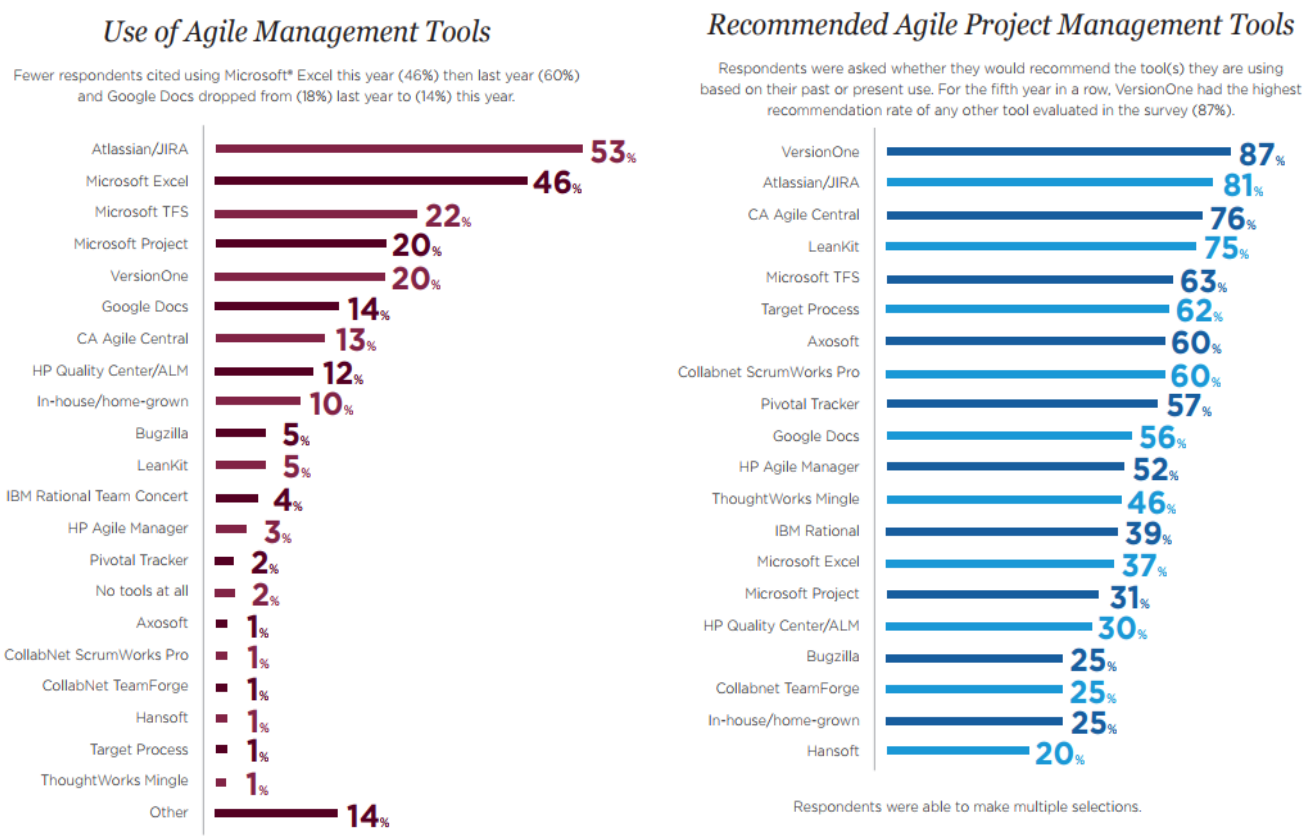


Figura 18 Uso y recomendación de herramientas ágiles (VersionOne, 2017)

Las cinco herramientas más usadas serían: ***Atlassian/Jira, Microsoft Excel, TFS, MS Project*** y ***VersionOne***, por el contrario, las cinco herramientas más recomendadas serían: ***VersionOne, Atlassian/Jira, CA, LearnKit*** y ***TFS***.

Observando las herramientas que se repiten en ambos informes e incluso en las *reviews* (análisis) de usuarios proporcionadas por **Gartner**, se saca en claro que las herramientas más usadas son:

- Atlassian/Jira
- Microsoft TFS
- VersionOne
- Rally (Fue adquirido y mejorado en 2015 por CA Technologies).

3.2.1. Atlassian / Jira

> "**Jira Software** es una herramienta ágil de gestión de proyectos compatible con cualquier metodología ágil, ya sea scrum, **Kanban** o la tuya propia. Desde tableros hasta informes ágiles, puedes planificar, supervisar y gestionar todos los proyectos de desarrollo de software ágil con una sola herramienta. Elige una metodología para ver cómo **Jira Software** puede hacer que tu equipo publique software de calidad con mayor rapidez" (Atlassian, 2017).

Características principales

Tal y como aparece en la web de atlasian (Atlassian, 2017), sus principales cualidades son:

- Herramientas ágiles para scrum: contiene las herramientas necesarias para la planificación de *sprints*, los **scrums** diarios, el seguimiento de los *sprints* y las retrospectivas.
- Alta configuración y personalización de: flujos de trabajo, cuadros de mando, tipos de incidencias, gráficos, diagramas e informes.
- Herramientas ágiles para **Kanban**: tablero que muestra el estado del trabajo en columnas y carriles.
- Metodologías mixtas: Algunos equipos ágiles han adoptado metodologías mixtas para respaldar el funcionamiento del equipo. Por ejemplo, *scrumban*[19] o *kanplan*[20].
- Metodología ágil a escala: preparado para ampliar la metodología ágil. Extensible gracias a una API en Java bien documentada.
- Modelo *on-premises*[21] y *SaaS*[22].

Según (Gartner Inc., 2015), las fortalezas de **Jira** son:

- Adquisición flexible de funcionalidades.
- Gran personalización de los flujos de trabajo.
- Gestor de documentación con *Confluence*.
- Soporte sólido para control de versiones e integración continua.

[19] Scrum + kanban
[20] Kanban + backlog
[21] Instalación local
[22] Software as a Service. Computación en la nube.

Y sus debilidades son:

- Contrato con el cliente poco flexible y mal definidos.
- Sincronización pobre con otras herramientas.
- Interfaz no muy amigable. Curva alta de aprendizaje.

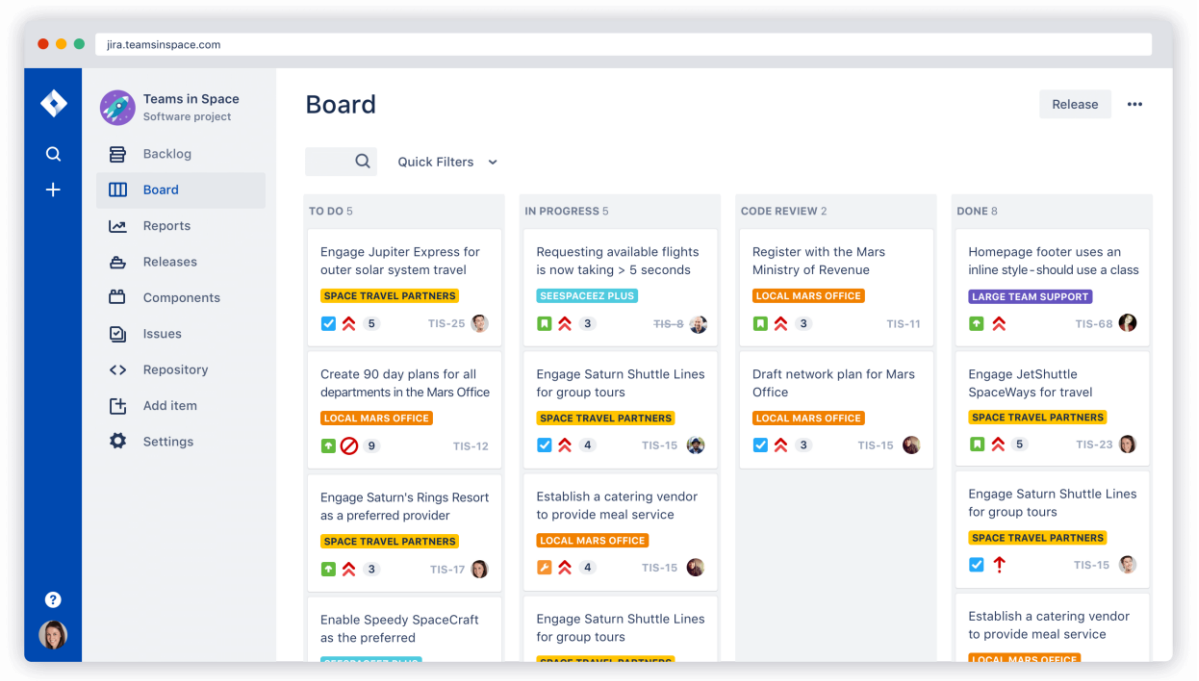


Figura 19 Tablero scrum en Jira (Atlassian, 2017)

3.2.2. Microsoft: Team Foundation Server (TFS)

> "**Team Foundation Service** es un nuevo servicio de Microsoft en el que se nos proporciona la herramienta de gestión de ciclo de vida de aplicaciones **Team Foundation Server (TFS)** bajo la modalidad de *SaaS*.
>
> Esto quiere decir que, para poder beneficiarnos de las ventajas de usar esta herramienta, no tendremos que pagar por diferentes licencias, ni montar nuestra propia pequeña granja de servidores o pasar por una serie de tediosos procesos de instalación (Microsoft, Escolar, 2017)."

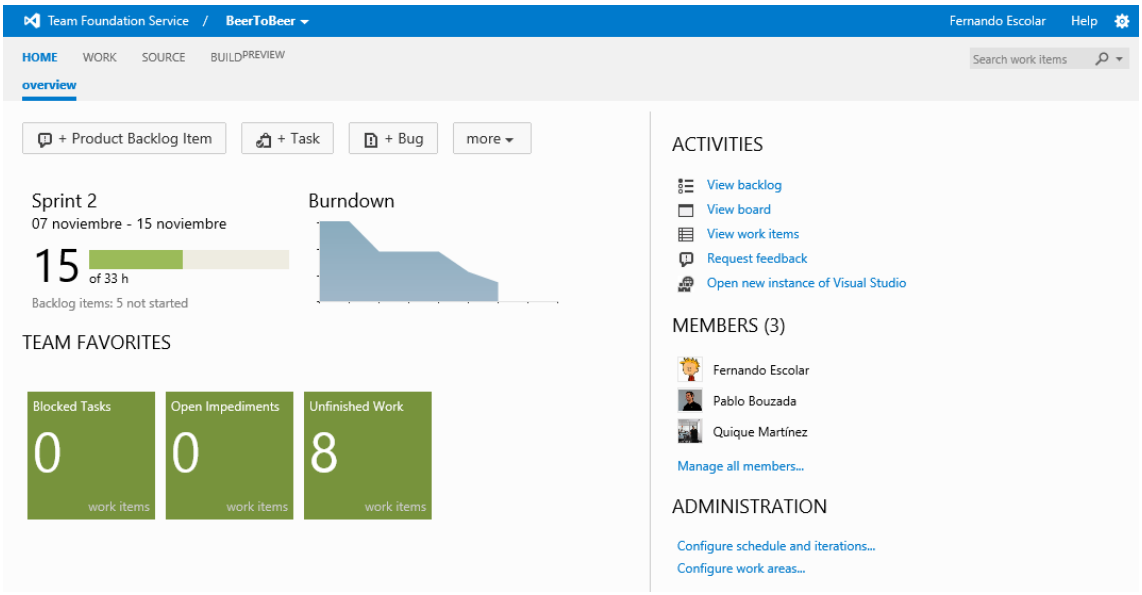


Figura 20 Interfaz TSF (Microsoft, Escolar, 2017)

Características principales

- Gestión del equipo y su trabajo.
  - Gestión del *Backlog*.
  - *Sprint* actual.
  - Gestión de elementos de trabajo.

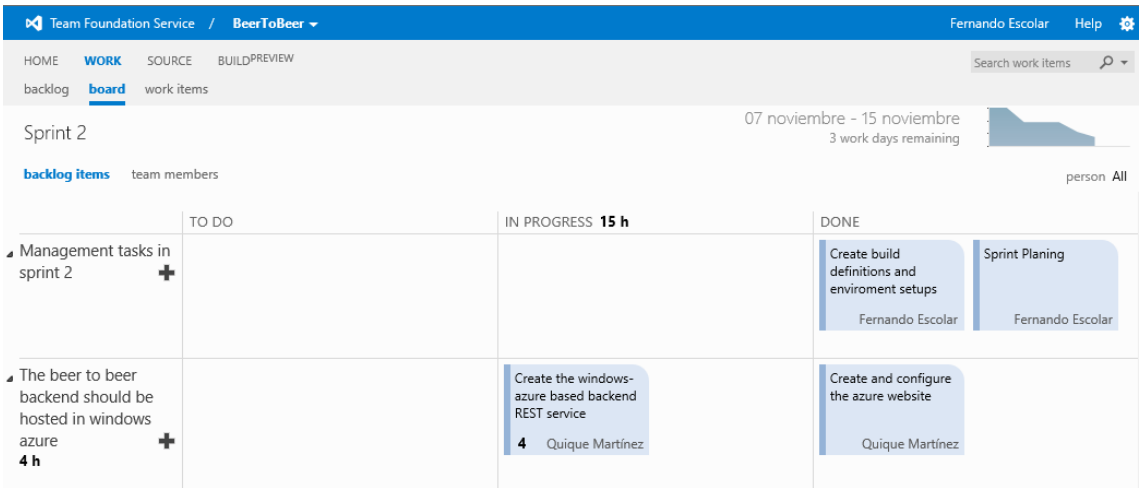


Figura 21 TSF backlog board (Microsoft, Escolar, 2017).

- Control de código fuente.
  - Desde el portal web también se tendrá acceso a una visión de todos los archivos almacenados en el control del código fuente.

- Automatización de *builds*[23], pruebas y despliegues.

[23] Compilación y construcción de la aplicación

Según (Gartner Inc., 2015), las fortalezas de **TFS** son:

- Las funcionalidades son fáciles de implementar.
- Despliegue de funcionalidades adicionales por parte de **Microsoft**.
- Dispone de funcionalidades ágiles y tradicionales.

Y sus debilidades son:

- Cambio hacia el móvil.
- La administración de requisitos se hace con el paquete *Office*.

3.2.3. VersionOne

> "**VersionOne ALM** facilita la gestión del el ciclo de vida de la aplicación. Permite a los equipos, en todos los niveles, creación de estrategias, mejoras en el desarrollo de software y realización rápida de entregas. **VersionOne ALM** ofrece una cartera ágil y unificada, herramientas de gestión de proyectos, así como una gran colaboración en toda la empresa, comunidad, gestión de calidad e inteligencia empresarial" (VersionOne, 2017).

Características principales

- Cartera ágil y unificada.
- Herramientas para la gestión del **LF**[24].
- Herramientas para la gestión de calidad.
- Herramientas para la inteligencia empresarial.
- Entorno colaborativo a todos los niveles.
- Plataforma fácilmente integrable con otras herramientas de desarrollo software.
- Facilita el cambio hacia metodologías ágiles.

Según (Gartner Inc., 2015), las fortalezas de **VersionOne** son:

- Licencias flexibles.
- Distribución *Sass* y *on-premise*.
- Fácil de implementar.
- Escalable.

[24] Life Cycle

Y sus debilidades son:

- Decae su efectividad en organizaciones grandes.
- Limitada funcionalidad en torno a la gestión de calidad.

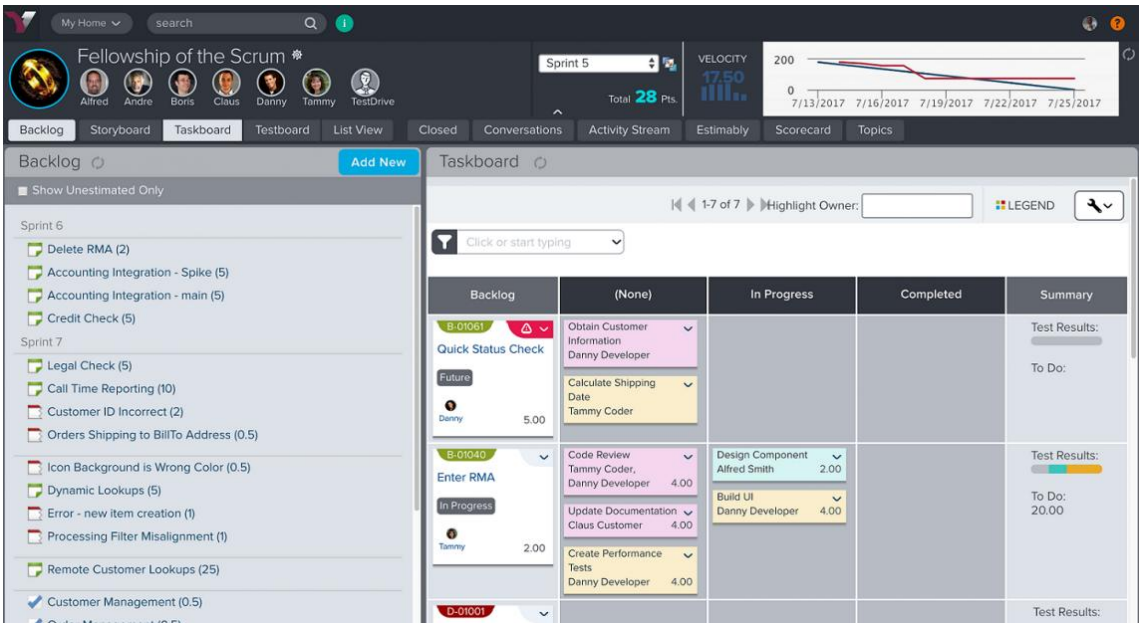


Figura 22 VersionOne backlog board (VersionOne, 2017)

3.2.4. Rally (CA Technologies)

Según (Gartner Inc., 2015), **Rally** está profundamente involucrado en el desarrollo de software ágil, y se enfoca en herramientas y servicios para ayudar a las compañías a adoptar y escalar prácticas ágiles.

Fortalezas:

- El proveedor ofrece funcionalidad de configuración estándar para proyectos ágiles, programas y administración de carteras.
- Rally tiene una sólida comprensión de los principios ágiles, DevOps y SAFe.
- Los equipos de desarrollo ágil y calificado brindan una sólida experiencia en transformación y escala ágiles.

Debilidades:

- La instalación local.
- No muy óptimo para el desarrollo tradicional.
- Tiene competidores similares a menor coste.

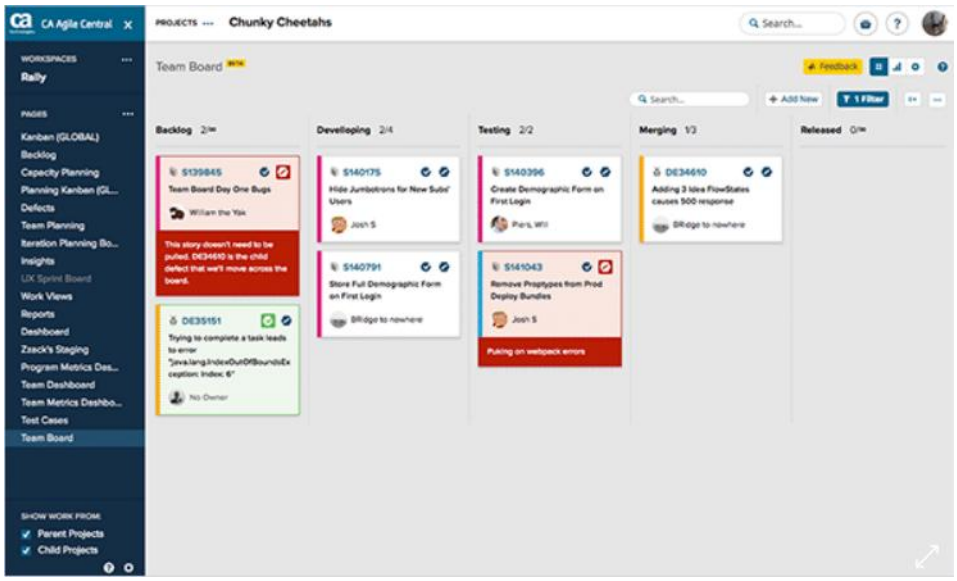


Figura 23 – CA Rally Board (CA Technologies, 2017)

3.2.5. Otras herramientas ágiles open source

Existen otras herramientas de código abierto en el mercado con similares funcionalidades y capacidades que las nombradas anteriormente y que dependiendo de la tipología de la empresa en la que se quiera implantar puede ser más que recomendable su uso. Según recoge un artículo del 2016 elaborado por la comunidad opensource.com (Muilwijk, 2016), las herramientas ágiles de código abierto más valoradas por lo usurarios son:

- MyCollab
- Odoo
- Taiga
- OrangeScrum
- Tuleap Open ALM
- Agilefant
- Redmine
-]Project-open[
- OpenProject
- LibrePlan
- ProjectLibre
- Trello
- Asana

3.3. Cuadro comparativo de las distintas herramientas

Según los factores más determinantes de las herramientas ágiles expuestos por **IBM** (IBM Software Rational, 2012) y **VersionOne** (VersionOne, 2017), y tras realizar búsquedas exhaustivas en diferentes portales web como ***G2crowd***[25]***, Project Management Zone***[26] y ***Finances Online***[27] comparando las distintas cualidades de cada herramienta, así como las puntuaciones otorgadas por los usuarios y según las características de cada aplicación comentadas en el apartado anterior recogidas en la web oficial de cada producto, se propone la siguiente tabla y algoritmo para elegir la herramienta ágil que mejor se adapte al proyecto.

Primero se plantea la tabla de puntuaciones según las herramientas a estudiar con valores entre 0 y 5[28]. En este caso se escogen las herramientas propietarias más usadas, que son: **Jira, MFS, CA Rally** y **VersionOne**, y se añaden otras tres herramientas más habituales pero basadas en software libre, como son: **Trello**, **AgileFant** y **Asana**. Así también se cubre una de las características más importante: el coste de la herramienta.

Descripción de los factores a evaluar:

- **Capacidades ágiles:** Enfoque a la persona por encima del proceso. Ayudar a la colaboración entre todas las partes interesadas en el proyecto.
- **Gestión integrada del ciclo de vida:** Planning, iteraciones (sprints), entregas y seguimiento del proyecto. Backlog y repositorios.
- **Facilidad de uso:** Personalización del *dashboard (tablero)*. Entorno interactivo.
- **Análisis e informes:** Crear informes, gráficas, etc... Comparación de datos.
- **Workspace:** Capacidad de tratar y personalizar distintas metodologías (XP, Scrum, Kanban, etc.). Interacción con la aplicación, (arrastrar y soltar elementos). Personalización del entorno.
- **Gestión del programa:** Planificación entre equipos distribuidos, administración de épicas, tareas, etc.
- **Despliegues e integridad:** *API web services* e integración con otras herramientas; como control de versiones, interacción continua, etc.
- **Comunicación y colaboración:** Notificaciones por email y RSS. Informes y seguimiento para los miembros del equipo.
- **Soporte de la comunidad:** Gran comunidad de usuarios.
- **Escalabilidad:** Adaptación de la herramienta al crecimiento y nuevas necesidades de la empresa.
- **Flexibilidad:** Compatible con los cambios del equipo.

[25] https://www.g2crowd.com
[26] https://project-management.zone
[27] https://comparisons.financesonline.com/
[28] Estas puntuaciones corresponden al estudio hecho de las distintas herramientas en el año 2017. Hay que tener en cuenta que estas herramientas están en continuo desarrollo, teniendo que ajustar los valores de la matriz base cada vez que tengan alguna actualización.

- **Disponibilidad en móvil:** A tener en cuenta, IOS, Android y Windows Phone.

Otros factores importantes:

- **Seguridad.**
- **Costes.**
- **Dimensión de la empresa.**
- **Sass y on-premise.**

FACTORES	JIRA	TFS	V.ONE	CA	TRELLO	A.FANT	ASANA
Capacidades ágiles	3	2	3	3	3	3	2
Gestión integrada del ciclo de vida	3	3	3	3	2	3	1
Facilidad de uso y personalización	1	1	2	2	4	2	4
Análisis e informes	3	2	3	3	3	2	1
Workspace	2	2	2	2	2	1	2
Gestión del programa	3	2	3	3	2	2	2
Despliegues e integridad	3	2	2	2	3	1	1
Seguridad	4	5	4	3	2	2	2
Soporte scrum & kanban	5	5	5	5	4	2	3
Comunicación y colaboración	3	2	3	2	2	3	2
Soporte de la comunidad	3	2	3	1	3	3	1
Escalabilidad y flexibilidad	3	2	3	2	2	2	2
Disponibilidad móvil	2	3	1	2	2	2	3
Coste asequible	3	1	1	2	5	5	5
Sass	1	1	1	1	1	1	1
Windows local	1	1	1	1	1	0	1
Linux local	1	0	0	0	0	1	1
Mac local	1	0	0	0	1	0	1
Start up	2	3	3	3	2	2	2
Pymes	3	2	3	3	2	2	3
Grandes compañías	3	2	3	3	1	1	1

Ahora, cuando se tenga que decidir qué herramientas sería la más conveniente para el proyecto, se rellenará el siguiente formulario:

FACTORES	VALOR (DEL 0 AL 5)
Capacidades ágiles	
Gestión integrada del ciclo de vida	
Facilidad de uso y personalización	
Análisis e informes	
Workspace	
Gestión del programa	
Despliegues e integridad	
Seguridad	
Soporte scrum & kanban	
Comunicación y colaboración	
Soporte de la comunidad	
Escalabilidad y flexibilidad	
Disponibilidad móvil	
Coste	
Sass	
Windows local	
Linux local	
Mac local	
Start up	
Pymes	
Grandes compañías	

Una vez rellenado el formulario, los valores se multiplican por el valor correspondiente a su misma posición en la tabla base, haciéndolo por cada columna de la tabla, es decir, una vez por cada herramienta. Luego se sumarán las columnas y aquella opción que tenga una puntuación más alta será la herramienta elegida.

4. Creación sitio wiki de soporte

Para la creación del wiki de soporte, se ha escogido la plataforma gratuita **Wikispaces**, https://www.wikispaces.com/ y se ha dado de alta un nuevo sitio con el nombre **https://jgarcianavarro.wikispaces.com**. Existirá un usuario administrador que será el encargado de configurar todo el espacio wiki, de crear las pantallas y asociar al proyecto los distintos usuarios.

Se aclara que el objetivo del presente capítulo, no es explicar la configuración o el funcionamiento concreto de una plataforma wiki, ya que dependiendo de la plataforma elegida, su interfaz, opciones o configuración cambiarán. Por consiguiente, la meta consiste en crear una plantilla de administración basada en la metodología **Scrum** para la gestión de un proyecto usando un sitio wiki, saber qué pantallas o artefactos ha de tener, así como la información a rellenar, los eventos y tareas del proyecto, etc. y así poder analizar si es óptimo su uso para dar soporte a la administración de un proyecto de desarrollo de *software*.[29]

4.1 Estructura del sitio wiki

Ahora se irán creando las distintas pantallas del sitio. Se puede cambiar el tipo de aplicación wiki desde el menú ajustes:

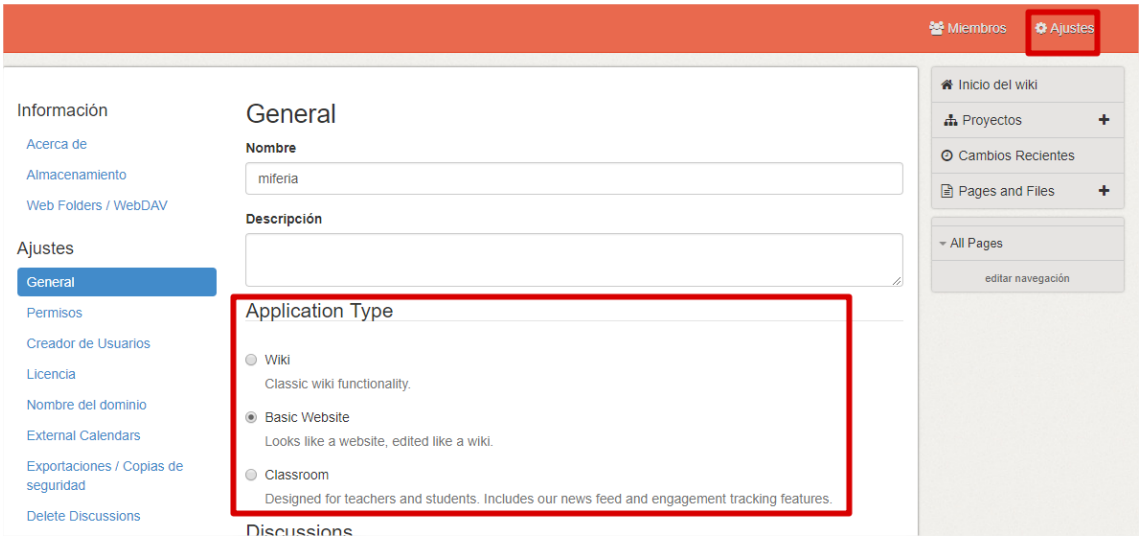


Figura 24 Cambiar estructura del sitio wiki[30]

[29] Se pueden consultar todas las funcionalidades del wiki en http://helpcenter.wikispaces.com/ enlazadas directamente desde el Anexo 2

[30] Todas las imágenes del presente capítulo son de elaboración propia

- Estilo **Classroom**

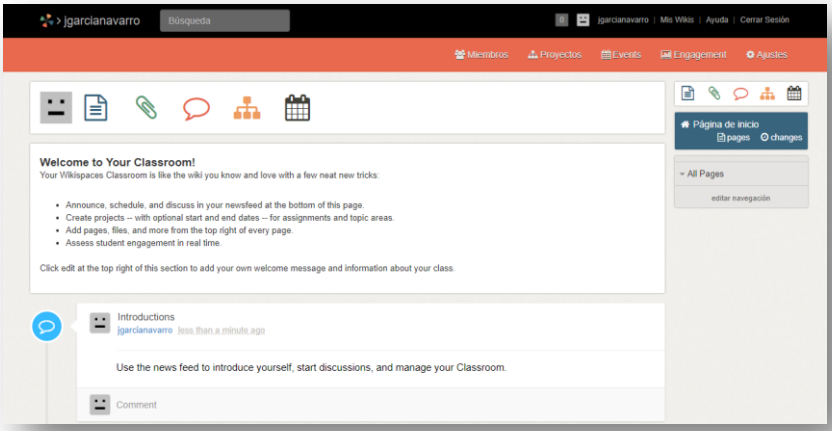


Figura 25 Panel de administración de wikispace – Estilo Classroom

- Estilo **Básico y Wiki**

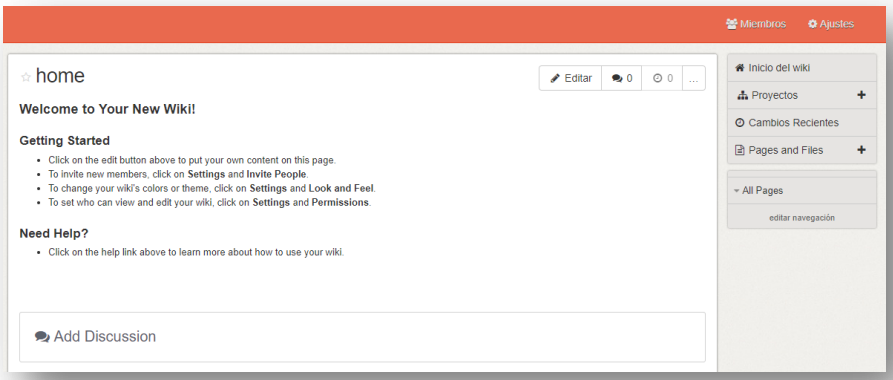


Figura 26 Estructura básica

Se trabajará con el estilo **Classroom** por tener un mejor diseño UX y se van a implementar en un proyecto tipo, los distintos artefactos y características de la metodología **Scrum**. Lo primero que se hace es crear el proyecto:

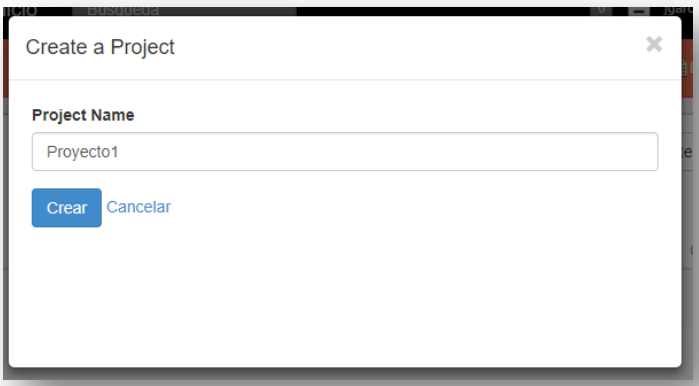


Figura 27 Crear un nuevo proyecto

Una vez creado, se añade los distintos artefactos implicados en él, y a cada usuario se le asigna los artefactos a los que pertenece.

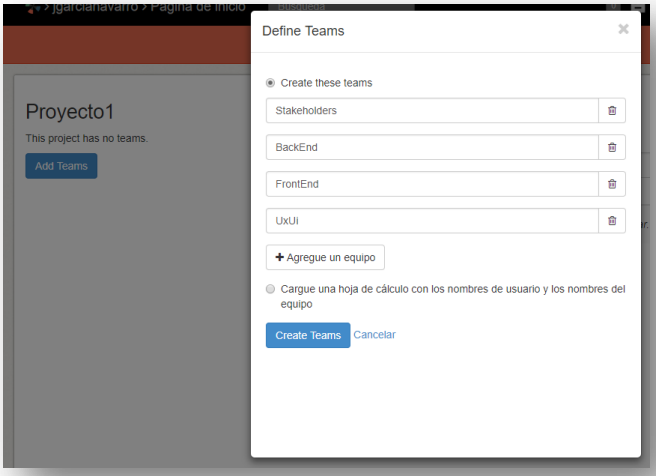


Figura 28 Crear artefactos

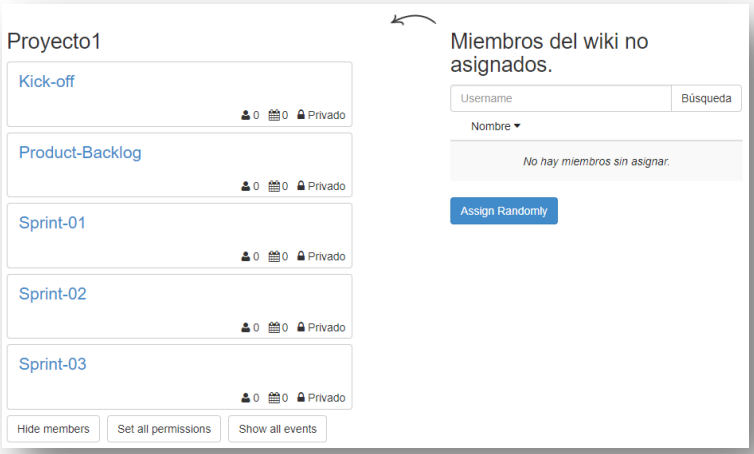


Figura 29 Asociar miembros

Se creará dentro del proyecto tantos "*teams*" (artefactos principales) como sean necesarios. Dentro de cada uno existirá la página principal, donde se pondrá los datos más significativos y se podrán crear, a su vez, otras pantallas necesarias. Como se aprecia en la imagen, las páginas de primer nivel que cuelgan directamente del proyecto serían: **Kick-off, Product-Backlog**, y los diversos **Sprints**.

Para comenzar se configura el artefacto **Kick-off**, del cual colgarán toda la información relativa al inicio del proyecto, como la descripción del mismo, las partes implicadas, costes, tecnologías, etc.

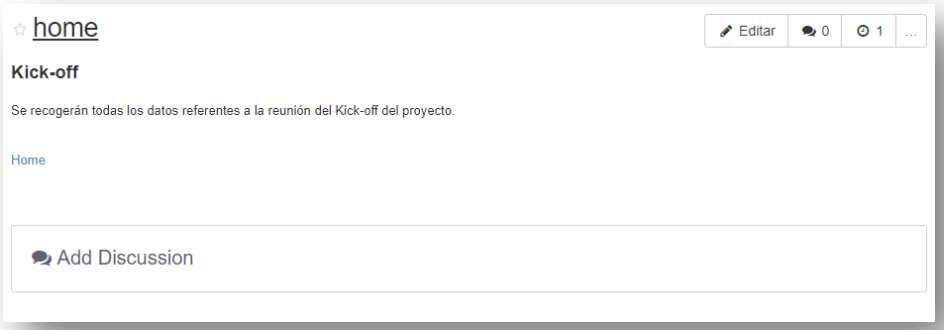


Figura 30 Kick-off

Luego se configurará las páginas de **Product Backlog**, creando dentro de él la página de historias de usuario y **Release Plan**:

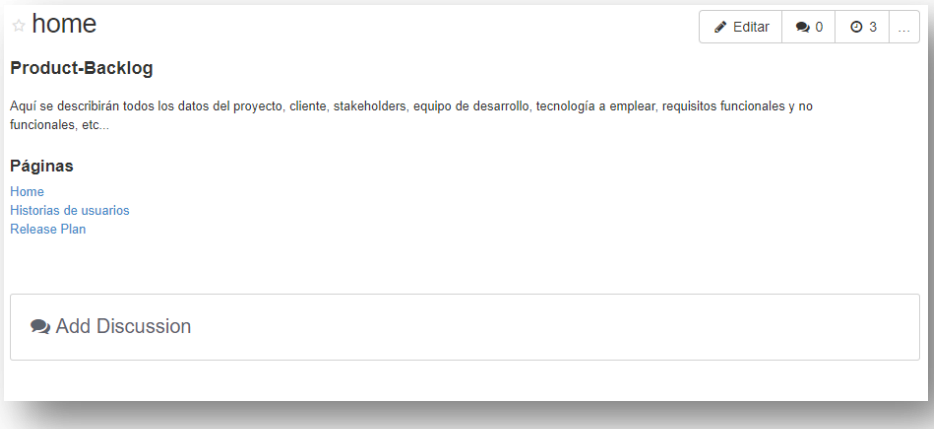


Figura 31 Product Backlog

Y por último se irán creando las páginas de los diferentes **Sprints**, incluyendo dentro otras páginas como: **Planning Meeting, Spring Backlog, Daily Scrum, Sprint Review** y **Sprint Retrospective**.

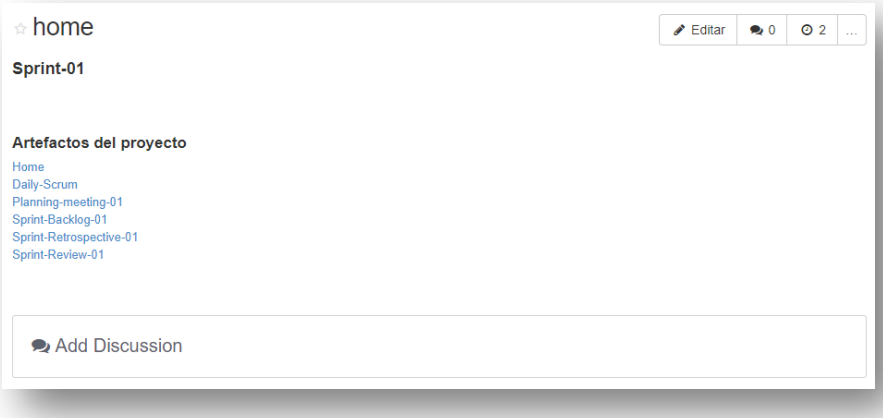


Figura 32 Página inicial de un Sprint

La estructura de pantallas quedará de la siguiente forma:

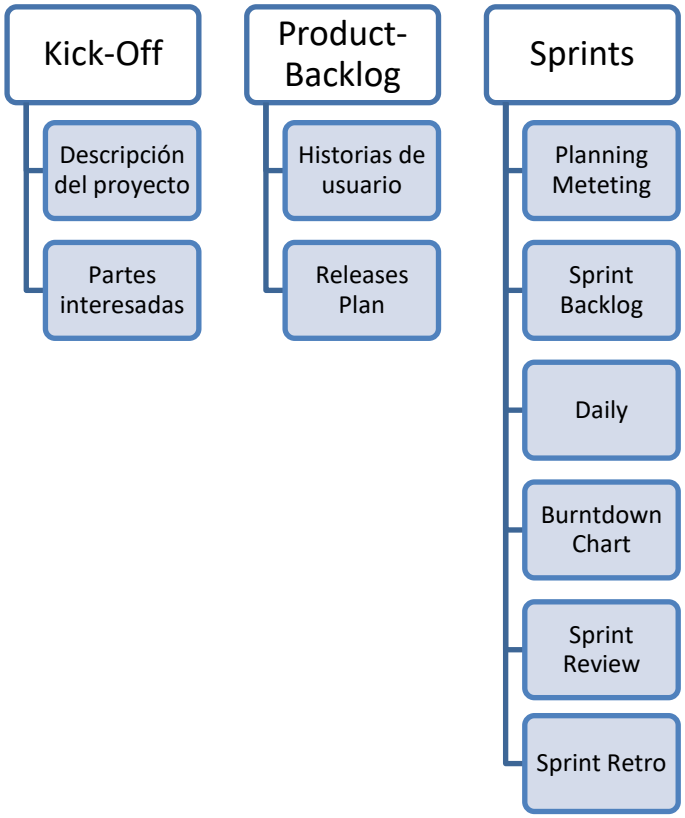


Figura 33 Estructura de pantallas de la plantilla wiki

Los **Sprints** se repetirán tantas veces como sea necesario.

4.2 Creación de plantillas

Ahora se crearán las plantillas con los formatos de las distintas páginas, para poder emplearlas cada vez que se inicie un **Sprint** o incluso un proyecto completo.

La creación de una nueva plantilla se realiza desde la pantalla donde se detallan todas las páginas del sitio pulsando el botón "*New template*":

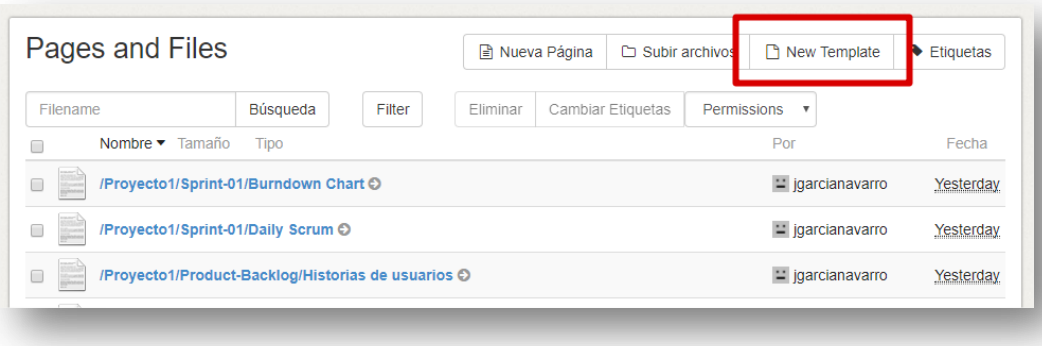


Figura 34 Nueva plantilla

Ahora se puede crear una plantilla desde cero o a partir de una página.

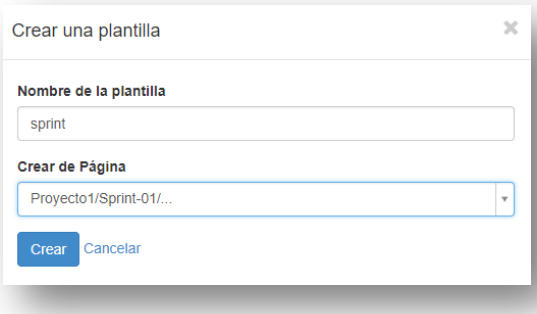


Figura 35 Popup de creación de plantilla

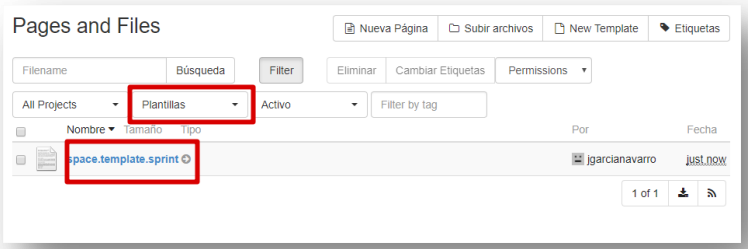


Figura 36 Plantilla creada

Se hace el mismo proceso para las páginas que se detallan a continuación.

4.2.1. Product Backlog

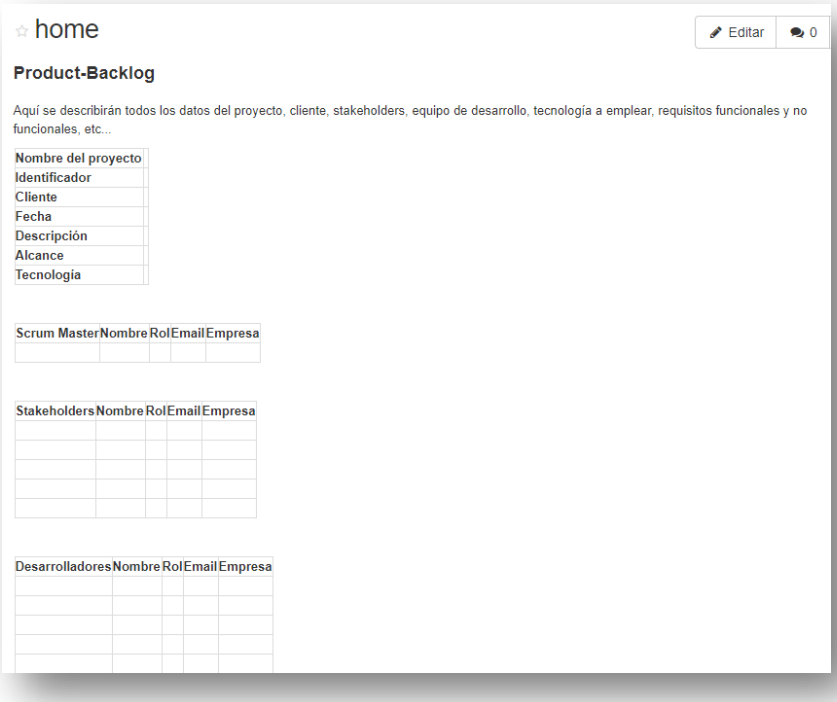


Figura 37 Plantilla para Product backlog

En la pantalla principal del **Product Backlog** se recogerá la información acerca de los datos generales del proyecto así como de las personas que intervienen en él.

a) Historias de usuario

Dentro del **Product Backlog**, se pasará a especificar las distintas historias de usuario que se tendrá que desarrollar durante el proyecto. A cada una se le dará una estimación en horas[31] y estas historias son las que se irán realizando durante los determinados **Sprints**.

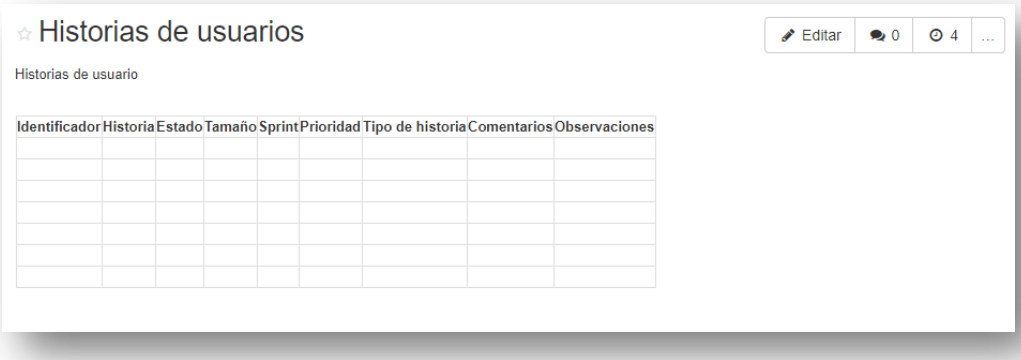


Figura 38 Plantilla para las historias de usuario

b) Fechas de entrega

Dentro del apartado **Product Backlog** también se detallarán las distintas entregas que se harán al cliente.

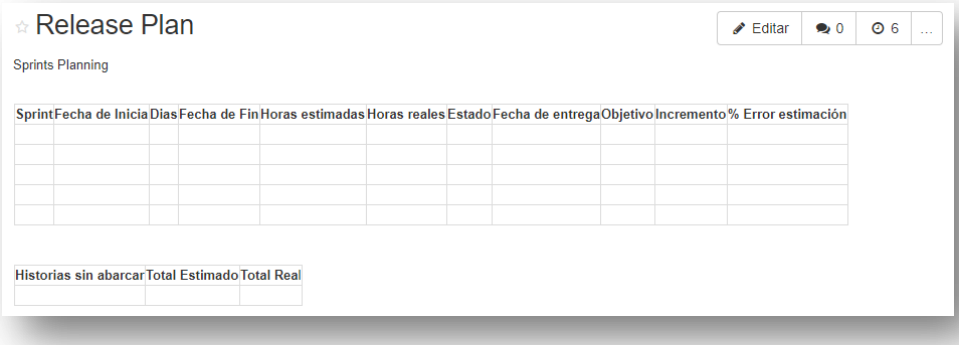


Figura 39 Plantilla para las fechas de entrega

[31] Existen diferentes técnicas de estimación que añadiremos en los anexos del **TFG**.

4.2.2. Sprints

En la página inicial de cada **Sprint**, se recogerá su información principal y de ella colgarán las siguientes páginas:

Figura 40 Pantalla principal de un Sprint

a) Sprint Retrospective

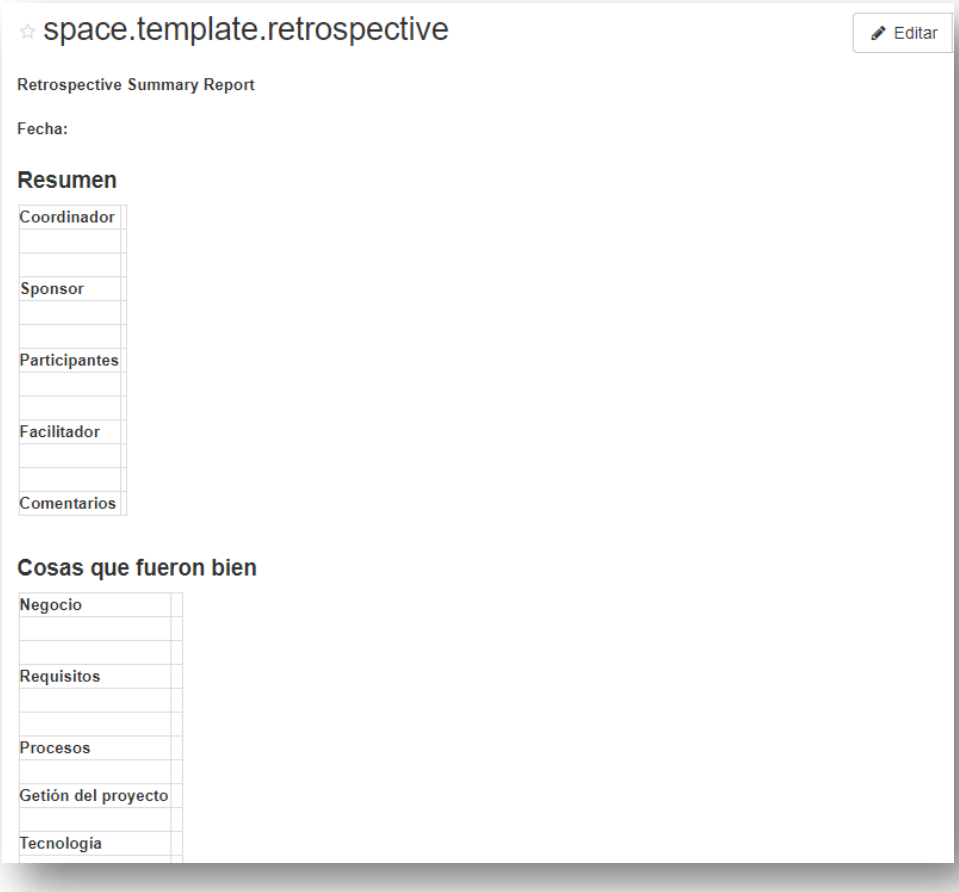


Figura 41 Pantalla de la Retrospective

b) Sprint Backlog

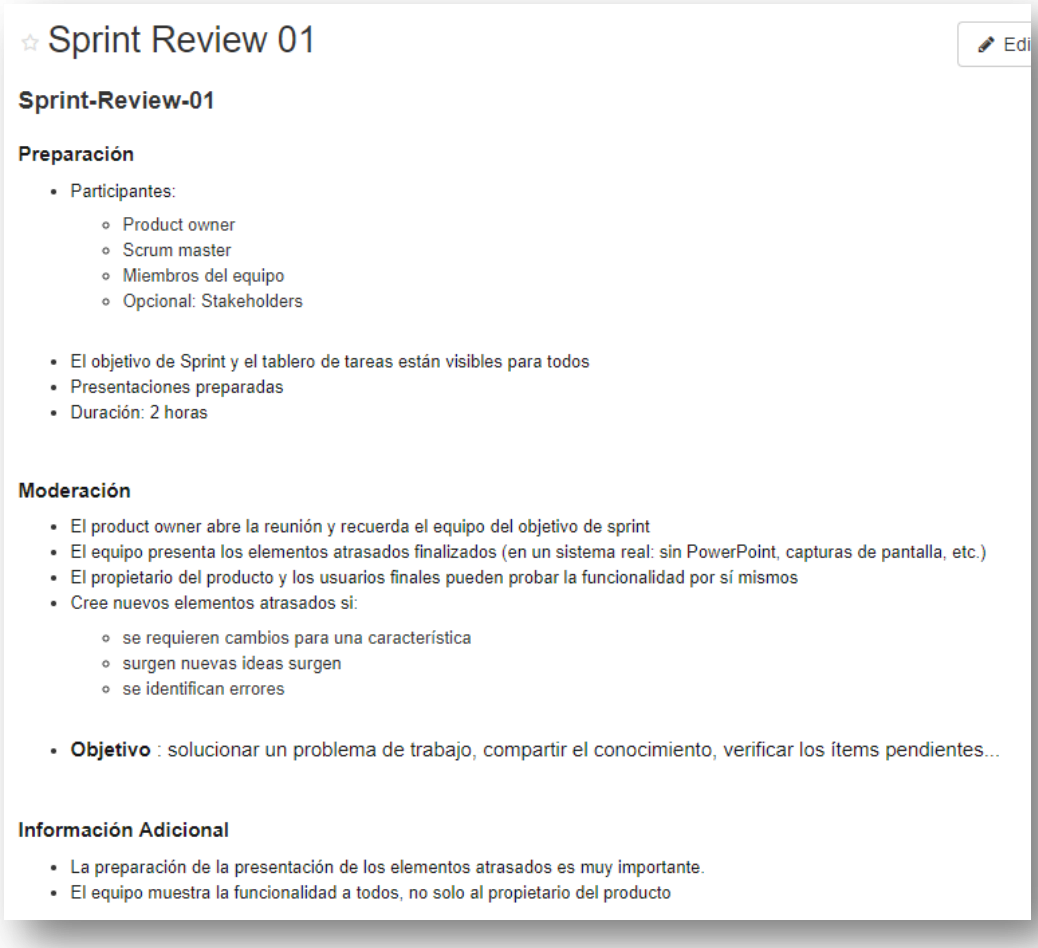
Sprint Backlog 01

Sprint Backlog 01

Duración del Sprint en días				Esfuerzo	Tiempo restante por día						
Tendencia calculada en base a la última			Horas totales								
Nombre de tarea	Historia ID	Responsable	Estado	Estimado	1	2	3	4	5	6	7

Figura 42 Plantilla para los Sprints

c) Sprint review



Sprint Review 01

Edi

Sprint-Review-01

Preparación

- Participantes:
  - Product owner
  - Scrum master
  - Miembros del equipo
  - Opcional: Stakeholders
- El objetivo de Sprint y el tablero de tareas están visibles para todos
- Presentaciones preparadas
- Duración: 2 horas

Moderación

- El product owner abre la reunión y recuerda el equipo del objetivo de sprint
- El equipo presenta los elementos atrasados finalizados (en un sistema real: sin PowerPoint, capturas de pantalla, etc.)
- El propietario del producto y los usuarios finales pueden probar la funcionalidad por sí mismos
- Cree nuevos elementos atrasados si:
  - se requieren cambios para una característica
  - surgen nuevas ideas surgen
  - se identifican errores
- **Objetivo** : solucionar un problema de trabajo, compartir el conocimiento, verificar los ítems pendientes...

Información Adicional

- La preparación de la presentación de los elementos atrasados es muy importante.
- El equipo muestra la funcionalidad a todos, no solo al propietario del producto

Figura 43 Pantalla del Sprint Review

4.2.3. Gráfico Burndown

Uno de los principales artefactos de **Scrum** sin duda es el **Burndown Chart**, mediante el cual se puede comprobar si un **Sprint** se va a terminar en el tiempo adecuado, proporcionando la capacidad de encauzarlo si la tendencia no es la correcta.

Como las páginas *wikispace* no admite la inserción de gráficas, se ha insertado un *widget* en las páginas del **Burndown** de cada **Sprint**, que carga una hoja de cálculo externa, que se deja preparada en *Google Docs*, como plantilla para la creación de dicho gráfico.

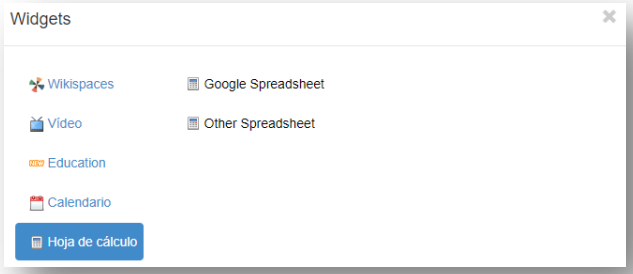


Figura 44 Insertar hoja de cálculo

Se publica la hoja de cálculo de *Google Docs*, generando un código **HTML** que se incluirá dentro del *widget* "Hoja de cálculo".

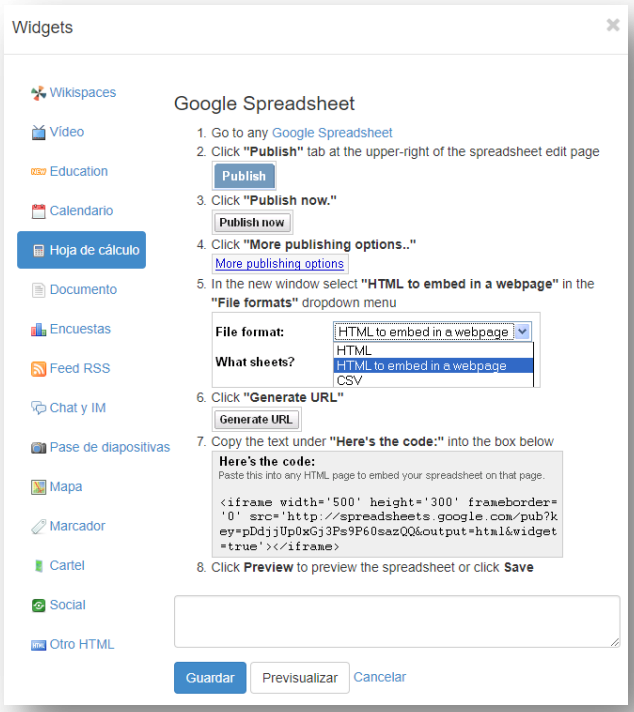


Figura 45 Embeber el código HTML

Quedando un gráfico **Brundown** como el del siguiente ejemplo:

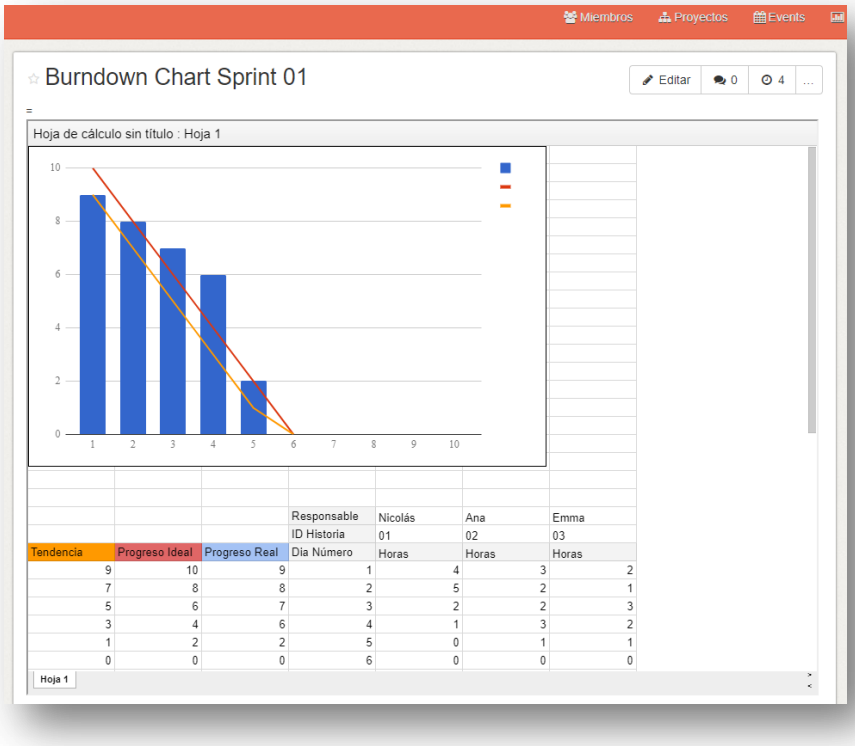


Figura 46 Pantalla Burndown Chart

Todos los cambios que se hagan en la tabla *Excel*, se verán reflejados en la gráfica.

4.2.4. Otras pantallas

Las demás pantallas solamente recogerán información detallada sobre el evento o artefacto del proyecto al que hacen referencia. Por ejemplo, la pantalla de **Kick-off**, recogerá toda la información previa a la reunión y un resumen de lo acontecido en ella.

4.3. Crear plantilla completa del proyecto

Una vez terminada toda la estructura base de un proyecto, se va a guardar una copia completa de todo el wiki, así cada vez que se inicie un proyecto nuevo no habrá que crear nuevamente todo, si no, solamente cargando la plantilla se tendrá disponible un nuevo proyecto en blanco.

En la pestaña ajustes, se accede a la opción “Exportaciones / Copias de seguridad” y se crea la copia.

Figura 47 Exportar wiki

El archivo comprimido se descargará al ordenador personal pero también se quedará en el sitio wiki. Ahora cada vez que se inicie un nuevo proyecto, se podrá crear partiendo de la base ya guardada.

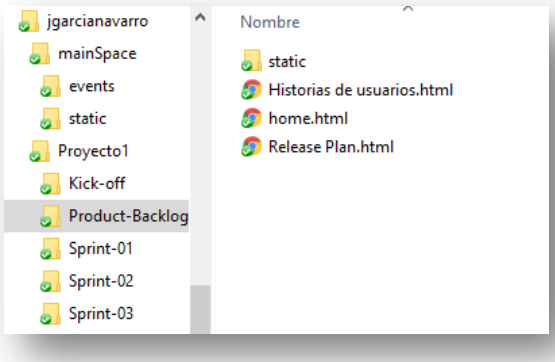


Figura 48 Estructura de carpetas del sitio wiki

El archivo descargado contendrá en formato HTML estático toda la estructura del proyecto.

4.4. Añadir comentarios

En todas las páginas de la wiki, se puede insertar comentarios, como por ejemplo, dudas, aclaraciones o información extra. En el pie de cada página se puede ver el siguiente recuadro:

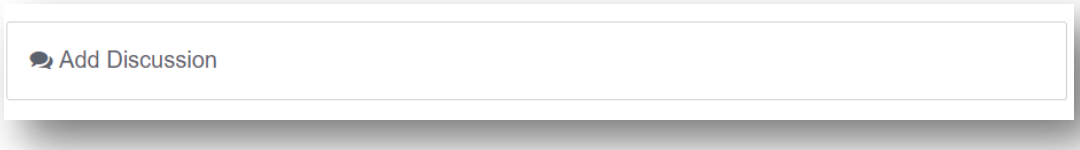


Figura 49 Iniciar una discusión

Pulsando dicho botón aparecerá el formulario:

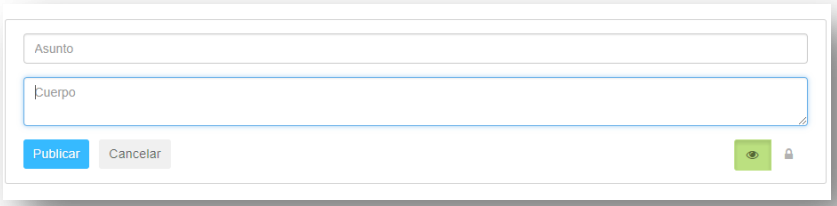


Figura 50 Escribir el comentario

Ahora se podrá publicar cualquier comentario e incluso dejar marcada la opción de que se informe cuando otro usuario responda.

4.5. Programar eventos

Otra posibilidad que proporciona la wiki es la creación de eventos en la agenda del proyecto. Pulsando el icono "*add event*":

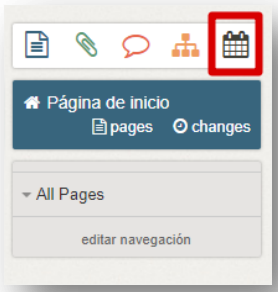


Figura 51 Iniciar evento

Se abre la siguiente pantalla emergente:

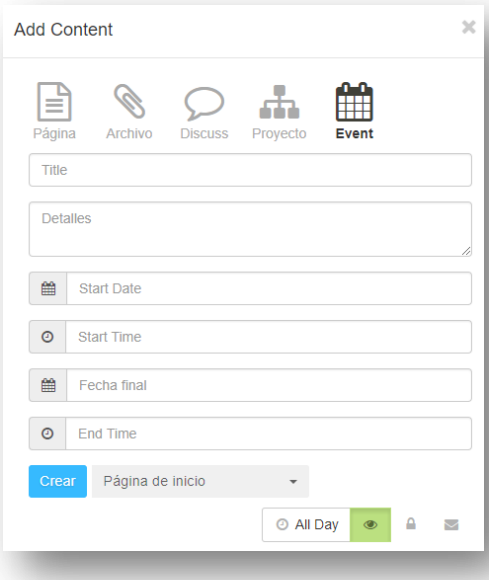


Figura 52 Confirmar evento

Donde se puede especificar el asunto, el mensaje, la fecha y hora de inicio del evento, y la fecha y hora de finalización. También se puede seleccionar que se avise por email a los miembros asociados a esa página. Por ejemplo, si se crea el evento en la página de **Sprint Review**, se le enviará a aquellos miembros asociados a ella.

4.6. Agregar miembros

El administrador de la wiki puede invitar a distintos usuarios a que se incorporen al proyecto. Para ello se tiene un panel de administración, que aparece tras pulsar el botón “miembros” donde se puede o bien invitar directamente a los usuarios o crear un enlace para compartirlo y que se agreguen ellos mismos.

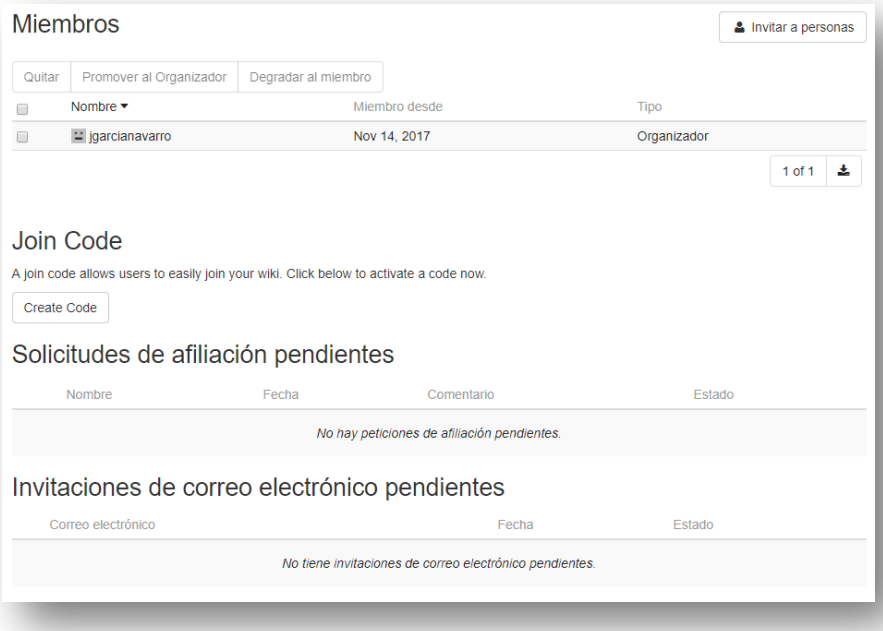


Figura 533 Agregar miembros al sitio wiki

Una vez agregado el usuario al sitio wiki el administrador tendrá que darlo de alta en cada página a la cual va a tener acceso. Desde la pantalla principal del proyecto se arrastra el usuario deseado al artefacto al que puede tener acceso.

Figura 544 Añadir miembros al proyecto

4.6.1. Agregar usuarios de forma masiva

En la pantalla ajustes del sitio, se puede importar, desde un archivo externo, todos los usuarios que se necesite. De esta manera se hace más cómoda la inclusión de miembros al proyecto.

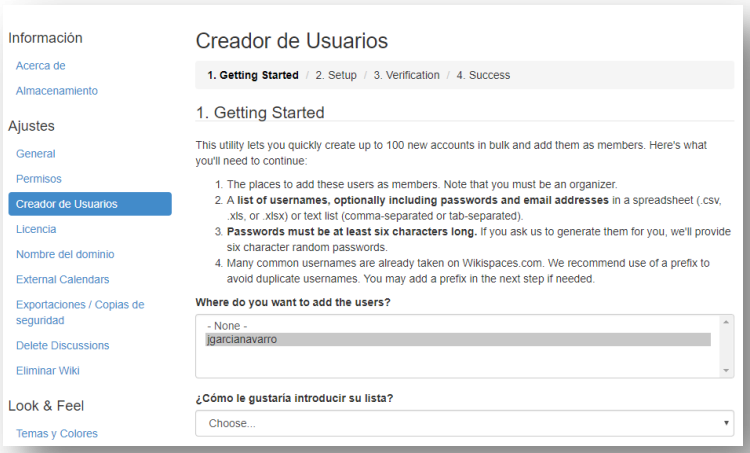


Figura 555 Creación masiva de usuarios

4.7. Asignar permisos y roles

Una vez se tengan incorporados al wiki (proyecto) y a las distintas páginas a los usuarios, se les pueden asignar determinados permisos según las necesidades.

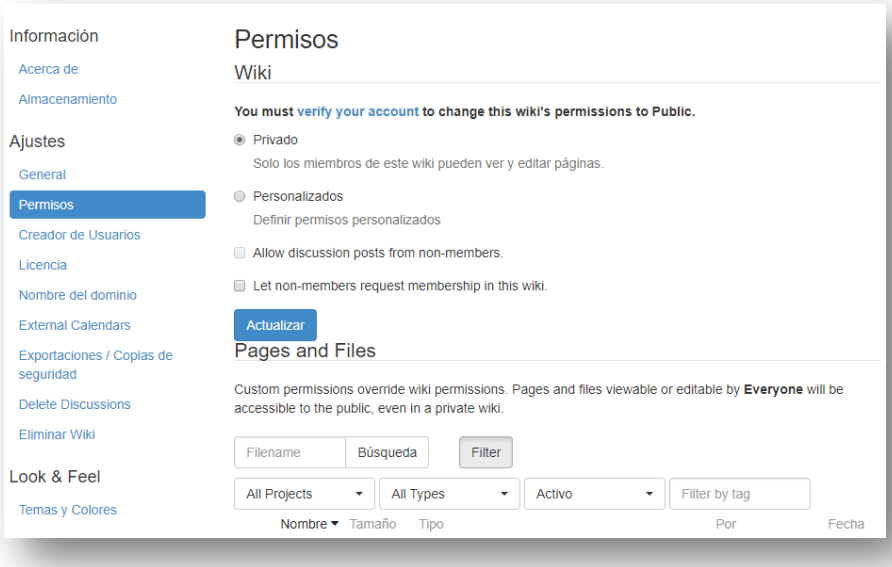


Figura 566 Asignar permisos

5. Simulación de la gestión de un proyecto

El presente capítulo servirá de prueba empírica de todo lo que se ha analizado en los capítulos anteriores. De este modo, a partir de unos requisitos "reales", se realizará la gestión de un proyecto de desarrollo software.

Se elegirá qué tipo de metodología se adaptará mejor a las especificaciones del proyecto y qué herramienta de software sería la más recomendada. Por último se configurará una wiki, partiendo de la estructura creada en el capítulo anterior, para usarla como sitio de soporte y gestión del proyecto hasta llegar al primer **Sprint Retrospective**.

5.1. Descripción del proyecto

Partiendo de la descripción del pliego técnico que se adjunta en el **Anexo 3**, el proyecto consiste en crear un diseño nuevo del portal miferia.com, diferenciando un diseño específico para resoluciones de escritorio y otro específico para dispositivos móviles.

Una vez aprobado el nuevo diseño, se realizará una nueva maquetación del portal, haciéndolo compatible con los distintos navegadores web y que tenga un diseño *responsive* para que también pueda ser visualizado de manera correcta desde el navegador de cualquier dispositivo móvil.

Por último, se creará una aplicación móvil híbrida mediante Cordova compatible tanto en Andorid como en IOS. Esta **app** al abrirla servirá de *launcher*[32] cargando en un Webview el portal web.

Se detallan las principales características del proyecto:

- El cliente estará involucrado directamente en el desarrollo del proyecto.
- Hasta que el cliente no apruebe el diseño del portal no se empezará con la maquetación y desarrollo del producto.
- La maquetación y desarrollo no sufrirán cambios de alcance.
- Se tendrá que hacer entregas periódicas funcionales del producto.
- El proyecto tiene una fecha fin cerrada, el 30 de Junio del 2018.
- Los integrantes del equipo deberán estar asignados al 100% y salvo motivo de fuerza mayor, deberá ser el mismo durante todo el proyecto.

Se hará la simulación desde el punto de vista del adjudicatario, que será una pequeña empresa **Start-up**, que una vez ganado el proyecto, realizará los pasos que se detallan en los siguientes apartados.

[32] Lanzador

5.2. Selección de metodología

Ahora se rellena el formulario planteado en el capítulo 2 del presente **TFG** para determinar qué metodología, tradicional o ágil, sería la más adecuada para realizar la gestión de este proyecto.

FACTOR	VALOR			
Prioridad de negocio	Cumplimiento - Valor			
	○	●	○	○
Tiempo del proyecto	Largo - Corto			
	○	○	○	●
Tamaño del equipo	Grande - Pequeño			
	○	○	○	●
Interacción con el cliente	Mínima - Máxima			
	○	○	○	●
Ambiente del desarrollo	Controlado - Incertidumbre			
	○	○	●	○
Dirección	Organizativa - Colaborativa			
	○	○	●	○
Rigidez del producto	Cerrado - Ampliable			
	●	○	○	○
Requisitos	Claros - Ambiguos			
	○	●	○	○
Riesgo de fallos	Bajos - Altos			
	○	○	●	○
Enfoque de desarrollo	Procesos - Personas			
	○	○	●	○
Necesidad de documentación	Alta - Baja			
	○	○	○	●
Probabilidad de cambios en el equipo	Alta - Baja			
	○	○	○	●
Roles intercambiables	No flexible - Flexible			
	○	○	●	○
Frecuencia de entregables	Baja - Alta			
	○	○	○	●
VALORES	**1**	**2**	**3**	**4**
TOTAL	**1**	**4**	**15**	**24**
VALORACIÓN	**44/14 = 3,14**			

Luego se suman todos los puntos obtenidos y se divide por el número de factores (14) dando un total de 3.14 puntos. Si se analiza con respecto a la recta de los valores:

Se observa claramente que el planteamiento del proyecto contiene más características cercanas a una metodología ágil, siendo la elegida para la gestión del proyecto, que a una tradicional.

Por último, se analizará cuál de las distintas metodologías ágiles es la que mejor se puede adaptar al caso de estudio. Tal como se comenta en el capítulo 2 del presente **TFG**, se rellena el siguiente formulario donde se le dará una puntuación según la necesidad del factor o funcionalidad del 0 al 5, siendo 0 que no se necesita y 5 que tiene la importancia más alta:

FACTOR	VALOR (0 al 5)
Actividades de puesta en marcha	3
Adaptación a Incertidumbre	3
Cambiar plan de trabajo	0
Descripción de procesos	4
Documentación de usuario	2
Gran impacto al cambiar a miembros del equipo	3
Informe de calidad	0
Integración de cambios	2
Interacción con cliente	5
Iteraciones cortas	5
Modelado	3
Necesidad de gestionar el proyecto	4
Necesidad de interacción con usuarios finales	0
Offshoring	0
Peso ligero	5
Política de refactoring	0
Pruebas	4
Requisito funcional pueden cambiar	1
Respetar alto nivel de calidad	3
Rigurosidad en fechas de entrega	5
Satisfacción del usuario	5
Se pueden cambiar indicadores	0
Uso del sistema	0

Multiplicando el valor de cada columna, por las cantidades de la matriz base de las distintas metodologías ágiles y sumando los resultados de cada una (columna) quedaría:

FACTOR	DSDM	XP	LEAN	KANBAN	SCRUM
Actividades de puesta en marcha	3	3	0	0	3
Adaptación a Incertidumbre	3	3	0	3	3
Cambiar plan de trabajo	0	0	0	0	0
Descripción de procesos	4	4	4	4	4
Documentación de usuario	2	0	2	0	2
Gran impacto al cambiar el equipo	3	0	3	0	3
Informe de calidad	0	0	0	0	0
Integración de cambios	2	2	2	2	2
Interacción con cliente	5	5	5	5	5
Iteraciones cortas	5	5	5	0	5
Modelado	3	3	0	0	3
Necesidad de gestionar el proyecto	4	0	0	0	4
Necesidad de interacción con usuarios finales	0	0	0	0	0
Offshoring	0	0	0	0	0
Peso ligero	0	5	5	5	5
Política de refactoring	0	0	0	0	0
Pruebas	4	4	4	4	4
Requisito funcional pueden cambiar	1	1	1	1	1
Respetar alto nivel de calidad	0	0	3	0	0
Rigurosidad en fechas de entrega	5	0	0	0	5
Satisfacción del usuario	5	5	5	0	5
Se pueden cambiar indicadores	0	0	0	0	0
Uso del sistema	0	0	0	0	0
TOTAL	**49**	**40**	**39**	**24**	**54**

Se aprecia claramente como la metodología ágil que mejor se adapta las características del proyecto es la metodología **Scrum**. Por lo tanto, se opta por emplear esa metodología como base para la gestión del proyecto.

5.3. Selección de la herramienta de gestión

Se parte del supuesto que la empresa solicitante es una **Start-up**, realizando la siguiente valoración de las herramientas de gestión:

FACTORES	VALOR (DEL 0 AL 5)
Capacidades ágiles	5
Gestión integrada del ciclo de vida	1
Facilidad de uso y personalización	5
Análisis e informes	3
Workspace	1
Gestión del programa	3
Despliegues e integridad	3
Seguridad	2
Soporte scrum & kanban	5
Comunicación y colaboración	4
Soporte de la comunidad	5
Escalabilidad y flexibilidad	0
Disponibilidad móvil	0
Coste	5
Sass	5
Windows local	0
Linux local	0
Mac local	0
Start up	4
Pymes	2
Grandes compañías	0

Por lo tanto, si se aplica estos valores a la matriz base expuesta en el capítulo 3, donde se puntúan la capacidad de las distintas herramientas según unos factores y necesidades claves, nos genera el siguiente resultado:

FACTORES	JIRA	TFS	V.ONE	CA	TRELLO	A.FANT	ASANA
Capacidades ágiles	15	10	15	15	15	15	10
Gestión integrada del ciclo de vida	3	3	3	3	2	3	1
Facilidad de uso y personalización	5	5	10	10	20	10	20
Análisis e informes	9	6	9	9	9	6	3
Workspace	2	2	2	2	2	1	2
Gestión del programa	9	6	9	9	6	6	6
Despliegues e integridad	9	6	6	6	9	3	3
Seguridad	8	10	8	6	4	4	4
Soporte scrum & kanban	25	25	25	25	20	10	15
Comunicación y colaboración	12	8	12	8	8	12	8
Soporte de la comunidad	15	10	15	5	15	15	5
Escalabilidad y flexibilidad	0	0	0	0	0	0	0
Disponibilidad móvil	0	0	0	0	0	0	0
Coste	15	5	5	10	25	25	25
Sass	5	5	5	5	5	5	5
Windows local	0	0	0	0	0	0	0
Linux local	0	0	0	0	0	0	0
Mac local	0	0	0	0	0	0	0
Start up	8	12	12	12	8	8	8
Pymes	6	4	6	6	4	4	6
Grandes compañías	0	0	0	0	0	0	0
SUMA TOTAL	**146**	**117**	**142**	**131**	**152**	**127**	**121**

La herramienta mejor valorada en general ha sido **Trello**, que según las características del proyecto sería una buena opción, ya que se ha dado mucho valor al coste asequible de la herramienta al ser la empresa adjudicataria una **Start-up**. La herramienta propietaria más valorada sería **JIRA**, que quedaría en segundo lugar en la clasificación global.

5.4. Creación del wiki de soporte del proyecto

Una vez elegida la metodología ágil **Scrum** como metodología a seguir, se va a comprobar de forma empírica la viabilidad del uso de un sitio wiki como soporte a la gestión ágil de un proyecto de desarrollo de software. Se creará un nuevo proyecto en el sitio wiki **http://jgarcianavarro.wikispaces.com/**, usando las plantillas ya realizadas para tal efecto en el capítulo 4 del presente **TFG**. Esta simulación llegará hasta la finalización del **Sprint** 1 del proyecto.

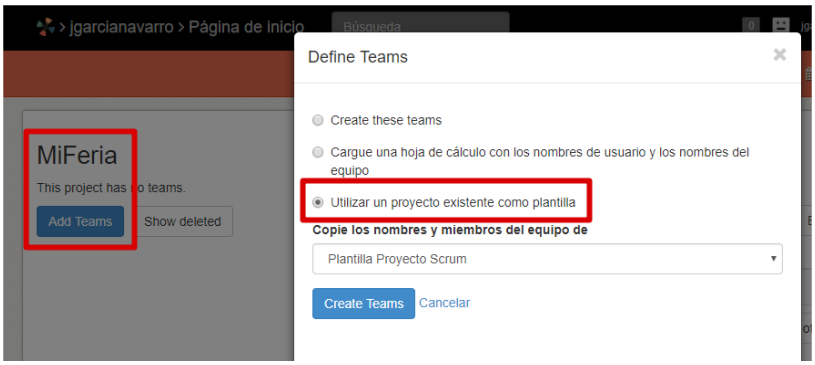


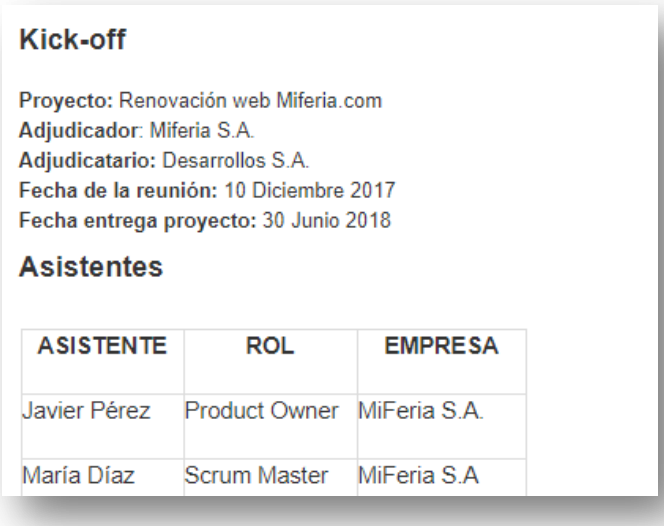
Figura 577 Creación del proyecto MiFeria

Se configura cada artefacto con las mismas páginas que el prototipo y se vuelcan las plantillas para obtener exactamente su misma estructura. El proyecto creado dentro del wiki será accesible desde la dirección **http://jgarcianavarro.wikispaces.com/project/manage/MiFeria**

5.5. Simulación Kick-off

Se prepara la reunión de inicio del proyecto y a ella asistirán los interesados en él. Por parte del cliente estarán el **Product Owner**, el trabajador integrado en el equipo de desarrollo y otros **Stakeholders**. Por el lado de la empresa desarrolladora participarán el gerente, el jefe de proyecto y los desarrolladores que intervendrán durante todo el proceso.

Esta reunión sirve para presentar al equipo y tener una primera toma de contacto con el proyecto. Además se planifican y estructuran sus objetivos y requisitos, se repasan el alcance y el calendario del mismo, se comentan las dudas e inquietudes y se da por comenzado el proyecto. En la wiki se recoge todo este proceso.



Kick-off

Proyecto: Renovación web Miferia.com
Adjudicador: Miferia S.A.
Adjudicatario: Desarrollos S.A.
Fecha de la reunión: 10 Diciembre 2017
Fecha entrega proyecto: 30 Junio 2018

Asistentes

ASISTENTE	ROL	EMPRESA
Javier Pérez	Product Owner	MiFeria S.A.
María Díaz	Scrum Master	MiFeria S.A

Figura 588 Página de Kick Off

5.6. Simulación Product Backlog

En la preparación del **Producto Backlog**, se realiza la reunión en la cual se obtendrán todas las historias de usuario a raíz de los requisitos funcionales, así como el calendario final de demostraciones y entregas.

- **Product Backlog**

Product-Backlog

Nombre del proyecto	Renovación web Miferia.com
Identificador	RWM-010
Cliente	Miferia S.A.
Fecha	15 Diciembre 2018
Descripción	Renovación diseño y web de Miferia.com y creación app híbrida
Alcance	Diseño – Maquetación – Desarrollo Front Angular
Tecnología	Sketch, Html 5 + CSS 3 + JS, Angular, Cordova

Figura 59 Página de Product Backlog

- **Historias de usuario**

Ahora se crean las historias de usuario y se dividen en tareas que se estimarán y se irán añadiendo a los distintos **Sprints**.

☆ Historias de usuario — Editar | 0 | 4

Identificador	Historia	Estado	Tamaño	Sprint	Prioridad	Tipo de historia	Comentarios	Observaciones
HU1	Como cliente quiero tener tener un diseño intuitivo y funcional para encontrar la información rápidamente	Pendiente	100		1	Diseño	El diseño deberá ser aprobado por el cliente	-
HU2	Como cliente quiero que la maquetación del portal se ajuste correctamente a la resolución del monitor para que todos los elementos sean accesibles	Pendiente	50		2	Maquetación	Maquetación responsive	Depende de HU1
HU3	Como usuario quiero ver el portal web desde el navegador	Pendiente	50		2	Maquetación	Maquetación responsive	Depende de HU1

Figura 590 Página de historias de usuario

- **Descomposición de historias en tareas**

Tareas y estimación

ID	Tarea	Estimación	Sprint	Responsable
HU1.1	Diseñar el menú superior de todo el portal	2 días	1	Elena Gil
HU1.2	Diseñar el menú lateral	1 día	1	Elena Gil
HU1.3	Diseñar la home	10 días	1	Elena Gil
HU1.4	Diseñar pantallas móvil	10 días	2	Elena Gil
HU2.1	Maquetar el menú superior	2 días	1	Jesús Salguero
HU2.2	Maquetar el menú lateral	2 días	1	Jesús Salguero

Figura 601 Página de tareas y estimación

- **Release Planning**

Sprints Planning

Sprint	Fecha de Inicio	Dias	Fecha de Fin	Horas estimadas	Horas reales	Estado	Fecha de entrega	Objetivo	Incremento	% Error estimación
1	08-01-2018	15	26-01-2018	208	215	Entregado	29-01-2018	Tener terminado y aprobado el diseño. Comienzo de la maquetación y de la arquitectura angular	7 horas	3.3%
2	22-01-2018	15	09-02-2018	248	-	Pendiente	12-02-2018	Tener terminada la maquetación en escritorio y móvil. Tener toda la comunicación con el Back lista.		
3	12-02-2018	10	23-02-2018	88	-	Pendiente	26-02-2018	Integración maqueta y front. Preparación de los launcher.		
4										

Figura 612 Página de fechas de entregas

5.5. Simulación Sprint 01 (Día 7)

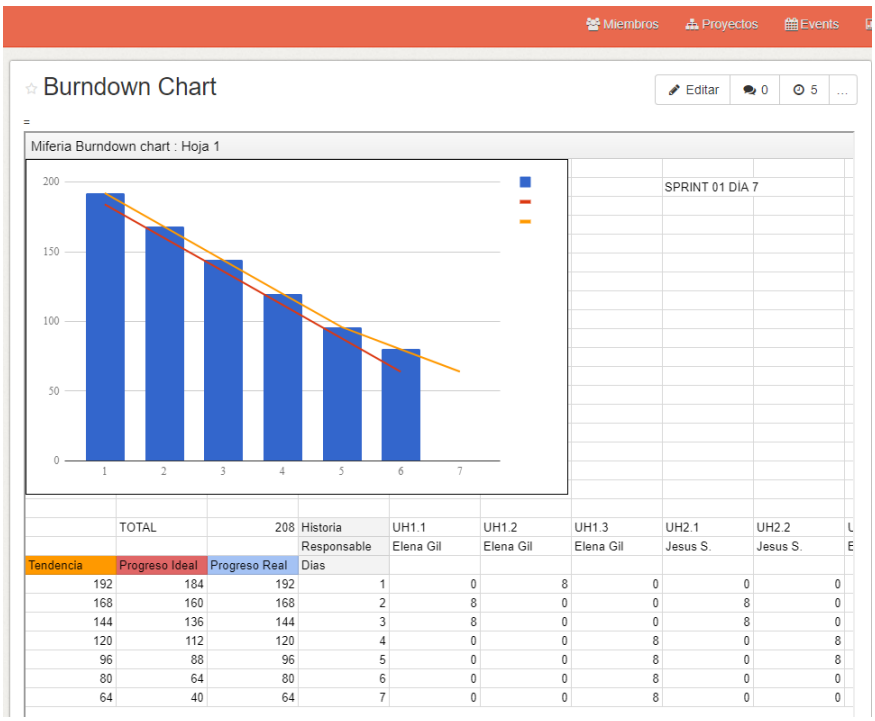
- **Sprint Backlog hasta el día 7**

Sprint Backlog 01 (Día 7)

Duración del Sprint en días				Esfuerzo	Tiempo restante por día						
Tendencia calculada en base a la última			Horas totales	208	16	24	24	16	24	24	16
Nombre de tarea	Historia ID	Responsable	Estado	Estimado	1	2	3	4	5	6	7
Diseñar el menú superior de todo el portal	HU1.1	Elena Gil	Finalizado	16 h		8	8				
Diseñar el menú lateral	HU1.2	Elena Gil	Finalizado	8h	8						
Diseñar la home	HU1.3	Elena Gil	En progreso	80h				8	8	8	8
Maquetar el menú superior	HU2.1	Jesús Salguero	Finalizado	16h					8	8	
Maquetar el menú lateral	HU2.2	Jesús Salguero	Finalizado	16h		8	8				
Crear arquitectura base en Angular	HU4.1	Ernesto Sevilla	Finalizado	24h	8	8	8				
Securización de la aplicación	HU4.2	Ernesto Sevilla	Finalizado	24h				8	8	8	
Preparación del multilenguaje	HU4.3	Ernesto Sevilla	En progreso	24h							8

Figura 623 Página del Sprint Backlog en el 7º día

- **Burndown Chart Sprint 01**



	TOTAL	208	Historia	UH1.1	UH1.2	UH1.3	UH2.1	UH2.2	U
			Responsable	Elena Gil	Elena Gil	Elena Gil	Jesus S.	Jesus S.	E
Tendencia	Progreso Ideal	Progreso Real	Dias						
192	184	192	1	0	8	0	0	0	
168	160	168	2	8	0	0	8	0	
144	136	144	3	8	0	0	8	0	
120	112	120	4	0	0	8	0	8	
96	88	96	5	0	0	8	0	8	
80	64	80	6	0	0	8	0	0	
64	40	64	7	0	0	8	0	0	

Figura 634 Burndown Chart del Sprint-1 en el séptimo día

6. Conclusiones

6.1. Conclusión general

Una vez finalizado el **TFG** se llega a la conclusión de que no se puede afirmar categóricamente que una metodología de gestión de proyectos de desarrollo de software sea mejor que otra. La elección de una metodología tradicional o una ágil dependerá del tipo de proyecto y de las necesidades específicas del cliente.

Sucede lo mismo con las distintas metodologías ágiles que existen. Hay que analizar concretamente las características del proyecto, al equipo, los plazos, etc., para determinar qué metodología ágil sería la más conveniente. Una de las principales conclusiones sobre estas metodologías, es que puede ser recomendable no centrarse exclusivamente en la aplicación exacta de una de ellas, sino que puede resultar muy beneficioso el incorporar distintos conceptos de ellas para establecer una metodología más correcta para nuestro proyecto.

Con respecto a las herramientas de gestión de proyectos ágiles, es conveniente realizar una elección muy precisa, ya que el uso de un software que no se adapte al equipo o al proyecto puede retrasar el desarrollo del proyecto e incluso provocar pérdida de información. Para dicha elección entran en juego factores muy importantes como el coste de licencias, escalabilidad, flexibilidad, facilidad de uso, etc. Por lo tanto, es muy importante poder probar dichas herramientas, estudiarlas y saber si aportarían valor a la gestión del proyecto.

Así pues, para aportar una ayuda a la hora de realizar estas elecciones, se ha propuesto una serie de formularios y fórmulas para conocer qué metodología o herramientas se adaptarían mejor al proyecto. La clave de los métodos expuestos está en la puntuación que se le otorga a cada una de las características de las metodologías o herramientas, para saber cuál conviene en cada caso concreto.

En referencia al uso de un sitio wiki de soporte para la gestión de un proyecto ágil, se llega a la conclusión de que aún siendo un buen mecanismo para llevar el control de ciertas partes del proyecto, pudiendo asignar roles a los usuarios, permisos, compartir información, subir archivos, etc., su administración se hace de una forma muy manual. Existen en el mercado diversas herramientas libres, incluso gratuitas, que pueden realizar la misma función pero donde los datos ya están relacionados, evitando la necesidad de duplicar manualmente la información, y que cuentan con una interfaz visual mucho más potente.

6.2. Logro de objetivos

Revisando los objetivos marcados al inicio del **TFG**, se puede afirmar que prácticamente se han cumplido todos, pero sería muy interesante analizar más herramientas de gestión e incluso haber podido probarlas todas, pero su tiempo y dedicación excedería el presente trabajo y además, algunas herramientas requieren pagos para poder probarlas en su totalidad.

Con respecto al sitio wiki, sería interesante dotarlo de un mejor aspecto visual e integrar los distintos miembros al proyecto y comprobar su interacción, pero esto requeriría crear varias cuentas de correo, verificarlas, etc. Aún así, ha cumplido la función de comprobar el uso del sitio para la gestión de un proyecto.

6.3. Seguimiento de la planificación y metodología

La planificación y metodología del trabajo ha sido bastante correcta y acertada, tras la corrección hecha después de la entrega de la PEC 1. Al final, tal como se explica en el punto anterior, lo único que se ha tenido que disminuir es el número de herramientas a estudiar.

6.4. Líneas de trabajo futuro

Se puede ampliar los principales factores y características de las distintas metodologías y herramientas, para realizar una valoración mucho más concreta y así acertar más a la hora de realizar la elección. Y realizar pruebas con varias definiciones de proyectos, para depurar lo máximo posible esos algoritmos planteados.

Probar otros servidores *wiki* y consultar sus **API** para poder enfocar, aún más, el sitio *wiki* y convertirlo en una mejor herramienta de gestión de proyectos.

Analizar otras herramientas de gestión y llevar el seguimiento de las actualizaciones que vayan realizando, como se ha comentado anteriormente, las herramientas se van mejorando y los valores expuestos en la matriz base de referencia pueden verse afectados.

7. Glosario

ADLM: Application Development Lifecycle Management.

AJAX: Asynchronous JavaScript and XML (JavaScript asíncrono y XML)

API: Application Programming Interface.

APMS: Agile Project Management Software.

Cordova: Entorno de desarrollo de aplicaciones móviles, Apache Cordova permite, a los programadores de software, construir aplicaciones para dispositivos móviles utilizando CSS3, HTML5, y Javascript.

CSS: Cascading Stylesheets (hoja de estilo en cascada)

DevOps: Development and Operations.

DSDM: Dynamic Systems Development Method.

HTML: HyperText Markup Language (lenguaje de marcas de hipertexto).

LF: Life Cycle (Ciclo de vida).

LSD: Lean Software Development.

PMBOK®: Project management body of knowledge. Marco conceptual y colección de lo que los profesionales consideran "buenas prácticas generalmente aceptadas en gestión de proyectos".

PMI: Project Management Institute.

PMLC: Projecto Management Life Cycle.

PMP: Project Management Profesional.

PRINCE2: Projects in Controlled Enviroments.

SaaS: Software as a Service.

SAFe: Scaled Agile Framework.

SDLC: Systems Development Life Cycle.

TFG: Trabajo Fin de Grado.

TFS: Team Foundation Server.

TI: Tecnologías de la información.

Webview: Componente que permite a las aplicaciones de Android mostrar contenido en el navegador.

Widget: Pequeño programa que son ejecutados por un motor de widgets.

XML: eXtensible Markup Language (Lenguaje de Marcado Extensible)

XP: Metodología Extreme Programming.

8. Bibliografía

Agile Alliance. (2001). Agile Alliance Org. Recuperado el 1 de Noviembre de 2017, de https://www.agilealliance.org/

Agile Manifesto Org. (2001). Recuperado el 30 de Octubre de 2017, de http://agilemanifesto.org/iso/es/principles.html

Atlassian. (2017). *Atlassian.com*. Recuperado el 12 de Noviembre de 2017, de https://es.atlassian.com/software/jira/agile#kanban

Beck, K. (2004). *Extreme Programming Explained: EMBRACE CHANGE.* Addison Wesley.

CA Technologies. (2017). *CA Technologies*. Recuperado el 3 de Diciembre de 2017, de https://www.ca.com/us/company/acquisitions/rally-is-now-ca-technologies.html

CeoLevel. (2017). *Ceolevel - Expertos en project management*. Recuperado el 08 de Diciembre de 2017, de http://www.ceolevel.com/4-tecnicas-para-estimar-pert-delphi-planning-poker-tshirt

FinancesOnline. (2017). *FinancesOnline*. Retrieved Noviembre 10, 2017, from https://project-management-software.financesonline.com/c/agile-project-management

Gartner Inc. (28 de Noviembre de 2017). Recuperado el 15 de 10 de 2017, de https://www.gartner.com/events-na/applications/wp-content/uploads/sites/2/2017/07/predicts_2017_application_de_316983.pdf

Gartner Inc. (2017). *Gartner peer insights*. Retrieved Noviembre 18, 2017, from https://www.gartner.com/reviews/market/application-development-life-cycle-management

Gartner Inc. (2015). *Magic Quadrant for Application Development Life Cycle Management.*

Iacovelli, A., & Souveyet, C. (2008). Framework for Agile Methods Classification. En J. Ralyté, I. Mirbel, & R. Deneckère, *Engineering Methods in the Service-Oriented Context* (págs. 91-101). Parías: Springer.

IBM Software Rational. (2012). *A practical guide to choosing theright agile tools for your team.* New York.

InfoQ. (4 de 10 de 2015). *InfoQ*. Recuperado el 5 de Noviembre de 2017, de https://www.infoq.com/articles/standish-chaos-2015

International Technical Support Organization IBM. (July de 2007). *Red Books IBM.* Recuperado el 20 de Octubre de 2017, de The IBM Rational Unified Process for System z: http://www.redbooks.ibm.com/redbooks/pdfs/sg247362.pdf

Junta de Andalucía. (3 de Abril de 2017). *Junta de Andalucía.* Recuperado el 06 de Diciembre de 2017, de http://www.juntadeandalucia.es/contratacion/document/download?refCode=2017-0000010982&refDoc=2017-0000010982-0

Martínez-Berganza, F. E. (26 de Mayo de 2017). *MSDN Microsoft - Team Foundation Service, tu herramienta ALM en la nube.* Recuperado el 12 de Noviembre de 2017, de https://msdn.microsoft.com/es-es/communitydocs/alm/team-foundation-service

Mendes Calo, K., Estevez, E., & Fillottrani, P. (s.f.). *http://sedici.unlp.edu.ar.* Recuperado el 3 de Noviembre de 2017, de http://sedici.unlp.edu.ar/bitstream/handle/10915/21086/Documento_completo.pdf?sequence=1

Menzinsky, A., López, G., & Palacios, J. (Julio de 2016). *Scrum Manager.* Recuperado el 2 de Noviembre de 2017, de http://scrummanager.net/files/scrum_manager.pdf

Microsoft. (2017). *MSDN Microsoft*. Recuperado el 20 de Octubre de 2017, de https://msdn.microsoft.com/es-es/library/jj161047(v=vs.120).aspx

Microsoft, Escolar. (26 de Mayo de 2017). *MSDN Microsoft - Team Foundation Service, tu herramienta ALM en la nube.* Recuperado el 12 de Noviembre de 2017, de https://msdn.microsoft.com/es-es/communitydocs/alm/team-foundation-service

Muilwijk, R. (28 de Marzo de 2016). *OpenSource.com*. Recuperado el 13 de Noviembre de 2017, de https://opensource.com/business/16/3/top-project-management-tools-2016

Palacio, J., & Ruata, C. (Julio de 2009). *Scrum Manager.* Recuperado el 18 de Octubre de 2017, de http://www.scrummanager.net/files/sm_proyecto_apuntes_12.pdf

Palacio, J., & Ruata, C. (2012). *Scrum Manager.* Recuperado el 19 de Octubre de 2017, de http://www.scrummanager.net/files/gestion_visual_kanban_apuntes121.pdf

PMI. (Marzo de 2013). *Project Management Institute*. Recuperado el 10 de Octubre de 2017, de https://www.pmi.org/-/media/pmi/documents/public/pdf/business-solutions/project-management-skills-gap-report.pdf

Poppendieck, M., & Poppendieck, T. (2003). *Lean Software Development: An Agile Toolkit.* Addison Wesley.

Pradel Miquel, J., & Raya Martos, M. (2015). ntroducción a la ingeniería del software. UOC.

The Standish Group. (2015). *http://www.standishgroup.com.* Recuperado el 15 de Noviembre de 2017

Uniminuto. (2011). Cuatro enfoques metodológicos para el desarrollo de Software RUP - MSF - XP - SCRUM. *Revista Inventum* , 64-78.

VersionOne. (2017, Abril 6). *Explore VersionOne.* Retrieved Noviembre 11, 2017, from 11th Annual State of Agile Report: https://explore.versionone.com/state-of-agile/versionone-11th-annual-state-of-agile-report-2

VersionOne. (2017, Abril 6). *Explore VersionOne.* Retrieved Noviembre 2017, from 11th Annual State of Agile Report: https://explore.versionone.com/state-of-agile/versionone-11th-annual-state-of-agile-report-2

VersionOne. (25 de Enero de 2017). *Info VersionOne.* Recuperado el 03 de Diciembre de 2017, de http://info.versionone.com/Agile-Platform-Evaluator.html

Wells, D. (2009). *Extreme Programming Org*. Recuperado el 2 de Noviembre de 2017, de http://www.extremeprogramming.org

Wikiversidad Contributors. (30 de Noviembre de 2017). Recuperado el 10 de Noviembre de 2017, de Wikiversidad: https://es.wikiversity.org/w/index.php?title=Procesos_de_desarrollo_software&oldid=113048

Wikiversidad Contributors. (Septiembre de 2015). *Wikiversidad*. Recuperado el 1 de Noviembre de 2017, de https://es.wikiversity.org/w/index.php?title=Dynamic_Systems_Development_Method&oldid=109349

Wysocki, R. K. (2014). *Effective Project Management – Traditional, Agile, Extreme – 7th Edition.* Wiley.

9. Webgrafía

Otras webs y blogs consultados:

- http://informatica.blogs.uoc.edu/2012/10/08/buenas-noticias-para-la-gestion-de-proyectos/
- https://www.infoq.com/articles/standish-chaos-2015
- http://www.laboratorioti.com/2016/05/16/informe-del-caos-2015-chaos-report-2015-bien-mal-fueron-los-proyectos-ano-2015/
- http://www.mountaingoatsoftware.com/blog/agile-succeeds-three-times-more-often-than-waterfall
- https://modelometodoygestion.wordpress.com/2017/02/21/chaos-report-15-scrum/
- https://proyectosagiles.org/control-predictivo-control-empirico/
- https://www.pmi.org/learning/library/agile-versus-waterfall-approach-erp-project-6300
- https://msdn.microsoft.com/es-es/library/jj161047(v=vs.120).aspx
- https://www.scrummanager.net/bok/index.php?title=Gesti%C3%B3n_predictiva
- https://www.obs-edu.com/es/blog-project-management/administracion-de-proyectos/tipos-de-proyectos-y-sus-principales-caracteristicas
- http://www.liderdeproyecto.com/articulos/pmbok_no_es_una_metodologia.html
- https://inusual.com/blog/gestion-de-proyectos-agile-o-tradicional
- https://www.beeva.com/beeva-view/metodologiasagiles/agilismo-vs-metodologia-tradicional/
- https://www.obs-edu.com/es/blog-project-management/temas-actuales-de-project-management/los-principios-que-rigen-el-lean-software-development
- https://www.recursosenprojectmanagement.com/software-de-gestion-de-proyectos/
- https://www.lancetalent.com/blog/8-herramientas-para-la-gestion-de-proyectos-profesionales/
- http://blog.masterinprojectmanagement.net/20-software-gratuitos-para-la-gestion-de-proyectos/
- http://pmopartners.es/cuadrante-magico-de-gartner-para-la-gestion-de-proyectos-y-portafolio-de-proyectos-de-ti-basada-en-la-nube/
- https://projectmanagers.org/gartner-magic-quadrant-ppm-software-tools-2016/
- https://www.odpe.com/es/software-gestion-de-proyectos-cual-es-el-mejor/
- https://bbvaopen4u.com/es/actualidad/gestion-de-proyectos-agile-y-scrum-una-seleccion-de-plataformas-utiles
- http://www.iebschool.com/blog/herramientas-gestion-agil-proyectos-agile-scrum/
- https://project-management-software.financesonline.com/c/agile-project-management
- https://www.softwareadvice.com/project-management/agile-comparison/
- https://blog.capterra.com/agile-project-management-software/
- http://tracks.roojoom.com/r/28211#/trek?page=7
- https://www.softwareadvice.com/resources/agile-project-management-user-trends-2015/
- http://info.versionone.com/Agile-Platform-Evaluator.html
- https://agileuniverse.wordpress.com/2010/10/04/workflow/

10. Anexos

ANEXO 1. Técnicas de estimación de tareas

Se detallan algunas técnicas de estimación (CeoLevel, 2017):

- **PERT (Project Evaluation and Review Techniques)**

  Es una técnica de estimación por 3 puntos donde se toman 3 valores:

  - La estimación en el peor de los casos, pesimista (P).
  - La estimación en el mejor de los casos, optimista (O).
  - La estimación más probable (M).

  Ahora a cada tarea se le aplica la siguiente fórmula (O+P+4M)/6 y se obtendrá un valor más cercano a la realidad.

- **Método Delphi**

  Hay que llegar a un consenso entre expertos. Se elabora un cuestionario que deberá ser contestado por expertos en la materia. Se comparten los resultados y se vuelve a hacer el mismo cuestionario y se vuelve a compartir los resultados. Este proceso se repite hasta alcanzar un consenso. Es una técnica bastante precisa pero conlleva bastante tiempo realizarla.

- **Planning Poker**

  Esta variante del método **Delphi** necesita un grupo de cartas con la secuencia de *Fibonacci*, incluyendo el cero. Algunos mazos incluyes cartas con el signo de interrogación para mostrar total incertidumbre.

  Existe la figura de un moderador que gestionará el proceso. Cuando se va a estimar una tarea, el empleado con más conocimiento de la misma pasa a exponerla y, a continuación, cada participante coloca una carta boca abajo que representa su estimación, normalmente, los valores representan puntos de esfuerzo.

  Luego se muestran todas las cartas, y las puntuaciones más altas y más bajas pasan a explicar y justificar sus valoraciones. El proceso se vuelve a repetir hasta que se alcance un consenso.

- **T-Shirt Sizing**

  Es una variante de la técnica planning poker, pero en vez de usar la serie de *Fibonacci*, se usan las tallas de las camisetas de ropa: *XS, S, M, L, XL* y *XXL*. Para así no asociar el valor numérico a horas de trabajo y poder crear una mejor abstracción del esfuerzo a realizar por cada tarea.

ANEXO 2. Manual Wikispace

Para más información acerca del uso, edición y administración de la plataforma *wikispaces*, a continuación se enlazan las distintas páginas de ayuda del sitio oficial http://helpcenter.wikispaces.com/:

Empezando

- Registrarse (sin invitación)
- Registrarse (con invitación)
- Unirse a Wiki
- Creando una Wiki
- Compartir contenido con otros

Configuración de Wiki

- Opciones comunes de Wiki
- Información
- Configuración
- Look & Feel
- Estadística

Configuraciones de la cuenta

- Barra de navegación global
- Tablero de usuario
- Sección de cuenta
- Sección de configuración
- Mensajería privada

Manejo de herramientas

- Barra de navegación
- Área de miembros
- Área de páginas y archivos
- Registro de cambios
- Historial de página

Editor de Wikispaces

- Editando una página
- Barra de herramientas del editor - Formateo
- Barra de herramientas del editor - Agregar contenido
- Barra de herramientas del editor - Comentando

Herramientas del administrador

- Proyectos
- Eventos
- Creador de usuarios
- Evaluación

ANEXO 3. Pliego de prescripciones técnicas

PLIEGO DE PRESCRIPCIONES TÉCNICAS[33] PARA LA CONTRATACIÓN DE LA PRESTACIÓN DE SERVICIOS PARA LA ACTUALIZACIÓN Y MEJORA DEL DISEÑO DEL PORTAL MIFERIA.COM[34]

1.- OBJETO DEL PLIEGO

El presente pliego tiene por objeto la contratación de la prestación de los siguientes servicios para la actualización y mejora del diseño del portal miferia.com y la creación de su aplicación móvil híbrida.

- **Diseño UX/UI:** creación de nuevos diseños para el portal web, tanto para versión de escritorio como para la de dispositivos móviles.
- **Página web:** maquetación de todo el portal web adaptándolo al nuevo diseño responsive.
- **Aplicación móvil:** creación de una aplicación móvil híbrida con compatibilidad mínima con **Andorid** e **IOS**. La **app** es un *launcher* que abrirá el portal web.

2.- DESCRIPCIÓN DE LOS TRABAJOS

La prestación de servicios consistirá en la ejecución de los trabajos descritos a continuación:

2.1.- PÁGINA WEB

2.1.1.- INTRODUCCIÓN

A. Descripción funcional

La web de “Mi feria” cuenta con una estructura sencilla compuesta por un menú de 7 secciones, cada una de ellas con varios sub-apartados. Se trata de un espacio donde se publica información actualizada sobre la feria y donde se incluyen enlaces a los diferentes perfiles y canales en redes sociales del evento. La web está disponible en español e inglés.

B. Descripción tecnológica

El front está desarrollado en Html 5 + CSS3 + AJAX. El proyecto web no está maquetado conforme a criterios 'responsive' y no permite que los contenidos se adapten al espacio en función del dispositivo utilizado: ordenador portátil, tableta o teléfono móvil. Actualmente la web solo está optimizada para navegadores chrome. La parte front es totalmente independiente del back, ya que la actualización y administración de las página se hacen mediante llamadas a una API REST desarrollada en Java. A través de WebView y Cordova se crearán las aplicaciones móviles híbridas.

[33] Pliego modificado y basado en una oferta de licitación pública (Junta de Andalucía, 2017)
[34] Portal ficticio

2.1.2.- TAREAS A REALIZAR

Se plantea la necesidad de trabajar en una renovación gráfica del portal web para la VI edición de la feria. Esta renovación pasa por un nuevo diseño con mejores funcionalidades UX/UI acordes a los nuevos estándares. Además se necesita que la web tenga una maquetación totalmente responsive y que se cree la aplicación móvil que al ejecutarla ejecute un Webview de la web. Para afrontar las necesidades descritas, se plantean 2 áreas de trabajos:

A. **Renovación visual de la web:** Se solicita una renovación y mejora visual del portal web, que permitan al usuario reconocer con claridad que nos encontramos ante una nueva edición de la feria. Dicho restyling incluirá principalmente cambios en las tipografía utilizadas y en las imágenes estructurales de la página web.

B. **Aplicación móvil híbrida:** Se solicita la creación de una aplicación híbrida del portal para publicarla tanto en Android como en IOS. Se usará Webview y Cordova.

3.- RECURSOS HUMANOS

La empresa licitadora deberá contar como mínimo en su estructura con un equipo técnico compuesto por profesionales de los siguientes perfiles, presentando al efecto declaración responsable conforme al Pliego de Cláusulas Administrativas Particulares:

Nº	Perfil	Experiencia
1	Coordinador/a de los trabajos.	3
1	Diseñador/a UX/UI	3
1	Maquetador/a con experiencia en desarrollos responsive	3
1	Desarrollador/a con experiencia en proyectos de aplicaciones híbridas y en desarrollo frontend con Angular.	3

La empresa adjudicadora proporcionará un empleado que estará involucrado durante el desarrollo de todo el proyecto para resolver las dudas y verificar el desarrollo correcto del producto. También exige, que salvo casos de fuerza mayor y justificables, los profesionales implicados en el proyecto no cambien durante su desarrollo y tengan una asignación del 100% al mismo.

La empresa adjudicataria, deberá ofrecer garantías de disponibilidad del personal que conforme el equipo destinado exclusivamente a la realización de las prestaciones, dando respuesta inmediata a los trabajos y servicios requeridos, además de tener en cuenta la integración de un empleado a su proyecto que aprobará el o no el diseño propuesto por el adjudicatario.

4.- DURACIÓN DE LOS TRABAJOS

El plazo de ejecución se extenderá desde la firma del contrato hasta el 30 de Junio del 2018.

5.- CONDICIONES GENERALES DE REALIZACIÓN

5.1.- ACCESIBILIDAD

Teniendo en cuenta la gran diversidad de usuarios finales que pueden hacer uso del Portal, el acceso y uso de las funcionalidades del mismo deberán ser lo más sencillo y accesible posible.

- **Cliente:** Acceso al portal a través de cualquier navegador del mercado, con el que puedan contar los posibles dispositivos y sistemas operativos de los clientes, ya sea de licencia o de fuente abierta y a través de la descarga de la aplicación desde Andorid o IOS.
- **Facilidad de uso y navegación**.

5.2.- SEGURIDAD Y CONFIDENCIALIDAD DE LA INFORMACIÓN

El adjudicador facilitará a la empresa adjudicataria la información de que disponga relacionada con el objeto del presente contrato. El adjudicatario queda expresamente obligado a mantener absoluta confidencialidad y reserva sobre cualquier dato que pudiera conocer con ocasión del cumplimiento del contrato, especialmente los de carácter personal, que no podrá copiar o utilizar con fin distinto al que figura en este pliego, ni tampoco ceder a otros ni siquiera a efectos de conservación, sin el consentimiento expreso, por escrito, del adjudicatario.

5.3.- TRATAMIENTO DE DATOS DE CARÁCTER PERSONAL

En cualquier caso, el adjudicatario queda obligado al cumplimiento, en su totalidad, de la Ley Orgánica 15/99 de 13 de Diciembre de Protección de Datos de Carácter Personal y el Real Decreto 994/1999, de 11 de junio, que establece el Reglamento de Medidas de Seguridad de los ficheros automatizados que contengan datos de carácter personal.

6.- OTRAS CONSIDERACIONES

La empresa adjudicataria deberá llevar a cabo reuniones periódicas de coordinación con el responsable actual del portal para el seguimiento y planificación de los trabajos solicitados. Estas reuniones tendrán lugar semanalmente en los Servicios Centrales de la organización.