UNIVERSITAT POLITÈCNICA DE VALÈNCIA

etsinf
Escola Tècnica Superior d'Enginyeria Informàtica

Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Integración de tableros Kanban en una herramienta que apoya la gestión ágil del trabajo

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Pablo Jiménez Izquierdo

Tutor: Patricio Letelier Torres

Curso 2018-2019

Dedicatoria

Primero de todo, me gustaría dedicar este trabajo a mi familia, por brindarme la oportunidad de estudiar este maravilloso grado y apoyarme siempre que lo he necesitado.

También quiero acordarme de todos los profesores que he tenido durante el grado. Sin ellos hoy no estaría aquí.

Por último, también quiero dedicárselo a todos mis compañeros que he ido conociendo a lo largo de estos cuatro maravillosos años, sin ellos no habría sido igual.

Agradecimientos

En primer lugar, me gustaría agradecer a mi tutor Patricio Letelier Torres por su ayuda y dedicación durante todo el proyecto, solucionando cualquier dificultad que me ha ido surgiendo a lo largo de todo este tiempo.

También quiero agradecer a Marc Pérez – Serrano por brindarme ayuda a la hora de la integración en Worki.

Por último, pero no por ello menos importante, agradecer a todas las personas que han estado día a día apoyándome durante la realización del trabajo

Gracias.

Resumen

Los métodos ágiles han cambiado la forma de trabajar de las empresas software. Entre las más populares destacan Scrum, Extremme Programming o Kanban. De estos tres métodos, Kanban es uno de los que más ha crecido en los últimos años y en el que nos centraremos para llevar a cabo este trabajo. Los tableros kanban son la técnica esencial del método Kanban, el tablero kanban es lo mejor para gestionar visualmente todo el flujo de trabajo. El objetivo de este TFG es desarrollar un tablero kanban para una herramienta de gestión ágil del trabajo: Worki.

Palabras clave: tableros kanban, métodos ágiles, flujos de trabajo.

Abstract

Nowadays, agile methods have changed the way that software companies work. Among the most popular agile methodologies stand out Scrum, Extremme Programming or Kanban. About these three methods, kanban is one of the most that has grown in the last years and in which we are going to focus this project. Kanban boards are the essential way of the Kanban method, kanban boards are one of the best ways to manage visually all the workflow. The main goal is to develop a kanban board for an agile work management tool: Worki.

Keywords: kanban boards, agile methods, workflows.

Tabla de contenido

Índice de figuras

Índice de tablas

1. Introducción

La utilización de tableros kanban ofrece una forma sencilla para representar el estado del trabajo. En los últimos años esta técnica que comenzó a utilizarse en desarrollo de software se ha ido extendiendo a otros contextos, llegando actualmente a ser una de las técnicas más populares para gestión visual del trabajo. El uso de tableros kanban está ligado a las metodologías ágiles, y en particular se asocia con el método Kanban, para el cual esta técnica es fundamental.

El método Kanban [37] está enfocado al trabajo de equipo, este ha ido obteniendo más importancia a lo largo del tiempo ya que promueve un buen flujo de trabajo. Para el método Kanban la visualización del trabajo es algo realmente importante, por eso aparecen en escena los tableros en este método, los cuáles aportan a los equipos la visualización de que se está trabajando actualmente. Además, sugiere la limitación del trabajo en progreso y así conseguir prevenir el exceso de compromiso en un número de tareas que el equipo será capaz de terminar.

El uso de tableros kanban estimula el buen flujo de trabajo terminado, favoreciendo la colaboración entre miembros de un equipo y ofreciendo una forma muy sencilla y efectiva de seguimiento del estado del trabajo. Actualmente su popularización está aumentando, según el artículo *State of Agile* [36] que se hace cada año, el uso de tableros kanban ha crecido de un 50% a un 60%. Además de ser la herramienta más utilizada el año pasado como herramienta de gestión de proyectos. Un tablero kanban esencialmente es un workflow (las columnas corresponden a actividades) y los ítems que contiene representan unidades de trabajo encargadas a un equipo. En la Figura 1 podemos observar el diseño de un tablero tradicional.

Figura 1. Tablero Kanban con post-its [29].

Existen muchas herramientas que ofrecen soporte para tableros kanban. La gran mayoría ofrece funcionalidades que simulan un tablero físico usando post-it. Sin embargo, hay muchas situaciones en las cuáles la gestión adecuada de un tablero kanban es más exigente, tales como:

- El equipo realiza diversos tipos de trabajo con distintos workflows (distintos diseños de tableros kanban).
- Cuando el volumen de ítems no terminados es muy grande (cientos de ítems).
- Cuando el trabajo es muy dinámico en cuanto existencia de trabajos en paralelo para un mismo ítem, o suele haber re-trabajo, o son frecuentes los cambios y re-priorización del trabajo.

Worki[1] es una herramienta para la gestión ágil del trabajo que ha sido desarrollada en el ámbito académico de la escuela de informática de la UPV. Worki utiliza una variante de tablero kanban para poder abordar estas debilidades presentes en otras herramientas. Sin embargo, Worki no cuenta con una visualización estándar de columnas y fichas (utiliza tablas), lo cual le resta atractivo visual.

1.1 Objetivo

El objetivo de este TFG es integrar en Worki una visualización del trabajo mediante un tablero kanban estándar, manteniendo todas las funcionalidades ya ofrecidas por Worki y que contribuyen a una gestión más completa y precisa del trabajo. El front-end de Worki está desarrollado en Angular, por lo cual se evaluarán componentes que faciliten la implementación de un tablero kanban en esta tecnología. Con el componente que se seleccione se realizará una integración en la interfaz de Worki conectándolo con sus funcionalidades y se validará la efectividad de la integración realizando pruebas con casos reales de uso de Worki. La idea es desarrollar un tablero parecido al que podemos ver en la Figura 2.

[1] Worki https://www.tuneupprocess.com/

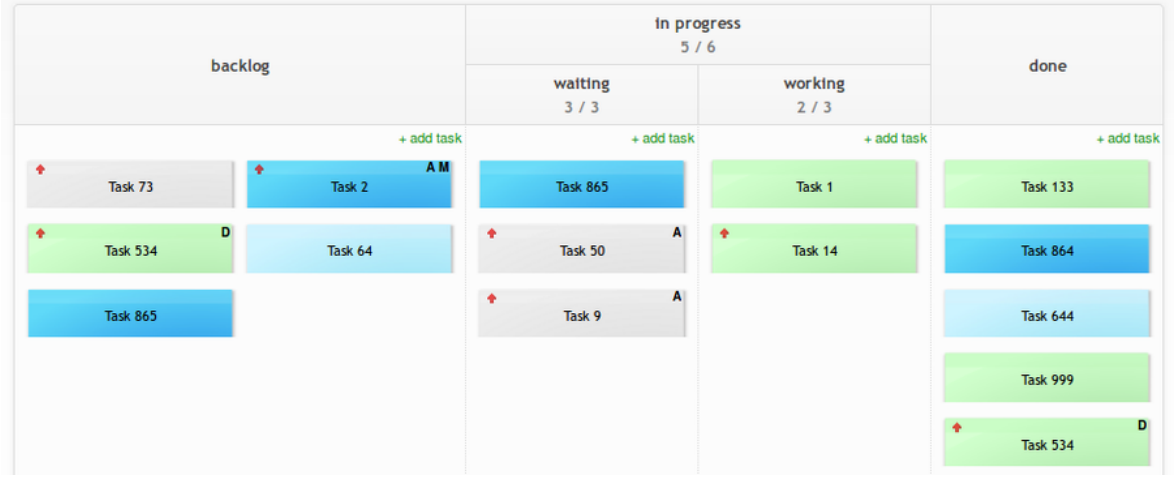

Figura 2. Tablero Kanban básico [30].

1.2 Estructura del trabajo

Tras esta breve introducción, vamos a observar como se encuentra estructurado este trabajo, explicando cada uno de los capítulos.

En el capítulo 2 repasaremos que son las metodologías ágiles y el porqué de su aparición. Seguiremos con la explicación al detalle del método Kanban y se hará un breve repaso a qué es la gestión visual del trabajo y su importancia durante el tiempo.

En el capítulo 3 se explicará como diseñar y utilizar tableros kanban para sacarles el mayor partido posible.

En el capítulo 4 investigaremos diferentes herramientas software externas que aportan tableros kanban para apoyar a los equipos de desarrollo. Estudiaremos qué ofrece cada una de ellas para acabar realizando una tabla comparativa entre ellas.

En el capítulo 5 nos centraremos en la herramienta Worki, explicando qué es y para qué sirve. Explicaremos todas y cada una de sus funcionalidades, como la vista tabular con la que cuenta para mostrar la información en un grid.

En el capítulo 6 hablaremos de las tecnologías que se han utilizado para desarrollar el tablero kanban. Repasaremos brevemente para qué sirve cada una de ellas y se aportarán varios ejemplos para entenderlas a la perfección.

En el capítulo 7 realizaremos una investigación similar a la del capítulo 4. En este caso investigaremos los diferentes componentes que se han encontrado para realizar el desarrollo del tablero kanban para Worki.

En el capítulo 8 se explicará al detalle todo el diseño y desarrollo realizado que ha sido necesario para integrar en Worki el tablero kanban con todas las funcionalidades.

Finalmente, en el capítulo 9 se encuentran las conclusiones obtenidas tras la realización de este trabajo.

2. Métodos ágiles y visualización del trabajo

En este capitulo repasaremos qué son las metodologías ágiles y porqué surgieron. A continuación, explicaremos detalladamente el método Kanban, además de realizar un breve repaso a qué es la gestión visual del trabajo y su importancia durante el tiempo. Por último, estudiaremos los scrumboards o tableros para Scrum, ya que son una diferente versión a los tableros kanban.

2.1 Metodologías tradicionales vs Metodologías ágiles

Las metodologías de desarrollo de software son cruciales para determinar el éxito o fracaso de un proyecto. Después de todo, las metodologías ponen en práctica un conjunto de procesos, considerados como buenas prácticas para lograr todos los objetivos necesarios tanto de negocio, funcionalidad, costes, etc. Si se hace una mala elección de la metodología o un mal uso de ésta puede conducir a que un proyecto fracase [8].

Para hablar de las metodologías ágiles primero las compararemos con las metodologías de software tradicionales. En cuanto a éstas, han resultado ser efectivas, sobre todo, en aquellos contextos donde hay poca incertidumbre, y en proyectos de gran tamaño. Su proceso de desarrollo lleva asociado una predisposición hacia el control total del proceso a través de una estricta definición de actividades, artefactos y roles. Los procesos suelen ser secuenciales y están basados en mucha documentación, lo cual hace que sean poco flexibles a cualquier cambio [16].

La metodología tradicional más conocida es RUP (*Rational Unified Process*), la cual promueve una documentación exhaustiva de todo el proyecto, además de seguir con un plan que fue definido en la fase inicial del desarrollo. Una de las características más relevantes de las metodologías tradicionales es que si tratamos de realizar algún cambio, esto conllevará un alto coste, por lo tanto, no ofrecerá soluciones para aquellos proyectos cuyo contexto está más propenso a cambiar. La mayoría de las metodologías tradicionales se centran en la documentación, planificación de procesos a través de plantillas, revisiones o técnicas de administración [8].

Como reacción a las metodologías tradicionales, surgieron las denominadas metodologías ágiles, que intentan reducir el fracaso por subestimación de costos, tiempos y funcionalidades en cualquier proyecto de desarrollo software [16].

Hoy en día, donde en la mayoría de los proyectos los requisitos cambian habitualmente, estas metodologías se adaptan mejor por asumir dichos contextos de incertidumbre. Existen dos diferencias principales y fundamentales, la primera es que las metodologías ágiles son adaptativas. La segunda es que todas las metodologías ágiles están más orientadas a las personas que a los procesos [9].

La Tabla 1 contiene las principales diferencias entre ambos tipos de metodologías. Estas diferencias no afectan solamente al proceso, si no también a todo aquello que rodea al equipo y su organización [9].

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas proveniente de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Preparadas para cualquier tipo de cambio durante el proyecto	Resistencia a los cambios
Impuestas por el equipo (internamente)	Impuestas externamente
Proceso poco controlado	Proceso controlado, con muchas normas
Inexistencia de contrato tradicional o al menos bastante flexible	Existencia de un contrato prefijado
El cliente es parte del equipo de desarrollo	Reuniones donde el cliente interactúa con el equipo
Equipos pequeños y trabajando en el mismo lugar	Grupos grandes
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Poco énfasis en la arquitectura del software	Arquitectura esencial y expresada mediante modelos

Tabla 1. Comparación entre metodologías ágiles y tradicionales [9].

En la Tabla 1 se puede observar que las metodologías tradicionales hacen uso de artefactos y roles para el tener el proceso controlado, además de tener una gran documentación de toda la arquitectura del software utilizando para ello modelos. En cambio, las metodologías

ágiles ignoran las definiciones de artefactos y roles para centrarse en el cliente, el cual forma parte del equipo. También es posible observar que difieren en el tamaño de estos equipos, mientras que en las metodologías agiles los equipos son pequeños, en las tradicionales la mayoría de los equipos son grupos grandes.

¿Pero cuándo surgen todos estos nuevos métodos? En febrero de 2001, después de una reunión que se celebró en Utah-EEU, es cuando se crea el término "ágil" aplicado al desarrollo de software. En total participan un grupo de 17 expertos de la industria del software. De esta reunión surgió una organización que se conoce como *The Agile Alliance*[2], cuya labor es impulsar todos aquellos conceptos que estén relacionados con el desarrollo ágil de software y ayudar en todo lo posible a que las entidades adopten estos mismos [9].

Toda la filosofía ágil se encuentra resumida en el documento conocido como Manifiesto Ágil [9]. Según el Manifiesto hay que seguir los siguientes valores que podemos observar en la Figura 3:

Figura 3. Valores del manifiesto ágil [1]

- El individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas.
- Desarrollar software que funcione más que conseguir una buena documentación.
- La colaboración con el cliente más que la negociación de un contrato.
- Responder a los cambios antes que seguir estrictamente un plan.

[2] Agile Alliance https://www.agilealliance.org/

Los valores anteriores son los que inspiran a los doce principios, que se recogen en el Manifiesto y que hacen que se diferencie un proceso ágil de uno tradicional. Los doce principios están resumidos en la Figura 4 que observamos a continuación [9]:

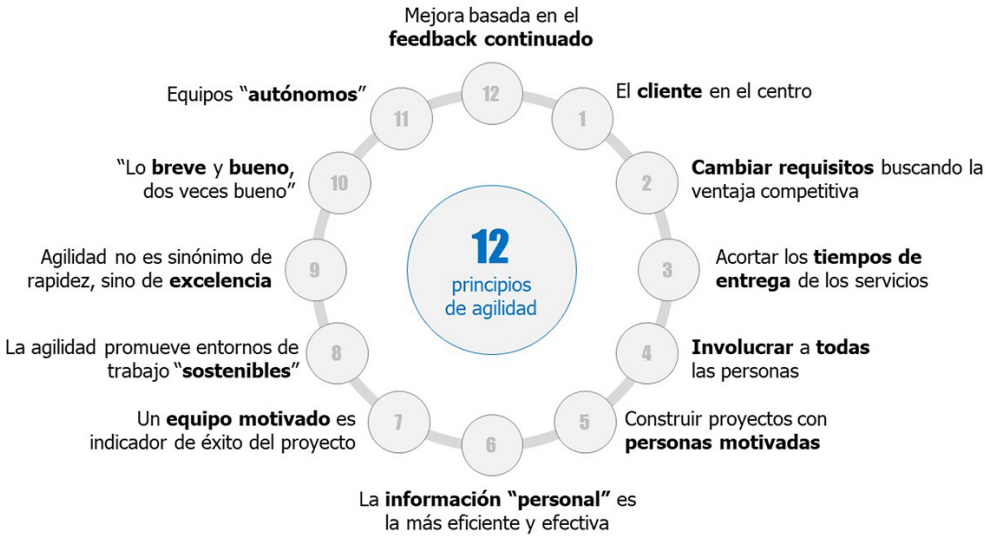

Figura 4. Doce Principios del Manifiesto Ágil [5].

2.2 Método Kanban

Dentro de todo el grupo de métodos ágiles (*Scrum*, *Extremme Programming*, *Agile Inception*, etc.) vamos a centrarnos en el método Kanban. Kanban es uno de esos métodos que nos facilita obtener resultados exitosos en la mejora de los procesos sin tener que realizar grandes inversiones de dinero. Para definir qué es el método Kanban existen múltiples definiciones que vamos a enunciar a continuación:

Raymond [20] define Kanban como "*un dispositivo de señalización desarrollado por Toyota para el movimiento de partes en un sistema de producción por demanda, generalmente mediante el uso de una tarjeta física. El objetivo de Kanban es minimizar el WIP (Work in progress), o el stock entre los procesos. Para lograr esto, Kanban se asegura que el proceso superior produzca partes, sólo si el proceso inferior las necesita; Por demanda, se entiende que los trabajadores del proceso inferior consumen las partes que necesitan de los procesos superiores*".

Para Acevedo et al. [2] Kanban es "*una técnica de gestión de producción basada en un sistema pull (halar) que se fundamentan en la autogestión de los procesos, eliminando la programación centralizada. Se produce y se transporta lo que se demanda en los procesos consumidores, manteniendo en rotación solo aquellas cantidades que garantiza la continuidad del consumo. Cuando se interrumpe el consumo se detiene la producción. Es una herramienta para conseguir la producción Justo a tiempo -JIT-".*

En su traducción literal, la palabra Kanban proviene de las palabras kan (看 o カン) que significa visual y ban (板 o バン) que significa tarjeta o tablero [21]. El método Kanban forma parte de la filosofía de gestión de operaciones JIT (*just in time*), es decir, un sistema que tiende a producir todo lo requerido, en el momento que sea necesario, con la calidad específica y sin desaprovechar ningún recurso del sistema y satisfaciendo la demanda en el tiempo requerido. Pero para implementar este tipo de filosofía es necesario que se aplique de un sistema de control de producción del tipo Pull, donde la producción de ítems es acorde al ritmo que necesitan estos [17].

El método Kanban es conocido mayoritariamente por emplear un tablero para que el equipo de desarrollo pueda visualizar todo el flujo de trabajo, y así conseguir evitar cualquier cuello de botella o saturación que pueda impedir la entrega del proyecto. Este tablero estará compuesto por fichas, cada una de ellas representará un ítem de trabajo, y por columnas, que serán las actividades por las que deben de pasar todos los ítems de trabajo. La identificación rápida de las fichas es algo vital, los miembros del equipo deben de perder el menor tiempo posible para así emplearlo en tareas que ayuden a la finalización de la entrega [31].

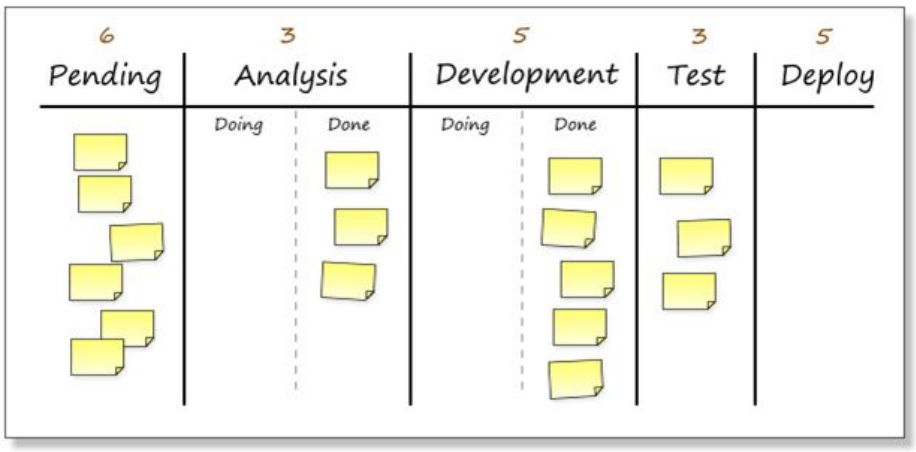

Figura 5. Tablero Kanban con múltiples actividades [27].

Como vemos en el tablero de la Figura 5, las columnas del tablero serán todas las actividades por las que deben pasar los ítems de trabajo. Para considerar que un ítem de trabajo ha terminado tendrá que pasar por cada una de las actividades, en este caso, desde "Pending" hasta la actividad "Deploy". Todos los miembros del equipo de desarrollo deben tener un flujo de trabajo, encolando aquellas tareas más prioritarias y así no interrumpir a alguien para saber que hacer en cada momento. A lo largo del desarrollo, las fichas deben de ir fluyendo de izquierda a derecha hasta terminar cada una de ellas, es decir, se trata de un workflow secuencial.

Una de las principales ideas del Kanban es el uso de la limitación del "Trabajo en Progreso" o más conocido como "Work in Progress" (WIP). ¿Qué significa esto? El número de tareas que es posible realizar en cada actividad del proyecto debe ser conocido por todos los miembros del equipo. Si el limite del WIP no se respeta, el trabajo se estancará y por lo tanto no se podrá continuar. No importa si el proyecto es complejo, simple o pequeño, siempre existe una cantidad óptima que es posible realizar sin perder en eficiencia. [18]

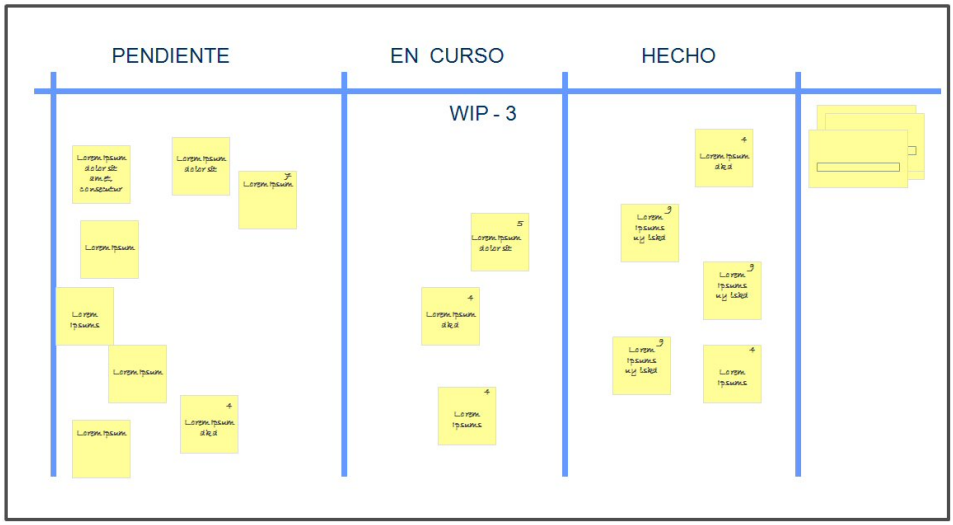

Figura 6. Tablero Kanban estableciendo WIP [28].

Para entender el concepto anterior vamos a ver el ejemplo del tablero de la Figura 6. Como podemos ver, hay establecido un WIP en tres sobre la columna "En curso". ¿Esto qué quiere decir? No podrán haber más de tres ítems que estén en dicha actividad. Los miembros del equipo de desarrollo han considerado necesario establecer este límite en esa actividad ya que consideran que su límite de trabajo es como máximo tres, si lo sobrepasan, podría llegar a crearse un cuello de botella y se irían acumulando muchos ítems de trabajo en la misma actividad sin poder acabar ninguno, o simplemente, el equipo no trabajaría lo más rápido que puede llegar a ser.

Hay una serie de principios que se fomentan en el método Kanban [4]:

- **Calidad**: Todo lo que se hace hay que hacerlo bien.
- **Minimización**: Hacer lo justo y necesario sin centrarse en tareas secundarias o innecesarias que puedan distraer.
- **Mejora continua**: Ir mejorando al paso del tiempo los desarrollos, teniendo en cuenta los objetivos a lograr.
- **Flexibilidad**: Las tareas entrantes han de ser priorizadas según las necesidades del momento.
- **Construcción y mantenimiento de una relación a largo plazo con proveedores.**

Existen varios beneficios asociados a la utilización de Kanban como método de gestión, según [14] los tres siguientes representan un resumen importante:

1. *Estímulo del rendimiento:* Es más fácil detectar cualquier problema existente y ajustar el flujo de trabajo al necesario para ganar en eficiencia. Kanban es un método muy flexible por lo que es posible perfeccionar los procesos para obtener los mejores resultados posibles.

2. *Organización y colaboración:* Permite obtener beneficios del enfoque visual, mediante el uso de columnas y tarjetas. Es posible trabajar a la vez con el tablero y colaborar con los compañeros.

3. *Distribución del trabajo:* Cómoda visión general de los trabajos en curso y menos tiempo dedicado a la distribución. Al ser imperfectas las estimaciones, obtener un flujo constante de tareas reducirá su tiempo de espera y el tiempo dedicado a la asignación de tareas.

Entre los objetivos que tiene el método Kanban, uno de ellos es conseguir desarrollar un producto de calidad, ya que se obliga a que cada fase del proyecto finalice su tarea como es debido. Aparte de contribuir a acabar con los cuellos de botella que puedan darse durante cualquier fase del proyecto. Para conseguir estos propósitos, se han definido cuatro reglas [14]:

1. *Empieza con lo que haces ahora:* Kanban no es un sistema que te explica como debes hacer tu trabajo, es un método de producción. Ayuda a decidir si lo que se está haciendo está haciéndose bien o hay que modificar algo.

2. *Acepta el cambio:* Todos los miembros del equipo que utilicen el método Kanban han de estar dispuestos a realizar cambios que mejoren las rutinas de trabajo.

3. *Respeta el proceso en curso, los roles y responsabilidades de cada uno:* En un proyecto de desarrollo de software algo necesario es que cada miembro del equipo tenga claro qué es lo que debe hacer y cuáles sus funciones dentro del equipo.

4. *Liderazgo en todos los niveles:* Gestionar de forma correcta las tareas del equipo es otro elemento básico. Cada miembro debe de tener claro su función y que la ejecute adecuadamente.

Pero si queremos que el enfoque Kanban funcione, deben de estar presentes los siguientes elementos [14]:

1. *Visualizar el flujo de trabajo:* Kanban recomienda el uso de un panel con tarjetas, donde se definen todas las tareas y indica en que fase del proyecto se encuentra. Esto es conocido como el tablero Kanban.

2. *Limitar el trabajo en curso:* Hacer mucho trabajo pero dejarlo a medias no sirve de nada. Si algo se empieza hay que acabarlo, este es un principio básico del método Kanban.

3. *Gestión del flujo:* Aparte de visualizar todo el flujo de trabajo, es necesario también controlarlo para ver si funciona como debe ser o si existe algún problema solucionarlo.

4. *Dejar claras las reglas del proceso:* Para aplicar bien un método primero hay que entender como funciona.

5. *Mejora en equipo:* Continua mejora constante acordada en equipo.

En el desarrollo de este método, es bastante común ver tarjetas de tareas que se conocen como “Tareas Kanban”, que pueden estar pegadas o no en un tablero o en una pared. Esta tarjeta

contiene información de la tarea que le corresponde, como puede ser un identificador, un nombre, tiempo estimado, el nombre del agente que tiene asignada la tarea, etc. Además, es posible la utilización de un sistema Kanban en cascada tradicional, pero con flujo, donde podemos encontrarnos varios procesos por donde las tareas van moviéndose como "diseño", "programación", "pruebas" [4].

2.3 Gestión visual del trabajo

Para tener una idea del trabajo que estamos realizando de forma clara y concisa es importante hacer uso de una gestión visual. Actualmente, en la mayoría de las empresas de software, hay muchos proyectos que están en marcha a la vez, cada cual con su equipo o con varios para sacarlo en adelante. No hay dos proyectos iguales, cada uno es un mundo distinto, pero lo único que tienen en común es la labor de gestión. Gracias a la gestión visual va a ser posible que el equipo pueda ver todas las tareas que debe realizar, de una forma sencilla para que sea posible centrarse en la gestión de estas mismas [26].

Para explicar este concepto podemos utilizar diversas analogías que dejarán claro qué es realmente la gestión visual. Cuando vemos a los policías de trafico sabemos que son ellos por sus uniformes distintivos, sus gestos de presencia, sus coches, sus direcciones y signos de tráfico. Las líneas de tráfico están pintadas de tal forma que estén separadas y así hacen que los conductores no pasen sus líneas. Las líneas de alerta que están al lado de los arcenes emiten un ruido para avisar al conductor que puede estar en peligro. Esto es lo que trata de hacer la gestión visual, pero en un contexto muy diferente [24].

Históricamente, la gestión visual y por correspondencia, la visualización de datos ha estado muy presente en el mundo. Dos mil quinientos años A.C, la unidad de longitud de los egipcios (*Cubit*) era utilizada para proyectos de construcción y en otras áreas relacionadas como estándar de medida visual. Seiscientos años A.C, el general chino Sun Tzu utilizó banderas, gongs y señales de fuego para la comunicación y dirección de su ejercito. En 1917 se creó el diagrama de Gantt para el control visual de la producción en Frankford Arsenal. En 1920 Charles Edward Knoeppel estableció las relaciones entre la eficiencia industrial y el control de los artefactos atreves de métodos gráficos. Alrededor de 1935 el pensamiento just-in-time (JIT), del cual la gestión visual está constituida a grandes porciones, fue personificado en el manual 10-centimerr-thick por el fundador de la empresa Toyota. Y así sucesivamente hasta la actualidad, donde la

gestión visual va aumentando cada vez y se hace más importante para los procesos de desarrollo [24].

Además, el concepto de gestión visual ha generado una serie de mejoras a los procesos donde se ha ido incorporando [24]. Esto se recoge en la Tabla 2 donde se muestra las diferentes funciones que realiza la gestión visual:

Función	Definición
Transparencia	La habilidad de un proceso de producción de comunicarse con gente.
Disciplina	Hacer un hábito de utilizar los procedimientos correctamente.
Mejora continua	Innovar y mejorar incrementalmente.
Facilitación del trabajo	Facilitar los esfuerzos de la gente en tareas ya conocidas.
Entrenamiento en el trabajo	Aprender de la experiencia o integrar trabajo con aprendizaje.
Gestión de datos	Utilizar los datos en estadísticas.
Simplificación	Realizar esfuerzos en monitorizar, procesar, visualizar y distribuir el sistema para equipos y personas.
Unificación	Eliminar parcialmente los 4 límites principales y crear empatía con la organización a través de compartir información efectiva.

Tabla 2. Funciones de la gestión visual [24].

Kanban es una metodología que hace un principal uso de técnicas visuales para ver en que estado se encuentra cada tarea, representando cada uno de ella en pizarras llenas de post-it, tarjetas, etc. El tablero tiene el mismo número de columnas como de actividades que se aplican a todos los ítems de trabajo [18].

Gracias a esta visualización el equipo conoce fácilmente el estado del trabajo, ver en qué está trabajando cada persona, ver quién tiene algo que realizar y la prioridad de los ítems de trabajo [18].

2.4 Scrumboards

Scrum es otro método ágil que ayuda a trabajar más eficazmente. Obviamente, no te dice todo lo que tienes que llevar a cabo, pero si te proporciona algunas restricciones y directrices. Por ejemplo, mientras que Kanban te obliga a hacer uso de un tablero visual y limitar el WIP, Scrum propone tener iteraciones de duración fija. Scrum es más restrictivo que Kanban [13].

En Scrum también se considera como una buena práctica hacer uso de un tablero para gestionar su *Sprint Backlog*, el denominado scrumboard, donde se encontrarán aquellas tareas que el equipo de desarrollo se ha comprometido a realizar durante la duración de un sprint, por lo que Kanban no es el único método que utiliza un tablero para gestionar el trabajo visualmente. Mientras que en el método Kanban se limita el WIP por estado en flujo de trabajo, Scrum tiene en cuenta cual es la capacidad del equipo cuando se negocia qué se va a llevar a cabo durante un Sprint [13].

Frecuentemente, un scrumboard tiene este aspecto a lo largo de sus diversas etapas de un sprint, que podemos observar en la Figura 7:

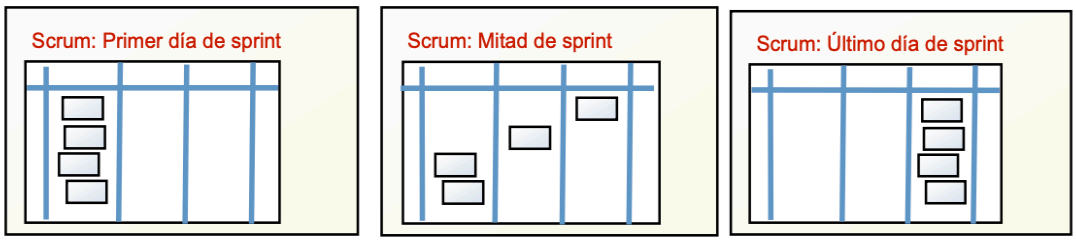

Figura 7. Ejemplo de scrumboard [13]

El scrumboard suele dividirse en columnas etiquetadas como *No Empezado, En desarrollo, Terminado* por donde las tareas van pasando a lo largo del Sprint [25].

El equipo de desarrollo inspecciona el scrumboard diariamente para obtener información de forma rápida y sencilla del estado de las tareas. Un scrumboard pertenece solamente a un

equipo Scrum. Normalmente, el equipo es multi-funcional, es decir, contiene todos los conocimientos para llevar a cabo todos los elementos de la iteración. Además, el equipo es el único que puede realizar modificaciones sobre el mismo [13].

Una de las situaciones más comunes es la aparición de un ítem de trabajo cuando ya ha empezado el Sprint y alguien quiere añadirlo. ¿Cómo se solucionaría esto? Aquellos que trabajen con Kanban, lo añadirán en la columna "Pendiente", y cuando estén listos para trabajar con él lo abordarán. En Scrum no sucede lo mismo, el equipo de trabajo no la añadirá al tablero ya que se ha comprometido a realizar los ítems de trabajo que se encuentra en el *Sprint Backlog*, por lo tanto, añadirán el nuevo ítem de trabajo a la pila de producto. Podemos decir que Scrum se resiente a los cambios durante la iteración [13].

La limpieza del tablero tiene lugar una vez finalizado el Sprint. Al iniciarse un nuevo Sprint y una vez finalizada la reunión de planificación del Sprint ya habrá un nuevo scrumboard, con las nuevas tareas en la primera columna. Puede parecer que todo esto pueda significar una pérdida de tiempo, pero para equipos experimentados estas tareas no conllevan demasiado tiempo [13].

Gracias al scrumboard, los miembros del equipo pueden visualizar de forma clara [13]:

- Visión general de todo el proyecto.
- Todas las historias de usuario y cuando deben ser implementadas y estar completas.
- La estructura del Sprint.
- Todas las tareas que deben realizarse durante el Sprint.
- Los ítems de trabajo que se encuentran en progreso.
- Qué se está cumpliendo y qué no.
- Qué mejoras pueden realizarse para mejorar el proceso de desarrollo.
- El resultado final del Sprint.

En conclusión, aunque el scrumboard es básicamente un tablero kanban, solo muestra los ítems pertenecientes al Sprint en el cual se está trabajando. Como veremos más adelante, en el tablero kanban de Worki se mostrarán el estado de todos los ítems, pertenezcan o no a algún Sprint.

3. Diseño y uso de tableros Kanban

En este capítulo vamos a ver cómo tenemos que diseñar y utilizar los tableros kanban para así sacarles el mayor partido posible. Pero antes de esto, haremos un breve repaso a qué son los workflows en el proceso de desarrollo de software.

3.1 Workflows en el proceso de desarrollo de software

El desarrollo de software es un procedimiento donde intervienen roles, actividades y artefactos. Existen diferencias entre el proceso de desarrollo de software y los demás, pero hay algunas que son más significativas: la aparición de muchas situaciones imprevistas donde el equipo debe actuar de forma rápida y eficaz, y la importancia de tener comunicación y colaboración con los demás integrantes del equipo y responsables de otras actividades [32].

Los workflows están muy presentes en el desarrollo de software tanto si se utiliza un enfoque tradicional o ágil. Normalmente, en un proceso iterativo e incremental, cada incremento debe pasar al menos por tres actividades: Definir, Programar, Probar. Si se identifican otras actividades se podrían añadir sin ningún tipo de problema [32].

Siempre que nos encontremos ante la definición de un nuevo workflow vamos a tener que tomar tres decisiones: identificar que actividades van a componer el workflow, qué roles tendrán asociados cada actividad, y, por último, qué integrantes del equipo ejercerán cada rol. Para Scrum no existen roles específicos para el trabajo técnico, si no que se utiliza un rol común para todos como podría ser "*Development Team*". Este planteamiento no es incompatible con la existencia de las actividades que se han mencionado antes, solo que el rol tendrá un papel de agrupador de actividades. Para entender esto vamos a ver tres workflows cuya diferencia radica en los roles que se asocian a cada actividad [32].

En la Figura 8 podemos observar un WF Scrum Básico Ideal donde el encargado de gestionar el *Product Backlog* es el *Product Owner*. Todas las actividades que se encuentre en dicho ámbito serán realizadas por el mismo, mientras que el equipo participará en las actividades de Programación y Pruebas además de ayudar al *Product Owner* en la definición y estimación de los ítems de trabajo [32].

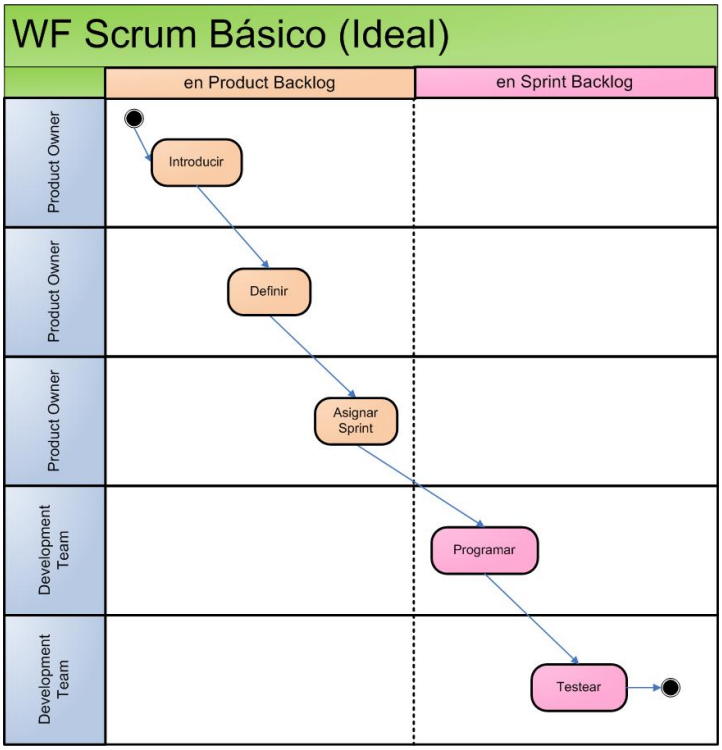

Figura 8. Workflow básico para Scrum [32].

Sin embargo, una situación que sucede a menudo es cuando el equipo debe asumir las actividades de las que se encargaba el *Product Owner* además de las suyas. Aún así, el *Product Owner* debería seguir siendo responsable de dichas actividades [32].

En la Figura 9 se muestra una variante, el WF Scrum Básico "con roles tradicionales" donde un ítem en una actividad es realizado por roles específicos en cada actividad del flujo de trabajo. Esto no implica que los miembros del equipo no puedan realizar otras actividades, si se asignaran los roles de manera permanente a un miembro, esto si que estaría totalmente en contra de las ideas de Scrum [32].

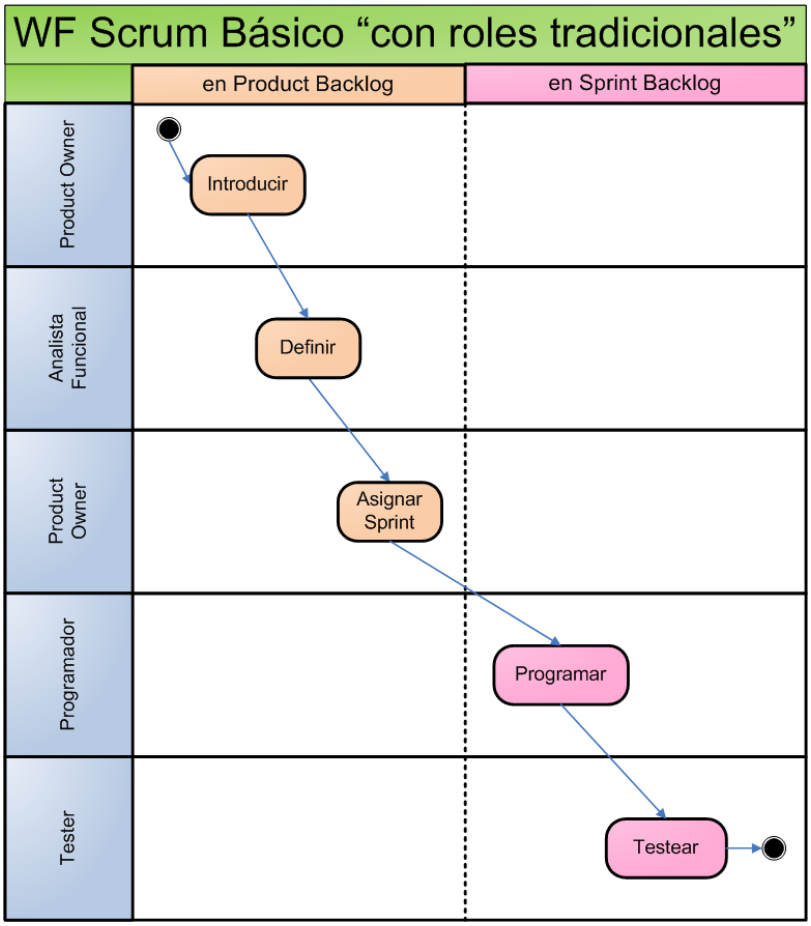

Figura 9. Workflow con roles tradicionales para Scrum [32]

Pero si queremos sacarle el máximo partido a un workflow necesitamos que estos sean flexibles, esta característica ayudará a que se encuentren preparados para cualquier desafío que aparezca en el desarrollo de software. En el siguiente capitulo vamos a ver que características tienen los Workflows Flexibles [32].

3.2 Características requeridas por workflows flexibles

Un Workflow Flexible tendría que ejercer como si se tratase de una guía de referencia respecto a la vida que se espera que tenga un ítem, desde que se crea hasta que se termina. No se debería especificar al máximo detalle todas las actividades posibles, ni representar aquellas posibles situaciones de re-trabajo. En resumen, un workflow debería centrarse nada más y nada menos que en un flujo ideal. Un ítem puede que tenga un tratamiento diferente dependiendo del producto, no siempre es necesario realizar las mismas actividades. Por lo tanto, los workflows representan las necesidades para procesar un ítem en el desarrollo de un producto [33].

Para poder aprovechar todas las características de un Workflow Flexible vamos a necesitar de una herramienta que nos ayude a abordar las situaciones posibles que se presentan durante su gestión. Si se quiere integrar workflows al desarrollo de software es necesario seguir la siguiente funcionalidad básica [33]:

1. Gestión de workflows.
2. Asignación de workflows a un producto.
3. Al crear un ítem asignarle un workflow instantáneamente.
4. Asignar personas a roles del workflow.
5. Al finalizar una actividad en cierto ítem, este debe a pasar al estado pendiente en la siguiente actividad del workflow asignado.
6. Uso de un tablero kanban o scrumboard para la gestión visual de todos los ítems.
7. Registro de seguimiento de las actividades realizadas a un ítem.

Gracias a estas funcionalidades obtendremos workflows integrados en el proceso de desarrollo. Además, para que sean flexibles es necesario hacer uso de ciertos mecanismos, añadiendo las siguientes funcionalidades especificas [33]:

1. Permitir realizar saltos hacia actividades anteriores o siguientes. Puede darse el caso de decidir llevar un ítem a una actividad previa (situación de re-trabajo) o omitir algunas actividades que se consideren innecesarias.
2. Cuando sea necesario realizar re-trabajo sobre un ítem y se considere que es de poca magnitud, sería interesante continuar la actividad actual a la vez que se trabaja sobre el ítem en la actividad anterior.
3. Trabajo en paralelo para una misma actividad donde varios miembros del equipo podrían trabajar juntos colaborando entre ellos.
4. Si se considera que hay un workflow que se adapta mucho mejor a un ítem que el actual, hay que cambiar de workflow el ítem.
5. Modificar el workflow cuando sea necesario.
6. Cambiar a los roles del workflow los agentes que hayan sido asignados.
7. Asignar por defecto el agente a un ítem en cierta actividad siempre y cuando se hayan pre asignado agentes a los roles del workflow. Si por el contrario no se quiere contar con la pre asignación de agentes, al finalizar una actividad pasarla a la siguiente sin ningún miembro del equipo asignado.
8. Añadir actividades que no se encuentren definidas en el workflow.

3.3 Diseño básico de tableros kanban

Un buen diseño del tablero Kanban unido a una aclaración de todas las reglas del juego con las cuáles el equipo de desarrollo interactuará con el tablero, nos proporcionará unos resultados excelentes [34]. La Figura 10 presenta un tablero Kanban con un diseño simple.

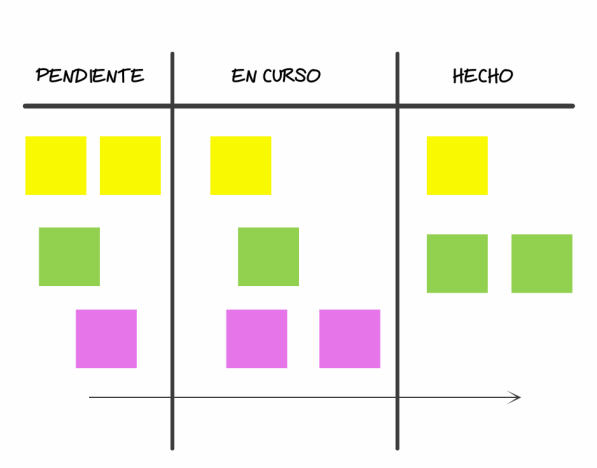

Figura 10. Tablero Kanban [34]

Todas las fichas del tablero van fluyendo de izquierda a derecha, y cada una representa un ítem de trabajo que el equipo de desarrollo debe abordar. Las columnas "Pendiente", "En curso" y "Hecho" son los diferentes estados por lo que la tarea puede pasar. En esencia, todas las fichas deberían de fluir a través de las diferentes columnas del tablero hasta que lleguen a estar finalizadas, y si esto se hace con buen ritmo significará que el trabajo va siendo terminando incrementalmente. En algunos casos es posible la existencia de saltos de columnas hacia adelante o hacia atrás [34].

La mayoría de los ítems en el desarrollo de software normalmente son de tipo: Nuevo Requisito, Mejora o Corrección de Fallo y aunque puedan parecer diferentes todos necesitan del mismo procesamiento con diferente intensidad. Es decir, un ítem de tipo Corrección de Fallo no requerirá de tanta definición como otro de tipo Nuevo Requisito o Mejora. El problema viene al mezclar ítems que no tienen el mismo procesamiento, si no se crea un tablero aparte para estos, el equipo de desarrollo debería tener claro que actividades han de realizarse para unos ítems o cuáles para otros. También hay que considerar que con cuantos menos tableros Kanban trabaje un mismo equipo, mejor gestión integrada se obtendrá y más aún si es el mismo ámbito de trabajo. Una línea de trabajo es un producto en desarrollo o mantenimiento, o un servicio. Sería

conveniente tener tableros por cada línea de trabajo, así se evita tener que decorar de alguna forma las fichas para saber a que línea de trabajo pertenecen [34].

Si queremos añadirle más detalle al tablero, por ejemplo, para poder saber si un ítem está pendiente, en proceso o terminado en una actividad especifica se pueden utilizar sub-columnas To Do, Doing y Done, solapando la columna Done de una actividad y la columna To Do de la siguiente, como se puede observar en la Figura 11 [34]:

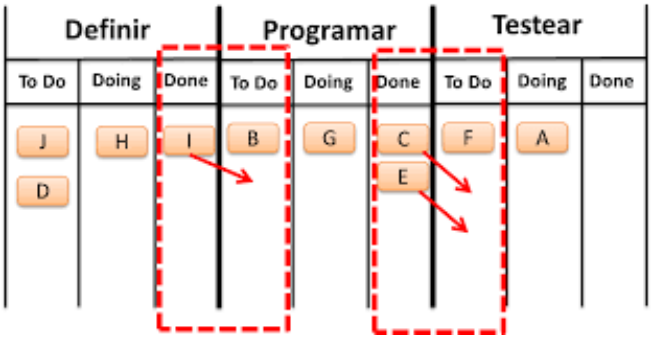

Figura 11. Tablero Kanban solapando varias columnas [35]

Ya que al finalizar cualquier tarea ya se sabe si es conveniente para el ítem pasarlo a la siguiente columna o tener que saltarse ciertas columnas, fusionamos ambas sub-columnas en una única columna. Aún así hay algunas situaciones más que es necesario tener en cuenta [35]:

- Existe un momento donde el equipo de trabajo ya ha identificado que debe realizar un cierto ítem, pero no dispone de toda la información necesaria para su comienzo. Para esto, resulta efectivo añadir una columna "Introducir" así cuando un ítem finaliza esta actividad tenemos asegurado que contiene toda la información necesaria para llevarlo a cabo.
- En el momento en el que se va a evaluar la prioridad de un ítem respecto a los demás que ya la tienen, es conveniente añadir una actividad cuyo nombre sea, por ejemplo, "Ordenar". Esta prioridad no es importante solo para el ítem, si no para el conjunto de todos, pues hay que intentar procesar los ítems de acuerdo con este orden. En esta columna acumularemos todos los ítems que todavía no tengan un orden establecido.

Por lo tanto, la columna "Pendiente" podríamos substituirla por estas dos columnas nuevas.

Si queremos empezar a aplicar tableros Kanban es recomendable hacerlo de una forma minimalista, es decir, menos tableros y que tengan menos actividades. Para seguir lo anteriormente comentado podríamos seguir estas recomendaciones [34]:

- Cuando nos encontremos en situaciones multi-proyecto donde cada línea de trabajo puede que tenga su propio tablero es conveniente utilizar algún software externo ya que nos podemos quedar sin espacio físico donde poner todos los tableros Kanban.
- Cuando una única persona es quien realiza actividades consecutivas e inmediatas puede que la mejor opción sea fusionar estas columnas en una única columna. Pero cuando se realizan distintas tareas dentro de una actividad y no siempre en el mismo orden, una opción sería dividir la columna original en dos columnas.

Si conseguimos que el tablero siempre esté actualizado, refleje el estado en el que se encuentra el trabajo del equipo y que la gente no se olvide de su existencia, podremos asegurar en gran parte que su implantación ha tenido éxito. Para ello, todos los miembros del equipo han de participar y realizar las operaciones oportunas sobre el, como mover un ítem cuando da comienzo o ha finalizado. Para mantener el tablero "vivo" podríamos seguir algunas de estas recomendaciones [34]:

- Realizar reuniones para poner en común el trabajo del equipo.
- Sería interesante añadir una imagen en la ficha de quien está trabajando en determinada tarea.
- Si pasan los días o semanas y las fichas del tablero no se mueven la gente perderá el interés por ver el estado del tablero. Esto puede suceder por estar tratando con tareas que resulten ser Épicas (taras muy "grandes"), sería conveniente dividir dicha tarea y así obtendremos tareas más sencillas de abordar y el flujo del tablero no se verá afectado.
- Un tablero físico tiene limitaciones. Es conveniente utilizar alguna herramienta software que lo reemplace.

3.4 Diseño de tablero kanban usando Sprints

Como se ha comentado anteriormente, Scrum también hace de uso de tableros para visualizar el flujo de los ítems a partir de las actividades, estos tableros son conocidos como scrumboards. Que utilice un tablero kanban no implica que incorpore otros elementos del método

Kanban. Hay que saber diferenciar entre el tablero kanban y el método Kanban (con K, mayúscula) [35].

Aplicar Sprints o iteraciones es una de las prácticas agiles que utiliza Scrum. Estos aparecen cuando el cliente espera un compromiso respecto a plazos, alcance y costos. Dentro de un Sprint se terminan una serie de ítems entre la fecha de inicio y la fecha final. [35].

En todos los tableros kanban existe un grupo de actividades que se encuentran a la izquierda del tablero cuya función es preparar el ítem para que otro grupo de actividades (a la derecha del kanban) lo implementen. Al trabajar con Sprints hay que tener claro que idealmente todos los ítems deben de estar preparados antes de que se introduzcan en el Sprint para asegurarse de que todo el contenido del Sprint puede terminarse dentro del plazo acordado [35].

Para Scrum existen dos contenedores de ítems diferentes: el *Product Backlog* y el *Sprint Backlog*. En el *Product Backlog* se encuentran todos los ítems que no han sido incluidos en el Sprint, mientras que el *Sprint Backlog* contiene todos los ítems que se han de realizar durante el Sprint. Entonces, todas esas actividades que sirven para preparar el ítem se harán mientras se encuentren en el *Backlog* y las demás una vez estén introducidas en el Sprint. Esto puede verse ilustrado en la Figura 12 [35].

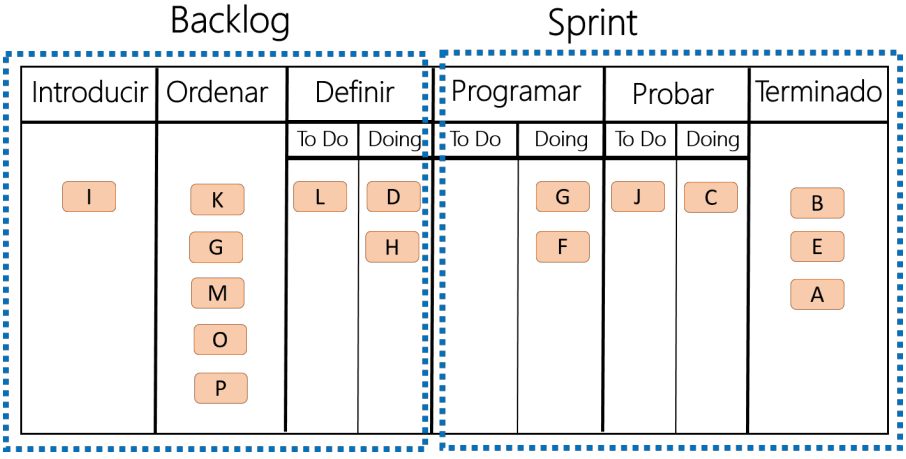

Figura 12. Tablero Kanban dividido en Product Backlog y Sprint Backlog [35]

Otra forma bastante frecuente en las herramientas que dan soporte para tableros kanban es la de eliminar todas las actividades durante el Sprint, asignándoles un conjunto de tareas mínimas que deben darse para considerar que el ítem está terminando: To Do, Doing y Done. Por lo tanto, al finalizar el Sprint todos los ítems deben de estar en la columna Done. Esta forma no

es realmente útil ya que hay que realizar un mayor esfuerzo para gestionar todos los ítems, aparte de tener que gestionar todas las tareas que tienen asociadas dentro del Sprint.

Una práctica que sugiere Scrum es hacer coincidir el final de un Sprint con el comienzo del otro. Para ello hay que llegar al final con bastante trabajo preparado para incluirlo en el Sprint y así iniciarlo al momento. Por lo tanto, cuando se vaya terminando el trabajo del sprint debería empezar a prepararse el siguiente. Esto significar que deberíamos tener bastante trabajo acumulado en la columna Definir y aquellos ítems que estén terminados no deberían pasarse a Programar. Para solucionar esto, es necesario añadir otra columna más que actúe como buffer para aquellos ítems que se han terminado en la columna Definir. En la Figura 13 se ilustra esta nueva columna [35].

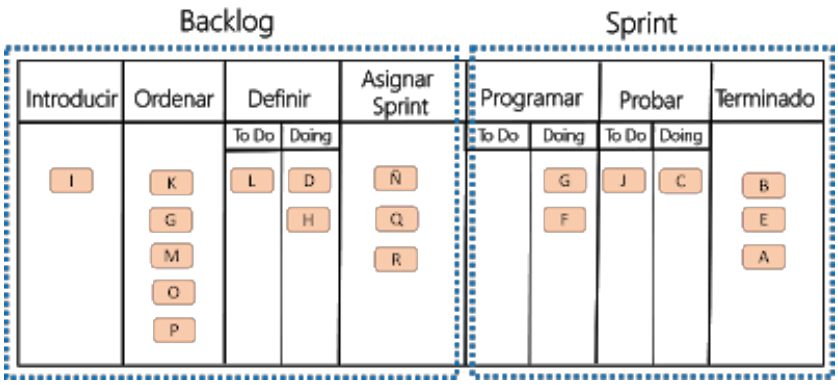

Figura 13. Tablero Kanban añadiendo columna buffer [35].

Existen algunas situaciones que pueden llegar a crear confusión [35]:

- **El Sprint ha finalizado y todos los ítems no han sido terminados.** Una opción es alargar el sprint, aunque Scrum sugiere seguir un mismo ritmo de trabajo y se consigue a través de Sprints que duran exactamente igual. La otra variante y que se considera mejor práctica es devolver aquellos ítems que no se hayan terminado a la columna Asignar Sprint y volviendo a evaluar su prioridad.
- **Existen ítems en el Sprint que no están realmente preparados para dar comienzo.** En este caso valdría con devolver el ítem a las actividades de preparación y una vez terminado pasarlo a las tareas de desarrollo del Sprint ya que se considera que pertenece a él.
- **Detección de fallos en la preparación de un ítem.** Si en algún momento se detectan errores en alguna actividad que se encuentra en el Sprint actual y se está realizando sobre ella alguna actividad y se detecta que es un error "grande" habría que devolver el ítem a su actividad correspondiente marcándolo como re-trabajo. Una vez se ha solucionado el

error habría que hacer una evaluación para ver si es viable devolverla a la actividad desde donde se volvió atrás o a alguna actividad intermedia. Si el error es "pequeño" sería posible mantener el ítem en la actividad actual y marcarlo para que se resuelva de forma prioritaria.

- **Se terminan todos los ítems de un Sprint antes de su finalización.** Lo recomendable sería añadir más ítems al Sprint para mantener la regularidad de los Sprints.
- **Momento en el que se quiere constituir todo el contenido de todos los Sprints hasta el termino del proyecto.** El tablero solo contiene los ítems del Sprint actual, los demás se encuentran en el *Backlog*. Si preasignamos todos los ítems a sus Sprints correspondientes hasta que se haya distribuido todo el contenido estaremos cometiendo un error, ya que puede haber ciertos cambios en ítems, cambios de prioridades, ítems que al final no se realizan y sería muy difícil tener que volver a establecer todo el contenido otra vez. Además, es muy complicando saber en un tablero qué ítems están asignados a un Sprint y cuáles a otros. Si lo que se pretende es gestionar todo el alcance del proyecto bastaría con señalar en el *Backlog* cuáles no están incluidos en el proyecto.
- **Ítems que son épicas.** Si el ítem aún se encuentra lejos de incluirlo en el Sprint no deberíamos de preocuparnos. Por el contrario, si se encuentra cerca ya de incluirlo hay que estudiar como se va a realizar dicha inclusión. Si es posible descomponer la épica en ítems independientes entonces introduciríamos estas descomposiciones en el Sprint y sería una buena solución, en el caso de que sea complicado descomponerla, pero es posible de realizar en un Sprint, podría mantenerse como un único ítem en el cual trabajarían varios miembros del equipo. Otra solución sería desarrollar incrementalmente la épica. Para ello tendríamos en un ítem una primera versión que se incluiría en un Sprint y otro ítem en el *Backlog* con el trabajo restante a la épica. Y así sucesivamente hasta terminar todo el trabajo de la épica.
- **Utilización de "Sprints Flexibles".** Esta situación se da cuando no se acuerda con el cliente una fecha de finalización de Sprint. Aquí se pueden pasar ítems no terminados al siguiente Sprint cuando se da por finalizado el actual, además de poder extender o acortar el Sprint sin ningún tipo de problema.
- **Diferentes workflows en el *Backlog* o Sprint.** Cuando un ítem se encuentre en una actividad no aplicable debería saltarse.

4. Herramientas de apoyo para Tableros Kanban

En este capítulo vamos a investigar varias herramientas software externas que aporten tableros kanban para apoyar a los equipos de desarrollo. Intentaremos obtener el máximo de información de cada una para ver qué aportan a los equipos y las funcionalidades que tienen sus respectivos tableros. Para finalizar, realizaremos una tabla comparativa para resumir todo lo que se comente anteriormente. Este análisis nos ayudará a la hora de desarrollar nuestro tablero para Worki, pues sabremos que cosas debe de soportar y cuáles pueden ser opcionales.

4.1 Jira

Jira[3] es una herramienta de desarrollo software para los equipos ágiles. Jira permite que todos los miembros del equipo puedan planificar, supervisar y publicar software. Además de crear historias de usuarios e incidencias, planificar los Sprints para que se adapten a las necesidades del cliente y distribuir las tareas para todo el equipo de trabajo. También es posible crear informes que mejoren el rendimiento del equipo a través de datos en tiempo real que el equipo puede utilizar. Algo muy importante es que Jira permite entre elegir un flujo de trabajo o crear uno propio que se adapte a las necesidades del equipo.

En la Figura 14 podemos ver un ejemplo de un tablero kanban utilizando la herramienta Jira. En él podemos observar que cada ficha contiene toda la información necesaria para que los miembros del equipo sepan en el estado que se encuentra la unidad de trabajo. También existe un contador de fichas en cada columna del tablero. Para proyectos a gran escala esto puede ayudar para saber como va el flujo del trabajo. En el menú izquierdo tenemos el botón para añadir un nuevo ítem, por lo que no es posible añadirlo directamente desde el tablero. Además, se puede observar, que las fichas no siguen ningún tipo de ordenación en la columna, da igual que tengan mayor o menor número de orden.

[3] Jira https://es.atlassian.com/software/jira

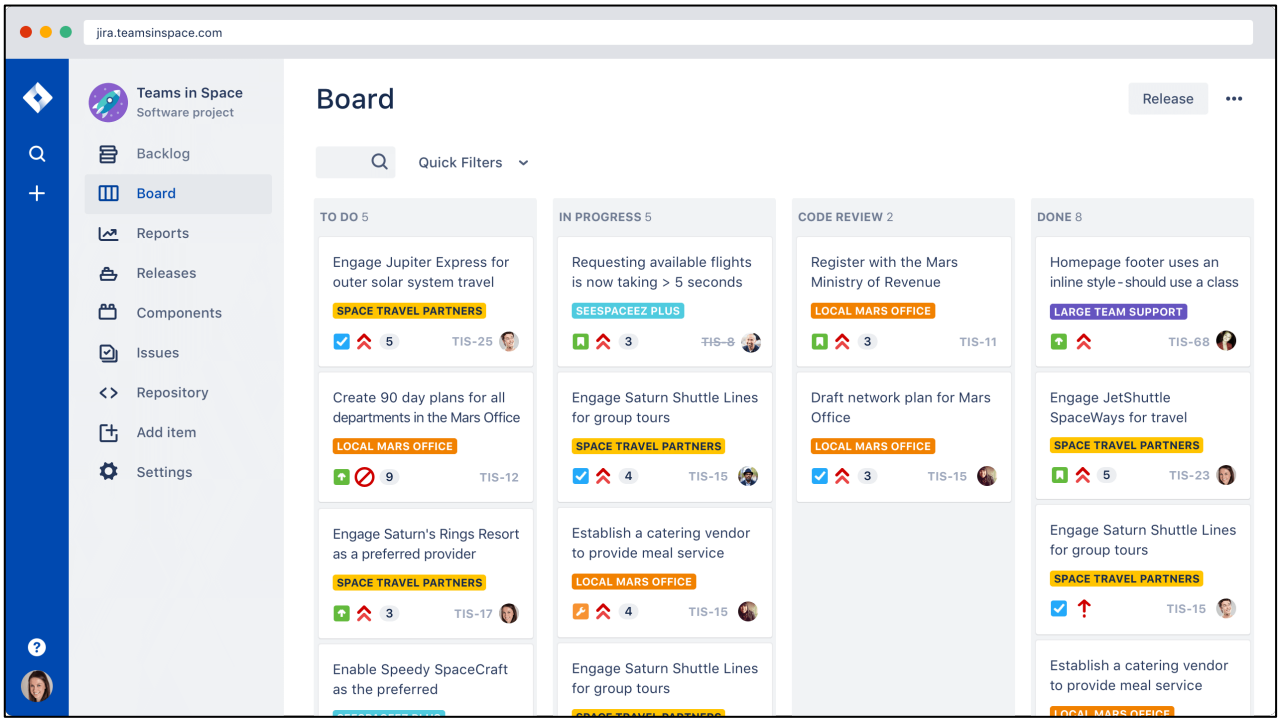

Figura 14. Tablero kanban en Jira [11].

El *drag and drop* de las fichas a cualquier columna también está habilitado en esta herramienta, lo mismo pasa para las columnas, es decir, podemos cambiar el orden de las columnas. Estas funcionalidades aportan mucha libertad al control del tablero kanban. Cuando una columna se encuentra vacía, no se puede ocultar para que no ocupe espacio en la pantalla.

En la Figura 15 podemos observar que se puede editar cualquier información de la ficha si se accede a ella haciendo *click*. Como vemos, es posible filtrar en el tablero kanban, como puede ser por actividad, miembro del equipo, etc.

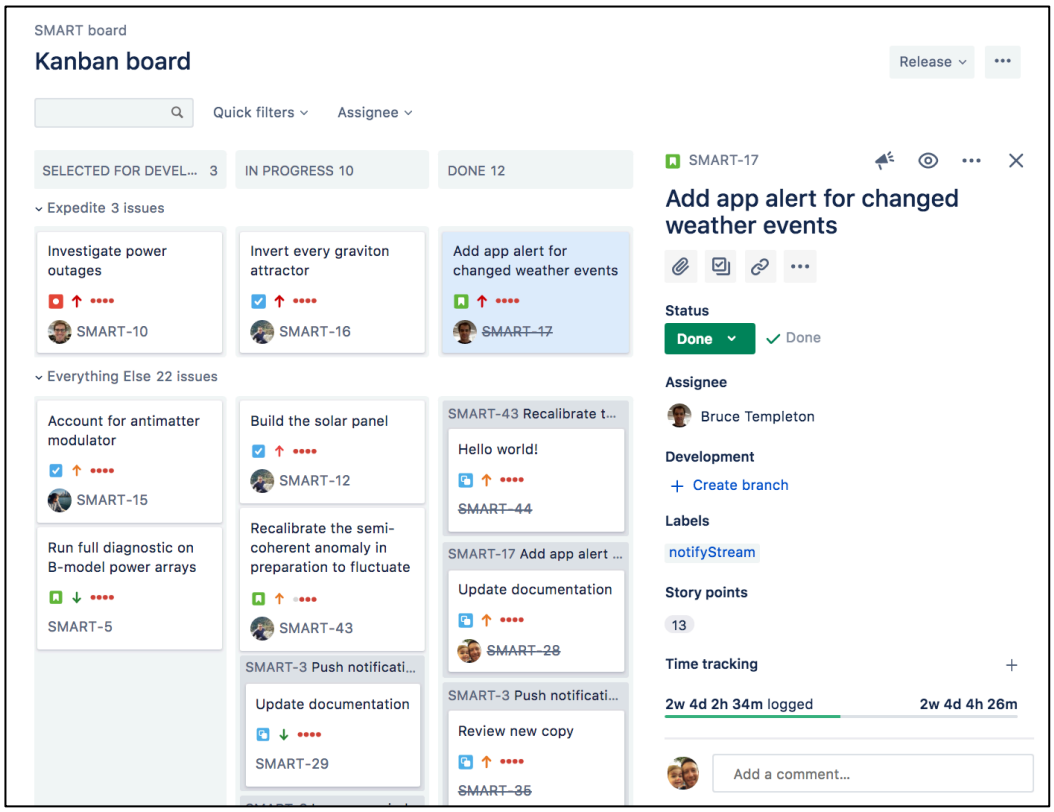

Figura 15. Tablero kanban en Jira [11].

4.2 Targetprocess

Targetprocess[4] es una plataforma visual que ayuda a los usuarios a que adopten nuevas técnicas agiles o que mejoren las que realizan actualmente. También da la posibilidad de desarrollar nuestro propio *framework* para llegar a conseguir agilidad comercial y ver todo el flujo de valor desde la organización entera. Además, aporta visibilidad en todos los incrementos que se producen de los productos y ciclos de entrega.

Como podemos ver en la Figura 16, las fichas contienen la información necesaria, aunque éstas no se encuentran ordenadas en las columnas. Es posible visualizar diferentes productos en el mismo tablero kanban, pero se pueden aplicar filtros en él y reducir el tamaño de búsqueda. Si accedemos a la unidad de trabajo a través de la ficha podemos editar cualquier información. En esta herramienta también pueden moverse todas las fichas a cualquier columna, añadiéndole

[4] Targetprocess https://www.targetprocess.com

flexibilidad al tablero. Es posible editar el workflow que se muestra en el tablero y adaptarlo a las necesidades del equipo. Se puede ocultar cualquier columna haciendo *click* en la flecha que se encuentra a la izquierda del nombre de la columna. En la parte derecha de la columna es posible observar un contador de fichas por columna también. Si movemos una ficha de una columna a otra ésta se priorizará acorde a los puntos que tienen la ficha anterior y posterior a él. También es posible añadir nuevos ítems directamente desde el tablero.

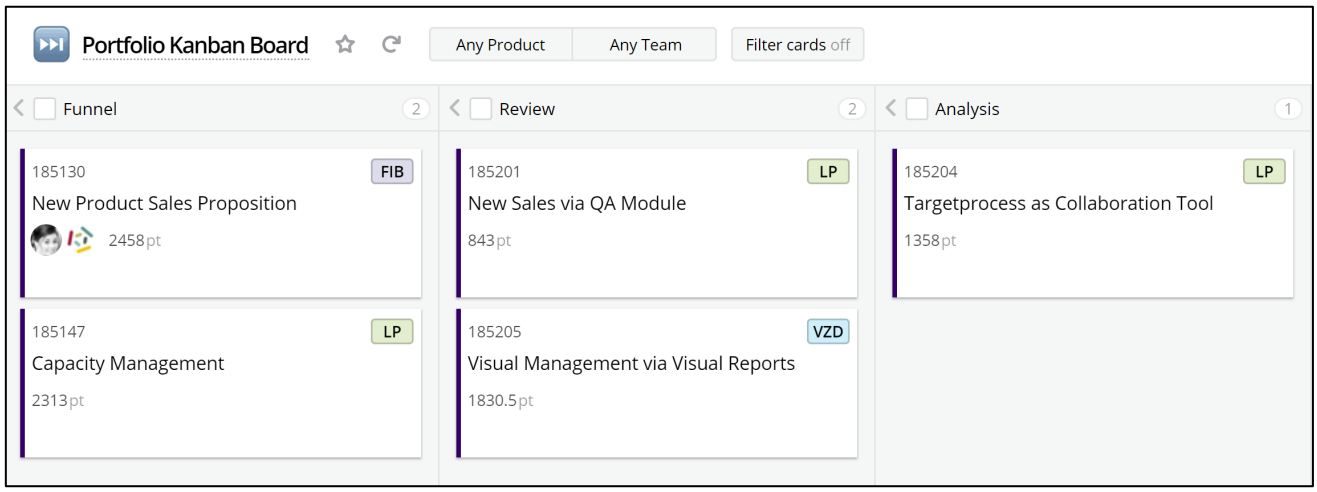

Figura 16. Tablero kanban en Targetprocess [23].

En la Figura 17 podemos ver otro tablero kanban utilizando la herramienta Targetprocess:

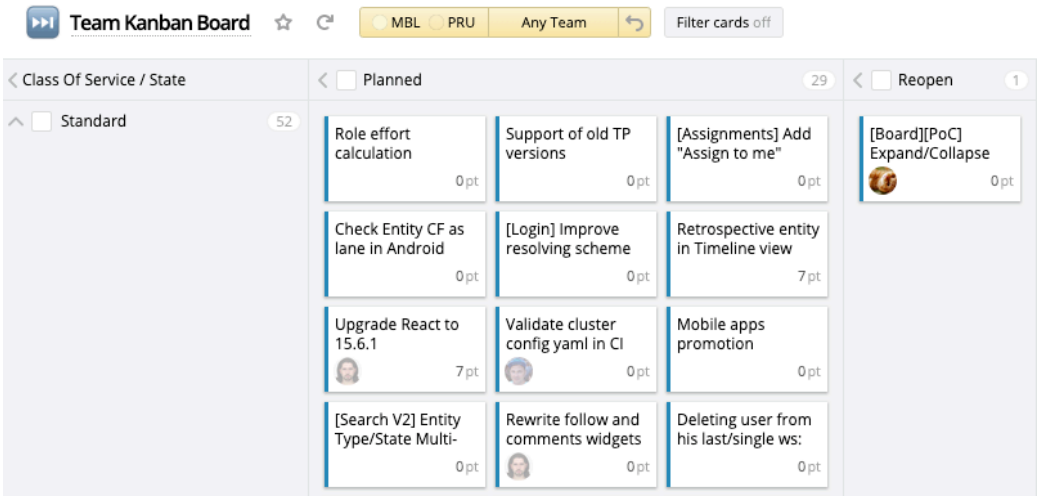

Figura 17. Tablero Kanban en Targetprocess [23]

4.3 Pivotal Tracker

Pivotal Tracker[5] es otra herramienta que aporta una visualización de tablero kanban para aquellos equipos ágiles. Vamos a ver algún tablero realizado con esta herramienta y las características que nos puede ofrecer.

En la Figura 18 podemos observar un tablero kanban utilizando la herramienta Pivotal Tracker. Como podemos ver, la ficha contiene toda la información necesaria para que cada miembro del equipo pueda tener claro de forma instantánea en el estado que se encuentra la unidad de trabajo. En Pivotal Tracker es posible mover las columnas de posición, como sucedía en Jira. Tampoco existe ninguna priorización de las fichas en las columnas, pero si que cuenta con un contador de fichas por cada columna. Aquí si es posible añadir nuevas fichas directamente al tablero, a través del icono + que se encuentra en la parte derecha de la columna. En esta herramienta no es posible modificar el workflow, siempre serán las mismas actividades para el tablero kanban. También es posible ocultar una columna por si se encuentra vacía y así que no ocupe espacio en la pantalla.

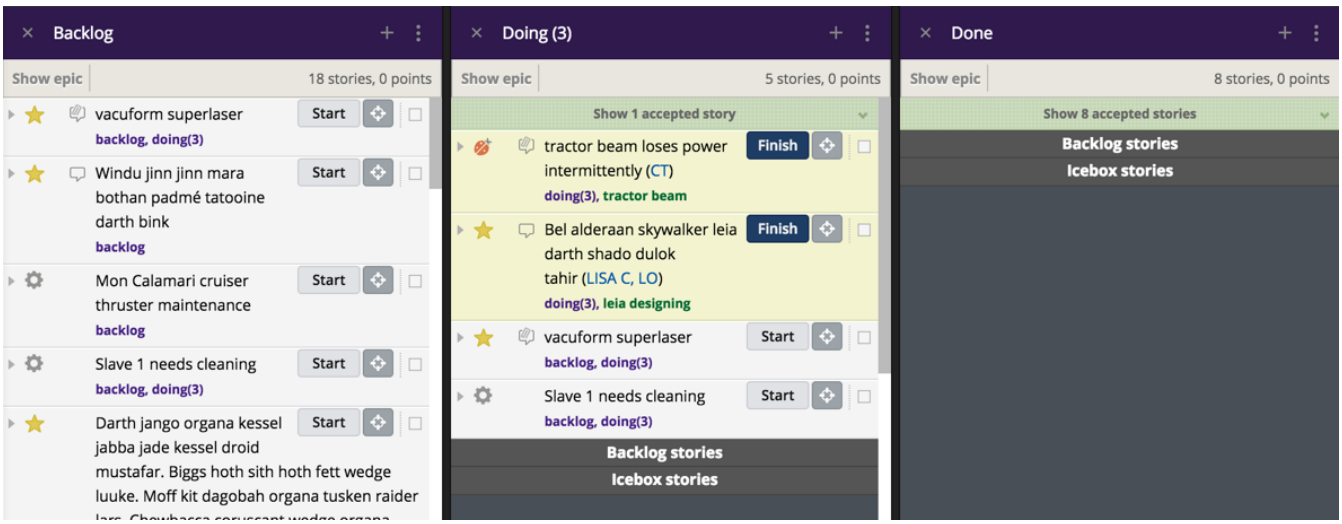

Figura 18. Tablero Kanban en Pivotal Tracker.

[5] Pivotal Tracker https://www.pivotaltracker.com

4.4 Tabla comparativa

Gracias a este estudio hemos obtenido información sobre qué debería ofrecerse en un tablero y qué características pueden añadirse en caso de que se consideren necesarias. Ahora vamos a pasar a resumir este apartado en una tabla comparativa entre todas las herramientas que han sido presentadas.

Las características que se van a comparar son las siguientes: el *drag and drop* de las fichas a cualquier columna, es decir, poder mover las fichas desde cualquier sitio a cualquier otro. Lo mismo con las columnas, si existe la posibilidad de mover las columnas para cambiar su orden. Si es posible controlar el flujo de trabajo del tablero, como puede ser modificar el workflow activo, crear uno nuevo, etc. También es importante la información que se muestra en la ficha, es importante que cuente con toda la información necesaria para que los miembros del equipo comprendan rápida y fácilmente en el estado que se encuentra. Varias herramientas ofrecen la posibilidad de añadir nuevas fichas directamente desde el tablero, por lo tanto, veremos cuáles pueden y cuáles no. Tener un contador de fichas en cada columna puede resultar útil en aquellos proyectos de gran magnitud, así que veremos que herramientas lo ofrecen. Poder ocultar columnas cuando se encuentren vacías puede ser una funcionalidad para tener en cuenta, por lo que, esta característica la vamos a comparar también. Todas las herramientas permiten la edición de la unidad de trabajo una vez se accede a ella a través de la ficha, así es más fácil para el desarrollador cambiar alguna información en una determinada situación sin tener que perder mucho tiempo en ello, por lo que la añadiremos a nuestra tabla comparativa.

Característica	Jira Software	Targetprocess	PivotalTracker
Drag and drop de las fichas	Sí	Sí	Sí
Cambiar de orden las columnas	Sí	No	Sí
Control del flujo de trabajo	Sí	Sí	No
Fichas con información clara	Sí	Sí	Sí
Añadir directamente nuevas fichas	No	Sí	Sí
Contador de fichas	Sí	Sí	Sí
Ordenación de los fichas	No	No	No
Ocultar columnas si están vacías	No	Sí	Sí
Varios productos en el mismo tablero	No	Sí	No
Editar el ítem al acceder a él	Sí	Sí	Sí
Priorizar un ítem al moverlo	No	Sí	No

Tabla 3. Tabla comparativa de herramientas software

Como podemos observar en la Tabla 3 hay varias funcionalidades que todas soportan y que no deberíamos dejar de pasar en nuestro desarrollo, entre ellas se encuentran el *drag and drop* de las fichas y la posibilidad de editar el ítem accediendo a el desde el tablero. Ésta última puede ser realmente útil para realizar cambios sobre un ítem sin perder mucho tiempo. Sin embargo, que ninguna herramienta de soporte a la ordenación de los ítems en las diferentes actividades es algo que hay que destacar, pues, esto puede crear confusión al equipo de desarrollo sobre que ítems deben realizarse primero y cuáles no son tan prioritarios. También queremos contar con fichas que tengan la suficiente información para los miembros del equipo. Que ninguna herramienta ofrezca la posibilidad de ordenar las fichas en las columnas resulta curioso, pues consideramos que, para nuestro tablero, las fichas deberán estar ordenadas por prioridad para que los miembros del equipo de desarrollo puedan saber cuáles son las más importantes y deben abordar.

5. Workflows y tablero Kanban en Worki

Worki es una herramienta desarrollada en el ámbito académico de la escuela de informática de la UPV y que ayuda a la gestión ágil del trabajo. Esta herramienta cuenta con la visualización de una variante a un tablero kanban. Worki cuenta con varios diagramas para supervisar el trabajo de un proyecto, como el Diagrama de flujo acumulado, el *Burndown,* etc. Además, posibilita todas las operaciones para gestionar proyectos, actividades, líneas de trabajo, Sprints, workflows, etc.

Actualmente, la principal forma de visualizar todas las tareas es a través de un grid, el cual hace la función del tablero, pero no existen ni columnas ni fichas que se puedan mover de una actividad a otra. Este grid cuenta con una serie de filtros de donde se obtienen la información que se desea. Una vez se ha seleccionado la información de los filtros, aparece el grid donde se encuentran las tareas y un pequeño grid para ver las actividades que se están aplicando con un contador de ítems en cada actividad, o lo que es lo mismo, el workflow. En la Figura 19 podemos observar cómo se visualizan las tareas actualmente en Worki:

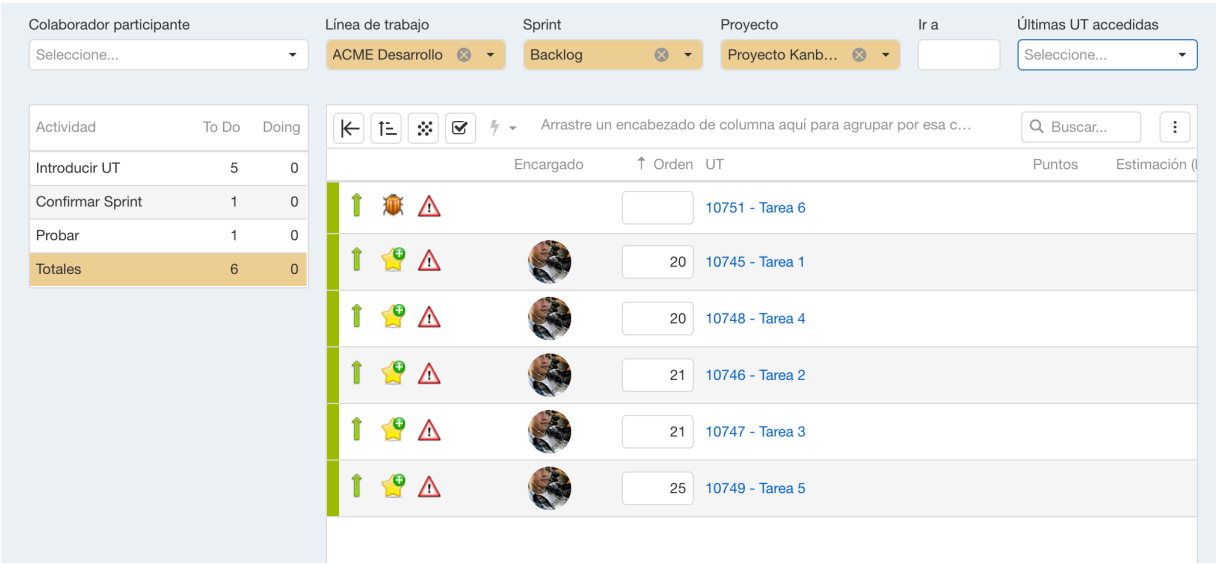

Figura 19. Visualización estándar en Worki

En cada unidad de trabajo (UT) es posible visualizar:

- Imágenes del flujo en el que se encuentra y el tipo de tarea.
- Imagen del colaborador asignado.
- Nombre de la tarea
- Línea de trabajo a la que pertenece.
- Sprint al que está asignado.
- Nombre del proyecto en el que se encuentra.
- Número de orden de la tarea.
- Nombre del workflow al que pertenece.
- La estimación en horas de la tarea.
- Actividad en la que se encuentra.
- La importancia, urgencia y riesgo que conlleva.

Desde Worki se puede controlar todo el flujo de los ítems de cada proyecto, desde iniciar/finalizar una actividad, decidir a que actividad va a ser destinada, asignarle un colaborador a una actividad, añadir a la unidad de trabajo cualquier información necesaria para que el desarrollador pueda entenderla a la perfección, como puede ser, asignarle una línea de trabajo y/o proyecto, añadirla al Sprint al cual pertenece, establecerle un número de orden para priorizar la UT y muchas cosas más.

Aparte de todo lo que sabemos ya de Worki, hay que comentar las funcionalidades especiales con las que cuentan los workflows en Worki, y que seguramente resulten un desafío a la hora de implementar el tablero kanban. En Worki, las UT se ordenan a través de un orden, el cual se asigna una vez se crea la UT. En la visualización estándar de Worki, todas las UT se encuentran ordenadas por este orden, por lo tanto, cuando implementemos nuestro tablero, las UT deberán estar ordenadas en todas las columnas, y si movemos alguna ficha de lugar deberá cambiarse el orden también. Puede darse el caso, aunque no es muy común, de existir una UT donde están trabajando varios miembros del equipo en paralelo y se encuentren en diferentes actividades del workflow. Cuando desarrollemos nuestro tablero, debemos de tener en cuenta este caso, pues habrá que mostrar la misma UT en diferentes columnas del tablero. Otra funcionalidad que permite Worki, es poder tener varios workflows asignados a una misma línea de trabajo. Como sabemos, las columnas de un tablero hacen referencia a las actividades de un workflow, por lo que no será posible mostrar las UT en el tablero de dos workflows diferentes. Para

solucionar esto, será necesario añadir un nuevo filtro de búsqueda y que sirva para seleccionar un workflow determinado. Además de todo esto, el tablero kanban pueden integrarse con Sprints y/o Proyectos, por lo que si se selecciona un Sprint especifico en los filtros, solo habrá que mostrar aquellas UT que pertenezcan a ese Sprint en un determinado workflow.

Teniendo en cuenta todo esto, sería interesante contar con un tablero kanban donde fuera posible visualizar todos o casi todos los datos que se visualizan en el grid, pero de una forma más clara y dinámica, obviamente manteniendo todos los protocolos y funcionalidades que ya ofrece Worki y se han comentado anteriormente. En esta nueva visualización, todas las actividades del workflow harían la función de las columnas, y en cada una de ellas, aparecería una ficha por cada unidad de trabajo, cada una en su actividad correspondiente.

6. Tecnologías utilizadas

El front-end de la aplicación web de Worki está desarrollada con Angular, por tanto, hay que buscar componentes kanban para Angular. En este capítulo nos centraremos en estudiar las tecnologías utilizadas para el desarrollo de nuestro tablero.

6.1 Angular 

Angular[6] es uno de los *framework* para aplicaciones web más populares de estos últimos años. Fue creado para facilitar el desarrollo de aplicaciones web SPA (Single Page Application). Una de las filosofías de Angular es separar completamente el front-end y el back-end. Utiliza el Modelo Vista Controlador (MVC) para facilitar el desarrollo y conseguir que las pruebas sean más efectivas y fáciles de utilizar [3].

6.1.1 Instalación de un proyecto Angular

En este apartado, vamos a ver qué necesitamos realizar para instalar un mini proyecto de la última versión de Angular. Actualmente, Angular se encuentra en la versión 7 aunque es posible instalar cualquier de las otras versiones disponibles.

El primer paso es instalar la ultima versión de NodeJS[7], para ello entraremos a su web oficial y descargaremos el instalador más actualizado que encontremos. NodeJS es un entorno de ejecución para JavaScript que está orientado a eventos asíncronos, con esto podremos construir aplicaciones escalables.

Una vez tengamos instalada la última versión de NodeJS deberemos actualizarnos el gestor de paquetes de node y descargar todas las dependencias necesarias en su versión más actual. Este gestor ya se habrá descargado al realizar la descarga de NodeJS, por lo que solo deberemos de preocuparnos en actualizarlo. Para ello, desde la consola podemos lanzar el siguiente comando:

[6] Angular https://angular.io
[7] NodeJS https://nodejs.org/es/

```
npm install -g npm@latest
```

El siguiente paso es borrar la caché del gestor de paquetes de node a través del comando que vemos:

```
npm cache clean --force
```

A continuación, desinstalaremos todos los paquetes anteriores que tengamos instalados de Angular CLI para nuevamente volver a instalarlo en su última versión. Para desinstalar utilizaremos los siguientes comandos:

```
npm uninstall -g angular-cli
npm uninstall -g @angular/cli
```

Y para volver a instalarlo utilizaremos el siguiente comando:

```
npm install -g @angular/cli@latest
```

Angular CLI (*Command Line Interface*) es el intérprete de línea de comandos que nos permitirá crear proyectos de Angular con su esqueleto básico de una aplicación Angular. Esta herramienta viene proporcionada por el equipo de Angular, pues facilita todo el proceso de inicio de todas las aplicaciones con Angular. Además, también sirve en las etapas de desarrollo o de pruebas.

Ahora que ya tenemos todo instalado y actualizado a la ultima versión, es hora de crear un nuevo proyecto de Angular 7. Abrimos la terminal y escribimos el siguiente comando:

```
ng new
```

Seguidamente deberemos responder al asistente. En la Figura 20 se muestran los siguientes pasos para completar la instalación del proyecto:

1. Nombre del proyecto.
2. Deberemos responder No a si queremos añadir *routing* a nuestra aplicación.

3. Nos preguntará si queremos utilizar un formato especifico para nuestros estilos CSS, a lo que pulsaremos la tecla *enter* simplemente.
4. Ahora ya solo queda esperar a que se genere nuestro nuevo proyecto en Angular 7.

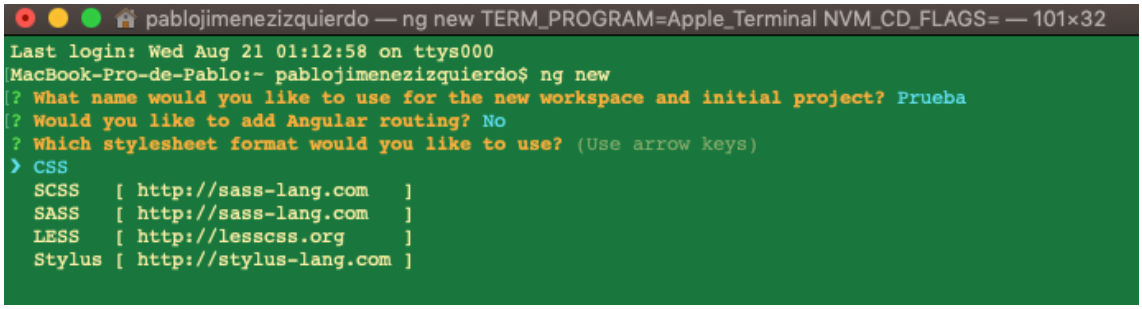

Figura 20. Asistente de instalación del proyecto Angular

Para lanzar nuestro nuevo proyecto en el servidor local de pruebas tan solo debemos entrar al directorio donde se encuentra nuestra aplicación y lanzarlo, para ello podemos utilizar los siguientes comandos:

```
cd NOMBRE_DEL_PROYECTO
ng serve
```

Para acceder al servidor de pruebas tan solo tendremos que navegar a la siguiente *url* en cualquier navegador web: **http://localhost:4200/** .

6.1.2 Estructura de una aplicación Angular

Ahora que ya hemos generado nuestro proyecto y hemos comprado su ejecución, vamos a pasar a estudiar la estructura de una aplicación Angular. Hay una gran cantidad de carpetas y archivos, pero la mayoría son para configurar la aplicación.

6.1.2.1 Visual Studio Code 

Visual Studio Code[8] será nuestro editor de código para ver y editar cualquier archivo de nuestro proyecto. Tiene integrado un terminal que nos facilitará realizar cualquier comando de Angular CLI y es posible mejorarlo añadiéndole cualquier extensión disponible.

6.1.2.2 Carpetas y Ficheros principales

En la Figura 21 podemos ver la estructura básica de un proyecto Angular. Dentro encontraremos muchos archivos de diferentes tipos. Especialmente, los archivos que tengan la extensión **.ts** son para ficheros *TypeScript*, el cual es una evolución de *JavaScript* y que nos ayudará a la hora de desarrollar nuestro código.

Los archivos principales son los siguientes:

- **angular.json:** configuración del propio CLI.
- **package.json:** dependencias de las librerías utilizadas y los scripts.
- **src/:** carpeta donde se encuentran todos los archivos fuentes.
  - **index.html:** fichero HTML índice estándar.
  - **main-ts:** el fichero TypeScript que sirve para arrancar la aplicación.
  - **app/:** carpeta donde se encuentra el código especifico de nuestra aplicación
    - **app.module.ts:** la raíz de todo el árbol de módulos que se encuentran en la aplicación.
    - **app.component.ts:** la pagina web es un árbol de componentes, éste es el componente raíz.
    - **app.component.html:** parte visual del componente, lo que se conoce como vista.

[8] Visual Studio Code https://code.visualstudio.com/

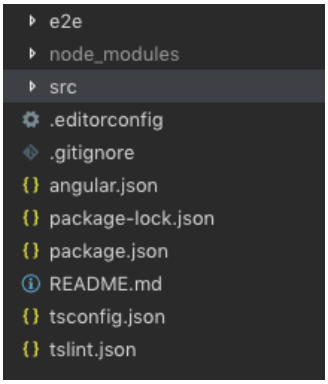

Figura 21. Estructura general de un proyecto Angular

6.1.3 Visualización inicial del proyecto

Si arrancamos nuestro proyecto y abrimos nuestro servidor de pruebas de Angular observaremos la aplicación inicial que se nos ha creado como podemos ver en la Figura 22. Ahora ya solo es cuestión de ir diseñando la pagina web a gusto del usuario.

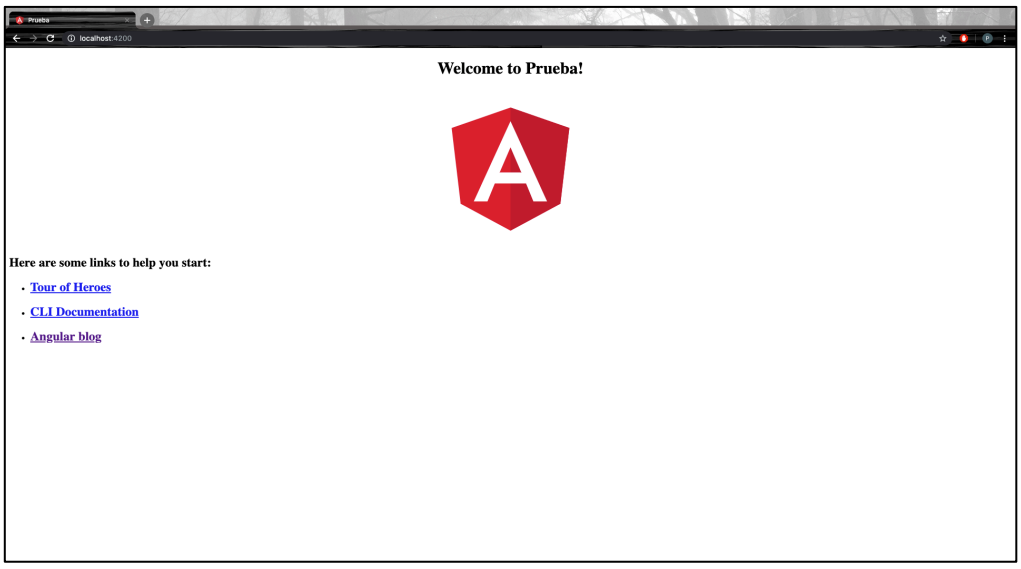

Figura 22. Visualización inicial del nuevo proyecto con Angular 7.

6.2 CSS

Una vez tengamos nuestro tablero funcionando correctamente, habrá que darle un buen aspecto para que no resulte muy complicado de visualizar y resulte bonito para el usuario. Para ello, vamos a utilizar CSS.

CSS[9] (*Cascading Style Sheets*) que en español viene a significar "Hoja de estilos en cascada" es un lenguaje de diseño gráfico y lo vamos a utilizar para darle el diseño visual adecuado a nuestro tablero kanban. Son denominados en cascada por que se van aplicando de arriba abajo y si existe algún caso de ambigüedad se sigue alguna norma para solucionarlo. CSS trata de implantar la separación de presentación y contenido, es decir, separar los documentos donde algunos contengan solamente la información y los datos y otros todo lo relacionado con el aspecto visual. ¿Qué ventajas nos aporta tener únicamente un documento CSS?

1. Si hay que realizar modificaciones solo hay que hacerlas en un único documento y no en todos.
2. Eliminación de estilos duplicados en varios lugares.

6.2.1 Ejemplo CSS

Para entender mejor el funcionamiento del CSS vamos a ver un ejemplo sencillo de como puede arreglar un simple código CSS para un HTML básico. Imaginemos que necesitamos crear una página web para un formulario de registro, primero de todo habrá crear el código HTML para esta web. En la Figura 23 podemos ver el código HTML para nuestro ejemplo. Se trata de un formulario básico donde se introducen datos personales del usuario y un *dropdown* para elegir a la comunidad autónoma que pertenece el usuario.

[9] CSS https://www.w3schools.com/css/

```html
<body>
    <form action="#" method="post" class="formulario">
        <div><h2>Formulario de registro</h2></div>
        <div><label>Nombre : <input type="text" name="name"></label></div>
        <div><label>Apellidos : <input type="text" name="apellidos" ></label></div>
        <div><label>Correo electrónico : <input type="text" name="email"></label></div>
        <div><label>Contraseña : <input type="password" name="password"></label></div>
        <div>
            <label>Edad : <input type="number" name="number" min="13" max="80"></label>
            <label>
                Com :
                <select id="pais">
                    <option>Cataluña</option>
                    <option>Comunidad Valenciana</option>
                    <option>Madrid</option>
                    <option>Aragón</option>
                    <option>Castilla La mancha</option>
                    <option>Andalucia</option>
                    <option>Extremadura</option>
                </select>
            </label>
        </div>

        <div><input type="submit" name="submit" value="Registrar"></div>


    </form>
</body>
```

Figura 23. Código HTML del ejemplo

En la Figura 24 vemos el resultado de nuestro código HTML en la web. Se trata de un diseño súper pobre donde todo se encuentra muy junto y no es muy atractivo para la vista. Para solucionar estos problemas entra en acción el CSS, con el que podremos ordenar de una mejor forma todos los elementos de la vista y darles un diseño más formal.

Figura 24. Formulario de ejemplo

El código CSS al principio puede resultar complicado de entender, su curva de aprendizaje es muy exponencial, pero una vez se entiende como funciona es sencillo y se pueden hacer cosas muy espectaculares. En la Figura 25 vemos todo el código CSS utilizado.

```css
.formulario input[type='text']:focus, .formulario input[type ='password']:focus
{
    outline: none;
    box-shadow: 0 0 0 3px rgb(206, 213, 215);

}
.formulario input[type='number']
{
    padding: 7px 6px;
    width: 62px;
    border: 1px solid #CED5D7;
    resize: none;
    box-shadow: 0 0 0 3px #EEF5F7;
    margin:  5px 0;
    font-size: 9pt;
}
.formulario select
{
    padding: 7px 6px;
    width: 124px;
    border: 1px solid #CED5D7;
    resize: none;
    box-shadow: 0 0 0 3px #EEF5F7;
    margin:  5px 0;
}
.formulario input[type='submit']
{
    border: 1px solid #CED5D7;
    box-shadow: 0 0 0 3px #EEF5F7;
    padding: 8px 16px;
    border-radius: 20px;
    font-weight: bold;
    text-shadow: 1px 1px 0px white;

    background: #e4f1f6;
    background: -moz-linear-gradient(top,#e4f1f6 0%, #cfe6ef 100%);
    background: -webkit-linear-gradient(top,#e4f1f6 0%, #cfe6ef 100%);
}
.formulario input[type='radio']
{
    margin-left: 20px;
    margin-bottom: 10px;
}
```

Figura 25. Código CSS de ejemplo

Y gracias a nuestro código CSS obtenemos el resultado final que podemos ver en la Figura 26. Ahora ya hemos obtenido un formulario mucho más elegante donde se puede ver todo con más claridad y es más atractivo para la vista.

Figura 26. Visualización final del formulario de ejemplo.

7. Componentes para soporte Kanban

Una vez ya sabemos que tecnologías vamos a utilizar, es hora de pasar a estudiar los componentes que nos ofrezcan las características apropiadas para implantar nuestro tablero kanban. Por lo tanto, se ha hecho una búsqueda de aquellos componentes que ofrecen una solución para desarrollar un tablero kanban con Angular. Como resultado de la búsqueda, se han elegido las siguientes herramientas externas, todas cuentan con un componente kanban: Syncfusion, DlhSoftTeam, DayPilot y JQWidgets. A continuación, estudiaremos las funcionalidades que ofrecen cada una de ellas para acabar realizando una tabla comparativa y escoger con cual realizaremos nuestro desarrollo para Worki.

7.1 Syncfusion

Syncfusion[10] provee más de mil componentes UI para WinForms, WPF, ASP.NET WebForms, MVC, Core, UWP Xamarin, JavaScript, Angular y React.

Entre todos los componentes que pueden ser utilizados se encuentra el tablero kanban. Entre las características más importantes y que están disponible podemos encontrar una plantilla para la ficha y así poder reproducirla de la forma que se desee, posibilidad de editar el ítem de cualquier modo, existencia de filtros y muchas características más que se van a mencionar de ahora en adelante.

- Columnas: Es posible añadir todas las columnas que sean necesarias con el nombre que se desee.
- Workflow: Posibilidad de establecer el flujo de trabajo de las columnas y así imposibilitar el movimiento de fichas entre columnas que no siguen el flujo.
- *Swim lanes*: Nueva forma de categorización horizontal de los ítems que aporta transparencia al workflow. Con la posibilidad de mover ítems entre diferentes *swim lanes* si se da el caso. En la Figura 27 se ilustra esta nueva forma de categorizar los ítems.

[10] Syncfusion https://www.w3schools.com/css/

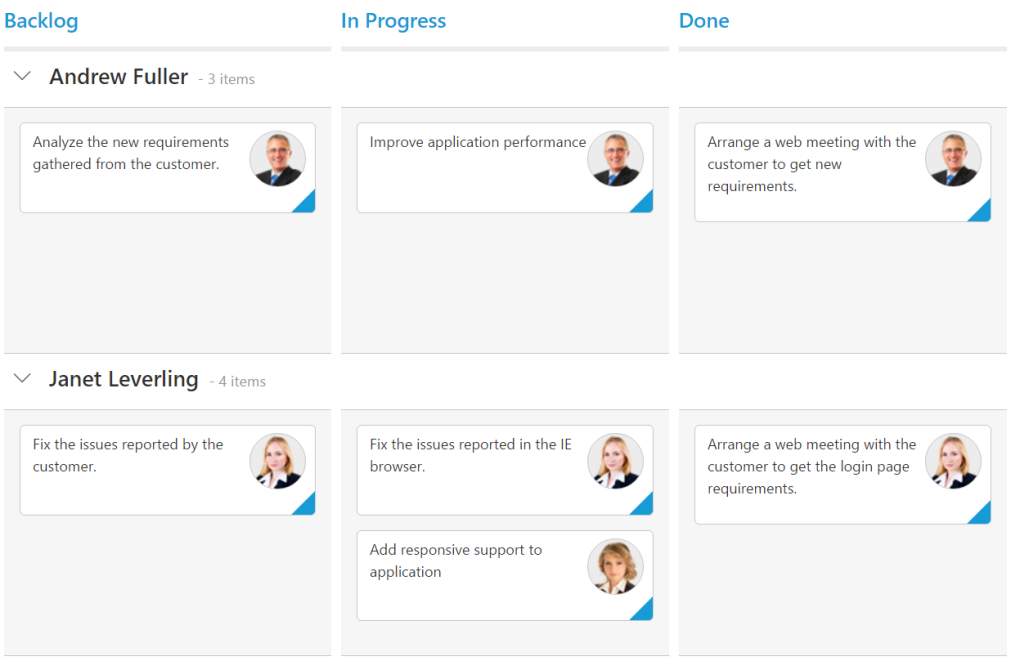

Figura 27. Swim lanes en la herramienta Syncfusion [22]

- *Drag and drop*: Existe la posibilidad de no permitir el movimiento de fichas entre columnas, aunque por defecto está disponible.
- Filtros: Es posible añadir filtros al tablero kanban. En la Figura 28 se muestran los filtros de Syncfusion.

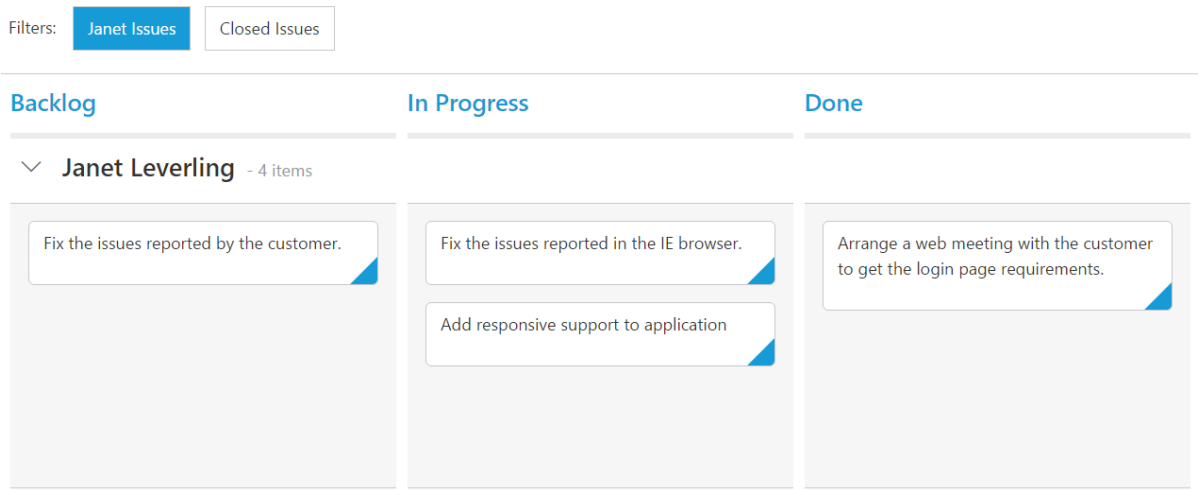

Figura 28. Filtros en la herramienta Syncfusion [22]

- Plantilla de ficha modificable a través de html.
- Edición de la ficha a gusto del desarrollador.
- Comportamiento "responsive" dependiendo del tamaño de la pantalla por donde se esté visualizando el tablero.

Un tablero kanban básico desarrollado gracias al componente de Syncfusion quedaría como puede verse en la Figura 29:

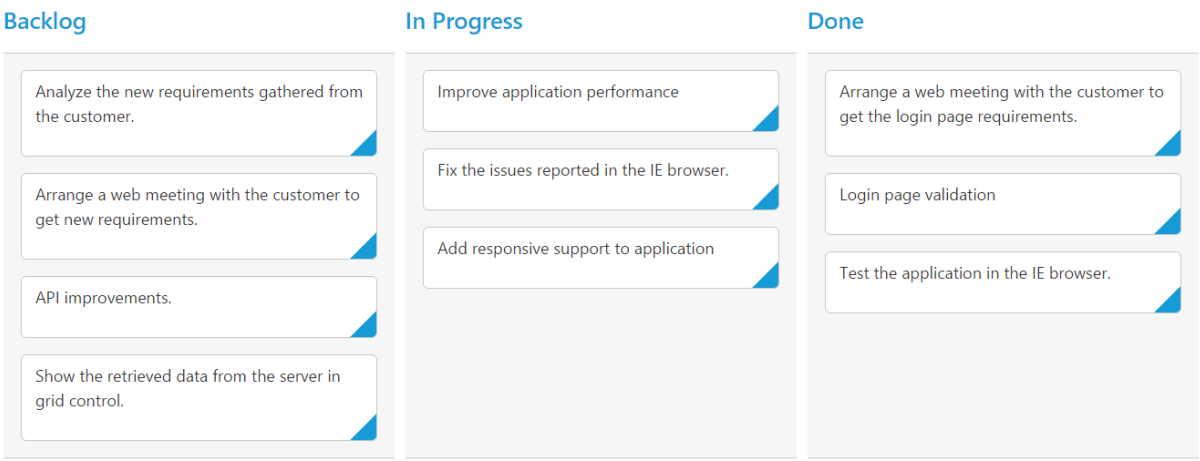

Figura 29. Tablero Kanban con la herramienta Syncfusion [22]

7.2 DayPilot

Otro tablero kanban que es posible utilizar es el que nos ofrece DayPilot[11]. Puede que sea el componente que menos características proporciona, pero vamos a repasarlas.

- Mismo comportamiento en las columnas que los demás componentes: añadir todas aquellas que sean necesarias y con el nombre deseado.
- *Drag and drop* disponible para los ítems, columnas y *Swim lanes*.
- Posibilidad de añadir *Swim lanes* si se considera oportuno.
- Añadir, eliminar, y modificar ítems en tiempo real.

[11] DayPilot https://www.daypilot.org/

En la Figura 30 podemos observar un tablero kanban utilizando este componente:

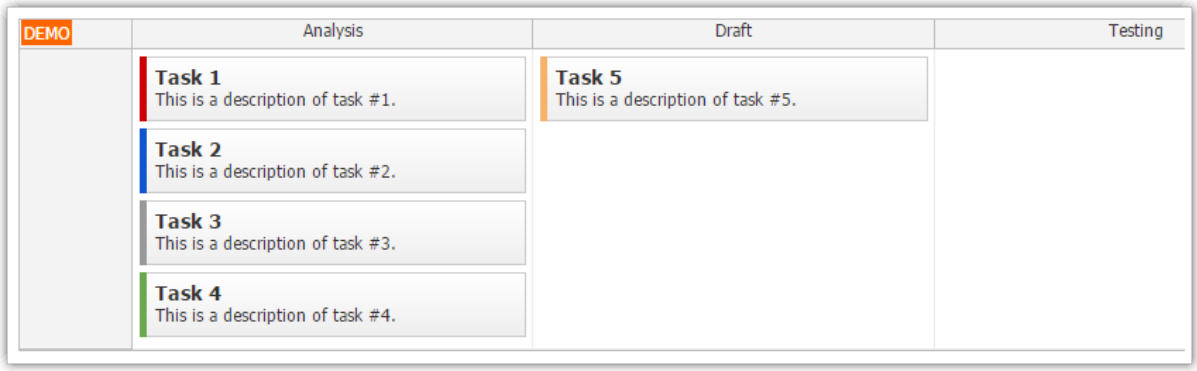

Figura 30. Tablero kanban con la herramienta DayPilot [6]

7.3 DlhSoftTeam

DlhSoftTeam[12] ofrece los controles de Gantt Chart, Schedule Chart, Load Chart, PERT Chart, Network Diagram, Tree-grid, Organizational Chart y componentes y marcos genéricos dirigidos a las plataformas y herramientas de desarrollo de Microsoft® .NET, desde WPF a HTML5 y JavaScript®., TypeScript, ASP .NET y UWP (Universal Windows® Platform), y una aplicación de administración de proyectos de Windows® para usuarios finales. Recientemente, también han recurrido al desarrollo nativo de macOS e iOS utilizando Apple® xCode, el desarrollo nativo de Android con Android Studio, el desarrollo multiplataforma con Apache Cordova y Xamarin, y las interfaces 3D y holográficas con Unity.

DlhSoftTeam también ofrece un componente kanban para desarrollo web en Angular. Entre las características más importantes se encuentra la posibilidad personalizar el tablero y se trata de un producto gratis por tiempo ilimitado. Vamos a pasar a ver qué es posible realizar con este componente:

- Separación de ítems en grupos como pueden ser las “story”.
- *Drag and drop* disponible.
- Añadir todas las columnas que se deseen.

[12] DlhSoftTeam https://dlhsoft.com

- Editar la ficha de una forma limitada (color, nombre, estado en el que se encuentra, grupo al que pertenece).
- Posibilidad de crear nuevos ítems en el tablero kanban.
- Es posible editar los ítems del tablero kanban en tiempo real.

En la Figura 31 encontramos un ejemplo de un tablero kanban básico desarrollado con el componente de DlhSoftTeam:

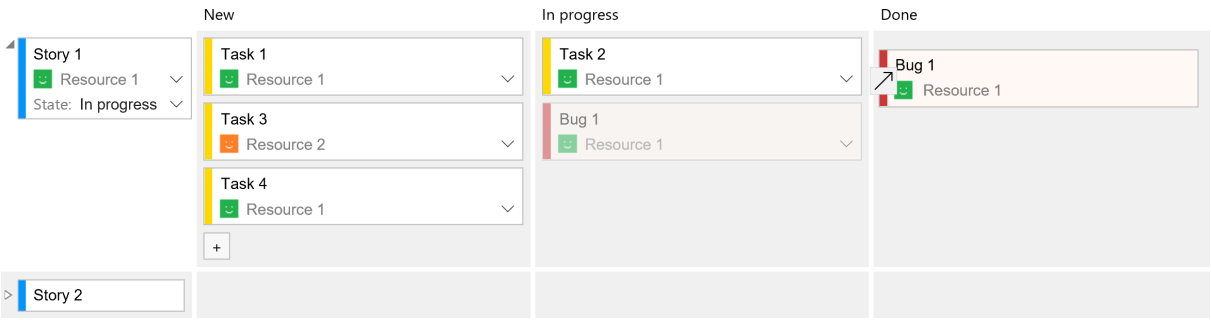

Figura 31. Tablero Kanban con la herramienta DlhSoftTeam [7]

7.4 JQWidgets

JQWidgets[13] provee soluciones completas para desarrollar páginas web profesionales y aplicaciones móviles. Todo está construido a partir de los estándares y tecnologías como HTML5, CSS, JavaScript y JQuery. JQWidgets posibilita el desarrollo de webs reactivas y que son bonitas de cara al usuario. Es posible el uso de JQWidgets con TypeScript, frameworks populares como Angular, Vue, React. Las últimas versiones de JQWidgets proporcionan la posibilidad de crear aplicaciones UI modernas en Angular sin la necesidad de requerir de referencias externas de JQuery.

Características del tablero kanban JQWidgets:

- Añadir todas las columnas necesarias.
- Por defecto, las columnas se pueden colapsar o expandir para que no ocupen espacio si están vacías.
- Añadir, eliminar y modificar ítems en tiempo real.

[13] JQWidgets https://www.jqwidgets.com/

- *Drag and drop* disponible en los ítems.
- Plantilla para la ficha para construirla a gusto del desarrollador y así poder modificarla a través de CSS.
- Comportamiento "responsive".
- Unión de diferentes tableros en uno.
- Batería de métodos para obtener cualquier información necesaria para el desarrollo.

El tablero resultante utilizando el componente JQWidgets puede verse en la Figura 32.

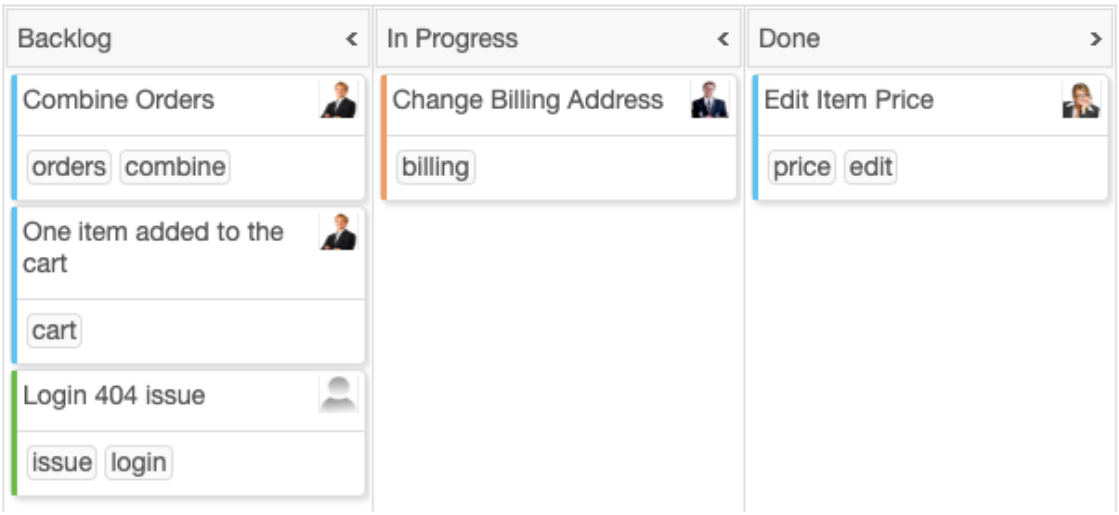

Figura 32. Tablero Kanban con la herramienta JQWidgets [12]

7.5 Tabla comparativa

Una vez se han presentado los diferentes componentes que pueden ser utilizados, es hora de realizar una comparación entre ellos y elegir con cual se va a realizar el desarrollo del tablero kanban para la herramienta Worki. Para esta comparación vamos a tener en cuenta todas aquellas características que se han ido enumerando anteriormente en la descripción de los componentes y se hará un estudio para ver cual de todos es el más viable o aporta cosas realmente útiles a nuestro desarrollo.

Característica	**Syncfusion**	**DlhSoftTeam**	**DayPilot**	**JQWidgets**
Control de columnas	Sí	Sí	Sí	Sí
Expandir y colapsar columnas	No	No	No	Sí
Drag and drop de columnas	No	No	Sí	No
Edición del ítem	Sí	Sí	Sí	Sí
Plantilla HTML para la ficha	Sí	No	No	Sí
Añadir ítems en tiempo real	No	Sí	Sí	Sí
Eliminar ítems en tiempo real	No	Sí	Sí	Sí
Modificar ítems en tiempo real	Sí	Sí	Sí	Sí
Drag and drop para los items	Sí	Sí	Sí	Sí
Comportamiento "responsive"	Sí	Sí	Sí	Sí
Añadir swimlanes	Sí	No	Sí	No
Añadir grupos al tablero	No	Sí	No	No
Crear workflow para el control del flujo	Sí	No	No	No
Añadir filtros al tablero	Sí	No	No	No
Batería de métodos	Sí	No	No	Sí
Varios tableros en uno	No	No	No	Sí
Gratuito	Sí	No	No	Sí

Tabla 4. Tabla comparativa entre las diferentes herramientas expuestas.

Una vez completado el análisis y realizada la tabla comparativa (Tabla 4), podemos observar que hay 2 componentes que sobresalen a los otros en cuanto funcionalidad se refiere: Syncfusion y JQWidgets. Además, ambas opciones son totalmente gratuitas, contando siempre con apoyo del equipo de desarrollo del componente por si surge cualquier duda. Ambos tienen bastantes cosas en común, vamos a centrarnos en las diferencias. Todo lo diferente que nos aporta Syncfusion no nos va a ayudar en nuestro desarrollo, por ejemplo, poder añadir filtros a nuestro tablero no es necesario que el componente lo ofrezca, ya que Worki ya cuenta con sus propios

filtros. La otra funcionalidad que permite Syncfusion, es añadir *Swim Lanes*, pero esto no aporta nada significativo a nuestro desarrollo. Mientras que JQWidgets ofrece características bastante interesantes y que puede que nos sirvan de ayuda para nuestro tablero. Por ejemplo, la posibilidad de expandir o colapsar columnas que estén vacías es algo interesante, así aquellas columnas sin fichas no ocupan espacio innecesario en la pagina web. También es interesante, poder eliminar o añadir ítems en tiempo real desde el tablero, si los miembros del equipo pueden ahorrarse el mayor tiempo posible haciendo estas acciones puede beneficiar a que la entrega de los proyectos sea más rápida. Por lo tanto, la elección para desarrollar el tablero kanban en la herramienta Worki es: JQWidgets.

8. Diseño e implementación de tablero kanban en Worki

En este capítulo vamos a detallar paso a paso todo el procedimiento que se ha seguido para desarrollar el tablero kanban en Worki, desde el tablero inicial hasta el resultado final. Veremos los estilos finales que se han diseñado para las fichas y columnas, explicando por qué se han diseñado de tal manera, además, también repasaremos cómo obtenemos los datos para mostrarlos en el tablero.

8.1 Inicio

Lo primero de todo ha sido crearse un proyecto Angular incluyendo todas las dependencias necesarias para que el componente JQWidgets pueda ser utilizado. Una vez está todo el proyecto creado y las dependencias añadidas lo siguiente es comprobar qué se muestra como tablero kanban básico de los que proporciona JQWidgets. En la Figura 33 se muestra el primer tablero obtenido:

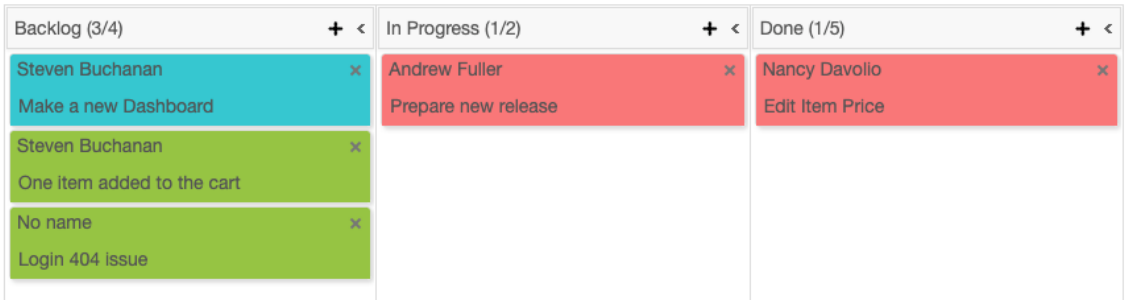

Figura 33. Tablero Kanban inicial.

Una vez obtenida la primera prueba del kanban, se ha comprobado las funcionalidades que vienen por defecto en el tablero y si son realmente útiles para nuestro desarrollo. Entre ellas podemos destacar:

- *Drag and drop* de todos los ítems a cualquier columna.
- Añadir nuevos ítems en tiempo real.

- Cualquier usuario puede eliminar los ítems del tablero.
- Es posible colapsar y expandir las columnas.

Entre las características destacadas, poder eliminar o añadir ítems en tiempo real no es algo que todo usuario deba poder hacer por lo que es una característica que será lo más seguro eliminada.

8.2 Fichas del tablero

Si nos centramos en el aspecto de la ficha, no nos aporta mucha información que no sea el colaborador al que está asignada y el nombre de la tarea, por lo que hay que realizar un nuevo diseño de la ficha. Este diseño se puede realizar más fácilmente gracias a la plantilla HTML que nos aporta JQWidgets. Para el nuevo diseño de la ficha se ha decidido que sea del estilo al que podemos ver en la Figura 34:

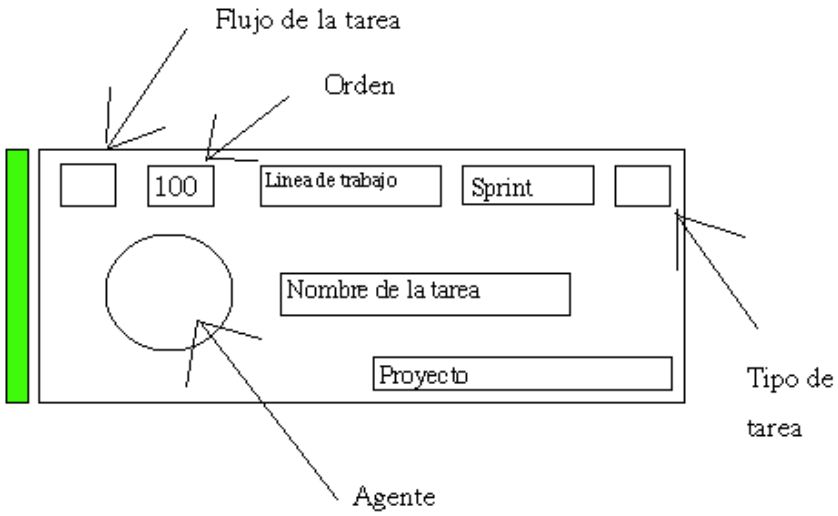

Figura 34. Boceto con el estilo de la ficha

Como podemos observar en el boceto, se ha decido que se muestre la siguiente información:

- Imagen del agente que tiene asignada dicha tarea, añadiéndole un *popup* cuando se deja el ratón encima para saber el nombre de la persona. Existen varios casos donde

no se muestra la imagen del colaborador. Puede que la tarea en concreto no tenga colaborador asignado, entonces se mostrará una imagen por defecto (color gris). En el caso que haya varios colaboradores asignados aparecerá una imagen de varios colaboradores.

Imagen mostrada	Caso
	Colaborador asignado
	Sin colaborador
	Varios colaboradores

Tabla 5. Diferentes tipos de colaboradores y sus respectivas imágenes.

- Nombre de la tarea en cuestión. Se ha establecido un tamaño máximo de 3 líneas para no sobrecargar de mucho texto la ficha.
- Sprint en el que se encuentra la tarea actualmente.
- Nombre de la línea de trabajo a la cual pertenece.
- Número que corresponde al orden de la tarea por el cual serán ordenadas en el tablero.
- Nombre del proyecto al que pertenece.
- Como se puede apreciar, en el borde izquierdo de la ficha aparece un color. Este color indica el estado en el que se encuentra la tarea. Si el color es rojo la tarea estará ACTIVE, si aparece el color naranja quiere decir que se encuentra DOING y, por el contrario, si aparece un color verde significa que el estado es TO DO.
- Imagen correspondiente al tipo de tarea, añadiéndole también un *popup* para saber el nombre:

Imagen	Descripción
	Fallo crítico
	Fallo leve
	Fallo importante
	Fallo moderado

	Tarea
	Nuevo requisito
	Idea
	Mejora
	Ticket fallo

Tabla 6. Diferentes imágenes para los tipos de tareas

- Imagen para mostrar el tipo de flujo de trabajo en el que se encuentra la tarea:

Imagen	**Descripción**
	Adelante
	Trabajo en paralelo
	Salto adelante
	Salto atrás
	Salto a la misma actividad

Tabla 7. Diferentes imágenes para los tipos de flujo

Una vez ya está claro que tipo de información se va a mostrar en los ítems del kanban, se ha modificado toda la plantilla HTML para poder mostrar cada punto detallado anteriormente. Se trata de un HTML donde existe una etiqueta <*div*> que contiene a todas las demás. En cada una de las otras etiquetas <*div*> se muestra todos los datos que se han considerado necesarios. El HTML resultante lo podemos ver en la Figura 35:

```
'<div class=\'jqx-kanban-item\' id=\'\'>' +
'<div class=\'item-container\'>' +
'<div class=\'jqx-kanban-item-tipo\'></div>' +
'<div class=\'jqx-kanban-item-sprint\'></div>' +
'<div class=\'jqx-kanban-item-flujo\'></div>' +
'<div class=\'jqx-kanban-item-linea\'></div>' +
'<div class=\'jqx-kanban-item-order\'></div>' +
'<div class=\'jqx-kanban-item-agente\'></div>' +
'<div class=\'jqx-kanban-item-actividad\'></div>' +
'<div class=\'jqx-kanban-item-proyecto\' style=\'text-align: right !important;\'></div>' +
'</div>' +
'</div>';
```

Figura 35. HTML para las fichas del tablero

A este HTML se le han aplicado los siguiente estilos CSS que podemos ver en las Figuras 36 y 37:

```css
.jqx-kanban-item {
    border-radius: 10px;
    padding-top: 5px;
    min-width: 330px;
}
.item-container {
    border-left: 7px solid rgb(156,188,0);
}
.jqx-kanban-item-tipo {
    float: right;
    margin-right: 8px;
    cursor: default;
}
.jqx-kanban-item-sprint span{
    line-height: 16px;
    margin-top: 3px;
    margin-left: 5px;
    color: black;
    float: right;
    padding: 0px 12px 0px 12px;
    margin-right: 2px;
    background-color: rgba(255,255,51,0.5);
    font-size: 9px;
    border-radius: 5px !important;
}
.jqx-kanban-item-flujo {
    float: left;
    margin-left: 5px;
    cursor: default;
}
```

Figura 36. Estilo CSS

```css
.jqx-kanban-item-linea span{
    float: right;
    margin-top: 3px;
    font-size: 9px;
    background-color: rgba(147,112,219,0.5);
    padding: 0px 12px 0px 12px;
    line-height: 16px;
    border-radius: 5px !important;
}
.jqx-kanban-item-order span{
    float: left;
    margin-top: 3px;
    font-size: 11px;
    margin-left: 5px;
}
.jqx-kanban-item-agente {
    clear: both;
    padding: 10px 5px;
    display: -webkit-box;
    display: -ms-flexbox;
    display: flex;
    -webkit-box-align: center;
    -ms-flex-align: center;
    align-items: center;
    -webkit-box-pack: start;
    -ms-flex-pack: start;
    justify-content: flex-start;
}
```

Figura 37. Estilo CSS

En la Figura 38 podemos observar el resultado obtenido tras aplicar los estilos y el cambio de HTML. Se ha conseguido una ficha donde todos los datos que se muestran están en su sitio correspondiente del boceto, sin estar todos ellos muy juntos para que los miembros del equipo puedan ver la información de forma clara y rápida.

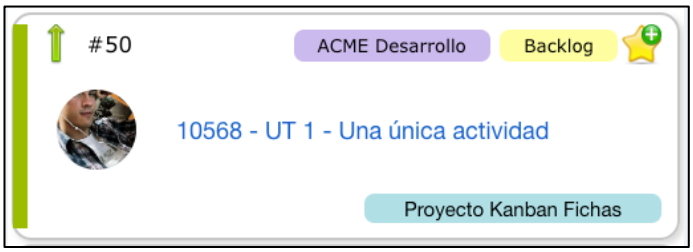

Figura 38. Resultado final para el estilo de la ficha.

8.3 Columnas

Una vez el diseño de las fichas ha sido terminado, es hora de pasar a las columnas del tablero. Las columnas en un tablero equivalen al workflow, es decir, son las actividades por las que deben de pasar los ítems para considerarse terminados. Para éstas, tan solo con mostrar el nombre de ella misma y el número de fichas que contiene es suficiente. El motivo de indicar el número de ítems contenidos es para que en proyectos grandes el equipo de desarrollo pueda saber, en todo momento, donde puede haber una acumulación de ítems o el número de ítems que existe en cada actividad. En la Figura 39 podemos ver estos detalles.

Figura 39. Estilos para las columnas

Además, se ha considerado oportuno la posibilidad de colapsar las columnas, por si en algún caso estamos buscando cierto ítem en cierta actividad en el kanban, así sería posible no tener las demás columnas ocupando espacio en la pantalla. También se ha introducido que automáticamente se colapsen las columnas que no tengan ítems en ella para que tampoco ocupen espacio innecesario. En la Figura 40 podemos observar el desarrollo de esta funcionalidad.

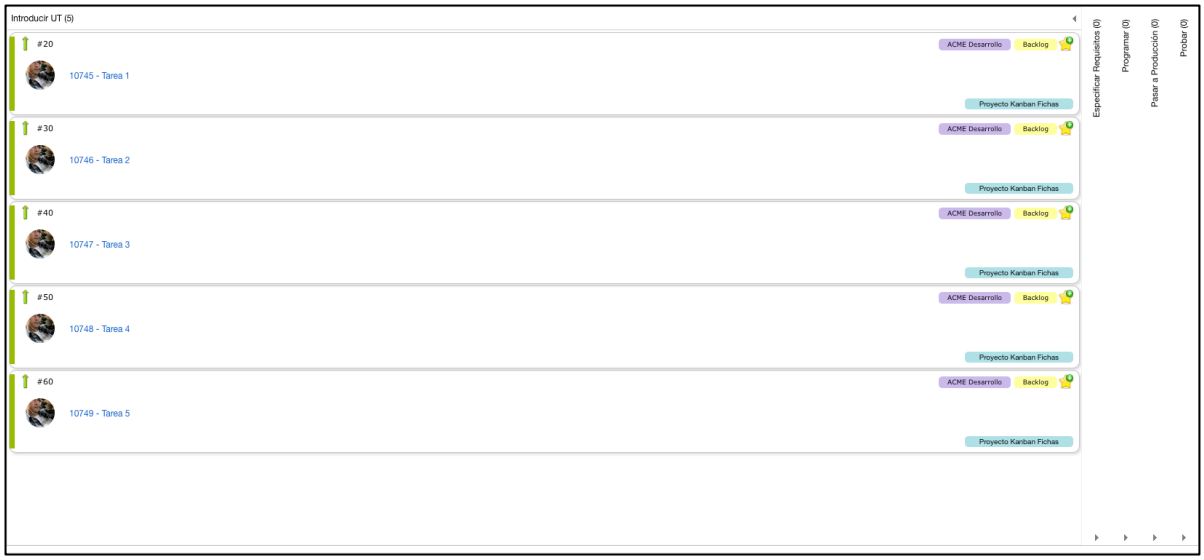

Figura 40. Columnas colapsadas.

Para el tamaño de columnas se ha establecido un tamaño fijo para que el tamaño de las fichas sea lo suficientemente grande para poder ver toda la información con claridad. Cuando se colapsan las columnas, las demás aumentan en tamaño por lo que para ítems que contengan nombres muy largos nos vendrá muy bien.

8.4 Orden de los ítems

Anteriormente se ha comentado que todos los ítems tienen un orden por el cual se ordenaban en las columnas. En la primera carga inicial no hay ningún tipo de problema ya que se ordena la lista de ítems que se debe de mostrar antes de pintarla en pantalla, pero ¿qué pasa cuando se mueve un ítem? Si tratamos de mover una ficha a otro sitio, el orden no se actualizará y esto es algo que no debería suceder, ya que si movemos una ficha más arriba o más abajo de una columna lo que realmente estaremos haciendo es cambiarle su priorización en cuanto a los demás y esto debería verse reflejado en el número de Orden.

Aquí empieza uno de los desafíos del desarrollo, pues ningún método que nos proporciona JQWidgets para el control del tablero nos servía para solucionar este problema. Tras varios intentos fallidos con los métodos proporcionados por JQWidgets, se decidió hacerlo a través del código HTML del tablero, utilizando la técnica de *web scrapping*. Esta técnica sirve para recopilar información automáticamente de la Web. En un momento determinado, se observó que era posible

reconocer que ítem se había movido a través del HTML, pues con solo obtener el número de orden del ítem movido y del superior e inferior de donde ha sido situado, es posible recalcular el nuevo orden. Para el nuevo orden se ha seguido la siguiente fórmula:

$$Nuevo\ orden = \frac{Orden\ ítem\ superior + Orden\ ítem\ inferior}{2}$$

Vamos a ver un ejemplo, para el caso de la Figura 41, si movemos la tarea 5 entre la tarea 4 y la tarea 5, el nuevo orden para ésta debería ser 45.

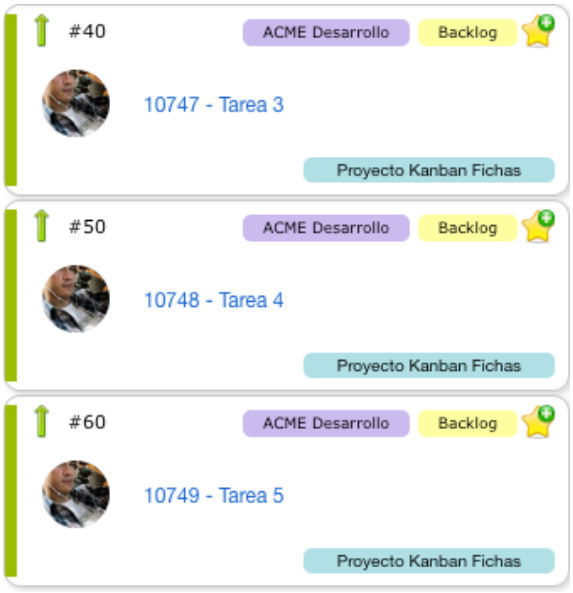

Figura 41. Orden de las fichas en el tablero.

Existen dos casos donde no se sigue la anterior fórmula, cuando se mueve a la primera posición o última. Para estas dos situaciones se sigue la misma norma, volver a priorizar con el orden del ítem superior o inferior dependiendo del caso.

8.5 Drag and drop de los items

Algo esencial en un tablero es la posibilidad de mover las fichas en él, pero ¿es conveniente qué se puedan mover desde cualquier columna a cualquier otra columna? Si movemos de posición una ficha dentro de la misma columna tan solo significara que la estaremos priorizando de nuevo, por el contrario, si movemos una ficha hacia adelante o hacia atrás esto

puede significar que consideremos que se ha acabado cierta activad y se va a continuar con la siguiente, o que necesita re-trabajo. Dotar de cierta facilidad a los usuarios para mover todas las fichas de lugar puede llegar a ser un problema, y teniendo en cuenta que la herramienta Worki ya tiene implementado un control de flujo avanzando para los ítems, utilizar este para el tablero puede ser lo más conveniente.

Por lo tanto, el siguiente paso ha sido deshabilitar el *drag and drop* de las fichas a columnas que no sean la suya. Una vez deshabilitado, el siguiente paso es aplicar el control de flujo de Worki al tablero, para ello, se ha añadido un enlace al nombre de cada ítem para navegar a la pestaña desde donde es posible editar la ficha o controlar su flujo (finalizar la actividad actual, pasar a la siguiente, volver atrás, asignar uno o varios colaboradores, añadir actividades en paralelo, etc.).

8.6 Casos especiales

Como hemos comentado anteriormente, hay funcionalidades para los workflows de Worki que son un poco especiales. Es conveniente mostrar como se han resuelto cada una de ellas.

Para empezar, hay veces que en las líneas de trabajo existen tareas que están en alguna actividad que no pertenece al workflow. Se cree conveniente que estas tareas deben aparecer también en el tablero, y como solución, siempre que exista una tarea de este tipo se añadirá automáticamente al final del tablero una columna auxiliar donde aparecerán todas estas tareas. El nombre de esta columna es Actividades fuera del Workflow. También se ha considerado necesario añadirle a la ficha en este caso que tarea se está realizando sobre ella. Esta nueva columna y nuevo estilo de ficha puede verse en la Figura 42.

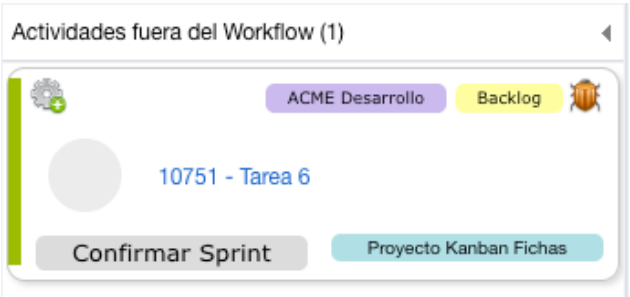

Figura 42. Ficha con actividad fuera del Workflow.

El otro caso especial aparece cuando hay dos miembros del equipo de desarrollo trabajando en la misma tarea, pero en diferentes actividades del workflow. En el tablero se considera conveniente que aparezcan ambas, aunque sea la misma tarea, por lo tanto, la solución ha sido mostrar la misma tarea en las diferentes actividades donde están trabajando los miembros del equipo. En la Figura 43 puede observarse lo comentado anteriormente.

Figura 43. Unidad de trabajo en distintas actividades.

8.7 Como se obtienen los datos

Hasta ahora, se ha estado trabajando con datos que se introducían a mano, es hora de obtenerlos automáticamente. Para obtenerlos, se han añadido arriba del tablero una serie de filtros con los que el usuario puede seleccionar que ítems quiere visualizar en el tablero.
Los filtros añadidos han sido los siguientes:

- **Colaborador**: El usuario podrá selecciona un colaborador en concreto y solo se visualizarán las tareas que estén asignadas a dicho colaborador. Puede estar vacío.

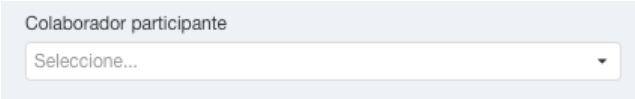

Figura 44. Filtro del colaborador.

- **Línea de trabajo**: Si se selecciona una línea de trabajo solo aparecerán aquellas tareas que pertenezcan a ella. Puede estar vacío.

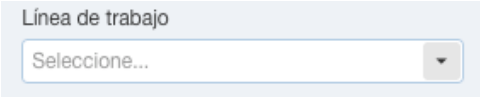

Figura 45. Filtro de la línea de trabajo.

- **Sprint**: Al seleccionar una línea de trabajo, aparecerán los Sprints que pertenezcan a la línea de trabajo con la posibilidad de visualizar las tareas por Sprints. Puede estar vacío.

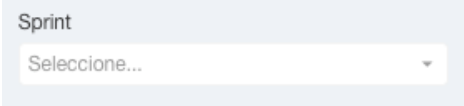

Figura 46. Filtro del Sprint.

- **Proyecto**: Al igual que los Sprints, solo aparecerán aquellos proyectos que pertenezcan a la línea de trabajo seleccionada. Puede estar vacío.

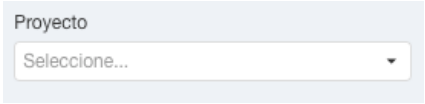

Figura 47. Filtro del Proyecto.

- **Workflow**: Es el único filtro que no puede estar sin seleccionar. Por defecto, aparecerán todos los workflows disponibles en la herramienta Worki, y como en los demás filtros, aparecerán aquellos ítems que pertenezcan al workflow seleccionado.

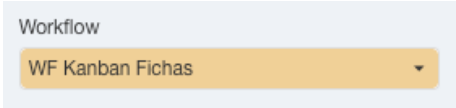

Figura 48. Filtro del Workflow

Estos filtros permiten al usuario visualizar bastante información variada, pero ¿cómo recoge internamente el tablero esta información? Para obtener toda esta información, el tablero hace dos llamadas a la API de Worki, una para recoger la información de las columnas y la otra para obtener todo respecto a cada tarea. Una API, es la abreviatura de interfaz de programación automática. A través de ella es posible recoger datos, actualizarlos, añadir nuevos o incluso eliminarlos. Por lo tanto, una vez se recoge la información se hacen todos los cálculos necesarios y se pinta por pantalla todos los ítems.

La API de Worki está creada a través de Swagger. Swagger es un *framework* para documentar APIs Rest a través de diversas fuentes como pueden ser archivos de configuración, XML, C#, JavaScript, etc. Además, existe la posibilidad de describir, producir, consumir y visualizar las APIs.

La primera llamada a la API sirve para obtener las actividades del workflow que se ha seleccionado en los filtros. La *url* es la siguiente:
http://cliente.tuneupprocess.com/ApiWeb/test/Workflows/{idWorkflow}/Actividades

En la Tabla 5 podemos ver la lista de parámetros de dicha llamada con una breve descripción, ejemplo y si son necesarios.

Parámetro	Descripción	Necesario	Ejemplo
idWorkflow	Id del workflow del cual se quieren obtener las actividades	Sí	1

Tabla 8. Tabla con los parámetros de la llamada a la API

La otra llamada a la API se hace para obtener las actividades pertenecientes a dicho workflow, teniendo en cuenta también lo demás filtros seleccionados. La *url* es la siguiente:
http://cliente.tuneupprocess.com/ApiWeb/test/Uts/DetalleKanban?idAgente={idAgente }/idProducto={ idProducto }/idSprint={ idSprint }&idProyecto={ idProyecto }&idActividad={ idActividad }&idWorkflow={ idWorkflow }

Como antes, en la Tabla 6 estarán todos los parámetros de la llamada con una breve descripción, ejemplo y si son necesarios o no.

Parámetro	Descripción	Necesario	Ejemplo
idAgente	Id del Agente que se ha seleccionado en los filtros	No	1
idProducto	Id del Producto seleccionado en los filtros	No	10
idSprint	Id del Sprint seleccionado	No	2
idProyecto	Id del Proyecto seleccionado en los filtros	No	5
idWorkflow	Id del Workflow al cual pertenecen las tareas	No	50

Tabla 9. Tabla con los parámetros de la llamada a la API

8.8 Estructura del código

Todas las aplicaciones web que estén desarrolladas con Angular están formadas por componentes. Los componentes son pequeñas partes lógicas que controlarán un trozo de pantalla o de vista. Todo lo que aparece en pantalla está controlado y gestionado por estos elementos. Nuestro tablero kanban está dentro de un componente y los filtros de selección son otro componente. Como podemos ver en la Figura 49, estos dos se comunican, es decir, el componente de los filtros cada vez que un filtro se cambia, le pasa al componente del tablero todos los valores y este hace las llamadas a la API para obtener los datos y mostrar el tablero kanban. Por lo tanto, el componente del tablero se comunica con la API de Worki y de ella recibe los datos, sobre los cuáles se hacen todas las transformaciones necesarias para formar el tablero de la forma adecuada. Además, existe una comunicación adicional, cada vez que se accede al perfil de una unidad de

trabajo (UT) a través del tablero kanban haciendo *click* sobre el nombre de la UT, el componente del tablero se comunica con el componente de las UT para decirle que UT debe mostrar.

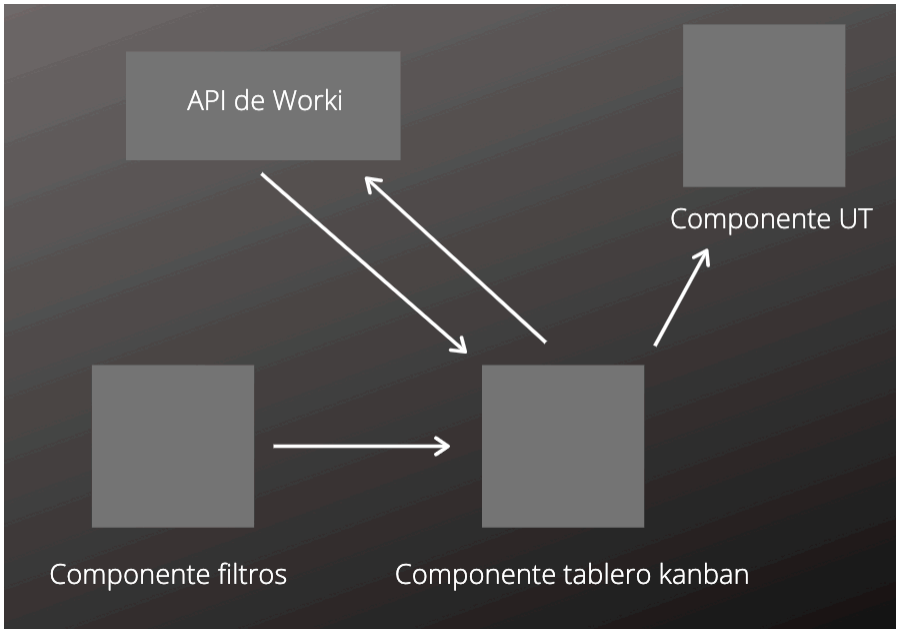

Figura 49. Esquema de la comunicación del código.

8.9 Resultado final

Finalmente, una vez está todo integrado con la herramienta, lo último que se ha modificado han sido los estilos de la letra, tamaño y fuente para que concuerden con los de la herramienta Worki. También se han añadido *scrollbars* al tablero kanban idénticos a los que se utilizan en Worki para dar la sensación de consistencia en toda la página web. El resultado final obtenido lo vemos en la Figura 50:

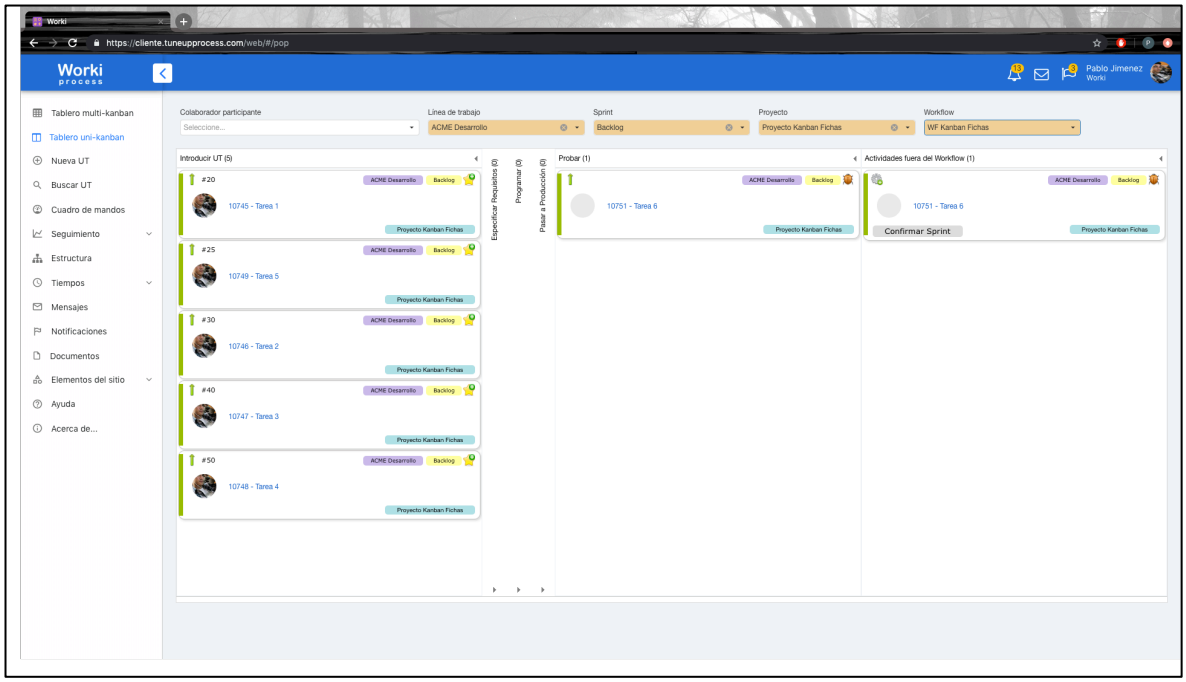

Figura 50. ***Resultado final del tablero.***

9. Conclusiones y trabajo futuro

Se ha implementado una satisfactoria integración de tableros kanban en la herramienta Worki. Se han desarrollado las funcionalidades necesarias para el tablero kanban manteniendo la coherencia y compatibilidad con la representación tabular previamente existente en Worki. Además de haber añadido aquellas restricciones que se han considerado necesarias para que no haya una total libertad sobre él. El conocimiento previo que tenía el autor del TFG sobre el *framework* Angular ha resultado de gran ayuda para el desarrollo del tablero. Antes de integrar el tablero kanban al proyecto de Worki se realizaron pruebas para comprobar su correcto comportamiento. La integración del código del tablero kanban en el proyecto de Worki fue probablemente la parte más difícil del trabajo.

Cuando se integró el tablero en Worki se detectaron fallos, cuando se hicieron pruebas con casos reales, sobre todo en las situaciones especiales de los workflows de Worki que se han comentado en capítulos anteriores. La elección de un buen componente como JQWidgets, ha sido clave, pues todo ha resultado mucho más fácil a la hora de implementar los detalles con los que cuenta el tablero kanban. Tras realizar pruebas con resultado satisfactorio, se ha dado como terminado el desarrollo del tablero, al menos en lo que es una primera versión. El tablero kanban ya se está utilizando desde principios de Julio en implementaciones reales, con buenos resultados en todas ellas.

Por ejemplo, en un futuro, la posibilidad de añadir ítems directamente desde el tablero puede que resulte necesario y así evitar trabajo de más a los miembros del equipo. También cabe la posibilidad de modificar el *drag and drop* de los ítems, es decir, habilitar el *drag and drop* de los ítems para que sea posible mover cada ítem a todas las columnas del tablero.

La Figura 51 ilustra la cronología del trabajo asociado a este TFG. El trabajo se empezó a realizar en enero de 2019, con el estudio de los componentes para ver con cual se iba a realizar el desarrollo del tablero kanban para Worki. Una vez se eligió el componente, se empezó a desarrollar una versión preliminar del tablero en un servidor local. A partir de aquí se investigó qué funcionalidades debía de ofrecer el tablero y cuáles podían no ser desarrolladas. Una vez el tablero estaba completo se realizaron pruebas en el servidor local y así poder empezar con la integración del tablero en Worki.

Uno de los mayores riegos que teníamos respecto a la integración era el problema de añadir nuevas dependencias a un proyecto Angular ya construido, ya que era necesario instalar todas las dependencias de JQWidgets para poder hacer uso de su componente kanban. Sorprendentemente no resultó ser muy complicado por lo que tan solo faltaba trasladar el proyecto donde se encontraba el tablero kanban desarrollado al proyecto de Worki. Una vez todo estaba integrado y en funcionamiento, se empezó a utilizar en Worki.

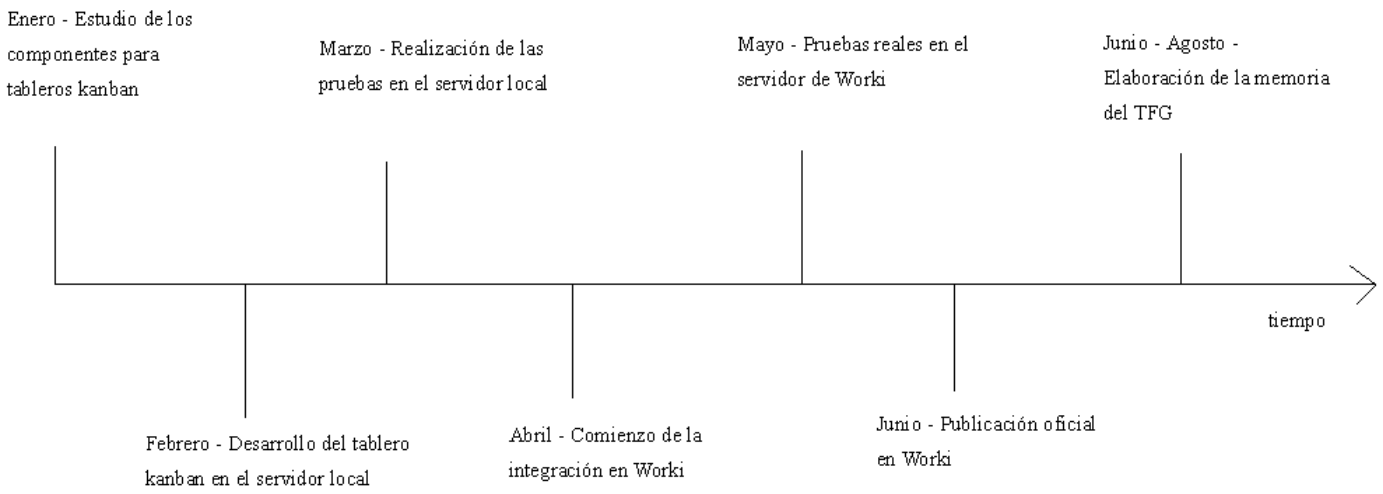

Figura 51. Cronología del trabajo durante 2019

Personalmente, la realización de este trabajo me ha ayudado a aprender como realizar un proyecto desde cero. Además, he aprendido conceptos sobre las metodologías ágiles que antes desconocía, aparte de entender cómo funciona el método Kanban. Por supuesto, he aprendido como desarrollar una memoria mucho más formal de las que había realizado anteriormente. También he podido desarrollar una buena gestión del tiempo para poder realizar todo el trabajo dentro de los plazos acordados para el TFG.

Hay diferentes asignaturas que me han servido de ayuda para realizar el trabajo, entre ellas: Integridad e Interoperabilidad (IEI), Proyecto de Software (PSW), Programación (PRG) o Proyecto de Ingeniería del Software (PIN).

Referencias

[1] A., J. C. (12 de Enero de 2018). *Johana Chuquino*. Obtenido de: http://johanachuquino.com/desarrollo-software-secuencial-agile/ , Consultado en Julio de 2019

[2] Acevedo, J., Urquiaga, A., & Gómez, M. (2001). *Gestión de la Cadena de suministro. Centro de estudio de Tecnología de Avanzada (CETA) y Laboratorio de Logística y Gestión de la Producción (Logespro).* Ciudad Habana.

[3] Documentación de Angular. Obtenido de https://angular.io

[4] Arango Serna, M. D., Campuzano Zapata, L. F., & Zapata Cortes, J. A. (2015). Mejoramiento de procesos de manufactura utilizando Kanban. *Revista Ingenierías Universidad de Medellín, 14*(27), 221-234.

[5] Arrizabalaga, I. (15 de Enero de 2019). *Axia Team*. Obtenido de https://www.axiateam.com , Consultado en Julio de 2019

[6] Imágenes de DayPilot. Obtenido de DayPilot: https://www.daypilot.org , Consultado en Julio de 2019

[7] Imágenes de DlhSoft. Obtenido de DlhSoft: https://dlhsoft.com , Consultado en Julio de 2019

[8] G. Figueroa, R., J. Solis, C., & A. Cabrera, A. (s.f.). Metodologías Tradicionales vs. Metodlogías Ágiles.

[9] H. Canós, J., Letelier, P., & Penadés, M. C. (2003). Metodologías Ágiles en el Desarrollo de Software. 1-8.

[10] Imágenes de HostProton. Obtenido de https://hostproton.com , Consultado en Julio de 2019

[11] Imágenes de Jira Software. Obtenido de Atlassian: https://www.atlassian.com/es/software/jira , Consultado en Julio de 2019

[12] Imágenes de JQWidgets. Obtenido de JQWidgets: https://www.jqwidgets.com , Consultado en Julio de 2019

[13] Kniberg, H., & Skarin, M. (2010). *Kanban y Scrum – obteniendo lo mejor de ambos.* C4Media Inc.

[14] Leiva, A., Sanchez, H., & Navarta, C. (17 de Octubre de 2017). Metodologías y Herramientas para los Sistemas de Gestión de Proyectos de Cursos de posgrado Presencial, Semi-presencial y a Distancia.

[15] Mitra, D., & Mitrani, I. (1990). Analysis of a Kanban discipline for cell coordination in production lines. *I, Manamegement Science, 36*, 1548-1566.

[16] Navarro Cadavid, A., Fernández Martínez, J. D., & Morales Vélez, J. (2013). Revisión de metodologías ágiles para el desarrollo de software. *PROSPECTIVA, 11*(2), 30-39.

[17] Parra Ortega, O. J. (s.f.). sistemas de producción tipo kanban: Descripción, componentes, diseño del sistema, y bibliografía relacionada. *Panorama*, 11-22.

[18] Pérez Pérez, M. J. (2012). Guía Comparativa de Metodologías Ágiles. Valladolid, España.

[19] Imágenes de Pivotal Tracker. Obtenido de Pivotal Software: https://www.pivotaltracker.com , Consultado en Junio de 2019.

[20] Raymond S, L. (2006). *Custom Kanban: Designing the System to Meet the Needs of Your Environment.* Productivity Press.

[21] Rubio, M. (22 de Junio de 2009). *Altenwald*. Obtenido de https://altenwald.org/ , Consultado en Junio de 2019.

[22] Imágenes de Syncfusion. Obtenido de Syncfusion Inc.: https://www.syncfusion.com , Consultado en Junio de 2019.

[23] Imágenes de Tagertprocess Cyprus. Obtenido de Targetprocess software: https://www.targetprocess.com , Consultado en Junio de 2019.

[24] Tezel, A., Koskela, L., & Tzortzopoulos, P. (2009). The functions of visual management. Salford.

[25] Tomaselli, G. P., Acuña, C. J., Estayno, M., & Lenkovich, C. (s.f.). SCRUM: Una revisión de la literatura.

[26] Gutiérrez Plaza, J., Borillo Doménech R. (2012) 2ª Conferencia Agile-Spain. Universitat Jaume I. Servei de Comunicació i Publicacions. 51-55

[27] Álvarez Diz, I. (2016). Kanban o cómo ser flexibles, pero poniendo límites. QA: NEWS.

[28] Imágenes de Scrum Manager. Obtenido de Scrum Manager: https://www.scrummanager.net , Consultado en Junio de 2019.

[29] Imágenes Gestión de proyectos (2012). Obtenido de Gestión de Proyectos: http://www.gestiondeproyectosit.es , Consultado en Junio de 2019.

[30] Imágenes de Kanban tool. Obtenido de Kanban tool: https://kanbantool.com , Consultado en Junio de 2019.

[31] Rubio, M. (2009). Kanban: El Método Toyota aplicado al software. Altenwald.

[32] Letelier, P. (2011). Workflows flexibles. Parte I:Reconociendo workflows en el proceso de desarrollo de software. Agility At Work. Obtenido de: http://agilismoatwork.blogspot.com/search/label/Workflows , Consultado en Junio de 2019.

[33] Letelier, P. (2011). Workflows flexibles. Parte II: Característica requeridas por workflows flexible. Agility At Work. Obtenido de: http://agilismoatwork.blogspot.com/search/label/Workflows , Consultado en Junio de 2019.

[34] Letelier, P. (2015). Tableros kanban. Parte I: Diseño básico para gestionar flujo de trabajo. Agility At Work. Obtenido de: http://agilismoatwork.blogspot.com/search/label/Workflows , Consultado en Junio de 2019.

[35] Letelier, P. (2015). Tableros Kanban. Parte II: Diseño de tablero kanban para aplicarlo con Scrum (desarrollo usando Sprint). Agility At Work. Obtenido de: http://agilismoatwork.blogspot.com/search/label/Workflows , Consultado en Junio de 2019.

[36] State of Agile report. Obtenido de: https://agilebb.nl/wp-content/uploads/2018/04/versionone-12th-annual-state-of-agile-report.pdf , Consultado en Julio de 2019

[37] Anderson, J. David (2010). Kanban: Successful Evolutionary Change for Your Technology Business.