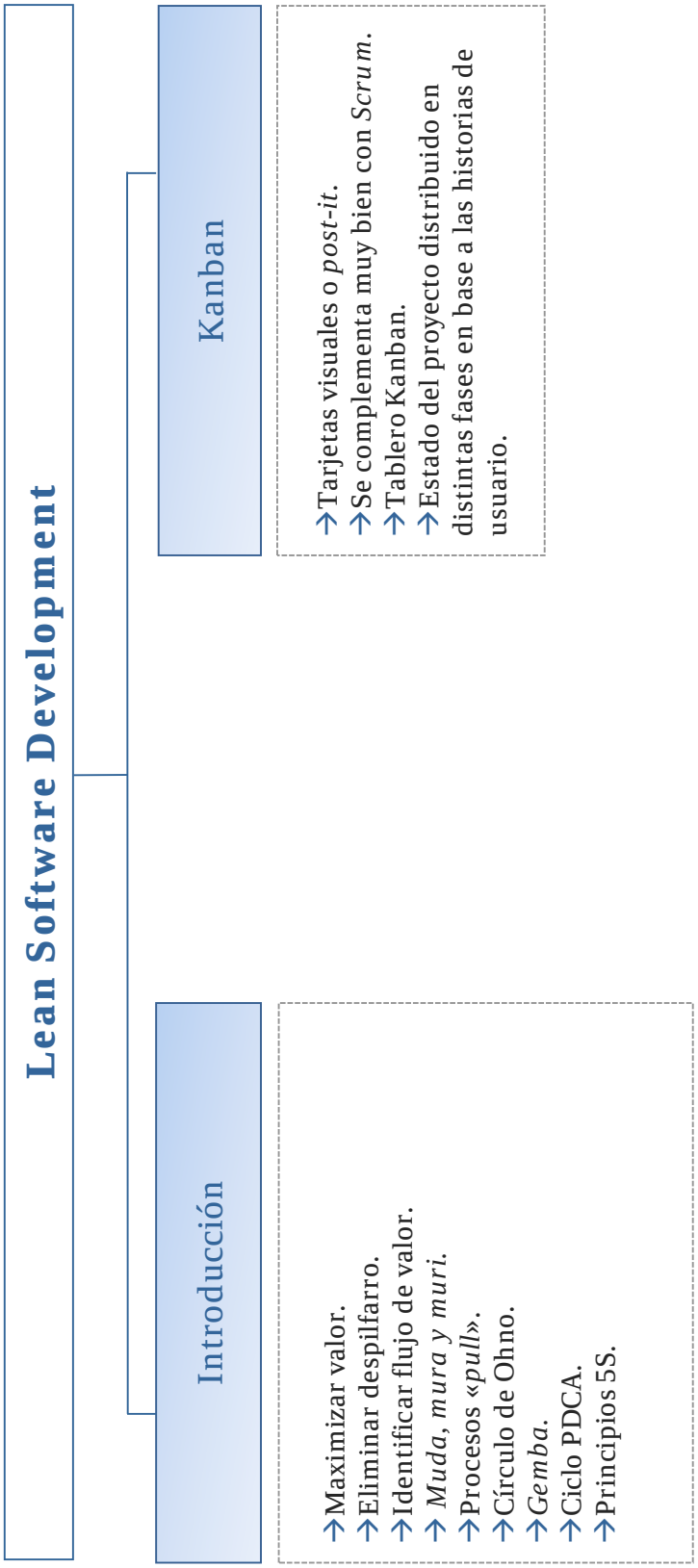
Lean Software Development

TEMA 11

Esquema



Ideas clave

11.1. ¿Cómo estudiar este tema?

Para estudiar este tema lee las **Ideas clave** que encontrarás a continuación.

En este tema se estudian los principios de lo que se conoce como *Lean Software Development* y su aplicación para el desarrollo de *software*.

11.2. Introducción

De acuerdo a *Lean Enterprise Institute* (LEI), ***lean*** es un término que se utiliza para tomar conciencia de que todo negocio ha de **maximizar el valor que se le da al cliente** a la vez que **elimina** lo que en *lean* se denomina **despilfarro o desperdicio** (**del japonés *muda* y del inglés *waste***), es decir, se intenta reducir los recursos necesarios para dar ese valor al cliente.

Por tanto, de acuerdo a *lean*, el objetivo final de toda organización ha de ser el de seguir un proceso de creación de valor perfectamente definido que no presente ningún tipo de despilfarro y que permitirá obtener la excelencia en el valor que se proporciona a cada cliente. La identificación del **flujo de valor** para el cliente es uno de los desafíos de *Lean*. Womack y Jones (2012) definen el flujo de valor como el conjunto de acciones necesarias que harán pasar el producto de la organización por tres tareas que toda organización considera críticas:

- **Solución de problemas.** Esta tarea se inicia en la fase de concepción del producto, sigue durante el diseño detallado e ingeniería del producto y finaliza en su producción.
- **Gestión de la información.** Esta tarea comienza con la recepción del pedido hasta su entrega.
- **Transformación física.** Esta tarea abarca desde la materia prima del producto hasta su acabado, cuando se entrega al cliente.

La eliminación del despilfarro (*muda*) es un objetivo fundamental del pensamiento *lean*. El término *muda* engloba a su vez siete tipos diferentes de despilfarro (Poppendieck y Poppendieck, 2003):

- **Sobreproducción**. Este despilfarro hace referencia a la producción temprana o excesiva con respecto a lo que va a demandar el cliente. La sobreproducción, además puede afectar a otros tipos de despilfarro, como por ejemplo, un exceso de inventario.
- **Inventario**. Este despilfarro, consecuencia directa del anterior, conduce a un exceso de materia prima, *work in progress* (WIP), o productos terminados que no cuentan con un valor añadido para el cliente. Este tipo de despilfarro suele causar obsolescencia, daños sobre los productos, transporte y almacenamiento debido principalmente por previsiones imprecisas, malos suministradores, excesivos tiempos muertos y crear productos sin que los demande el cliente.
- **Procesamiento extra**. Este tipo de despilfarro se debe al exceso de trabajo sobre un producto que hace que supere los estándares del cliente y que hace, por tanto, que ese extra de procesamiento sobre el producto sea innecesario ya que al cliente no le supone un valor añadido.
- **Movimiento.** Este despilfarro de debe a movimientos excesivos por parte de los empleados para poder llevar a cabo sus tareas como por ejemplo, las herramientas que deben utilizar se encuentran demasiado lejos, el trabajo se ralentiza por falta de formación o por no saber a dónde ir a preguntar por una mala organización de la empresa. Este tipo de despilfarro hará que se interrumpa el flujo de producción y en consecuencia el tiempo de producción se verá incrementando
- **Transporte.** Este tipo de despilfarro se debe a movimientos innecesarios entre partes de distintos procesos. Este tipo de despilfarro también incrementa el tiempo de producción, así como el *work in progress*, lo que puede afectar incluso a la calidad final del producto.
- **Esperas.** Este tipo de despilfarro se debe a la espera por parte de los empleados para disponer de la maquinaria o de las piezas necesarias para realizar su trabajo.
- **Defectos.** Este tipo de despilfarro se debe a la fabricación de productos con defectos que no van a pasar el control de calidad. Este tipo de despilfarro hace que aumenten los costes ya que será necesario rectificar el producto para que obtenga la calidad esperada.

Aparte del despilfarro, clasificado en TPS en su término japonés como *muda*, se identifican también en *lean* otros dos tipos de desperdicios que se deberían eliminar y que están asociados a los procesos de las organizaciones:

- ***Mura* (irregularidad)**. Este término hace referencia a la poca homogeneidad y a la inconsistencia que se da en los resultados obtenidos en lo que respecta a calidad y volumen de los productos fabricados.
- ***Muri* (trabajo tensionante).** Este término hace referencia a trabajos que se imponen debido a una mala organización en la empresa, tales como, la realización de tareas peligrosas, hacer trabajar más rápido de lo habitual, llevar cargas físicas muy pesadas... *Muri* también hace referencia a las malas condiciones del puesto de trabajo, lo que afectará al rendimiento y buen funcionamiento de la organización.

Conviene aclarar que *lean* no es ninguna metodología tal y como se entiende en el sentido convencional, por lo que no va a decir qué es lo que se debe hacer exactamente para conseguir la excelencia en el valor que se le da al cliente. El término *lean* se refiere más bien a la síntesis de un conjunto de principios y a un pensamiento o filosofía a seguir a la hora de construir productos de cualquier tipo. Algunos de los principios más importantes que recoge *lean* son los que se indican a continuación (Burlton y Matthijssen, 2013) y (Garzás, Enríquez de S. e Irrazábal, 2013):

- **Valor que se da al cliente y calidad.** Definir qué es lo que representa valor para el cliente y centrarse en las actividades que proporcionan dicho valor. En este proceso, se procurará eliminar todos los defectos, así como detectar y solucionar todos los problemas lo más pronto posible.
- **Eliminación del despilfarro**. Las organizaciones y los procesos que tienen implantados están llenos de despilfarro, por lo que hay que eliminar todo aquello que resulta prescindible y que no aporta valor al cliente (sobreproducción, tiempos de espera, inventarios, transportes innecesarios...).
- **Mejora continua.** Todos los procesos han de seguir un flujo continuo, es decir, han de estar en movimiento, ser dinámicos, para que se pueda añadir valor de una manera continuada. Este principio de *lean* conocido como *Kaizen* parte de la idea de que todo es susceptible de ser mejorado. En *lean* se asume que la perfección no existe, pero hay un esfuerzo de superación continua para intentar alcanzarla. Por ello, todos los esfuerzos del personal involucrado en los procesos se ven dinamizados para que los procesos puedan mejorar con el mínimo coste posible y sin poner en riesgo la estrategia de la organización, así como su competitividad en el mercado.

- **Procesos «*pull*»**. Producir solamente lo necesario, de acuerdo a lo que el cliente necesita evitando bloquear recursos (personas, infraestructura...) cuando su demanda no es inmediata. El concepto «*pull*» se basa en el hecho de que el volumen de producción ha de ser acorde a la demanda del cliente (es decir, necesidades reales donde el cliente es el motor de la producción) y no construir producto a la espera de que el cliente lo compre (procesos «push»).
- **Flexibilidad.** Capacidad para producir de una manera rápida diferentes tipos de productos manteniendo la eficiencia a pesar de que el volumen de producción se vea reducido.
- **Relación a largo plazo con los proveedores.** Firmar acuerdos con los proveedores en los que se permita compartir información y asumir los riesgos de los costes.

Aunque el concepto de *lean* se suele considerar sinónimo del método de fabricación de la empresa automovilística Toyota, conocido como *Toyota Production System* (TPS), realmente este término no fue acuñado por los japoneses, sino por tres autores estadounidenses, James P. Womack, Daniel T. Jones y Daniel Roos, que en los años 90 utilizaron el término *lean* para describir en su libro *The Machine That Changed the World* (Womack, Jones y Roos, 2007) **el proceso de negocio que seguía esta empresa japonesa desde los años 50**.

En este sentido, resulta interesante comentar también que, para su aplicación en Japón, se fijaron en el modelo de producción en masa de vehículos establecido por Henry Ford (ver Figura 1) con el que consiguió que sus coches estuvieran al alcance de más gente bajando el tiempo y los costes de producción, en base a una cadena de montaje más eficiente, ordenada y con piezas estandarizadas fabricadas en serie tal y como se ilustra en la Figura 2.

Figura 1. Henry Ford posa con uno de los vehículos resultado de su modelo de producción, el Ford T. Fuente: http://www.eleconomistaamerica.com/ .

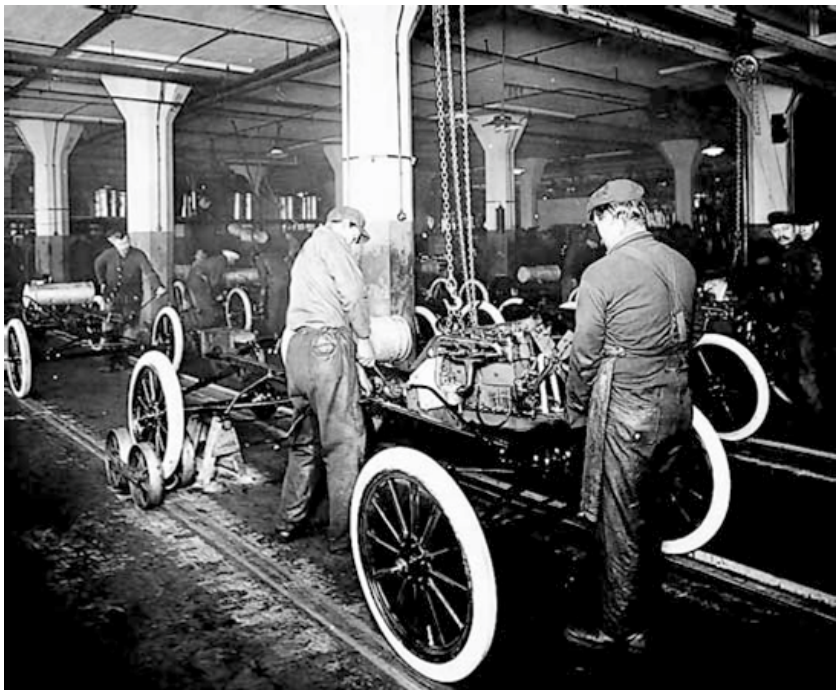


Figura 2. Cadena de montaje en la fabricación de un vehículo Ford. Fuente: http://www.biografiasyvidas.com/monografia/ford/fotos4.htm .

Taiichi Ohno (ver Figura 3) fue el creador del sistema de producción de Toyota, TPS, y autor del libro *Toyota Production System: Beyond Large-Scale Production* (Ohno, 1988). La estrategia definida por Taiichi Ohno para construir un sistema de producción libre de despilfarro y reducir por tanto los gastos de producción, estaba basada en tres principios fundamentales: (i) construir estrictamente lo necesario; (ii) eliminar todo aquello que no genere valor para el cliente; y (iii) si algo no va bien, parar la producción y solucionar los problemas.

Figura 3. Taiichi Ohno. Fuente: http://www.gauzak.eu/taiichi-ohno/ .

Aparte de estos tres principios de Taiichi Ohno, que constituyen lo que sería la base del pensamiento *lean*, existe otro elemento también de especial relevancia en *lean*, y es lo que se conoce como el círculo de Ohno. El círculo de Ohno debe su nombre a que Taiichi Ohno solía dibujar con tiza un círculo en el suelo y permanecía allí, dentro del círculo, durante un tiempo determinado para dedicarse exclusivamente a observar el sistema de producción.

De esta manera descubría qué era lo que no se estaba haciendo bien o era susceptible de mejora y poder actuar así en consecuencia, solucionando los problemas detectados o mejorando el proceso. He aquí el gran poder de la observación, las ventajas que ofrece el ir y observar el lugar donde realmente se fabrica el producto y ver cómo se trabaja, implicando incluso a la alta dirección, como era este caso, ya que Taiichi Ohno era el vice-presidente de Toyota (de ahí la importancia de que la alta dirección de la organización también se involucre si se desea que el pensamiento *lean* pueda ser llevado a cabo con éxito).

Esta visión de ir al lugar donde realmente se fabrica el producto y poder entender la forma de trabajar es la que recoge el término japonés *Genchi Genbutsu*, donde *Genchi* (referido normalmente como *Gemba*, aunque su significado no es exactamente el mismo) significa «ubicación real» y *Genbutsu* significa «productos reales». Por tanto, en todo proceso de mejora de una organización habrá que ir al *Gemba*, es decir, al lugar real como promueve el círculo de Ohno, y observar el modo de trabajar, tal y como se ilustra en la Figura 4.

Figura 4. Ejemplo de aplicación del círculo de Ohno. Fuente: http://gembapantarei.com/2007/07/top_10_improvement_tools_named_after_lean_sensei/ .

Sin embargo, aunque todo en *lean* parece ideal, la solución perfecta y más o menos sencillo de implementar, conviene resaltar que son muchas las empresas en todo el mundo que fracasan en su intento de implantar *lean* por seguir un camino equivocado. La resistencia al cambio, el giro de 180º que muchas organizaciones han de realizar en su modo de trabajar, con todo el volumen de trabajo adicional que esto conlleva hasta que empieza a funcionar y el esfuerzo para la mejora continua que nunca acaba (Toyota, por ejemplo, a pesar de su TPS ya implantado, sigue en un ciclo de mejora de sus procesos para poder hacer las cosas mejor a como lo han venido haciendo hasta ahora), pueden hacer que la cultura *lean* acabe fracasando estrepitosamente.

Burlton y Matthijssen (2013) establecen cuatro factores de éxito que se han de considerar a la hora de implantar esta cultura en una organización:

- **Compromiso por parte de la dirección.** Como ya se destacaba con anterioridad, la dirección debe asumir el camino que va a iniciar cuando se decide a adoptar *lean*. En este aspecto, la dirección ha de estar dispuesta a realizar cambios profundos en su forma de hacer las cosas, especialmente a nivel de dirección, donde se requiere más involucración, colaboración y deseo de compartir y sin perder en ningún momento la estrategia de negocio de su organización.
- **Focalización en el valor que se le da al cliente y la calidad**. La organización ha de saber qué es exactamente lo que para el cliente representa valor y qué no lo es, donde la eficiencia y la calidad en el trabajo debe primar sobre los costes requeridos. En este sentido, hay que tener en cuenta que siempre que se haga lo que desea el

cliente, es decir, los productos se hacen bien (del inglés *the right product*), que no viene a decir lo mismo que estar bien hechos (del inglés *the product right*), la organización se ahorrará dinero con total seguridad.

- **Implicación de todo el personal que hace el trabajo real**. Cuando la dirección realice cambios en la forma de trabajar, en la unidad organizativa de la empresa... deberá tener siempre en cuenta, involucrar y mantener informadas, a todas las personas que realmente están realizando el trabajo de fabricación de los productos, ya que son quienes realmente tienen el mayor conocimiento sobre qué es lo que puede aportar valor al cliente y dónde está el despilfarro. Los cambios a realizar para adoptar una cultura *lean* no se deben imponer a las personas sin ningún tipo de justificación ni de explicación. Si todos los afectados por la adopción de *lean* entienden el objetivo a alcanzar, hay respeto, reconocimiento y van todos juntos en el mismo camino, la probabilidad de conseguir el éxito se ve gratamente incrementada.
- **Organización por procesos**. De cara a poder implantar de una manera exitosa el pensamiento *lean* en la forma de trabajar, la definición correcta de los procesos es un requisito imprescindible. Sin embargo, en la mayoría de las empresas, especialmente en aquellas dedicadas a proporcionar servicios, los *workflows* de los procesos de negocio brillan por su ausencia. De acuerdo a Sharp y McDermott (2001): «el flujo de información y de control del proceso de negocio tiene que estar definido a través de un *workflow*», donde como *workflow* se puede entender «la especificación de una secuencia ordenada de actividades llevadas a cabo en una organización» (Eshuis y Wieringa, 2001).

  De la misma manera, para Marshall (2000), todo proceso de negocio constituye un *workflow* tratado en forma de escenario. Es decir, un escenario va a mostrar el flujo de trabajo que se da en el proceso de negocio. Por tanto, se puede deducir, bajo esta perspectiva, que un *workflow* sería de manera simplificada un modelo de un proceso de negocio concreto que define el qué (objetivo del proceso), cómo (actividades), quién (responsables) y cuándo (orden) de un proceso de negocio real.

  De una manera más formal, la organización *Workflow Management Coalition* (WfMC) define un *workflow* como, «la automatización total o parcial de un proceso de negocio (o proceso, de forma general), en el cual los documentos, la información y las actividades irán pasado de un participante a otro del proceso para que se realicen acciones de acuerdo a un conjunto de reglas procedimentales» (WfMC, 1999).

Burlton y Matthijssen (2013) resumen todos estos aspectos que se han mencionado con anterioridad vinculados a *lean* en lo que los autores denominan *Casa de Lean* (del inglés *House of Lean*), tal y como se muestra en la Figura 5. De acuerdo a la Figura 5, lo que primero necesita una organización es una base sólida donde se puedan utilizar e integrar a largo plazo las técnicas basadas en *lean*.

Las técnicas que permiten eliminar el despilfarro, así como las que mejoran el flujo de valor de la organización son importantes para aumentar el valor que se le proporciona al cliente, pero no únicas, ya que técnicas orientadas a visualizar los procesos, garantizar la calidad, medir el rendimiento y mantener una comunicación continua, de cara a mejoras, también son muy importantes.

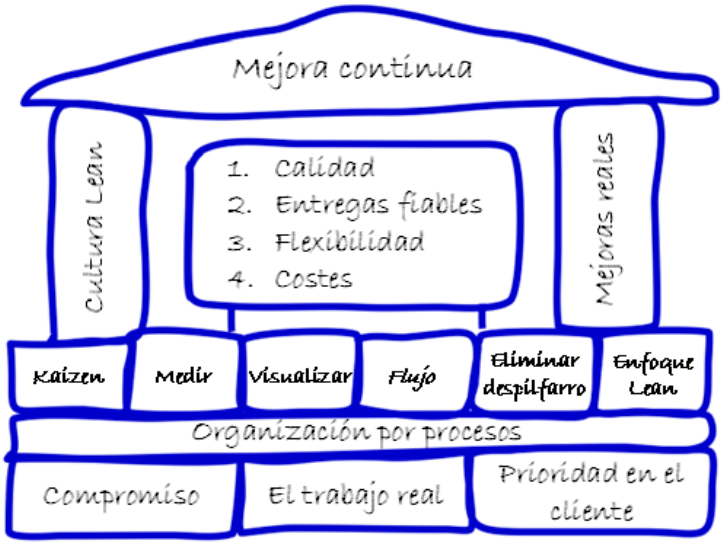


Figura 5. *Casa de Lean* (Burlton y Matthijssen, 2013).

Una técnica muy vinculada al pensamiento *lean* y que se utiliza en las organizaciones para la resolución de problemas es lo que se conoce como ciclo de mejora continua de *Deming* o *Plan-Do-Check-Act* (PDCA), tal y como se ilustra en la Figura 6. El ciclo de *Deming* o PDCA fue introducido en Toyota por el autor que le dio nombre, el Dr. W. Edwards Deming, un estadístico norteamericano que definió los siguientes pasos como método de mejora en la fabricación de productos:

- ***Plan* (planificar).** En esta etapa se ha de responder a la pregunta: ¿qué se debe hacer, cuándo, quién debe hacerlo, cómo y utilizando qué?

- ***Do* (hacer).** En esta etapa se llevan a cabo las tareas programadas y se documentan las acciones realizadas.
- ***Check* (verificar).** En esta etapa se determina si las tareas llevadas a cabo han dado los resultados esperados y se documentan las conclusiones.
- ***Act* (actuar).** En esta etapa se ajustan los planes basándose en toda la información recogida y se documenta el proceso.

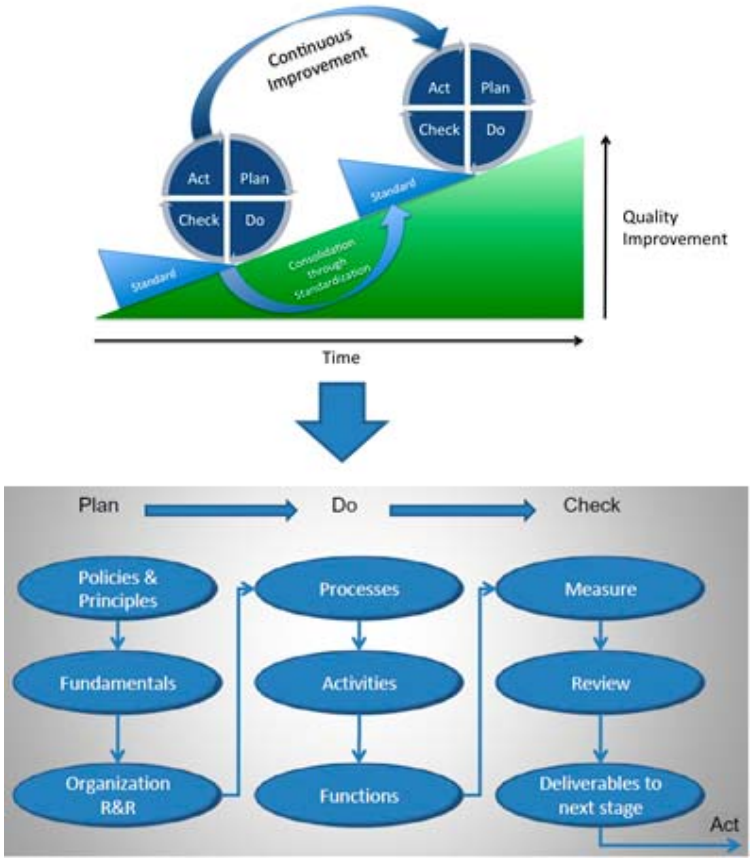


Figura 6. PDCA (fuente: Wikipedia).

Otra visión de PDCA ilustrada por un erudito de Lean, Jeffrey K. Liker, autor del libro *The Toyota Way: 14 Management Principles from the World's Greatest Manufacturer* (Liker, 2004), es la que se muestra en la Figura 7.

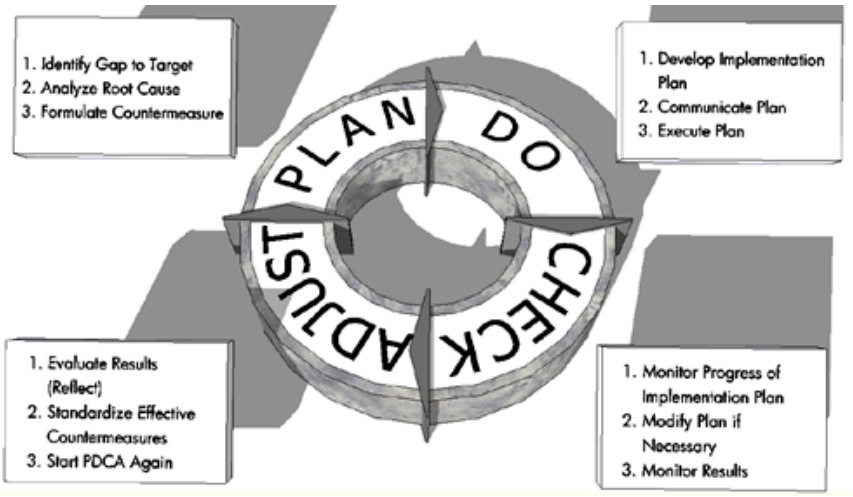


Figura 7. Resolución de problemas con PDCA bajo el pensamiento Lean. Fuente: http://www.sodertalje.se/mainupload/dokument/Kommun%20o%20demokrati/Om%20kommunen/Lean%20i%20S%C3%B6dert%C3%A4lje/Toyota%20Way%20and%20Services%20revised.pdf

Aparte del enfoque de mejora continua en el proceso de creación de los productos, existe otro enfoque también de especial interés para esta asignatura, también de origen japonés, conocido como *Total Productive Mainteinance* (TPM) que se encarga de la gestión de la calidad durante el proceso de mantenimiento de los productos que fabrica una organización. Uno de los pilares de TPM es lo que se conoce como los **principios 5S**. Los principios 5S definen una serie de disciplinas que se han de seguir cuando se mantiene un producto, o de cara a su mantenimiento (Martin, 2009):

- ***Seiri* (organización en el sentido de ordenación).** Este principio establece que una organización ha de saber en todo momento dónde se encuentra cada cosa, es decir, que se coloque en el lugar que le corresponde.
- ***Seiton* (orden en el sentido de sistematización).** Muy vinculado al anterior, este principio establece que hay que definir un lugar para cada cosa y cada cosa ha de estar en su lugar (del dicho norteamericano *A place for everything, and everything in its place*).
- ***Seiso* (limpieza).** Este principio establece que el puesto de trabajo ha de mantenerse limpio y libre de cables, basura, chatarra...
- ***Seiketsu* (estandarización).** Relacionado con el anterior, este principio establece que la organización determina qué es lo que se va a hacer para mantener el puesto de trabajo limpio.
- ***Shutsuke* (disciplina).** Este principio establece que hay que tener la disciplina de seguir todas las prácticas, estándares, etc. que se hayan definido en la organización para realizar el trabajo y tener la voluntad de cambiar en la forma de trabajar si la

aplicación de las prácticas, guías, etc. reflejan que no se estaba haciendo bien y están especificando una nueva forma de llevarlo a cabo.

De lo explicado hasta ahora, resulta inevitable relacionar todo lo que establece *lean* a la *ingeniería de software* (eliminar despilfarro, observar el lugar de trabajo, para detectar errores y proponer mejoras, principios 5S...). Por ejemplo, los principios 5S aplicados al desarrollo de *software* podrían tener la siguiente interpretación (Martin, 2009):

- ***Seiri* (organización en el sentido de ordenación).** Aplicado al desarrollo de *software*, se podría seguir, por ejemplo, con una alta cohesión de cada componente y un nombrado coherente de los elementos que forman parte del código, para facilitar su posterior localización en caso de cambios en las funcionalidades del sistema.
- ***Seiton* (orden en el sentido de sistematización).** Aplicado al desarrollo del *software*, en este caso, se referiría a que cada porción de código debe localizarse en el lugar que se le asignó, donde el programador espera encontrarlo, y en caso de no ser así entonces hay que refactorizar el código para su reubicación.
- ***Seiso* (limpieza).** Aplicado al desarrollo de *software* se focalizaría, por ejemplo, en eliminar todos los comentarios del código que sean redundantes y que no sirvan para entender la *historia* que se cuenta a través del código, que nos explica cómo está implementado el sistema.
- ***Seiketsu* (estandarización).** Aplicado al desarrollo del *software* se trataría de que todo el equipo de desarrollo lea y siga los libros de referencia que se hayan establecido como estándares para el desarrollo del *software* de la organización (guías de estilo, buenas prácticas en construcción de *software*...).
- ***Shutsuke* (disciplina).** Aplicado al desarrollo de *software* se trataría, por ejemplo, cuando un programador ha de seguir una norma de estilo para el nombrado de los métodos de las clases del sistema y se da cuenta que es distinto a como lo venía haciendo hasta ahora. El programador no ha de ignorar la norma y debería cambiar todo su código de acuerdo a lo que se describe en el documento correspondiente. Aquí la premisa sería que hacer el código legible es tan importante como hacerlo ejecutable.

Mary Poppendieck y Tom Poppendieck son los autores que, a través de su libro *Lean Software Development*, realmente popularizaron el término *lean* aplicado al desarrollo de *software* (Poppendieck y Poppendieck, 2003). El desarrollo ágil, por ejemplo, tiene también como objetivo eliminar todo el despilfarro posible que suele acompañar el proceso de construcción de *software*. La tabla 1 muestra la similitud del despilfarro en el

desarrollo de *software* con los 7 tipos de despilfarro definidos por *lean* que se han mencionado anteriormente y que están vinculados a la fabricación de productos físicos.

Tabla 1. Equivalencias de tipos de despilfarro entre fabricación y desarrollo de *software* (Poppendieck y Poppendieck, 2003)

Despilfarro en la fabricación	Despilfarro en el desarrollo de *software*
Inventario	Trabajo incompleto
Procesamiento extra	Procesos extras
Sobreproducción	Características extras
Transporte	Intercambio de tareas
Esperas	Esperas
Movimiento	Movimiento
Defectos	Defectos

Siguiendo la tabla 1, el despilfarro en el desarrollo de *software* viene definido de la siguiente manera (Poppendieck y Poppendieck, 2003, 4-8):

- **Trabajo incompleto.** Este despilfarro traerá como consecuencia que el *software* que está sin terminar se quede obsoleto y haya que replantearse un nuevo desarrollo para el sistema en su totalidad, rehaciendo lo que ya estaba implementado.
- **Procesos extras**. En el desarrollo de *software* se suele generar mucha documentación en papel que hay que entregar al cliente y que realmente no le aporta ningún valor (tampoco resulta de valor para el desarrollo del sistema en sí), pero que simplemente se entrega al cliente porque así está definido, como entregables (del inglés *deliverable*), en los pliegos del contrato. Por tanto, hay que valorar qué documentación exactamente tiene valor para el cliente de cara al proyecto, y sobre todo seguir tres reglas básicas: (i) evitar documentación extensa; (ii) la documentación que se entrega al cliente ha de ser de alto nivel; (ii) la documentación se ha de realizar offline, es decir que no sea necesario acceder a Internet para poder tener acceso a ella.
- **Características extras.** Muchas veces parece una buena idea incluir características extras al sistema pensando que son sencillas de incluir y que al cliente le van a gustar, o bien, porque el programador, en algunas ocasiones, quiere probar una nueva tecnología en el sistema para ver cómo funciona, pero no realmente porque el sistema solicitado por el cliente así lo requiera. Sin embargo, hay que ser conscientes que a la hora de desarrollar *software*, hay que realizar trazas sobre cada elemento

incorporado, compilar, integrar, probar cada vez que se modifique una línea de código, diagrama, modelo... y por supuesto, cualquier característica extra que se haya incluido ya tendrá que ser mantenida durante toda la vida del sistema.

- **Intercambio de tareas.** El asignar personas a varios proyectos a la vez es una fuente de despilfarro muy significativa. Está demostrado que el tener a una persona en más de un proyecto hace que se multipliquen las interrupciones y el intercambios de tareas de un proyecto a otro lo que va a retrasar cada proyecto de manera importante incumpliendo los plazos estimados y establecidos inicialmente.
- **Esperas.** Durante un proceso de desarrollo de *software*, las esperas por cosas que se necesitan para continuar el trabajo constituye también una de las grandes fuentes de despilfarro. Retrasos en el inicio del proyecto, retrasos en la contratación del personal para el proyecto, retrasos debido a una documentación excesiva de los requisitos, retrasos en las revisiones y aprobaciones, retraso en las pruebas y retrasos en los despliegues constituyen un despilfarro importante a la hora de construir *software*.
- **Movimiento.** En este caso cabe plantearse el movimiento que se ha de llevar a cabo para obtener respuesta cada vez que un programador tiene una duda sobre algún aspecto de la implementación. Hay que considerar si se tiene a mano personal en la organización que desarrolla el *software* para poder resolver todas las dudas técnicas o hay que preguntarle al cliente o a algún representante del cliente, si tiene que irse a otra planta del edificio para comprobar cómo han ido las pruebas sobre el código que ha implementado. Es por ello, como ya se ha visto en anterioridad con el desarrollo ágil, que se recomienda que todo el equipo de desarrollo (programadores, los que realizan las pruebas del *software*, representantes del cliente que podría ser alguien de la misma organización que desarrolla el *software*...) se encuentre ubicados en una misma sala. Pero no solamente son las personas la que tienen que moverse de cara, en su caso, a resolver dudas y otros aspectos vinculados al desarrollo del *software*, también los requisitos se mueven, cuando pasan de los analistas a los diseñadores y después pasa a los programadores, y de ahí a los que realizan las pruebas, es decir, los *testers* o ingenieros de pruebas, como se les suele denominar en castellano. Cada traspaso o movimiento que se realiza de un artefacto puede generar despilfarro en el proyecto.
- **Defectos.** Dependiendo del impacto del defecto detectado y el tiempo que se tarde en detectar el defecto en el desarrollo del *software* se podrá generar más o menos despilfarro. En este sentido, no tiene nada que ver con la gravedad del defecto, ya que un defecto crítico que se ha detectado rápidamente, por ejemplo, en tres minutos, no va a generar mucho despilfarro. Sin embargo, un defecto mínimo, por simple y sencilla que sea su resolución, si transcurre mucho tiempo hasta su descubrimiento,

por ejemplo, varias semanas, puede constituir una fuente importante de despilfarro para el proyecto.

A pesar de las similitudes en muchos aspecto entre el pensamiento *lean* asociado a TPS y el *lean* aplicado al desarrollo de *software*, lo que ocurre, como ya se ha demostrado y citado en incontables ocasiones, construir *software* no es igual que construir coches, por lo que también se dan diferencias entre el desarrollo de *software* y la producción en el pensamiento *lean*, tal y como se ilustra en la Tabla 2.

Tabla 2. Diferencias entre producción y desarrollo de *software* (Poppendieck y Poppendieck, 2003).

Desarrollo. **Diseño de la receta**	Producción. **Elaboración del plato**
La calidad está determinada por su uso	La calidad viene definida de acuerdo a si es conforme a los requisitos
Resultados variables son buenos	Resultados variables son malos
Las iteraciones generan valor	Las iteraciones generan despilfarro (implican rehacer el trabajo)

Una de las diferencias fundamentales de las que se muestran en la Tabla 2 y que requiere un poco más de detalle es la que se refiere a calidad. La calidad vista desde la producción de productos físicos está definida en función de si es conforme a los requisitos que se especificaron en el diseño o la *receta*.

De esta manera, por ejemplo, para elaborar un plato, se sigue la receta y la calidad vendrá en función si ha seguido correctamente los pasos y ha utilizado los ingredientes adecuados (otra cosa sería la calidad del servicio, por ejemplo, a la hora de servir el plato en un restaurante, pero este aspecto ya entraría en otro contexto distinto que no se va a tratar en este tema).

Sin embargo, la calidad en el *software* depende de la **integridad percibida** y de la **integridad conceptual** (Poppendieck y Poppendieck, 2003). La **integridad percibida** depende del cliente y se obtiene como el conjunto de lo que sería funcionalidad, usabilidad, fiabilidad y coste del producto. La **integridad conceptual** hace referencia a que todos los conceptos vinculados al *software* se encuentren distribuidos de una manera cohesiva, es decir, se mantiene un equilibrio entre flexibilidad, mantenibilidad, eficiencia y un comportamiento reactivo.

Así, por ejemplo, si una compañía aérea tiene en la web sistemas distintos para reservar billetes dependiendo de si es en clase turista o clase preferente, claramente se está viendo que no hay buena cohesión entre los distintos componentes del sistema global de la compañía aérea, ya que el sistema de reservas en este caso debería seguir un mismo y único diseño independientemente del tipo de billete que se reserve (misma entrada de datos, mismo número de campos...).

Mary Poppendieck y Tom Poppendieck resumen en 7 los principios aplicables de *lean* al desarrollo de *software* (Garzás, Enríquez de S. e Irrazábal, 2013) (Poppendieck y Poppendieck, 2003):

- **Eliminar despilfarro.** Este principio se basa en eliminar todo aquello que no genera valor para el cliente, con la misma filosofía del *lean* de TPS.
- **Amplificar el aprendizaje.** Este principio se basa en que el desarrollo de *software* es un proceso de aprendizaje continuo en el que, por ejemplo, a través de reuniones frecuentes y cortas del equipo de desarrollo con el cliente o sin el cliente, se pueda ir aprendiendo sobre la funcionalidad que ha de implementar el sistema y la mejor forma de llevarla a cabo.
- **Retrasar las decisiones lo más que se pueda.** Este principio se basa en que hay que retrasar las decisiones todo lo que se pueda para controlar mejor la incertidumbre en muchos aspectos vinculados al *software*.
- **Realizar entregas lo más rápido posible.** Este principio se basa en que hay que hacer entregas del *software* lo más rápido posible para poder ir recibiendo *feedback* del cliente e ir orientando mejor el proyecto.
- **Dar más valor al equipo.** Este principio se basa en que los roles en el proyecto deben cambiar. Así, los directivos aprenderán a escuchar y valorar al equipo de desarrollo (y en este caso no se está refiriendo al aspecto económico), que les explicarán las acciones que se podrían llevar a cabo en el proyecto de cara a la implementación y sugerencias de cómo mejorarlo.

Por último, merece la pena incluir una cita de otro erudito de *lean*, Ash Maurya, de la que se podría decir que resume todo este paradigma: «La vida es demasiado corta para construir algo que nadie quiere».

11.3. *Kanban*

Cuando una organización se decide a adoptar prácticas ágiles, la opción más común es utilizar ***Scrum* acompañado de *Kanban***, ya que ambos enfoques se complementan muy bien, aunque por separado presentan aspectos bien diferenciados, tal y como se resume en la Tabla 3.

Característica	*Scrum*	*Kanban*
Iteraciones de duración predeterminada	Obligatorio	Opcional
Compromiso del equipo de desarrollo con el trabajo de una iteración	Obligatorio	Opcional
Estimación	Obligatorio	Opcional
Métrica utilizada para la mejora del proceso y la tarea de planificación	Velocidad	*Lead time*
Roles	*Product Owner* *Scrum Master* Equipo de desarrollo	-
Equipos multifuncionales	Obligatorio	Opcional, se permiten equipos especializados
Tamaño de las tareas	Deben poder terminarse en un *Sprint*	Sin restricciones
Gráficos obligatorios	*BurnDown*	-
Limitaciones del WIP	Implícitamente en el *Sprint*	Explícitamente a partir del WIP
Añadir tareas en una iteración	No se permite	Sí, siempre que exista capacidad en función del WIP
Responsabilidad de las tareas en una iteración	Equipo específico	Se pueden compartir entre equipos o personas
Borrado de las pizarras	Finalización del *Sprint*	Nunca

Kanban tiene su origen en una palabra japonesa cuyo significado es **«tarjetas visuales»** y fue utilizado por primera vez en la compañía Toyota. Taiichi Ohno se fijó en la forma de trabajo de los supermercados norteamericanos a la hora de reponer los productos en las estanterías. La idea era que una vez que se producía la demanda de un coche se utilizaran las tarjetas para saber todas las piezas que se necesitaban para su construcción, cuántas estaban disponibles y cuántas faltaban.

De esta manera se podía planificar la fabricación del coche, controlar el inventario y coordinar los distintos pasos del proceso de la fabricación del vehículo siguiendo así la filosofía *lean* del *just-in-time*, o justo a tiempo como se diría en castellano, es decir, de cara a aumentar la productividad en la fabricación de los productos, el objetivo es producir exclusivamente los elementos que se necesitan, en su cantidad justa (recordar procesos «*pull*»), y en el momento que se necesitan eliminando todo despilfarro.

La Figura 8 muestra un ejemplo de la tarjeta original de Toyota y la Figura 9 muestra otro ejemplo de tarjeta Kanban más legible.

Figura 8. Tarjeta *Kanban* utilizada por Toyota. Fuente: http://www.toyota-global.com/company/vision_philosophy/toyota_production_system/just-in-time.html .

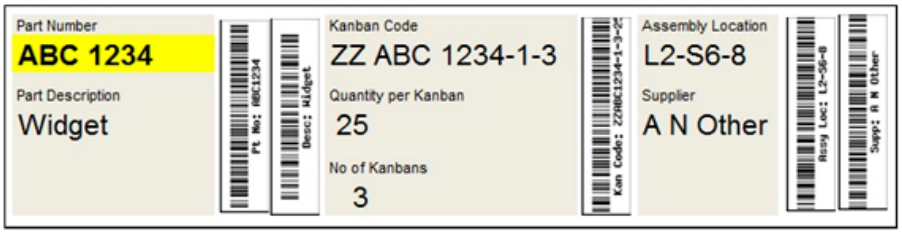
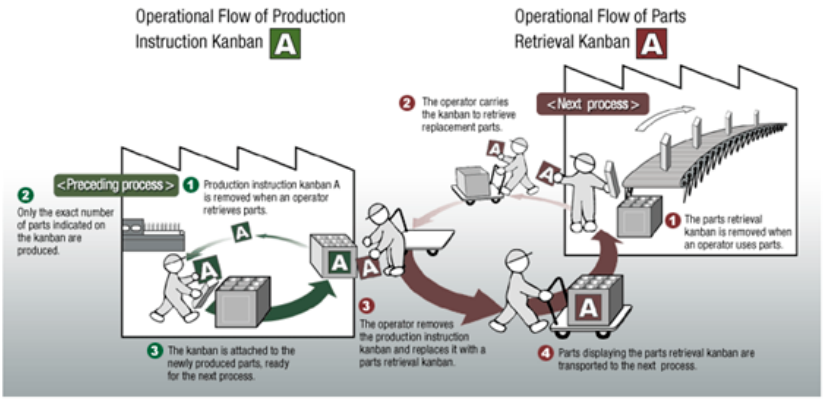


Figura 9. Ejemplo de tarjeta Kanban. Fuente: https://altacuncta.wordpress.com/2013/12/02/que-es-un-kanban/ .

El esquema general del sistema *Kanban* ideado por Toyota es el que se muestra en la Figura 10.



La primera aplicación del sistema *Kanban* al desarrollo de *software*, vinculado de forma muy estrecha con el pensamiento *lean*, la llevó a cabo David Anderson, como presidente de la empresa Modus Cooperandi (Anderson, 2009).

Para el desarrollo de *software*, *Kanban* al igual que *Scrum* utiliza un desarrollo incremental, historias de usuario, tareas, etc., pero en este caso, además, hace uso de un tablero, el denominado tablero *Kanban*, que va a permitir a cada miembro del equipo de desarrollo obtener una visualización completa del estado de implementación del *software*.

La idea es que el tablero se cuelga en la pared y se divide en una serie de columnas que reflejan el estado del proyecto en lo que a tareas se refiere. Básicamente: tareas que hay que hacer, tareas pendientes, tareas que se están realizando y tareas que se han terminado. Los más puristas de *Kanban* añaden una columna más que refleja el estado de ánimo de cada miembro del equipo de desarrollo, normalmente con el uso de *caritas*. De esta manera, las tarjetas *Kanban*, que normalmente serán *post-it*, se irán colocando en cada columna según corresponda.

La Figura 11, Figura 12 y Figura 13 muestran ejemplos de tableros *Kanban* utilizados en distintos proyectos *software*.

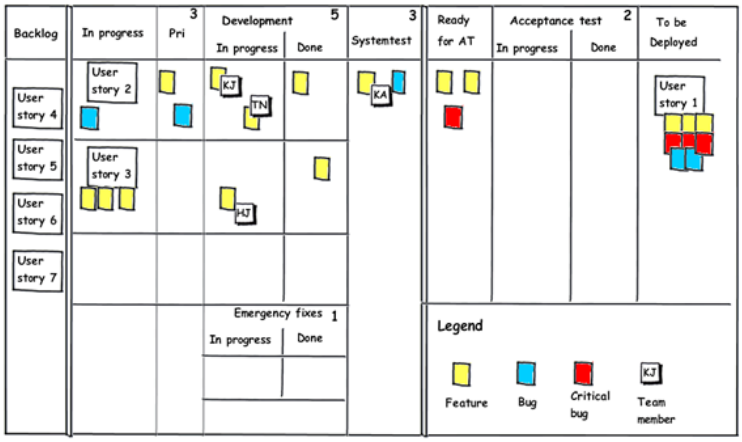


Figura 11. Ejemplo de tableros *Kanban*. Fuente: https://ketiljensen.wordpress.com/2009/10/31/kanban-the-next-step-in-the-agile-evolution/

Figura 12. Ejemplo de tableros *Kanban* (Garzás, Enríquez de S. e Irrazábal 2013).

Figura 13. Ejemplo de tableros *Kanban*. Fuente: http://www.snl19.es/5-claves-gestion-proyectos-metodologia-kanban/ .

También se puede utilizar esta técnica del tablero y las *caritas* en la reunión de retrospectiva de la iteración llevada a cabo (en *Scrum* la conocida como reunión *Sprint Retrospective*), para poder mostrar así de manera gráfica el grado de satisfacción en distintos aspectos, tal y como se ilustra en la Figura 13, donde se valora la técnica, la metodologías, el ecosistema y la filosofía empleada en la iteración.

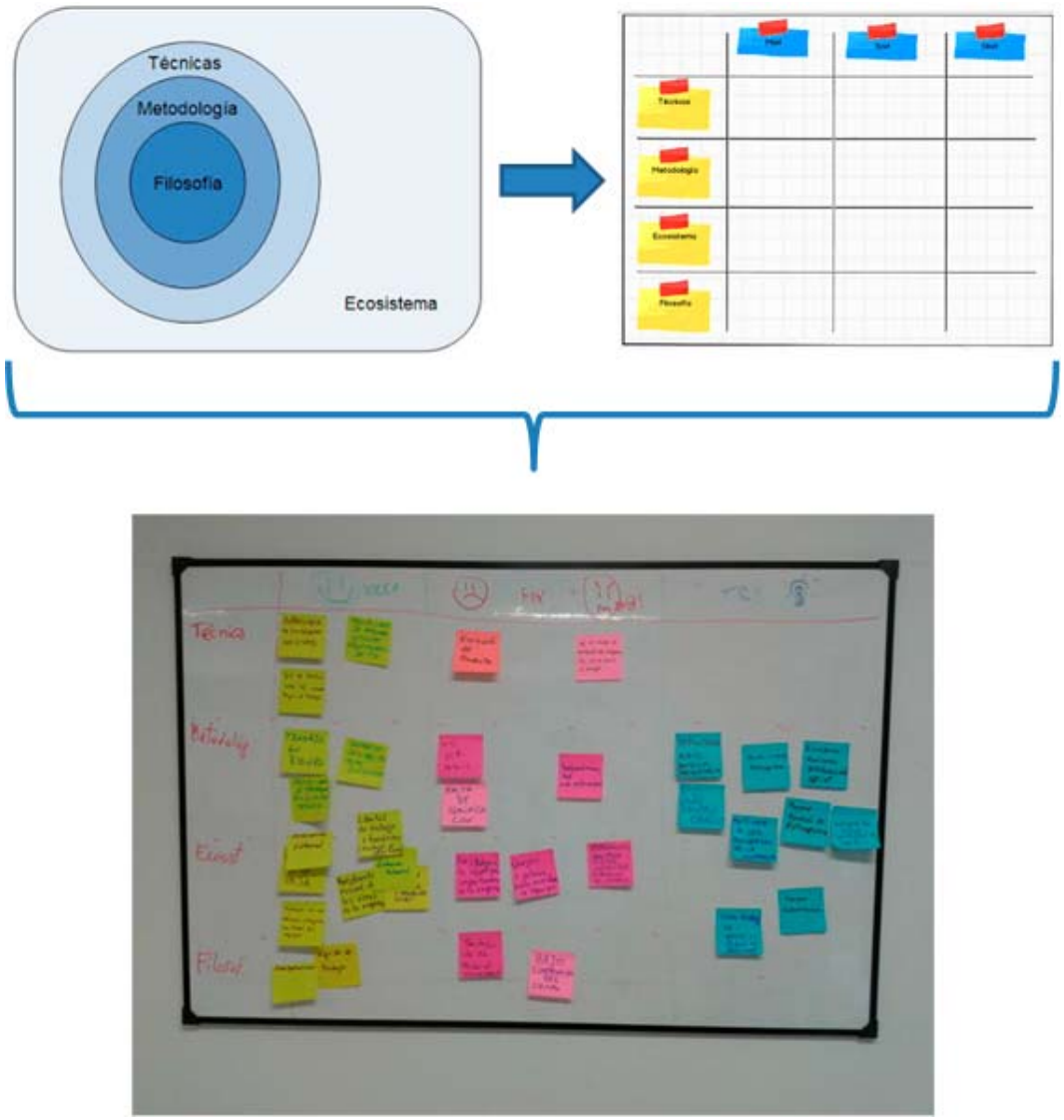


Figura 13. Tablero para la reunión de retrospectiva. Fuente: http://www.elproximopaso.net/2011/12/dinamica-de-retrospectiva-3-caras-y-4.html

11.4. Referencias bibliográficas

Anderson, D. (2009). A kanban system for software engineering. *Best Tech Videos*. Recuperado de http://es.scribd.com/doc/13748689/A-Kanban-System-for-Software-Engineering

Burlton, R. y Matthijssen, P. (2013). The Second Wave of Lean. *BPTrends articles*. Recuperado de http://www.bptrends.com/bpt/wp-content/uploads/11-05-2013%20ART-SecondWaveofLean-Burlton%20%20Mathijssen_revPJM.pdf

Eshuis, R. y Wieringa, R. (2001). An Execution Algorithm for UML Activity Graphs. *Actas de International Conference on the Unified Modeling Language (UML)*. Toronto, Canada. Springer Verlag.

Garzás, J., Enríquez de S., J.A. e Irrazábal, E. (2013). *Gestión Ágil de Proyectos Software*. Kybele Consulting.

Liker, J. (2004). *The Toyota Way: 14 Management Principles from the World's Greatest Manufacturer*. McGraw-Hill Professional.

Marshall, D. (2000). *Enterprise Modeling with UML. Designing Successful Software through Business Analysis*. Addison-Wesley.

Martin, R.C. (2009). *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall.

Ohno, T. (1988). *Toyota Production System: Beyond Large-Scale Production*. Productivity Press.

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

Sharp, A. y McDermott, P. (2001). *Workflow Modeling. Tools for Process Improvement and Application Development*. Artech House.

WfMC. (1999). Workflow Management Coalition. Terminology and Glosary. *WfMC-TC-1011 v3. Technical Report*. Recuperado de http://www.wfmc.org/

Womack, J.P. y Jones, D.T. (2012). *Lean Thinking: Cómo utilizar el pensamiento Lean para eliminar los despilfarros y crear valor en la empresa*. Gestión 2000. Traducción al castellano del libro: Womack, J.P. y Jones, D.T. (2003). *Lean Thinking: Banish Waste and Create Wealth in Your Corporation*, Revised and Updated. Simon + Schuster UK.

Womack, J.P., Jones, D.T. y Roos, D. (2007). *The Machine That Changed the World: The Story of Lean Production. Toyota's Secret Weapon in the Global Car Wars That Is Now Revolutionizing World Industry*. Scribner.

Lo + recomendado

No dejes de leer...

¿Cómo identificar los despilfarros?

Tornos, I. (2004). ¿Cómo identificar los despilfarros? *Forum Calidad*, 16(156), 58-60.

En este artículo el autor explica cómo se pueden identificar lo que en *lean* se conoce como despilfarro para aumentar la velocidad en un sistema logístico.

No dejes de ver...

Eric Ries habla en la Universidad de Toronto sobre *Lean Startup*

Vídeo de Eric Ries, autor del libro *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*, con subtítulos en castellano, en el que destaca que cuando se es emprendedor y se inicia un nuevo negocio se ha de invertir el esfuerzo en aquello que aporta valor al cliente para poder tener éxito y no fracasar en el intento.

Accede al vídeo desde el aula virtual o a través de la siguiente dirección web:
https://www.youtube.com/watch?v=vFmHqkiDIQQ

Four principles *lean* management. get lean in 90 seconds

Vídeo muy breve e ilustrativo para entender los principios de *lean*, de qué manera se pueden utilizar para eliminar los despilfarros y crear valor para los clientes de una organización.

Accede al vídeo desde el aula virtual o a través de la siguiente dirección web:
https://www.youtube.com/watch?v=wfsRAZUnonI

Intro to kanban in under 5 minutes

Vídeo breve de menos de 5 minutos que ayuda a entender de manera muy general la técnica Kanban.

Accede al vídeo desde el aula virtual o a través de la siguiente dirección web:
https://www.youtube.com/watch?v=R8dYLbJiTUE&feature=youtu.be

Introducción a *lean service management*

Charla del profesional Antonio Valle sobre *lean* para su aplicación en la gestión de servicios de TI.

Accede al vídeo desde el aula virtual o a través de la siguiente dirección web:
http://media.fib.upc.edu/fibtv/streamingmedia/view/22/431

+ Información

A fondo

The second wave of lean

Burlton, R. y Matthijssen, P. (2013). The Second Wave of Lean. *BPTrends articles*.

En este artículo se analiza la problemática de adoptar *lean* en las organizaciones y plantea la posibilidad de que quizá sea necesario mejorar este paradigma para que pueda aplicarse con mayor éxito en las empresas.

Accede al artículo desde el aula virtual o a través de la siguiente dirección web: http://www.bptrends.com/the-second-wave-of-lean

The genealogy of lean production

Holweg, M. (2007). The genealogy of lean production. *Journal Operations Management*, 25(2), 420 - 437.

En este artículo el autor presenta los hitos más importantes vinculados a la investigación llevada a cabo sobre *lean*.

Disponible en el aula virtual bajo licencia CEDRO

Enlaces relacionados

Instituto *Lean*

Sitio web equivalente al portal de Lean Enterprise Institute, *Inc*. (LEI), pero para España.

Accede a la página desde el aula virtual o a través de la siguiente dirección web:
http://www.institutolean.org/es/

Lean Enterprise Institute

Sitio web de Lean Enterprise Institute, Inc. (LEI). LEI es una organización sin ánimo de lucro que organiza eventos para difusión del pensamiento *lean* y recoge información, cursos, etc., vinculados a *lean*.

Accede a la página desde el aula virtual o a través de la siguiente dirección web:
http://www.lean.org/

The Lean EDGE

Interesante blog en la que distintos autores hablan sobre temas vinculados a *lean*.

Accede a la página desde el aula virtual o a través de la siguiente dirección web:
http://theleanedge.org/

Jamie Flinchbaugh ...on lean culture

Interesante blog gestionado por Jamie Flinchbaugh, apasionado de *lean*, en la que trata distintos aspectos relacionados con este pensamiento.

Accede a la página desde el aula virtual o a través de la siguiente dirección web:
http://jamieflinchbaugh.com/

Bibliografía

Garzás, J., Enríquez de S., J.A. e Irrazábal, E. (2013). *Gestión Ágil de Proyectos Software*. Kybele Consulting.

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

Womack, J.P. y Jones, D.T. (2012). *Lean Thinking: Cómo utilizar el pensamiento Lean para eliminar los despilfarros y crear valor en la empresa*. Gestión 2000. Traducción al castellano del libro: Womack, J.P. y Jones, D.T. (2003). *Lean Thinking: Banish Waste and Create Wealth in Your Corporation*, Revised and Updated. Simon + Schuster UK.

Womack, J.P., Jones, D.T. y Roos, D. (2007). *The Machine That Changed the World: The Story of Lean Production-- Toyota's Secret Weapon in the Global Car Wars That Is Now Revolutionizing World Industry*. Scribner.

Test

1. ¿Cuál es el objetivo principal de *lean*?

A. Reducir costes.

B. Eliminar despilfarro.

C. Mejorar la calidad de los productos.

D. Eliminar las jerarquías verticales en las organizaciones.

2. Indicar cuáles de las siguientes afirmaciones es correcta:

A. *Lean* es una forma de pensar sobre cómo mejorar el negocio.

B. *Lean* es una metodología que nos dice qué debemos hacer para mejora el negocio.

C. *Lean* centra su atención en la excelencia en el valor ofrecido al cliente.

D. *Lean* es una técnica creada para el desarrollo ágil.

3. ¿Cuáles de las siguiente opciones constituyen un factor de éxito de *lean* de acuerdo a Burlton y Matthijssen?

A. Enfoque dirigido a procesos.

B. Valor del cliente y calidad.

C. Gestión de la resistencia al cambio.

D. La dirección dictamina cómo se van a hacer las cosas sin involucrar al resto del personal.

4. ¿Qué tipos de desperdicio identifica el sistema de producción de Toyota, TPS?

A. *Seiro*, *seiton*, *seiso*, *seiketsu* y *shutsuke*.

B. *Seiri*, *seiton* y *muda*.

C. *Muda*, *mura* y *kaizen*.

D. *Muda*, *muri* y *mura*.

5. ¿Que define la filosofía de las 5S?

A. Se deben minimizar los recursos asignados para la fabricación de productos.

B. Cada cosa tiene que estar en un lugar determinado e identificado.

C. Solamente se utilizarán normas y estándares elaborados de manera interna en la organización.

D. Se seguirán las normar y estándares establecidos, pero el personal podrá tener iniciativa para aplicar su propio criterio.

6. ¿Qué tipo de despilfarros engloba el concepto de muda en el pensamiento *lean*?

A. Exceso de inventario.

B. Exceso de personal.

C. Demasiada comunicación con el cliente.

D. Transporte.

7. ¿Qué pasos define el ciclo de Deming?

A. Planificar, actuar, hacer y verificar.

B. Es una idealización que no se va a dar nunca en estado puro.

C. Planificar, hacer, verificar y actuar.

D. Planificar, estimar, probar y mejorar.

8. ¿Qué objetivo tiene el círculo de Ohno?

A. Observar desde el lugar real si los trabajadores cumplen su horario.

B. Observar desde el lugar real cuál es la forma de trabajar.

C. La detección de problemas y opciones de mejora a través de la observación del lugar de trabajo.

D. La mejora continua a través del ciclo PDCA.

9. ¿Qué aspectos asociados a la documentación de acuerdo al enfoque *Lean Software Development* se han de tener en cuenta de cara al cliente?

A. La documentación ha de ser sencilla y con un alto nivel de abstracción.

B. La documentación ha de ser lo más extensa posible para que el cliente pueda tener todos los detalles del proyecto.

C. La documentación siempre ha de estar disponible online para que pueda acceder a ella desde cualquier dispositivo.

D. La documentación debe aportar toda la información de diseño detallado del sistema.

10. ¿Cuáles son las diferencias básicas de *Kanban* y *Scrum*?

A. En *Scrum* se muestra el estado del proyecto de una manera visual mientras que en *Kanban* no.

B. En *Kanban* se muestra el estado del proyecto de una manera visual mientras que en *Scrum* no.

C. *Kanban* es más adaptativo que *Scrum* lo que ofrece más posibilidades de personalización.

D. *Kanban* incluye el pensamiento *lean* y *Scrum* no.

