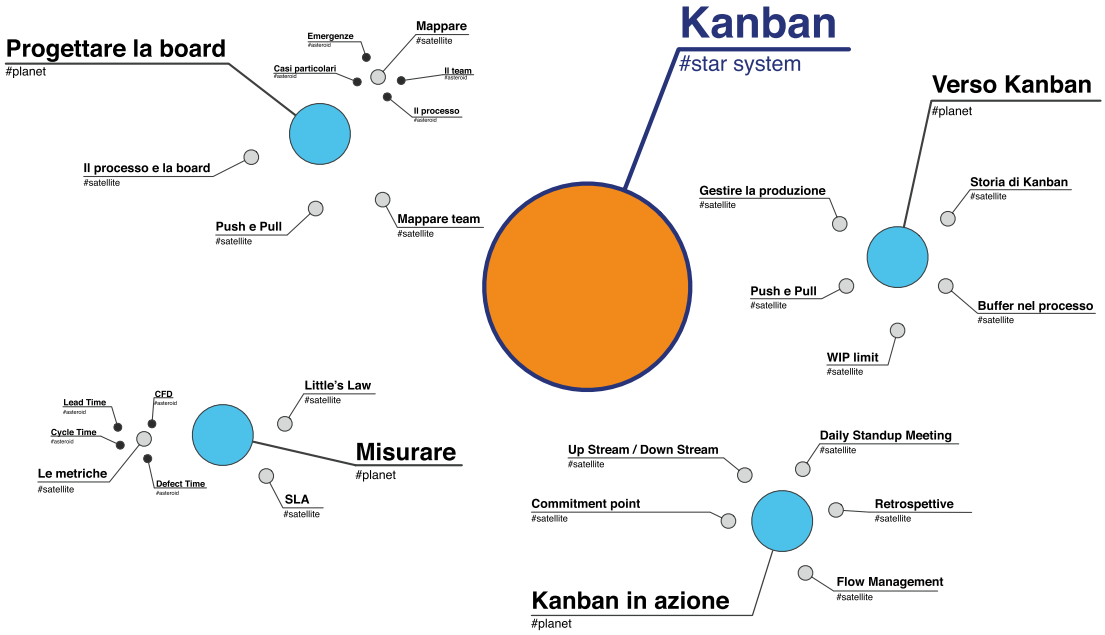
PARTE IV
KANBAN

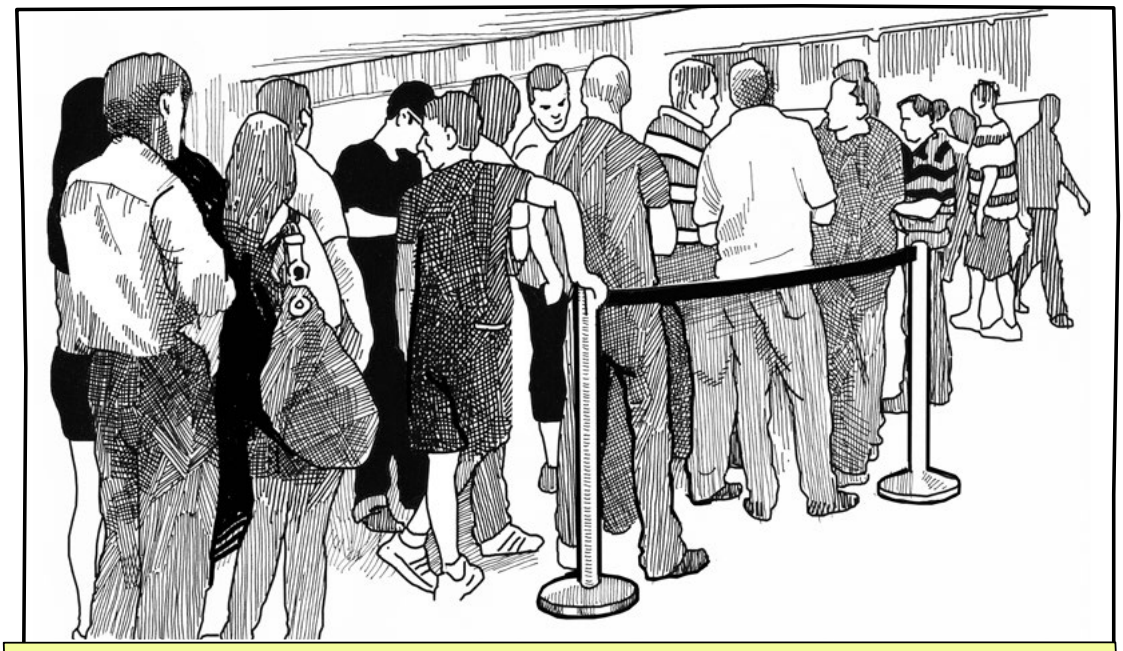
Giovanni Puliti



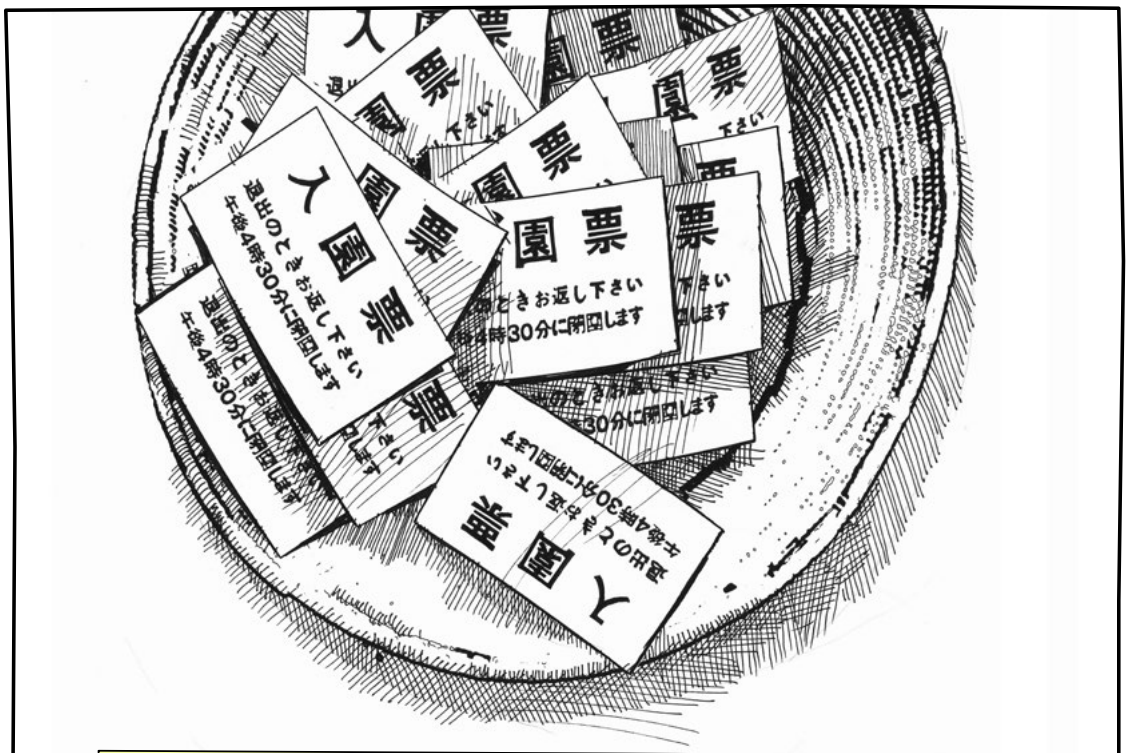
Parte 4
Il sistema Kanban

Nella lingua nativa del Giappone, "kanban" significa "segnale visivo" e la leggenda narra che la sua invenzione si deve ai guardiani del giardino imperiale giapponese di Tokyo… sul pianeta Terra.

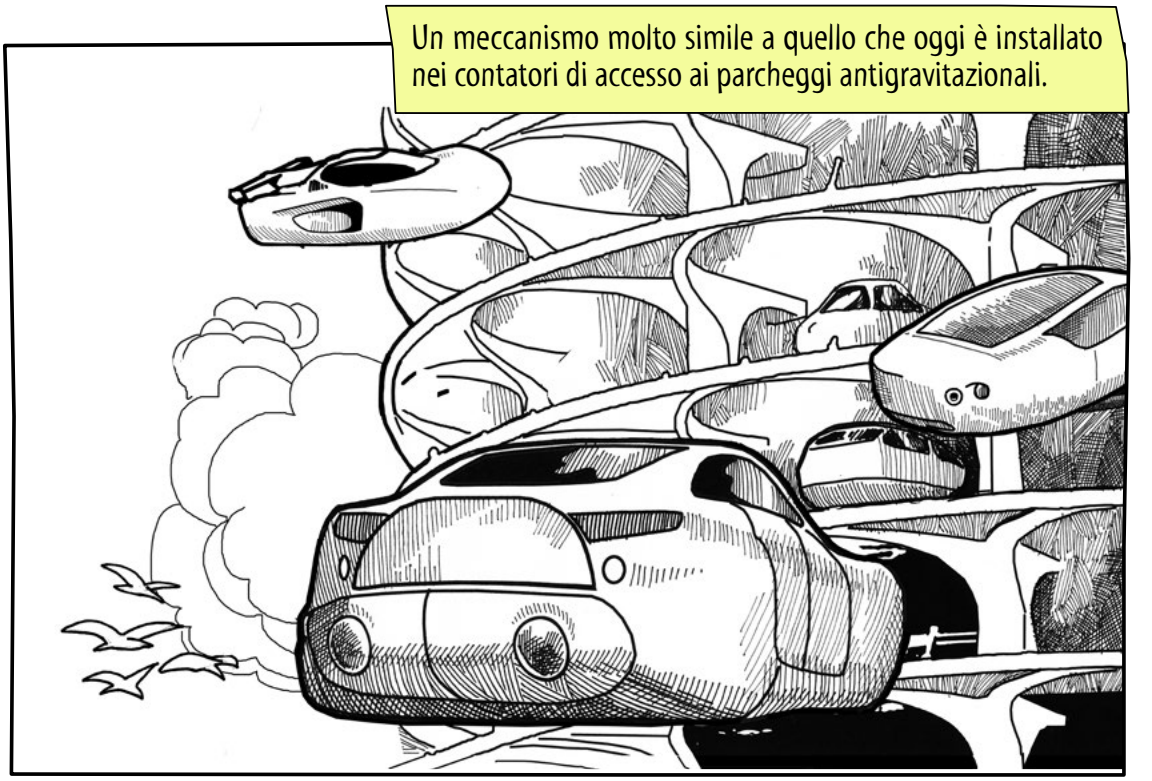
Per tenere sotto controllo il numero dei visitatori, fu inventato un meccanismo molto semplice per gestirne il flusso.



All’entrata, ogni visitatore doveva prelevare da un contenitore una piccola tessera intarsiata. All’uscita, doveva restituire tale tessera. Quando non vi erano tessere nel contenitore, nessun altro visitatore poteva entrare.

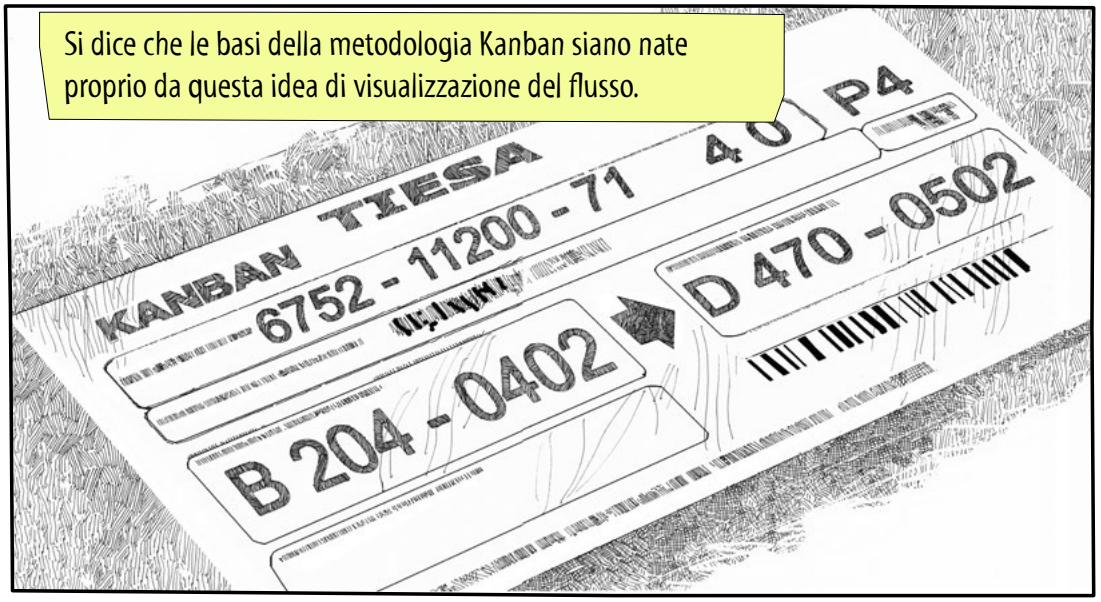


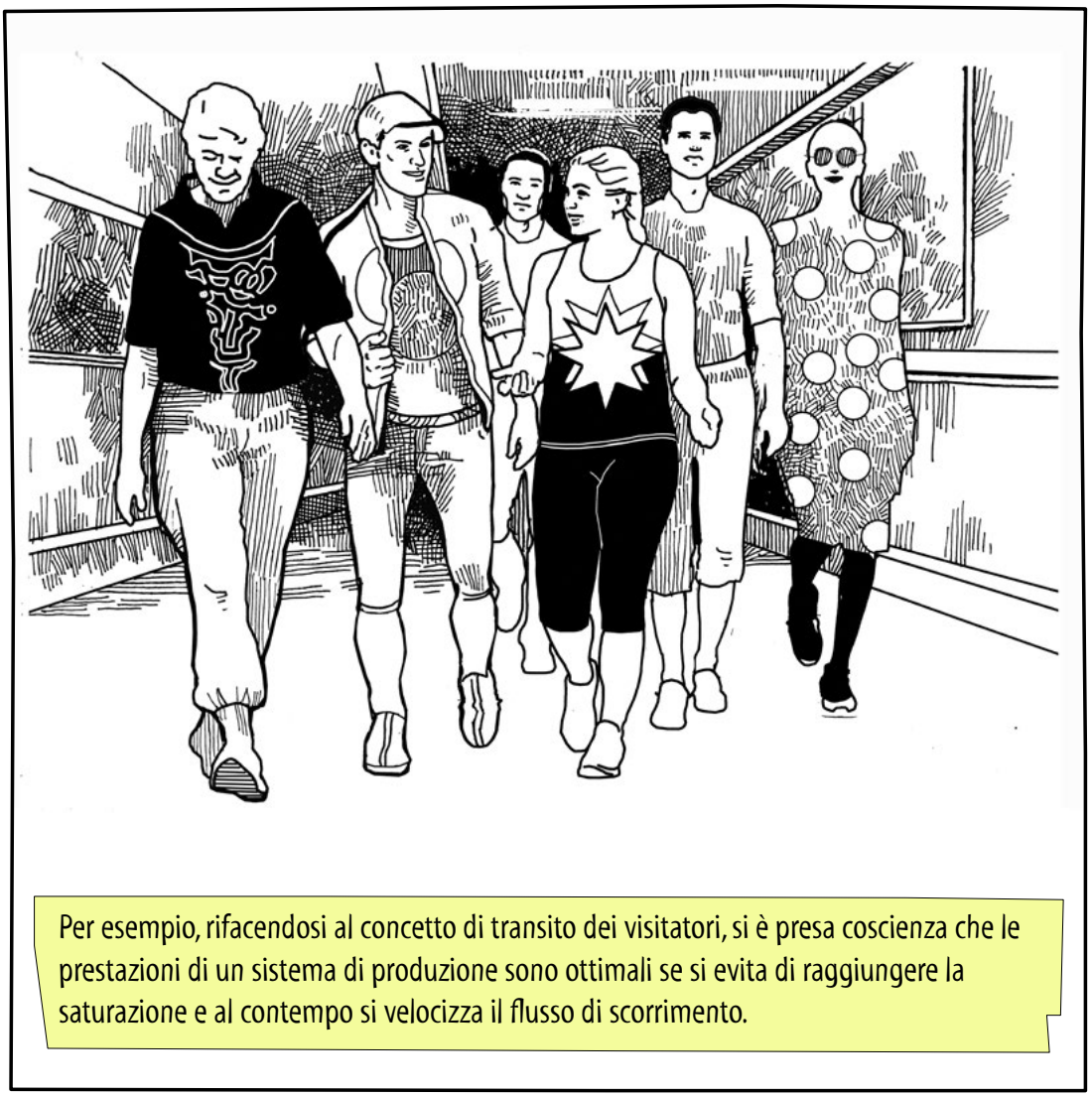
Il numero di tessere nel cestino corrispondeva al numero massimo di visitatori ammessi al giardino.


Un meccanismo molto simile a quello che oggi è installato nei contatori di accesso ai parcheggi antigravitazionali.


E cosa c'entra tutto questo con la metodologia Kanban?
KT
NAV1
FEET

Si dice che le basi della metodologia Kanban siano nate proprio da questa idea di visualizzazione del flusso.

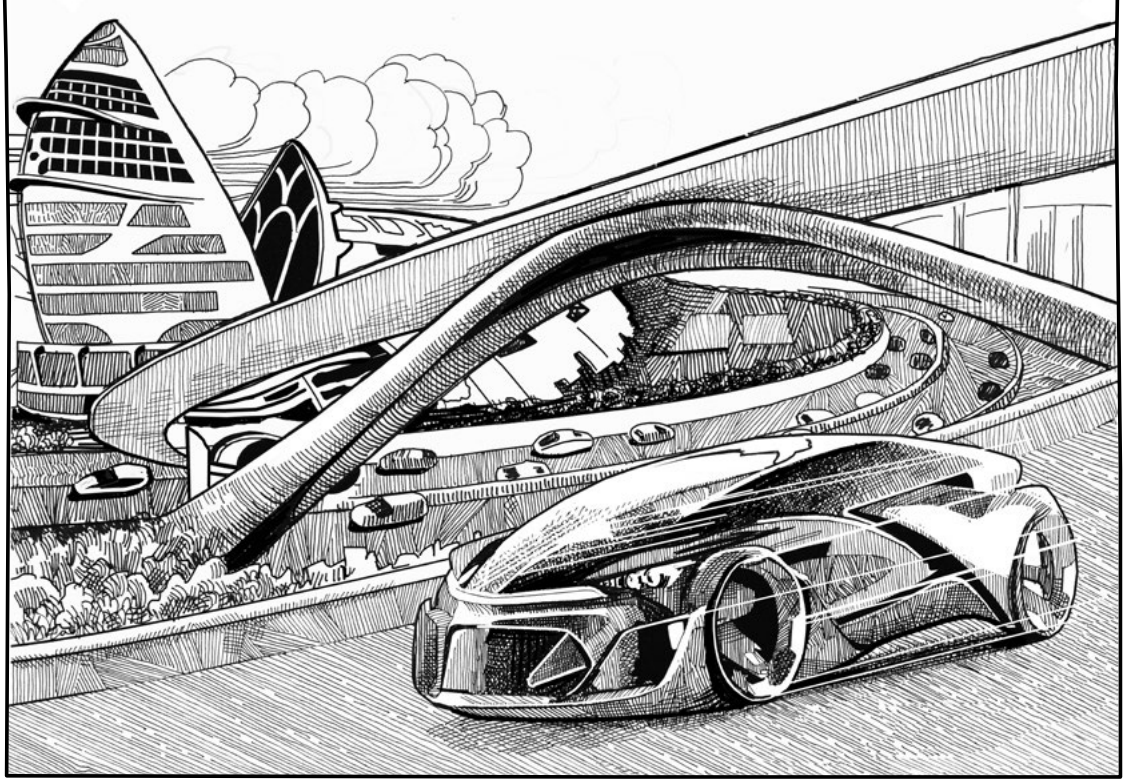




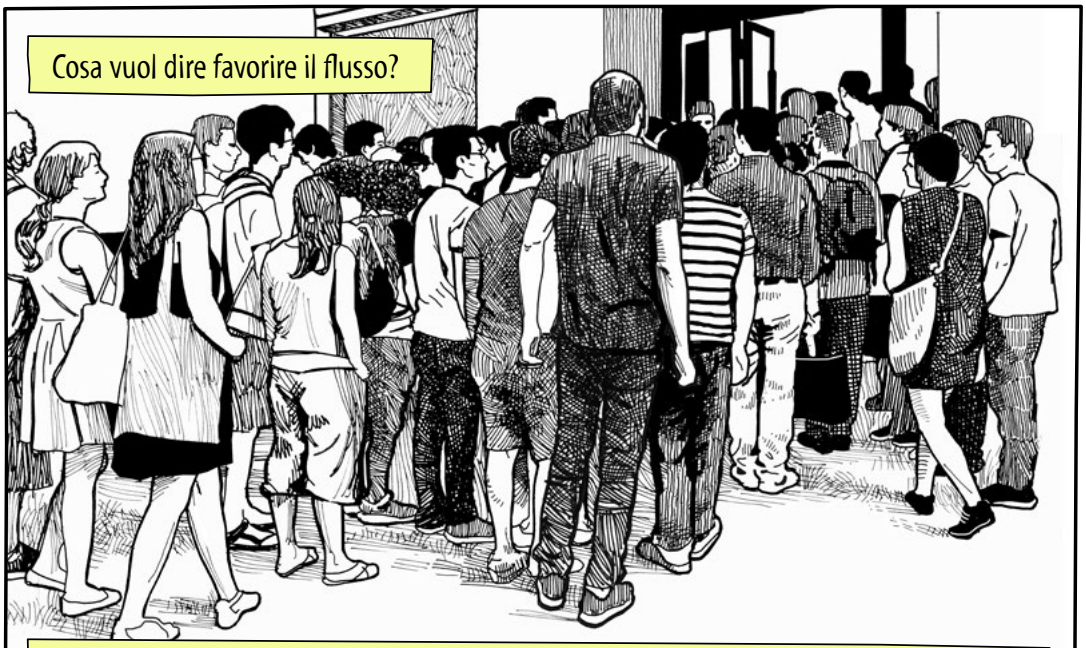
Per esempio, rifacendosi al concetto di transito dei visitatori, si è presa coscienza che le prestazioni di un sistema di produzione sono ottimali se si evita di raggiungere la saturazione e al contempo si velocizza il flusso di scorrimento.

È un fenomeno che i terrestri avevano sperimentato sulla loro pelle passando molte ore bloccati in coda nelle loro auto a combustibili fossili…

I principi messi in atto nel giardino imperiale giapponese sono alla base dei regolamenti delle moderne linee di collegamento che percorriamo tutti i giorni con i nostri sprinter: aumentare la banda (numero di mezzi che possono transitare in un istante) non basta; per esempio è necessario ridurre la velocità media e possibilmente semplificare la viabilità per favorire lo scorrimento.



Cosa vuol dire favorire il flusso?



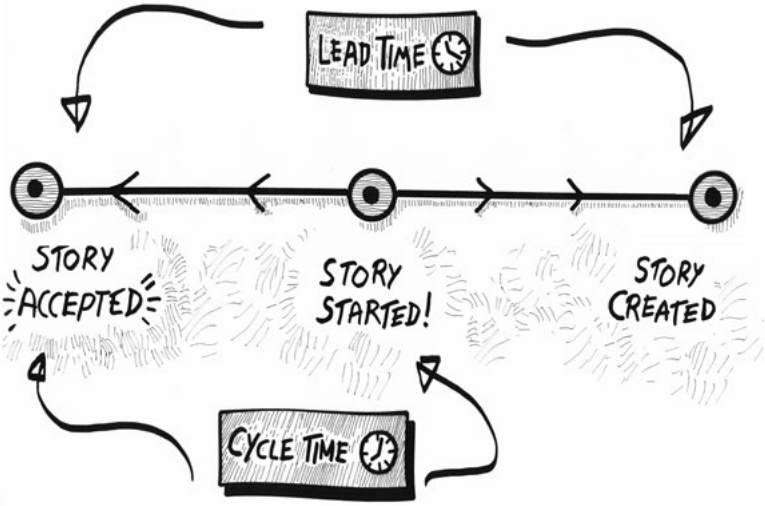
Significa, per esempio, cercare di impedire l'insorgere di ingolfamenti nel percorso, qualunque esso sia, per non incorrere in un rallentamento o peggio ancora in un blocco. Vale per una coda di persone, per una autovia o per un processo di produzione all'interno di una moderna fabbrica.

Grazie all'uso delle tecniche Kanban, gli esperti di processi di produzione si resero conto che era utile semplificare il processo in modo da stabilizzarlo. Un noto adagio diceva: meglio una lavorazione lenta ma costante, che una che procede per sbalzi di velocità.

È quello che succede in una scala mobile che procede lentamente, ma senza soluzione di continuità e senza sbalzi.

Con questo approccio si punta a fare meno cose in parallelo, ma a velocizzarne la lavorazione.

Una celebre frase dice: “Stop starting, start finishing”.



Minore è il tempo di lavorazione, maggiore sarà la prestazione complessiva.

Prima di intervenire è necessario osservare il flusso all’interno del processo di produzione del sistema

Misurare e gestire il flusso

Esplicitare le regole del processo
Identificare le possibili opportunità di miglioramento.

Limitare il Work-in-Progress (WIP)

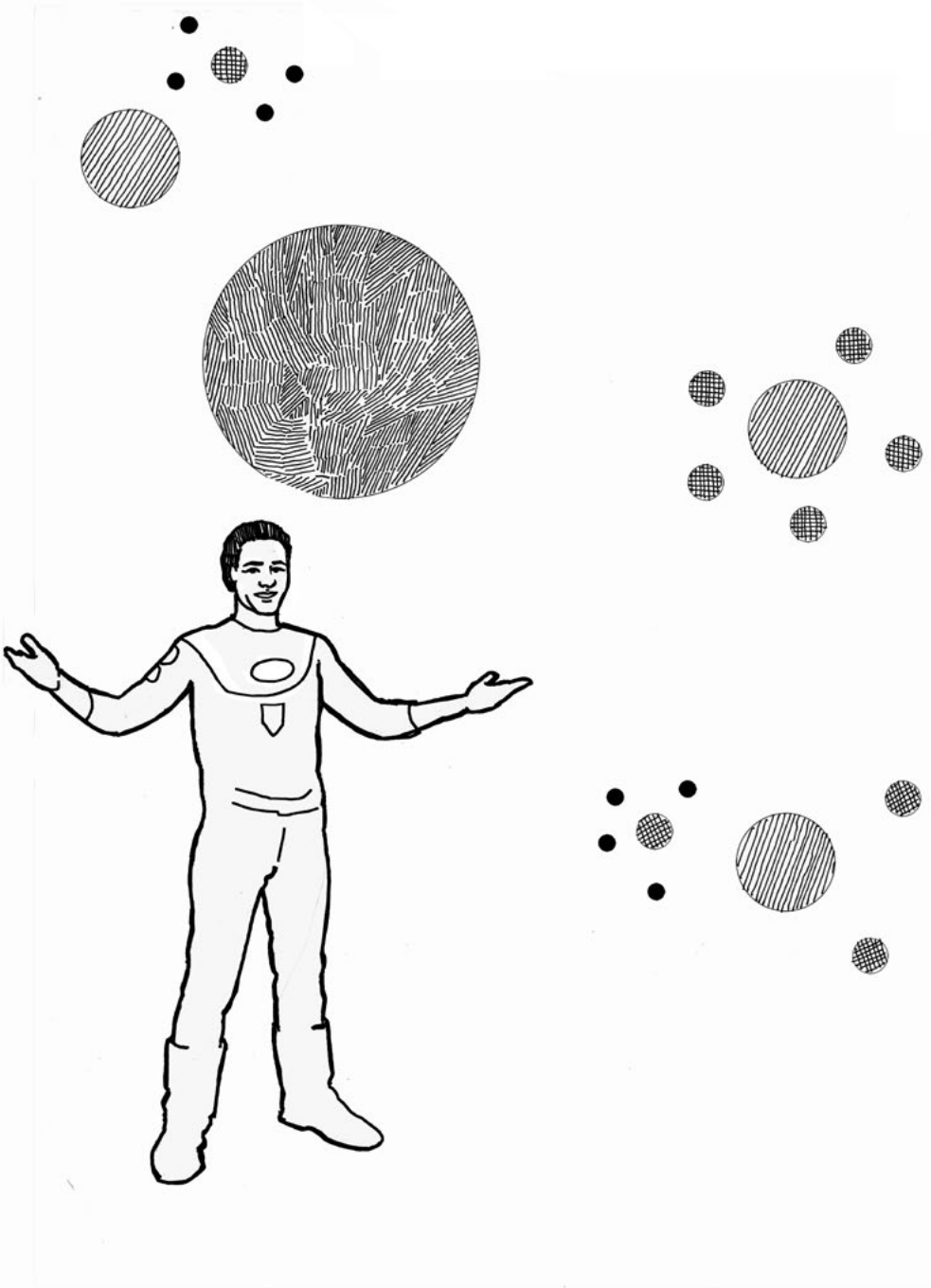
Kanban quindi ha a che fare con il miglioramento e con il cambiamento del modo di organizzare il lavoro. I suoi principi sono molto semplici.


E come fecero a fare tutte queste cose?
Per esempio, come facevano a visualizzare il flusso?
Ci fu un momento in cui si misero in discussione i complessi strumenti e processi di project management, preferendo cose più semplici, tipo una lavagna su cui si attaccavano dei cartellini adesivi.


E funzionava?
Sì, in effetti funzionò. Soprattutto perché questi semplici strumenti permettevano di instaurare la discussione tutti insieme di fronte a grafici o numeri. La condivisione delle informazioni contenute in questi "information radiator" era la cosa più importante.

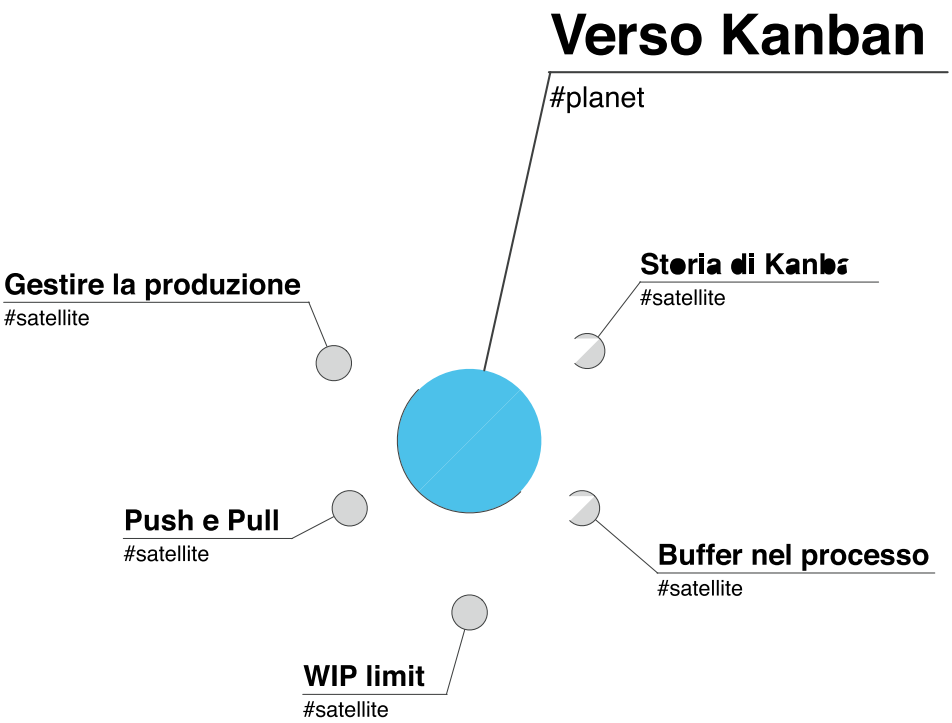

Interessante… project management fatto coi biglietti di carta…
E gli altri principi di Kanban quali sono?


Sono questi…
condividere un percorso di cambiamento che abbia come obiettivo la crescita e l'evoluzione dell'organizzazione
usare gli strumenti di cui si è già in possesso: partire da quello che si sa fare senza aspettare di avere imparato nuove cose
rispettare l'organizzazione attuale



Su questo sistema solare vedremo quindi come, grazie all'uso di lavagne con bigliettini attaccati e un po' di matematica, sia possibile non solamente controllare un processo, ma anche introdurre nell'organizzazione un miglioramento a livello sistemico.

Capitolo 1
Origine e principi di Kanban



Introduzione ai principi Lean Agile

In questo capitolo introduciamo i concetti fondamentali della **metodologia Kanban** (con la "K" maiuscola); tramite una serie di esempi vedremo anche come implementare una **kanban board** (con la "k" minuscola).

NOTA: Esiste una differenza importante tra *Kanban* con l'iniziale maiuscola (la metodologia sviluppata da D.J. Anderson) e *kanban* con l'iniziale minuscola (il cartellino segnalatore utilizzato inizialmente nel Toyota Production System). In tal senso le parole di D.J. Anderson sono particolarmente esaustive: "...Kanban (con la 'K' maiuscola) è il metodo di cambiamento evolutivo che utilizza un sistema *pull* basato sui cartellini e sulle *board* kanban (con la 'k' minuscola), sulla visualizzazione dei flussi e su altri strumenti per catalizzare l'introduzione di idee Lean all'interno del mondo dello sviluppo tecnologico e delle attività IT".

Vedremo come, tramite Kanban, sia possibile gestire un processo di produzione: in tal senso Kanban può essere considerata anche una metodologia di *project management*. Ma vedremo soprattutto come sia possibile osservare, valutare, far emergere i problemi per poi risolverli. Per questo motivo personalmente considero **Kanban** un vero e proprio strumento di **sense making**.

I principi base sui quali si poggia la metodologia sono per la maggior parte tratti dal lavoro sviluppato in Toyota da Taiichi Ōno durante la formulazione del cosiddetto **Toyota Production System**, noto nel mondo occidentale come **Lean Production**. Per questo motivo, per comprendere **Kanban** è impossibile prescindere dalla comprensione dei principi base della **Lean** tanto le due cose sono strettamente collegate fra loro.

È per questa ragione che ho pensato di cominciare questo capitolo presentando una serie di esempi, presi dalla vita di tutti i giorni, con cui spiegare in modo semplice concetti come **just in time**, produzione **trainata** dal mercato, dimensione dei **lotti** di produzione, limitazione delle attività **in parallelo**; tutti aspetti alla base della **Lean Production** e che tramite **Kanban** si possono tenere sotto controllo per migliorare il processo di produzione.

Non entreremo nel dettaglio della produzione snella e del Toyota Production System, sebbene una comprensione dei principi di base sia utile se non indispensabile. Per chi fosse interessato a maggiori approfondimenti su questi argomenti, si consiglia la lettura dei due testi più importanti in materia: *Toyota Way* [TW] e *La macchina che ha cambiato il mondo* [MCW].

La pizza al taglio, ossia il Lean nel negozio all'angolo

Il primo esempio che presentiamo è quello di una pizzeria al taglio che ha aperto da poco in paese. Im questa piccola bottega c'è un pizzaiolo esperto che, dopo molti anni di lavoro alle dipendenze di altri principali, ha deciso di mettersi in proprio. Useremo un nome di fantasia per questo simpatico ristoratore e nella migliore tradizione della pizza napoletana, lo chiameremo Antonio, per gli amici "Totò".

Antonio, nel piccolo negozio ha messo un **banco** con una **vetrina** dove gli avventori possono vedere le **pizze** appena sfornate, già tagliate in **fette** o **spicchi**. Accanto alla vetrina c'è il bancone per le consegne e una piccola cassa.

Normalmente Antonio tiene in questa vetrina quattro o cinque tipi di pizza, scelte fra quelle più richieste dalla clientela. Il pomeriggio, all'uscita della scuola vicina, molti bambini si fermano coi genitori per fare merenda e per questo motivo una delle pizze più richieste è una specie di focaccia dolce farcita di cioccolata. La sera invece gli adolescenti preferiscono pizze dai sapori più forti, piccanti o magari qualche ricetta più dietetica.

Dietro il bancone Antonio ha installato una macchina **spianatrice** — con la quale, partendo da palle di pasta preparata manualmente, si formano le basi di pasta cruda — e un paio di **forni** elettrici per la cottura.

Antonio, da esperto pizzaiolo, è in grado di sfornare sempre la giusta quantità di pizze per ogni tipologia, tanto che normalmente l'attesa media dei clienti è sempre molto ridotta.

Gestire una pizzeria con un approccio Lean

Il modo con cui Antonio gestisce la sua pizzeria è molto efficiente e vi si possono trovare applicati molti dei principi della filosofia di produzione Lean messa a punto da Toyota nel dopoguerra: gestione del magazzino **just in time**, produzione **pull** "trainata"

Figura 1. Antonio, il pizzaiolo della pizzeria a taglio sotto casa, applica i principi della Lean Production che sono i capisaldi della metodologia Kanban.

dal mercato, limitazione del **work in progress** sono tutti principi utili quando si deve gestire un processo di produzione quale esso sia: dalla multinazionale alla pizzeria sotto casa.

Produzione snella: approccio pull e gestione del magazzino just in time

Nella pizzeria, Antonio fa in modo di avere in ogni momento **poche pizze pronte** nella sua vetrina, cosicché che la clientela possa sempre gustare pizze calde appena sfornate.

Lo scopo di Antonio è quello di ottimizzare la produzione, ossia fare in modo che il bancone non sia mai completamente vuoto — i clienti dovrebbero aspettare tutto — o troppo pieno — pizze fredde da riscaldare — ottenendo così anche un altro risultato: a fine giornata, gli sprechi saranno stati minimi.

In termini Lean, potremmo dire che Antonio applica un processo di lavorazione "trainato" dal mercato (approccio **pull**) e organizza la sua produzione secondo un meccanismo **just-in-time**, per cui non si creano in anticipo grandi sovrapproduzioni che poi devono essere vendute per svuotare il bancone, corrispondente in termini industriali al magazzino dei prodotti finiti. Entrambe queste strategie sono possibili grazie al fatto che nel processo di lavorazione sono inserite delle "aree di parcheggio", di cui parleremo diffusamente sotto.

Nei momenti in cui non c'è molto da fare, quando per esempio il bancone è pieno o c'è poca clientela nel negozio, Antonio si mette a svolgere quelle attività non urgenti, che potremmo definire **a bassa priorità**, come per esempio preparare le palle di pasta che poi andranno in frigorifero, oppure pulire il laboratorio, oppure riposarsi.

Il suo obiettivo è fare in modo che sul bancone ci siano sempre tre o quattro pizze pronte: conoscendo il tempo necessario per realizzare una pizza (il cosiddetto **cycle time di processo**), appena vede che le scorte scendono sotto un certo quantitativo, si mette a preparare una nuova pizza.

È quindi il flusso dei clienti che comprano le pizze, ossia il **ritmo** di **svuotamento** del **magazzino**, che determina il processo di lavorazione: in gergo si dice che il **processo viene tirato** (*pull* in inglese) dal fondo.

"Spingere" la produzione: approccio push

Se Antonio applicasse un approccio opposto (**push**), si concentrerebbe sulla produzione delle pizze cercando prima di tutto di immaginare quante ne riuscirebbe a vendere: in questo caso la sua unica preoccupazione sarebbe quella di "spingere" la produzione dal forno verso il bancone e quindi verso la bocca dei clienti, in modo da non accumulare troppe pizze pronte per essere infornate. Per perseguire questo approccio, dovrebbe fare molta attenzione alla pianificazione e controllare accuratamente l'organizzazione del suo processo di lavorazione, valutando in particolar modo i vari "colli di bottiglia" della produzione: il forno, la preparazione delle palle di pasta, il bancone e così via.

Antonio sa che questo lavoro di controllo e pianificazione è indubbiamente più complicato, sicuramente meno efficiente dell'altro (**pull**); inoltre a fine giornata qualche spreco in più lo si avrebbe sempre.

Messa a punto delle varie attività: task lenti, task bloccanti, task poco importanti

Antonio nella sua piccola bottega mette in atto anche altri **principi** tipici della **produzione snella**. Egli infatti lavora da **solo**, anche se in quel negozio ci sono un **paio** di forni e una sola pressa che spiana le palle di pasta per la pizza.

Questa configurazione non è casuale. La macchina pressatrice infatti svolge un compito molto **semplice** e piuttosto **rapido** che dura solo pochi secondi. Il forno invece è probabilmente la fase della lavorazione più **lunga**, dato che per cuocere una pizza sono necessari circa alcuni minuti a seconda del tipo di pizza. Per questo motivo, nella stanza **una pressatrice** è più che sufficiente mentre disporre di **un solo forno** avrebbe rallentato il processo, creando code in attesa. In realtà dopo qualche tempo Antonio si è potuto permettere un forno più grande, in grado di cuocere fino a dieci pizze contemporaneamente, cosa che, di fatto, permette non solo di velocizzare il tempo di preparazione di una singola pizza, ma di eliminare ingolfamenti in entrata al forno. In questo modo egli può smettere di preoccuparsi della gestione delle varie fasi intermedie e concentrarsi sull'unico punto veramente importante: il numero di pizze pronte ed esposte nella vetrina.

Piccoli lotti di lavorazione (one piece flow)

Una cosa che Antonio gestisce con la massima attenzione è la fase più delicata della lavorazione, la **cottura**: essendo la fase più lunga, e la più delicata, dell'intero processo, è molto importante non trascurarla. Per questo motivo egli spesso sacrifica altre fasi della lavorazione: sia che stia guarnendo una base di pasta, sia che stia servendo un cliente, spesso si interrompe per infornare o per togliere una pizza pronta (**task lento** ad **alta priorità** perché **bloccante** sugli altri).

In realtà Antonio gestisce quest'alternanza, per esempio fra il servire un cliente o interrompersi per infornare una pizza, in modo estremamente attento; come ogni commesso che lavora al pubblico, conosce bene il valore di ogni singola fase del suo lavoro: per esempio interrompere il condimento di una pizza ha un impatto diretto solo sul processo di lavorazione. Sospendere momentaneamente il servizio di un **cliente** è invece una cosa che potrebbe costare un prezzo maggiore in termini di immagine o di soddisfazione della clientela.

Dato che sa intrattenere con simpatia i propri clienti, può permettersi di interrompere il servizio al bancone per cuocere una pizza; Antonio sa che però può farlo solo per pochi istanti. Se per esempio decidesse di seguire la lavorazione di cinque pizze per volta, tutte le fasi sarebbero più impegnative in termini di tempo e quindi anche le interruzioni al servizio al bancone diventerebbero intollerabili per la clientela.

Oltre al problema della gestione della clientela, l'esperienza ha ormai insegnato ad Antonio, che preparare più pizze contemporaneamente non è conveniente: a volte si creano "ingorghi" durante la lavorazione delle pizze, per esempio in entrata del forno, oppure ci sono fasi della lavorazione completamente scariche (p.e., vetrina vuota). Antonio sa quindi che il trucco è gestire **lotti di lavorazione piccoli**, per esempio una o due pizze per volta, limitando il numero di attività che svolge in parallelo. In questo modo, nel tempo ha scoperto che la produzione si velocizza e la clientela è più contenta. Senza saperlo, Antonio, limitando il numero di operazioni svolte in parallelo e gestendo quindi lotti piccoli, sta applicando uno dei principi fondamentali della **produzione snella** messa a punto in Toyota.

La legge di Little

Infatti, mella definizione del Toyota Production System, grazie al lavoro di Taiichi Ōno si comprese l'importanza di **velocizzare il flusso** della lavorazione riducendo il più possibile i lotti di produzione. Gli studi sulla teoria delle code e più precisamente della "legge di Little" confermano la bontà di queste idee.

John D.C. Little è un fisico statunitense noto fra le altre cose per aver formulato la legge che da lui prende il nome: "il numero medio di clienti in un sistema è uguale al tasso medio di arrivo moltiplicato per il tempo medio nel sistema".

Riportando il discorso nell'ambito di un processo produttivo si potrebbe dire che il numero di **pezzi in lavorazione** (**WIP**, *work in progress*) è dato dal **tempo di attraversamento** (**Ta** , ossia il tempo è necessario per lavorare un pezzo) moltiplicato per un coefficiente di **prestazionalità** (**Th**, il *throughput* del sistema, ossia quanto si è veloci nel produrre un singolo pezzo). Questa relazione è esprimibile secondo l'equazione:

```
WIP = Th * Ta
```

Tornando all'esempio della pizzeria, normalmente un cliente che entra nella pizzeria di Antonio vorrebbe essere servito prima possibile; analogamente, ad Antonio interessa ridurre il tempo di lavorazione in modo che sia possibile limitare il **magazzino** (numero di pizze pronte) per evitare sprechi (pizze fredde o avanzi a fine giornata). Dato che la filiera è corta e non ci sono dipendenze da sistemi esterni, in questo caso **tempo di attesa** e **tempo di lavorazione** sono praticamente (quasi) la stessa cosa. Nei casi più complessi sarebbe necessario aggiungere un fattore correttivo, ma la sostanza non cambia: si deve trovare un modo per **ridurre** al minimo possibile il `Ta`. Ribaltando la formula di prima si ottiene che:

```
Ta = WIP / Th
```

ossia un `Ta` piccolo si può avere sia aumentando le prestazioni, sia **riducendo** il numero di cose che **contemporaneamente** sono **in lavorazione**.

Aumentare le prestazioni del sistema non sempre è cosa facile, mentre diminuire la dimensione del lotto è certamente realizzabile. Ridurre al minimo il lotto di lavorazione porta a quello che nel sistema Toyota viene chiamato **one piece flow**: si produce **un pezzo per volta** sulla base delle effettive richieste che arrivano dal campo.

Lo *one piece flow* spesso non è realizzabile a causa della variabilità del sistema (variazioni della domanda, dipendenze da linee di lavorazioni esterne, setup operativi); Per questo spesso si introducono delle aree di sosta: sono i cosiddetti **buffer**, elementi in grado di assorbire tale variabilità.

Le aree di sosta nel processo: i buffer e la gestione delle code

Gran parte delle buone pratiche che Antonio riesce a mettere in atto nel suo negozio (**pull**, **just in time**, **WIP limit**) sono possibili grazie a un altro importante elemento: le zone di stazionamento rappresentate nel nostro esempio dal piano di lavoro dove prepara e appoggia i semilavorati, o dalla vetrina delle pizze pronte.

Nella terminologia **Lean** queste aree sono dette **buffer di processo**, ossia vere e proprie "aree di parcheggio" in grado di assorbire gli sbalzi del processo di lavorazione o di domanda. Il **mura**, che in giapponese significa appunto "sbilanciamento", in Lean è una delle forme di spreco più gravi.

Grazie all'uso di tali aree di sosta, si può realizzare un processo in modalità **pull**: quando termina la lavorazione di un **semilavorato** di prodotto — nel nostro esempio potrebbe essere la pizza stesa ma non ancora farcita — questo viene appoggiato nella zona di parcheggio — il piano di lavoro vicino al forno — in attesa che sia preso ("tirato", *pull*) e messo in lavorazione per la fase successiva.

Inoltre, per mezzo dei **buffer** è possibile gestire eventuali dipendenze da fasi esterne del processo. Per esempio, una pizza con prosciutto crudo deve essere guarnita dopo la cottura: al termine del tempo di cottura, Antonio può estrarre la pizza dal forno, appoggiarla su un piano e guarnirla con calma per esempio dopo aver infornato altre pizze. Se lasciasse la pizza in forno si brucerebbe, se si mettesse a guarnirla prima di infornarne altre, sprecherebbe del tempo prezioso, dato che la cottura è un processo lento e autonomo.

La **dimensione** di tali **buffer** — per esempio il numero di pizze sul piano prima del forno, o delle pizze cotte ed esposte in vetrina — è una variabile molto importante: un **buffer** troppo piccolo (poche pizze in vetrina pronte per essere vendute) rischia di esaurirsi prima che Antonio abbia il tempo di preparare altre pizze; se il buffer invece è troppo grande, il pericolo è quello di non riuscire a vendere tutte le pizze esposte prima che queste perdano la loro freschezza e si rovinino. Normalmente, sia per l'esempio delle pizze che nei processi di produzione più complicati, il valore ottimale si ottiene dopo alcuni **esperimenti** sul campo; ma torneremo su questo aspetto quando parleremo di progettazione della **kanban board**.

Analogamente decidere il numero e la posizione dei buffer nella catena di lavorazione non è banale: in questo caso, essendo la lavorazione è a **catena corta**, possono bastare

due soli buffer lungo la linea di lavorazione per stabilizzare il processo. Nei casi più complessi, i processi di produzione si avvalgono di uno o più **stazioni di parcheggio**.

In ottica di miglioramento del processo, informazioni utili si possono ricavare dal conteggio del **numero** di **attività parcheggiate o dalla misurazione** della loro **permanenza nelle aree di parcheggio**. In linea di principio, buffer troppo affollati o attività parcheggiate da troppo tempo, sono sintomi di un sistema sub performante.

Kanban quotidiano: dal giardino imperiale alle code in autostrada

Per iniziare a parlare di Kanban in modo più specifico, possiamo prendere in esame uno scenario completamente diverso, ma molto esplicativo, che probabilmente ogni lettore avrà vissuto in prima persona nella sua vita di tutti i giorni: lasciare l'auto in un parcheggio a pagamento in cui **entrata** e **uscita** siano **regolati** da sbarre automatiche, sistemi di pagamento e, soprattutto, **posti limitati** e **numerati**.

Se il numero di automobilisti che desiderano parcheggiare è ben al di sotto della capienza del parcheggio, in genere nessuno si accorge del meccanismo che regola il parcheggio stesso: si arriva, si preleva un biglietto alla macchinetta, la sbarra si alza e si procede verso un posto libero.

Se invece il parcheggio ha raggiunto la **capacità massima**, normalmente all'entrata si formano delle **code** di auto in attesa di poter entrare nel parcheggio. In questo caso, essendo il numero di posti disponibili limitato dalla capienza massima del parcheggio, per **ogni** automobilista che **lascia** il parcheggio si **libera** un **posto** per la vettura di un altro che vuole entrare.

Questa organizzazione è possibile perché un semplice **contatore** tiene traccia delle auto che entrano e di quelle che escono e blocca il rilascio di nuovi biglietti quando si arriva a un certo limite.

Tale sistema si basa su un meccanismo inventato molti anni fa presso il parco del Palazzo Imperiale giapponese dove, in primavera, moltissime persone accorrono per poter assistere al bellissimo spettacolo della fioritura dei ciliegi; per garantire l'armonia e la bellezza del parco è necessario limitare il numero di visitatori contemporaneamente presenti nel giardino, cosa che fin dai tempi antichi viene ottenuta con un meccanismo estremamente semplice ma altrettanto efficace.

Pur essendo l'ingresso al parco gratuito, all'entrata ogni visitatore deve prelevare da un contenitore un "gettone", una **tessera visuale**: questo è il significato del termine giapponese **kanban** (scritto con la "k" minuscola) che vuol dire appunto, "cartellino", "tessera".

Nel giardino imperiale ogni visitatore deve tenere con sé questa **tessera** per tutto il tempo in cui rimane dentro il parco. Quando esce, dovrà riporre la sua **kanban** nel contenitore apposito. Quando non ci sono tessere **kanban** nel contenitore, nessun visitatore può più entrare nel parco: avere la tessera in mano è la condizione per poter entrare. Non appena un visitatore esce, depone il suo "gettone" kanban e un altro visitatore può riceverlo ed entrare a sua volta. Semplice. Lineare. Ripetibile.

In Toyota, fin dagli anni Cinquanta del secolo scorso, si è potuta tenere sotto controllo la produzione delle automobili in fabbrica proprio grazie a questo metodo di gestioe del flusso: contenitori e **kanban cards** (dei cartellini plastificati), senza l'ausilio di complessi sistemi informativi.

Le code in autostrada

Un'altra interessante esperienza che ogni lettore ha sperimentato almeno una volta è quella che si vive quando si percorrere in **automobile** un tratto di una **autostrada** particolarmente congestionato. In questo caso si può facilmente osservare che maggiore è il numero di auto che percorrono quel tratto, minore sarà la velocità con la quale si potrà percorrere quel tratto.

A volte tali rallentamenti portano a improvvisi e temporanei, quanto misteriosi, blocchi della circolazione del traffico, che poi magicamente riprende a muoversi senza un apparente motivo: non ci sono incidenti, non ci sono blocchi della viabilità. Questo fenomeno in gergo si chiama **Phantom Work Jam** o "ingorgo fantasma" (concetto approfondito al Capitolo 7 del libro di Stephen Denning [RM]).

Il **Phantom Work Jam**, oltre che essere legato al numero di auto che transitano per un determinato tratto di strada, è anche funzione della velocità di percorrenza media: **numero** di **auto** in **transito** e **velocità media** sono infatti due fattori che stressano il sistema in modo diretto.

Anche in questo caso il fenomeno è spiegabile applicando la legge di Little: essendo costante l'ampiezza delle corsie dell'autostrada (ossia il troughtput `Th`, in questo caso detto anche "capacità di canale"), maggiore è il numero di auto in transito (`WIP`), maggiore sarà il tempo di attraversamento (`Ta`), ossia si viaggia più lentamente. Più ci si avvicina al **limite massimo del sistema**, maggiore sarà la probabilità che si verifichino ingorghi.

Dato che un'autostrada può essere assimilata a un processo di produzione molto lungo (filiera lunghissima), nei momenti di criticità entra in gioco un altro fattore, il cosiddetto **effetto Forrester** (o "effetto frusta" o *bullwhip effect*) di cui si parla nell'Appendice A.

Per semplicità possiamo dire che quando siamo al limite della capacità del sistema, piccole oscillazioni si propagano amplificandosi lungo tutta la filiera. Un piccolo rallentamento di un'auto si trasmette aumentando il suo effetto, fino al punto in cui alcuni automobilisti sono costretti a una brusca frenata o al blocco del veicolo. Forrester ci dice poi che per smaltire l'ingorgo serve molto più tempo di quello che è necessario per la sua creazione.

Nel caso del traffico stradale, non essendo possibile nella maggior parte dei casi limitare il numero di auto in transito, un buon modo per evitare ingorghi fantasma è ridurre il limite di velocità; in tal senso alcuni interessanti esperimenti alla viabilità sono stati condotti in Gran Bretagna.

Breve storia di Kanban: dai cartellini al Project Management

Verso l'inizio degli anni Quaranta del secolo scorso, gli ingegneri di Toyota stavano lavorando all'ottimizzazione dei propri processi di produzione. **Taiichi Ōno**, anche prendendo spunto dal lavoro di William Edwards Deming, si impegnò per introdurre nuovi principi di quella che sarebbe diventata poi la **produzione snella**: ne abbiamo parlato diffusamente nella prima Parte di questo libro, *Verso Agile*.

Una delle fonti che dette maggior ispirazione a Ōno arrivò da un settore inaspettato, quello della grande distribuzione alimentare. In quel periodo stavano nascendo i primi supermercati dentro i quali i commessi e gli addetti al rifornimento del magazzino dovevano trovare soluzioni ottimali proprio per gestire merci con un così alto tasso di degradazione: di sicuro, un cesto di insalata impiega molto meno tempo per perdere di "freschezza" rispetto a quello necessario a un modello di automobile per diventare obsoleto...

In quel caso, il flusso di approvvigionamento era guidato dal livello di rifornimento a scaffale: **solo** quando l'**ultimo elemento** di un certo prodotto era prossimo alla vendita, si procedeva al riordino; non prima. Furono quelli i primi esempi di gestione **just-in-time**, che ispirò i manager giapponesi della Toyota nel mettere a punto un nuovo processo di approvvigionamento.

Taiichi Ōno fu impressionato anche dall'elevato livello di efficienza di un supermercato in termini di risorse impiegate: in genere sono necessari solo pochi commessi, grazie al **pull** (i clienti svuotano gli scaffali con la merce) e al **just in time**.

Portando in Toyota le idee di Deming e tramite la **gestione** in tempo reale dei **flussi** di **entrata** e di **uscita** dei prodotti, nella fabbrica Toyota riuscirono a migliorare le prestazioni complessive del sistema: stava nascendo un nuovo modo di lavoro. Per chi fosse interessato a leggere tutta la storia della nascita del **Lean** e del **Toyota Production System**, si consiglia la lettura dei due testi più importanti [TW] e [MCW].

Segnale visivo

Il principio che animò questo processo di trasformazione è descrivibile con un semplice slogan: "migliorare la comunicazione tramite una gestione visuale del processo".

La parola **kanban** indica infatti una "segnalazione visiva", un "cartello" e infatti buona parte della metodologia, quando fu creata, faceva uso di **cartellini** visuali e di **lavagne** dove questi cartellini erano attaccati per monitorare gli **stati** di **avanzamento** della produzione, l'**approvvigionamento**, l'acquisto o la movimentazione dei **materiali**.

Il sistema, in modo estremamente naturale, stimolava il confronto e il dialogo all'interno del team, velocizzando lo scambio delle informazioni e agevolando quindi il processo decisionale sul cosa e come organizzare il lavoro. Fra gli obiettivi che Ōno e il suo staff si dettero durante il lavoro di messa a punto del Toyota Production System, vi era quello di velocizzare il flusso della lavorazione, limitare l'accumulo delle merci in entrata o dei prodotti finiti in uscita, di stabilizzare il flusso massimizzando le performance.

Kanban come metodologia di gestione e controllo

A partire dal 2007, **Kanban** (con la "K" maiuscola per distinguerlo dal cartellino) è stato introdotto come **metodologia** di **gestione** e **controllo** dello **sviluppo** del **software** grazie al lavoro di David J. Anderson, il quale si è ispirato ai principi del Lean e del Toyota Production System messo a punto da Ōno in oltre venti anni di lavoro. Anderson ha formalizzato le sue idee in un libro [SEC] pubblicato nel 2010.

Caratteristiche di Kanban

Nel suo lavoro, Anderson ha analizzato i principi della produzione snella e, sulla base di esperienze personali e di confronti con altre metodologie di gestione, ha tentato di adattare i principi della **produzione snella** alla **gestione** dello **sviluppo software**, integrando le sue idee nel più ampio quadro delle metodologie agili.

I cinque aspetti fondamentali

Nel libro, Anderson ha sintetizzato in **cinque punti** gli aspetti più importanti di Kanban:

- Visualizzare il flusso di lavoro: dare una rappresentazione visiva del processo di lavorazione, ponendo in risalto, per ogni step del processo, il valore prodotto.
- Limitare il Work-in-Progress (WIP): porre dei limiti espliciti sul lavoro eseguibile all'interno di ogni fase della lavorazione.
- Misurare e gestire il flusso: misurare e gestire il flusso per avere sempre informazioni attendibili e poter quindi prendere decisioni coerenti col sistema; per ogni azione intrapresa, visualizzare sempre le conseguenze.
- Esplicitare le regole del processo: è di fondamentale importanza la condivisione e il rispetto delle regole del processo da parte di tutti gli interessati; secondo Anderson questo è il modo più semplice ed efficace per ottenere gli obiettivi di processo, pur garantendo il miglioramento delle prestazioni e la riduzione degli sprechi.
- Identificare le possibili opportunità di miglioramento: creare nell'organizzazione una cultura in cui il continuo miglioramento del sistema, del processo e delle performance siano obbiettivi condivisi in tutti i membri della comunità; in giapponese, tale cultura del miglioramento continuo è nota con il nome di kaizen, parola ricorrente nel lavoro di Taiichi Ōno durante la creazione del Toyota Production System.

Tre regole importanti

Questi cinque punti possono essere considerati i cinque obiettivi fondamentali che dovrebbero essere perseguiti tramite **tre regole importanti**, che sono alla base di Kanban:

- Usare gli strumenti di cui si è già in possesso: partire da quello che si sa fare senza aspettare di avere imparato cose nuove.
- Condividere un percorso di cambiamento che abbia come obiettivo la crescita e l'evoluzione dell'organizzazione.

- Rispettare l'organizzazione attuale: il processo di cambiamento ed evoluzione non deve imporre stravolgimenti drastici nel processo, modifiche traumatiche nei ruoli e nelle persone, ma introdurre le novità in modo graduale e compatibile con le necessità e gli obiettivi. Per questo sono così importanti la visualizzazione del processo, la misurazione, la definizione esplicita delle policy.

Molte delle indicazioni fin qui viste sono prese per la maggior parte dai principi del **Lean** che **Kanban** sposa in pieno: si potrebbe quasi dire che l'obiettivo principale di **Kanban** è quello di funzionare come **attivatore** dei **principi Lean** all'interno dei sistemi di **produzione** del **software**. Taiichi Ōno era solito dire: "Lo scopo del kanban è di far emergere i problemi e metterli in relazione con l'attività di miglioramento continuo".

Conclusioni

Attraverso un certo numero di esempi relativi ad attività quotidiane e facilmente comprensibili, si sono introdotti i concetti essenziali di un sistema di produzione basato su principi Lean.

Abbiamo poi raccontato brevemente la storia dei cartellini kanban adottati inizialmente nel Toyota Production System e di come molti dei principi Lean utilizzati nella produzione industriale siano stati adattati al mondo della gestione dello sviluppo del software grazie al metodo Kanban, messo a punto da D. J. Anderson a partire dalla fine del decennio passato.

Nei prossimi capitoli affronteremo la metodologia Kanban sul piano operativo, illustrandone pratiche e strumenti.

Riferimenti

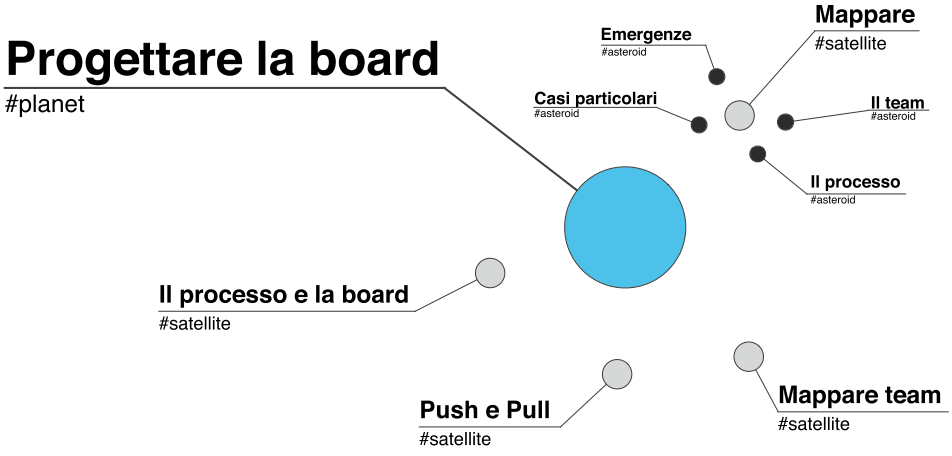
[TW] Toyota Way. 14 principi per la rinascita del sistema industriale italiano
http://www.toyota-way.it

[MCW] Womack J.P. – Jones D.T. – Roos D., *The Machine That Changed the World: The Story of Lean Production*, Harper Business, 1991 (trad. it. *La macchina che ha cambiato il mondo*, Rizzoli, 1999)

[SEC] David J. Anderson, *Kanban: Successful Evolutionary Change for Your Technology Business*, Blue Hole Press, 2010

[RM] Stephen Denning, *The Leader's Guide to Radical Management: Reinventing the Workplace for the 21st Century*, Jossey-Bass, 2010

Capitolo 2
Progettare e usare la kanban board



Kanban in pratica

In questo capitolo tramite una serie di esempi visuali, verrà mostrato come progettare e come utilizzare una **lavagna kanban**: lo scopo è quello di fornire delle indicazioni pratiche su come introdurre la metodologia Kanban all'interno della propria organizzazione.

Progettare la prima kanban board

Un modo per realizzare la lavagna kanban è quello di iniziare dalla raccolta su una **parete** o su un **pannello** di una serie di **cartellini** corrispondenti alle attività che compongono il processo che si vuol gestire con Kanban. L'uso di uno spazio ampio e libero da schemi predefiniti può aiutare a comprendere meglio lo stato attuale del lavoro all'interno dell'organizzazione (figura 2).

Successivamente si può procedere provando a schematizzare il **processo di lavorazione**, per esempio provando a raggruppare i cartellini corrispondenti ad attività simili. Un'altra strada potrebbe essere quella di ordinare i cartellini in base a una qualche regola: per esempio in alto quelli che dovranno essere presi in lavorazione per primi. Fra le molte possibilità, una pratica piuttosto comune è quella di iniziare la scomposizione della lavagna in colonne definendo tre aree: quella corrispondente alle attività ancora da svolgere, quelle in lavorazione e quelle già terminate (figura 3).

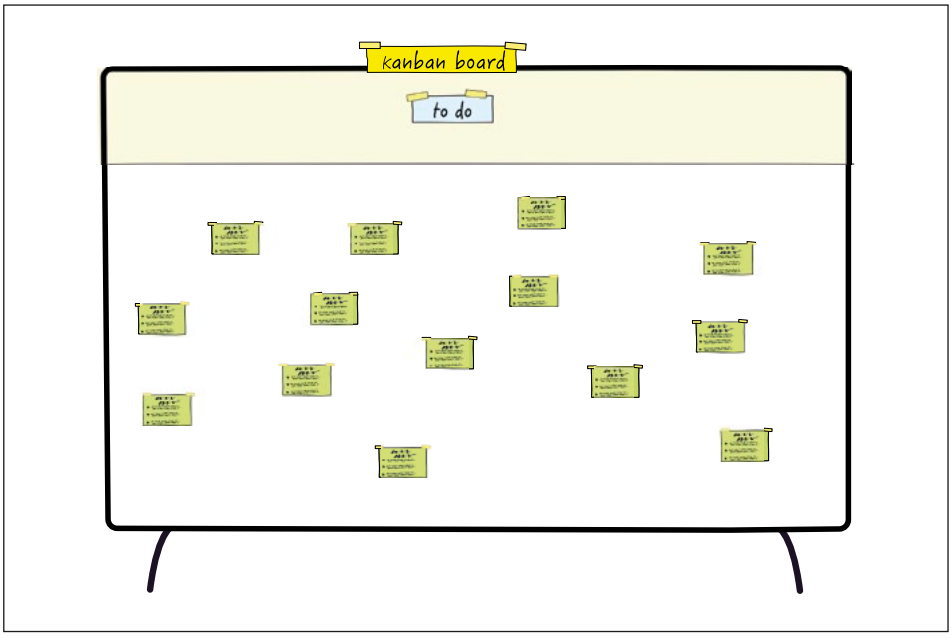


Figura 2. Per prima cosa si dispone in modo libero su una parete l'elenco delle attività che si dovranno svolgere.

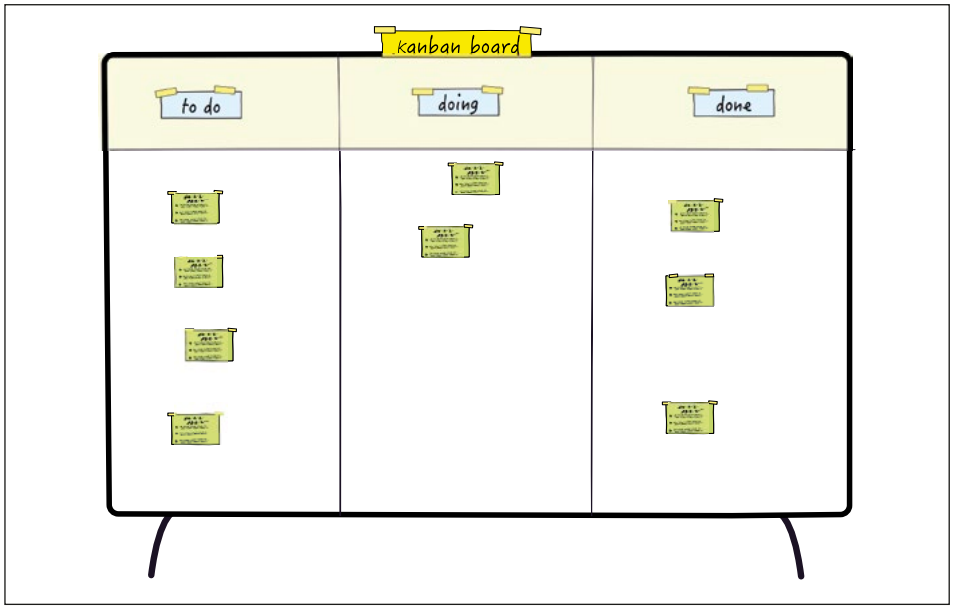


Figura 3. Il primo miglioramento che si può introdurre è una suddivisione in colonne per raggruppare le attività da svolgere, quelle in lavorazione e quelle che invece sono già terminate.

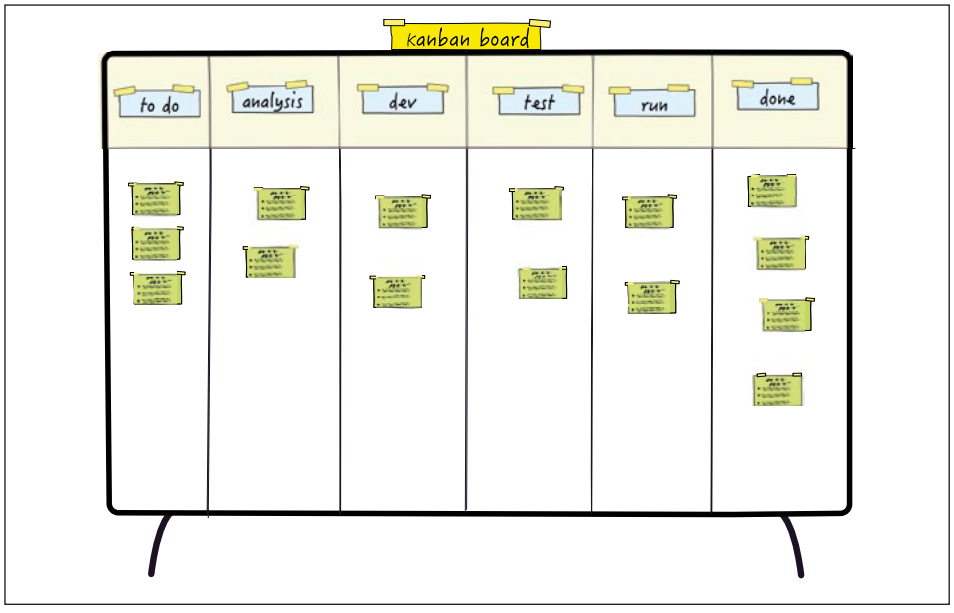


Figura 4. La colonna della lavorazione può essere scomposta nella corrispondenti attività di dettaglio. In un processo di sviluppo software, potrebbero essere le classiche analisi, sviliuppo, test e deploy.

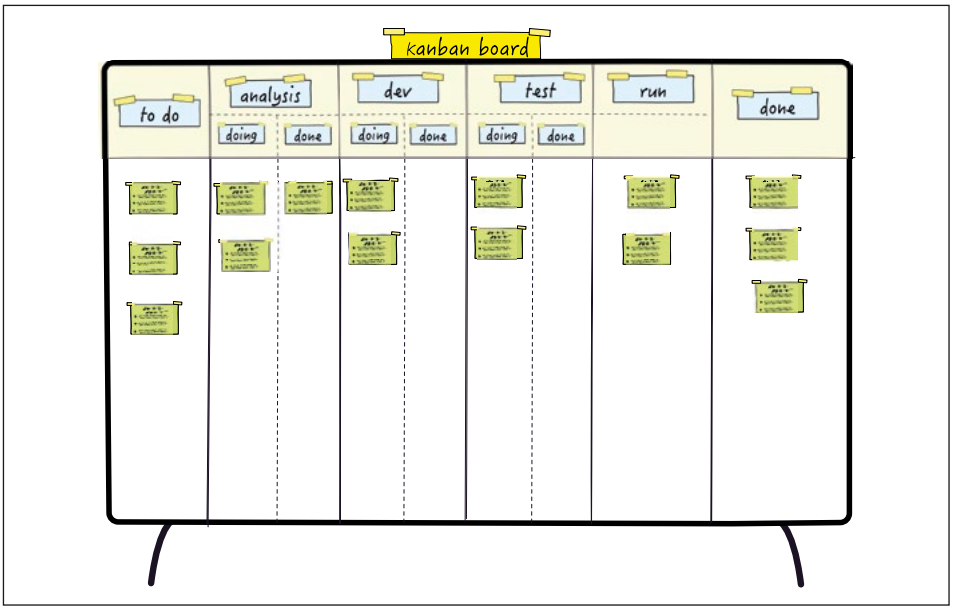


Figura 5. Il passo successivo potrebbe essere quello di introdurre le colonne di parcheggio, in modo da gestire il passaggio da un approccio push a uno pull.

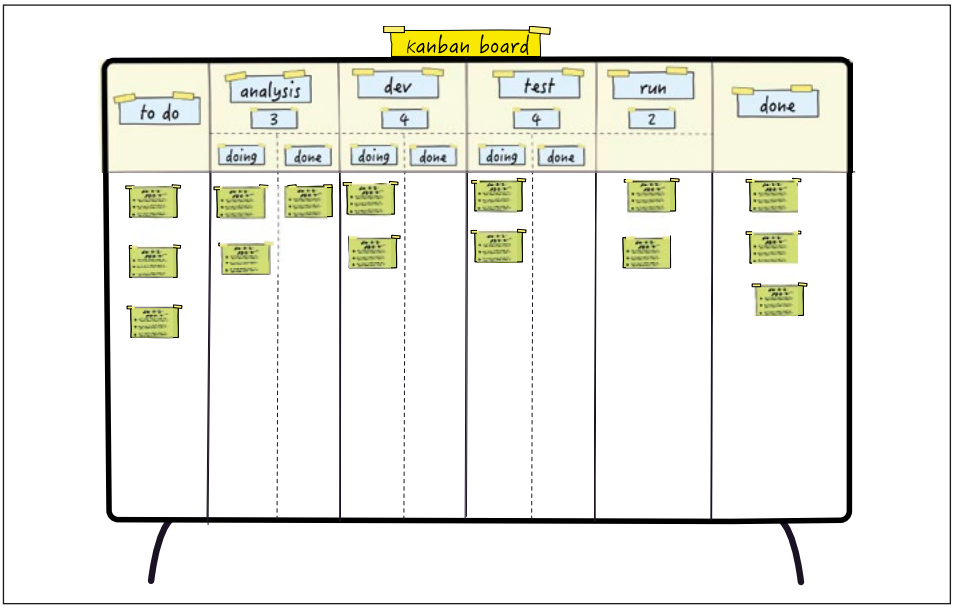


Figura 6. Come ultimo raffinamento si possono introdurre i limiti WIP (Work In Progress) sul numero massimo di elementi in lavorazione contemporanea.

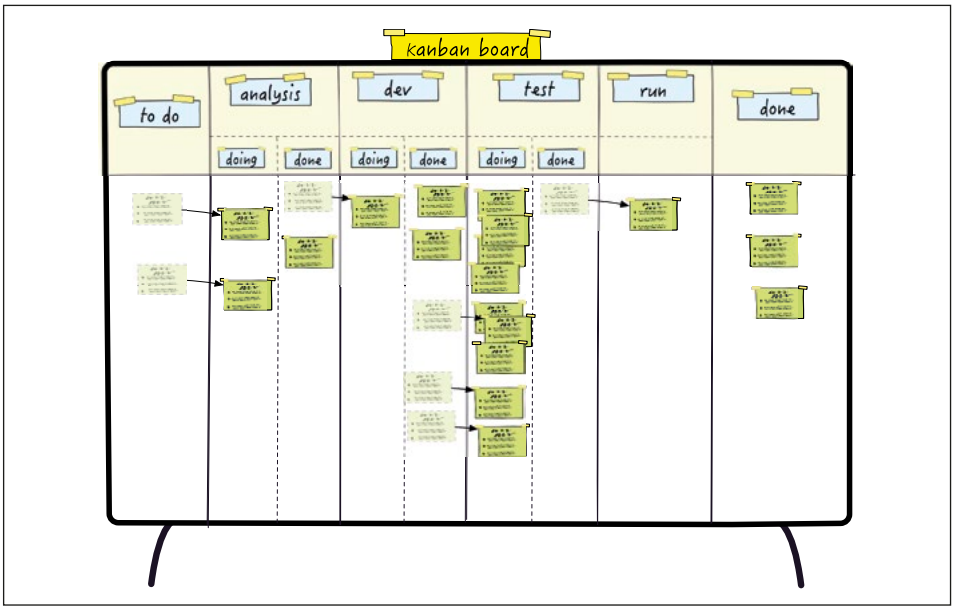


Figura 7. Le attività svolte in contemporanea (Work In Progress, WIP) non devono essere troppe. Se non si impone alcun limite di lavorazione (WIP limit), vi è una elevata probabilità che si verifichino degli ingorghi prima delle fasi lente (colli di bottiglia).

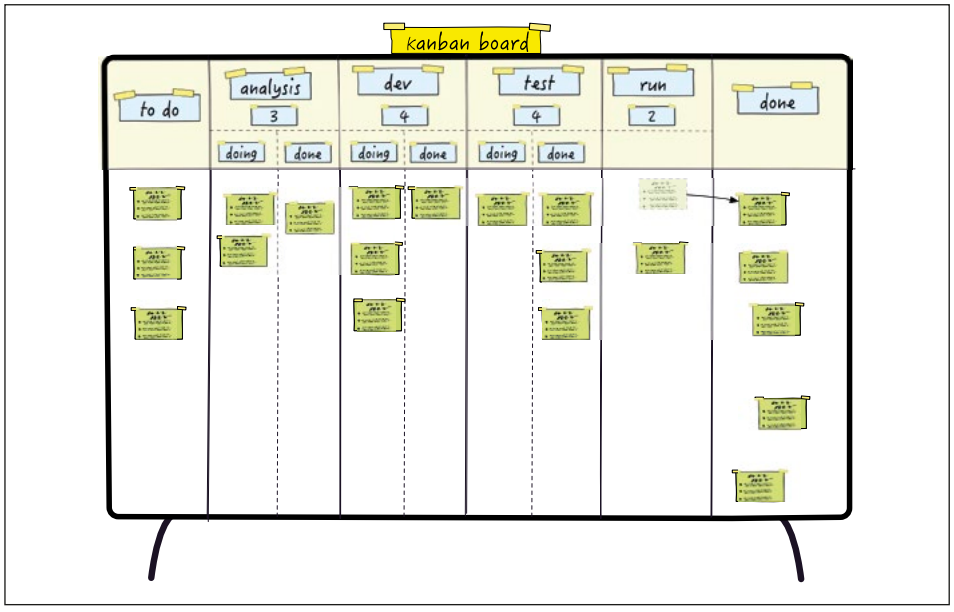


Figura 8. I limiti di lavorazione favoriscono l'approccio pull, evitando l'insorgere di colli di bottiglia delle attività.

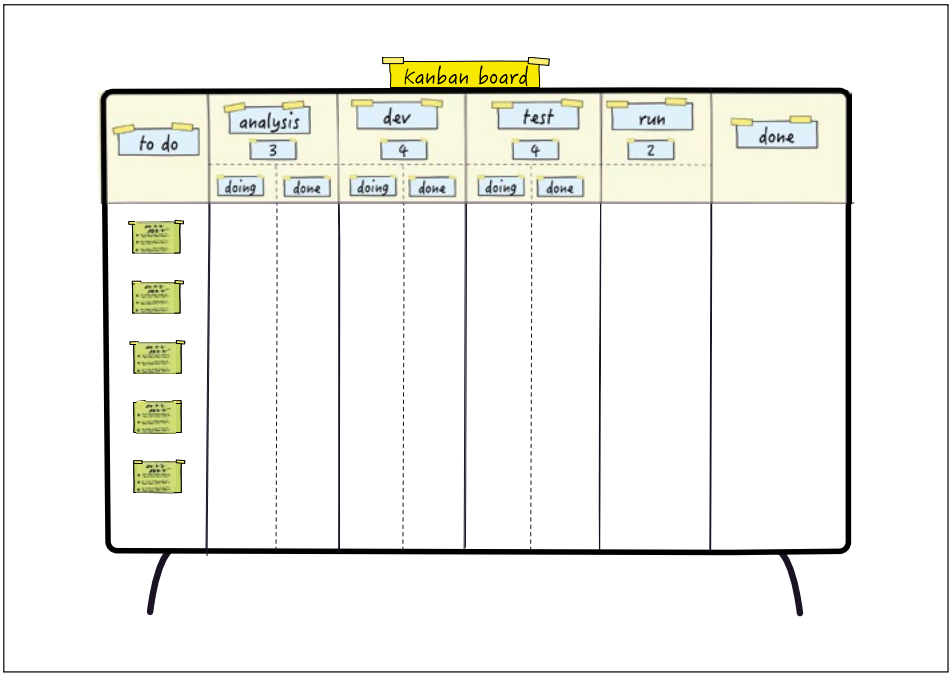


Figura 9. Situazione iniziale: ancora nessuna attività in lavorazione.

Successivamente per **dettagliare** ulteriormente il processo, si potrebbe **suddividere** la colonna corrispondente alla lavorazione, in modo da specificare le varie **fasi del processo**. Nella figura 4 per esempio sono state create le colonne corrispondenti alle tipiche fasi di un processo di sviluppo software: analisi, sviluppo, test e, infine, deploy (in questo caso identificato dall'etichetta **run**).

Nella figura 5 si nota l'introduzione delle colonne di **buffer** (o **code**) necessarie per "parcheggiare" le attività per le quali si è terminata la lavorazione relativa alla colonna corrente in attesa che siano messe in lavorazione nella fase seguente, il che comporta lo spostamento nella colonna successiva.

Infine, sulle varie fasi del ciclo di lavorazione si possono introdurre dei **limiti** al numero di attività (**WIP limit**) che sono eseguibili in contemporanea (figura 6).

In assenza di un **WIP limit** o in presenza di un valore non adatto, data la naturale propensione che molte persone hanno nell'iniziare nuove cose prima di aver terminato quelle in corso, si potrà notare un accumulo di cartellini in determinate posizioni sulla board, ossia dei veri e propri **ingorghi** di attività non completate

Nella board raffigurata nella figura 8 la presenza di un **vincolo imposto** sulle **colonne** favorisce — di fatto, obbliga — lo "scodamento" (approccio pull) dei cartellini sulle colonne di destra, impedendo quindi l'accumulo visto in precedenza in figura 7; in questo modo si ottiene una lavorazione più omogenea delle varie attività.

Simulazione di un flusso di lavorazione

Passiamo adesso a **simulare** la **gestione** di un tipico **processo** di lavorazione tramite una board kanban. Per fare questo si partirà con una situazione iniziale come quella raffigurata in figura 9, dove ancora nessuna attività è stata presa in carico per la lavorazione.

NOTA: il flusso che andremo a disegnare sulla board ricalca le tipiche fasi di un processo sequenziale che ricorda molto il modello a cascata (analisi, sviluppo, test e deploy). Questa scelta ha solo finalità di semplicità espositiva.

Il primo passo (figura 10) è quello di mettere **in lavorazione due attività** nella prima fase che è quella dell'analisi. Dato che il **WIP** (Work In Progress) **limit** della lavorazione è di **2 task**, questa operazione è consentita.

Non appena è **completata** la **lavorazione** di **uno** dei due task di analisi, questo verrà **parcheggiato** nella colonna delle **cose fatte** (figura 11), pronto per essere preso in lavorazione in sviluppo. In questo momento si libera un **WIP** sulla colonna di analisi per cui un altro task potrà essere messo in lavorazione in analisi.

Proseguendo la simulazione (figure 12 e 13) si osserva che i vari task si **spostano verso destra** dando luogo a un vero e proprio **flusso** di lavorazione.

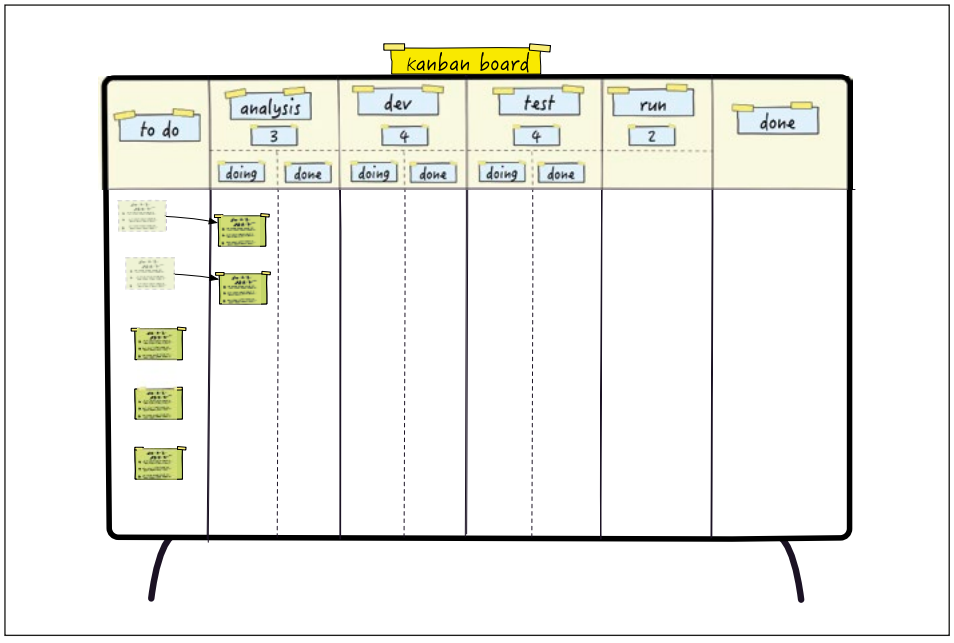


Figura 10. Si spostano verso destra le prime due attività dal backlog degli elementi pronti per la lavorazione.

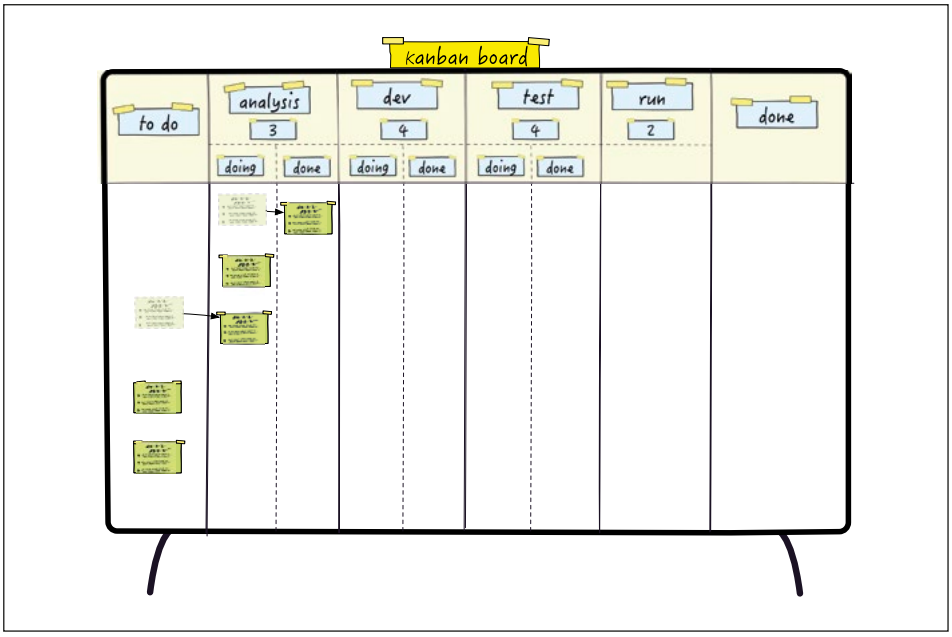


Figura 11. La prima attività ha completato la fase di analisi e viene spostata nella colonna del "done".

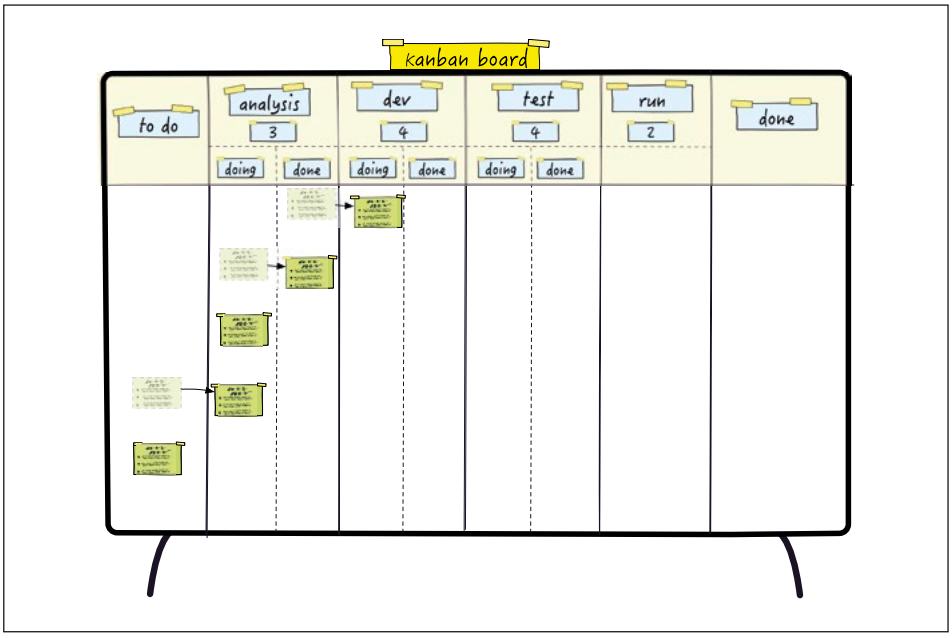


Figura 12. La prima attività viene spostata nella lavorazione "dev".

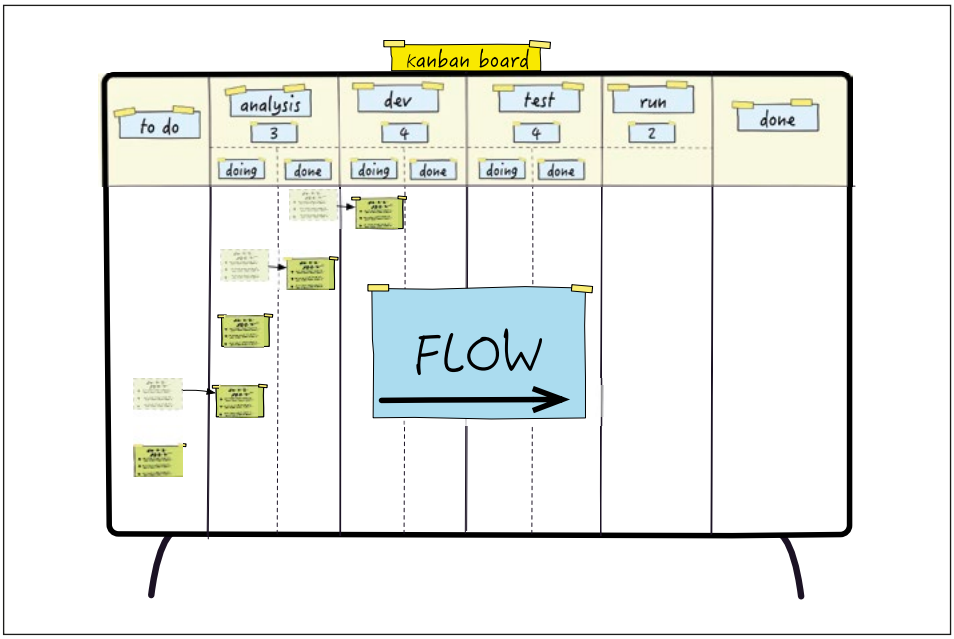


Figura 13. Con questo meccanismo si mette in atto un movimento verso destra che corrisponde al flusso della lavorazione.

Gli avatar del team

Spesso vi è la necessità di avere una visione aggiornata di **chi** sta svolgendo le attività in lavorazione: un modo semplice per ottenere questo obiettivo è quello di associare delle **icone rappresentanti** la persona o le persone che stanno effettuando la lavorazione.

Per le icone i possono utilizzare figure di fantasia, disegni astratti o vere e proprie foto delle persone (figura 14).

Quando una **persona** prende **in carico** un'attività, preleva il proprio avatar e lo **associa** al **cartellino**; viceversa, quando l'**attività** è **terminata**, l'**avatar** viene staccato dal cartellino e rimesso nell'area di **parcheggio**.

L'uso degli avatar permette di verificare in modo semplice e immediato l'impegno delle varie persone del team sulle varie attività, così come di coordinarne lo spostamento quando si presentano attività critiche che richiedano un rapido intervento.

Inoltre, limitando il numero degli avatar disponibili fisicamente sulla board (per esempio mettendone solo 3 o 4 per ogni persona), si può ridurre il numero di cose che ogni membro del team svolge in parallelo.

Se si guarda per esempio la figura 15, notiamo che Andrea è impegnato su tre differenti attività, quindi la sua capacità operativa è al limite massimo: dovrà terminare le attività che ha in carico prima di prenderne altre.

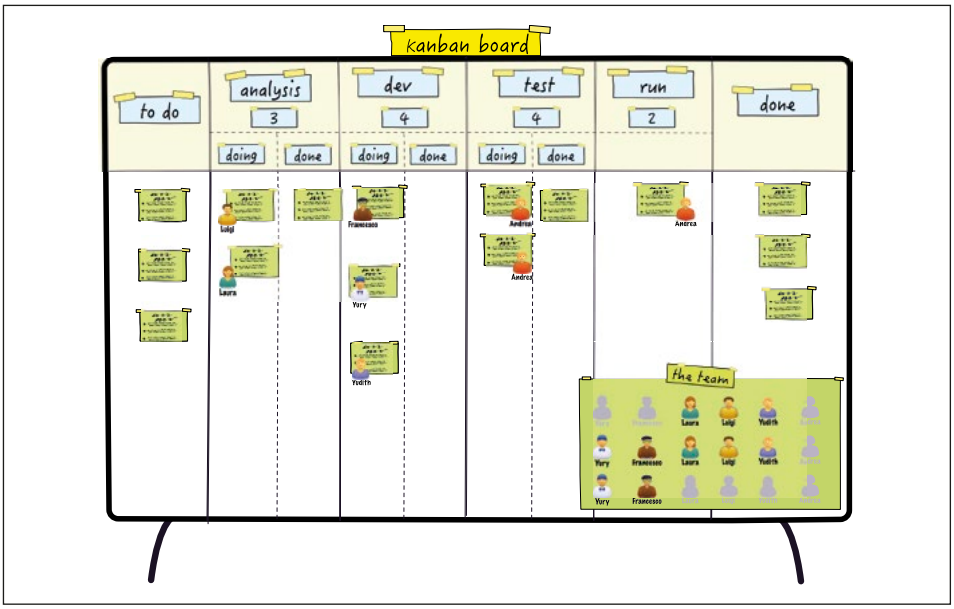


Figura 14. Gli avatar sono molto utili per rappresentare in modo rapido e visuale chi sta facendo cosa. In questo caso una parte della board è utilizzata come deposito di tutte le **icone delle persone** *del team (parcheggio degli avatar).*

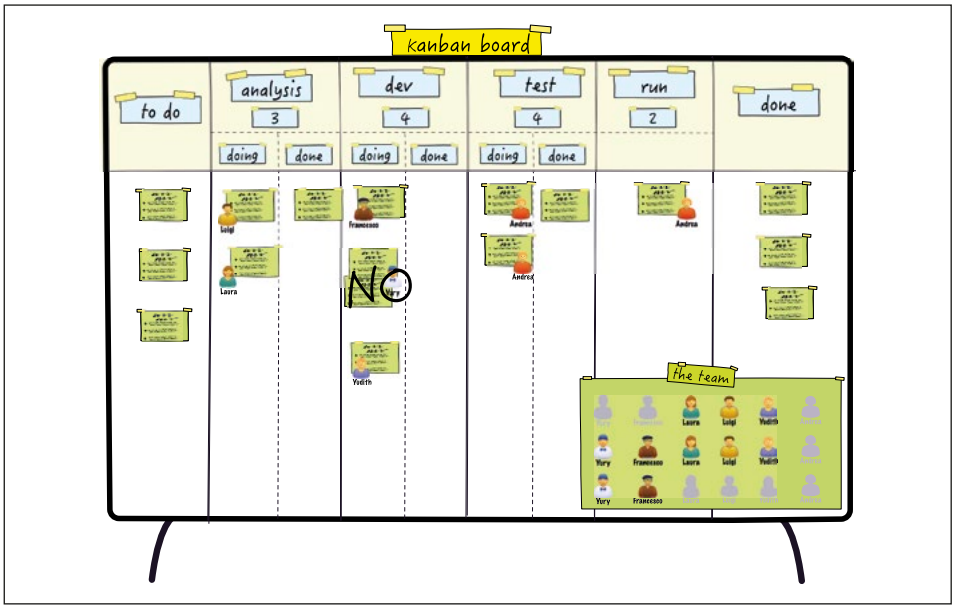


Figura 15. L'area di parcheggio può contenere un numero limitato di avatar per evitare che la singola persona faccia troppe cose in parallelo.

Il numero degli avatar da assegnare a ogni membro del team può essere uguale per tutti, oppure come in realtà accade, è funzione del tipo ti attività che le persone devono svolgere.

Il lavoro in un team cross-funzionale

Nelle figure 16, 17, 18 e 19 viene raffigurata una tipica situazione in cui le persone del team si spostano fra le varie attività o lavorano in coppia per velocizzarne il completamento. Questo tipo di organizzazione si verifica quando il team si dice **cross-funzionale**, ossia quando è composto da persone capaci di svolgere diverse attività previste dal processo, se non addirittura tutte.

Avere team cross-funzionali non deve essere visto come un obiettivo obbligatorio, ma offre certamente molti vantaggi; per esempio migliora l'efficienza, riduce le dipendenze dai singoli — se un membro del team è assente, si possono comunque portare a compimento le attività in carico a lui—, abbassa la pressione sul singolo distribuendo attività ai colleghi, previene i colli di bottiglia e altro ancora.

L'uso degli avatar sulla board permette al team di organizzarsi meglio e di far "fluire" le competenze in maniera adeguata.

Si può comprendere come questo sia possibile per esempio seguendo nella figura 16 e successive, gli spostamenti dell'avatar di Yury, il quale, dopo aver terminato un'attività di sviluppo, sposta il cartellino corrispondente nella colonna del finito.

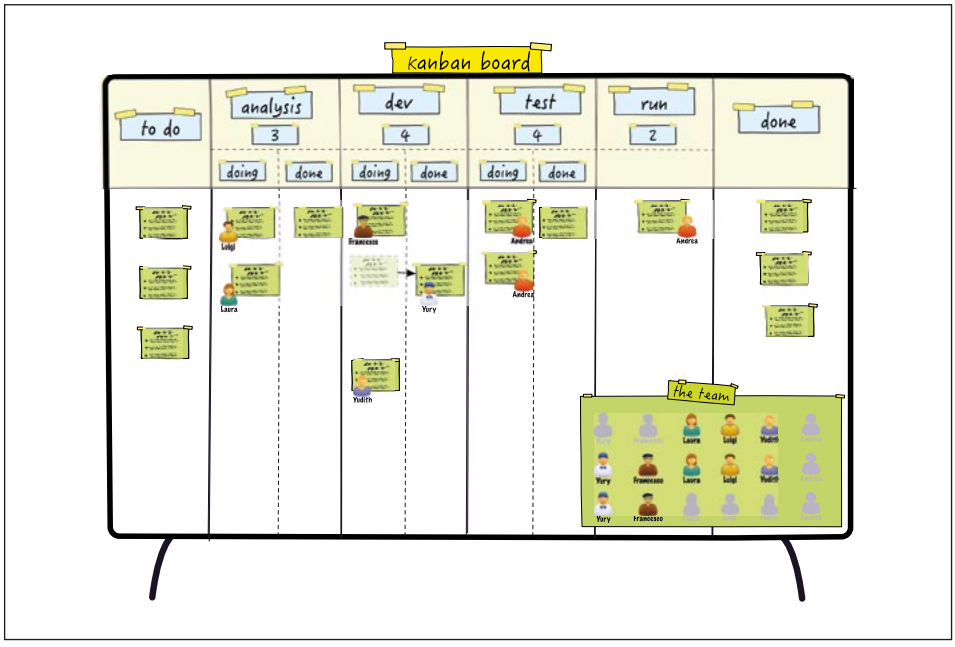


Figura 16. Yury, completato il suo lavoro, si rende libero per fare altro.

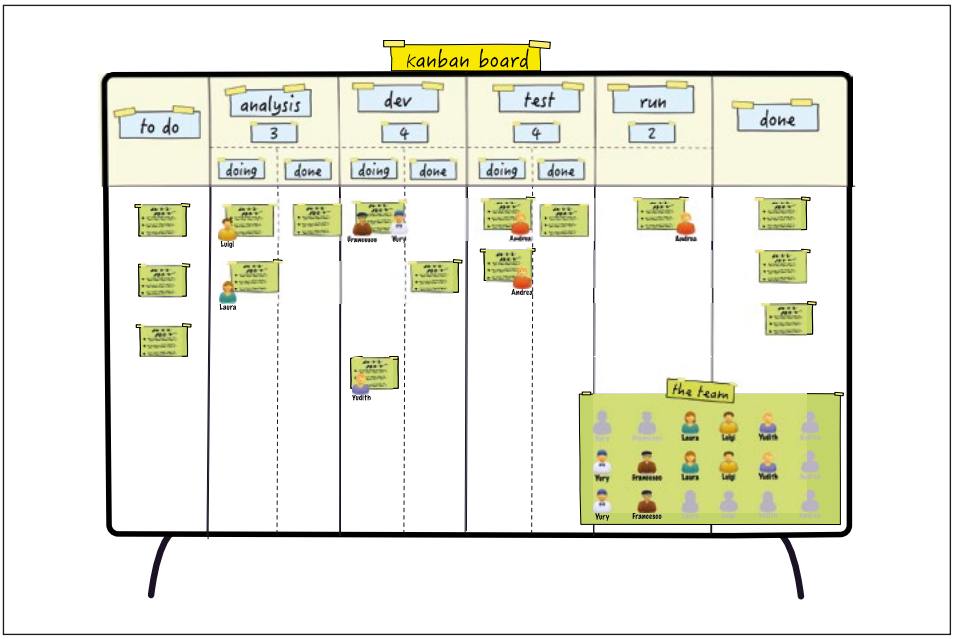


Figura 17. Yury, dopo aver terminato il suo task, non ne inizia uno nuovo, ma si mette in coppia con Francesco per aiutarlo a completare prima il suo lavoro.

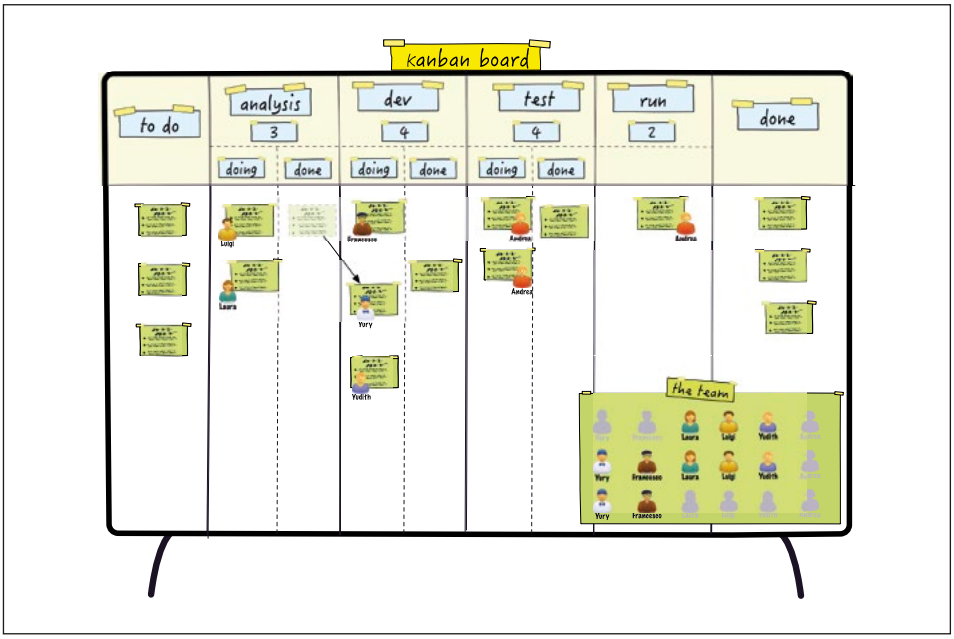


Figura 18. In alterntativa Yury invece potrebbe mettersi a lavorare un nuovo task.

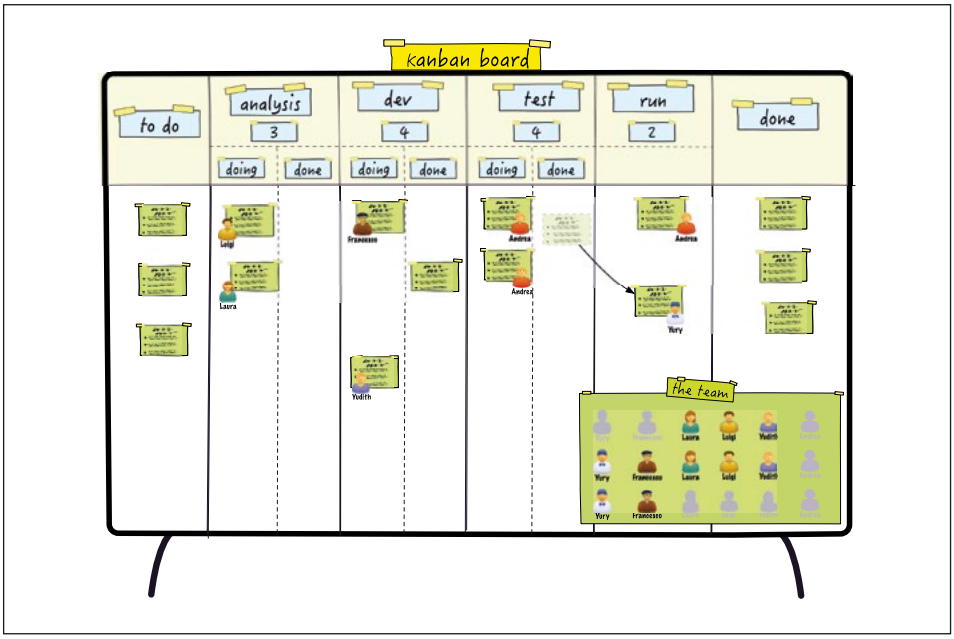


Figura 19. Se Yury è in grado di eseguire il test o il deploy, potrebbe mettersi a dare una mano ad Andrea per "scodare" le attività nella parte destra della board.

Dato che Yury è in grado di svolgere il lavoro dei suoi colleghi, a questo punto ha varie possibilità: per favorire il completamento delle attività, potrebbe mettersi a lavorare insieme a Francesco per completare più rapidamente il task di quest'ultimo (figura 17).

In alternativa, Yury potrebbe prendere in carico un nuovo task: per esempio (figura 18) potrebbe prendere un task dalla colonna **done** dell'analisi e metterlo in lavorazione per lo **sviluppo**.

Qualora Yury fosse in grado di svolgere più attività (oltre a sviluppare codice), potrebbe aiutare i colleghi per esempio nella fase di test o di deploy finale. Nella figura 19 collabora con Andrea nelle attività relative di test (parte destra della board).

Gestione delle emergenze

Normalmente la sequenza con cui le varie attività sono messe in lavorazione segue l'ordinamento dei cartellini nella colonna iniziale del **ToDo**: la priorità potrebbe essere stabilita in base all'importanza o all'urgenza della richiesta (come in un sistema di help desk) o semplicemente al tempo di arrivo.

A volte capita che il team debba interrompere la normale pianificazione per svolgere alcune attività ad alta priorità — spesso delle vere e proprie emergenze — come quelle relative a un bug bloccante in produzione o a una variazione da inserire urgentemente nel prodotto.

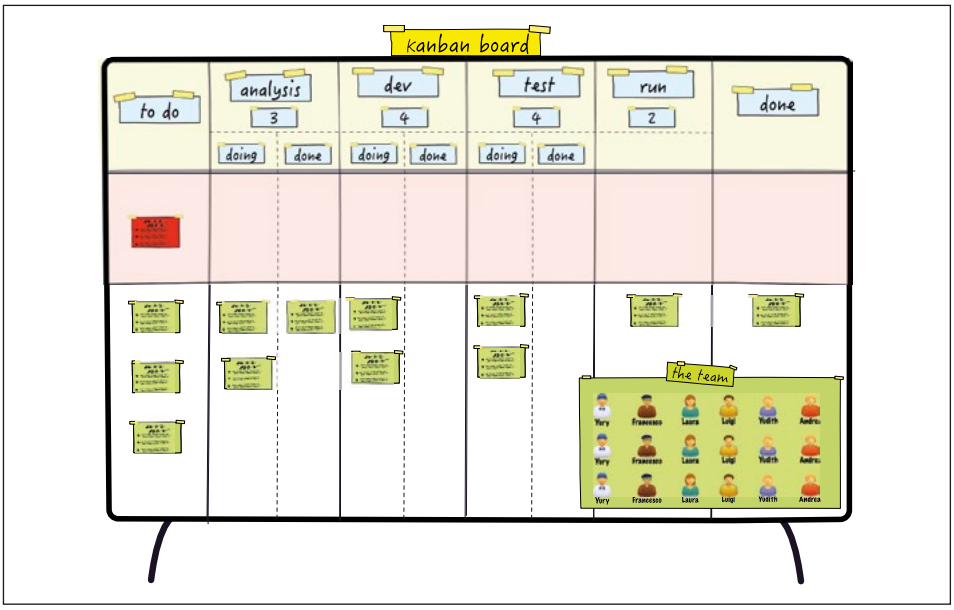


Figura 20. Una corsia di emergenza permette di gestire quelle attività che dovranno essere evase in tempi molto rapidi.

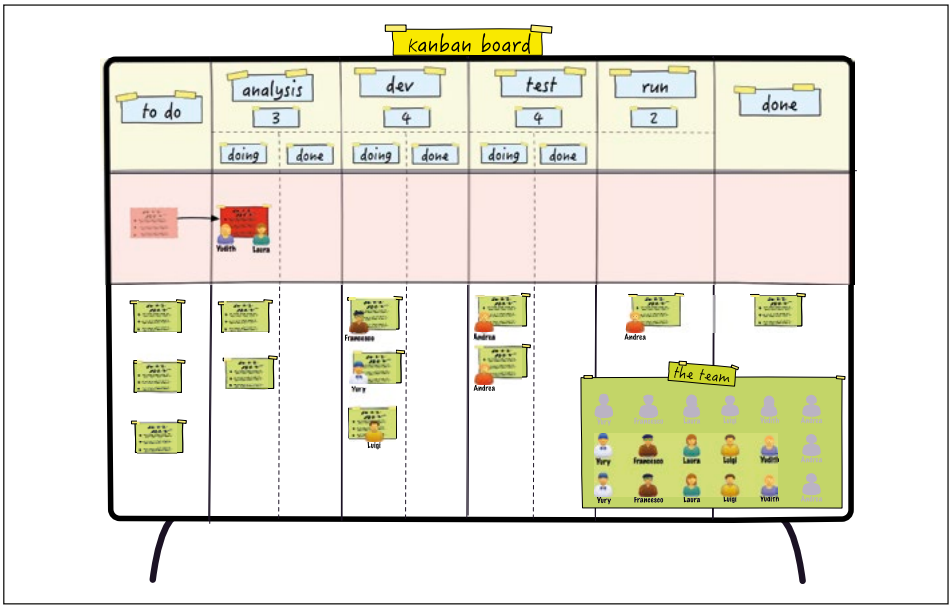


Figura 21. L'emergenza è presa in lavorazione. Due persone interrompono il loro lavoro "normale" e si dedicano a tempo pieno ad analizzare il problema. I rispettivi cartellini rimasti liberi sono lasciati nella loro posizione anche se di fatto occupano WIP senza che nessuno vi stia lavorando.

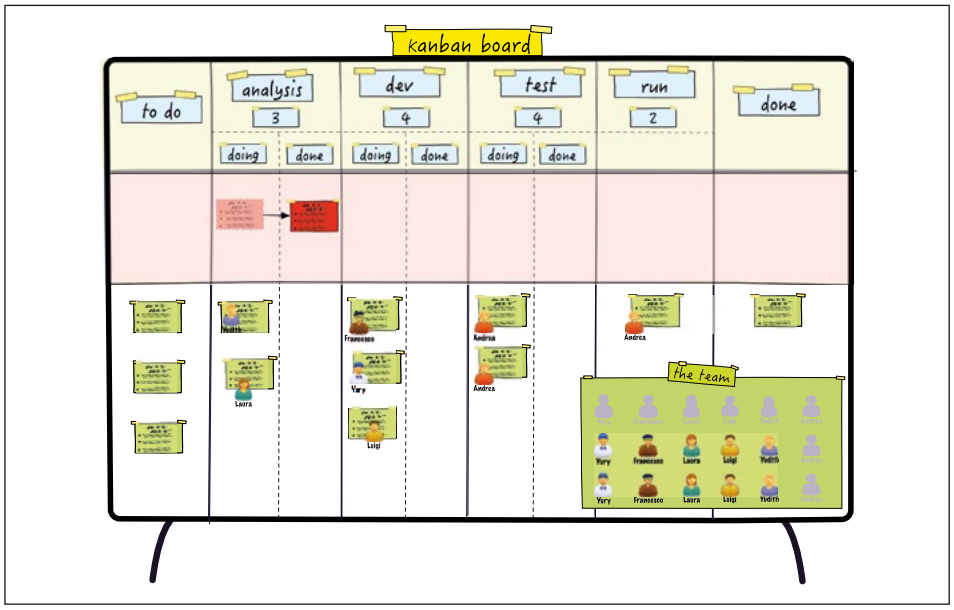


Figura 22. L'analisi è terminata; le due persone spostano il cartellino nella sottocolonna "done" e si rimettono a svolgere il proprio lavoro "normale" che avevano interrotto.

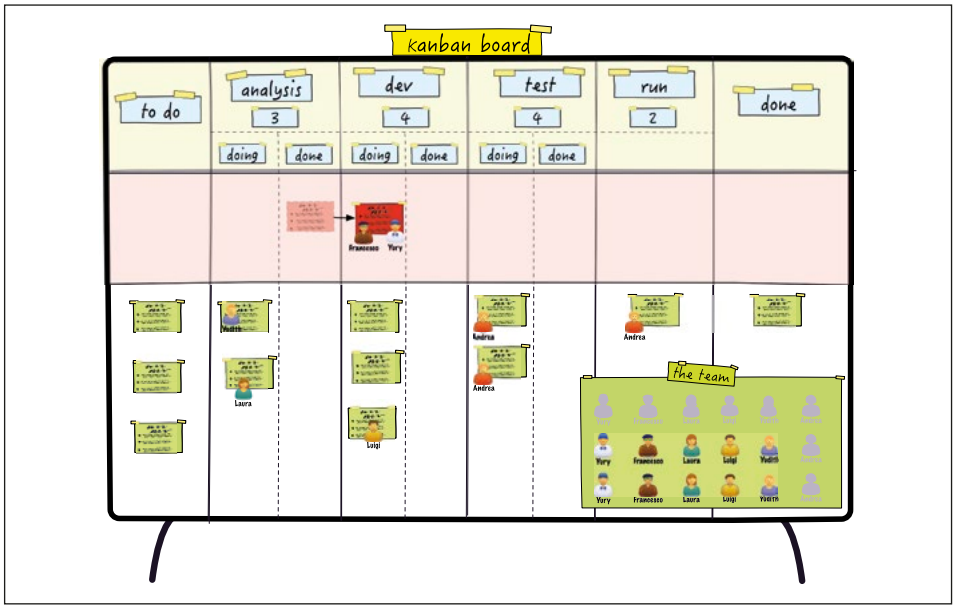


Figura 23. Adesso altri due membri del team interrompono il loro lavoro di sviluppo per implementare la funzionalità urgente. I passi successivi (verifica e rilascio) proseguono in modo analogo, fino a che il bug non è completamente risolto.

Per gestire queste situazioni, è utile apportare alcune piccole modifiche alla board; un modo è quello di inserire una apposita corsia orizzontale detta "corsia delle emergenze" o **Expedite Lane** (con riferimento alla corsia preferenziale riservata ai mezzi di soccorso) spesso identificata con un colore particolare (figura 20).

Analogamente anche i cartellini potrebbero essere colorati in base al livello di importanza, in analogia al codice colore che viene assegnato all'accettazione al pronto soccorso: rosso, giallo, verde, bianco o qualsiasi altra scala di colore ben comprensibile.

Nelle figure 21, 22 e 23 si può seguire la sequenza di azioni di una tipica procedura di emergenza.

Gestione dei difetti di lavorazione

Una variante del caso precedente si ha quando il team, durante la verifica di una attività in implementazione, trova un **difetto** prima che questo sia messo in produzione: si immagini il caso di un test che fallisce durante la verifica di una qualche parte del software (figura 24).

Un modo per gestire questa situazione potrebbe essere quello di **riposizionare** il cartellino corrispondente al test fallito, nella colonna della lavorazione necessaria per risolvere il problema (per esempio nuovamente in analisi o in sviluppo). Questa soluzione

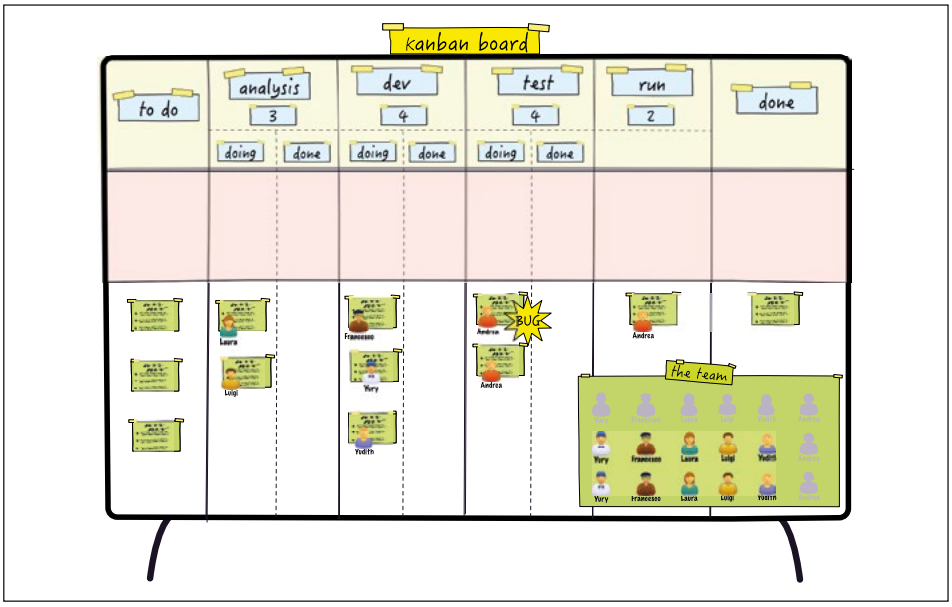


Figura 24. Caso simile al precedente ma con una sua specifica diversità è quello del difetto di lavorazione: il bug è trovato prima che la funzionalità sia messa in produzione. Una prima soluzione consiste semplicemente nel "retrocedere" il cartellino nella colonna di lavorazione necessaria per risolvere il problema.

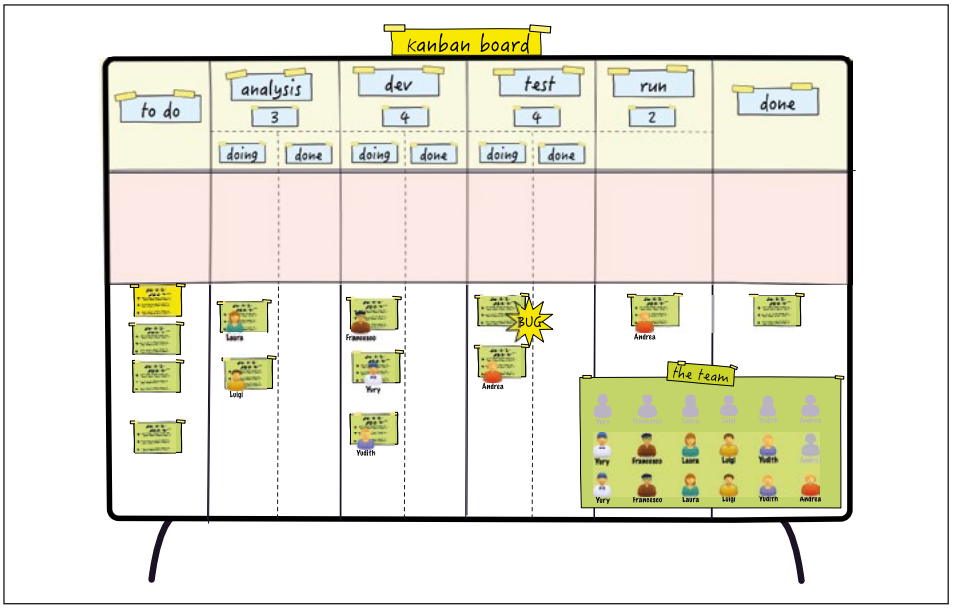


Figura 25. Un altro approccio alla gestione degli errori: in questo caso il cartellino associato all'attività che ha generato un errore viene "parcheggiato" (cartellino con segno rosso) nel punto dove si è verificato il problema, mentre uno nuovo "urgente" (in giallo), viene creato e messo in lavorazione. Il cartellino parcheggiato di fatto "consuma unWIP" per cui questa soluzione tende a portare a un ingolfamento del processo.

è quella più **semplice** da realizzare e anche la meno impattante a livello organizzativo: niente resta in sospeso nel punto dove ha generato l'errore e quindi si libera un WIP per altre attività che possono essere portate avanti.

Una variante, più scomoda da realizzare ma più **in linea** con i **principi agili**, potrebbe essere quella di lasciare il cartellino in errore nella sua posizione e di inserire in lavorazione un nuovo cartellino che andrà a "chiudere" il problema emerso (figura 25, 26 e 27).

In questo caso il cartellino con il bug rimane in attesa della soluzione del problema occupando un **WIP**, cosa che a tendere potrebbe portare al blocco del sistema: se ci sono molti cartellini "abbandonati" in attesa di una soluzione, niente altro potrà essere eseguito fino a che i problemi non sono stati risolti.

Questa strategia ha certamente un costo: secondo la filosofia Lean messa a punto da Ōno, questo costo deve essere da sprone non solo per risolvere il problema rapidamente ma, anche e soprattutto, per apportare le necessarie modifiche che evitino a quel problema di verificarsi nuovamente.

Interessante in tal senso la citazione di Teruyuki Minoura, ex presidente di Toyota Motor Manufacturing North America:

"Se si verifica un problema nello one-piece flow, si interrompe l'intera linea di produzione. In questo senso [la produzione Lean] è un sistema molto poco efficace. Ma quando si ferma la produzione, tutti sono costretti a risolvere immediatamente il problema: quindi devono riflettere, e riflettendo crescono e diventano membri migliori del team, e persone migliori."

Nella produzione di massa, invece, si fa largo uso di magazzino dei prodotti finiti e questo non fa emergere i problemi: è come se il sistema "nascondesse" i propri errori confinandoli nel magazzino invece di prenderli immediatamente in carico e per questo motivo un tale sistema di produzione è meno propenso al miglioramento continuo.

Suddivisione del flusso di lavoro: scatter & merge

Le board viste fin qui si adattano molto bene a quei casi in cui il processo è scomponibile in maniera **sequenziale**: ogni colonna corrisponde a una **fase** del processo di lavorazione; ogni attività, per essere completata, deve **passare** per **tutte** le **colonne** della board.

A volte però **non** è possibile **scomporre** il processo in **fette verticali**, perché alcune fasi della lavorazione possono richiedere attività che devono essere eseguite in parallelo. Si pensi per esempio alla produzione di un qualche componente che preveda lo sviluppo di una parte di **hardware**, di **firmware** di sistema e di **software** applicativo. Lo

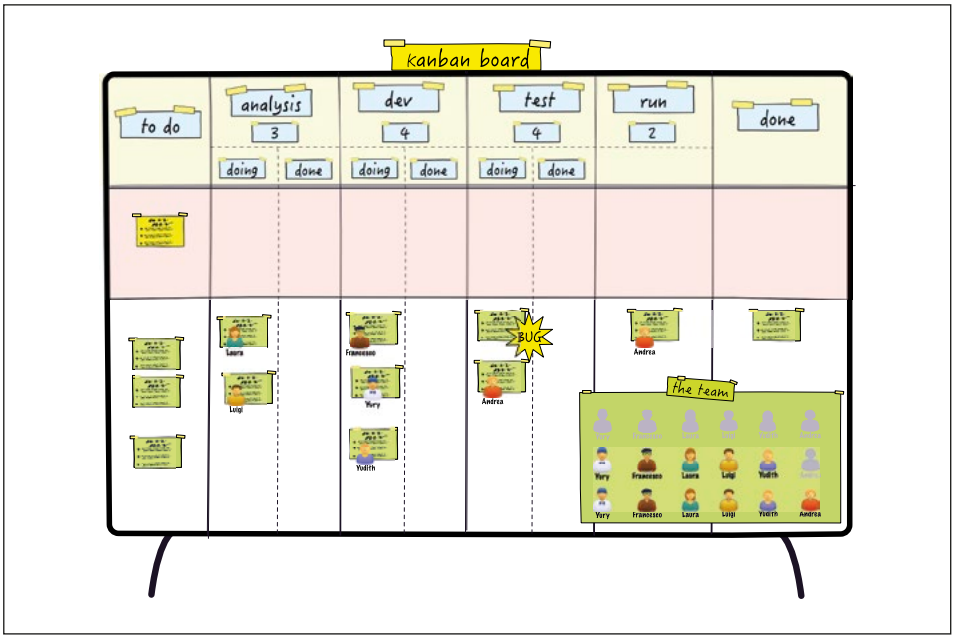


Figura 26. I passi successivi sono analoghi al caso precedente. Il cartellino giallo urgente viene messo in lavorazione con priorità nella corsia delle emergenze.

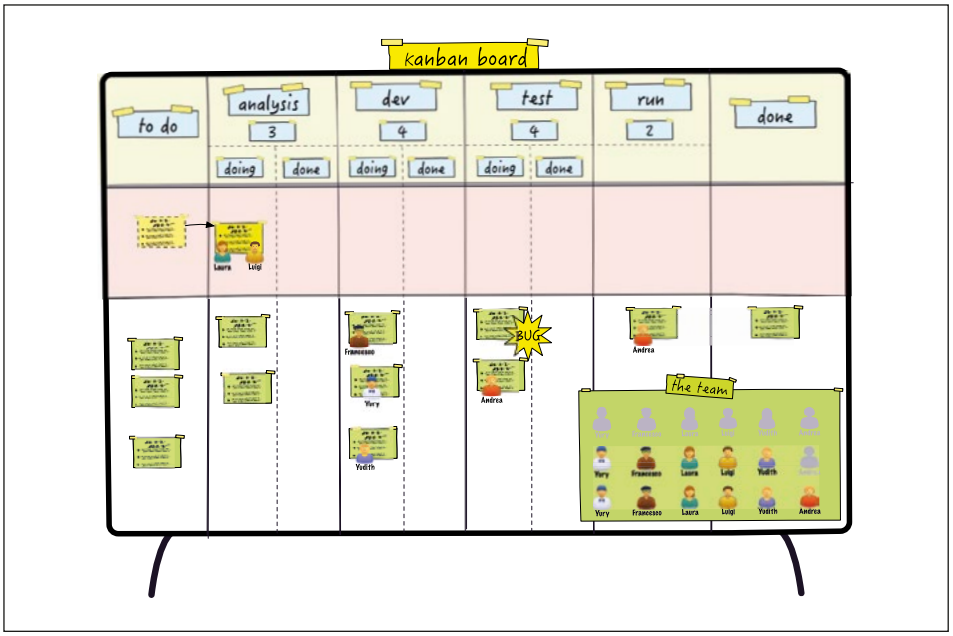


Figura 27. Anche in questo caso i membri del team potranno lavorare in coppia, oppure singolarmente.

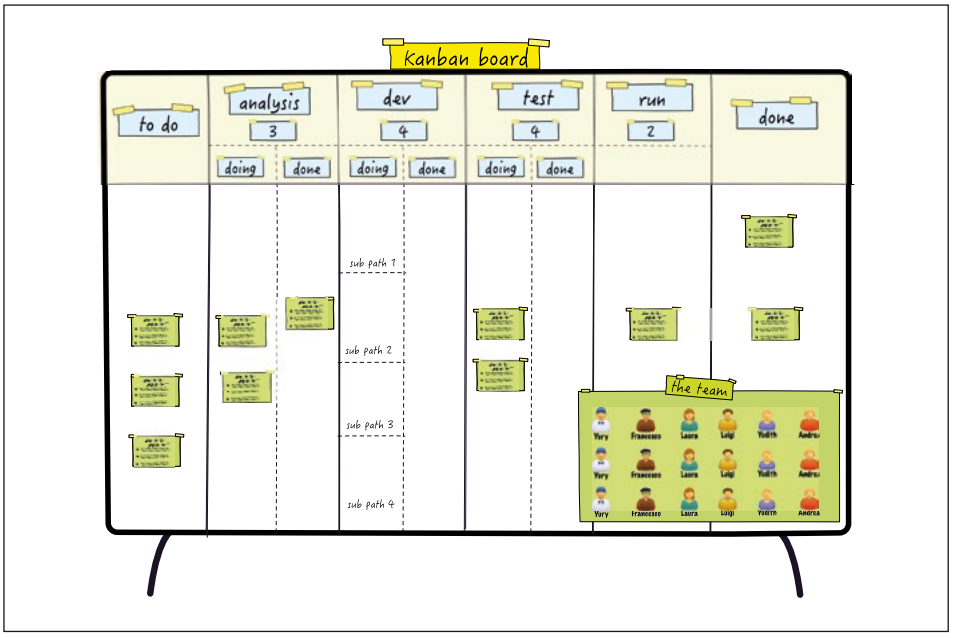


Figura 28. In questo caso lo sviluppo di un componente prevede differenti sottoattività.

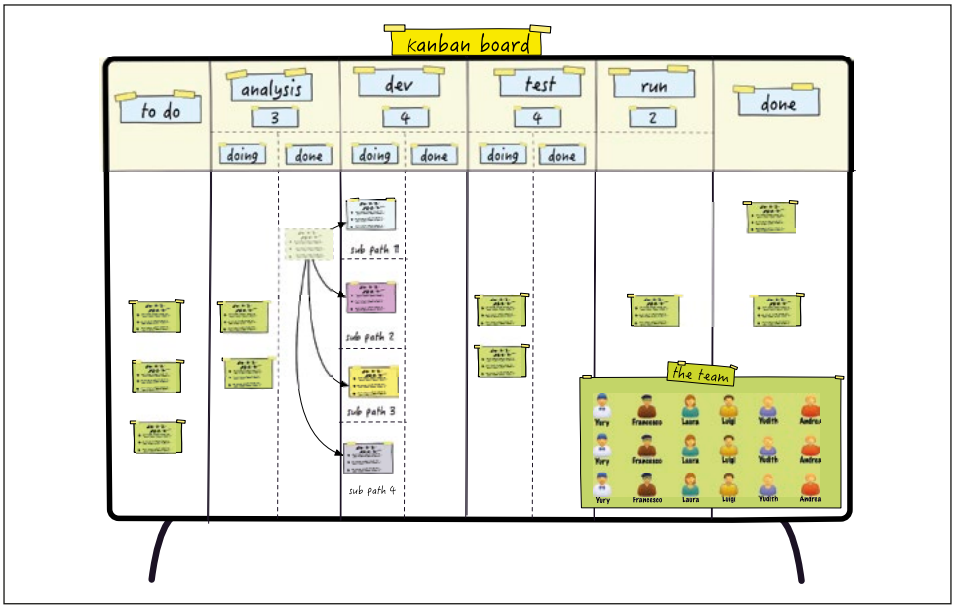


Figura 29. Ogni cartellino, per procedere nella lavorazione, viene "suddiviso" in altrettanti sotto-cartellini, uno per ogni sotto-attività.

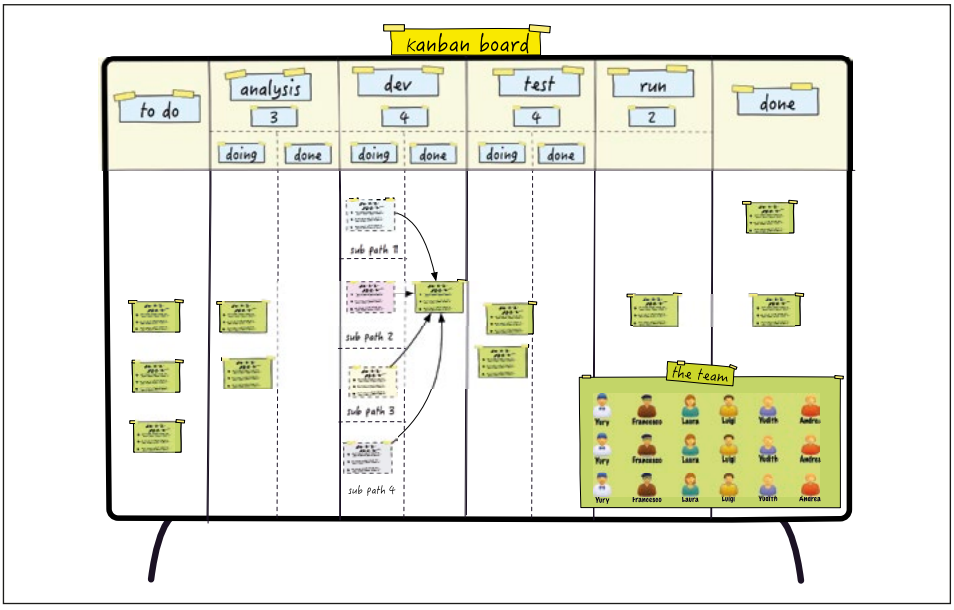


Figura 30. Solo quando tutte le sottoattività sono state eseguite, si può giungere alla riunificazione, ricomponendo i diversi "sotto-cartellini" in un unico cartellino che continua il suo viaggio nel flusso della lavagna kanban.

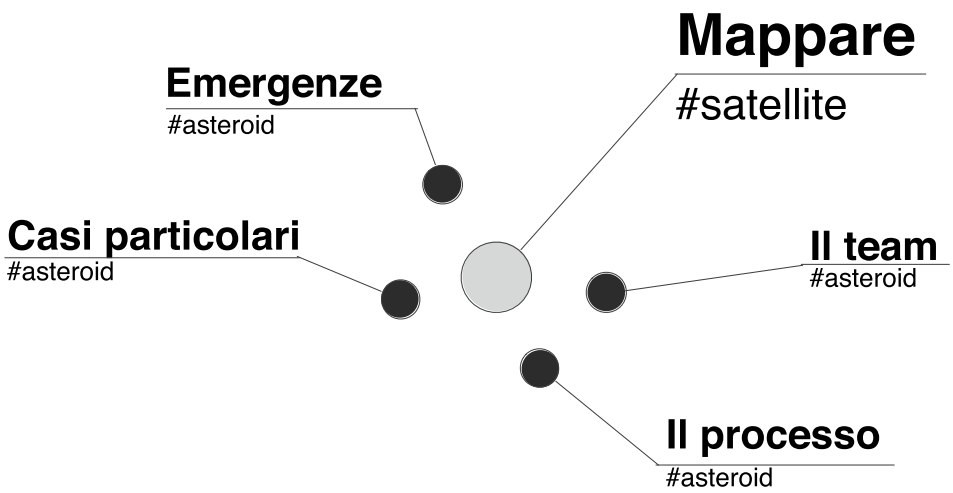
sviluppo potrebbe essere quindi diviso in queste tre linee di lavoro mentre il test potrebbe essere eseguito sul componente **dopo** che tutte le varie parti siano state **sviluppate** e **integrate**. Per poter gestire questo tipo di situazioni si può procedere a modificare la board introducendo delle corsie orizzontali solo nella colonna relativa allo sviluppo, come riportato nella figura 28.

In questo caso, un cartellino che arriva nella colonna di sviluppo, verrà **scomposto** in 4 **sotto-cartellini** che verranno poi **ricongiunti** in un **unico** cartellino man mano che lo sviluppo sarà terminato (figure 29 e 30).

Conclusioni

Abbiamo visto i passi necessari a creare una lavagna kanban. Nel capitolo successivo si vedrà come costruire una board che permetta di mappare un processo specifico.

Capitolo 3
Mappare il processo su una kanban board



Introduzione

Nei capitoli precedenti si è visto come è fatta una lavagna kanban e quali sono i principi Lean su cui tale strumento si basa. Siamo però partiti da un flusso ideale, creato a tavolino proprio per spiegare principi e meccanismi

Un passaggio che spesso preoccupa i neofiti è quello relativo alla creazione della **kanban board** per mappare un **processo reale** e permetterne una corretta ed efficace gestione. In realtà creare una board di questo tipo non è affatto complicato e anzi si possono seguire varie tecniche. Oltre a quello che raccontiamo in questo capitolo, esistono per fortuna molti libri e articoli sul tema, in grado di fornire validi suggerimenti.

In questo capitolo verrà mostrato, passo dopo passo, come costruire, una kanban board che rappresenti il processo in esame e che consenta di gestirne gli stati di avanzamento, di evidenziare i punti di miglioramento e altro ancora.

Passo numero 1: mappare il workflow

Mappare un **processo** di **produzione** tramite una **kanban** board è una attività che può essere svolta **iterativamente** in modo **incrementale**, aggiungendo ad ogni passaggio un dettaglio o una nuova parte.

Il suggerimento è di iniziare dall'individuazione del flusso del processo, o **workflow**; il flusso può essere definito anche in maniera grossolana, ma c'è un aspetto a cui va prestata la massima attenzione: cosa sta **dentro** e cosa sta **fuori** del processo, in modo di individuare approssimativamente il **contorno** del sistema che si vuole modellare.

In questa fase ci si può avvalere di strumenti grafici piuttosto semplici come un diagramma di flusso o un diagramma di stato UML (figura 31); utile in questa fase è l'utilizzo della tecnica del **Value Stream Mapping** per **identificare** i punti del flusso in cui si **genera valore**, punti che diverranno quelli su cui porre maggior attenzione nel processo di controllo tramite la kanban board.

In prima istanza non è importante definire i dettagli di ogni fase della lavorazione, ma ci si può limitare a disegnare il processo tramite una serie di **black box** in cui siano definite le **interfacce** di **input** e di **output** rispetto a sistemi esterni: lo scopo è separare ciò che partecipa alla realizzazione del prodotto finito, e che può quindi essere considerato come **parte integrante** del sistema, da ciò che invece può essere considerato un **elemento esterno**, con il quale il sistema deve **comunicare.**

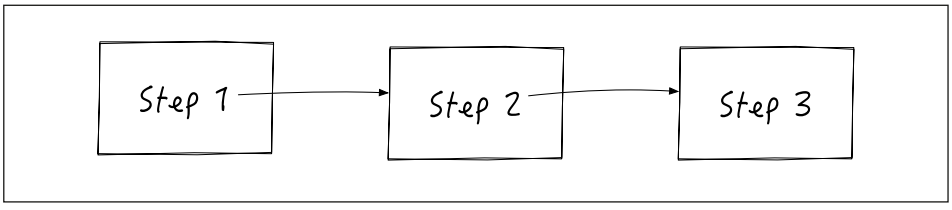


Figura 31 Si definisce un primo modello sintetico del processo.

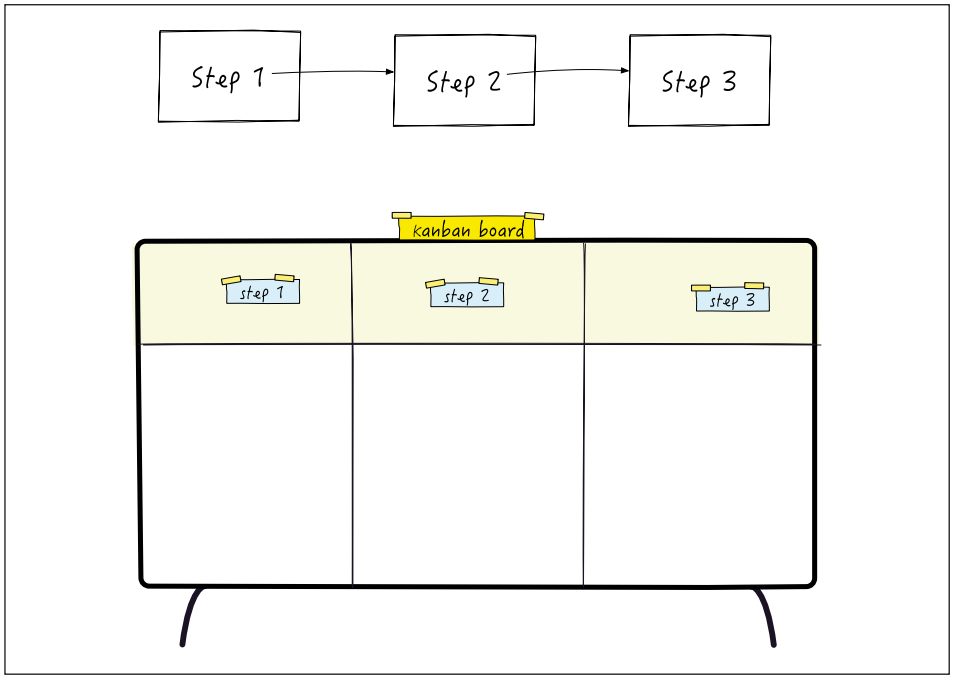


Figura 32. Dal workflow alla board.

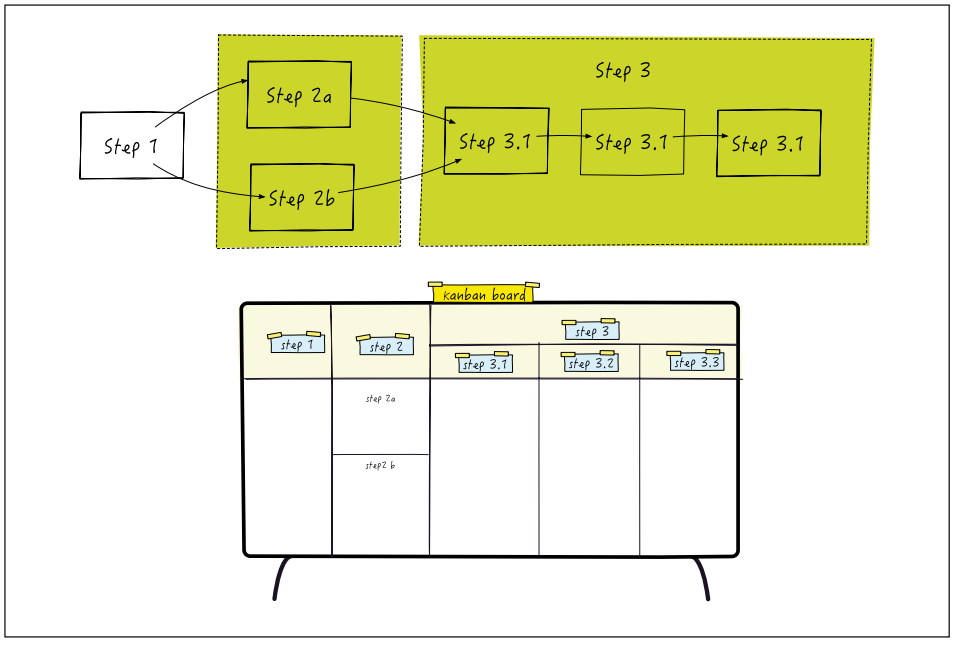


Figura 33. Cambia il livello di dettaglio del workflow, cambia la forma della board.

Disegnare la board

Dopo aver disegnato schematicamente il **workflow**, la realizzazione della kanban board, almeno in una sua prima rappresentazione approssimativa, è piuttosto semplice: ogni **passaggio** può essere rappresentato da una **colonna** della board.

Il livello di **dettaglio** della board dipende spesso dal grado di **precisione** che si vuole utilizzare per monitorare lo stato di avanzamento del processo di lavorazione: si potrebbe passare da un generico **ToDo**, **Doing**, **Done** — dove il **Doing** rappresenta tutto il processo di lavorazione nel suo complesso — a qualcosa di più dettagliato tramite l'inserimento di una **colonna** per ogni **fase** della lavorazione.

Non c'è una regola che possa aiutare a stabilire il livello di dettaglio nella definizione della board: quante colonne mettere, quali fasi rappresentare e quali invece accorpare con le altre. Una metrica potrebbe essere la frequenza o l'importanza degli eventi che si vogliono rappresentare con le colonne della board: mappare eventi rari o che contribuiscono marginalmente nella creazione di valore potrebbe introdurre complessità non necessaria. Utile in questo caso una analisi tramite il **Value Stream Mapping**.

Parlando in termini generali, la strategia è quella di aggiungere un dettaglio per volta e verificarne l'utilità.

Mappare la realtà del processo, non una sua rappresentazione ideale

Grazie alla sua capacità di **fornire visivamente** una **rappresentazione statica**, ma anche **dinamica**, dell'organizzazione, una kanban board è uno strumento estremamente

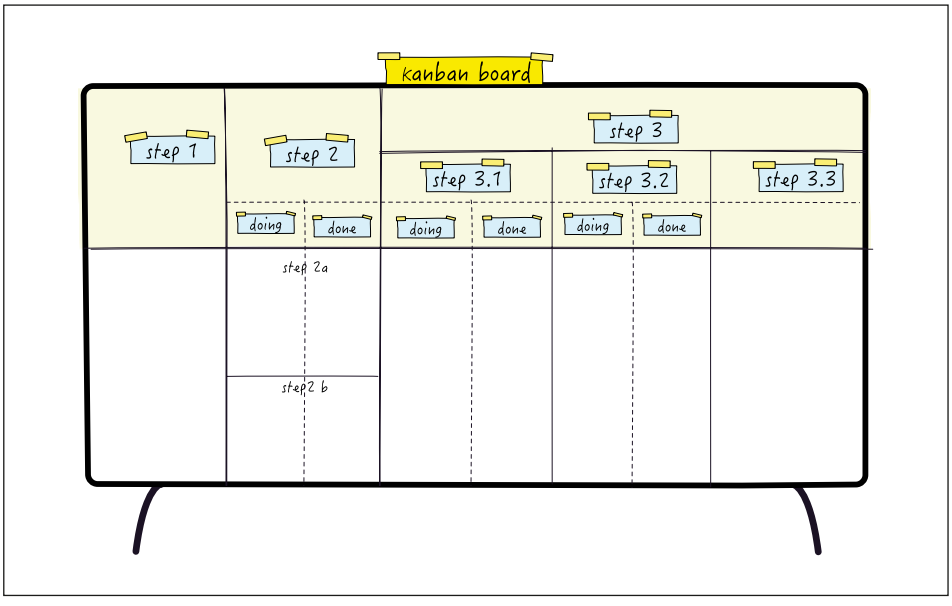


Figura 34. Si inseriscono le colonne buffer.

efficace nel **mettere** in **luce pregi** e **difetti** del processo: non sempre tutte le persone possono sentirsi a proprio agio con uno strumento che evidenzi in modo così diretto eventuali inefficienze nell'organizzazione.

Per questo motivo, affinché la diagnosi sia utile, conviene sempre rifarsi allo **schema reale** e non a quello "ideale" riportato magari in qualche documento ufficiale e utilizzato per una qualche presentazione in pubblico; capita alle volte che il processo messo **in atto** sia ben **differente** da quello **concordato** sulla **carta** in fase di progettazione. Ammettere questa discrepanza non è semplice specialmente se è una informazione non condivisa ufficialmente con i "piani alti".

Durante il processo di creazione della board è consigliabile rendere pubblico lo scopo del lavoro (trovare punti di inefficienza e migliorarli), per evitare che uno strumento di aiuto diventi causa di conflitti all'interno dell'organizzazione.

Passo numero 2: introduzione delle aree di buffer

Dopo aver disegnato le colonne, il passo successivo potrebbe essere quello di introdurre le aree di **parcheggio** o colonne **buffer** (vedi fig 34).

Passo numero 3: limitare la lavorazione in parallelo tramite il WIP limit

Dopo aver introdotto le colonne buffer, il passo successivo potrebbe essere dato dall'aggiunta, in testa alle colonne, dei **WIP limit**, in modo da vincolare il **numero** di **attività** che potranno essere svolte **in parallelo** all'**interno** di ogni **fase**.

Anche in questo caso non esistono regole o formule magiche per la scelta del limite del WIP: si consiglia quindi di seguire un approccio sperimentale per tentativi.

Il valore ottimale in genere si trova dopo alcune prove valutando, per ogni variazione, gli impatti sulle performance complessive sul sistema: per esempio si potrebbe individuare come metrica di riferimento il lead time medio necessario per completare la lavorazione delle attività prese in esame, prendendo il valore medio per avere una valutazione che tenga conto delle differenti tipologie di attività.

Uno strumento leggermente più raffinato e puntuale sulle varie fasi di lavorazione è quello che valuta il carico di lavoro di ogni step del processo: fasi troppo cariche (troppi di cartellini in lavorazione o in attesa di essere lavorati) o scariche del tutto (colonne vuote) sono indicatori di uno sbilanciamento sul processo, sbilanciamento che potrà essere "pareggiato" modificando i valori di limite del WIP. Ttipicamente un WIP troppo basso crea un ingorgo a monte e uno svuotamento a valle. Agendo sui vari **WIP limit** si può quindi rendere più armonico il bilanciamento del carico, cosa che in ultima analisi si ripercuote su un lead time minore.

Passo numero 4: definizione delle card

Oltre alla progettazione e al raffinamento della board, si può lavorare per aggiungere dettagli e informazioni ai **cartellini** corrispondenti ai task in lavorazione. Ogni

cartellino dovrebbe contenere tutte le **informazioni significative** relative all'attività da svolgere: dovrebbe essere presente un **titolo** che esprima in modo chiaro scopo e tipologia della attività, dovrebbe essere riportare l'**owner** o comunque il responsabile, la data di **nascita** (quando il cartellino è stato inserito nella lavorazione) e una **data** di **consegna** richiesta (oltre la quale il non completamento rappresenta un problema). Utile anche, qualcosa che ne specifichi la "classe" ossia l'importanza, urgenza o semplicemente il valore.

Di seguito sono descritti nel dettaglio i campi utilizzati più frequentemente.

Titolo

Un buon **titolo** dovrebbe essere sintetico e dovrebbe essere sufficientemente chiaro e "parlante", affinché possa comunicare l'obiettivo dell'attività e il beneficio che porta al sistema o a una persona specifica.

Qualora si usi una board fisica, il titolo dovrebbe essere al massimo una o due parole scritte in grande in modo che siano visibili anche da lontano.

Se necessario, ulteriori informazioni possono essere inserite in un sottotitolo o in una descrizione.

Date di creazione e di completamento

Sono due campi molto importanti per capire l'**età** del cartellino, ossia quanto tempo si sta impiegando per completare la lavorazione. Volendo raffinare le indagini si potrà contare anche il tempo che un cartellino passa nelle colonne di buffer in modo da arrivare calcolare il **lead time** e il **cycle time**. La **data di completamento** ovviamente va compilata a termine della lavorazione.

Deadline

La data **ultima** di completamento ammissibile.

Created by / owner

Chi ha creato il cartellino o meglio chi ne è il "**proprietario**".

Priorità

La **priorità** si esprime con un numero o un codice che indica quanto sia urgente una determinata attività.

Segnalazione di impedimento (impediment warning)

A volte, attività importanti rimangono bloccate a causa di **impedimenti** di vario tipo. Se il blocco dovesse protrarsi oltre un certo limite, si può attaccare alla card un qualche elemento **distintivo** (per esempio un pallino adesivo rosso) in modo da **evidenziare** che c'è un problema che rallenta o blocca del tutto la lavorazione di un elemento critico.

Tipologia di attività

Il **Task Type** è riferito sia alla classe di servizio che a una specifica tipologia di attività all'interno del processo (p.e.: analisi o sviluppo o altro).

Descrizione

Una breve **spiegazione** del lavoro da svolgere, degli obiettivi, dei benefici e dei beneficiari.

Note

Annotazioni varie, formato libero.

Requisiti per definire l'attività come completata

In Kanban, i **requisiti di completamento** non sono nulla di diverso da ciò che in Scrum si chiamano **Acceptance Criteria**, vale a dire un elenco di criteri che permettano di stabilire quando un'attività si possa considerare davvero **completata**.

Devono essere "collegati" alle policy definite per stabilire come e quando un cartellino possa passare da una colonna alla successiva.

Storia

La **history** è una serie di appunti sugli eventi che hanno caratterizzato l'attività in questione, una sorta di cronistoria della "vita" della card. Dalla data della prima immissione nel processo, al completamento, alla revisione e successiva re-immissione in lavorazione per correzioni di vario tipo; utile anche segnare l'elenco delle persone che hanno contribuito alla lavorazione della card.

Passo numero 5: definizione delle classi di servizio

Una board realizzata seguendo i passi descritti fino a questo punto, potrebbe essere uno strumento già molto utile per esempio per controllare lo stato di avanzamento delle attività di un team di sviluppo. Un ulteriore e importantissimo passaggio è quello di introdurre la possibilità di gestire le **diverse tipologie** di attività.

Nella realtà, infatti, alcune sono molto **urgenti** tanto che ogni minuto speso per la soluzione costa all'azienda molti soldi: si pensi a un bug bloccante in produzione.

Altre invece possono attendere per essere messe in lavorazione quando non ci sono urgenze da completare.

Infine ci sono attività a bassa priorità che possono essere lavorate solo quando non ce ne sono altre in coda.

La classificazione delle attività potrebbe essere fatta anche in funzione di altri parametri, per esempio considerando la persona che può svolgere quel lavoro.

In Kanban si parla di **classi di servizio**, per indicare la differente modalità di presa in carico in funzione dell'urgenza o dell'importanza.

Identificare le classi di servizio

Per inserire la gestione delle classi di servizio all'interno di una kanban board, la prima cosa da fare è identificare le varie tipologie di attività: per esempio in un progetto software si potrebbero considerare le seguenti:

- implementazione di nuove funzionalità;
- bug fixing a bassa priorità;
- bug fixing ad alta priorità;
- documentazione;
- attività di formazione a supporto degli utenti;
- setup ambienti e installazione software di supporto;
- studio per realizzazione di prototipi.

Per ognuna di queste categorie si prova a definire le priorità, per esempio valutando il **Cost of Delay**, eventuali **SLA (Service Level Agreement)** contrattualizzati o altro ancora.

Fatto questo si possono creare delle **macrocategorie** (dette appunto **classi di servizio**) dentro le quali far ricadere le varie attività. Ogni classe dovrebbe avere una **priorità** associata oltre alla **policy** di gestione. Per esempio, si potrebbero definire queste tre classi:

- **Classe 1** –priorità **bassa**. Costo del delay non influente. Trattamento specifico consiste nel mettere in fondo al backlog, lavorare se non c'è altro da fare. Quando la coda delle attività di questa classe supera le 5 unità, mettere in lavorazione quella più anziana.
- **Classe 2** – priorità **standard**. Costo del delay varia in base agli accordi contrattuali (rilasci ad ogni iterazione), è significativo ma non enorme. Trattamento specifico: lavorazione urgente ma non bloccante.
- **Classe 3** – priorità **alta**. Costo del delay: ogni giorno passato in lavorazione costa 1000 €. Trattamento specifico: lavorazione urgente e bloccante; ogni altra attività deve essere interrotta. Se necessario, più persone possono mettersi a lavorare su un task di questo tipo.

Classi di servizio per armonizzare le attività

Utilizzare le **classi di servizio** permette di applicare una gestione **coerente** delle varie attività, per esempio razionalizzando la modalità di intervento in base alle policy specificate per una determinata classe. Da notare che l'introduzione di politiche di gestione particolari, per limitare il **cost of delay**, porta contemporaneamente altri costi: bloccare la produzione o interrompere il lavoro di più persone per risolvere un bug in produzione ha ovviamente un costo, anche se nella maggior parte dei casi non è evidente e quindi si tende a **sottovalutarlo**. Grazie all'osservazione delle **metriche** che si possono inserire in una kanban board, è comunque possibile fare delle valutazioni in tal senso.

Dal punto di vista operativo, le **classi di servizio** possono essere introdotte tramite un **codice colore** particolare per i cartellini da apporre alla kanban board, oppure tramite la creazione di corsie (**swimlanes**, figura 35) specifiche per ogni classe.

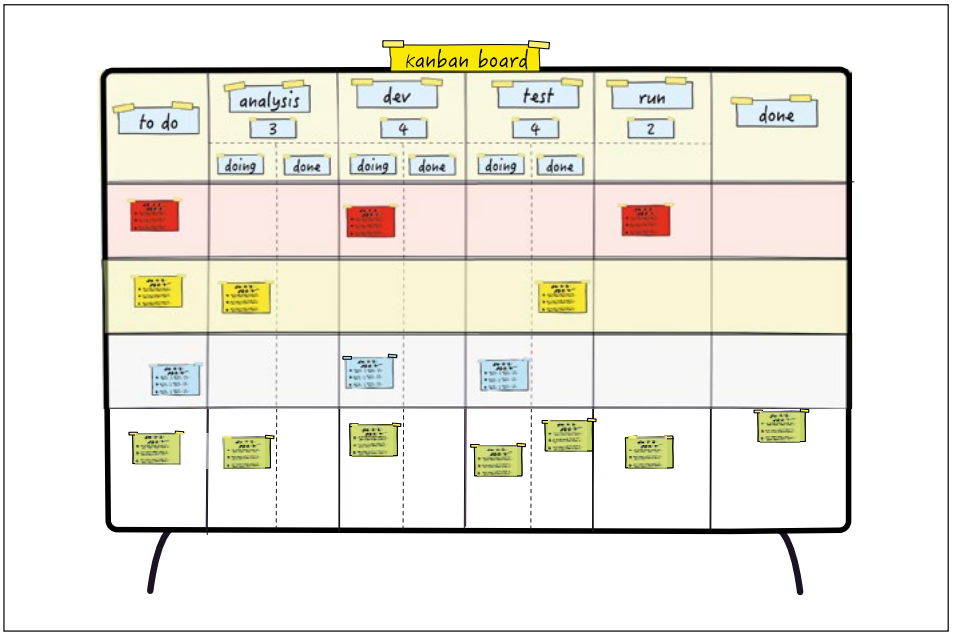


Figura 35. Sulla kanban board, è possibile segnalare l'appartenenza a una determinata classe di servizio grazie a specifici codici colore (rosso, giallo, verde, etc.) applicati ai cartellini, o per mezzo di "corsie dedicate" a ciascuna classe di servizio.

Passo numero 6: migliorare la board tramite le metriche

L'inserimento delle **colonne buffer**, la definizione dei **WIP limit**, così come la definizione delle varie **colonne** della board, sono decisioni che dovrebbero essere sempre soggette a valutazioni e riconsiderazioni sulla base dell'osservazione diretta del sistema e dell'efficacia della board stessa.

Spesso la configurazione migliore si verifica grazie alla naturale capacità che ha una board di fornire una **rappresentazione visuale** del sistema. Altre volte invece si rendono necessari approfondimenti basati su sperimentazioni e analisi retrospettive, successi e fallimenti. Non esiste una soluzione buona a prescindere, ma si deve trovare di volta in volta la configurazione migliore. Misurare quindi la risposta del sistema è un modo oggettivo per valutare l'efficacia delle varie prove.

Oltre alle metriche di cui si è già parlato, **lead time** e il **cycle time, molto utilizzate anche** il **Cumulative Flow Diagrams** (**CFD**), il **Defect Rate** e il **Blocked Items**. La scelta della metrica da usare è spesso funzione del tipo di indagine che si vuole fare ma soprattutto del grado di maturità dell'organizzazione.

Tipicamente si consiglia di inserire l'uso delle metriche in modo graduale, a mano a mano che l'organizzazione acquisisce capacità sia di lettura che — soprattutto — di intervento.

Avremo modo di parlare di **metriche** nel prossimo capitolo, quando affronteremo nello specifico questo argomento.

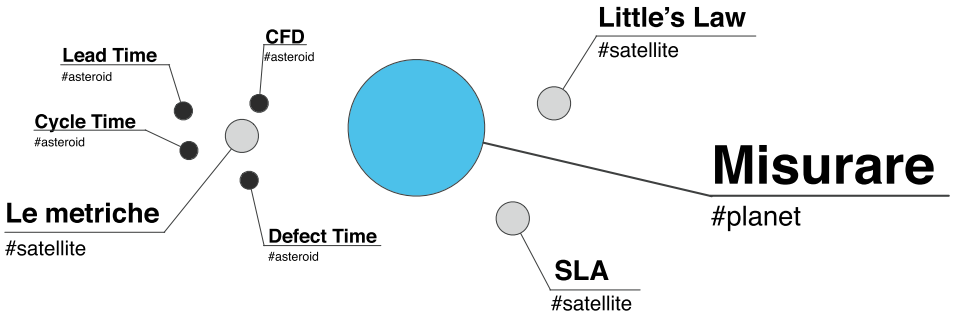
Conclusioni

In questo capitolo si è affrontato il tema della **mappatura** di un **processo** tramite la definizione di una board. Gli step visti sono i seguenti.

- mappare il workflow;
- introdurre le aree di buffer;
- limitare la lavorazione in parallelo tramite il WIP limit;
- definire le card;
- definire le classi di servizio;
- migliorare la board tramite le metriche.

È importante ribadire una volta di più che tutte le scelte che si fanno in questa fase della progettazione dovranno sempre essere **valutate** in funzione della capacità da parte della board di rispondere alle necessità di **monitoraggio** e di **miglioramento** del **processo**. Non esiste **una** soluzione ottimale, ma ogni decisione dovrà poi essere valutata e misurata sul campo sulla base dei risultati ottenuti. L'introduzione delle **metriche** può essere in tal caso un ottimo modo per poter valutare l'efficacia dello strumento adottato. Ne parleremo nel prossimo capitolo.

Capitolo 4
Misurare le performance



Introduzione

L'uso delle metriche è certamente molto importante sia per misurare le performance del team, sia e soprattutto per valutare l'efficacia di una modifica alla board.

In entrambi i casi, **misurare** permette di comprendere la bontà delle azioni intraprese: differenti strategie di lavoro, differenti atteggiamenti delle persone, differenti dettagli introdotti nella board.

Misurare con efficacia non è facile: occorre individuare una o più **metriche adatte** e misurare con rigore se si vogliono ricavare **informazioni utili** e non solo cifre. Se si vuole migliorare, non è tanto il valore puntuale di un KPI (*Key Performance Indicator*) ad essere significativo, ma l'andamento su medio-lungo periodo, ricavabile tramite **serie storiche** affidabili.

I KPI sono certamente utili per delineare un piano evolutivo, ma occorre **evitare** di **eleggere i numeri** a ruolo di **guida indiscussa** in una transizione agile. L'**ambiente** e il **contesto**, le **persone**, il **prodotto** sono tutti aspetti che spesso non è possibile misurare ma che rappresentano il **baricentro** di un'organizzazione e della sua evoluzione.

Misurazioni e obiettivi

Ogni organizzazione ha un suo livello di performance realistico (ottimale?): **spingere oltre** tale limite a volte non solo non porta alcun beneficio, ma anzi può essere fonte di una **degradazione** delle prestazioni o avere **conseguenze** sia sulle **persone** (stress, peggioramento delle dinamiche di gruppo, abbassamento della qualità della comunicazione) che sul **processo** (aumento dei difetti, peggioramento della qualità, ritardi nella consegna e altro ancora).

Per questo motivo è importante valutare l'**andamento** della serie storica, per enfatizzare i miglioramenti, ma anche per individuare prontamente un peggioramento.

Tenere stabili le metriche è fondamentale per avere valori comparabili; è altresì vero che se non si nota alcuna variazione del KPI scelto, è probabile che si stia guardando dalla parte sbagliata.

Prendendo il grafico della serie storica di un qualsiasi KPI, la discesa da un picco di massimo rendimento dovrebbe mettere in guardia: perché le cose stanno peggiorando? Cosa abbiamo fatto di diverso? E se volessimo invertire la tendenza cosa potremmo fare di diverso? Quali altre opzioni possiamo provare?

Metriche come strumento di allineamento

Le metriche sono quindi lo **strumento** per misurare gli effetti di una sperimentazione. Introdurre un **KPI** nel processo comporta quasi sempre un **costo diretto** — il dover effettuare le misurazioni — ma anche **indiretto**, legato per esempio alle conseguenze organizzative e politiche che si possono evincere dai risultati.

Chiedere alle persone di misurare qualcosa legato al loro lavoro può essere a volte una richiesta scomoda, un jolly da giocare con attenzione. La scelta di una metrica dovrebbe quindi tenere in considerazione anche questi aspetti: è consigliabile sceglierne

Figura 36. Provando a stressare un qualche fattore operativo nel team, le performance potrebbero, sul breve periodo, migliorare; insistendo oltre un certo limite, invece degradano bruscamente, con un trend molto più ripido di quello riportato in figura. Attenzione che le azioni che si devono compiere una volta superato il limite devono essere diverse per tipo e intensità: ossia è molto difficile recuperare un contesto che ha superato la soglia massima.

una compatibile con l'organizzazione, semplice da utilizzare e applicabile con costanza in maniera continuativa.

Le metriche dovrebbero essere quindi **strumenti** di **allineamento**, più che un criterio di valutazione del successo: allineamento in modo che **tutti** possano essere **informati** dello stato delle cose.

Quali metriche scegliere?

Dopo questa doverosa premessa, resta da rispondere alla domanda principale: cosa si misura? Fra le varie opzioni possibili, un buon punto di partenza, potrebbe essere quello di scegliere una metrica fra quelle che sono utilizzate nella maggior parte dei casi: **Cumulative Flow Diagram**, **Cycle Time/Lead Time**, **Defect Rate** e **Blocked Items**.

Cumulative Flow Diagram (CFD)

Il **Cumulative Flow Diagram** (CFD) è un diagramma con il quale si rappresenta lo stato di **avanzamento** della lavorazione: in un certo senso è paragonabile al **Burn Down Chart** rispetto al quale viene spesso preferito a causa della capacità di fornire una visione aggregata di varie metriche e con un dettaglio maggiore. Per questi motivi,

se da un lato il Burn Down Chart rappresenta lo strumento spesso utilizzato dai team alle prime armi, il **CFD** è invece adottato all'interno delle organizzazioni o dei team agili più maturi.

Il CFD è un diagramma in cui sull'asse orizzontale è riportato il **tempo** espresso tipicamente in **iterazioni** (se si adotta Scrum) o semplicemente **giorni** (opzione tipicamente utilizzata se si lavora in Kanban). In verticale invece è riportato il numero di cose da fare.

Come si può notare nella figura 37, il CFD mette a confronto nello stesso quadrante cartesiano la **curva del backlog**, ossia la totalità delle cose da fare, con i diagrammi corrispondenti alle varie **sottoattività** necessarie per completarle. L'ultima linea rappresenta l'ultima **attività** necessaria al completamento, in questo caso il **deploy finale**. Quando la prima curva **coincide** con l'ultima significa che tutte le cose da fare che erano in lista sono state completate.

L'**inclinazione** di queste linee è utile per comprendere come si comporta l'organizzazione o semplicemente come procede il progetto: per il backlog rappresenta la velocità con cui nuove cose sono messe nella lista delle cose da fare, mentre le altre curve indicano la performance puntuale delle varie sotto attività. Se l'inclinazione della curva del backlog è costantemente maggiore di quella del finito, significa che si stanno aggiungendo più cose da fare di quante il gruppo ne riesca realisticamente a realizzare.

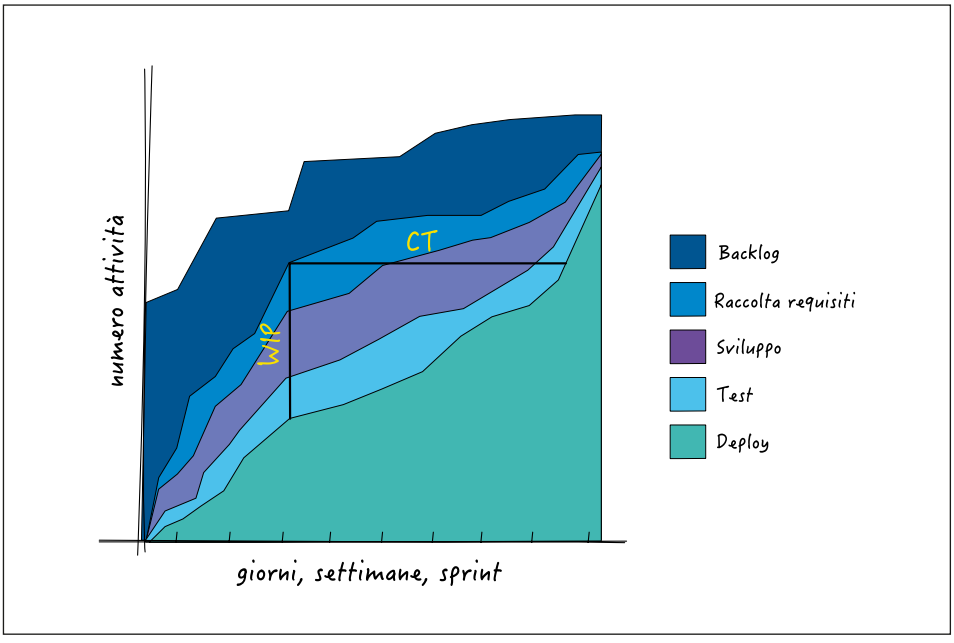


Figura 37. Il Cumulative Flow Diagram offre una visione d'insieme dell'andamento del totale delle cose da fare e delle attività necessarie per completare tali attività.

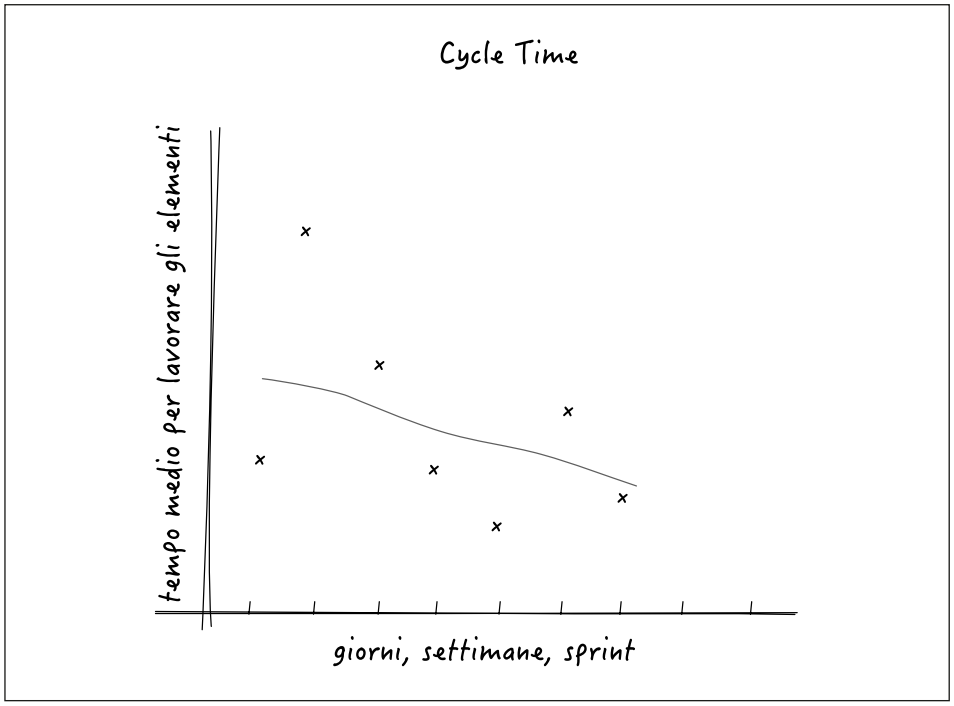


Figura 38. Riportando i punti dei vari Cycle Time su un piano cartesiano, si può visualizzare sia il trend che il grado di "sparpagliamento" ossia, usando un termine statistico, la varianza.

La **proiezione verticale** (nel disegno indicata con **vettore del WIP**) fra la curva relativa alla prima fase della lavorazione (in questo caso la raccolta delle specifiche, curva grigia) e quella corrispondente all'ultima fase di lavorazione (in questo caso il deploy finale, curva celeste,), rappresenta il numero di attività in lavorazione in ogni momento. Se il vettore del WIP aumenta, è probabile che sia in atto un rallentamento o, peggio ancora, un **collo di bottiglia**.

Nella figura per esempio. l'andamento del vettore del WIP è "sano": dopo un aumento iniziale (fenomeno naturale, visto che spesso è necessario il completamento di molte attività tecniche per poter rilasciare in produzione), diminuisce fino quasi ad azzerarsi.

La proiezione orizzontale rappresenta invece il tempo che impiega il team per completare la lavorazione di un elemento, ossia il cosiddetto **Cycle Time** medio (nel diagramma: linea **CT**). Se invece si traccia una linea che parte dal **tempo zero** si può calcolare il tempo complessivo che un elemento "trascorre" nel backlog prima che la sua lavorazione sia completata, ossia si ottiene il **Lead Time** medio (**LT**): da questo grafico si comprende meglio il concetto di questa grandezza, che è data da "tempo di attesa" più "tempo di lavorazione".

Un diagramma di questo tipo offre numerose informazioni circa le performance del gruppo di lavoro: il **CT** per esempio fornisce informazioni interessanti se si vuole capire come funziona il **team**; il **LT** invece, dato che rappresenta il tempo per evadere completamente una richiesta, è utile per valutare il livello di **servizio** offerto dall'organizzazione al cliente finale.

Per compilare un CFD è sufficiente che il gruppo di lavoro tenga traccia delle tempistiche di lavorazione (normalmente inizio e fine di un'attività) delle varie attività e sottoattività.

Andamento del Cycle Time e della varianza

Il CFD consente di ricavare il valore puntuale del Lead Time e il **Cycle Time** in un determinato momento. In ottica di miglioramento continuo è utile prendere in esame sia i **valori** misurati che il **trend** di tali metriche (figura 38).

Imparare a "confezionare" attività comparabili dal punto di vista dell'**effort** è un obiettivo comune a molte organizzazioni, dato che aiuta a stabilizzare il flusso e semplificare le attività di previsione: il trend della varianza è quindi un ulteriore importante indicatore per valutare l'efficacia degli sforzi del team nello stabilizzare il processo.

Da notare che il **Cycle Time** è un indicatore *a posteriori* e per questo dovrebbe essere utilizzato **in combinazione** con il **CFD** per avere una visione più completa.

Numero di elementi bloccati

Dato che il **tempo** di **esecuzione** di una attività (**CT** o **LT**) è proporzionale alle performance complessive, è evidente che ogni attività **bloccata** o **bloccante** ha ripercussioni negative sulle prestazioni dell'intero ciclo produttivo. Il **numero** di elementi **bloccati** è spesso ricavabile da altri strumenti come il **CFD** o il **CT**, anche se spesso i team preferiscono evidenziarne numero e tempo di risoluzione in modo esplicito, tramite metriche e diagrammi appositi.

Spesso il team utilizza **icone** dai **colori vivaci** e **rappresentativi** — rosso per i blocchi, arancione per gli impedimenti potenzialmente bloccanti — (figura 39) da attaccare sui cartellini della board: l'osservazione della board e della "macchia colore" permette di capire con un colpo d'occhio se ci sono problemi evidenti.

La figura 40 mostra un grafico dove sono riportati sia il numero di elementi bloccati che la media per la risoluzione del problema.

Defect Rate

Se il **numero** dei **difetti prodotti** da un team fornisce una indicazione puntuale e in tempo reale della qualità (bontà del processo, capacità delle persone, affidabilità dei test), monitorare l'andamento dei difetti prodotti è utile per avere una qualche indicazione sui benefici derivanti dalle eventuali modifiche apportate all'interno dell'organizzazione.

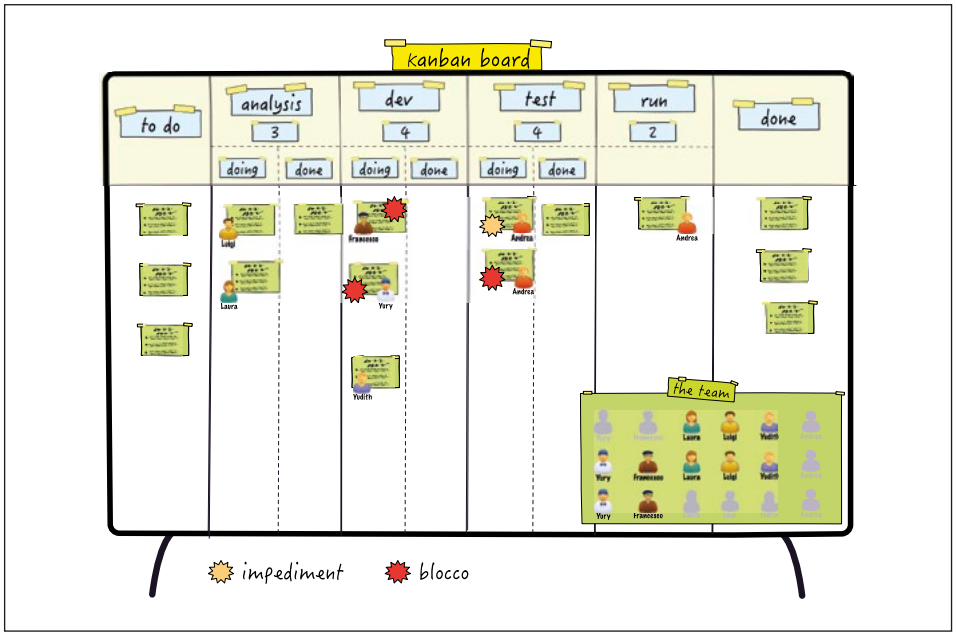


Figura 39. Apporre dei simboli colorati alle attività bloccate o con impedimenti è un modo semplice ed efficace per evidenziare se c'è un problema da qualche parte.

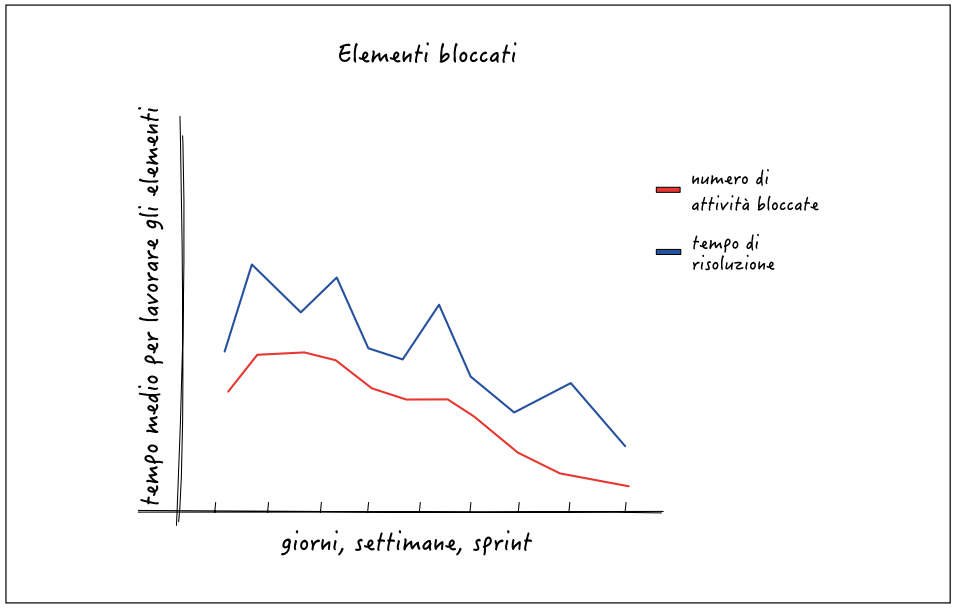


Figura 40. Grafico degli elementi bloccati e del tempo medio di risoluzione.

Come per le altre metriche viste fino ad ora, non è il valore puntuale a essere interessante quando il **trend**: se aumenta, perché aumenta? È perché è stato cambiato qualcosa nel processo di lavorazione, oppure perché sta aumentando la complessità del progetto? Di che tipo sono gli errori che si presentano? Sono difetti puntuali o di interdipendenza fra le varie parti? Sta peggiorando la struttura complessiva? L'architettura non è adatta al tipo di progetto che si sta realizzando? Manca un'adeguata politica di governance delle parti rilasciate? È stato deciso — e comunicato a tutti — cosa fare in caso di errore?

Anche quando le cose migliorano è utile interrogarsi sul perché: ci sono segni evidenti che il miglioramento sia dovuto a qualche azione o a elementi specifici, oppure è un caso? Dobbiamo aspettarci che possa peggiorare? Rispondere a queste domande può essere un ottimo punto di partenza per intraprendere le azioni opportune.

Sempre in tema di valutazione del trend, un paio di interessanti varianti sono il **numero dei difetti aperti** e il **tempo di risoluzione medio**: se un incidente può accadere per mille motivi, la capacità di risolverlo rapidamente è un buon indicatore della qualità organizzativa del lavoro del team.

Dalla misurazione alla previsione: definire i SLA

L'introduzione di metriche come quelle viste fin qui, fra l'altro, permette di fare supposizioni sul tipo di performance che potremo aspettarci dal sistema. Quando i numeri iniziano a stabilizzarsi, diventa meno rischioso impegnarsi a rispettare un certo livello di servizio (SLA, **Service Level Agreement**) nei confronti di clienti o stakeholder esterni.

Nel capitolo precedente abbiamo parlato di **classi** di **servizio** delle varie attività, e delle relative politiche di gestione; definire tali policy e mantenerle stabili nel tempo è un requisito per il miglioramento delle prestazioni del sistema. Le metriche possono confermarlo a posteriori.

Le classi di servizio e i corrispondenti SLA

Nei paragrafi precedenti, si era fatto l'esempio delle seguenti classi di servizio:

- **Classe 1: bassa** priorità. Costo del delay non influente. Il trattamento specifico consiste nel mettere in fondo al backlog, lavorare se non c'è altro da fare. Quando la coda delle attività di questa classe supera le 5 unità, mettere in lavorazione la più anziana.
- **Classe 2:** media priorità. Costo del delay varia in base agli accordi contrattuali (rilasci ad ogni iterazione): è significativo ma non enorme. Trattamento specifico: lavorazione urgente ma non bloccante.
- **Classe 3: alta** priorità. Costo del delay: ogni giorno passato in lavorazione costa 1000 €. Trattamento specifico: lavorazione urgente e bloccante; ogni altra attività deve essere interrotta. Se necessario, più persone possono mettersi a lavorare su un task di questo tipo.

Lavorando sulle metriche e analizzando quindi lo stato di salute del sistema, si potrebbero definire opportuni SLA in funzione della probabilità ossia del rischio; per esempio si potrebbero proporre i seguenti livelli di servizio:

- **Classe 1: bassa** – SLA: tempo medio di risoluzione 20 gg; il 90% dei ticket chiusi in 25 gg , tutti entro 30 gg.

- **Classe 2: standard** – SLA: tempo medio di risoluzione 10 gg; il 90% dei ticket chiusi in 15 gg, tutti entro 20 gg.
- **Classe 3: alta** – SLA: tempo medio di risoluzione 5 gg; il 90% dei ticket chiusi in 10 gg, tutti entro 12 gg.

Conclusione

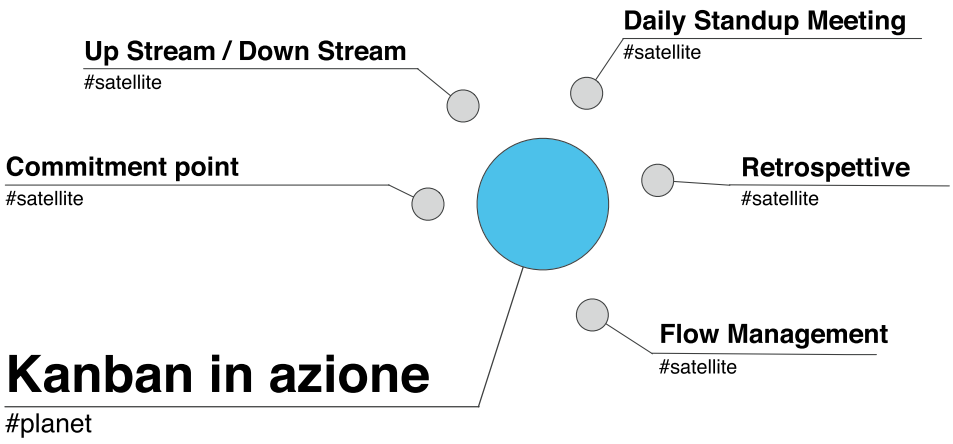
Termina qui la trattazione su misurazioni e metriche da inserire all'interno di un processo Kanban.

Per maggiori approfondimenti sull'uso delle metriche, si rimanda all'appendice di Mattia Battiston — anche se "appendice" potrebbe risultare un termine riduttivo visto che è una trattazione di ottimo livello — che sarà pubblicata nella versione finale completa di questo libro, e i cui argomenti possono essere già trovati in una prima versione nella serie di articoli *Metriche Kanban per il miglioramento continuo* pubblicati nel cosro del 2016 su MokaByte [MET].

Riferimenti

[MET] Mattia Battiston, *Metriche Kanban per il miglioramento continuo*, MokaByte, 2016
http://www.mokabyte.it/2016/02/kanbanmetrics-1/
http://www.mokabyte.it/2016/03/kanbanmetrics-2/
http://www.mokabyte.it/2016/04/kanbanmetrics-3/
http://www.mokabyte.it/2016/05/kanbanmetrics-4/
http://www.mokabyte.it/2016/06/kanbanmetrics-5/
http://www.mokabyte.it/2016/07/kanbanmetrics-6/

Capitolo 5
Kanban in azione: consigli pratici



Lessons learned...

A completamento di questa parte dedicata a Kanban vorrei condividere alcune tecniche e svariati suggerimenti frutto dell'esperienza diretta sul campo: si tratta di suggerimenti utili per risolvere alcuni aspetti concreti come la strategia di condivisione degli allineamenti o come gestione delle retrospettive.

Molte delle idee presenti in questo capitolo sono il frutto di sperimentazioni e tentativi "sul campo" e, pertanto, non è detto che siano adatte a tutti i casi. Se, di fatto, è impossibile trovare "la soluzione" valida per tutte le situazioni operando in un mondo complesso, può comunque essere il caso di condividere quanto raccolto, con la consapevolezza che in alcuni casi potrebbe essere applicabile, in altri no.

Quando si mettono "le cose" nella kanban board?

Una delle questioni più importanti nel processo di gestione di una **kanban board** è il processo di **alimentazione** della stessa. In tal senso può essere utile considerare la lavagna come un sistema di "condotte" nei cui "tubi" le attività scorrono seguendo un flusso che in genere è diviso in due fasi:

- una prima fase di raccolta, alimentazione e relativa selezione di nuove attività;
- una seconda fase di effettiva lavorazione.

Commitment point

Il punto che separa queste due fasi viene in genere definito in inglese **commitment point** dato che rappresenta il passaggio dalla fase in cui le attività sono delle opzioni a quella in cui il team si **impegna** realmente a svolgerle (**committed work**).

Le attività che sono in fase di valutazione, alla sinistra del **commitment point**, sono delle opzioni che potrebbero non passare la selezione. David J. Anderson, in una sua recente pubblicazione su Kanban [EKC], descrive questa parte utilizzando la metafora di un concorso di bellezza, dove le storie fanno a gara per essere scelte: solo quando un item supera il **commitment point** prende importanza; prima di allora, è semplicemente una opzione.

Una volta messa in lavorazione, un'attività non dovrebbe mai essere abbandonata se non in casi eccezionali: per questo motivo si sceglie la parola *commitment*, che indica il "prendersi seriamente un impegno". Dal punto di vista del cliente, la presa in carico di un'attività che passa il commitment point rappresenta una promessa, più o meno esplicita, che tale attività verrà portata a completamento, prima o poi.

Up Stream e Down Stream

Il processo di selezione (prima) e di implementazione (poi) può essere considerato come un unico flusso di lavorazione ("**work flowing like a stream**") che scorre da sinistra verso destra; per questo la zona a sinistra del **commitment point** viene chiamata **Up Stream** (il corso iniziale del flusso), mentre quella alla destra è detta **Down Stream** (la porzione finale del flusso).

Il **lead time**, di cui abbiamo già parlato in passato e di cui parleremo ancora, non dovrebbe tener conto del tempo in cui le attività sono in Up Stream: in questa fase, il tempo di attesa **non** dipende dal processo.

Per valutare e selezionare le attività presenti in Up Stream, si può ricorrere alle tecniche tipiche utilizzate in Product Management o Risk Management. La figura 41 spiega in modo piuttosto chiaro questo schema.

Spesso il **commitment point** è la prima colonna della board sulla quale si impone un WIP limit piuttosto basso.

A volte si impone un **WIP limit** anche in Up Stream per evitare ingolfamenti: inutile prendere in esame nuove idee se il gruppo di lavoro non è in grado di valutare nemmeno quelle già presenti in Up Stream.

Queue replenishment

Al **commitment point** si esegue il cosiddetto **queue replenishment**, vale a dire il "rifornimento della coda": quando ci sono caselle vuote nella prima colonna, bisogna scegliere quali storie meritano di essere tirate (**pull**) dentro il sistema. Il queue replenishment può avvenire **just-in-time**, cioè ogni volta che c'è uno slot vuoto, o essere cadenzato, qualora le persone necessarie per la valutazione/selezione non siano disponibili in modo continuativo e costante.

Il vantaggio della suddivisione in Up Stream e Down Stream

La separazione del processo in queste due fasi offre un interessante strumento di **sense-making**: il modo con cui le attività sono gestite cambia radicalmente se ci si trova a **destra** o a **sinistra** del **commitment point**.

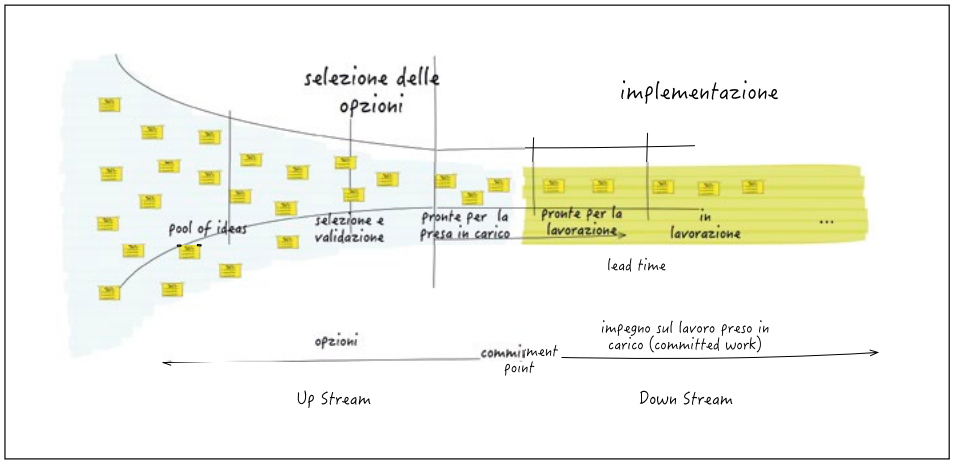


Figura 41. Il commitment point separa la parte di selezione delle attività potenziali (Up Stream) da quella della lavorazione delle attività selezionate (Down Stream).

Per esempio il tipo e il significato delle metriche cambia molto a seconda che si consideri l'Up Stream (cono delle opzioni) o il Down Stream (impegno a prendere in carico e completare il lavoro). Durante la fase di lavorazione, il **lead time** assume un significato più concreto e realistico, essendo eliminata ogni indecisione sul "lavorare o non lavorare" un determinato item; questo aiuta a formulare in modo più realistico ipotesi sulle tempistiche di termine dei lavori e di rilascio. In Up Stream, invece, si usano metriche differenti come la lunghezza delle code, il tempo di attesa in determinate sottofasi o addirittura il coefficiente di cancellazione degli item.

Quando si fanno gli standup?

Una delle colonne portanti di Kanban è certamente il modello dei **feedback loops**, dei quali lo stesso Anderson [EKC] ha sottolineato l'importanza:

"Di recente ho adottato un nuovo approccio all'insegnamento del metodo Kanban. Il nuovo corso di Lean Kanban 'Practicing the Kanban Method' si incentra sulle '7 Kanban Cadences', le riunioni tenute ciclicamente che guidano il cambiamento evolutivo e la consegna di servizi 'buoni per lo scopo' per cui sono creati."

Anderson sottolinea che, per implementare i vari **feedback loops**, non necessariamente si deve dar vita a 7 nuovi meeting, ma anzi si potranno interpretare in modo nuovo quelli già esistenti.

Feedback loops e daily standup

Fra i vari **meeting** possibili, certamente il **Daily Standup** preso in prestito da Scrum è uno dei più importanti dal quale si possono trarre maggiori benefici proprio nell'ottica di creare dei feedback.

Il formato tipico di uno **Daily Standup Meeting** di derivazione Scrum prevede che ogni persona elenchi tre elementi fondamentali:

- cosa è stato fatto il giorno prima;
- quali impedimenti si sono incontrati nel fare le cose il giorno precedente;
- cosa ci si appresta a fare il giorno stesso.

Questo approccio ricalca la filosofia tipica di Scrum, in cui ci si orienta fortemente all'analisi delle cose da fare: è un approccio **task-oriented** all'interno dello sprint.

In Kanban invece tipicamente l'attenzione è rivolta verso la **massimizzazione** del **flusso** ed è per questo motivo che può essere utile modificare tale meeting in modo da privilegiare un'analisi **flow-oriented**.

Dal "task-oriented" al "flow-oriented"

Un modo per ottenere questo risultato è quello di abbandonare il classico schema delle tre domande, per passare a una **valutazione** dei **cartellini** presenti sulla **lavagna** da destra a sinistra secondo quello che a volte viene detto stile *walk the board* ("camminare attraverso la lavagna") in modo da implementare un approccio **pull**.

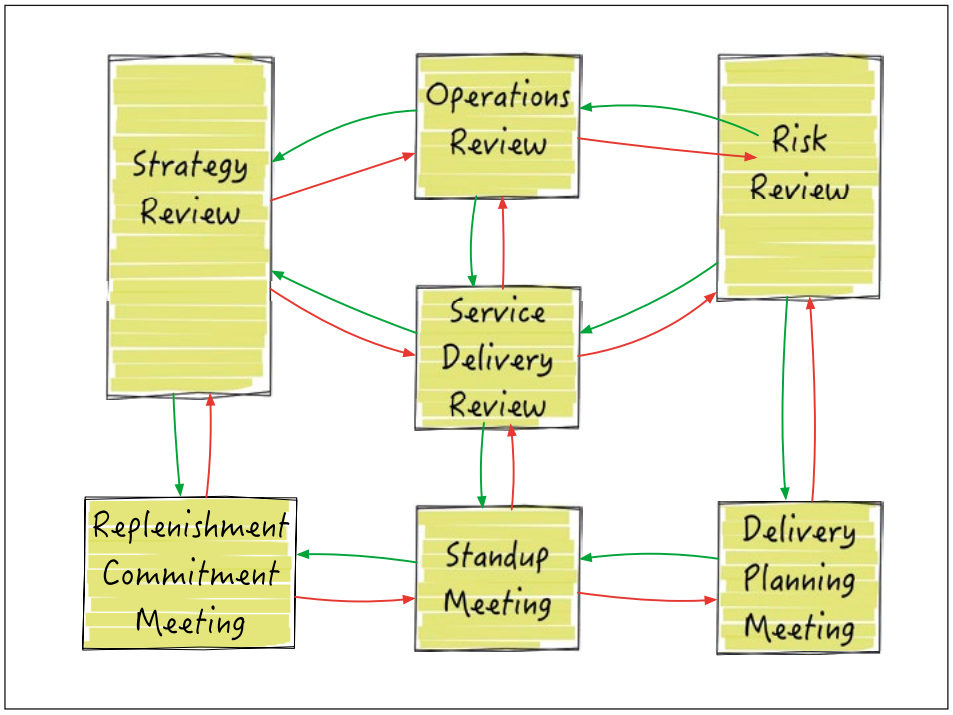


Figura 42. Il sistema dei feedback loops di cui parla Anderson può essere visto come un reticolo di interazioni interdipendenti il cui scopo ultimo è il miglioramento delle prestazioni del sistema, ossia ridurre il lead time medio.

Con questa nuova modalità, si segue lo schema riportato di seguito:

- Per prima cosa si analizzano gli eventuali elementi bloccati, cercando di individuare la causa che ne impedisce il completamento. Questi sono senza dubbio i problemi più importanti e conviene risolverli prima possibile: ha senso che vi si dedichi la maggior parte del tempo a disposizione.
- Secondariamente, conviene concentrare l'attenzione sulle attività presenti sulla expedite lane o più in generale sugli emergency items se presenti. È prioritario per il team velocizzare la lavorazione di tali attività, vale a dire risolvere il problema corrispondente, anche a costo di togliere tempo prezioso alla lavorazione di altre attività. In questo caso è utile marcare il cartellino corrispondente all'attività in analisi, per esempio utilizzando un simbolo visibile e comunicativo: un pallino rosso o un altro simbolo di allarme.
- A questo punto si può passare ad analizzare quelle attività che non si sono mosse dall'ultimo standup. Anche se probabilmente è ancora presto per considerarle bloccate, è probabile che siano a rischio. A volte, per il breve tempo intercorso dallo standup precedente, può essere plausibile che non si siano spostate nella board: in tal caso

la discussione non dovrebbe richiedere molto tempo. Qualora invece si dovesse sentire puzza di bruciato, potrà essere utile indagare un po' meglio per capire se si tratta di una situazione normale o se si prefigura un blocco.

- Infine si può passare ad analizzare gli altri elementi rimanenti: in questo caso, se i cartellini sono ancora molti, si può seguire la priorità delle classi di servizio delle attività, partendo da quelle a priorità maggiore (bugs, critical features, altro); aiuta in questi casi la classificazione delle attività per classi di servizio su cui dovrebbe essere definita una priorità.

Durante questo processo di valutazione, per decidere il piano di lavoro quotidiano, sono d'aiuto **dati** e **metriche**, come l'età di cartellino ("Da quanto tempo è in lavorazione questa attività?") o l'effort medio richiesto ("Quanto impiegano solitamente attività di questo tipo per essere completate?").

Vantaggi del daily standup meeting in stile "walk the board"

Seguire la priorità delle attività — si parte dalle attività bloccate per arrivare a quelle delle classi di servizio meno prioritarie — rende il processo di analisi più efficace, perché si agevola la consegna di attività terminate, velocizzando il rilascio di valore all'utente finale. Ma è anche efficiente: se il tempo del meeting termina prima di essere riusciti ad analizzare tutti i cartellini, si saranno trascurate solamente le attività a priorità più bassa, non bloccate, che in definitiva non danno preoccupazioni e che non richiedono quindi particolari discussioni... almeno per ora.

Inoltre, questo stile di conduzione del Daily Standup ha l'effetto di migliorare la collaborazione: alla domanda "Cosa dobbiamo fare oggi per spostare verso destra i cartellini?" in genere si ottiene un coinvolgimento e una partecipazione maggiore da parte di tutte le persone.

Per concludere, è importante notare che il Daily Standup non è altro che uno strumento, per cui può valere la pena di sperimentare o comunque di fare modifiche: si possono sfruttare quei 10 minuti per fare qualcosa di diverso. Per esempio nei gruppi di lavoro di Buffer [BUF] utilizzano lo standup non solo per misurare i **miglioramenti** del **processo** di **lavorazione**, ma anche per analizzare eventuali **miglioramenti personali** dei membri del team, sia per quello che concerne le conoscenze tecniche, che interpersonali.

Flow manager, flow management

Nel paragrafo precedente si è visto come il cambio di approccio nello standup permette di spostare il focus dal fare "la contabilità" dei task da svolgere, per tenere occupate le persone del team, al **miglioramento** del **flusso** di **lavorazione**; a tal proposito è interessante l'introduzione del **ruolo** del **Flow Manager**, una nuova figura che alcuni team stanno iniziando ad adottare con benefici tangibili.

Il **Flow Manager** concentra il suo operato su tutte quelle iniziative volte a velocizzare lo scorrimento dei cartellini sulla board e fra queste rientra anche il guidare un **Daily Standup** con lo stile **walk the board** appena visto.

È un **coach**, per cui **stimola la discussione** all'interno del team, porta all'attenzione delle persone il rispetto o meno delle regole che il team si è date, prima fra tutte il rispetto dei WIP limit. Guida la valutazione delle eccezioni stimolando la creazione di idee innovative.

Il **Flow Manager** può essere visto come una figura parallela allo ScrumMaster, con cui ci sono moltissime similitudini: va tenuto presente però che l'attenzione del Flow Manager si concentra principalmente sulla **velocizzazione** del **flusso** e meno sugli aspetti legati alla gestione del progetto [FM].

Quando e come si fa la retrospettiva?

La retrospettiva rappresenta un altro dei **feedback loop** fondamentali: visto che il cuore di Kanban è il **miglioramento continuo**, fare regolarmente delle retrospettive è lo strumento indispensabile per abilitare questo miglioramento.

Retrospettive: differenze tra Scrum e Kanban

Rispetto a Scrum, in Kanban **non** vi è il concetto di **iterazione** che in qualche modo detta il ritmo per l'organizzazione del lavoro, collocando la retrospettiva al termine dello sprint. Per questo motivo, in **Kanban** anche la **frequenza** con cui effettuare le **retrospettive** è **decisa** dal **team**: può essere ogni due settimane, una volta al mese, oppure quando si pensa che sia necessario. Spesso nei team molto maturi c'è la tendenza a non seguire una cadenza regolare, organizzando invece una retrospettiva non appena si pensa che possa essere utile.

Dal punto di vista degli strumenti utilizzabili si può far riferimento alle classiche tecniche di **Agile Retrospective**, utilizzate comunemente dai team agili [RTR]. A causa della popolarità di Scrum, dove la retrospettiva gioca un ruolo fondamentale, la maggior parte delle tecniche e dei formati utilizzati sono quelli adottati dai team che utilizzano Scrum come metodologia di lavoro.

Ciò nonostante, alcuni autori ritengono che passando a Kanban potrebbe essere utile provare a cambiare anche il formato e le tecniche utilizzate per fare le retrospettive. In ogni caso, il tema delle retrospettive è approfondito nella Parte 5 di questo libro.

Stop the Line Retrospective

Un primo interessante esperimento che alcuni team hanno provato a effettuare prende spunto dal lavoro fatto da Taiichi Ōno in Toyota, introducendo il concetto di **Stop the Line Retrospective** ("retrospettiva con interruzione della linea di produzione"): non appena si verifica un problema "grave" — ovvio che bisogna stabilire prima cosa sia un "problema grave" — il team interrompe il proprio lavoro e si riunisce per capire come risolverlo.

Non sempre questo è un approccio praticabile: capita che il problema sia più complicato di quello che sembra e non basta una riunione di un'ora per trovare la soluzione. Alle volte può capitare che questo approccio non porti ad avere sufficiente

coinvolgimento da parte di tutte le persone del team; il rischio, testimoniato da chi ha provato ad applicare questa tecnica, è che alcune persone si sentano prigioniere, aspettando impazientemente di tornare al proprio lavoro, cosa che di riflesso impatta negativamente anche su chi ha sollevato il problema.

Nella produzione *lean* messa a punto in Toyota questa politica funzionò probabilmente perché Ōno impose che tutti partecipassero in modo attivo, secondo un approccio "imperativo" che oggi valuteremmo poco in linea con i principi dell'agilità: bisogna però considerare il periodo storico e il contesto sociale del Giappone post-bellico. Nel contesto attuale di un team di sviluppo, è ipotizzabile che la cosa possa funzionare meno bene.

Pull the Problem

Una variante meno intrusiva e forse più efficace è quella nota con il nome di **Pull the Problem**, dove il team attacca su una lavagna i problemi incontrati: bug da risolvere, necessità di approfondire determinate tematiche o altro che possa impattare negativamente sul flusso della lavorazione.

Su tale board si impone tipicamente un **limite** al **numero massimo** di problemi: non appena tale limite viene **superato**, il team, al termine del primo standup disponibile, si ferma per un momento di riflessione — una retrospettiva appunto — in cui si cerca di analizzare i problemi. Se il tempo non dovesse bastare, si stabilirà una strategia per approfondire meglio il punto, individuando, ad esempio, una porzione di tempo in un altro momento.

Questa tecnica può funzionare meglio rispetto alla Stop the Line Retrospective vista sopra, perché il "tabellone dei problemi" è, a tutti gli effetti, una Kanban board: senza necessariamente arrivare al superamento del limite, tutto il team si impegna a prendere in lavorazione uno dei cartellini di problema appesi nella board non appena vi è un momento disponibile: per questo si usa il nome **Pull the Problem**.

Operations Review Meeting

Un'altra interessante evoluzione del concetto di retrospettiva è quella che Anderson chiama **Operations Review Meeting**, nota anche con il nome di Metrics Review, che tipicamente viene condotta una volta al mese.

Il team in questo caso si sofferma a rivedere lo stato delle metriche, cercando di evidenziare particolari trend o dati insoliti. Sulla base dell'analisi delle misurazioni effettuate, si discute di come stanno procedendo le cose e delle cause degli attuali trend. Utile in questo caso discutere degli esperimenti in corso, valutando gli eventuali effetti che hanno sul lavoro, sempre tramite l'analisi delle metriche in uso.

Quando si pulisce la colonna del "done"

Questo è un aspetto molto interessante soprattutto perché molti lo considerano un dettaglio: si ripulisce la colonna con le attività "già fatte" quando capita, tanto sono

attività terminate. In linea di principio può essere un approccio corretto, anche se forse potrebbe essere utile capire meglio cosa si intenda per **done**.

Alcuni considerano il termine **done** fuorviante; il team di sviluppo infatti tende a considerare terminata un'attività quando viene rilasciata in produzione o quando non c'è più alcun lavoro da fare; per il cliente invece l'attività è realmente terminata quando la funzione rilasciata risolve il suo problema.

In tal senso potrebbe essere più corretto chiamare l'ultima colonna **released**, e non semplicemente "terminata", e verificare di tanto in tanto se quella attività sta avendo l'effetto desiderato: "OK, è stata rilasciata, ma l'utente la sta utilizzando? È soddisfatto? I suoi problemi sono risolti? Sta avendo l'effetto desiderato in produzione? Sta dando il valore che avevamo stimato?".

A volte è utile separare la colonna del done in due parti: quello che è arrivato **oggi** in **done**, e quello che era **già** in **done**. In questo modo, allo standup successivo, si può avere un'idea più precisa di cosa è stato completato, enfatizzando le attività effettivamente completate dall'ultima riunione. Un piccolo cambiamento di nome che ha un'importante implicazione nella **product ownership** della lavorazione delle attività.

Quante storie far entrare nel sistema

Questo riguarda un po' il "quando" si mettono le cose nella kanban board. Molti team che arrivano da Scrum fanno l'errore di selezionare molte più storie di quante ne finiranno prima del prossimo **replenishment**, e quindi sprecano tempo a discutere dettagli e priorità di storie che neanche inizieranno.

Una semplice tecnica per iniziare è guardare quante storie sono completate solitamente tra un **replenishment** e l'altro, e tirare dentro lo stesso numero di storie. Un passo successivo potrebbe essere quello di iniziare a stimare le storie in base alla complessità o, più correttamente, all'**effort** necessario per il loro completamento.

Riferimenti

[EKC] D. J. Anderson, *Essential Kanban Condensed*
http://www.leankanban.com/guide

[RTR] E. Derby – D. Larsen, *Agile Retrospectives: Making Good Teams Great*, Pragmatic Bookshelf, 2006

[BUF] Buffer
https://goo.gl/cTx9pf

[FM] La figura del Flow Manager
http://goo.gl/HH1mlS

Appendice A
Lover's beer game: una metafora per comprendere il Lean

Introduzione

Con questa appendice, vogliamo parlare del cruciale argomento del **flusso di produzione** che è uno dei punti nodali del **Lean**. Attraverso una metafora cercheremo proprio di comprendere il rapporto tra **flusso degli elementi** in un processo e conseguenze sull'intero sistema.

Ci dedicheremo all'analisi delle **dinamiche** dei **sistemi** di **produzione complessi** e vedremo quali sono gli effetti derivanti da una mancanza di sincronizzazione fra domanda e offerta all'interno del processo di produzione. Le oscillazioni che ne scaturiscono possono portare a una situazione di criticità del sistema, se non addirittura al suo collasso. Il fenomeno alla base di questo esempio è detto **Effetto Forrester** e ha a che fare, fra le altre cose, con la sindrome dei **decisori a razionalità limitata**.

Per presentare il problema, useremo una simulazione nata da un'idea formulata negli anni Sessanta del secolo scorso presso la Sloan School of Management del MIT; si tratta di un **gioco** di **ruolo**, il **Lover's beer game,** in cui gli attori coinvolti devono agire rispettando alcune regole e muovendosi in un ambiente vincolato.

Figura 43. Il Beer Game è stato giocato in molte sessioni di serious gaming come strumento per comprendere alcuni concetti basilari della produzione snella (lean).

Giocare per comprendere la realtà

Si riporta in questo paragrafo il risultato di una tipica sessione di gioco; dalle innumerevoli "partite" giocate in tutti questi anni, nell'ambito dei vari corsi di **management** e **lean manufacturing**, è emersa una chiara tendenza: il più delle volte, il risultato finale è stato proprio quello che racconteremo nella storia.

Peter Senge, descrive dettagliatamente questo gioco nel libro *La quinta disciplina* [5TH] e, a tal proposito, ci conferma che il risultato finale **non** dipende dai singoli giocatori, ma dal modo in cui è configurato il **sistema giocatori + ambiente + vincoli**: "non sono le scelte delle singole persone o le cause esterne a condizionare il sistema, ma è il sistema stesso". Interessante notare che il gioco in realtà ha una **corrispondenza** forte con la **realtà**: nella storia recente ci sono stati infatti diversi casi in cui si è potuto assistere a comportamenti sistemici analoghi a quelli descritti in questo capitolo.

Si può prendere per esempio il caso della crisi della produzione dei microchip verificatosi nel 1985, in cui una serie di oscillazioni della domanda/offerta, impattarono sui prezzi e sulla disponibilità dei pezzi, nonché sui ricavi. Una situazione analoga si era verificata un decennio prima nel settore dei semiconduttori, in cui furono coinvolte aziende del calibro di **Siemens** e **Honeywell**. Un altro caso lampante fu quello che si ebbe verso la fine degli anni Ottanta nel settore dell'automobile, dove **General Motors**, **Ford** e **Chrysler** furono vittime di una situazione del tutto analoga.

Per motivi di spazio, la versione del gioco riportata in questo capitolo è necessariamente sintetica. Chi fosse interessato a maggiori dettagli, può senza dubbio far riferimento a *La quinta disciplina* [5TH], libro che è diventato una pietra miliare per la teoria del pensiero sistemico.

La birra Lover's, i tre "attori" e il processo

Questa storia si basa su un prodotto di fantasia, la **Lover's Beer**, una marca di birra inventata ma che faremo finta essere piuttosto popolare fra i giovani di una ipotetica regione degli USA. Questa birra è economica e viene consumata con regolarità ma **senza** raggiungere volumi di vendita delle birre più diffuse. Un bel giorno le vendite aumentano improvvisamente senza un evidente motivo; nel libro si spiega che questa variazione è dovuta all'apparizione della birra nel video musicale di un gruppo piuttosto famoso fra i giovanissimi: per questo tutti i ragazzi in grado di comprare alcolici iniziano a comprarla con più frequenza.

Commerciante, grossista, produttore: tre punti di vista

Gli attori coinvolti in questa storia sono, oltre ai clienti finali, un **commerciante al dettaglio** che vende fra le altre cose la birra Lover's, un **grossista** che rifornisce i negozi di alcolici della zona, fornendo anche la suddetta birra e, infine, la **fabbrica produttrice** della Lover's.

Per spiegare cosa succede in questa simulazione verranno fornite **tre versioni** della stessa storia, dal punto di vista del **commerciante** al dettaglio, del **grossista** e del

produttore. In questa simulazione si immagina che i tre attori **non comunichino** fra loro per telefono o di persona, ma si limitino a scambiarsi ordinativi e consegne tramite un sistema ormai rodato.

Un altro fattore da tenere presente è che, per motivi legati al processo di produzione e di distribuzione, ogni ordinativo viene poi consegnato **dopo 4 settimane**. A causa dell'organizzazione del sistema — un unico fabbricante, un rete di distribuzione limitata e un ampio numero di rivenditori finali — tale ritardo si propaga su tutti i nodi della rete.

Il punto di vista del commerciante

Il **commerciante** al dettaglio **Tom** vende nel suo negozio birre di vario tipo; ma in questo gioco concentriamo l'attenzione su un prodotto in particolare: la **birra Lover's**, prodotto che, forse in virtù del suo prezzo, è particolarmente apprezzato dai più giovani. Per questo motivo Tom sa che regolarmente, settimana dopo settimana, riesce a vendere **4 casse** di questa birra.

Gli ordinativi sono effettuati a inizio settimana e, a causa della regolarità delle vendite, Tom richiede sempre lo stesso quantitativo al grossista inviando un ordine che ormai non ha bisogno di ulteriori spiegazioni.

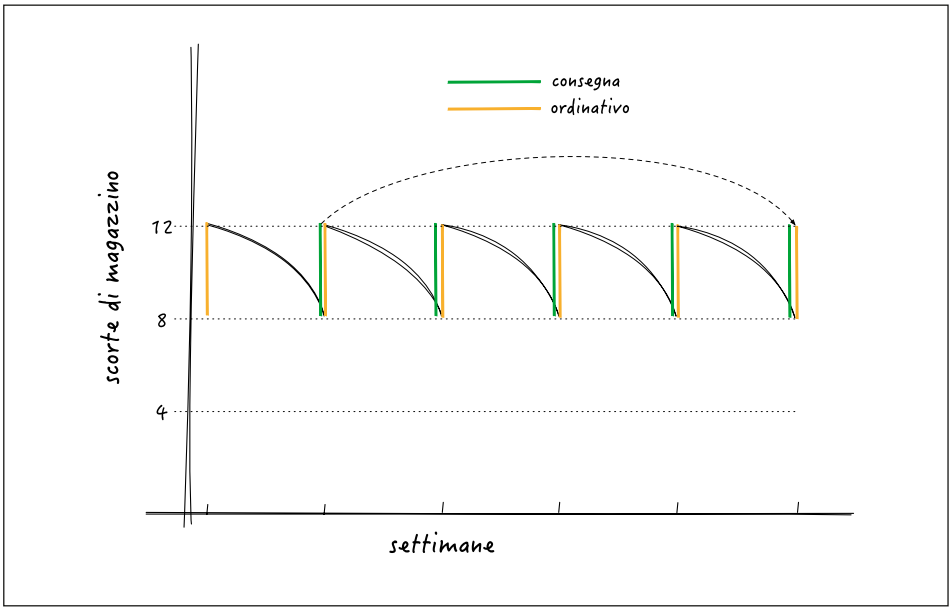


Figura 44. Il ciclo di vendita della birra Lover's, misurato al punto vendita al dettaglio, segue un andamento molto regolare; il negoziante, ordinando sempre lo stesso quantitativo di birra, è in grado di asaudire le richieste di tutti i suoi clienti e al contempo di mantenere stabili le scorte di magazzino.

Ordini e consegne

Per semplificare il ragionamento, immaginiamo che le **consegne** della merce e gli **ordinativi** per il futuro avvengano contestualmente, nello stesso istante: per esempio, al lunedì, il camion del grossista consegna a Tom le 4 casse di birra, e il commerciante consegna all'autista un foglio su cui è scritto l'ordine successivo.

Tom non conosce quindi il grossista, né tantomeno conosce i dettagli di chi e come produca la birra, fattore che, come potremo vedere fra un po', si rivelerà fondamentale... in negativo.

Egli sa che ogni ordine viene consegnato dopo 4 settimane, ma non se ne preoccupa, sia per la regolarità con cui riesce a vendere tale birra, sia perché tiene in magazzino una piccola scorta aggiuntiva, in modo che, ad ogni inizio settimana, abbia sempre 12 casse di questo prodotto. Quindi la scorta oscilla fra le 8 e le 12 casse, cosicché egli possa mantenere sempre un certo margine di sicurezza.

Questo accade regolarmente da molto tempo dando luogo a un andamento altalenante ma piuttosto ciclico e in media costante, che è rappresentabile con il grafico riportato in figura 44.

L'andamento delle vendite e dei rifornimenti, settimana dopo settimana

Prendiamo in esame una settimana qualsiasi di questo periodo in cui il comportamento è stabile, settimana che chiameremo convenzionalmente **settimana n. 1**: in questa settimana il commerciante viene rifornito dello stesso quantitativo di birra che ha ha venduto la settimana precedente.

Poi improvvisamente, **le cose cambiano**: senza un motivo apparente, al termine della settimana successiva, la **settimana n. 2**, le vendite aumentano raddoppiando da **4** casse a **8** casse. Il venditore non se ne preoccupa più di tanto, visto che la variazione, pur sostanziosa in percentuale, è comunque un evento isolato... o almeno così crede: forse è stato più caldo del solito oppure forse qualcuno ha dato una festa. Inoltre le scorte che Tom ha in magazzino (8 casse oltre alle 4 che oscillano in entrata e uscita) gli permettono di "assorbire" questo sbalzo.

Ciò nonostante, si prepara ad aumentare l'ordinativo il lunedì successivo — quello che dà inizio alla **settimana n. 3** — in modo da pareggiare l'aumento delle vendite riportando il magazzino a 12 casse. Purtroppo all'inizio della **settimana n. 3** il rifornimento che arriva è di sole 4 casse: non dimentichiamo infatti che **ogni consegna** si riferisce all'**ordine** fatto **4 settimane prima**.

Pertanto, complessivamente, il magazzino è sceso da 12 a 8 casse. Se la variazione nelle vendite è frutto di un caso isolato — qualche festa in più, il caldo dell'estate o chissà cos'altro — Tom sa che nell'arco di 4 settimane il suo magazzino tornerà ad avere una scorta di 12 casse: in fondo, la scorta serviva proprio per assorbire eventuali oscillazioni.

Per il momento Tom quindi non si preoccupa particolarmente e continua a comportarsi in **maniera razionale** sulla **base** della **propria esperienza**.

Cambiamenti decisivi

Durante la **settimana n. 3**, il commerciante comprende che quanto accaduto non era un caso isolato, ma che anzi le vendite continuano su un ritmo di 8 casse a settimana. Essendoci in magazzino solamente 8 casse, al termine della terza settimana il negoziante si trova senza birra.

Alla consegna del lunedì della **settimana n. 4**, riceverà solamente 4 casse di birra: l'ordinativo maggiorato è stato fatto solamente alla settimana n. 3 per cui le 8 casse arriveranno alla settimana n. 7.

Deve quindi risolvere rapidamente il problema e sa che probabilmente a metà settimana si troverà senza birra, andando quindi in debito di merce; per esempio potrebbe segnarsi il nome dei clienti e richiamare quando la merce sarà nuovamente disponibile.

Per questo, nemmeno le 8 casse ordinate la settimana precedente possono bastare: adesso non solo deve pareggiare lo svuotamento del magazzino ma anche fronteggiare il "debito" che si creerà fra pochi giorni. Per questo, decide di **agire di anticipo** e aumenta l'ordinativo a 12 casse.

Proseguendo questo trend, settimana dopo settimana Tom si trova ad aumentare sempre più il proprio ordinativo fino alla settimana 7: adesso il trend delle consegne dovrebbe invertirsi, dato che all'inizio di **settimana n. 7** dovrebbe finalmente arrivare la prima consegna maggiorata (8 casse).

Scarsità di merce

In realtà, al lunedì di **settimana n. 7**, il grossista consegna solo 5 casse rispetto alle 8 ordinate. E nelle settimane successive le cose non cambiano; anzi addirittura peggiorano. Completamente in balia di fattori indipendenti dalla sua volontà, il negoziante non può far altro che arrendersi all'evidenza dei fatti: **senza conoscere** il **motivo** della crescita della domanda né il motivo della scarsità delle consegne**, non** riuscirà a soddisfare le richieste dei propri clienti. Il problema principale è che la mancanza di birra Lover's ha rallentato la vendita di merce di contorno normalmente acquistata dai clienti che comprano la Lover's.

Il punto di vista del grossista

Susan è la responsabile degli ordini e delle vendite di un **grossista** di zona. Nel suo ruolo, Susan è testimone di un andamento analogo a quello che ha sperimentato Tom, il negoziante, anche se ovviamente con volumi certamente maggiori e su un altro tipo di scala: Susan è il riferimento per il rifornimento di **molti negozi** della sua zona, piccoli supermercati o genericamente intere zone molto ampie ma a bassa densità di popolazione.

Se un negoziante ragiona per le consegne in termini di casse di birra, l'unità di misura di Susan sono i **bancali** che sono movimentati tramite camion verso i vari negozianti della zona coperta dalla propria zona. Come il negoziante finale, anche Susan ha un **tempo di attesa di 4 settimane** per ricevere la birra che ordina alla fabbrica.

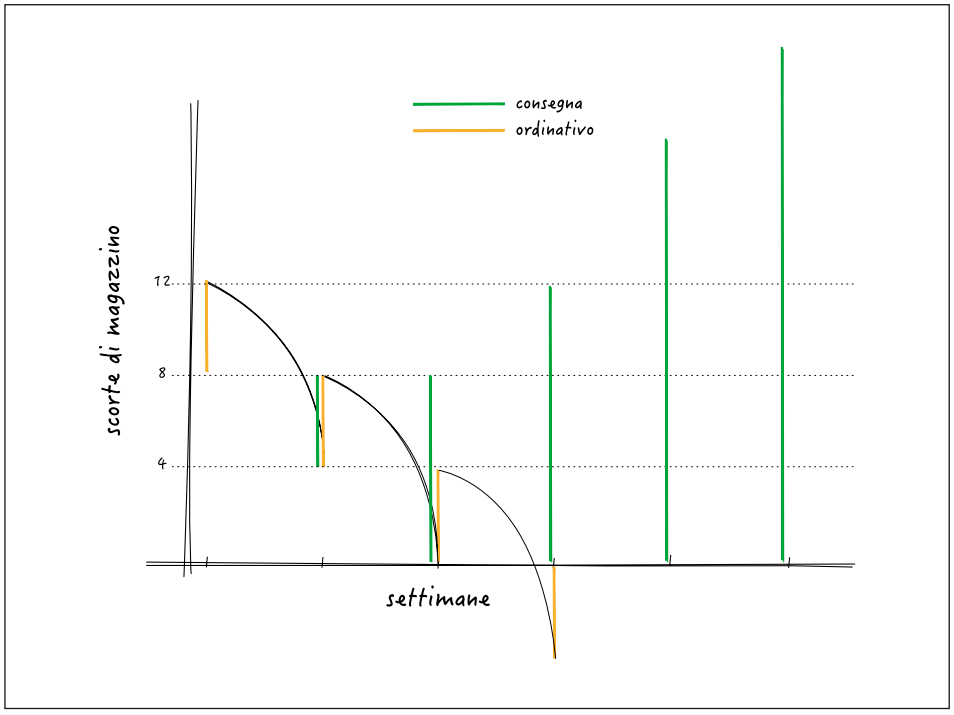


Figura 45. In questo grafico è riportato l'andamento delle vendite al negozio: l'alterazione nella domanda consuma in breve tempo le scorte del negozio e obbliga il negoziante a impegnarsi con i clienti più affenzionati a consegnare la birra non appena questa arriverà in negozio. Per questo il grafico scende sotto il livello dello zero.

L'andamento delle vendite del grossista

Anche Susan riscontra un aumento delle vendite alla **settimana n. 4**. A dire il vero aveva avuto una piccola **oscillazione** alla **settimana n. 3**, ma anche nel suo caso aveva ipotizzato si trattasse di una variazione più o meno casuale.

Il **grossista**, per la sua posizione all'interno della rete di vendita, di fatto **concentra** e **somma** l'**andamento** delle vendite dei **negozianti** che dipendono da lui. Per questo l'oscillazione che Susan nota alla terza settimana è quasi impercettibile, mentre quella della quarta è sostanziale e maggiore di qualche ordine di grandezza.

Il **grossista**, non appena si rende conto che il nuovo trend non è una semplice oscillazione, interviene modificando gli ordini: anche in questo caso interviene prima con un semplice incremento poi con maggiorazioni più sostanziose.

Susan pensa che se la variazione delle vendite non sarà troppo repentina, le **scorte** di magazzino dovrebbero essere sufficienti per assorbire l'aumento delle richieste in attesa che la fabbrica consegni gli ordinativi aumentati, vale a dire **dopo dopo 4 settimane**.

Scorte di magazzino insufficienti

Purtroppo per Susan, l'aumento di domanda da parte dei commercianti al dettaglio non si arresta, ma anzi aumenta man mano che passa il tempo. Se inizialmente la variazione poteva essere assorbita con le scorte di magazzino o comunque ridistribuendo le consegne ai vari negozianti, dopo qualche tempo arrivano alla dimensione di criticità massima.

Susan, con qualche settimana di ritardo rispetto al negoziante, sperimenta la sua criticità: non appena si rende conto che le nuove richieste stanno erodendo le scorte, reagisce prima razionalmente, aumentando quel tanto per colmare lo svuotamento del **magazzino**; successivamente, trovandosi in difficoltà, prova forzare la mano con la fabbrica aumentando in maniera "sostanziosa" gli ordinativi alla fabbrica.

Anche il **grossista** quindi inizia ad assumere un atteggiamento **emotivo**, guidato dall'**ansia** di non riuscire a evadere tutti gli ordini: i **commercianti** continuano a chiedere sempre più birra e il **grossista** non ne ha. Come avremo modo di vedere, la fabbrica aumenterà la produzione fra qualche settimana e i risultati si vedranno solo fra qualche tempo. Nel frattempo che fare?

Arrivano i nostri...

Verso la **settimana n. 16**, la **fabbrica**, che, in concomitanza dell'aumento delle richieste da pare dei grossisti, aveva deciso di potenziare le proprie strutture e di produrre più birra, inizia a rispondere all'aumento degli ordinativi consegnando più merce di prima, riuscendo in questo modo a coprire quasi tutti gli ordini fatti dai **grossisti** 4 settimane prima.

Susan inizia a essere fiduciosa: la merce appena ricevuta non è ancora sufficiente a coprire il debito di consegne, ma è rassicurata dall'inversione di tendenza da parte della fabbrica e si attende quindi che le prossime consegne potranno colmare tale carenza. I suoi ordinativi infatti sono aumentati in maniera esponenziale.

Inversione di tendenza

Contemporaneamente, dal canale di vendita si nota una leggera **flessione** negli ordinativi. Susan non se ne cura, pensa che molto probabilmente i negozianti, che avevano più o meno seguito lo stesso trend negli ordinativi, e avevano ecceduto un po' nella settimana precedente, adesso rallentano un po'.

In questa fase, con il produttore che inizia ad aumentare la produzione e il magazzino che finalmente inizia a riempirsi nuovamente, il rallentamento degli ordinativi da parte dei negozianti è visto come un fatto positivo. In fondo Susan ha ancora un debito considerevole con i negozianti.

Nelle settimane successive le consegne da parte della fabbrica iniziano ad essere regolari e rispettare quanto richiesto al momento dell'ordine **quattro settimane prima**, quando Susan aveva aumentato gli ordinativi del doppio e poi del triplo rispetto alla **media storica**.

Ora che le cose iniziano a mettersi al meglio, accade una cosa impensabile fino a poco prima: purtroppo, gli ordinativi da parte dei **commercianti** si bloccano del tutto. Nessun venditore finale sembra aver più bisogno di birra Lover's!

Lo stato delle cose

La cosa drammatica è che, nelle settimane seguenti, le consegne da parte del produttore, sempre in accordo con quanto richiesto 4 settimane prima, aumentano esponenzialmente. Adesso stanno consegnando 10 se non 20 volte rispetto alla **media storica**. La **merce** si **accumula** nel **magazzino** mentre i negozianti continuano a non ordinare nulla.

Susan si rende conto che, con il trend attuale, se anche i negozianti dovessero riprendere a ordinare birra Lover's, ci vorrà molto tempo per poter **smaltire** tutta quella merce che si sta accumulando sui bancali in magazzino. Probabilmente non ci riuscirà e probabilmente è a rischio la sopravvivenza della sua azienda.

Il tutto senza che il **grossista** abbia compreso il reale motivo di quanto accaduto. Ad oggi non ha ancora avuto modo di parlare né con qualcuno dei **commercianti** né con un rappresentante della **fabbrica** della Lover's Beer; è stata sempre troppo occupata a gestire gli sbalzi nella lavorazione. La sola cosa che può fare per le prossime settimane è bloccare completamente ogni ordinativo per l'acquisto di altra birra.

Il punto di vista del produttore

Presso la **fabbrica** di produzione della birra Lover's, l'andamento delle vendite segue un comportamento molto simile a quello che abbiamo visto presso il **grossista** e i singoli **rivenditori finali**, anche se ovviamente con volumi notevolmente maggiori.

Dopo una piccola variazione alle prime settimane, le richieste hanno registrato un sensibile **aumento**, con un trend che in poco tempo ha messo in difficoltà il reparto produttivo che non è riuscito a rimanere in pari con le sempre maggiori richieste. Analogamente, anche in questo caso, dopo questo picco di ordinativi, le richieste hanno subito un brusco **rallentamento**, per arrivare alla totale interruzione di ogni forma di ordinativo.

Le variazioni negli ordinativi

Il primo aumento negli ordinativi per i volumi della fabbrica non viene realmente percepito come **variazione della domanda**, ma come una **perturbazione circoscritta**. Anche in questo caso i responsabili alle vendite e alla produzione non hanno preso alcuna iniziativa, immaginando che l'alterazione potesse tranquillamente essere assorbita con le scorte di magazzino che sono sempre disponibili presso la fabbrica.

Quando successivamente si è visto che il trend degli ordinativi rimaneva in costante crescita, i responsabili delle vendite hanno compreso che non si trattava di una variazione isolata, ma di qualcosa di più strutturato.

In quell'istante si è provato a tamponare con un intervento sulle modalità di consegna, cercando di ottimizzare le poche scorte di merce disponibile, per esempio

provando a soddisfare i distributori più importanti o mandando solo parte della merce ordinata. Contemporaneamente i responsabili della produzione si sono attivati per intervenire strutturalmente: prima assumendo nuovi dipendenti, poi provvedendo a potenziare i macchinari per realizzare volumi maggiori di birra.

Purtroppo, prima che le nuove attrezzatura potessero entrare in funzione e potessero consentire un sostanziale incremento della produzione, gli ordinativi si sono interrotti di colpo. Tutto a un tratto nessuno voleva più birra.

Cosa era successo? Oltre al rammarico di non essere riusciti a soddisfare una notevole quantità di richieste (mancati profitti) la fabbrica si ritrova adesso con dei costi strutturali notevolmente aumentati (nuovi assunti e attrezzature da pagare). Per la fabbrica, se gli ordinativi non fossero tornati ai volumi registrati prima dell'interruzione, tutto ciò avrebbe potuto significare un grosso problema.

Il punto di vista del responsabile del marketing e vendite della fabbrica

Nel libro di Senge, il racconto prosegue con l'aggiunta di un'**appendice** in cui si narra del punto di vista di **Marc**, il **responsabile del marketing e delle vendite** della fabbrica. Marc è stato assunto di recente all'interno dell'azienda con lo scopo di far crescere le vendite e si ritrova a vivere in pieno tutta l'oscillazione: da un incremento incontrollato al successivo crollo.

Quando Marc vede che la birra inizia ad accumularsi nel magazzino, nel tentare di **comprenderne** le **cause** e trovare un rimedio, intraprende un viaggio — purtroppo tardivo — presso la rete di vendita andando a parlare con Susan, la responsabile di uno dei **grossisti** che distribuisce la birra Lover's. La conversazione che si instaura mette in evidenza in modo piuttosto chiaro alcune possibile cause di quanto è successo.

Il confronto tra fabbrica e grossisti

Susan accoglie Marc, mostrando le scorte all'interno del proprio magazzino, scorte che si sono accumulate non appena dalla fabbrica hanno iniziato a consegnare gli **ordinativi** maggiorati. Marc frustrato e arrabbiato si interroga a voce alta su come tutto questo sia stato possibile...

Susan, cercando di immaginare quanto tempo ci vorrà per smaltire tutte quelle scorte, condivide la frustrazione e la rabbia di Marc. Entrambi sono frastornati e preoccupati della situazione.

Marc esordisce: "È una tragedia... ma come è potuto succedere?".

Susan, dimostrando anche in questo caso una visione locale, perché non pensa di avere alcuna responsabilità, molto semplicemente ribatte: "Non è colpa nostra; sicuramente i commercianti al dettaglio non si sono saputi organizzare o forse non sanno gestire il mercato".

Marc, nella sua testa, dà la colpa ai rivenditori, accomunando, in preda a un momento di sconforto emotivo, **commercianti** al dettaglio e **grossisti**. Certamente quanto

accaduto è l'ennesima dimostrazione della variabilità del mercato e dell'incostanza dei consumatori. I due si congedano senza avere alcuna risposta per risolvere il problema, ma promettendo di risentirsi nei giorni successivi per provare a condividere qualche strategia.

Il confronto tra frabbrica e commercianti al dettaglio

Nel viaggio di rientro verso il suo ufficio, incidentalmente **Marc** si trova a passare davanti a un negozio che vende, fra le altre cose, la **Birra Lover's**. Per la prima volta da quando lavora alla Lover's, entra nel negozio di un **commerciante al dettaglio** per parlare con il titolare, **Andrew**.

Dopo le presentazioni di rito, Andrew porta Marc nel retro del proprio negozio per mostrare le giacenze di magazzino e anche in questo caso la scena, con le dovute proporzioni, è simile: il retro del negozio è pieno di casse di birra Lover's. Con il trend di vendita attuale, per vendere tutte quelle casse di birra ci vorranno molte settimane. Marc rimane pietrificato: se tutti i negozianti della zona e delle aree vicine sono nella medesima situazione, la fabbrica corre grossi rischi per la propria sopravvivenza, perché non dovrà produrre più nulla per mesi.

Marc chiede i motivi di questo stallo improvviso; il **commerciante** serenamente risponde: "Non saprei, non è colpa nostra. Dopo che è uscito quel video musicale in cui appariva la Lover's, abbiamo avuto un incremento nelle richieste dai nostri clienti; da 4 casse di media, siamo passati a 8 nel giro di una settimana..."

Marc interrompe: "Va bene. Prima le vendite sono esplose... Ma come mai poi sono crollate?"

Andrew ribatte: "No, attenzione! Le **vendite non sono né esplose né crollate**; noi adesso continuiamo a vendere 8 casse di birra a settimana rispetto alle 4 iniziali. Il problema è che voi non ci stavate mandando la birra che ci serviva e stavamo accumulando ritardi e per mantenere i nostri clienti abbiamo dovuto aumentare gli ordinativi".

Marc: "Ma noi abbiamo consegnato gli ordinativi non appena ci è stato possibile!"

Andrew: "Non so che dire... Forse il grossista ha sbagliato qualcosa, tant'è che vorremmo cambiarlo. Di fatto abbiamo ricevuto la birra con ritardo, addirittura in un certo momento, nonostante noi aumentassimo le richieste, le consegne arrivavano incomplete, in misura molto minore rispetto all'ordine. Quando poi finalmente avete iniziato a consegnare la merce, ci avete mandato prima la giusta quantità, poi molto più del necessario. A quel punto abbiamo dovuto interrompere gli ordini".

La storia fin qui

Ricapitoliamo brevemente la sequenza di eventi che si sono susseguiti:

- Una piccola oscillazione nella domanda da parte dei consumatori finali viene prima sottovalutata dalla rete di vendita, poi presa in seria considerazione dal negoziante, che prima sopperisce con le scorte di magazzino, poi provvede ad aumentare gli ordinativi al grossista.

- Le consegne avvengono con 4 settimane di "ritardo" dalla effettiva creazione dell'ordine: per 4 settimane quindi il negoziante non riceve alcun incremento della merce consegnata.
- Il negoziante, prima segue un approccio razionale; quando invece le scorte non bastano più, e non è in grado di soddisfare la sua clientela più affezionata, inizia a reagire in maniera impulsiva aumentando gli ordinativi in misura maggiore di quella realmente necessario. Confonde l'aumento della richiesta settimanale, con le richieste passate non soddisfatte, come se tutti i clienti che non hanno comprato nei giorni in cui la birra non era disponibile, si ripromettessero poi di comprare anche gli "arretrati".
- Questo schema si verifica con lo stesso andamento presso gli altri nodi della filiera (distribuzione e produzione), cosa che porta al blocco delle consegne: la fabbrica non era pronta all'aumento improvviso delle richieste.
- Non essendo disponibile prodotto già presso la fabbrica, i venditori finali dopo le 4 settimane canoniche cominciano a ricevere addirittura meno scorte di quelle ordinate. A questo, reagiscono in modo ancora più irrazionale aumentando le richieste.
- Finalmente il processo si sblocca, perché la fabbrica si attrezza per moltiplicare la sua capacità produttiva. Ma a questo punto, le richieste da parte di venditori finali e distribuzione si bloccheranno, perché entrambi si son visti recapitare un quantitativo di merce enormemente maggiore rispetto alle necessità.

Ancora una volta è necessario sottolineare che la variazione della domanda finale era minima e che nessun cliente probabilmente avrebbe poi comprato tutta la birra che non aveva trovato a scaffale nelle settimane precedenti.

Il processo quindi si traduce in una specie di **onda** fra domanda e offerta, che ha messo in difficoltà tutti i nodi della filiera: il negoziante prima non ha saputo soddisfare la richiesta dei clienti; poi si è ritrovato nel magazzino un'eccedenza di birra che per essere smaltita avrebbe richiesto molte settimane; il distributore ha subìto una sorte analoga e quindi, come il negoziante, ha interrotto gli ordinativi quando si è trovato nel magazzino bancali pieni di birra.

In fabbrica inizialmente non sono stati in grado di evadere l'incremento degli ordinativi ma non hanno subito alcun danno economico diretto, se non mancanza di un ricavo concreto; poi, quando hanno fatto gli investimenti necessari per incrementare la produzione, hanno subìto l'arresto della domanda, con conseguente impossibilità di ripagare gli investimenti fatti, oltre anche in questo caso ai magazzini pieni.

Alcune considerazioni

Anche se la storia nel libro di Senge prosegue ancora, noi ci interrompiamo in questo punto, dato che le cose viste mettono in evidenza già alcuni fattori molto interessanti. Questa storiella ha, ovviamente, tutti i limiti di una storia "didattica" e con scopi esemplificativi. Da una analisi distaccata della storia appare piuttosto chiaro dove sia il problema; ma perché è accaduto? Perché **ogni attore** della **filiera** ha commesso un errore

così banale agendo senza **tenere in considerazione le tempistica di consegna** e non ha saputo valutare gli aspetti **logistici** e le **conseguenze** delle proprie azioni?

Per provare a rispondere a queste domande, è forse utile analizzare la storia sia da un punto di vista **sistemico**, come fa Senge nel suo libro, sia da un punto di vista **organizzativo** e di **processo**, ossia applicando alcune delle considerazioni fatte in precedenza.

Considerazioni sistemiche

Perter Senge fornisce tre massime che emergono dalla storia appena raccontata:

- la struttura influenza il comportamento;
- la struttura dei sistemi umani è "sottile";
- la soluzione può arrivare da nuovi modi di pensare.

La struttura influenza il comportamento

Studi e analisi in svariati settori hanno dimostrato che **differenti persone** hanno prodotto **risultati simili** se messe a lavorare nello **stesso contesto** o in contesti simili.

Quando ci sono cali di performance o problemi di una qualche natura, la prima tendenza è quella di cercare qualcuno o qualcosa su cui far **ricadere la responsabilità**. In realtà le cause delle decisioni che prendiamo nelle nostre iniziative all'interno di un'organizzazione sono frutto dell'organizzazione stessa; quasi mai la responsabilità è in capo a un singolo o a qualcosa di esterno, ma ricade in gran parte sul **sistema organizzazione**, in cui ricomprendiamo le persone, le regole, i processi, i flussi informativi, gli stakeholder, le risorse. Probabilmente ognuno di noi, al posto di Tom si sarebbe comportato nello stesso modo.

La struttura dei sistemi umani è "sottile"

Siamo portati a pensare al concetto di struttura come un insieme di vincoli che agiscono sugli individui. Ma la struttura nei **sistemi complessi** può essere definita sulla base delle **sofisticate interrelazioni** che si creano al suo interno e che ne regolano il comportamento finale.

In un'azienda, come nell'esempio della birra Lover's, il risultato del sistema di vendita, dal consumatore al produttore, è dato dalla complessità **risultante** dalle azioni e decisioni dei vari attori che si muovono all'interno del sistema. Nessuno di questi può pilotare il sistema, ma eventualmente influenzarne il comportamento che sarà comunque il risultato dell'azione di tutti.

La soluzione può arrivare da nuovi modi di pensare

Quando ci troviamo ad agire per affrontare dei problemi all'interno di **organizzazioni complesse**, pur avendo le risorse e le capacità per **risolvere** determinare situazioni, non le esercitiamo perché troppo presi dal focalizzare la nostra attenzione sulle nostre decisioni, senza renderci conto che le nostre azioni interferiscono con quelle degli altri.

Nella storia della birra Lover's, i **singoli attori** osservano e si concentrano **solo** sul **proprio operato** cercando una **soluzione locale** che abbia ricadute immediate nel **proprio ambito**, senza invece considerare l'**intero** sistema in cui tutti agiscono e si influenzano.

E quindi?

Avere quindi una visione sistemica, come proposto da Peter Senge, è di fondamentale importanza per interpretare il comportamento degli attori nella simulazione della Lover's Beer.

Di seguito, proveremo a comprendere come sia possibile spiegare quanto successo nella storiella della Birra Lover's grazie agli strumenti tipici dell'agilità, applicando i concetti visti in precedenza: **Lean Production**, **Cycle Time**, **Kanban**, **metriche** e altro ancora.

Arricchiremo il nostro bagaglio di conoscenze introducendo un interessante fenomeno, detto **effetto Forrester** (o "effetto frusta" o *bullwhip effect*) e vedendo come questo sia causato fra le altre cose dai cosiddetti **decisori a razionalità limitata** e dalla mancanza di alcuni requisiti che sono ritenuti essenziali all'interno di un progetto agile.

Effetto Forrester

Da un punto di vista teorico, il comportamento del sistema preso in esame e le conseguenze delle azioni intraprese dai vari attori, sono spiegabili mediante il cosiddetto **rffetto Forrester**, detto anche **effetto frusta** o **Bullwhip**.

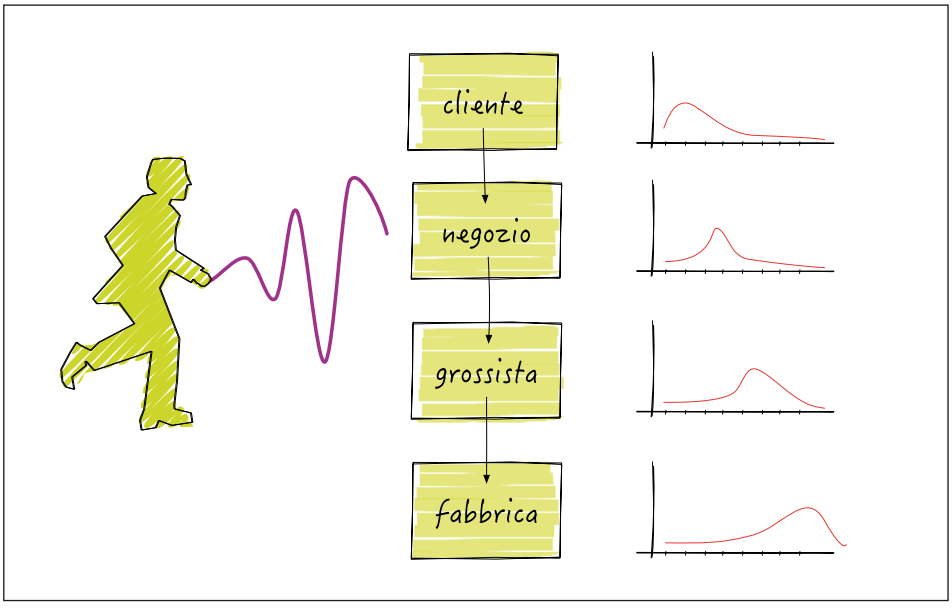


Figura 46. L'effetto Forrester o effetto "frusta".

In determinate condizioni, tale fenomeno si manifesta all'interno di una catena di produzione, tramite una oscillazione della domanda e dell'offerta portando alla formazione di una specie di onda che, proprio come la corda di una frusta, si propaga **amplificandosi** man mano che ci si allontana dal mercato finale e si risale la catena di produzione. È tipicamente applicabile ai sistemi di catena di fornitura ma, come vedremo, lo possiamo vedere in azione anche all'interno di un sistema IT come quello della "fabbrica" del software.

Cause principali dell'effetto "frusta"

Probabilmente la causa principale che innesca un effetto frusta è la presenza di attori che prendono decisioni sulla base di **informazioni circoscritte** solo al loro ambito di intervento e che, per questo, vengono chiamati decisori a "**razionalità limitata**": essi non hanno informazioni dalla periferia rispetto al loro contesto d'azione, ossia non hanno sotto controllo tutto il sistema nel suo complesso.

In questo contesto, maggiore è la **lunghezza** della catena di produzione/informazione, maggiore sarà l'**inefficienza** del sistema complessivo; più precisamente tanto più ci si allontana dal cliente finale — o più genericamente dal punto di verifica — maggiore sarà l'errore delle informazioni e quindi il rischio derivante dalle azioni intraprese. Per questo si parla di **onda**, oscillazione o genericamente di **effetto frusta** (figura 46).

Filiera lunga

In agilità, quando si deve sviluppare un prodotto software, o genericamente un prodotto IT, la lunghezza, o meglio la cortezza, della filiera di produzione è un aspetto preso in seria considerazione, visto che è noto a cosa possa portare una catena lunga.

In Scrum per esempio si cerca di ridurre il numero di intermediari organizzando team in cui tutti gli attori si parlano, collaborano e decidono. Per questo si cerca di eliminare gli **intermediari**, di mettere in **comunicazione direttamente** il team con il cliente/utente e il team con il management.

È per questo motivo che un Product Owner che faccia da intermediario fra il cliente e il team di sviluppo incorre nel rischio di allungare la catena.

Tempi di lavorazione

Altro fattore che impatta negativamente è il **tempo di lavorazione** del prodotto: come abbiamo avuto modo di vedere in modo dettagliato nei capitoli precedenti, il tempo di lavorazione, o **Lead Time**, impatta sulle prestazioni complessive del sistema.

In questo caso, è causa — o effetto, dipende dai punti di vista — delle decisioni prese da un lato o dall'altro della catena: se il negoziante avesse potuto avere la sua birra in tempi più rapidi, avrebbe potuto prendere le proprie decisioni in maniera più coerente con la realtà delle cose. Analogamente, se la fabbrica avesse evaso gli ordini in tempi minori, avrebbe potuto recepire i segnali dal mercato in modo più realistico e magari prendere per tempo le adeguate contromisure.

Latenza

Lunghezza della **catena** ed elevato **Lead Time** sono due fattori che ci portano a considerare le cose da un altro punto di vista, quello della **latenza** nella **risposta** del **sistema** alle **sollecitazioni esterne**.

Questo fenomeno si può spiegare con un paio di esempi piuttosto semplici: uno che probabilmente molti lettori possono aver sperimentato direttamente; l'altro solo se si è partecipato a un corso per imparare a pilotare un aereo.

Il primo è quello che si sperimenta quando per esempio si prova a guidare una bicicletta o una motocicletta con il tubo canotto dello sterzo che non si muove con la dovuta libertà (serraggio troppo alto o qualche cuscinetto bloccato). Per guidare una bici normalmente si apportano delle piccole e costanti correzioni alla traiettoria per rimanere in equilibrio. In questo caso il manubrio risponde in maniera dura o addirittura a scatti: la guida non è armonica ma anzi scattosa. Il ciclista inizia con delle piccole oscillazioni per finire con degli ondeggiamenti più ampi, che portano a volte a cadere.

Un effetto simile capita a chi pilota un aereo: per correggere l'assetto egli tira la cloche per alzare il muso. Ma l'effetto della azione non è immediato, per cui il pilota inesperto tira la cloche più del dovuto; dopo poco il muso dell'aereo si alza più di quanto necessario, e per questo il pilota dà una correzione vistosa verso il basso. Ma siccome anche in questo caso l'effetto non è immediato, il pilota alle prime armi comincia ad avere un po' di ansia… e abbassa troppo. Quando il muso si abbassa, allora tira di nuovo la cloche... e la tira troppo.

Di fatto in quel caso il tipo di risposta del sistema non può essere modificata, per cui pare che nell'addestramento ai piloti venga insegnato proprio ad evitare questo loop pernicioso, apportando piccole modifiche e riportando sempre la cloche in assetto.

Il tema della **latenza** nella risposta di un sistema è uno degli aspetti più importanti su cui si fonda buona parte della filosofia **agile**: seguire **iterazioni brevi** in Scrum, organizzare **frequenti incontri** con il cliente/utente per fare delle demo, raccogliere il maggior numero di **feedback** possibili sono tutte azioni che vanno nella direzione di aumentare la **reattività** del sistema, ossia intraprendere decisioni i cui effetti siano verificabili prima possibile.

Alcune cause dirette dell'effetto "frusta"

Accorciare la filiera, ridurre il lead time, aumentare la reattività sono tre strategie efficaci per limitare gli effetti negativi della **frusta**. Esistono poi anche altri fattori che possono essere causa di effetto Forrester e che in genere sono sintomi di un'elevata oscillazione della tendenza **domanda/offerta**.

Eccessivo livello di scorte

Un tipico errore è quello di rifornire il magazzino in maniera eccessiva a livello di **scorte**, per evitare **rotture di stock**, ossia trovarsi senza scorte. In **Kanban** questo spesso si traduce con un dimensionamento eccessivo delle colonne di buffer,

dimensionamento che smorza l'effetto oscillante e che quindi impedisce di avere una misurazione in tempo reale delle fluttuazioni del mercato. Le notizie arrivano in ritardo, le contromisure che si intraprendono potrebbero non essere più in linea con quello che succede sul campo.

Inefficacia delle previsioni di vendita

L'inefficacia frequente o costante delle **previsioni di vendita** è un altro esempio. In agilità questo aspetto è stato abbondantemente affrontato: fare previsioni è molto difficile, se non impossibile, per cui si preferisce fare delle sperimentazioni in un ambito protetto e con un coefficiente di rischio ridotto al minimo (p.e.: iterazioni brevi, piccoli lotti di lavorazione, e così via).

Limitare gli sbalzi della capacità produttiva

La capacità produttiva **non costante** amplifica le oscillazioni che arrivano dal committente. Per questo si dovrebbe sempre cercare di stabilizzare il più possibile il contesto di lavoro: limitare il turn over delle persone, non variare la lunghezza degli sprint e comunque tenere sprint piccoli, livellare il flusso di alimentazione del backlog.

Anche **frequenti cambiamenti ai piani di produzione** alimentano l'effetto frusta. In un progetto software, il piano di produzione potrebbe essere qualcosa che ricade nel contesto Product Ownership o, più in alto, essere parte della vision. In Agile il cambiamento è benvenuto e per questo le metodologie come Scrum o Kanban riescono a reagire prontamente a eventuali variazioni.

È noto però che un cambiamento "forte" ha implicazioni non banali sul processo di produzione. Per esempio è probabile che la velocity cambi, o che il trend delle cose da fare, ossia il Burn Down di Scrum, subisca una radicale perturbazione.

Cause indirette

Oltre alle cause viste fino a questo punto, ci sono alcuni aspetti che forse è più corretto chiamare **effetti** ma che condizionano il sistema stesso dando luogo a un qualcosa che rientra nella teoria dei **system dynamics**, circuiti caratterizzati da ciclicità e feedback rientrante.

Comportamenti irrazionali

In determinati momenti, gli attori del sistema mettono in atto comportamenti irrazionali, frutto dell'emotività, o che possono apparirlo a posteri, come nel caso dei **decisori con razionalità limitata**. Si opera in un contesto a **razionalità limitata** durante il processo decisionale quando la razionalità un individuo è **limitata** dalle **informazioni** che **possiede**, dai limiti cognitivi della sua mente, e dall'**ammontare finito** di **tempo** che egli ha a disposizione per prendere una decisione.

Svariati modelli economici danno per scontato che le persone abbiano una razionalità media, e possano in quantità sufficientemente grandi essere approssimati come

agenti in accordo alle loro preferenze. Il concetto di razionalità limitata rivede questo assunto per tenere conto del fatto che **decisioni perfettamente razionali spesso non sono realizzabili nella pratica,** proprio a causa della quantità finita di risorse "computazionali" disponibili per prenderle [RAZ].

Mancanza di coordinamento

Spesso manca un coordinamento nelle decisioni dei singoli membri della catena di produzione. Anche in questo caso la mancanza di coordinamento è spesso legata ad una razionalità limitata dei decisori, i quali sono quindi portati ad assumere un **comportamento localmente opportunistico**: ogni decisore pensa prima di tutto a ottimizzare **il proprio limitato** ambito operativo.

In questo caso, poiché i decisori mancano delle capacità e delle risorse per arrivare alla soluzione ottimale, essi applicano invece la loro razionalità solo dopo aver enormemente semplificato le scelte disponibili. In pratica, il decisore cerca una **soluzione soddisfacente piuttosto che la migliore** in assoluto [RAZ].

Conclusione

L'effetto Forrester impatta sul processo di produzione e vendita di un bene, cosa che nel campo IT potrebbe essere la messa in produzione o il rilascio di un determinato componente o prodotto software.

La tipica conseguenza in una catena di produzione e vendita è che tutti gli attori nel mezzo della catena sono portati a un più o meno cosciente aumento delle scorte di sicurezza: si cerca di evitare la **rottura di stock**. La formica che accumula scorte per l'inverno, infatti, non ha problemi di obsolescenza del prodotto, cambio di requisiti — mangia sempre le stesse cose — o richieste particolari.

In un progetto software, il **magazzino** in genere è rappresentato da tutto il software che non è ancora stato rilasciato e che rimane in attesa di essere quindi verificato e provato sul campo. Per questo, in un progetto agile si cerca di minimizzare questa forma di magazzino, proprio per limitare l'accumulo di parti di prodotto non testate e non provate sul campo.

In un progetto software, gli accumuli di magazzino non sono né un obiettivo rincorso, né tantomeno una forma di protezione (irrazionale), ma casomai manifestazione di scarsa organizzazione o mancanza di strumenti: per esempio non si fa **continuous integration**.

Si è visto con l'esempio della produzione della birra che a un certo momento il sistema nel suo complesso ha iniziato a funzionare in modo non ottimale, cosa resa evidente da **ritardo nelle consegne** ad ogni livello della organizzazione. Questo effetto si può avere anche in una organizzazione che produce software. Si pensi per esempio a un sistema di gestione dei bug di produzione gestito tramite Kanban. In questo caso la variabilità può essere molto alta e, contrariamente a quanto possano dire gli analisti di mercato che si lanciano in rocambolesche previsioni, molto poco prevedibile.

Questo è un tipico caso in cui ciclo di lavorazione lungo, scarsa visione di insieme, ottimizzazioni localizzate, scarso coinvolgimento del cliente finale possono amplificare l'onda dell'effetto frusta, dando come risultato più evidente un ritardo nella lavorazione dei ticket.

Riferimenti

[5TH] Peter Senge, La quinta disciplina: L'arte e la pratica dell'apprendimento organizzativo, Sperling & Kupfer, 2006

[LBG] Che cosa è il beer game?
http://maaw.info/TheBeerGame.htm

[RAZ] La voce "Razionalità limitata" su Wikipedia
https://it.wikipedia.org/wiki/Razionalit%C3%A0_limitata