SEGUIMIENTO Y COMUNICACIÓN

CONTROL DE AVANCE EN PROYECTOS ÁGILES

COMUNICACIÓN DEL AVANCE

FUENTES:

- COHN
- PMI

CONTROL DE AVANCE EN PROYECTOS ÁGILES

PMI

CONTROLAR EL CRONOGRAMA

- Determinar el estado actual del cronograma del proyecto mediante la comparación de la cantidad total de trabajo entregado y aceptado con respecto a las estimaciones de trabajo completado para el ciclo de tiempo transcurrido.
- Llevar a cabo revisiones retrospectivas (revisiones programadas para registrar las lecciones aprendidas) a fin de corregir y mejorar procesos si fuera necesario.
- Volver a priorizar el plan de trabajo pendiente.
- Determinar el ritmo al que se generan, validan y aceptan los entregables (velocidad) en el tiempo por iteración (duración acordada del ciclo de trabajo, normalmente 2 semanas o 1 mes).
- Determinar si el cronograma del proyecto ha cambiado.
- Gestionar los cambios reales conforme suceden.

CAP. 19
SEGUIMIENTO DEL PLAN DE LA *RELEASE*

- *Burn-down chart* de la *release*
- Gráfica de barras *burn-down* de la *release*
- Gráfica de lote de estacionamiento
- Gráfica de *throughput*
- Gráfica de velocidad

SEGUIMIENTO DE LA RELEASE

- Plan de la release:

  «En los próximos **4 meses** y **8 iteraciones** de **2 semanas** completaremos aprox. **240 puntos de historia** (o días ideales) de trabajo.»

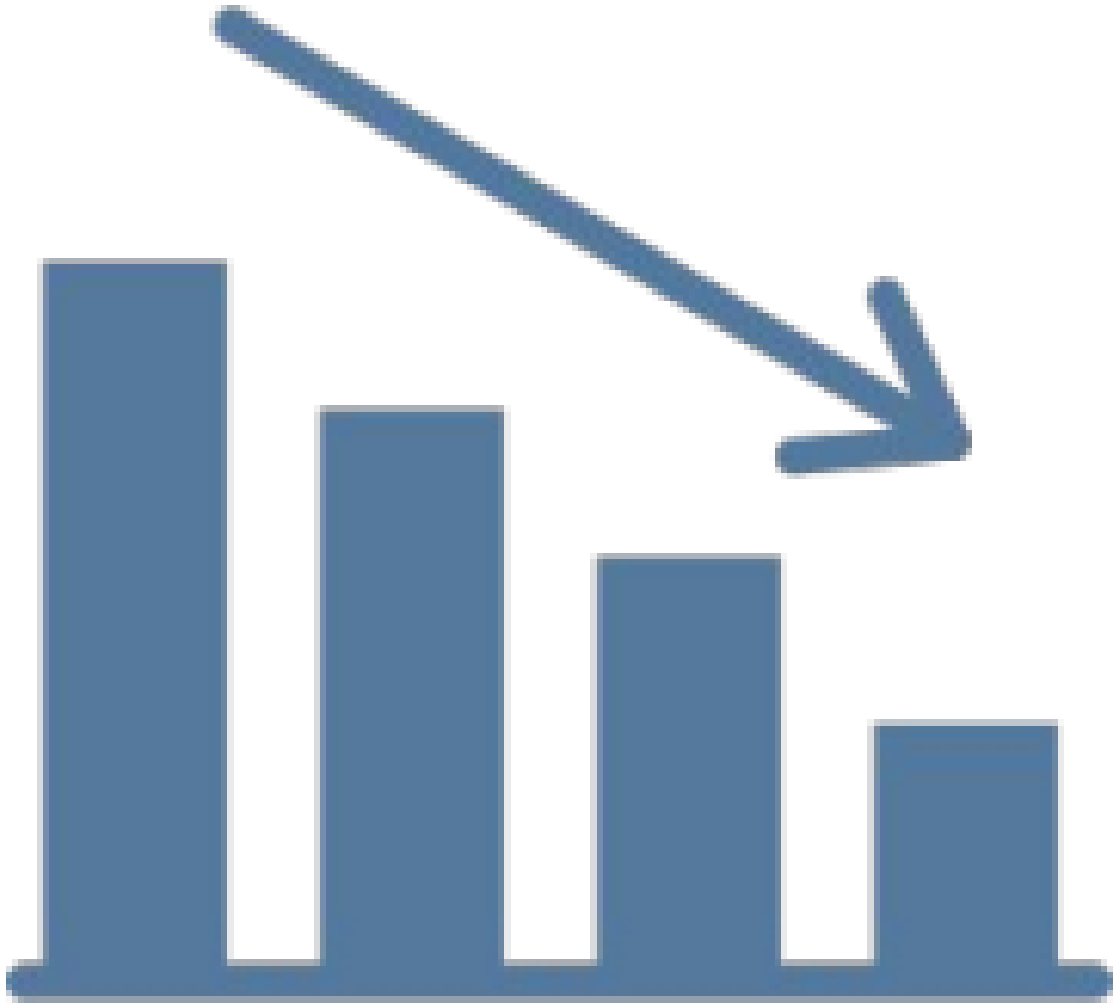
- Queremos evaluar dónde estamos:
  - Avance (trabajo completado)
  - Si hay cambios en el alcance
  - Si hay que reestimar

MEDIDA DE AVANCE: LA VELOCIDAD

- La velocidad mide la cantidad de trabajo completado en cada iteración.
- Completado:
  - Código bien escrito y bien factorizado
  - Chequeado
  - Limpio
  - Que cumple los estándares de codificación
  - Que pasó todas las pruebas.
- Se mide en puntos de historia o días ideales.
- Se calcula con regla todo o nada (0/100):
  - Si una historia está totalmente terminada, se suma todo el valor estimado.
  - Si está parcialmente completa, no se cuenta.

HISTORIAS INCOMPLETAS

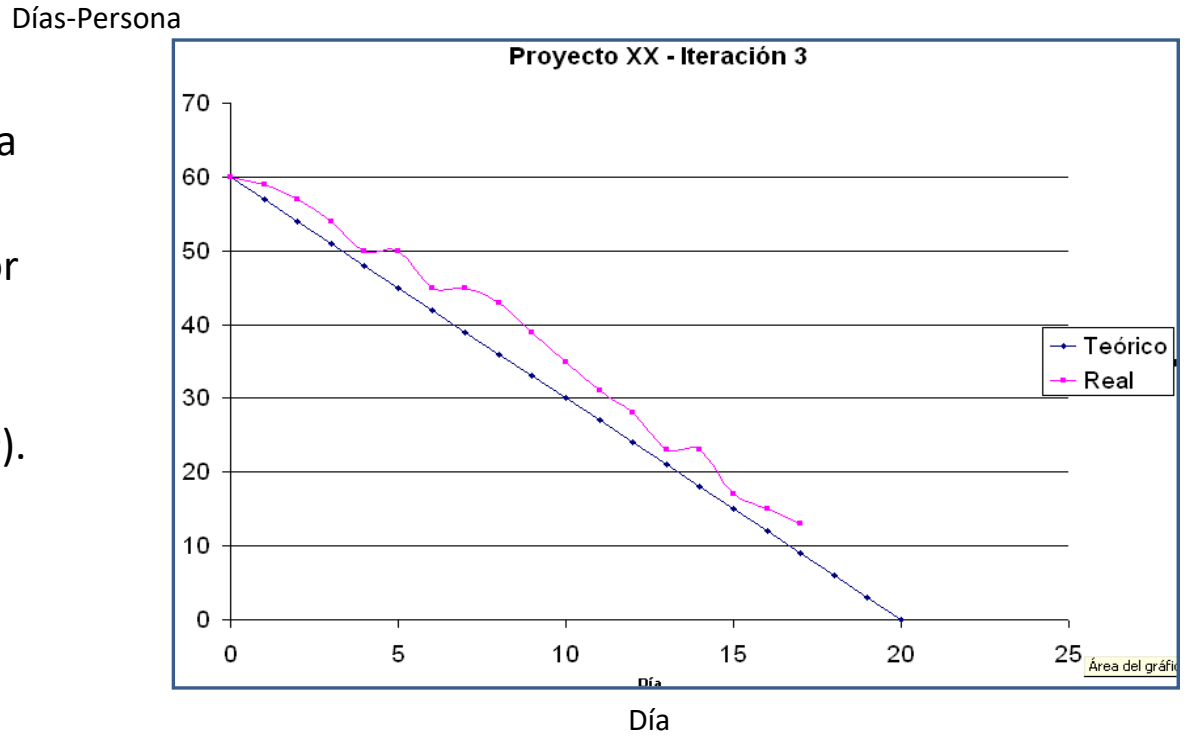
- Trabajo incompleto:
  1. difícil de medir.
  2. Rompe la confianza entre el cliente y el equipo de desarrollo.
     - Sacarla de la iteración o partirla y sacar parte.
     - Cambiar criterios de aceptación
  3. Aumenta el trabajo en curso →
     - disminuye el tiempo de entrega
     - Retrasa la retroalimentación
- ¿Por qué?
  - Estimación por debajo de lo real
  - Pusimos demasiadas historias en la iteración



BURN-DOWN CHART
GRÁFICA DEL TRABAJO PENDIENTE

BURN-DOWN CHART

- Muestra la evolución del trabajo que queda pendiente a lo largo del tiempo.
- En términos de EVG corresponde al valor que queda por ganar.
- Objetivo: facilitar el seguimiento del trabajo que queda por realizar en un periodo concreto (iteración o *release*).
- En los equipos de desarrollo ágiles, se suele aplicar
  - por iteración: registra el trabajo (o esfuerzo) pendiente de forma diaria.
  - por *release* (o para todo el producto): registra el trabajo (o esfuerzo) pendiente iteración a iteración.
- Ayuda al equipo a visualizar el trabajo o esfuerzo realizado y el que queda por realizar hasta la finalización de la iteración (o la *release*).
- Se pueden usar distintas unidades: historias de usuario, puntos de historia, tiempo ideal, horas-hombre, etc.

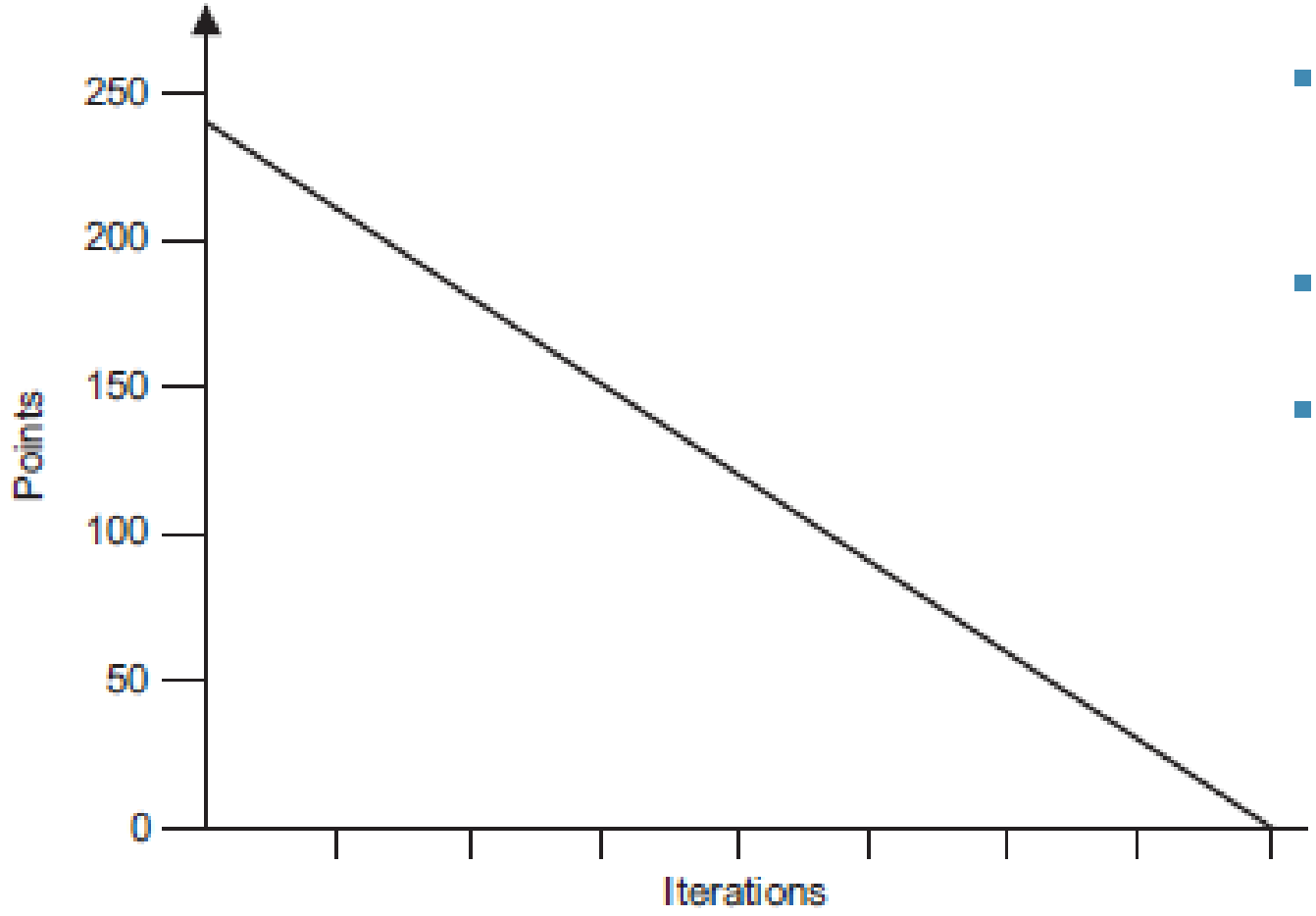


BURN-DOWN CHART

GRÁFICA *BURN-DOWN* DE LA *RELEASE*

- Cifras importantes:
  - Trabajo pendiente en la release
  - Tasa de avance del grupo (= velocidad)

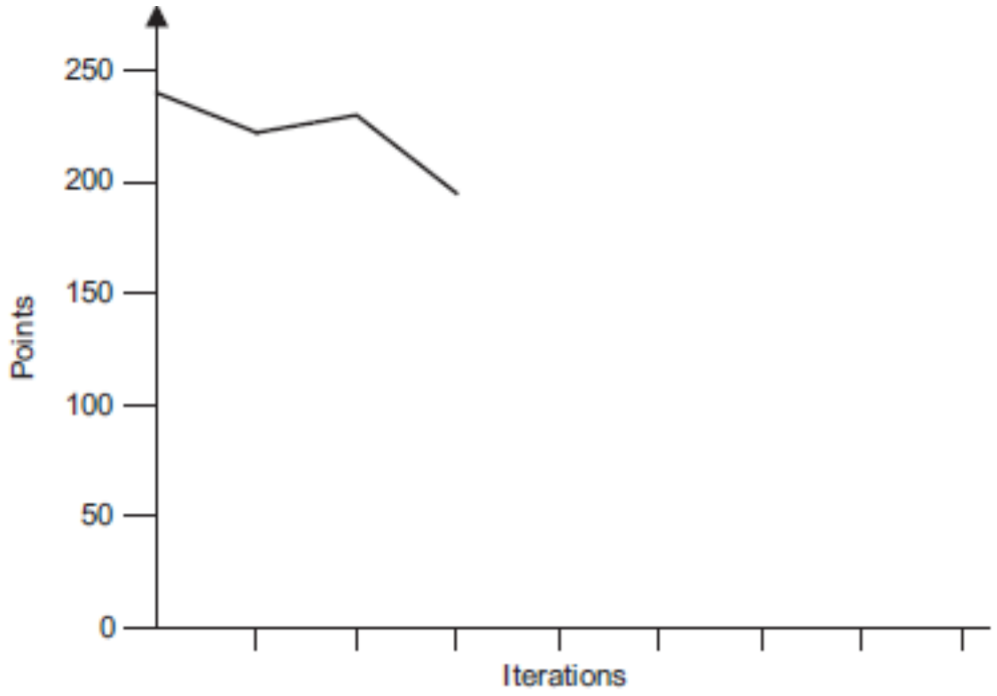
GRÁFICA *BURN-DOWN* DE LA *RELEASE* PLANIFICADA



- Total: 240 puntos de historia
- 8 iteraciones
- VE = 30 P. H.

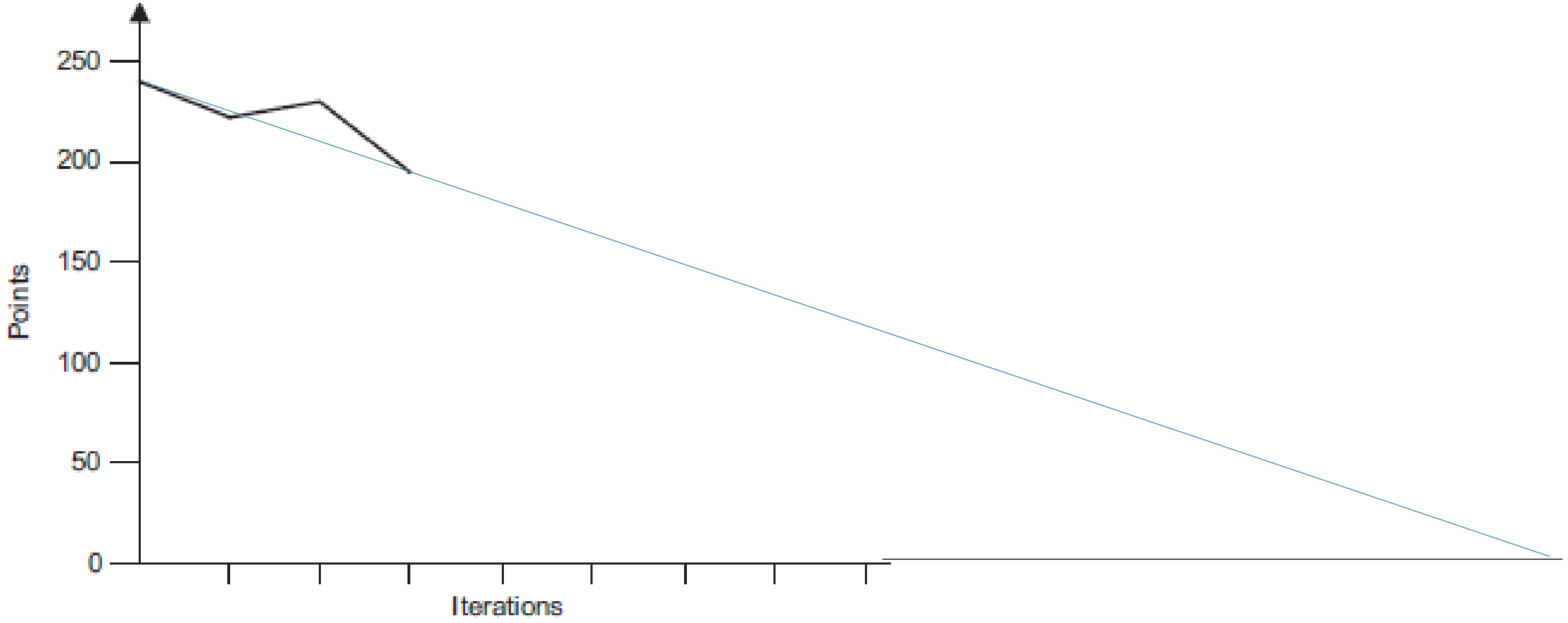
GRÁFICA *BURN-DOWN* DE LA *RELEASE* LO REALIZADO

- Nunca es una línea recta.
- Varía por:
  - Malas estimaciones
  - Cambios en las estimaciones
  - Cambios en el alcance



  - Puede mostrar un *burn-up* durante la iteración. Esto significa:
    - O aumentaron las estimaciones
    - O aumentó el alcance

PREDICCIÓN

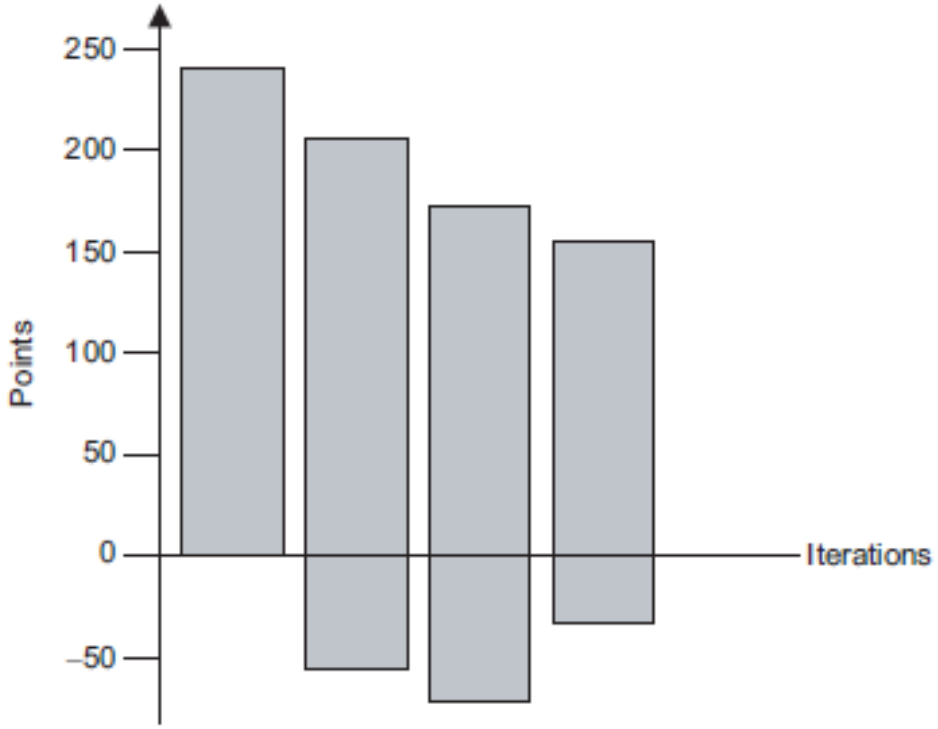


- Va a llevar más tiempo que el previsto.
- Nos indica dónde terminará si los factores no cambian.

GRÁFICA DE BARRAS *BURN-DOWN* DE LA *RELEASE*

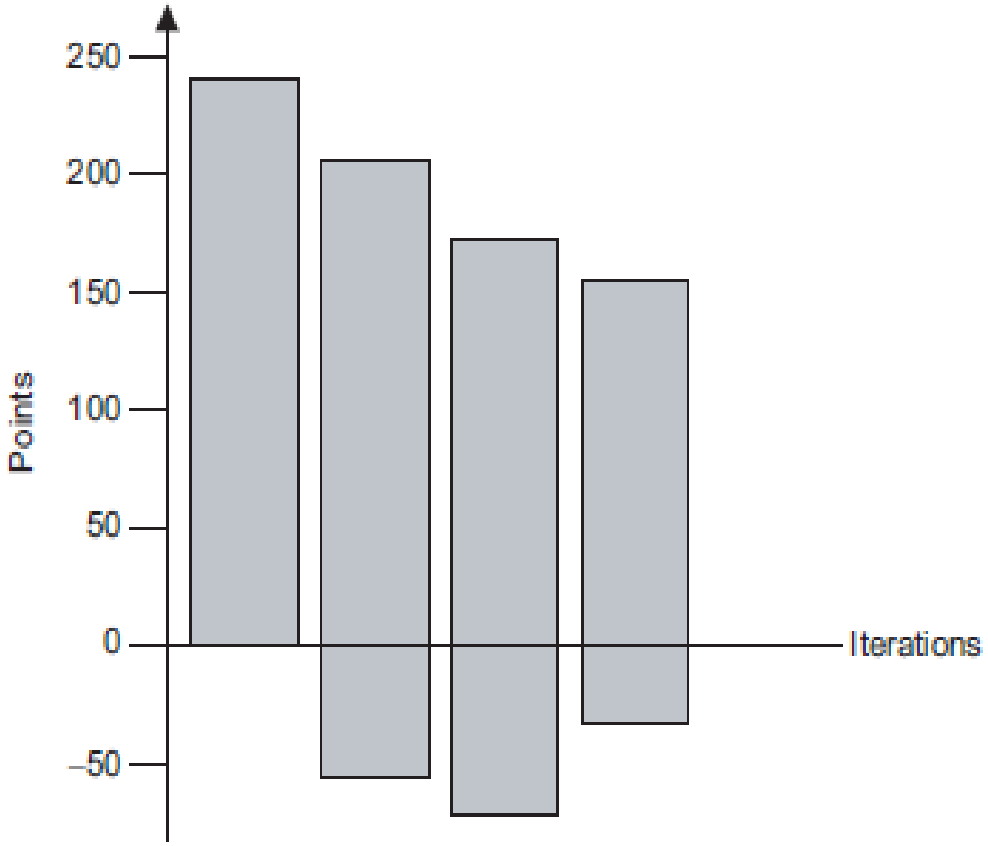
- Separa el avance de los cambios en el alcance de la *release*.

GRÁFICA DE BARRAS *BURN-DOWN*



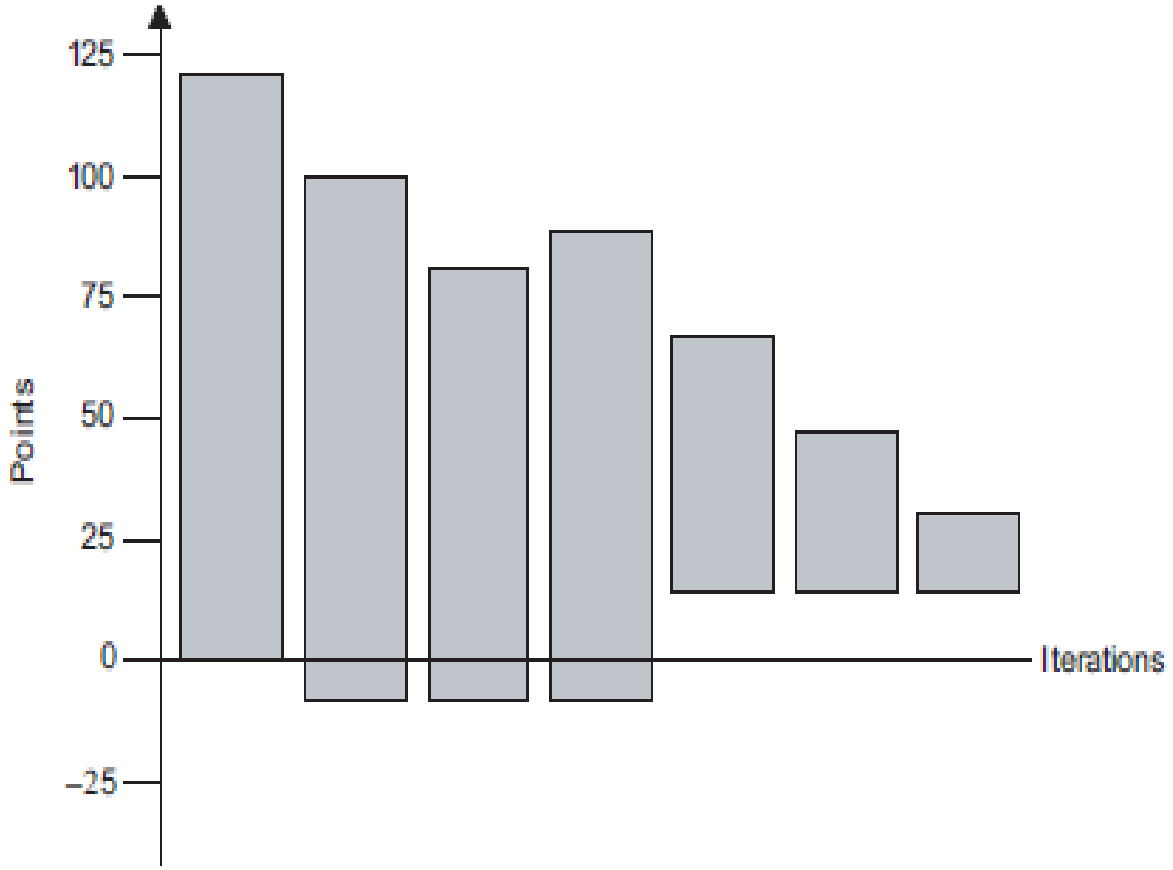
- Los cambios en el alcance se muestran bajando la barra por debajo del eje de las x.
- Cuatro reglas:
  - Se baja el tope superior cuando se complete trabajo.
  - Se baja o sube el tope cuando se reestima.
  - Se baja el fondo cuando se agrega alcance.
  - Se sube el fondo cuando se disminuye el alcance.

EJEMPLO - INTERPRETACIONES



1. La velocidad del equipo es la estimada. Se ve en el tope de las dos primeras barras.
2. Se agregó mucho trabajo. Se ve en el fondo de la 2.a barra.
3. Se ha agregado más trabajo del que ha sido completado. Dependerá de si luego la tendencia se confirma, o de cuán importante sea la fecha inicial de la *release*.
4. La cantidad total de trabajo pendiente en la *release* es mayor que cuando el proyecto empezó, porque el largo de la barra 2 es mayor que el de la barra 1.
5. En la 3.ª iteración se agregó alcance, pero menos que la vez anterior.
6. En la 3.ª iteración la velocidad bajó de 30 a 20, por
   - Se subestimaron algunas historias de las hechas
   - Un integrante enfermo o de licencia
   - Se reestimaron historias del alcance pendiente.

EJERCICIO

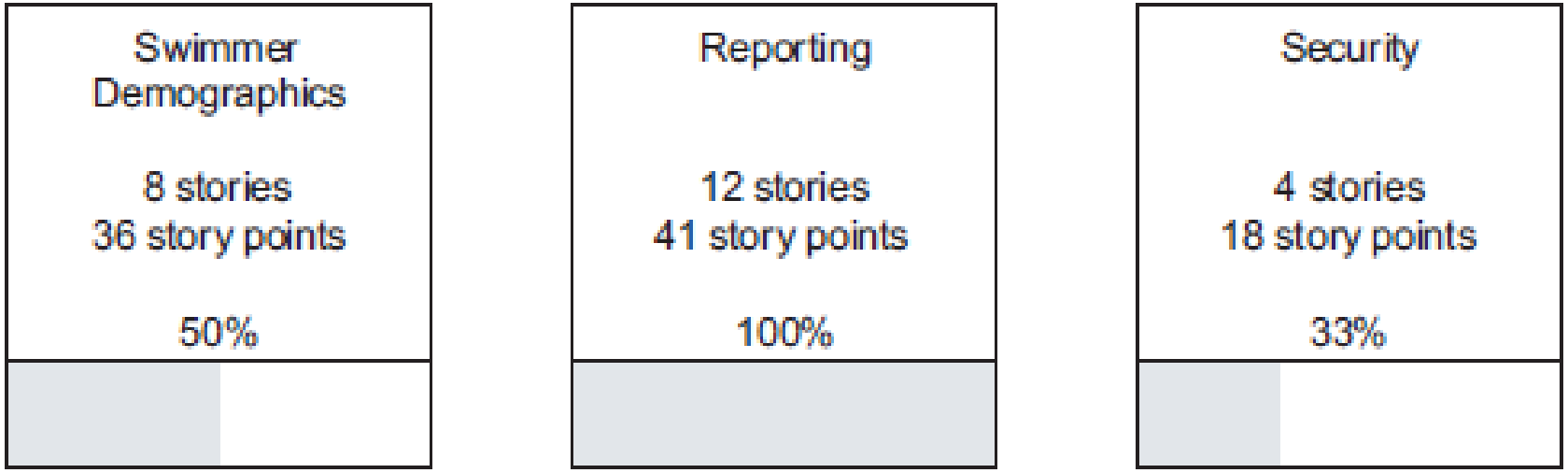


- ¿Velocidad en cada iteración?
- ¿Agregó el *product owner* trabajo? ¿En qué iteración?
- ¿Qué pasó durante la tercera iteración?
- ¿Redujo el *product owner* el alcance de la *release*? ¿Cuándo?

GRÁFICA DE LOTE DE ESTACIONAMIENTO

GRÁFICA DE LOTE DE ESTACIONAMIENTO

- Colocar una caja rectangular para cada tema de la *release*, con:
  - El nombre del tema
  - La cantidad de historias del tema
  - La cantidad de puntos de historia o días ideales para esas historias
  - El porcentaje de puntos de historia completas.



GRÁFICA DE LOTE DE ESTACIONAMIENTO

- No podemos decir cuáles historias están completas.
- Las cajas se pueden colorear para indicar si un tema
  - está completo,
  - está en tiempo,
  - necesita atención, o
  - está significativamente atrasado.
- Es útil para mostrar una vista de alto nivel del avance del equipo en la implementación de los temas planificados de la release.
- Se puede mostrar en una sola página.

THROUGHPUT CHART

THROUGHPUT CHART

- Indica la cantidad de elementos de trabajo realizados (no puntos de esfuerzo) en un periodo concreto.
- Muestra su evolución con la cadencia que interese (cada día, semana, mes, …).
- Esto permite a un equipo comprometido con la mejora continua visualizar sus avances en términos de productividad y predecir las fechas de finalización de sus elementos de trabajo.
- Para crearlo:
  1. Determinar la cadencia a usar.
  2. Contabilizar los elementos de trabajo finalizados en dicho periodo.

THROUGHPUT CHART

- Para predecir las fechas de finalización de un determinado trabajo se debe:
  1. Dividir el trabajo en n elementos de trabajo homogéneos.
  2. Determinar el *throughput* medio, una vez el equipo sea estable.
  3. Calcular el tiempo que llevará completar el trabajo aplicando la ley de Little:

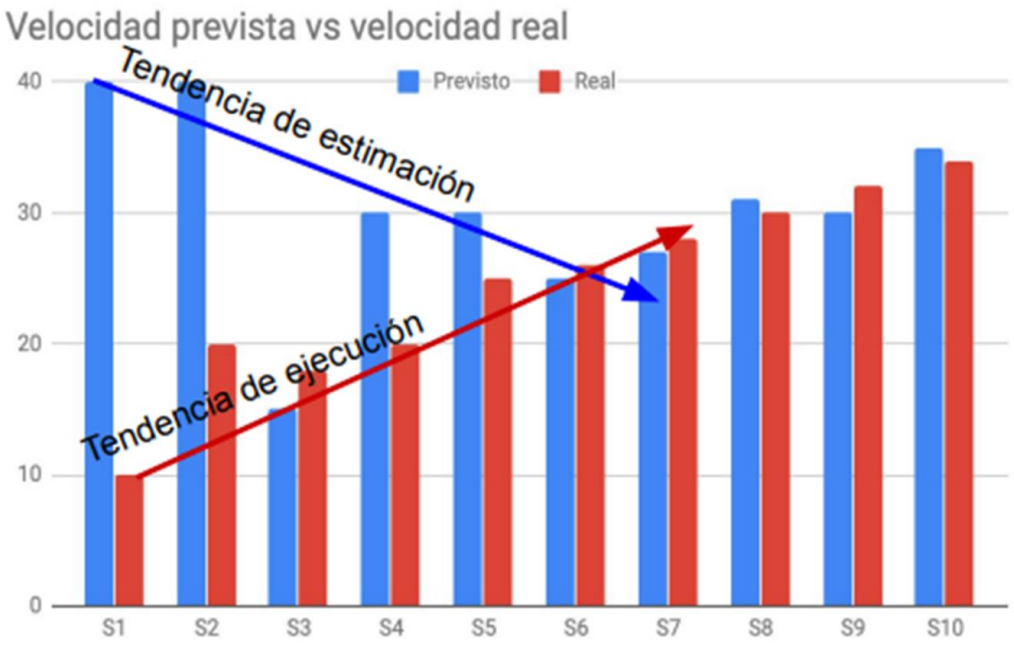
     tiempo = cant. de elementos de trabajo / *throughput*.
  4. Sumar el tiempo obtenido a la fecha de inicio del trabajo.
- Para que el *throughput* sea una métrica consistente y precisa, el tamaño medio de los elementos de trabajo debe ser homogéneo.

- Aunque el *throughput* debería mostrar la mejora del equipo, tiene un umbral de eficiencia. En el ejemplo gráfico, el equipo se estabiliza con un *throughput* de 120 elementos de trabajo realizados cada 15 días.

GRÁFICA DE VELOCIDAD

GRÁFICA DE VELOCIDAD

- Permite visualizar la evolución de la velocidad del equipo a lo largo de múltiples iteraciones.
- Muestra (para cada iteración) los puntos de esfuerzo estimados del equipo, así como los realmente alcanzados al finalizar cada iteración.
- La velocidad es una métrica que no se puede utilizar para comparar el rendimiento de equipos diferentes. Se debe limitar a la evolución del propio equipo.
- Dos datos por cada iteración:
  1. Estimación de esfuerzo total a realizar por el equipo, obtenido al planificar el trabajo al comienzo de la iteración (en la planificación del *sprint* para un equipo Scrum).
  2. Esfuerzo total del trabajo realizado durante la iteración, obtenido al final de la iteración (en la revisión del *sprint* para un equipo Scrum).



- Las divergencias entre lo estimado y lo realizado son más acentuadas en las primeras iteraciones, especialmente en equipos nuevos con inmadurez tecnológica o metodológica.
- Al avanzar las iteraciones, las divergencias deberían reducirse.
- Un equipo maduro en mejora continua se quedará unas veces por encima y otras por debajo de la estimación, pero siempre cerca de ella. Además, mostrará una tendencia creciente hasta llegar a su umbral de eficiencia.

CAP. 20
SEGUIMIENTO DEL PLAN DE LA ITERACIÓN

- El panel de tareas
- Gráfica *burn-down* de la iteración
- Gráfica de *burn-up*
- Diagrama de flujo acumulativo

EL PANEL DE TAREAS

- Es un pizarrón blanco, o un panel de corcho o un espacio en en la pared.
- Ayuda al equipo a organizar su trabajo y a visualizar el trabajo pendiente.
- Las columnas están etiquetadas.
- Los miembros mueven las tarjetas de tareas entre las columnas a medida que el trabajo avanza.

EL PANEL DE TAREAS

- Los integrantes no se autoasignan o son asignados trabajo hasta que estén prontos para hacerlo.
- El panel de tareas permite que todos vean en qué tareas se está trabajando y cuáles están disponibles para tomar.

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the... 8 Code the... 5 Test the... 6	√	Code the... SC 6 Code the... DC 4	Code the... LC 4	33
As a user, I can… 2	Code the... 8 Code the... 5				13
As a user, I can… 3	Code the... 3 Code the... 6	√	Code the... MC 4		13

FILAS

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the… 8 Code the… 5 Test the… 6	√	Code the… SC 6 Code the… DC 4	Code the… LC 4	33
As a user, I can… 2	Code the… 8 Code the… 5				13
As a user, I can… 3	Code the… 3 Code the… 6	√	Code the… MC 4		13

- Se colocan las tarjetas de historia y las tarjetas de tareas.
- Una fila por cada historia.

COLUMNAS

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the... 8 Code the... 5 Test the... 6	√	Code the... SC 6 Code the... DC 4	Code the... LC 4	33
As a user, I can… 2	Code the... 8 Code the... 5				13
As a user, I can… 3	Code the... 3 Code the... 6	√	Code the... MC 4		13

1. Historia
   - Contiene la tarjeta de historia.
   - Y la cantidad de puntos de historia.

COLUMNAS

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the... 8 Code the... 5 Test the... 6	√	Code the... SC 6 Code the... DC 4	Code the... LC 4	33
As a user, I can… 2	Code the... 8 Code the... 5				13
As a user, I can… 3	Code the... 3 Code the... 6	√	Code the... MC 4		13

2. Para hacer
   - Contiene todas las tarjetas de tarea para implementar la historia.
   - Cada una con su estimación.
   - Ancho de la columna puede ser más grande que una tarjeta.

COLUMNAS

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the... 8 Code the... 5 Test the... 6	√	Code the... SC 6 Code the... DC 4	Code the... LC 4	33
As a user, I can… 2	Code the... 8 Code the... 5				13
As a user, I can… 3	Code the... 3 Code the... 6	√	Code the... MC 4		13

3. Pruebas especificadas

- Indica si las pruebas de aceptación están prontas para esa historia.
- *Test-driven development*:
  - Escribir una prueba de falla antes de codificar.
  - Diseñar pruebas de aceptación de alto nivel para el feature antes de codificar.
  - Las condiciones de satisfacción de cada historia (identificadas en la planificación de la iteración) son pruebas de aceptación de la historia de alto nivel.
- La columna se chequea si las pruebas de aceptación de alto nivel de la historia están especificadas.
- No mover tarjetas a la columna *En curso*, a menos que las pruebas estén especificadas.

COLUMNAS

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the... 8 Code the... 5 Test the... 6	√	Code the... SC 6 Code the... DC 4	Code the... LC 4	33
As a user, I can… 2	Code the... 8 Code the... 5				13
As a user, I can… 3	Code the... 3 Code the... 6	√	Code the... MC 4		13

4. En curso
   - Contiene las tarjetas que han sido tomadas. Cada desarrollador toma una tarjeta de la columna *Para hacer*, pone sus iniciales y la mueve a la columna *En curso*.
   - A medida que va quedando libre.
   - No tomar más de una tarjeta al mismo tiempo. Eso ayuda a mantener un flujo de trabajo consistente y reduce el costo de cambiar de contexto entre múltiples tareas. → ancho de la columna no mayor a una tarjeta.

COLUMNAS

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the... 8 Code the... 5 Test the... 6	√	Code the... SC 6 Code the... DC 4	Code the... LC 4	33
As a user, I can… 2	Code the... 8 Code the... 5				13
As a user, I can… 3	Code the... 3 Code the... 6	√	Code the... MC 4		13

5. Para verificar
   - Típicamente hay una tarjeta para la tarea de verificar cada historia.
   - Se puede tomar cuando la tarea de codificar la historia llega a la columna *Done*.
   - Pero, si no hay tarjeta o si el implementador quiere que alguien la revise, la coloca en esta columna.

COLUMNAS

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the... 8 Code the... 5 Test the... 6	√	Code the... SC 6 Code the... DC 4	Code the... LC 4	33
As a user, I can… 2	Code the... 8 Code the... 5				13
As a user, I can… 3	Code the... 3 Code the... 6	√	Code the... MC 4		13

6. Terminada/Done
   - Cuando se terminó la tarea.

COLUMNAS

Story	To Do	Tests Ready	In Process	To Verify	Hours
As a user, I can… 5	Code the... 8 Code the... 5 Test the... 6	√	Code the... SC 6 Code the... DC 4	Code the... LC 4	33
As a user, I can… 2	Code the... 8 Code the... 5				13
As a user, I can… 3	Code the... 3 Code the... 6	√	Code the... MC 4		13

7. Horas

- Suma de las horas de trabajo restantes para la historia.
- Los desarrolladores pueden en cualquier momento
  - cambiar la estimación (tachar y colocar una nueva) de cualquier tarea
  - romper una tarjeta de tarea y reemplazarla por otras dos o tres, cada una con su propia estimación
- Cada mañana sumar las horas de cada columna y usarlas para la gráfica *burn-down* de la iteración.

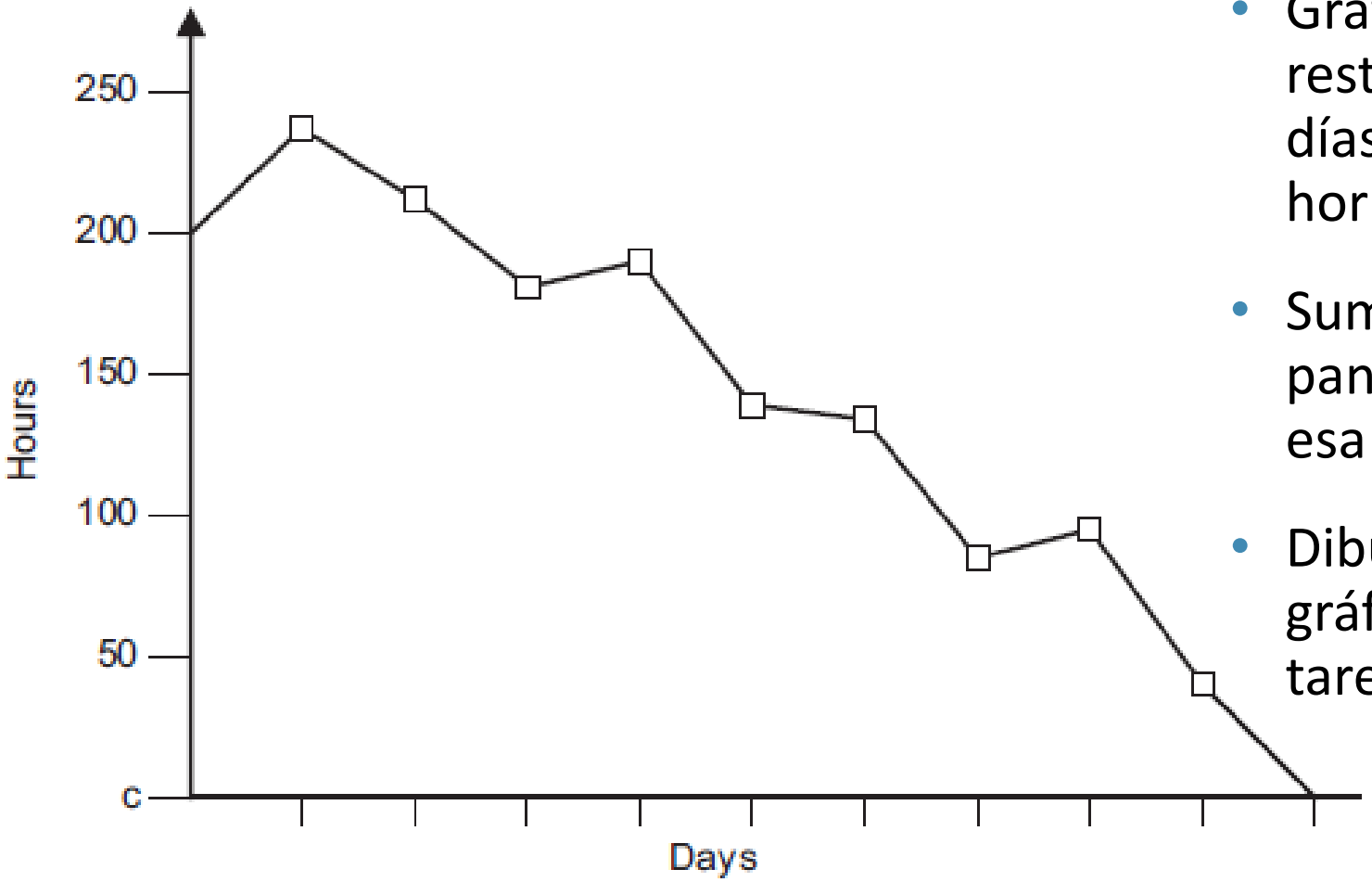
SEGUIMIENTO DE *BUGS* EN UN PANEL DE TAREAS

- Al comienzo de la iteración:
  - Se seleccionan los *bugs* a corregir.
  - Escribir una tarjeta de tarea para cada *bug* con su estimación.
  - Se colocan todas en una única fila.
  - El *product owner* decide qué porcentaje de la iteración se dedica a corrección de *bugs*.
- Defectos descubiertos en la iteración de alta prioridad:
  - Se estiman
  - Se agregan al tablero
  - Se quita de la columna *To do* una cantidad de trabajo equivalente.

GRÁFICA *BURN-DOWN* DE LA ITERACIÓN

- Es similar a la gráfica *burn-down* de la *release*, pero solo hace el seguimiento del trabajo en la iteración en curso.
- Muestra la cantidad de puntos de historia o de días ideales que faltan en el proyecto al comienzo de cada iteración.
- Grafica la cantidad de PH o DI restantes por día.

- Si iteraciones de una semana:
  - No es útil: para cuando se vean las tendencias, la iteración ya habrá terminado.
- Iteraciones de 2 semanas o más:
  - Es útil.



- Grafica cantidad de horas restantes en el eje vertical y los días de la iteración en el eje horizontal.
- Sumar todas las horas en el panel una vez al día y colocar esa cifra en la gráfica.
- Dibujar o agregar en papel la gráfica diariamente al panel de tareas.

SEGUIMIENTO DEL ESFUERZO REALIZADO

- Es más útil saber cuánto falta que cuánto se hizo.
- Saber cuánto se dedicó no siempre ayuda a mejorar la estimación.

VELOCIDAD INDIVIDUAL

- Es la cantidad de puntos de historia o días ideales completados por un integrante.
- No hacer seguimiento de la velocidad individual, sino de la del equipo.
- Seguir la velocidad individual no incentiva el trabajo en equipo.

BURN-DOWN CHART
EJERCICIO 1

		% de avance por historia al finalizar el día									
	# puntos	Día 1	Día 2	Día 3	Día 4	Día 5	Día 6	Día 7	Día 8	Día 9	Día 10
Historia 1	2	100	100	100	100						
Historia 2	3	50	50	75	75						
Historia 3	8	25	75	75	100						
Historia 4	3	0	25	50	100						
Historia 5	5	0	0	25	25						
Historia 6	2	0	0	0	100						
Historia 7	2	0	0	0	0						
Historia 8	5	0	0	0	0						
Historia 9	5	0	0	0	0						
Historia 10	2	100	100	100	100						
Historia 11	3	0	0	0	0						

Dado el siguiente avance de las historias de usuario durante el *sprint*:

a. Construya la línea base de planificación de la gráfica de *burn-down* del sprint.

b. Construya la *burn-down chart* del sprint al finalizar el día 4 del sprint.

c. Si a partir del día 5 el equipo mantiene el ritmo de productividad promedio que ha tenido hasta ese momento, ¿cuántos puntos habrá obtenido al finalizar el sprint?

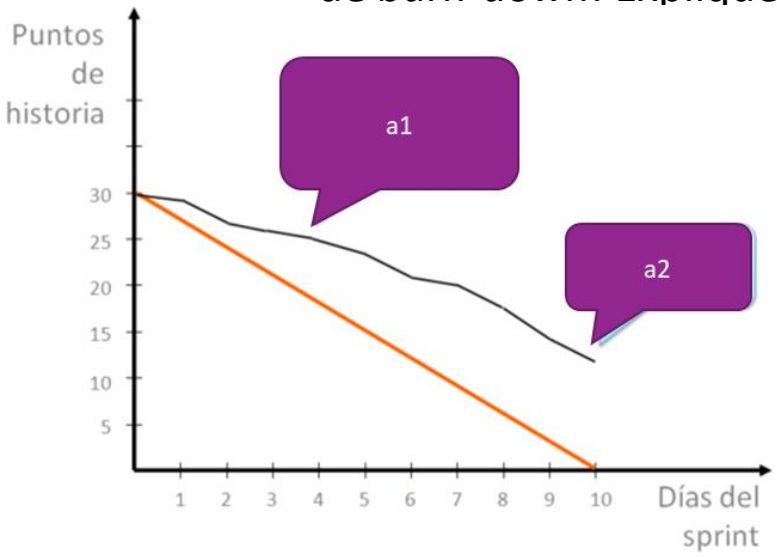
d. Si a partir del día 5 el equipo mantiene el mismo ritmo de productividad que en el último tramo, ¿cuántos puntos habrá obtenido al finalizar el sprint?

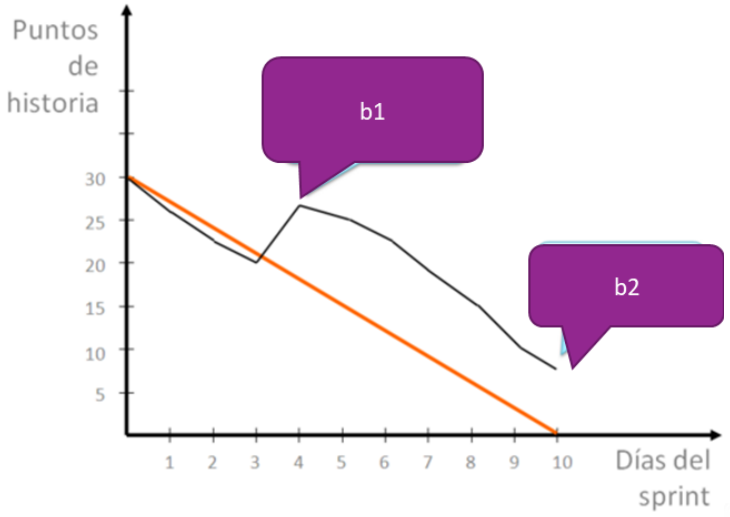
e. ¿Cuál de las dos opciones parece razonable en este caso? ¿Por qué?

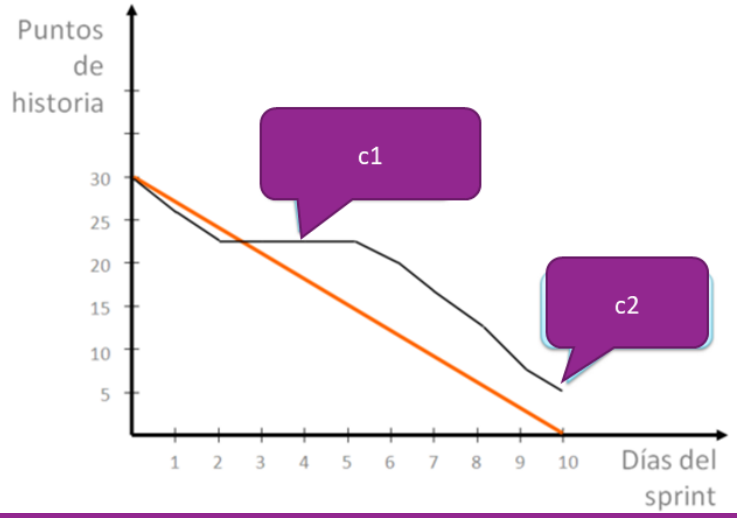
f. Reflexione el sentido de usar un diagrama de trabajo pendiente para hacer el seguimiento del avance en un *sprint* y para hacer el seguimiento del avance en una *release*. ¿Qué velocidad tomarían en cada caso?

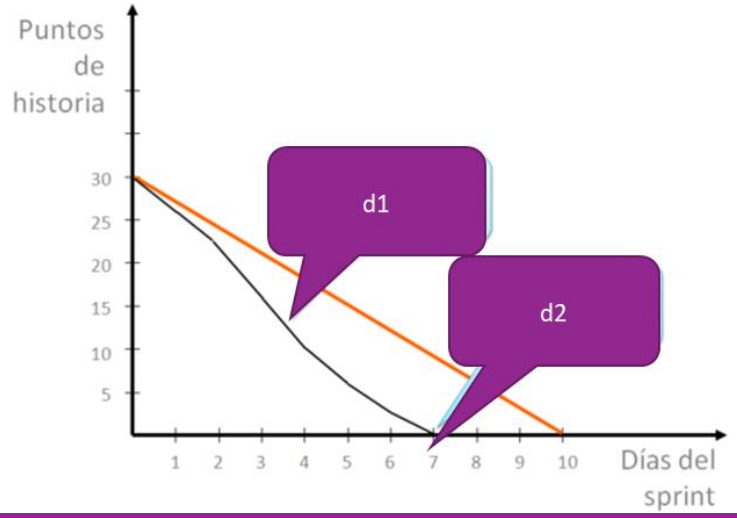
BURN-DOWN CHART
EJERCICIO 2

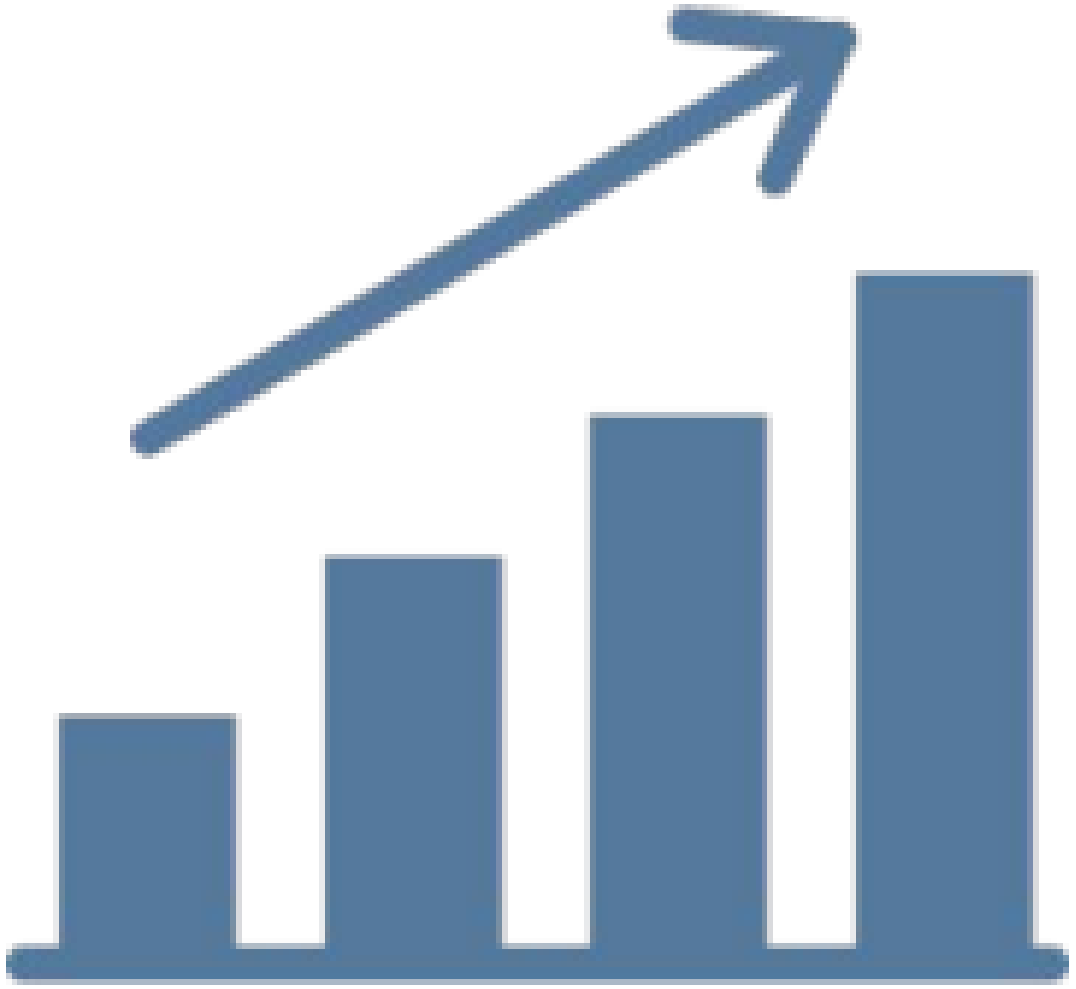
- Analice qué significan respecto al avance y a la compleción del *sprint* las siguientes gráficas de *burn-down*. Explique las situaciones señaladas:







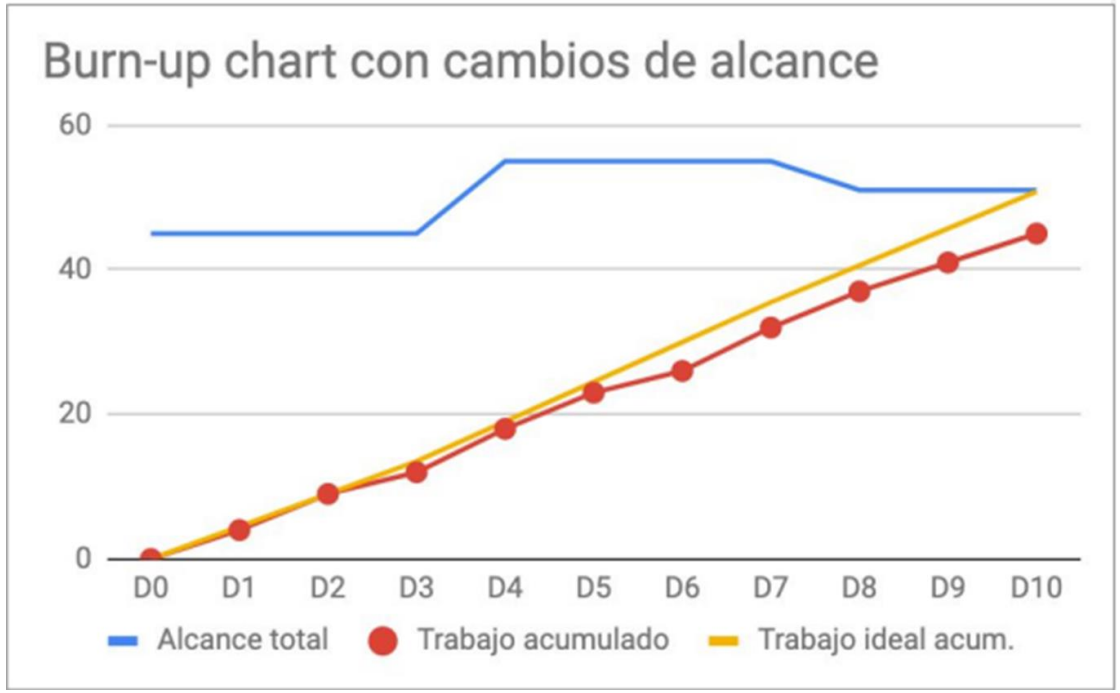


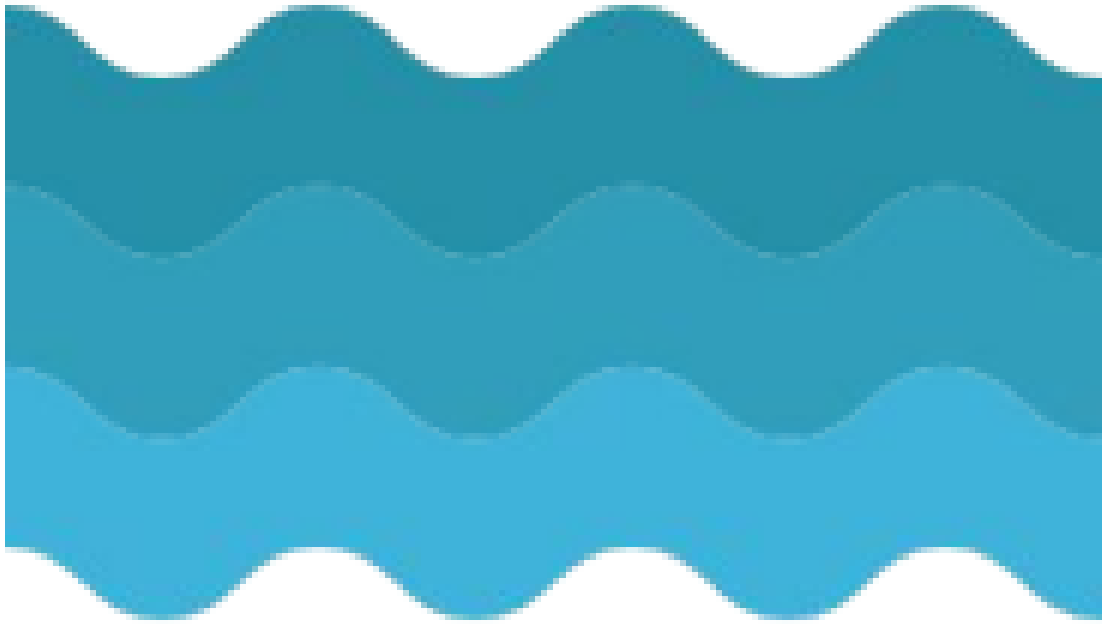


BURN-UP CHART

BURN-UP CHART

- Objetivo: facilitar el seguimiento del trabajo (esfuerzo) realizado en un periodo concreto (iteración).
- Muestra por separado el avance del trabajo realizado y el alcance (trabajo total a realizar).
- Permite ver claramente los cambios de alcance en la iteración.
- Al igual que el *burn-down chart*, se suele utilizar a nivel de iteración o *release*.





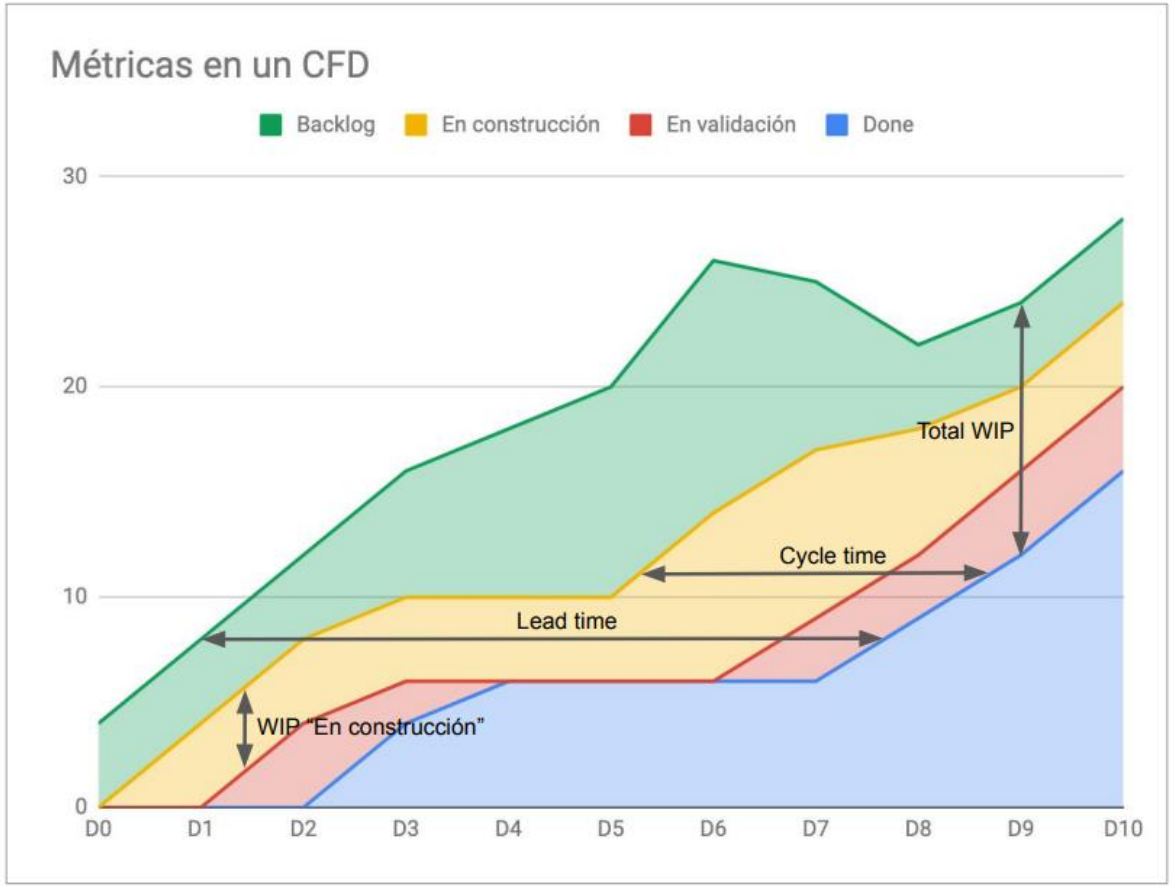
CUMULATIVE FLOW DIAGRAM (CFD)
DIAGRAMA DE FLUJO ACUMULATIVO

DIAGRAMA DE FLUJO ACUMULATIVO

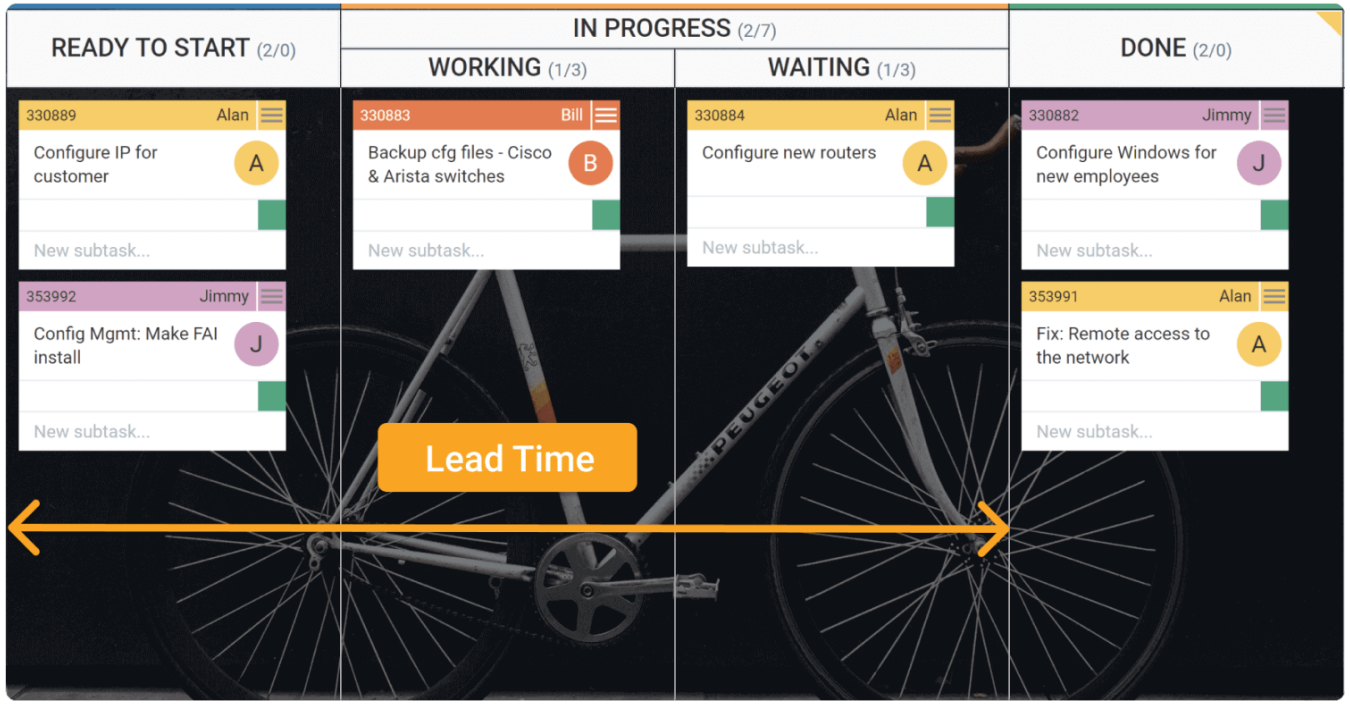
- Muestra la evolución en el tiempo del flujo de trabajo de un equipo.
- Refleja la cantidad de trabajo en curso (WIP) en cada estado, acumulando o apilando estas cantidades gráficamente, en periodos regulares.
- En el eje X se representa el tiempo (semanas, días, etc.)
- En el eje Y se representa el número de elementos de trabajo en cada estado del flujo de trabajo del equipo. Estos se apilan en orden inverso respecto al flujo. Alternativamente, se podrían reflejar puntos de esfuerzo.
- Para construir este gráfico es necesario disponer de la cantidad de elementos de trabajo que hay en cada estado cada día:
  1. Se comienza por el último estado del flujo (Done) y se marca el número de elementos.
  2. Se toma el anterior estado del flujo (En validación) y se marca el número de elementos apilándolo.
  3. Se continúa realizando el paso 2 (En construcción) hasta llegar al estado inicial del flujo (Backlog). Finalmente, se unen los puntos de cada estado y se colorean las distintas áreas resultantes.
  4. Cuando las bandas del CFD varían suavemente y su altura es estable, es síntoma de un buen equilibrio entre demanda y capacidad.

MÉTRICAS EN UN CFD

- ***Cycle time* (tiempo de ciclo)**: es el tiempo que pasa desde que se inicia un trabajo hasta que se entrega. Para el ejemplo: desde que entra en el estado *En construcción* hasta que llega a *Done*.
- ***Lead time* (tiempo de entrega):** es el tiempo que pasa desde que se solicita un trabajo (entra en el *backlog*) hasta que se entrega (*Done*).
- **WIP (*work in progress*) (trabajo en curso):** es la cantidad de elementos en curso para uno o varios estados del flujo de trabajo.
- ***Burn-up***: la evolución del último estado del flujo (*Done*) muestra cómo se va completando el trabajo, al igual que en un *burn-up chart*.

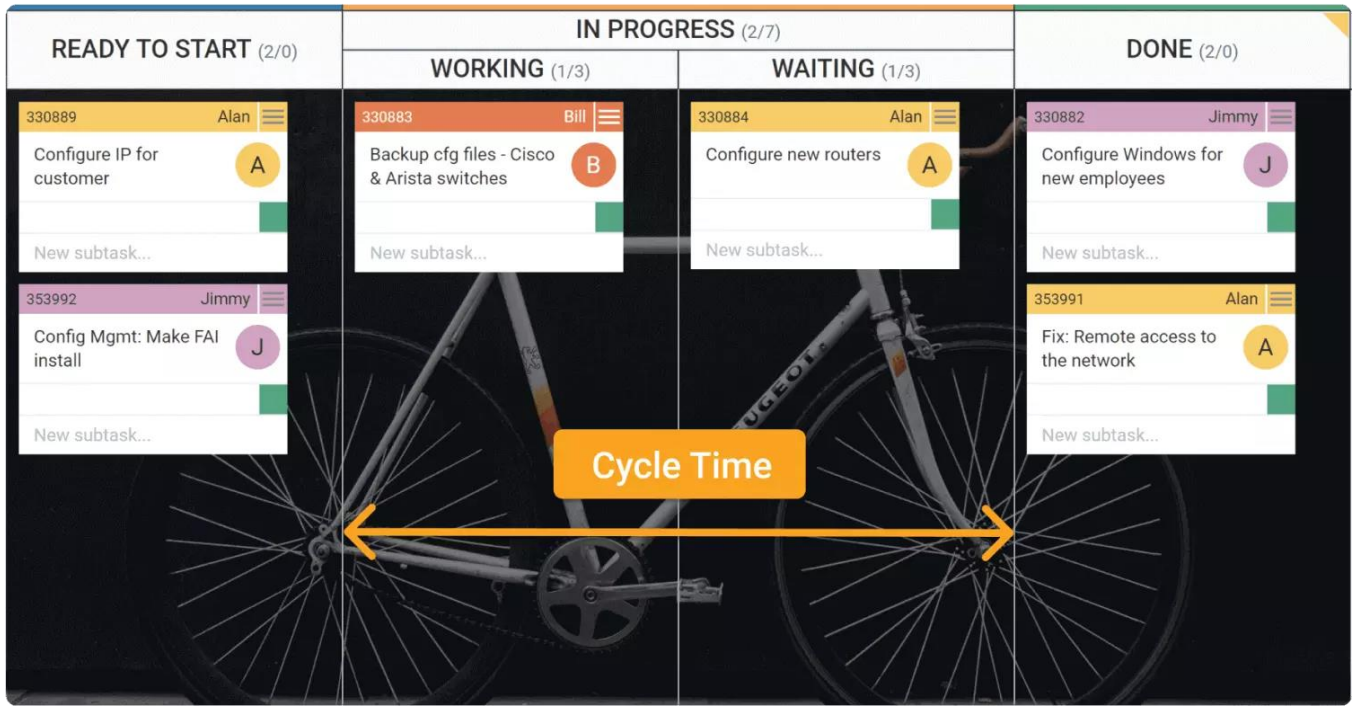


TIEMPO DE ENTREGA

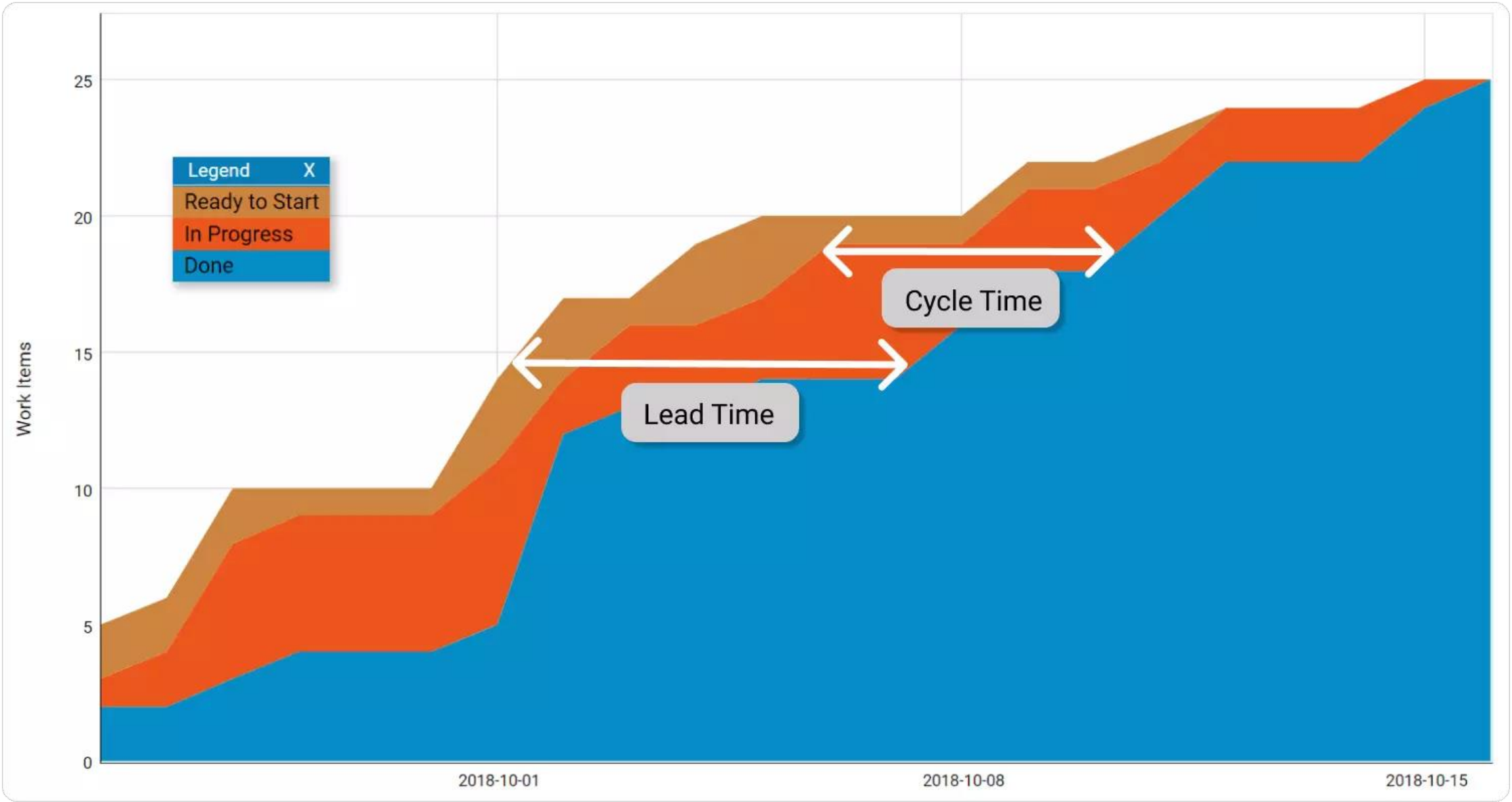


- El tiempo de entrega es el período entre la aparición de una nueva tarea en tu flujo de trabajo y su salida final del sistema.

TIEMPO DE CICLO

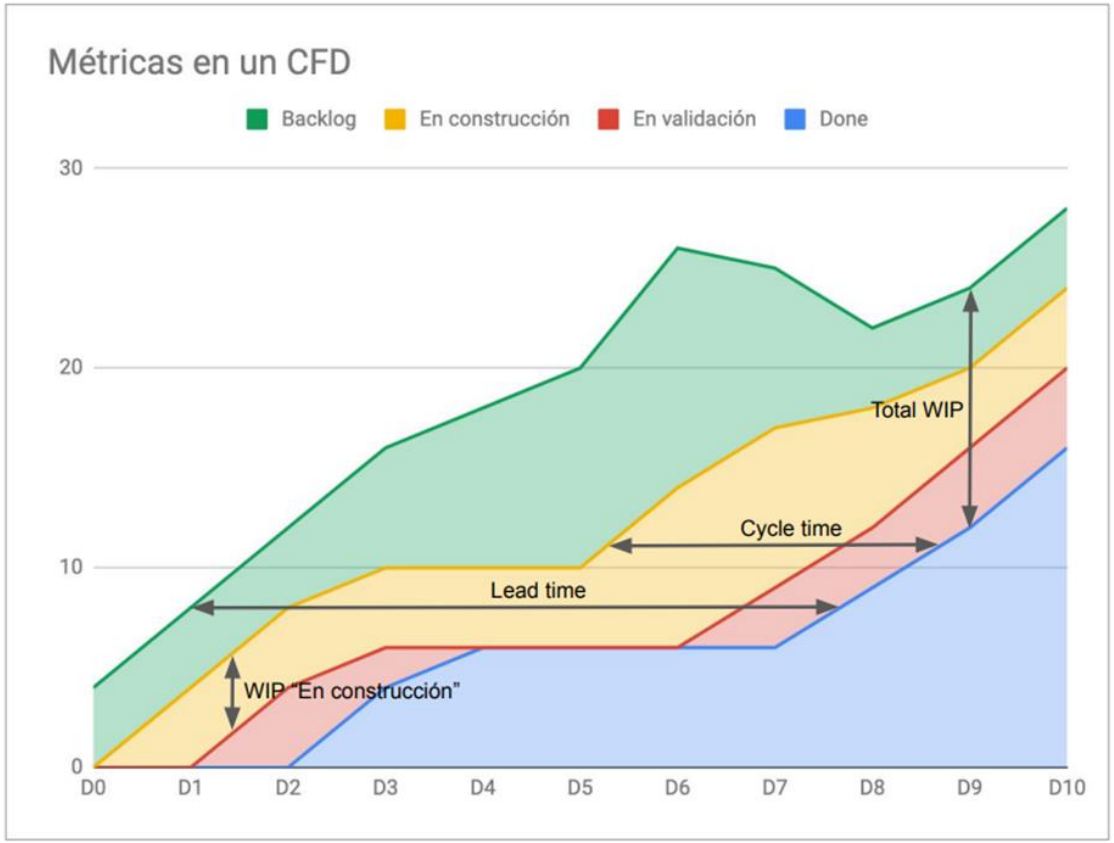


- El tiempo de ciclo comienza en el momento en que la nueva tarea entra en la etapa *En curso* y alguien está trabajando en ella.


Legend X
Ready to Start
In Progress
Done
Cycle Time
Lead Time
Work Items
25
20
15
10
5
0
2018-10-01
2018-10-08
2018-10-15

MÉTRICAS EN UN CFD

- ¿Por qué el WIP total incluye el *backlog*?
  - Para entenderlo hay que ver el sistema desde fuera, como una caja negra. Entran elementos (*backlog*), se procesan y se entregan (*Done*). Todo lo que no se ha entregado está en proceso (WIP total).
- Los «cortes» en un CFD:
  - Los cortes verticales muestran la cantidad de elementos de trabajo de cada estado para un momento dado.
  - Los cortes horizontales muestran los tiempos medios que los elementos de trabajo pasan en cada estado.

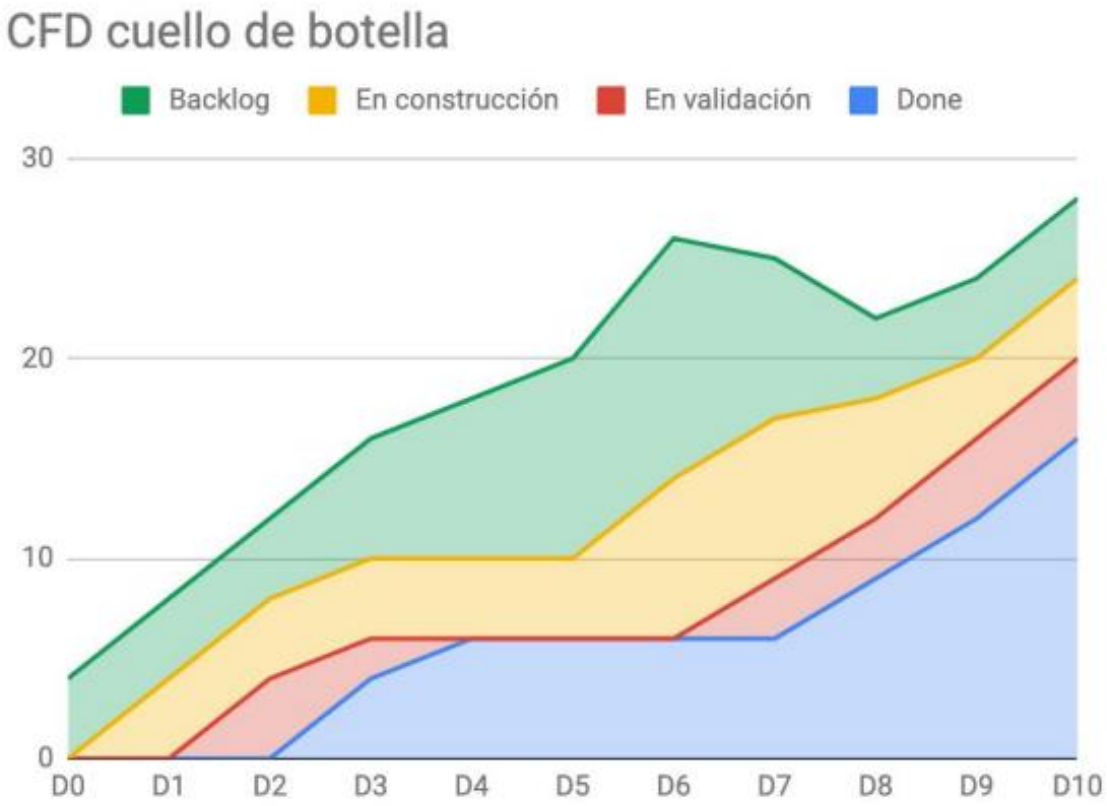


EJEMPLO
FLUJO CONTINUO EQUILIBRADO

- Un CFD habitualmente muestra un flujo continuo.
- La demanda se refleja con el primer estado del flujo de trabajo (en el ejemplo, el *backlog*).
- La capacidad global se refleja con el último estado (Done).
- Cuando las pendientes de las distintas rectas son iguales (son paralelas), significa que demanda y capacidad están alineadas.
- En el ejemplo se ve una situación ideal.

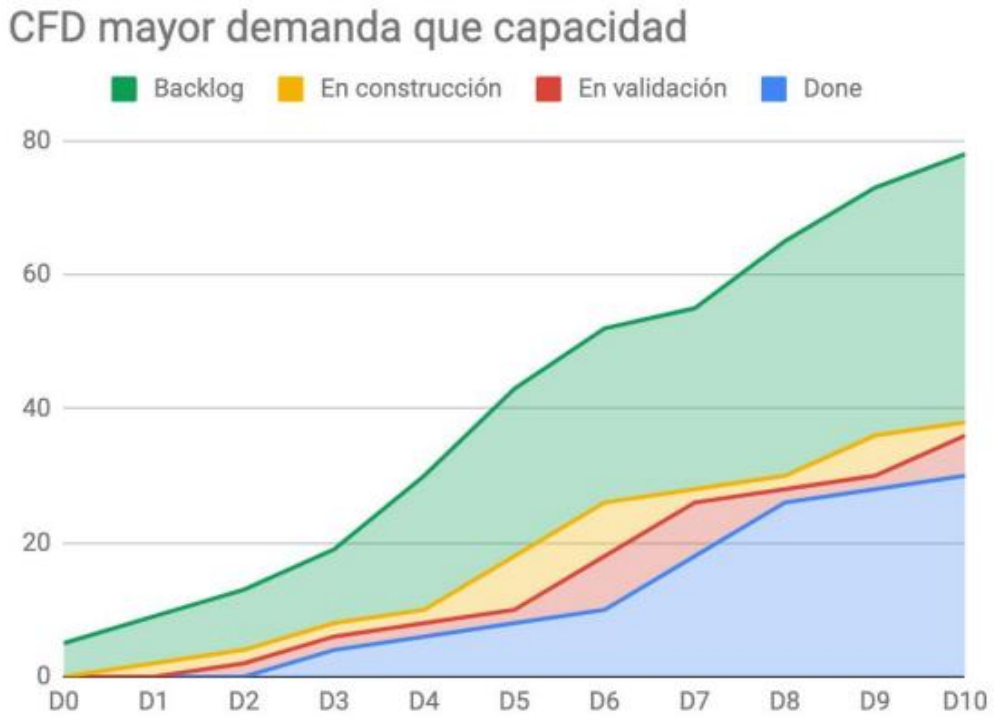
EJEMPLO - CUELLO DE BOTELLA

- Hasta el día 2 todo avanza equilibradamente.
- A partir de ahí algo se atasca en el estado *En construcción*. El tamaño del backlog comienza a crecer y en el estado *En validación* se van quedando sin trabajo.
- La situación se reconduce poniendo el foco en el estado donde hay problemas (se aumenta su capacidad), hasta que vuelven a equilibrarse capacidad y demanda.

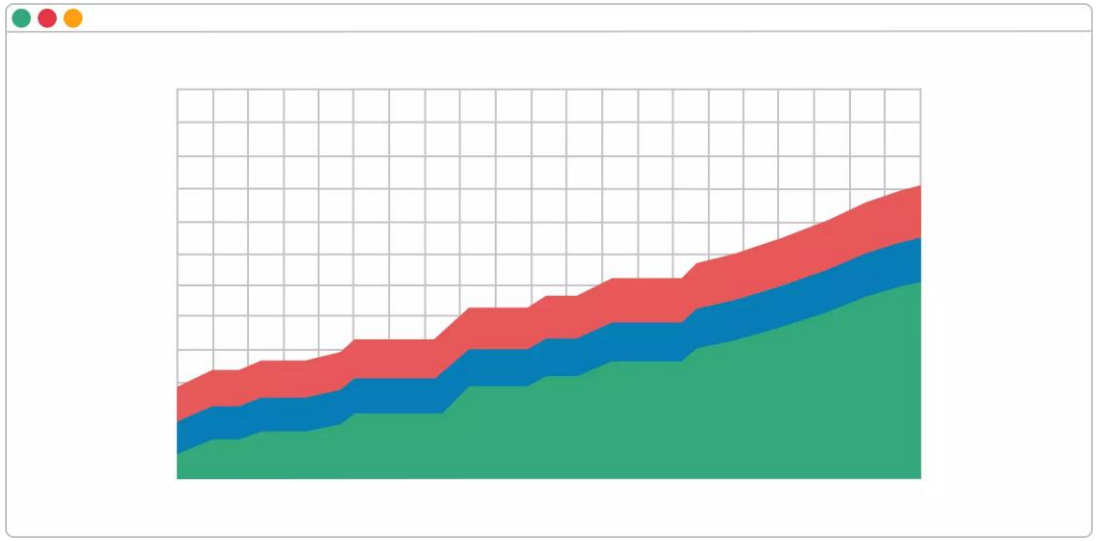


EJEMPLO - EQUIPO SATURADO

- Este CDF muestra como la demanda (representada en el *backlog*) crece a ritmo superior a la capacidad disponible.
- Se puede apreciar que el *backlog* va creciendo de tamaño, a pesar de que el equipo de trabajo tiene varios puntos de sobreesfuerzo para tratar de controlar la demanda.
- Si esta situación se mantiene, el equipo se saturará y probablemente baje la calidad de las entregas.

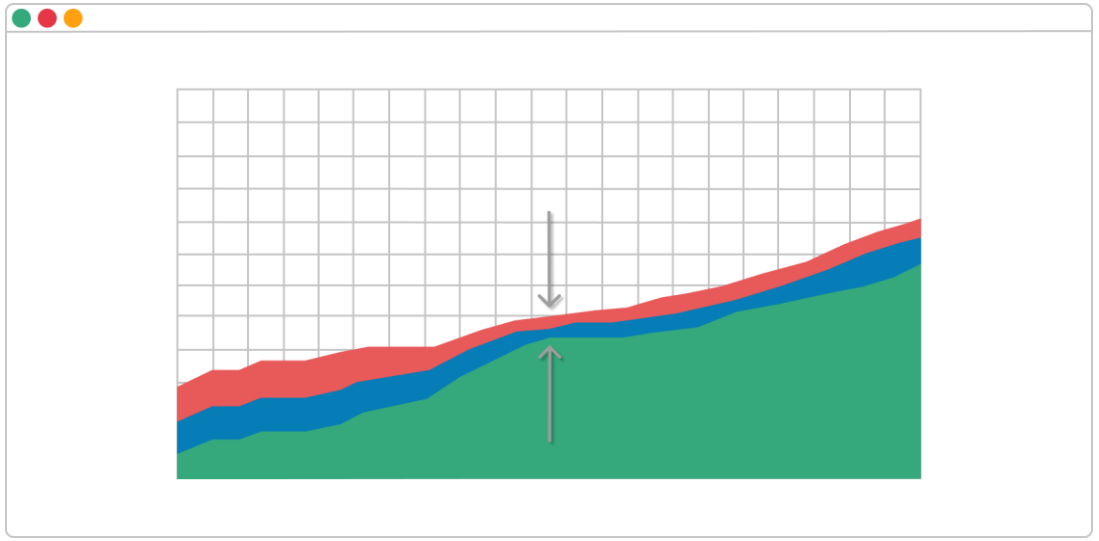


LAS BANDAS AVANZAN EN PARALELO



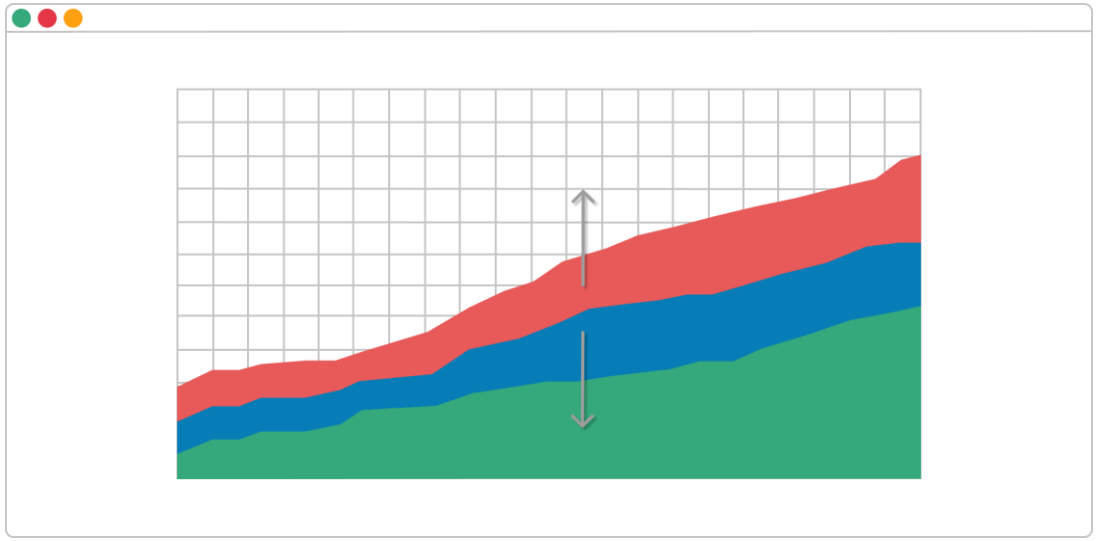
- El rendimiento es estable y que nuevas tareas están ingresando al flujo de trabajo en paralelo a las que salen.
- Este es el resultado ideal.
- Se pueden enfocar los esfuerzos en acortar los tiempos de ciclo de las tareas.

UNA BANDA SE ESTÁ REDUCIENDO RÁPIDAMENTE



- Si una banda en el DFA se está reduciendo continuamente, eso significa que el rendimiento de la etapa que representa es mayor que la tasa de entrada.
- Esta es una señal de que hay más capacidad de la que realmente se necesita en esta etapa y esta se debería reubicar para optimizar el flujo.

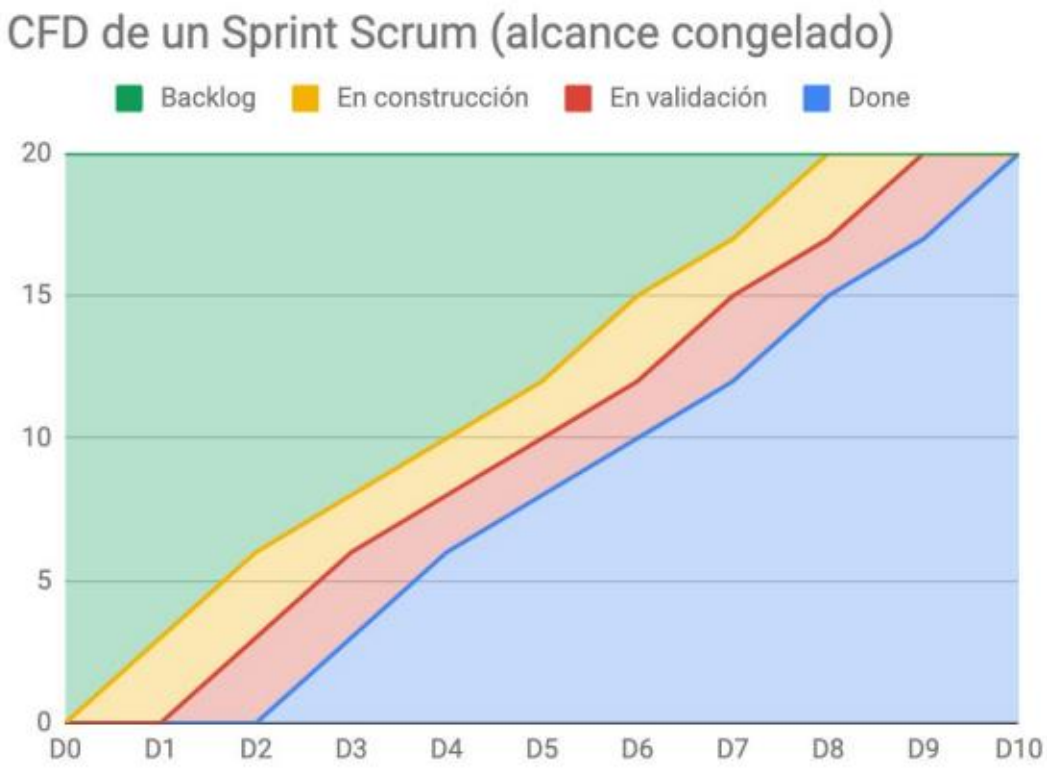
UNA BANDA SE ESTÁ ENSANCHANDO RÁPIDAMENTE



- La cantidad de tarjetas que entran en la etapa correspondiente en el tablero Kanban es mayor que el número de tareas que lo están dejando.
- Es un problema común que generalmente es causado por la multitarea y otras actividades de desperdicio que no generan valor.
- Reconsiderar los límites de WIP del tablero Kanban y concentrarse más en finalizar las tareas que están en curso antes de comenzar otras nuevas.

EJEMPLO *SPRINT* SCRUM

- Un CFD puede reflejar perfectamente la evolución de los elementos de trabajo en Scrum.
- En el ejemplo se refleja un *sprint*, lo cual tiene algunas particularidades:
  - El alcance se limita al *backlog* del *sprint*, confeccionado en la sesión de planificación del *sprint*.
  - El eje X (tiempo) se limita a la duración del *sprint* (10 días).
  - Al partir de un alcance dado (20), el *backlog* WIP comienza en su máximo y se reduce conforme pasa el tiempo.



EN SUMA

El diagrama de flujo acumulado es uno de los instrumentos de análisis más avanzados para la gestión de proyectos lean.

Proporciona una visualización concisa de las métricas de flujo.

Muestra cuán estable es el flujo y ayuda a entender dónde concentrarse para hacer que el proceso sea más predecible.

Proporciona una visión cuantitativa y cualitativa de los problemas pasados y existentes.

CAP. 21
COMUNICACIÓN DE LOS PLANES

COMUNICACIÓN DEL AVANCE

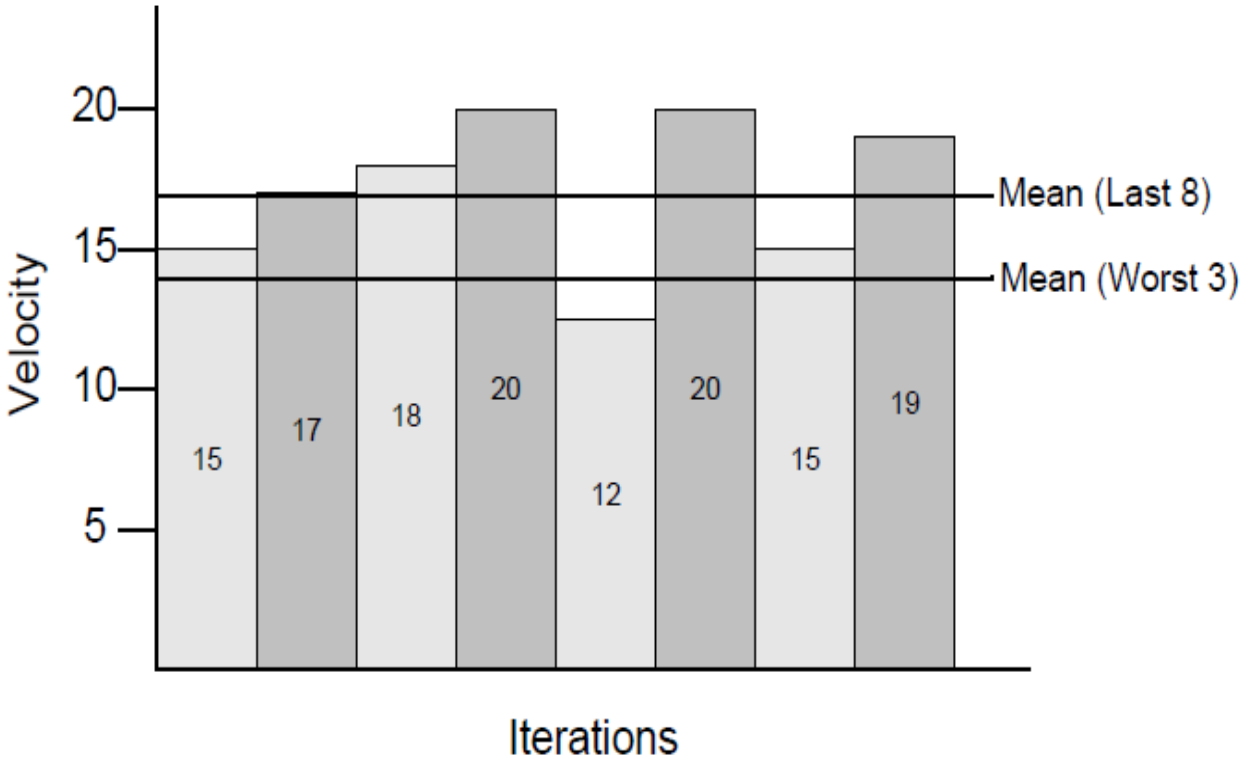
GRÁFICAS *BURN-DOWN*

- Las gráficas *burn-down* son el principal método para comunicar el avance.
- Son una función de la cantidad de trabajo pendiente y la velocidad del equipo.

PREDICCIÓN DE LA CANTIDAD DE ITERACIONES RESTANTES

- Para predecir la cantidad de iteraciones restantes dividir la cantidad de puntos de historia pendientes entre la velocidad del equipo.
- P. ej. 100 puntos pendientes. VE = 15. Iteraciones restantes = 100/15 = 6,66 → 7
- Sin embargo, la velocidad es fluctuante → usar un rango de velocidades para predecir la cantidad de iteraciones faltantes.
- Además, gráfica que muestre la velocidad del equipo por iteración.
- Pensar en un rango de velocidades.
- Usar:
  - La velocidad de la última iteración (muestra lo que acaba de pasar)
  - El promedio de las 8 iteraciones previas (muestra un promedio a largo plazo)
  - El promedio de las tres iteraciones con velocidad más baja de entre las últimas 8 (peor caso)

EJEMPLO



- La velocidad de la última iteración = 19
- Promedio de velocidad durante últimas 8 iteraciones = 17
- Promedio de las 3 velocidades más bajas en las 8 últimas iteraciones = 14

PREDICCIONES

Description	Velocity	Iterations	Total Points
Worst three	14	5	70
Last eight	17	5	85
Most recent	19	5	95

- El product owner podrá tener la certeza e que al menos obtendrá 70 puntos de historia adicionales durante las 5 iteraciones siguientes.
- No debería comprometerse con nada más allá de 85 puntos de historia.

TENDENCIAS DE LA VELOCIDAD

- Si se nota una tendencia a una velocidad en aumento:
  - no hacer nada al respecto.
- Si la velocidad parece estar disminuyendo:
  - buscar y eliminar la causa del declive
  - No planificar en una velocidad descendiendo

INFORME DE FIN DE ITERACIÓN

- Es útil
  - para difundir la información actual y
  - como documento histórico para usar en el futuro.
- Hacer el resumen lleva 30 m por iteración.

CONTENIDO DEL INFORME DE FIN DE LA ITERACIÓN

1. Contexto
   - Fechas
     - Fecha de inicio de la iteración
     - Fecha de fin de la iteración
     - Cantidad de días de la iteración
   - Personal
     - Personal disponible durante la iteración
     - Cantidad de días de trabajo planificados (puede variar de iteración en iteración por licencias)
     - Cantidad de días de trabajo reales (puede variar por enfermedad o días planificados de licencia que no se tomaron)
2. Métricas
   - Resultados de lo construido por día:
     - Cantidad de pruebas, si todas fueron exitosas
     - Cantidad de pruebas fallidas, si hubo una fallida
   - Gráfica *burn-down* de la iteración
   - Velocidad
3. Contenidos y evaluaciones
   - Resultados:
     - Historias, resultados, P. H. estimados, P. H. ganados
   - Revisión de la iteración:
     - Fecha y hora
     - Ítems identificados.

CONTEXTO

- Fechas

First Day of Iteration	September 1
Last Day of Iteration	September 14
Number of Working Days	9

- Personal

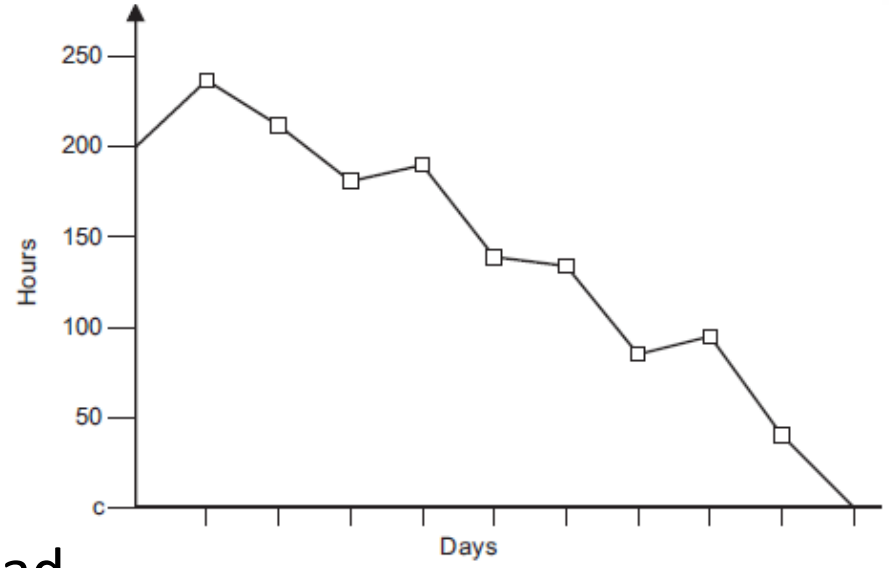
Name	Days Planned	Days Worked
Carina	9	9
Vadim	9	7
Sasha	8	9
Dmitri	9	9
Total	**35**	**35**

MÉTRICAS

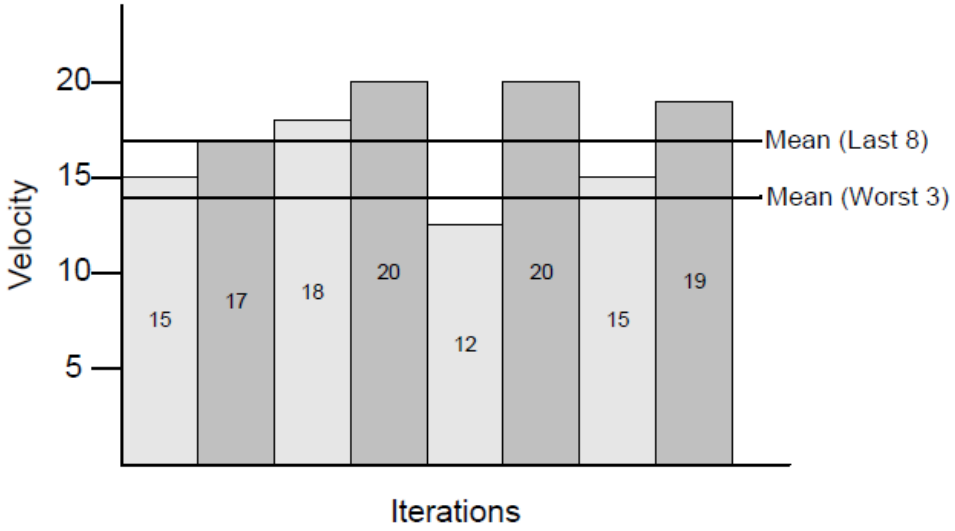
- Resultados de lo construido por día:

Metric	Status
Monday, June 6	Build failed
Tuesday, June 7	812 unit tests passed; 1431 FitNesse tests passed; 104 user interface tests passed
Wednesday, June 8	826 unit tests passed; 1433 FitNesse tests passed; 104 user interface tests passed
Thursday, June 9	1 unit test failed
Friday, June 10	841 unit tests passed; 1442 FitNesse tests passed; 105 user interface tests passed
…	…

- Gráfica *burn-down* de la iteración



Velocidad



CONTENIDOS Y EVALUACIONES

- Resultados:

Story	Result	Points Planned	Points Earned
As a coach, I can enter the names and demographic information for all swimmers on my team.	Finished	8	8
As a coach, I can define practice sessions.	Finished; bigger than planned	8	8
As a swimmer, I can see all of my times for a specific event.	Finished	3	3
As a swimmer, I can update my demographics information.	Started; not finished	1	0
Total		**20**	**19**

- Revisión de la iteración:
- La revisión de la iteración tuvo lugar el 15 de setiembre a las 9 h.
- Se identificaron los siguientes ítems:

Action Item	Responsible
Mark pointed out that the session planning logs used by most coaches he knows list information in the opposite order we show on the screen. We need to run those screens by some coaches.	Carina
Gary suggested we add a story about graphing a swimmer's results in an event.	Carina