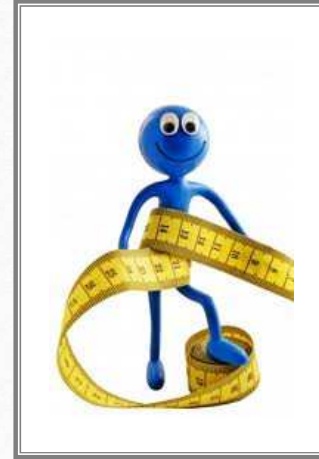


Universidad Tecnológica Nacional
Cátedra de Ingeniería de Software
Docentes: Judith Meles y Laura Covaro

Filosofía Lean



Judith Meles 1

1

Principios

Eliminar Desperdicios

Amplificar Aprendizaje

Embeber la Integridad conceptual

Diferir compromisos

Dar poder al equipo

Ver el todo

Respetar a la Gente

Principios Lean



2

2



3





4



5

Kanban en pocas palabras

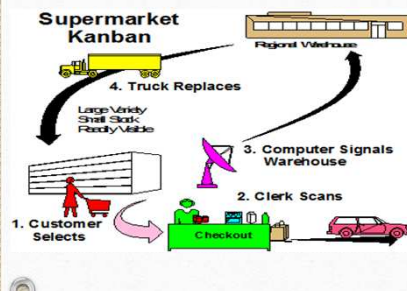
- **kan-ban** (看板) = Signal-card.



6

Kanban en pocas palabras-Just in Time

- A fines de 1940, Toyota comenzó a estudiar técnicas de almacenamiento y tiempo de stockeo de los supermercados



Taiichi Ohno



1953年当時の「かんばん」
後工程が必要なものを必要な時に必要な量だけ
前工程から引き取る。前工程を引き取った分だけ生産する。
「かんばん」は、それを実現するための道具のひとつ。

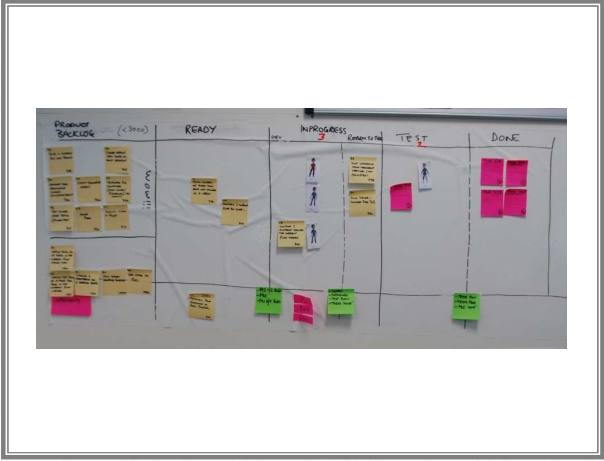
7

Kanban en pocas palabras: Administración de Colas

- Los cajeros se focalizan en tomar órdenes.
- El Barista se focaliza en proveer café.
- Separarlos por la cola permite que se absorba la demanda variable.
- Los cajeros se mueven a ayudar al Barista cuando no hay clientes esperando para hacer su pedido.
- Foco es en Flujo "fin a fin" FLOW = Centrado en el Cliente



8



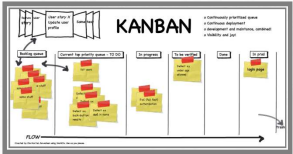
Kanban en pocas palabras

- Principios:
 - Visualizar el Flujo: Hacer el trabajo **visible**.
 - Limitar el **Trabajo en progreso (WIP)**
 - Administrar el flujo: Ayudar a que el **trabajo fluya**
 - Hacer explícitas las políticas.
 - Mejorar colaborativamente.

29

9

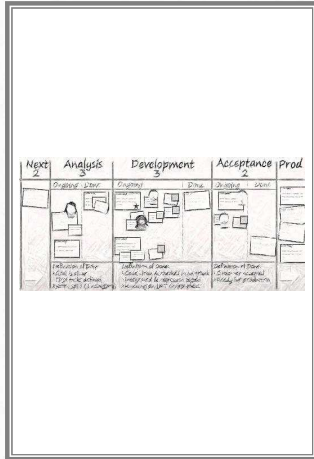
Kanban en el Desarrollo de Software



- El método fue formulado por David J. Anderson
- Es un enfoque para gestión de cambio.
- No es un proceso de desarrollo de software o una metodología de administración de proyecto.
- Kanban es un método para introducir cambios en un proceso de desarrollo de software o una metodología de administración de proyectos

10

10

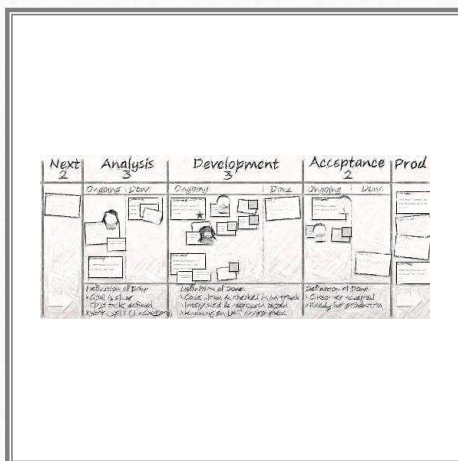


Kanban en el Desarrollo de Software

- Kanban aprovecha muchos de los conceptos probados de Lean:
 - Definiendo el Valor desde la perspectiva del Cliente.
 - Limitando el Trabajo en Progreso (WIP).
 - Identificando y Eliminando el Desperdicio.
 - Identificando y removiendo las barreras en el Flujo.
 - Cultura de Mejora Continua.

11

11

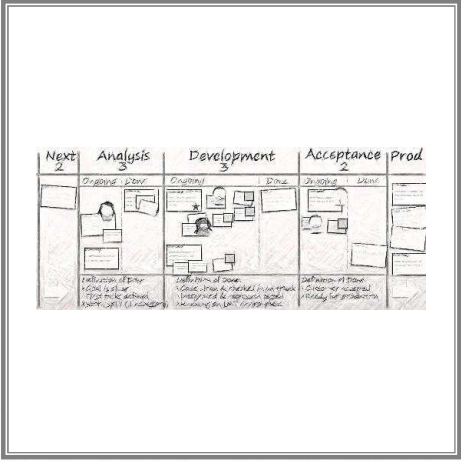


Kanban en el Desarrollo de Software

- Kanban fomenta la evolución gradual de los procesos existentes.
- Kanban no pide una revolución, sino que fomenta el cambio gradual.
- Kanban está basado en una idea muy simple: Limitar el trabajo en progreso (WIP).
- El Kanban (o tarjeta de señal) implica que una señal visual se produce para indicar que el nuevo trabajo se puede tirar ("pull") porque el trabajo actual no es igual al límite acordado.

12

12



¿Cómo aplicar Kanban?

- Empezar con lo que se tiene ahora.
- Entender el proceso actual.
- Acordar los límites de WIP para cada etapa del proceso.
- A continuación, comienza a fluir el trabajo a través del sistema tirando de él, en presencia de señales Kanban.

13

13

¿Cómo aplicar Kanban?

- Visualizar el flujo de trabajo:
 - Dividir el trabajo en piezas, las user stories son buenas para eso.



Visualice su trabajo



Use colores diferentes para diferentes tipos de trabajo



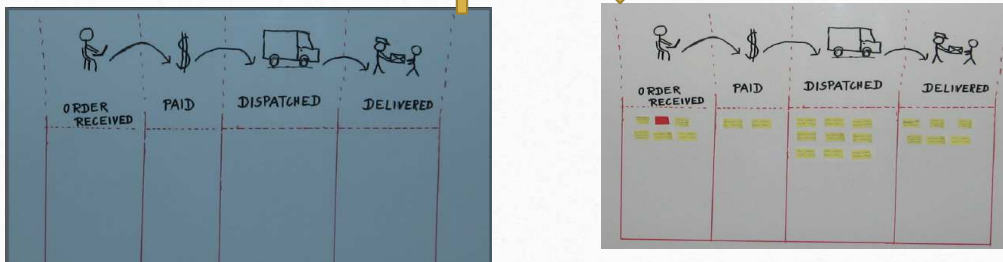
Escriba cada tarea en una nota diferente

14

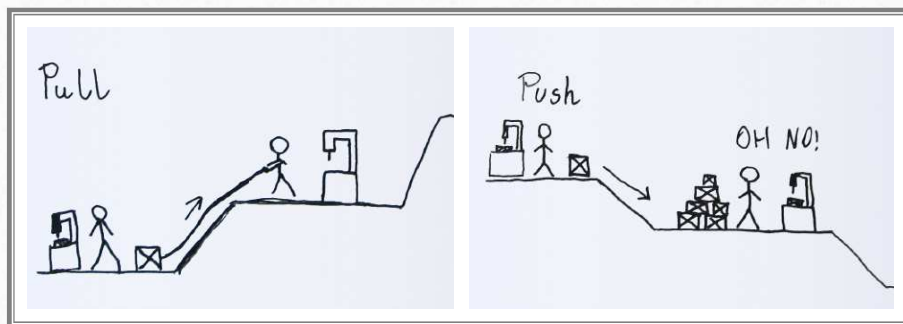
14

¿Cómo aplicar Kanban?

- Visualizar el flujo de trabajo:
 - Utilizar nombres en las columnas para ilustrar donde está cada ítem en el flujo de trabajo.
 - Distribuir el trabajo en las columnas: el trabajo fluirá de izquierda a derecha en las columnas.



15



Pull, no push !!!

16

16

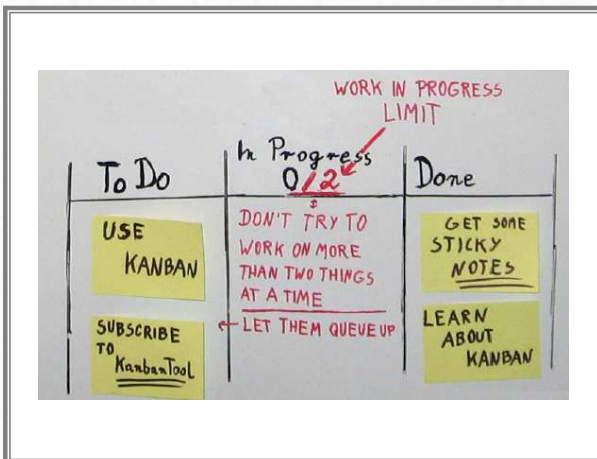
Por último, la auto asignación de tareas se ve reflejada con un avatar personalizado... :))



17

¿Cómo aplicar Kanban?

- Limitar WIP – Asignar límites explícitos de cuántos ítems puede haber en progreso en cada estado del flujo de trabajo.



18

18

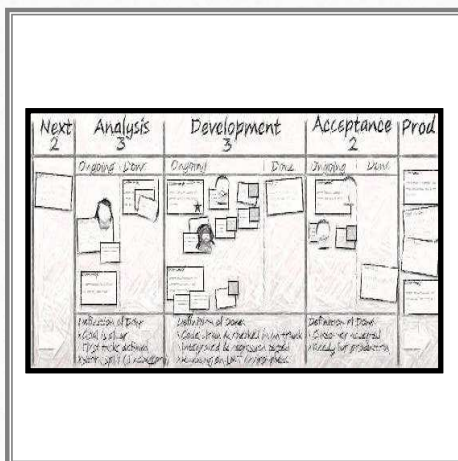


Ayudar a que el trabajo fluya....

Al 100 % de capacidad se tiene un rendimiento mínimo...

19

19

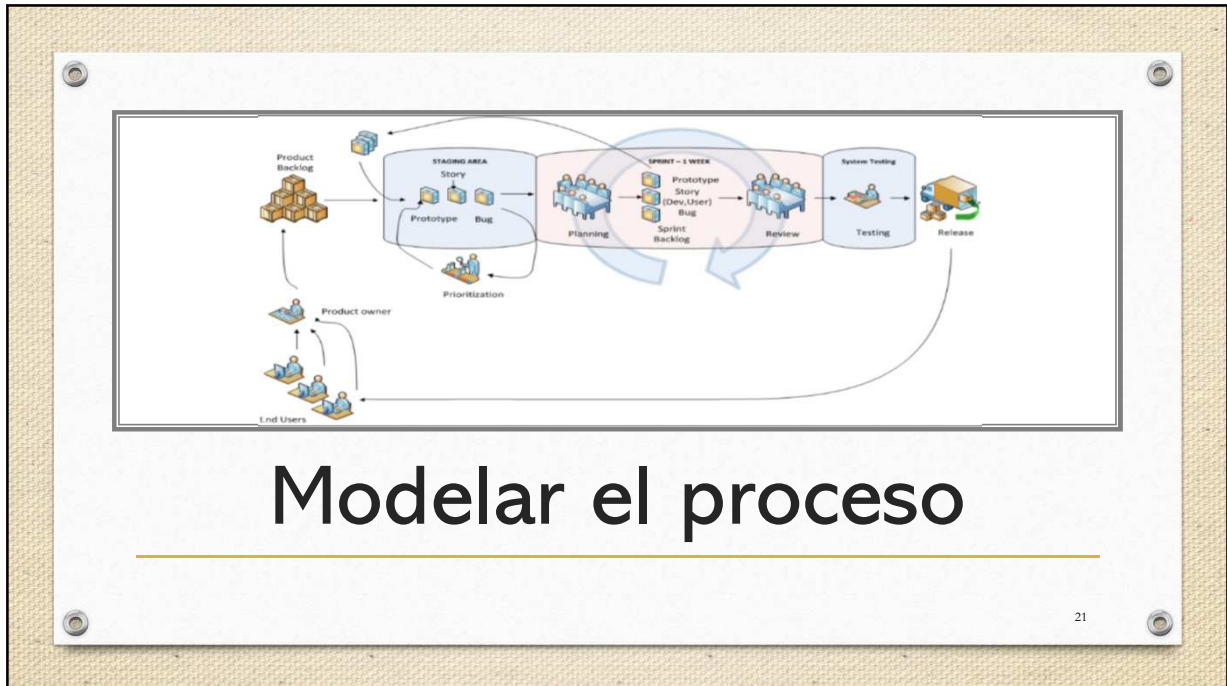


¿Cómo aplicar Kanban en nuestro proyecto?

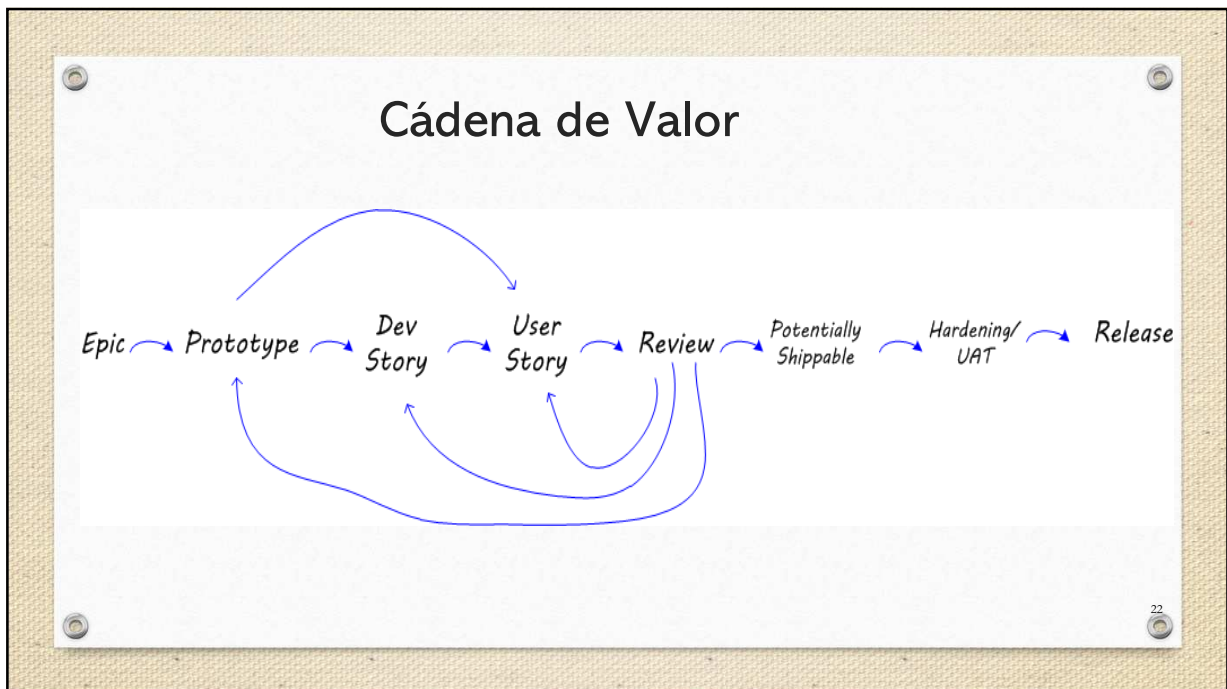
- **Proceso:** modelar nuestro proceso.
- **Trabajo :** decidir la unidad de trabajo.
- **Límites de WIP:** limitar el WIP para ayudar al flujo de trabajo.
- **Política:** definir políticas de calidad.
- **Cuellos de Botella y Flujo:** mover recursos a los cuellos de botella.
- **Clase de Servicio:** diferentes trabajos tienen diferentes políticas – definición de hecho ("done"), para cada estado.
- **Cadencia:** Releases, planificaciones, revisiones

20

20



21



22

23

Definir el proceso...

Cola de Producto	Análisis		Desarrollo		Listo para Build	En Testing		En Producción
	En progreso	Hecho	En progreso	Hecho		En Progreso	Listo para Despliegue	

23

Definir tipos de trabajo...

Asignando capacidad en función de la demanda

Requerimientos

- Caso de uso
- Historias de Usuario
- Porciones de Casos de Uso
- Características

Defectos

- Defectos en Producción
- Defectos

Desarrollo

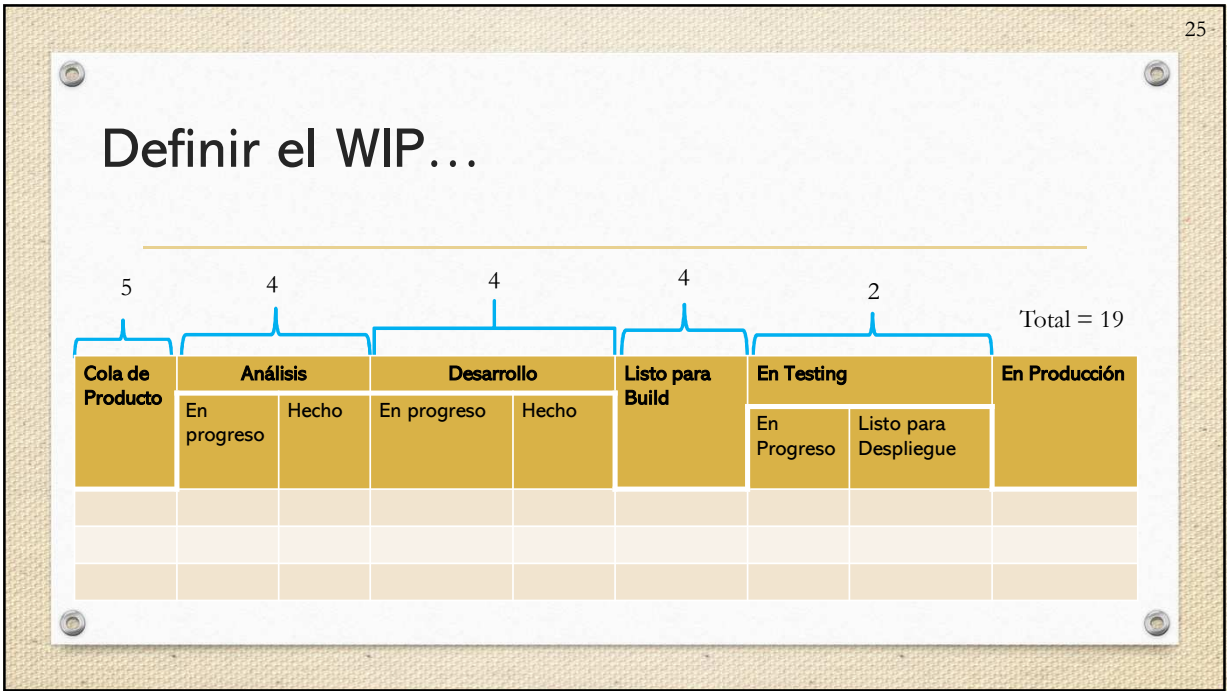
- Mantenimiento
- Refactorización
- Actualización de Infraestructura

Solicitudes

- Solicitud de Cambio
- Sugerencias de Mejora

24

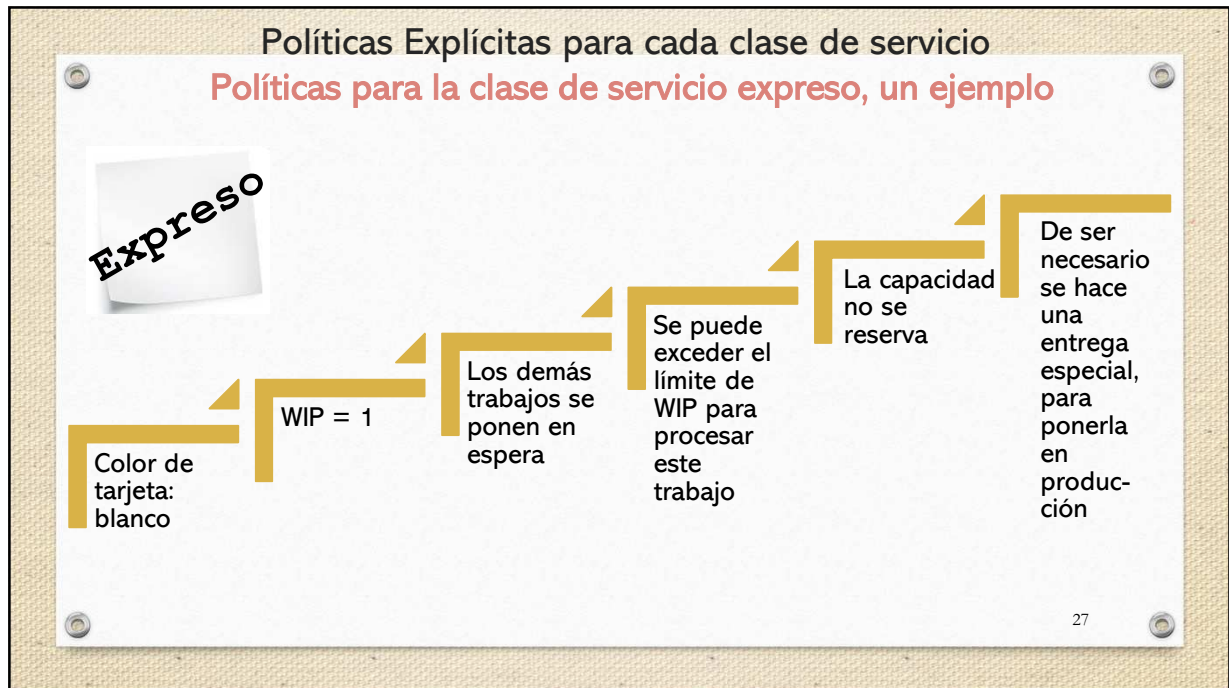
24



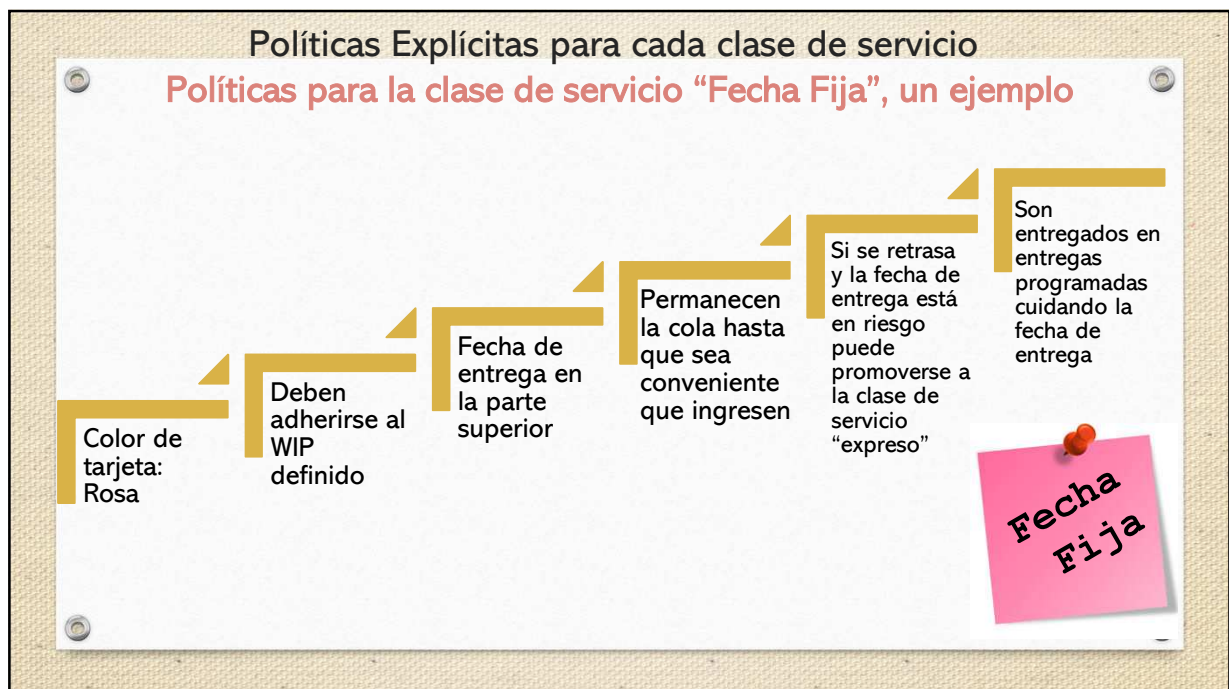
25



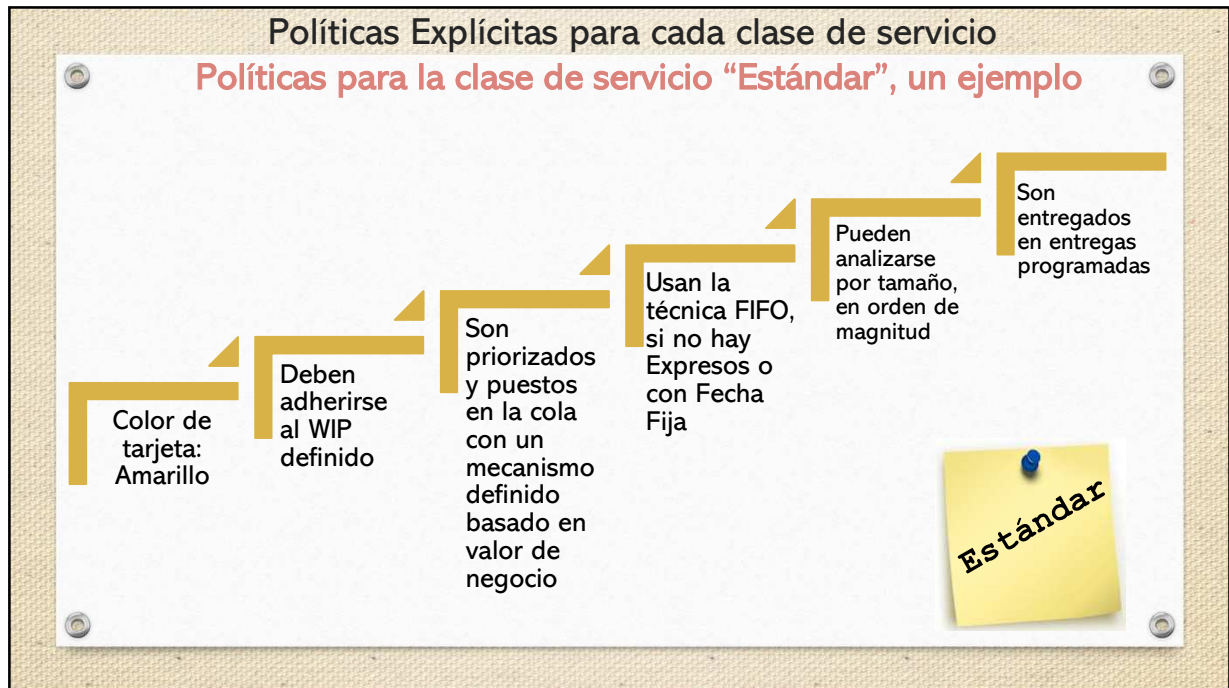
26



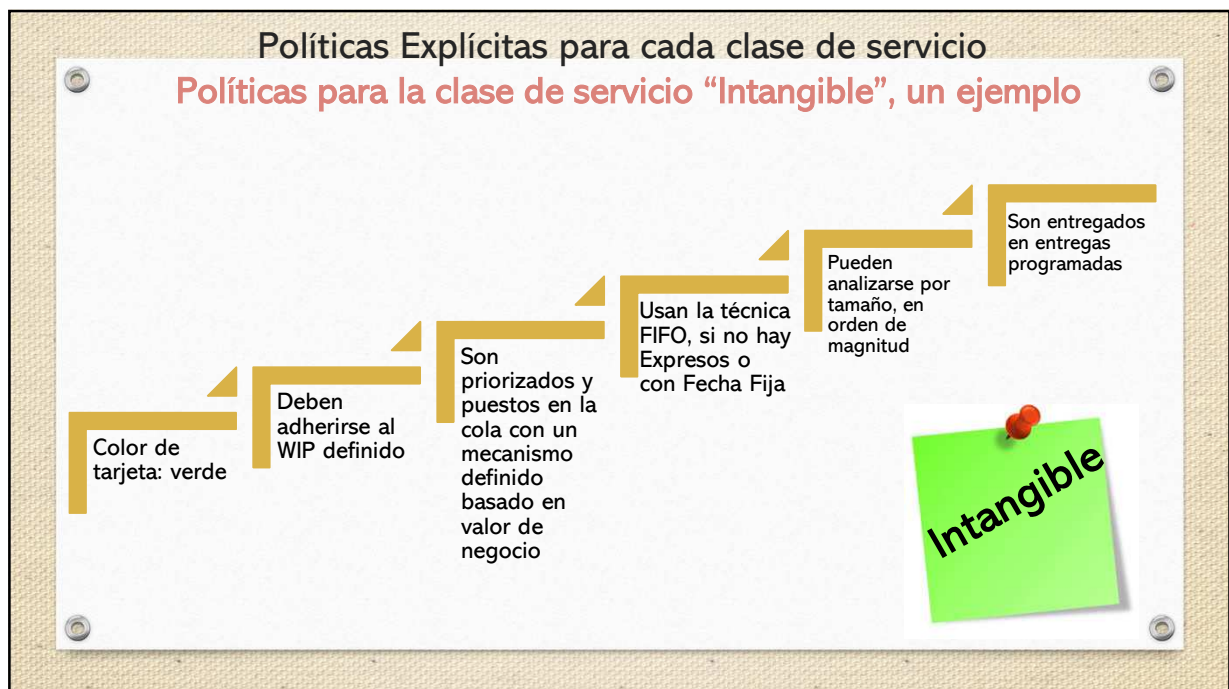
27



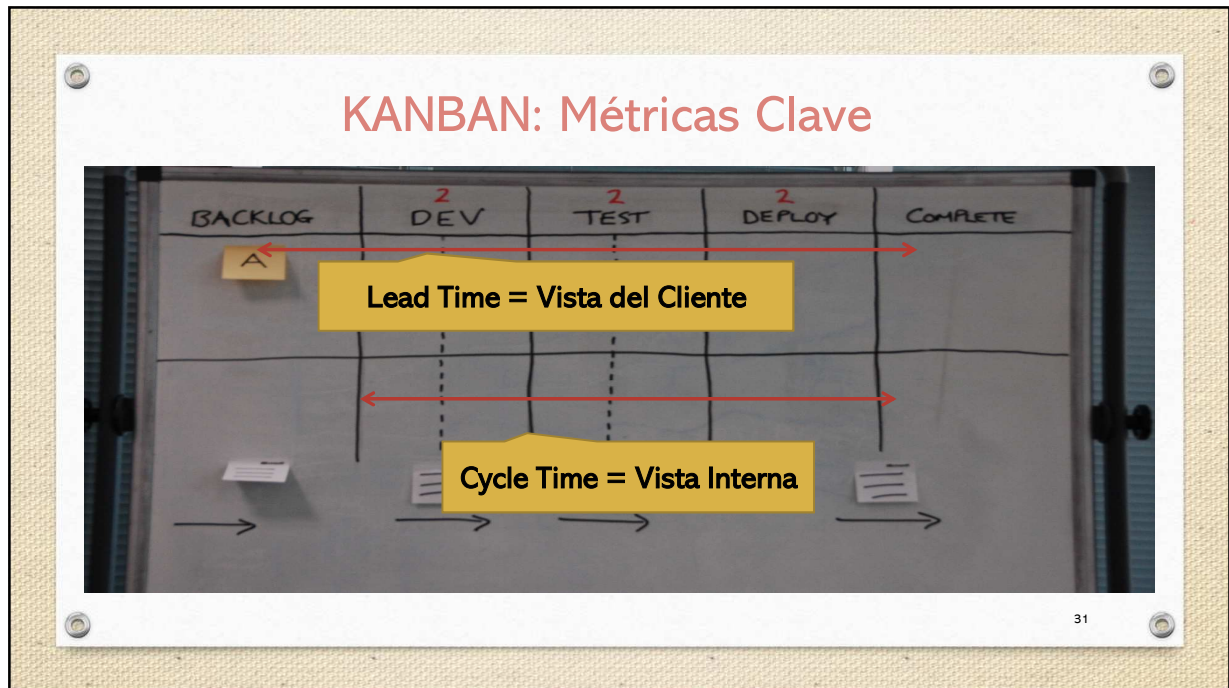
28



29



30



31

KANBAN: Métricas Clave

Cycle Time (Tiempo de ciclo)

- Es la métrica que registra el tiempo que sucede entre el inicio y el final del proceso, para un ítem de trabajo dado. Se suele medir en días de trabajo o esfuerzo.
- Medición más mecánica de la capacidad del proceso
- **Ritmo de Terminación**

Lead Time (Tiempo de entrega)

- Es la métrica que registra el tiempo que sucede entre el momento en el cual se está pidiendo un ítem de trabajo y el momento de su entrega (el final del proceso). Se suele medir en días de trabajo.
- **Ritmo de entrega**

32

KANBAN: Métricas Clave

Touch Time (Tiempo de Tocado)

- El tiempo en el cual un ítem de trabajo fue realmente trabajado (o "tocado") por el equipo.
- Cuántos días hábiles pasó este ítem en columnas de "trabajo en curso", en oposición con columnas de cola / buffer y estado bloqueado o sin trabajo del equipo sobre el mismo.

$$\text{Touch Time} \leq \text{Cycle Time} \leq \text{Lead Time}$$

Eficiencia del Ciclo de Proceso

$$\% \text{ Eficiencia ciclo proceso} = \text{Touch Time} / \text{Elapsed Time}.$$

33

33

More prescriptive

More adaptive



34

Scrum – Kanban: Similitudes

35

- Ambos son Lean y Ágiles
- Emplean sistemas de planificación Lean.
- Establecen límites WIP.
- La visibilidad del proceso es la base de su mejora.
- Objetivo: entrega temprana y frecuente de software.
- Equipos auto-organizados.
- División del trabajo en partes.
- Revisión y mejora continua del plan del producto, basado en datos empíricos.

35

Scrum – Kanban: Diferencias

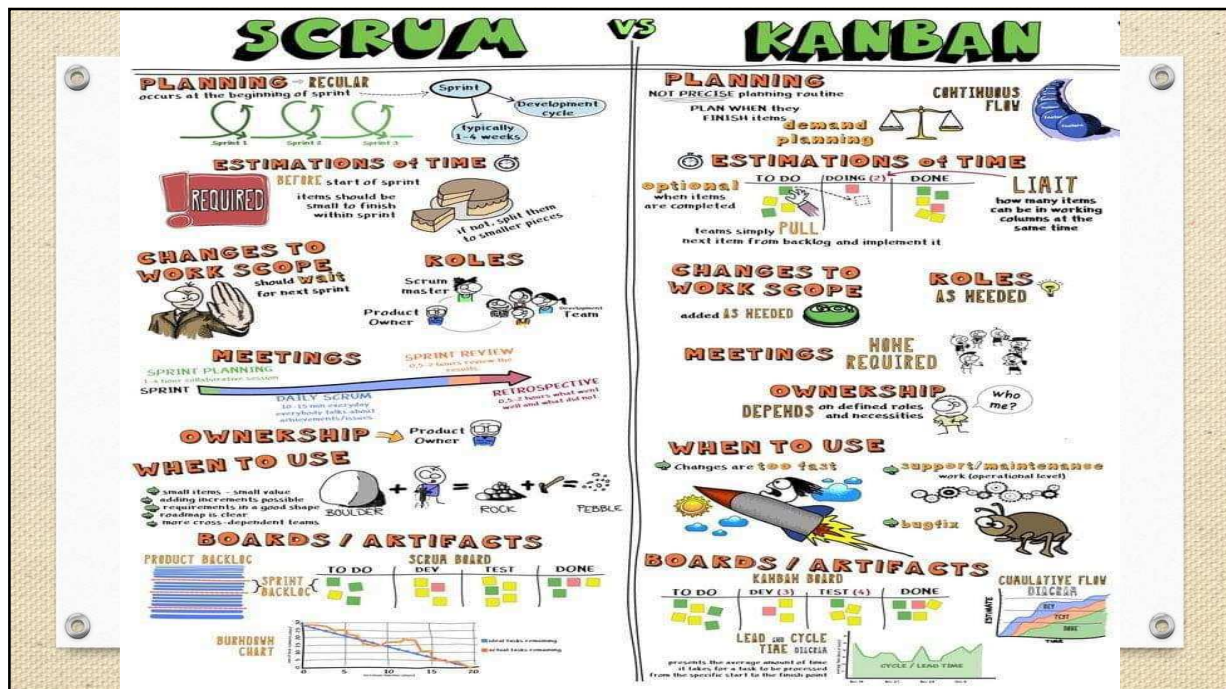
Scrum	Kanban
Iteraciones de tiempo fijo.	Tiempo fijo es opcional. La cadencia puede variar. Pueden estar marcadas por la previsión de los eventos en lugar de tener un tiempo prefijado.
Equipo asume un compromiso de trabajo por iteración.	El compromiso es opcional.
Métrica para planificación y mejora: Velocidad.	Métrica por defecto es Lead Time (Tiempo de Entrega o tiempo medio)
Equipos Multifuncionales.	Equipos Multifuncionales o especializados.
Funcionalidad divididas para poder completarse en un Sprint.	No hay prescripción respecto del tamaño de la funcionalidad.
Deben emplearse gráficos Burndown chart .	No se prescriben diagramas de seguimiento.

36

Scrum – Kanban: Diferencias

Scrum	Kanban
Limitación WIP indirecta (por Sprint).	Limitación WIP directa (marcada por el estado del trabajo)
Se deben realizar estimaciones.	Las estimaciones son opcionales.
No se puede agregar alcance en medio de una iteración.	Siempre que haya capacidad disponible se puede agregar trabajo.
Sprint Backlog pertenece a un equipo determinado.	Varios equipos pueden compartir pizarra Kanban.
Se prescriben tres roles (PO / SM/ Equipo).	No hay roles prescritos.
En cada sprint se limpia el tablero de seguimiento.	El tablero Kanban es persistente.
Product Backlog Priorizado.	La priorización es opcional.

37



38