


Métricas de DevOps.

Definición de uso de las métricas

- Lead Time: refiere al tiempo desde que se crea una tarea (se coloca en TO DO) hasta que se finaliza la tarea (pasa a DONE). Se mide en alguna unidad de tiempo, en este caso: días.
- Cycle Time: refiere al tiempo desde que se comienza una tarea (pasa a IN PROGRESS) hasta que se finaliza la tarea (pasa a DONE). Se mide en alguna unidad de tiempo, en este caso: días
- Touch Time: refiere al tiempo en el cual un ítem de trabajo fue realmente trabajado por el equipo. Esto es, el tiempo que estuvo la tarea en la columna IN PROGRESS menos el tiempo que estuvo bloqueada (por alguna restricción ajena a ella) y menos el tiempo que estuvo en cola en espera que alguien la comience. Se mide en alguna unidad de tiempo, en este caso: horas
- Flow Efficiency: se utiliza para evaluar la eficiencia con la que se realiza el trabajo a medida que fluye a través del tablero. Se centra en medir el tiempo real de trabajo en comparación con el tiempo total que lleva completar una tarjeta.
- Throughput: Se utiliza para evaluar y medir la capacidad y la productividad de un sistema, proceso o equipo en términos de la cantidad de trabajo completado en un período de tiempo determinado. La unidad que usamos es la cantidad de tarjetas hechas por día de trabajo, y la calculamos dividiendo la cantidad de tarjetas hechas para una entrega sobre el *lead time* de dicha tarjeta.

Se obtuvieron estos datos del movimiento de las tarjetas que queda registrado en Github en la issue de la entrega 2. Sin embargo, para la entrega 3 se ingresó tarde la tarjeta como issue en el desarrollo, por lo que se juntaron los datos por los comentarios anotados y por el control de versiones de la documentación, la cual incluye los casos de uso para esta ocasión.

Para el cálculo de la flow efficiency, se pasó el lead time de días a horas asumiendo que $\text{horas trabajadas} = \text{días trabajados} * \text{horas de trabajo por día}$. Como no trabajamos de manera constante en el proyecto, calculamos el promedio de horas trabajadas por día, el cual nos dio 1,9, el cual lo redondeamos a 2 y lo tomamos como las horas de trabajo por día.

Se adjunta un tablero donde se han calculado todas las métricas de las entregas 2,3 y 4, para que se constate su debido calculo:  [Metricas](#)

Informe de Análisis de Métricas de DevOps para el Equipo

Entrega	Leadtime promedio	Cycletime promedio	Touchtime promedio	Flow efficiency promedio	Throughput
2	2	1	0,75	18,75	1
3	13	3	9,9	38,07692308	0,15384615 38
4	4	2,5	1,25	15,625	1

En la cuarta iteración, se observa una reducción significativa en el Lead Time, el cual se redujo a 4 días en comparación con la iteración anterior. Teniendo en cuenta además la mejora abismal en el throughput (de 0,15 a 1), podemos afirmar con certeza que esto refleja una mejora en la eficiencia para completar las tareas en menos tiempo.

El Cycle Time también disminuyó a 2.5 días, que combinado también con el aumento de throughput indica una reducción en el tiempo de trabajo en cada tarea.

El Touch Time bajó a 1.25 horas, lo que indica una disminución en el tiempo dedicado al trabajo real en las tareas.

El Flow Efficiency disminuyó a 15.6%, lo que nos daría a entender que hubo más tiempo muerto en esta entrega que en la anterior, lo que a primera vista parece una disminución de la eficiencia. Sin embargo, esto sería erróneo ya que la eficiencia de flujo depende del touchtime, y el touchtime de la entrega anterior es prácticamente 9 veces más grande que el de esta entrega, siendo este el factor el que infla el flow efficiency de la entrega anterior.

¿Qué es lo que refleja la disminución de la eficiencia de flujo? Refleja que cualquiera el factor que entorpece el flujo, tiene una duración de tiempo constante, por lo que cuanto más tiempo se emplea codificando menor es su impacto. Esto lo deducimos de la fórmula de flow efficiency = $\frac{\text{touchtime}}{\text{leadtime}} = \frac{\text{touchtime}}{\text{touchtime} + \text{tiempo perdido}}$. Si el tiempo perdido es constante, entonces a mayor touchtime, flow efficiency tiende a 1.

Teniendo en cuenta que el leadtime de la entrega 3 aumentó significativamente pero el flow efficiency mejoró, y que el leadtime y el flow efficiency son inversamente proporcionales, podemos confirmar que el cuello de botella no se encuentra en la salida del backlog al ciclo de desarrollo.

Por lo tanto deducimos que el cuello de botella se encuentra en review con el PO, ya que es la única actividad restante que es de tiempo constante en todas las iteraciones. Tiene sentido que este sea el cuello de botella, ya que la review se realiza al terminar la entrega, por lo que una tarjeta "completa" tiene que esperar que se terminen todas las otras tarjetas antes de llegar a "Done". Esto explica también por que a pesar de tener el mismo throughput y touchtime similar al de la entrega 2, el flow efficiency empeoró ya se tenían que realizar más tarjetas, por lo que más tarjetas quedaron esperando en el cuello de botella.

Más adelante en la retrospectiva, se analizan más en detalle las métricas del equipo para identificar las causas de los cambios observados y buscar oportunidades de mejora en futuras iteraciones.