The Technology Value Stream

CSD: 380 DevOps
Austen Rhyce Erickson
Assignment 1.2
1/12/25

Defining Lead Time vs. Processing Time

LEAD TIME

- The total time it takes from when a feature is requested until that feature is delivered and available for use
- Encompasses the entire process, including time spent waiting or in handoffs
- Inherently customer-centric view of how long it takes to fulfill a feature request
- Examining lead time gives insight into inefficiencies in the entire value stream.
- It provides a customer-centric view of how long it takes to fulfill a request, capturing inefficiencies in the entire value stream.

Prometheus

Defining Lead Time vs. Processing Time

PROCESSING TIME

- The time actively spent working on a task or feature, NOT including any time the work is waiting or idle.
- It focuses on the work-in-progress portion of the value stream.
- Examining processing time gives insights into the efficiency of the actual work processes. Delays caused by external factors are ignored.

Defining Lead Time vs. Processing Time

Lead Time

Example: If a new feature is requested on January 4st and is delivered on January 10th, the lead time is 6 days.

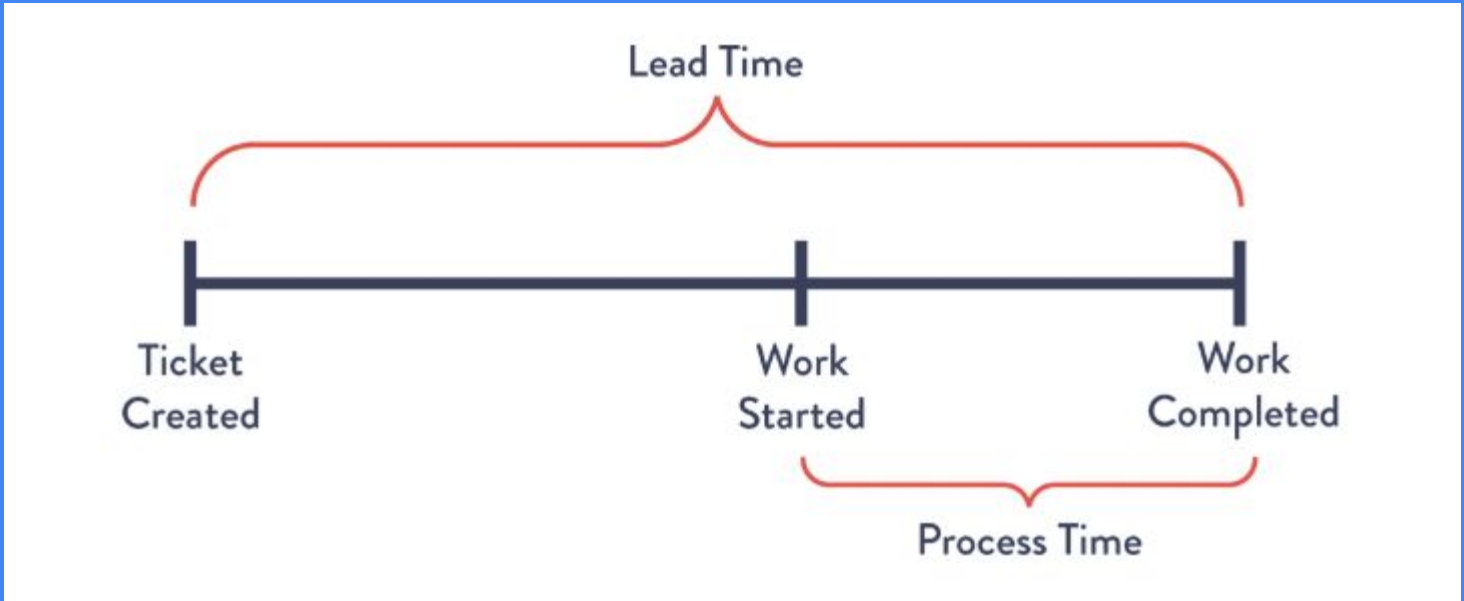
Processing Time

Example: If a developer spends 3 days actively coding a feature and 2 days testing it, the processing time is 5 days, even if the feature spent an additional 5 days waiting for review.

Both are metrics used to help with efficiency of the value stream

HashiCorp Vault

Defining Lead Time vs. Processing Time



An illustration taken from our textbook. Notice how the process time is included in the lead time

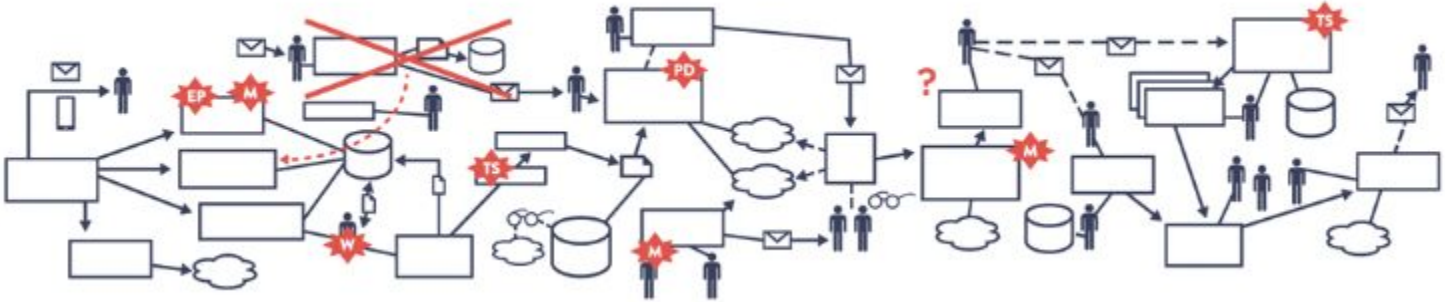
The Common Scenario: Deployment Lead Times Requiring Months

Jenkins

It is often the case, especially in large organizations, where lead times are very long – on the scale of months. The causes of this include:

- **WIP overload** – too many features in progress at the same time
- **Manual processes** – lack of automation
- **Workflow bottlenecks** – insufficient resources at certain stages / inefficient resource allocation
- **Inefficient handoffs** – poor communication/coordination between teams
- **Unclear prioritization** – shifting or unclear priorities
- **Long feedback loops** – delays in getting feedback from stakeholders
- **Excessive approval gates** – approval process is too involved/layered/complex
- **Infrequent releases** – low frequency of releases means more time waiting

The Common Scenario:
Deployment Lead Times Requiring Months



Our textbook illustrates well the madness that can happen with out of control lead times

Our DevOps Ideal:
Deployment Lead Times of Minutes

In contrast with long lead times, short lead times improve upon the characteristics of long lead times with automation and optimization

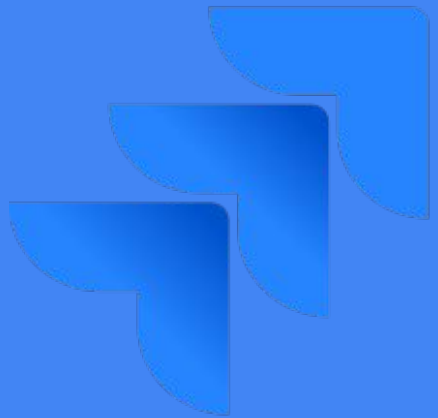
- Effective use of CI/CD
- Frequent deployments
- Regular feedback
- Quick approval processes – including timely pull request reviews

Benefits of Low Lead Times

- Increased customer satisfaction
- Increased market competitiveness
- Improved team morale

kubernetes

Grafana



Jira

Sources:

- Kim, Gene, et al. *The DevOps Handbook, Second Edition*. IT Revolution, 30 Nov. 2021.

- Atlassian. “DevOps Metrics.” *Atlassian*, www.atlassian.com/devops/frameworks/devops-metrics.

- “Cycle Time vs. Lead Time: A Comprehensive Guide.” *IT Revolution*, 10 June 2024, itrevolution.com/articles/cycle-time-vs-lead-time/.

- “73 Most Useful DevOps Tools: The Comprehensive List for 2023.” *Spacelift*, spacelift.io/blog/devops-tools.