Process Mining: From Theory to Practice

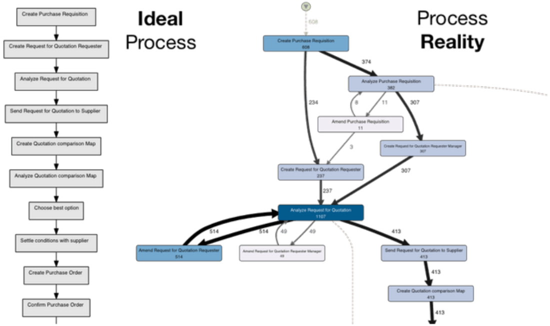
A Lightning Introduction

Garrett Gruss

CS8317 Sect 795A 1262

February 6, 2026

What is Process Mining?



- **Bridge between data science and process management**
- Discovers graphs from event log data
- Discover relationships between actors within event data
- Monitor event logs for deviations from graph

"Process mining is the missing link between model-based process analysis and data-oriented analysis techniques."

The Pioneer: Wil van der Aalst

Origins & Background:

- Eindhoven University of Technology (Netherlands)
- Founded process mining discipline in late 1990s/early 2000s
- Created PM4PY and ProM frameworks

Key Insight:

- Organizations collect massive event logs
- Traditional analysis misses hidden patterns
- Automated discovery reveals actual workflows

Original Motivation: Understanding Reality vs. Models

The Problem:

- Process models often outdated or idealized
- Gap between "how we think it works" vs. "how it actually works"
- Manual analysis too time-consuming and error-prone
- *Example:* Engineers manually tracing telemetry after failures

The Vision:

- **Process Discovery:** Automatically extract models from event logs
- **Conformance Checking:** Compare reality vs. intended process
- **Enhancement:** Identify bottlenecks and predict failures

Originally business processes, now applicable to any event-driven system

Process Discovery: The Alpha Miner Algorithm

How It Works:

- Van der Aalst's foundational algorithm (2004)
- Analyzes **temporal ordering** of activities in event traces
- Constructs Directly-Follows Graph (DFG) by correlating events

Event Trace Example:

Case 1: `[Created → Assigned → Started → Resolved → Closed]`
Case 2: `[Created → Assigned → Started → Escalated → Resolved → Closed]`

The Algorithm Identifies:

- **Direct succession:** If activity B follows A, create edge A $\to$ B
- **Parallelism:** Activities that occur in any order
- **Choice:** Alternative paths (escalated vs. direct resolution)
- **Transition counts:** Frequency of each path

Demo: IT Ticket Process Mining

Use Case: Converting semi-structured events into a system process diagram

Dataset:

- Synthetic IT helpdesk tickets
- Multiple departments (Finance, Engineering, HR, Operations, Sales)
- Various categories (Software, Hardware, Network, Access, Security)
- 52 tickets with complete lifecycle events

Tools:

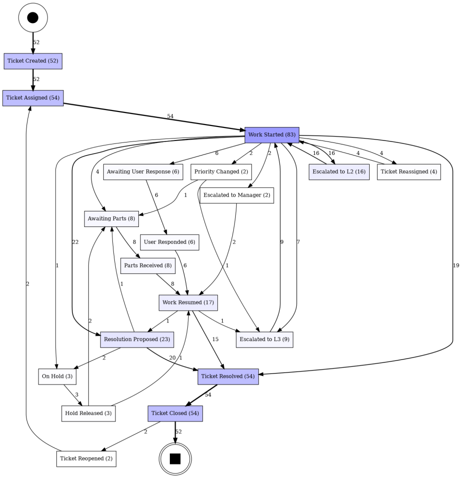
- PM4PY (Python process mining library)
- Pandas for data manipulation

This demo validates methodology for vehicle telemetry failure detection

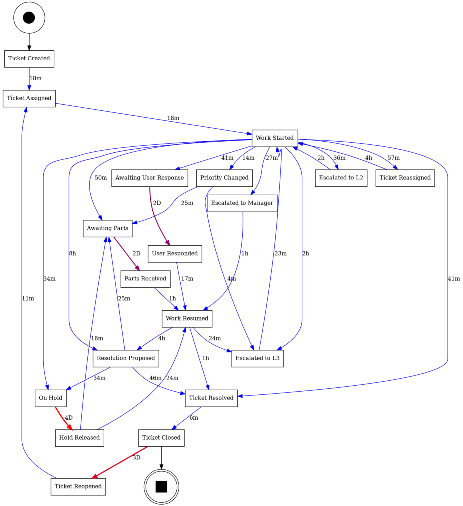
Demo Workflow

1. **Data Preparation**
   - Parse CSV logs into PM4PY event log format
   - Map: case_id (ticket_id), activity, timestamp
2. **Process Discovery**
   - Generate Directly-Follows Graph (DFG)
   - Performance DFG with time annotations
   - Markov chain visualization of variant flows
3. **Variant Analysis**
   - Identify nominal vs. alternate paths
   - 20 unique process variants discovered
   - Top variant: 28.85% (standard resolution path)

Process Visualizations



Directly-Follows Graph showing transition probabilities between ticket states



Performance DFG with average timing between events (in seconds)

Key Discoveries from the Analysis

Process Variants:

- **Happy Path (28.85%):** Created → Assigned → Started → Resolved → Closed
- **User Interaction (11.54%):** Includes "Awaiting User Response" cycle
- **Escalation Path (11.54%):** L2 escalation with additional work

Performance Metrics:

- Average wasted time: **10.81 hours per ticket**
- Flow rate (efficiency): **1.92%**
- High priority tickets: 55.6k seconds avg wasted time

Organizational Role Discovery

Technician Performance Analysis:

- **Most Efficient:** Sam Williams (5,023s avg wasted time)
- **Highest Workload:** Alex Martinez (59,326s avg wasted time)
- Discovered role-based patterns using PM4PY organizational mining

Category-Based Insights:

- Security tickets: Most efficient (16,702s avg)
- Hardware tickets: Longest delays (55,386s avg)
- Software tickets: Highest variety (16 tickets)