QUALITY ENGINEERING AND LEAN SIX SIGMA

High Defect Rate in Grade Calculation Module

Submitted by: Group 4
2024046 - Shameek Borkar
2024057 - Swasti Mishra
2024227 - Sanchu P. Thomas
2024261 - Dhruv Bhagat

OVERVIEW

Context

An IT services team managing an Academic Portal System identified critical quality issues, particularly a high defect rate (8.5 defects/KLOC) in the Grade Calculation Module, affecting student records, faculty operations, and institutional credibility.

Goal

Reduce defect density by 60% (from 8.5 to <3.5 defects/KLOC.

Scope

Covers the end-to-end grade calculation process from faculty submission through system computation, review/approval, to final publication; excludes policy changes, unrelated modules (attendance, reporting), and major architectural redesigns.

Approach

Apply DMAIC framework combined with Lean tools to identify root causes, eliminate waste, standardise processes, and implement preventive controls across the grade calculation and approval workflow.

DEFINE

Internal and External Customers

Category	Customer	Example Roles	Key Expectations
Internal	Students	UG & PG students	Accurate grades, error-free results, and timely updates.
	Faculty	Course instructors, Class mentors, HoDs	Reliable grade computation, minimal manual corrections, transparent score breakdowns.
	Academic Administration	Registrar, exam cell staff	Smooth grade processing, no recalculations, and reduced complaint handling.
	IT Department	Developers, QA team, tech support	Stable grade-calculation engine with low defect rate and predictable performance.
	Examination Committee	Controllers, coordinators	Consistent results, compliance with academic rules, accurate grade mapping.
External	Parents / Guardians	Receive student grade reports	Transparent, trustworthy grade information and no discrepancies.
	Accrediting Bodies	University boards, NAAC, NBA	Accurate academic records, zero defects in grade outcomes, reliable audit trails.
	Employers / Recruiters	Hiring partners	Consistent and reliable grade data with no anomalies or correction histories.

DEFINE

VOC AND CTQ

Voice of Customer (VoC)	Critical to X (CTQ)	Primary Metric for Improvement
"Stop sending wrong grades to students"	Critical to Quality	Defect Density (<3.5 defects/KLOC)

Primary Focus: Defect Density reduction from 8.5 to <3.5 defects/KLOC

DEFINE

Primary and secondary metrics

PRIMARY METRICS

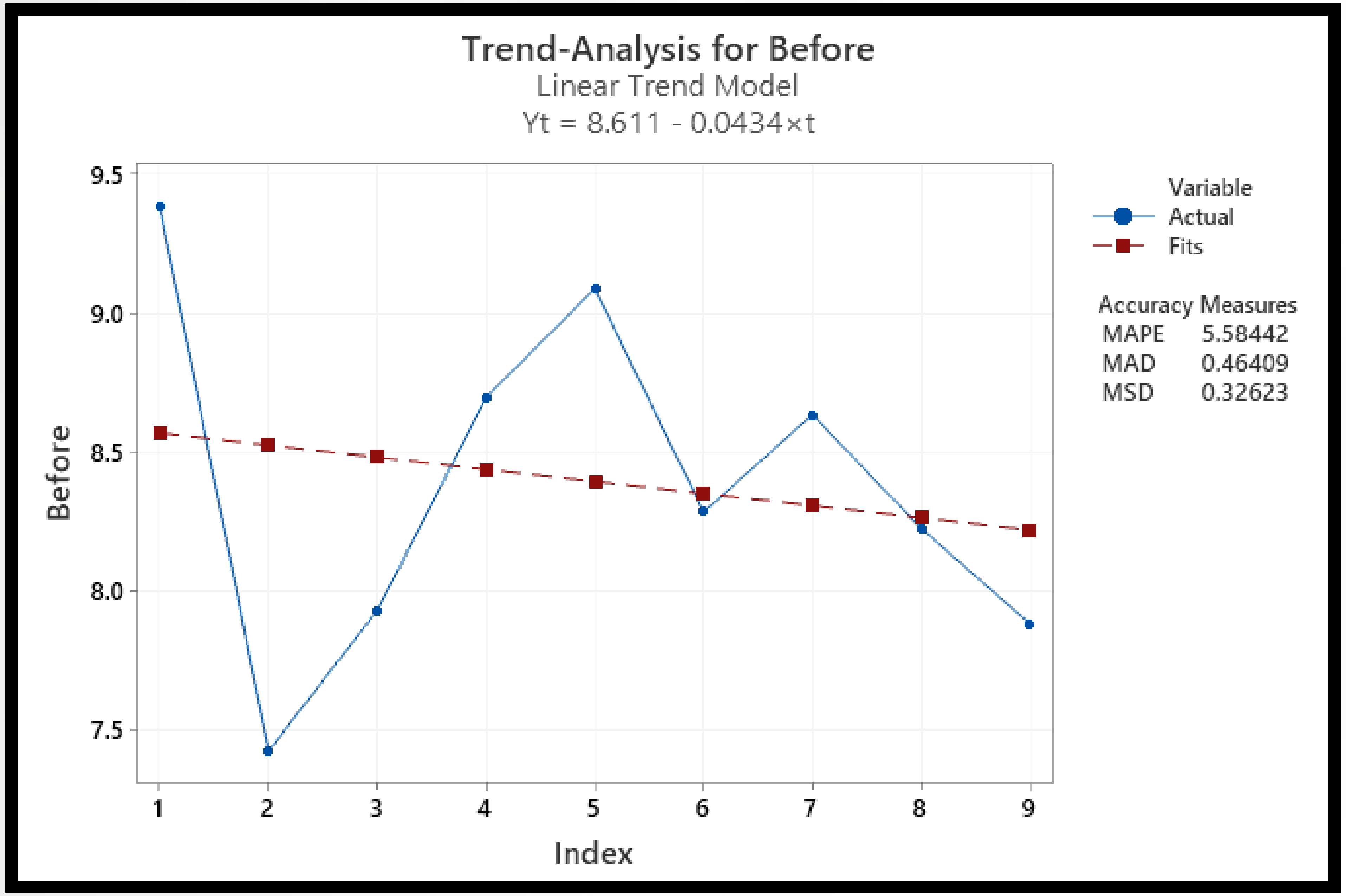
Metric	Definition	Reason
Defect Density	Number of defects per KLOC (thousand lines of code) or per function point.	Directly quantifies defect reduction.

SECONDARY METRICS

Metric	Definition	Purpose
Post-Release Defects (%)	% of defects found after deployment.	Evaluates final product reliability.

DEFINE

Baseline performance of primary metric (Over last 9 months Trend Chart)



Month	Metric Performance (%)
January	9.38
February	7.43
March	7.93
April	8.70
May	9.09
June	8.28
July	8.63
August	8.23
September	7.88

Inference :
The metric performance shows significant month-to-month fluctuation (ranging from 0.9% to 6.1% over 9 months with an average of 3.2%), indicating the process is out of control with no consistent pattern. This high variability suggests the presence of special cause variations that must be identified and eliminated before sustainable improvement can be achieved.

DEFINE

Project Charter

PROJECT INFORMATION

Field	Details
Project ID	LSS-2025-AC-001
Project Date	Nov 15, 2025
Organization	Academic Portal System - IT Services
Department	Quality Assurance / Testing
Project Duration	4.5 months (Nov 2025 - Mar 2026)

PROBLEM STATEMENT

The Grade Calculation Module shows an average defect density of 8.5 defects/KLOC against the 3.0 defects/KLOC standard, resulting in incorrect grade outputs, 20% increased rework, release delays, and monthly performance variability ranging from 0.9% to 6.1%, indicating an unstable and inefficient process.

DEFINE

GOAL STATEMENT

Reduce the defect density in the Grade Calculation Module by 60% (from 8.5 defects/KLOC to below 3.5 defects/KLOC) by March 2026.

BUSINESS CASE

The Grade Calculation Module is critical for ensuring accurate academic results. Frequent defects have caused significant rework, extended testing cycles, and reduced customer confidence.

Expected Benefits:

- Reduction in rework costs and improved efficiency
- Faster release cycles and better customer experience
- Increased software reliability and trust in automated grading
- Enhanced institutional credibility and compliance
- Reduced helpdesk complaints and manual interventions

Strategic Alignment: This improvement directly supports the organization's strategic initiative of achieving "First-Time-Right" software releases across all academic solutions.

In Scope:

- Analysis of all defects reported in the Grade Calculation Module
- Identification of root causes related to design, coding, and testing
- Implementation of preventive measures such as peer reviews, automated test coverage, and improved documentation
- Training of developers and testers on best practices

Out of Scope:

- Modifications to other modules such as Attendance or Report Generation
- UI/UX improvements not related to grade logic
- Large-scale architectural changes

DEFINE

Team

Role	Department	Responsibility
Project Sponsor	Product Engineering	Approves project, provides strategic direction
Project Leader	Quality Improvement	Leads execution, monitors progress and results
Process Analyst	Process Excellence	Performs data collection, defect trend analysis
Development Lead	Software Development	Implements code-level improvements and preventive controls
Testing Lead	Quality Assurance	Enhances test coverage and defect detection
Business Analyst	Product Management	Ensures process alignment with business requirements

Schedule

Phase	Timeline	Key Deliverables
Define Phase	15 Nov – 30 Nov 2025	Project charter approval, VOC and metric identification
Measure Phase	1 Dec – 20 Dec 2025	Baseline data, defect classification, trend analysis
Analyze Phase	21 Dec 2025 - 15 Jan 2026	Root cause identification and validation
Improve Phase	16 Jan – 28 Feb 2026	Implementation of corrective and preventive actions
Control Phase	1 Mar – 31 Mar 2026	Monitoring, control plan documentation, and handover

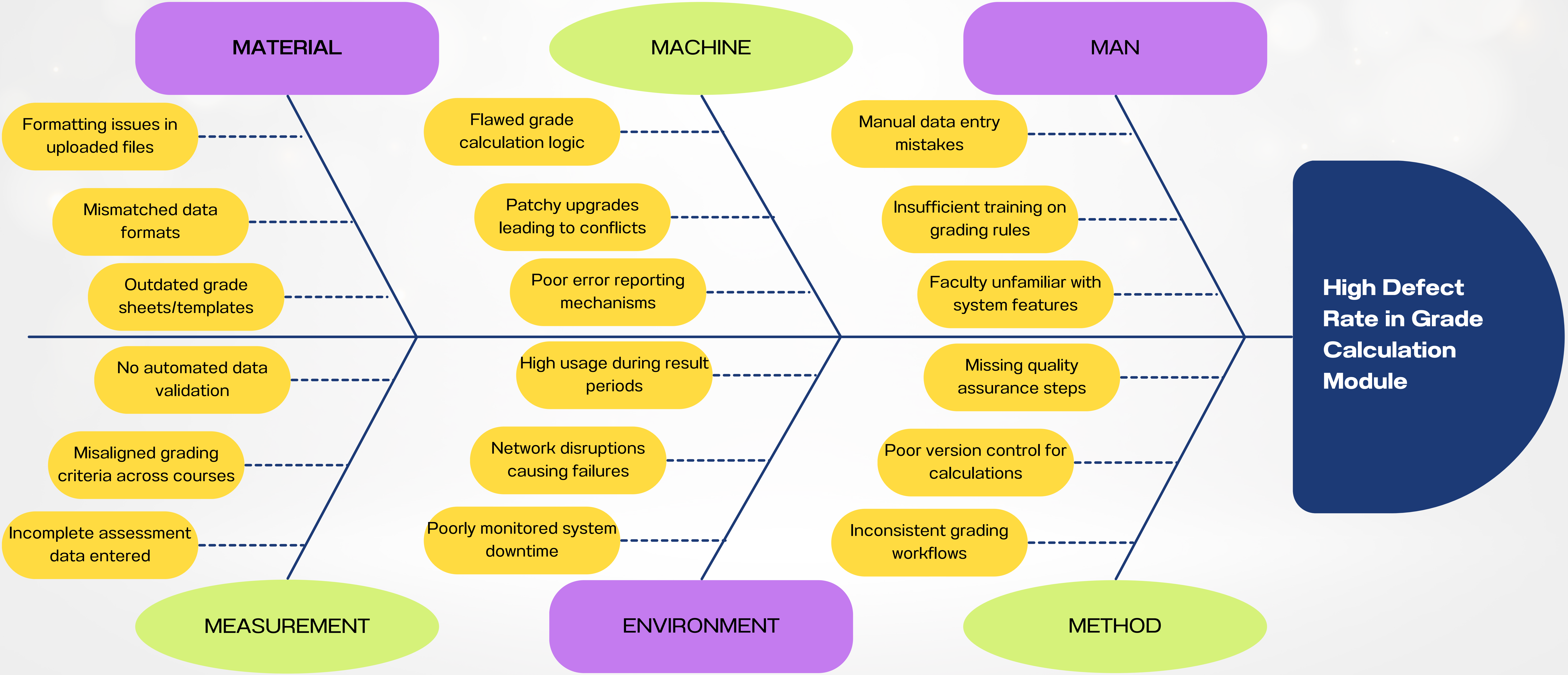
MEASURE

SIPOC Analysis

Suppliers	Inputs	Process	Outputs	Customers
Faculty	Student grades	Grades entered into the system	Grade reports	Students
ERP/System Admin	Assessment data (assignments, exams)	System executes grade calculation logic	Final grade logs	Faculty
IT Support Team	Weightage and grading rules	System validation and checks	Academic transcripts	Academic Office
Academic Office	Student enrolment data	Error detection and exception handling	Error/defect reports	System Admin/IT Support
Examination Department	System parameters (semester, curriculum rules)	Generate output for student and faculty access	Verified grade sheets	Parents & External Stakeholders

MEASURE

Problem Solving Fishbone Analysis



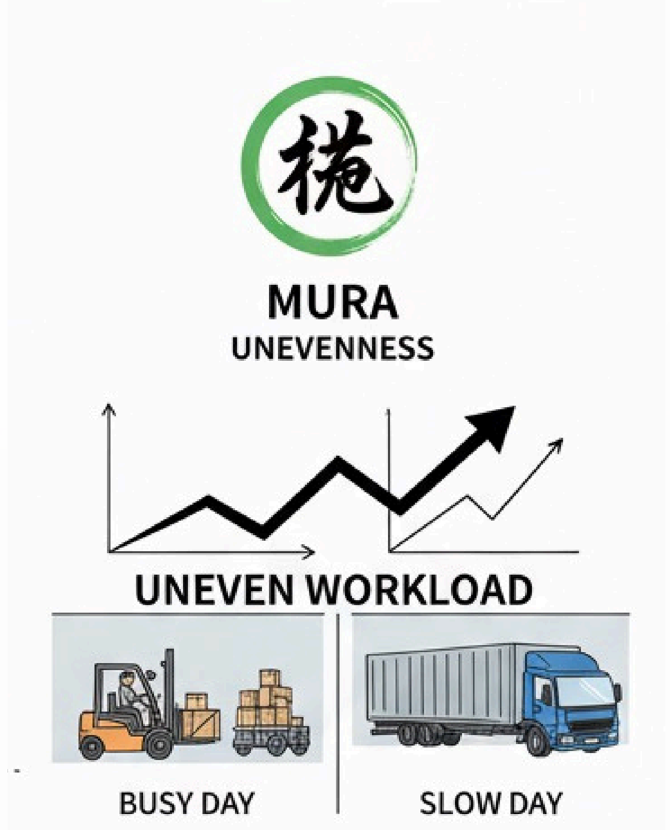
MEASURE

Identified Causes

COMMON CAUSES	SPECIAL CAUSES
Inadequate training on system usage	Patches or upgrades causing system conflicts
Manual data entry mistake	Sudden system crash during result processing
Incorrect weightage application	Incorrect file formats uploaded causing logic failures
Flawed grade calculation logic	Unauthorized manual override of grade
No standardized error-check protocol	Database corruption or synchronization issues
Complex grade calculation workflows	Exceptional user errors under unusual stress

MEASURE

Toyota 3M Model



- Manual re-entry of grades due to system errors
- Excess time spent fixing calculation defects
- Rework caused by wrong grade sheet formats
- Duplicate verification efforts for inaccurate output
- Time wasted resolving student complaints and escalations

- Inconsistent grading templates across departments
- Variable input quality due to different assessment formats
- Fluctuating system performance during peak result periods
- Irregular verification procedures for grade accuracy
- Uneven faculty usage of the system due to lack of training

- Overburdening IT staff during result publishing rush
- Faculty forced to manually calculate grades due to system trust issues
- System servers overloaded during deadline crunches
- Emergency manual override to meet publication deadlines
- Excessive dependency on one or two super-users

MEASURE

Types of Waste

DEFECTS

Wrong grade outputs due to misapplied weightage

OVERPRODUCTION

Multiple backups of the same grade file due to poor version control

WAITING

Waiting for the system to recalculate grades after a bug fix

UNUSED TALENT

Faculty forced to do data entry instead

TRANSPORTATION

Transferring grade sheets between systems manually

INVENTORY

Storing old grade versions due to lack of version control

MOTION

Faculty switching various systems to reconcile grades

EXTRA PROCESSING

Transferring grade sheets between systems manually

MEASURE

Action Plan

Problem Identified (Gemba Observation)	Cause Type	Lean Tool	Low-Hanging Fruit Action	Expected Benefit
Faculty manually correcting grade outputs	Defect, Overprocessing (Muda)	**Poka-yoke (Mistake-proofing)**	Add system-level validation rules	Fewer manual interventions, faster processing
Load spikes during result declaration	Muri	**Workload balancing / SMED**	Schedule staggered batch processing	Reduced load on servers, fewer crashes
Different grade formats across departments	Mura	**Standard Work / 5S**	Create standardized grade templates	Consistent input reduces errors
IT staff constantly fixing last-minute issues	Underutilized talent, Muri	**Kaizen / Visual management**	Map and automate error-prone steps	More time for improvement/automation projects
Version confusion in spreadsheets	Defect, Waiting (Muda)	**Version control / Kanban**	Use shared internal system for live updates	Real-time status tracking, no duplication
Unnecessary manual review of correct results	Overprocessing (Muda)	**Value Stream Mapping**	Identify and eliminate unnecessary review steps	Saves time and resources
Faculty lack training on system updates	Muri, Mura	**Training Matrix / Standard Work**	Conduct quick refresher training sessions	Reduced stress and errors, improved adoption

MEASURE

Cause and Effect Matrix

Potential Root Causes	Accuracy	Timely	Rework	Complaints	Stability	Total Score	Priority
Manual data entry mistakes	9	3	9	9	0	**261**	1
Flawed grade calculation logic	9	3	9	9	0	**261**	2
Incomplete assessment data entered	9	3	9	9	0	**261**	3
No automated data validation in system	9	3	9	9	0	**261**	4
Lack of ownership for data validation	9	1	9	9	0	**243**	5
Incorrect weightage configuration applied	9	1	9	9	0	**243**	6
No standardized error-check / verification protocol	9	3	9	3	0	**213**	7
System overload during result periods	1	9	3	3	9	**202**	8
Integration issues with SIS (student info system)	9	3	3	3	3	**186**	9
Network disruptions during grade processing	0	9	1	3	9	**176**	10
Patchy upgrades causing system conflicts/bugs	3	3	3	3	9	**168**	11
Insufficient training on grading rules & system	9	3	3	3	0	**165**	12

MEASURE

Data Collection Plan

Metric	Definition	Data Type	Sampling Method	Data Source	Collection Frequency	Target Use
Defect Density (Primary)	Defects per KLOC in the Grade Calculation Module.	Attribute/ Count	100% inspection of all code releases.	Defect Management System (Jira/Azure DevOps).	Weekly (for new defects), Monthly (for KLOC calculation).	**Establish DPMO and calculate the Sigma level.**
Manual Data Entry Mistakes	Count of incorrect grade submissions or configuration errors by faculty/admin per grading cycle.	Attribute/ Count	Audit of grade logs against source data.	Audit Logs, Academic Administration Records.	Per Grading Cycle (e.g., Mid-term, Final).	**Validate Manual Data Entry Mistakes as a critical root cause.**
Flawed Logic Defects	Count of defects traceable to incorrect formulas, weightage, or rounding rules.	Attribute/ Count	Defect Triage & Root Cause Analysis Logs.	Defect Management System.	Weekly, during defect resolution.	**Validate Flawed Grade Calculation Logic as a critical root cause.**
Incomplete Assessment Data	Count of missing or incomplete data required for final grade computation.	Attribute/ Count	System Pre-processing Error Logs.	Grade Module Log Files.	Per Grading Cycle.	**Validate Incomplete Assessment Data as a critical root cause.**
Rework Effort (Supporting)	Hours spent by faculty/staff correcting grade errors.	Continuous/ Time	Time tracking logs for rework activities.	Timesheets/ Project Management Tool.	Monthly.	**Quantify the Cost of Poor Quality (COPQ).**
Grade Calculation Accuracy Rate	% of grades calculated correctly without manual intervention	Continuous/ Ratio	Final Grade Verification Logs.	Academic Administration Records.	Per Grading Cycle.	**Measure the direct business outcome of the the process.**

MEASURE

Check for Special Causes and Normal Distribution

The historical data for Metric Performance (%) is X = {9.3, 7.4, 7.9, 8.6, 9.1, 8.2, 8.6, 8.2, 7.8} with n = 9 observations.

Check for Special Causes

Analysis Method: Trend/Run Analysis of the 9 data points

Result / Observation

- The values fluctuate between 7.43% and 9.38%, a total spread of 1.95, which is much smaller than the earlier dataset (spread 5.2).
- There are no sudden jumps or drops in performance.
- The points seem to vary normally around the average.

Check for Normal Distribution

- **Mean (μ):** 8.39%
- **Median:** 8.28%
- **Standard Deviation (σ):** 0.58
- **Range (Max - Min):** 1.95

The relationship between the mean and median is used as a preliminary indicator:

Result: The Mean (8.39%) is slightly greater than the Median (8.29%).

Conclusion: The new data shows lower variability and less indication of instability.

However, the slight right skew indicates that the data are not perfectly normal, so capability assumptions should still be applied cautiously.

MEASURE

C_p, C_{pK} for the Before Improvement data

The historical data for Metric Performance (%) is X = {9.3, 7.4, 7.9, 8.6, 9.1, 8.2, 8.6, 8.2, 7.8} with n = 9 observations.

Data and Parameters

- **Mean (μ):** 8.39%
- **Standard Deviation (σ):** 0.58
- **Lower Specification Limit (LSL):** 6.64% (Assumed minimum for Defect/Error Rate)
- **Upper Specification Limit (USL):** 10.14% (Assumed maximum tolerable error rate)

Process Potential Index (C_p)

$C_p = 0.95$

Conclusion: The process variation is **too wide** for the specifications (since $C_p < 1$).

Process Capability Index (C_{pk})

Capability Upper (C_{pu}) = 4.58

Capability Lower (C_{pl}) = -2.67

C_{pk} = Min(-2.67,4.58) = -2.67

Conclusion: The process is **not capable** ($C_{pk} < 1$) and is poorly centred, with the lower specification limit (LSL) being the primary constraint.

ANALYSE

Identifying the Critical Root Causes

- **Hypothesis:** At least one of the tested conditions results in a mean defect count statistically different from the baseline.
- **Metric:** Total Defects found in 1,000 Grade Calculation Cycles.

Simulated Grading Cycle	Baseline - Current Process	Fix for Manual Entry Only	Fix for Flawed Logic Only	Fix for Incomplete Data Only
1	22	16	12	15
2	25	18	10	14
3	24	17	13	16
4	26	19	11	14
5	23	15	14	17
Mean Defects	**24**	**17**	**12**	**15.2**

The comparison of means (X) clearly shows a difference:

- Flawed Logic Fix has the lowest mean defect count (X =12.0).
- Incomplete Data Fix is next lowest (X =15.2).
- Manual Entry Fix is next (X =17.0).

IMPROVE

Phased action Plan

Phase 1: Address Flawed Grade Calculation Logic (Most Critical)

Action	Description	Lean/Six Sigma Tool	Responsibility	Timeline
A1. Review and Standardise Logic	Conduct a 100% code review of all grade calculation formulas, rounding rules, and weightage application logic. Refactor code to align with academic regulations.	Standardised Work / Code Review	Development Lead, Process Analyst	2 weeks
A2. Implement Unit Test Coverage	Develop and implement comprehensive unit tests (covering edge cases, zero values, and rounding scenarios) with 95% coverage for the core calculation functions.	Defect Prevention Protocols	Testing Lead, Development Lead	3 weeks
A3. Introduce Poka-Yoke (Error-Proofing)	Create a Grade Calculation Audit Log that flags and documents any discrepancy between expected intermediate results and actual results before final storage.	Poka-Yoke	Development Lead	4 weeks

IMPROVE

Phased action Plan

Phase 2: Address Incomplete Assessment Data & Manual Entry

Action	Description	Lean/Six Sigma Tool	Responsibility	Timeline
B1. Enforce Mandatory Data Validation (MDV)	Implement system-level validation rules at the point of data entry (pre-processing checks) to prevent faculty/admin from saving incomplete or incorrectly formatted data.	Poka-Yoke / Jidoka (automation)	Development Lead	3 weeks
B2. Create Standardised Data Templates	Develop and enforce the use of a single, standardised template for assessment uploads across all departments to prevent formatting issues and mismatched data.	Standard Work / 5S (Sort/Set in Order)	Business Analyst, Academic Administration	1 week
B3. Enhance User Interface (UI) for Entry	Simplify the grade entry interface, using dropdowns and autofill where possible, to minimise the risk of Manual Data Entry Mistakes.	Kaizen / Workflow Optimisation	Development Lead, Testing Lead	2 weeks

IMPROVE

Phased action Plan

Phase 3: Control and Sustain

Action	Description	Lean/Six Sigma Tool	Responsibility	Timeline
C1. Conduct Targeted Training	Provide mandatory training to all faculty and administration on the new simplified grade entry interface and the standardised data templates.	Training Matrix	Process Analyst, Academic Administration	1 week
C2. Implement Control Charts	Establish Control Charts (P or U charts) to continuously monitor the post-implementation defect density and check for any recurrence of special causes.	Statistical Process Control	Process Analyst	Ongoing

IMPROVE

Choosing a dummy dataset (after-implementation metrics)

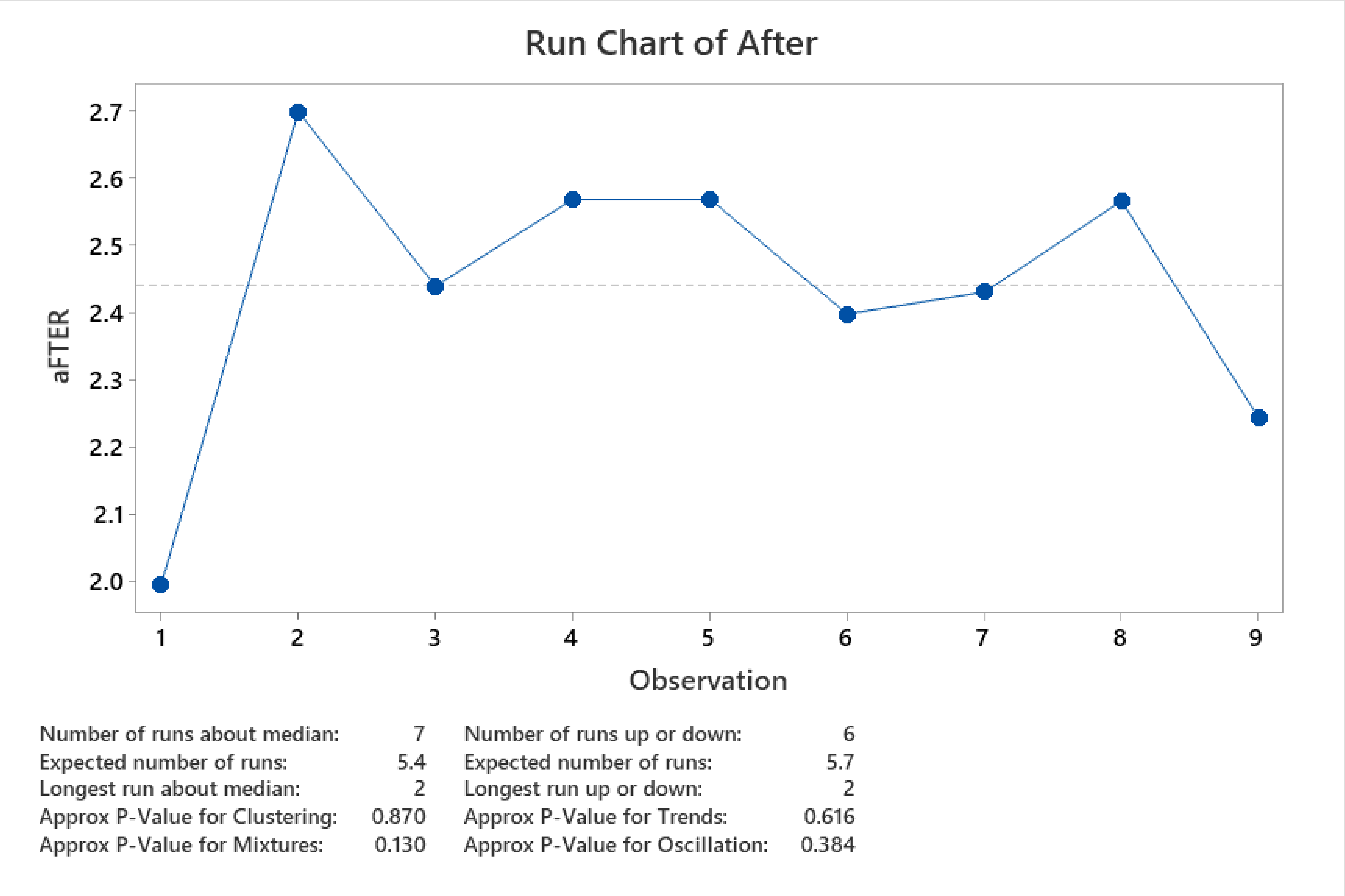
Month	Metric Performance (%)
January	2.00
February	2.70
March	2.44
April	2.57
May	2.57
June	2.40
July	2.43
August	2.57
September	2.24

Reasoning

- Using DMAIC, teams identified major defect drivers, removed root causes, and established controls—leading to a sustained drop in defects/KLOC.
- Standardized coding practices, review checklists, and quality gates reduced variation in development output, directly lowering defect density.
- This data-driven decision-making led to targeted improvements and a measurable reduction in defects/KLOC.

IMPROVE

To understand a Run-Chart (to Understand Special Causes)



Reasoning

The run chart shows the monthly after-performance values plotted over time, with most points fluctuating naturally around the median.

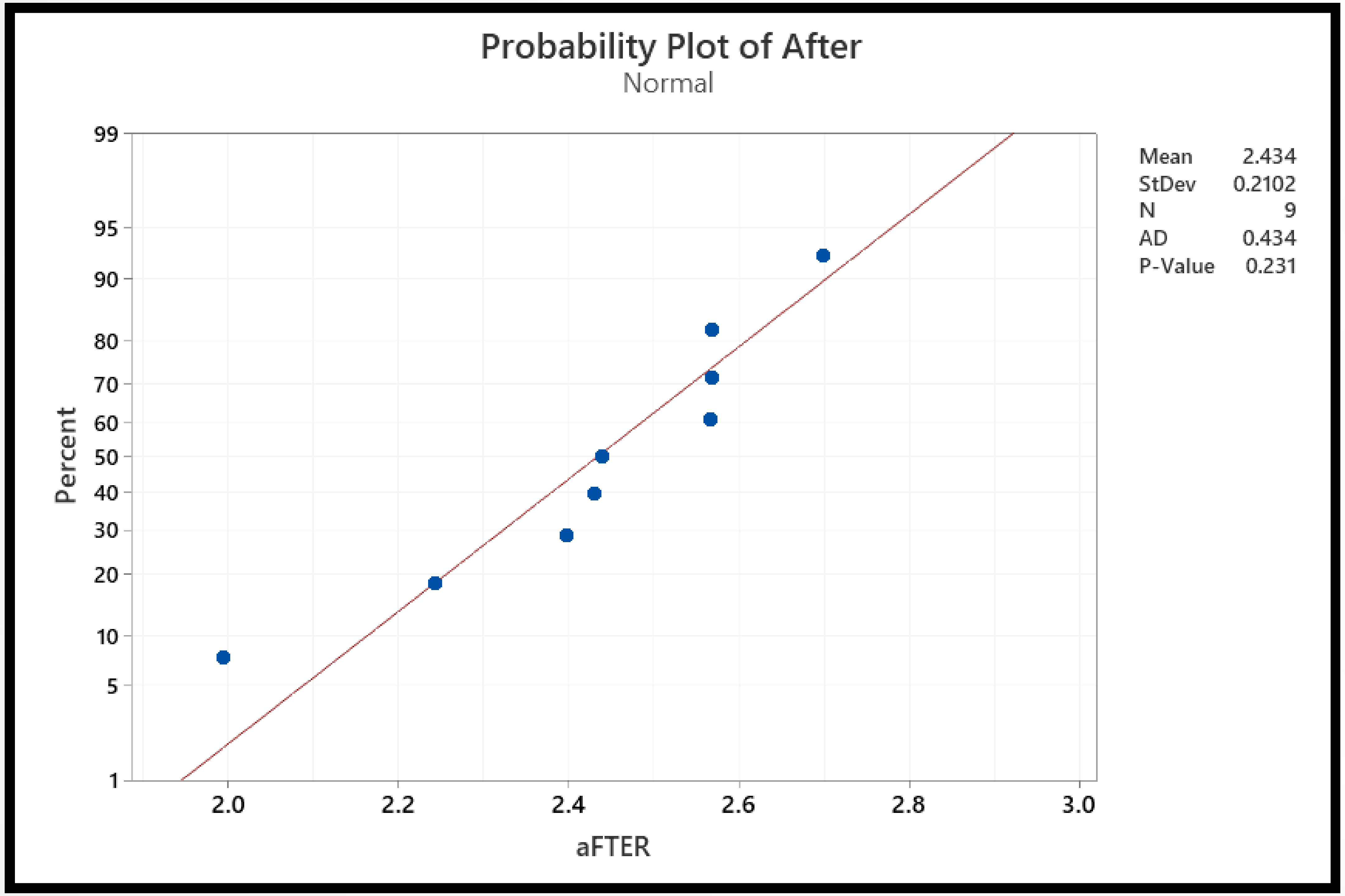
The number of runs is equal to the expected value, and all p-values are greater than 0.05, indicating no trends, shifts, or unusual patterns.

This means the process is stable and only influenced by common-cause variation.

Overall, the run chart confirms that the improved process is consistent and under control.

IMPROVE

To understand whether the data is normally distributed



Reasoning

Hypothesis taken

H_0 (Null Hypothesis):

The data follows a normal distribution.

H_1 (Alternative Hypothesis):

The data does not follow a normal distribution.

Observations:

The points fall almost exactly on the reference straight line, indicating that the after-performance data follows a normal distribution.

With a high p-value of 0.231, the plot confirms that the dataset is normally distributed

IMPROVE

Checking the Significant Difference (Before & After Improvement)

Descriptive Statistics

Sample	N	Mean	StDev	SE Mean
Before	9	8.395	0.617	0.21
After	9	2.434	0.210	0.070

Estimation for Difference

Difference	95% CI for Difference
5.960	(5.468, 6.452)

Test

Null hypothesis $H_0: \mu_1 - \mu_2 = 0$
Alternative hypothesis $H_1: \mu_1 - \mu_2 \neq 0$

T-Value	DF	P-Value
27.42	9	0.000

Reasoning

Hypothesis taken

H_0 (Null Hypothesis):

There is no difference in mean performance before and after the improvement (μ1=μ2)

H_1 (Alternative Hypothesis):

There is a difference in mean performance before and after the improvement (μ1 ≠ μ2)

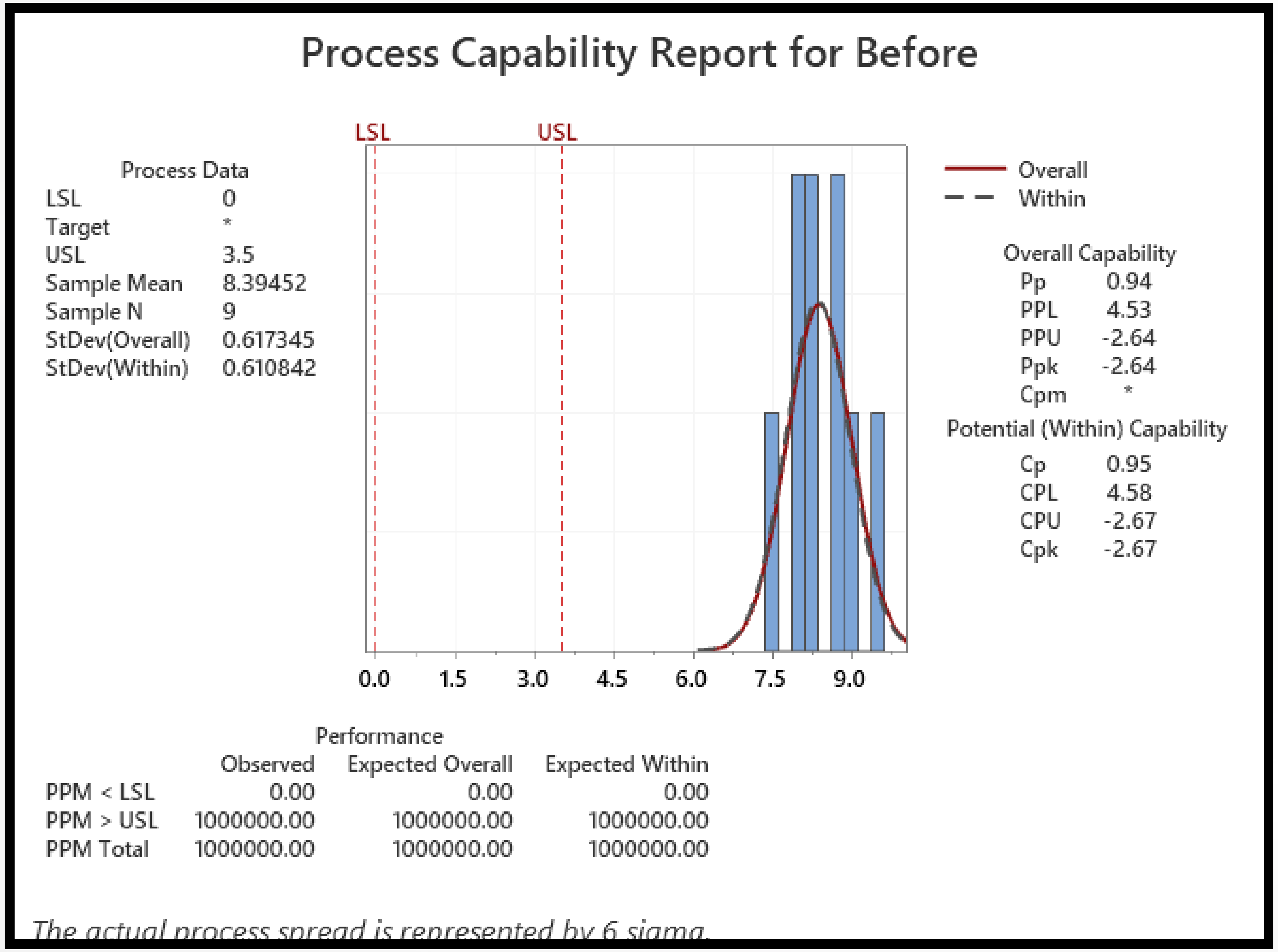
Observations:

The two-sample t-test produced a **p-value of 0.000**, which is less than the significance level of 0.05.

Therefore, we reject the null hypothesis and conclude that there is statistically significant difference between the before and after performance means.

IMPROVE

Process Capability Comparison (Before)



Reasoning

The process exhibits moderate variation (StDev ≈ 0.61 overall), which still creates a noticeable spread of values.

- The capability indices — Cp = 0.95 and Cpk = −2.67 (within capability) are below the desired threshold of 1.0. The negative Cpk value especially highlights that the mean is far beyond the upper specification limit

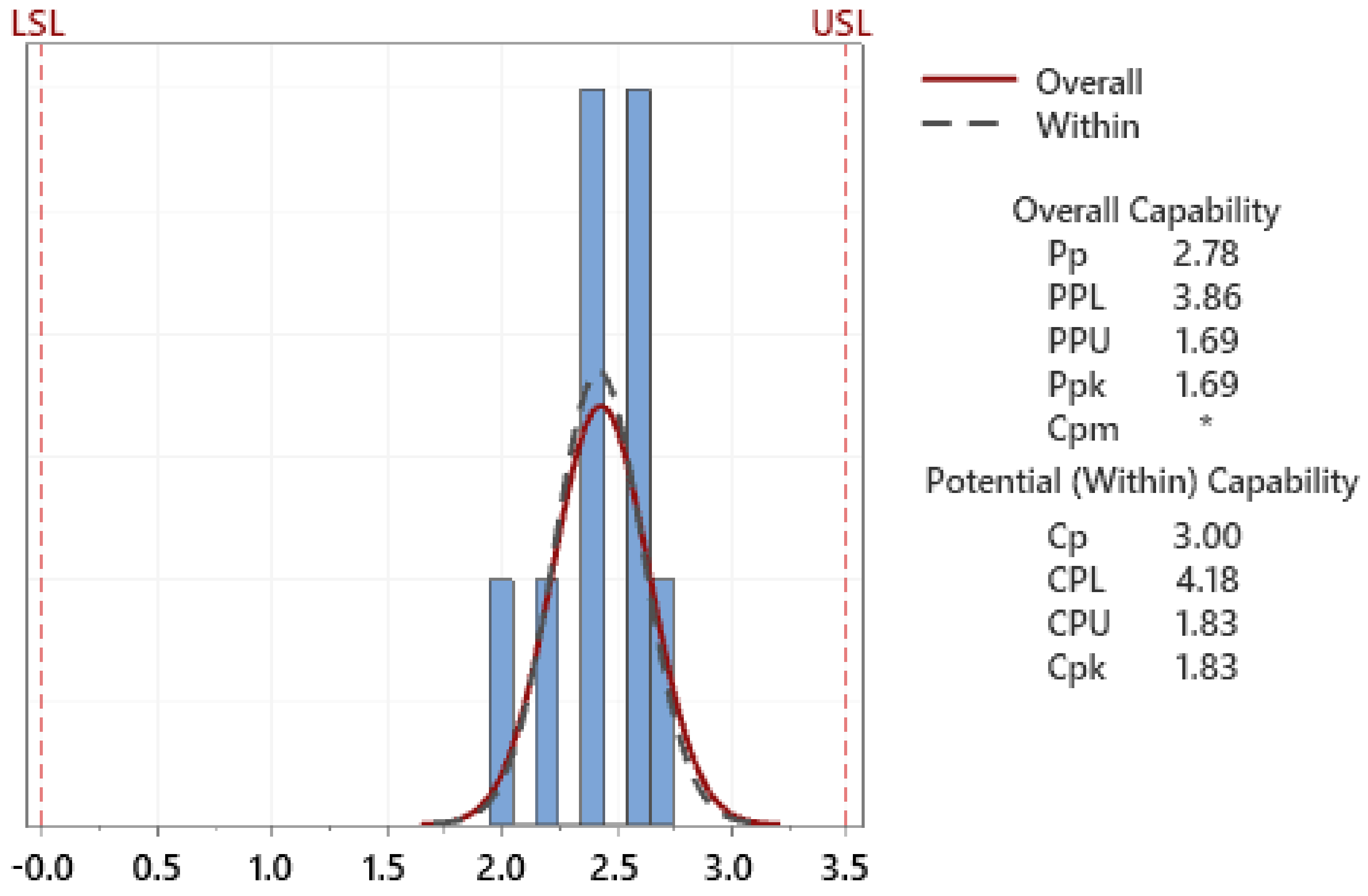
- The distribution is not centered, practically all produced values fall beyond the upper specification, leading to extremely high defect rates

IMPROVE

Process Capability Comparison (After)

Process Capability Report for After

Process Data	
LSL	0
Target	*
USL	3.5
Sample Mean	2.43447
Sample N	9
StDev(Overall)	0.210205
StDev(Within)	0.194182



Overall Capability	
Pp	2.78
PPL	3.86
PPU	1.69
Ppk	1.69
Cpm	*

Potential (Within) Capability	
Cp	3.00
CPL	4.18
CPU	1.83
Cpk	1.83

Performance

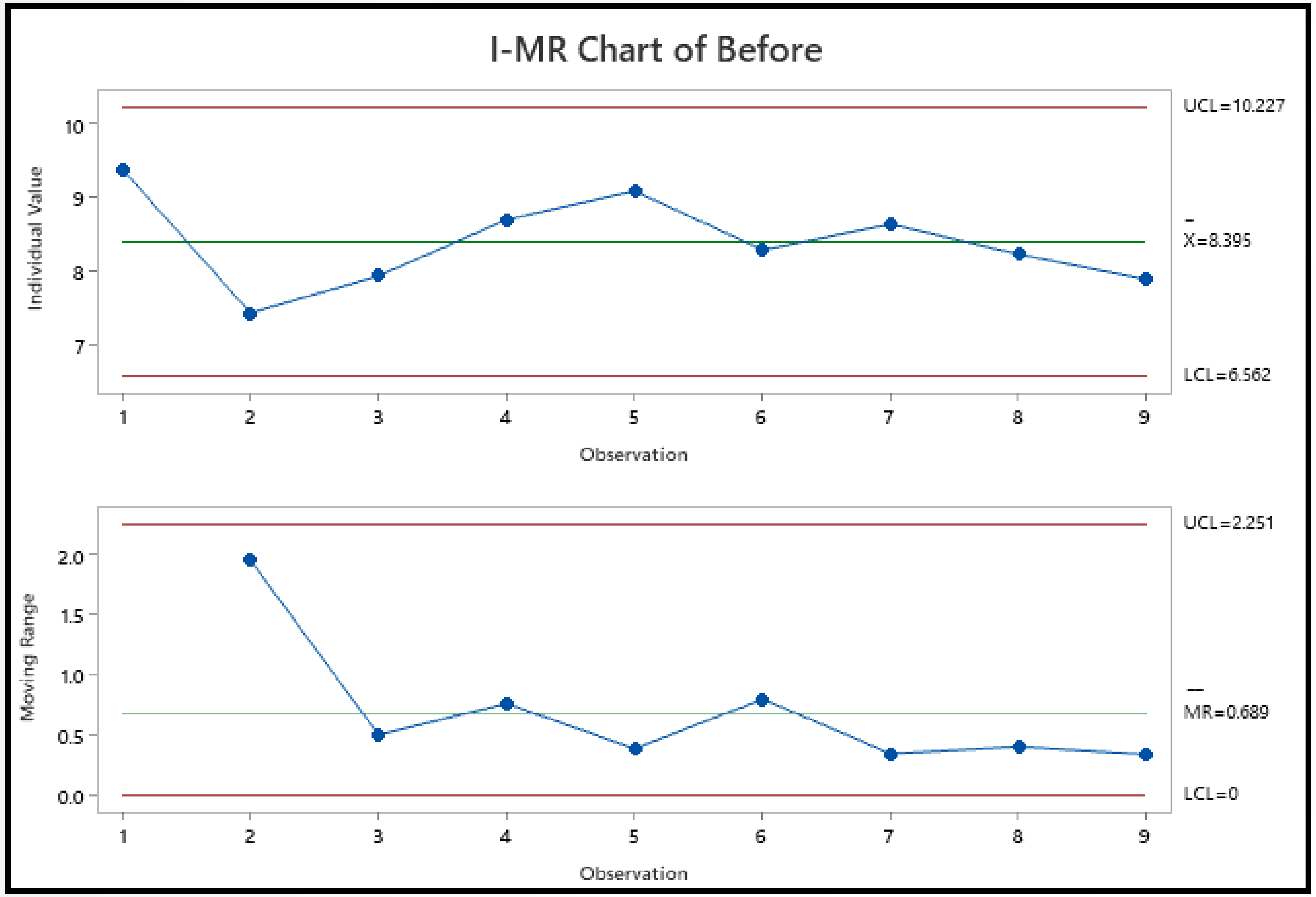
	Observed	Expected Overall	Expected Within
PPM < LSL	0.00	0.00	0.00
PPM > USL	0.00	0.20	0.02
PPM Total	0.00	0.20	0.02

Reasoning

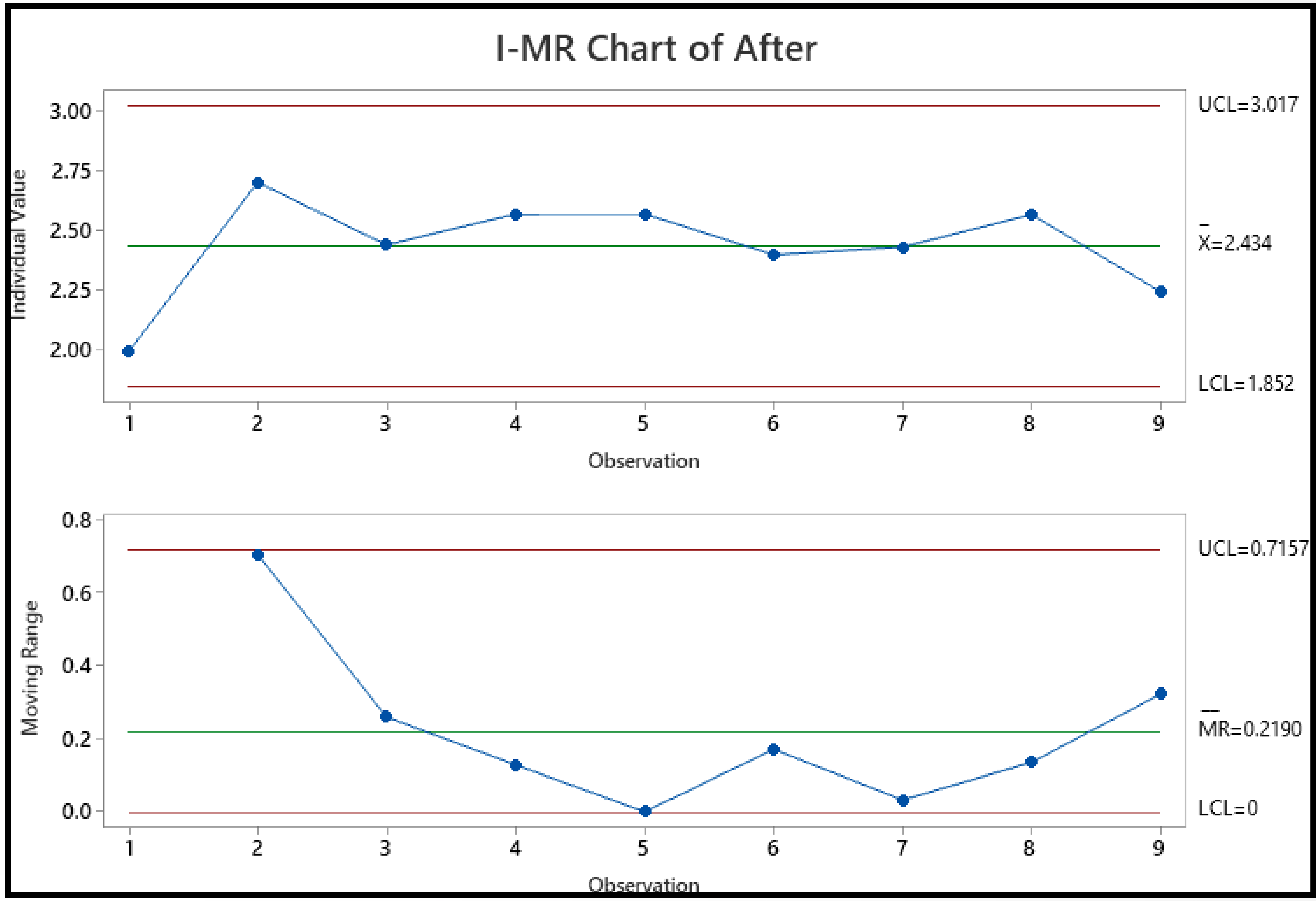
- With Cp = 3.00 and Cpk = 1.83, the process is now well within the specification limits and demonstrates strong capability. The high Cp reflects reduced variation relative to the spec width, while the elevated Cpk confirms that the process mean is centered adequately within the acceptable range.
- The process variability has reduced significantly, with a very low overall standard deviation (≈ 0.21).
- The after-performance data is tightly clustered around the process mean (≈ 2.43), and the fitted normal curve closely matches the histogram.

CONTROL

Control Chart (Before)



Control Chart (After)



- The Moving Range (MR) Chart before improvement shows frequent spikes and higher ranges (MR ≈ 0.689), while the after chart shows very small and uniform ranges (MR ≈ 0.2190), confirming a major reduction in short-term variability.
- No points are outside the control limits in either chart, but the After chart clearly demonstrates lower variation, improved centering, and overall better process control compared to the Before chart.

CONTROL

5S and poka yoke mechanism

5S Implementation

5S Pillar	Application in Grade Calculation Module
Sort (Seiri)	Remove obsolete test cases, redundant code, and unused calculation formulas from the system
Set in Order (Seiton)	Organize test scripts, documentation, and code repositories in standardized folders with clear naming conventions
Shine (Seiso)	Conduct regular code reviews and clean up technical debt; maintain updated documentation
Standardize (Seiketsu)	Create standard templates for grade calculation logic, defect reporting, and testing procedures
Sustain (Shitsuke)	Implement periodic audits, training refreshers, and compliance checks to maintain standards

CONTROL

5S and poka yoke mechanism

Poka Yoke (Error-Proofing) Mechanisms

Error Type	Poka Yoke Mechanism	Implementation
Grade Calculation Errors	Input Validation	Implement automated checks to validate input data ranges, formula parameters, and weightage totals before calculation
Formula Configuration Errors	Template Lock-In	Use pre-approved, locked formula templates that prevent unauthorized modifications
Data Synchronization Failures	Automated Alerts (Jidoka)	Real-time alerts when data mismatch is detected between modules
Deployment Errors	Pre-Deployment Checklist	Mandatory automated checklist verification before code deployment to production
Manual Override Risks	Role-Based Access Control	Restrict manual grade editing to authorized personnel with audit trails
Version Mismatch	Version Control Gates	Automated version compatibility checks before release

CONTROL

FMEA ANALYSIS

Failure Mode	Potential Effect	Severity (S)	Potential Cause	Occurrence (O)	Detection (D)	RPN	Recommended Actions
Incorrect grade formula	Wrong grades published	9	Formula configuration error	5	4	180	Implement validation tool; use locked templates
Database sync failure	Records mismatch	8	Network timeout	6	5	240	Add retry logic; real-time monitoring
System crash (peak load)	Delayed publication	8	Insufficient capacity	7	6	**336**	**Capacity planning; auto-scaling; stress testing**
Rounding errors	Minor discrepancies	5	Inconsistent logic	6	7	210	Standardize rounding rules; unit tests
Unauthorized modification	Integrity breach	10	Weak access controls	3	4	120	Strengthen RBAC; 2FA; alerts
Version mismatch	New bugs introduced	7	Manual deployment	5	6	210	Automate CI/CD; version gates
Missing test coverage	Defects escape	8	Incomplete scenarios	6	7	**336**	**Expand test library to ≥95% coverage**

CONTROL

CONTROL PLAN

Process Step	CTQ	Target	Control Method	Frequency	Reaction Plan	Owner
Formula Configuration	Formula Accuracy	100%	Pre-approved templates (Poka Yoke)	Per change	Reject & review	Dev Lead
Code Deployment	Defect Density	<3.5/KLOC	Control Charts (X-bar, R)	Per release	RCA if out of control	Test Lead
Grade Calculation	Accuracy	99%	Automated test suite	100%	Alert & manual verification	QA Team
Data Sync	Sync Accuracy	100%	Automated alerts (Jidoka)	Real-time	Auto-retry; escalate if failed	IT Team
Test Coverage	Code Coverage	≥95%	Coverage tracking tool	Per sprint	Expand tests if <95%	Test Lead
System Performance	Response Time	<3 sec	Performance dashboard	Continuous	Optimize if >3 sec	Dev Team
Post-Release Defects	Defect Leakage	<5%	Trend analysis	Monthly	RCA if >5%	PM
Customer Satisfaction	SPEI	≥90%	User surveys	Quarterly	VOC analysis if <90%	BA

CONTROL

CONTROL PLAN

Key Control Mechanisms

- Revised Controls for Defect/KLOC Monitoring
- Statistical Process Control: Control charts tracking Defect/KLOC trends across releases
- Visual Management: Real-time dashboard showing Defect Density (Defect/KLOC), SPEI, and uptime
- Audits: Monthly code quality and defect density compliance checks
- Training: Quarterly refresher sessions on secure coding, defect prevention, and quality standards

Escalation Triggers

Condition	Action	Escalation To
Defect density >5/KLOC	Immediate RCA	Project Leader

THANK YOU