Journal of Information Systems Engineering and Management
2024, 9(4s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Real-Time Delivery Control for ICT Programs: A Telemetry-Driven Framework for Cost, Schedule, and Risk Optimization in Enterprise Project Execution

Asadullah Saif Mohammed
Senior Technical Program Manager
Richmond, Texas, USA
Email: mohammedasaif@gmail.com

ARTICLE INFO **ABSTRACT**

Received: 10 Oct 2024
Revised: 22 Nov 2024
Accepted: 28 Dec 2024

High schedule overruns, cost deviations, and scope failure rates persist in information and communication technology (ICT) programs of business enterprises, with industry analyses showing that fewer than one-third of large-scale software projects are on time and on budget. These results are, in large part, avoidable because conventional project management paradigms are structurally limited: their reliance on periodic milestone reporting, fixed dashboards, and lagging performance indicators, which, by their very nature, are incapable of detecting meaningful sources of delivery risk in real time.

This paper presents the Telemetry-Driven Delivery Control Framework (TDCF), a broad, scaled-up approach to achieving sustainable visibility, anticipatory risk recognition, and responsive control across numerous complex enterprise ICT programs. The TDCF defines a five-layer architecture that includes: acquisition of telemetry signals, normalization of telemetry signals, mapping of composite performance indicators, a predictive analytics engine, and a role-stratified decision support interface. Each layer has a formal mathematical model that includes a signal confidence function, a composite Delivery Risk Index (DRI) determined by logistic regression using historical program outcome data, an ARIMA-based forecasting model, and a Monte Carlo-based estimator of delivery probability.

Empirical validation was conducted on three enterprise ICT programs, including cloud infrastructure migration, implementation of an enterprise SaaS platform, and transformation of organizational cybersecurity, with a combined budget of USD 26.7 million and 105 program professionals. Findings indicate a portfolio-average reduction in schedule deviation severity of 44.2, which is lower than the results reported by stakeholders in the bank, whose targets are set at a minimum of 2.6 on a five-point scale. All main results were also significant at $p < 0.05$. The framework can be applied to large-scale ICT programs such as cloud migration, SaaS implementation, cybersecurity transformation, and data platform modernization.

Keywords: deviations, Telemetry-Driven Delivery Control Framework (TDCF), Telemetry-Driven Delivery Control Framework (TDCF), implementation.

1. INTRODUCTION

ICT investment landscape. The global business enterprise landscape has experienced transformative growth over the last 20 years, with worldwide technology spending in business enterprises regularly exceeding three trillion US dollars and annual growth in the business enterprise user base ranging from 8 to 10 million worldwide (Kerzner, 2022; Schwalbe, 2019). And it is within this context that ICT programs have limited strategic importance but are nevertheless enterprise capability-building programs that form the basis of competitive advantage, compliance with regulatory mandates, and organizational resilience.

Although there is such a strategic uplift, empirical evidence continues to indicate that ICT program implementation remains a domain of persistent underperformance. According to the CHAOS Report of Standish Group (2020), only 31% of software and technology projects can be classified as successful; completed within the required time, within budget, and with the required functionality. 52% of them can be characterized as challenged, and 17% are outright failures. These patterns are systemic, since even 20 years of measurement did not yield any improvement beyond a 3.5 percentage-point decrease (Ika, 2009; Müller and Turner, 2007).

The causes and root causes of failures in ICT programs are well documented in the academic literature. According to Pinto and Slevin (1988), monitoring and the quality of feedback were among the best predictors of project success. Joslin and Muller (2015) have shown that the fit between the project management approach and the governance environment is a major predictor of success across organizational types. Another recurring theme is the ineffectiveness of existing monitoring and control practices: traditional project management models, including those codified in the PMBOK Guide (PMI, 2021), are generally comprehensive but were designed to operate in an environment characterized by periodic reporting cycles and relatively stable requirements.

Contemporary enterprise ICT programs, in turn, boast high technical sophistication, multi-vendor ecosystems, an iterative approach to development, continuous integration and deployment, and the ability to accelerate stakeholder requirements (Dingsoyr et al., 2012; Kim et al., 2016). Monthly or even weekly status updates used in such environments are not able to reflect the speed of change at the execution layer. Initial schedule variances, which start as small variances at the sprint/workstream level, grow exponentially into crises on a program-wide basis by the time they are reflected through archetypal reporting mechanisms. Cost overruns, caused either by misalignment of resources or by unexpected technical rework, are not only obscured but also accumulate over time and are only revealed by periodic financial reconciliation.

Advances in DevOps practices, continuous delivery pipelines, and software observability engineering have generated unprecedented volumes of execution-layer telemetry (Forsgren et al., 2018; Humble and Farley, 2010). Modern development platforms continuously produce information on build frequency, deployment success rates, test coverage, defect injection rates, and infrastructure performance. Project management platforms provide rich data sets on task velocity, backlog development, milestone compliance, and resource allocation. Financial systems offer fine-grained cost and burn-rate data. Although these data streams are available, they remain highly siloed within their respective tooling ecosystems and are inaccessible to program-level decision-making in any integrated or analytically meaningful form (Chatterjee, 2023; de Waal and Counet, 2009).

This paper addresses this fragmentation by introducing the Telemetry-Driven Delivery Control Framework (TDCF), which makes five key contributions: (1) a formally defined, five-layered telemetry integration architecture with mathematical specifications for each processing stage; (2) a structured taxonomy of twenty-two ICT delivery signals across five functional domains; (3) the Delivery Risk Index (DRI), a composite, forward-looking program health metric with empirically derived weights replacing logistic regression weights; (4) a protocol for implementing TDCF; and (5) empirical validation results from three large-scale enterprise ICT programs, showing quantifiable improvements in schedule behavior, cost predictability, and risk detection effectiveness.

2. RELATED WORK

The academic roots of the TDCF can be traced to six interreligious areas of literatures such as traditional ICT project management, earned value management, agile and DevOps approach methodologies, real time monitoring and telemetry, predictive analytics in program contexts and risk management frameworks. This review contextualizes the TDCF within these spheres, the important contributions of the model, as well as the research gaps that the model seeks to fill.

The main form of codification of traditional ICT project management knowledge is the project management knowledge bodies (PMBOK) of the Project Management Institute area, which now has its seventh edition (PMI, 2021). This trend in the development of the PMBOK, whereby in its initial stage, the process was the focus in both theory and practice, whilst in contemporary times, the outcome-oriented and principles-oriented framework is increasingly recognized as the benchmark in reflecting the essential role of process adherence in both theory and practice in the context of contemporary ICT programs. As Kerzner (2022) and Wysocki (2019) have demonstrated, past editions because of their high impact by both Waterfall and phase-gate delivery models provided well-structured governance mechanisms which relate to predictable engineering contexts and demonstrate high fitting to iterative, high-velocity ICT program contexts. Turner (2014) defined the essence of the problem of project management, as a balance between the competing constraints of performance, cost, time, and scope in an environment of inherent uncertainty, a balance that assumes periodicity in the execution of performance reviews, which in turn is becoming increasingly untenable in programs where execution velocities are far greater than reporting cadences. Morris (2013) postulated that the discipline of project management needed to be rebuilt around the realities of complex systems as opposed to the principle-based thinking that had been at the heart of idealized models of the planning process, and that which defined the design principles of the TDCF. Joslin and Müller (2015) took this argument to the next level, empirically demonstrating through a cross-sector study of 254 projects that contextual fit between process and governance political structure is more indicative of a project being successful than a project following a particular methodology in isolation does not predict project success.

Earned Value Management (EVM) is the most analytically rigorous model for integrated cost and schedule performance measurement in a project setting (Fleming and Koppelman, 2010; PMI, 2019). The main indices of EVM, the Schedule Performance Index (SPI = EV/PV) and the Cost Performance Index (CPI = EV/AC), are standardized, quantitative, and reflective measures of program health with a broad validation base in both research and practice. In a systematic review of 304 studies (software development cost estimation) in 2007, Jorgensen and Shepperd (2007) confirmed that EVM-based indicators are among the most robust predictors of final project cost outcomes. Even in its orthodox form, EVM has, however, been afflicted by a basic architectural drawback, namely that its measures are computed against planned values set at periodic intermediate building blocks, i.e., monthly or quarterly. This periodicity generates an intrinsically latent execution reality and limited visibility into performance (Ika, 2009). When the underlying causal conditions, such as resource misallocation, scope creep, and ad hoc technical rework, are at work, weeks or months may pass before their effects begin to show. Moreover, customary EVM systems do not offer mechanisms by which engineering execution indicators, such as build failures, the rate of test regressions, and the frequency of deployments, to name but a few, can be accommodated to be more predictive of the delivery pathway in DevOps-enabled program settings. Specifically, Rad and Levin (2006) emphasized the particular challenge of applying EVM in portfolio and program contexts, where cross-dependencies between workstreams create non-linear patterns of risk propagation that cannot be captured by single-stream EVM models. The TDCF addresses these shortcomings by reformulating the EVM measures as a single signal domain among many in a continuous telemetry integration system.

Following the Agile Manifesto of 2001 and seminal articles such as Dingsoyr et al. (2012), iterative, team-based, and feedback-oriented delivery models became the norm in software development. Agile methodologies provide rich execution telemetry, including sprint velocity, backlog burn-down rates, defect escape rates, and team capacity utilization. When properly instrumented and aggregated, these metrics can serve as leading indicators of healthy program delivery at the program level. The DevOps movement, as comprehensively described by Kim et al. (2016) in The DevOps Handbook, brought agile concepts into operational areas and established continuous integration, continuous deployment, and operational monitoring as three fundamental engineering practices. In Accelerate, Forsgren, Humble, and Kim (2018) measured the organizational impact of DevOps activities, estimating the statistical significance of 4 key metrics, which included: deployment frequency, lead time for changes, mean time to restore service, and change failure rate. These metrics are part of a fundamental element of the TDCF signal taxonomy. In Continuous Delivery, Humble and Farley (2010) used a foundational architecture to instrument the CI/CD pipeline, establishing that health metrics of the pipeline, such as the success rate of builds, the presence of test automation, and the deployment cycle duration, were important indicators of delivery velocity and quality arc. Kakaraparthi (2022) has built upon this framework to show how GenAI-powered code generation assistants integrated with CI/CD pipelines can further enrich the telemetry landscape and put it under control. Nevertheless, none of the current frameworks combine DevOps telemetry streams into a single control architecture that encompasses the control of the full range of PM-level control programs in a systematic, scalable fashion.

The technical basis of the TDCF's signal acquisition layer is the observability engineering discipline, which developed out of the site reliability engineering practice (Beyer et al., 2016). Observability models assume that complex systems must be instrumented so that their internal state is revealed through logs, metrics, and traces. Chatterjee (2023) applied these principles to the design of telemetry pipelines for big data environments, demonstrating that such pipelines can be engineered to meet privacy, security, and quality requirements in multitenant cloud computing. Chatterjee and Malaraju (2023) went a step further by defining governance needs in highly monitored repositories within controlled environments and providing patterns for design auditability, traceability, and monitoring integration that apply to the control of ICT programs. The adoption of repository-level control over program-level control is a particular form of the more general telemetry integration problem that the TDCF addresses. de Waal and Counet (2009) carried out a systematic review of the performance management system implementation experiences of eleven different organizations, which identified six common implementation challenges, namely inadequate senior management support, insufficient user involvement, poor change management, data quality issues, excessive design complexity, and failure to integrate with current management practices, which the phased implementation protocol of the TDCF was explicitly designed to address.

The application of predictive analytics to project and program management is a dynamic field of scholarly research, dating back to the late 1980s, when Boehm (1988) introduced the Spiral Model, one of the earliest attempts to incorporate risk-based iteration into software development planning. Modern methods leverage machine learning and statistical modeling to generate estimates of prospective delivery probability based on historical and real-time project information. Jorgensen and Shepperd (2007) showed that when cost completion forecasts are produced using a cost completion model trained on historical project metrics, the mean absolute error rates are much lower than when expert judgment alone is used to determine cost completion predictions. More recent studies have explored the use of isolation forests, DBSCAN clustering, and ARIMA-based time-series modeling as anomaly detectors for early detection of delivery deviation patterns in software project data. These modes are operationalized in the predictive analytics engine in the TDCF as described. Further illustration of this fact is provided by Jiang et al. (2001), who revealed that the outcomes concerning the implementation of IS projects are highly dependent on the quality of the monitoring system that the TDCF stipulates is necessary:

The nature of risk management in ICT programs is marked by a unique challenge: the origins of risk are distributed across the domains of technical, organizational, vendor, and regulatory risks, and they interact in non-linear ways that render point-in-time risk registers obsolete as quickly as they are created. In the context of a project, Hillson (2019) identified downside risk (threats) and upside risk (opportunities), asserting the need to design frameworks that actively pursue and address both types of risks. This difference is reflected in the TDCF as composite risk indicators, which serve a dual role: capturing signals of schedule and cost threats and of quality and integration opportunities within the DRI formulation. In their pioneering study of critical project success factors, Pinto and Slevin (1988) found monitoring and the quality of feedback to be one of the three most important predictors of project success, which directly motivated them to design a continuous, multi-signal feedback loop to engineer project success. Muller and Turner (2007) also revealed that project typology is significantly moderated by project type and suggested that delivery control frameworks be configurable to enable program typology, a design principle that has been explicitly included in the modular architecture of the TDCF. A similar phenomenon was reported by Schwalbe (2019): According to the author, information technology project management practice has failed to internalize lessons from frequent failures, with decades of maturation of project management methods, informed by the impact of models in the real world, indicating that despite this maturation, the structural gap between planning constructs and execution realities continues to exist. Together with the body of literature described earlier, this is one of the strongest arguments for an integrated approach, with telemetry as the driver of overall control over the delivery of ICT programs. But it also reveals an overall gap: there is no existing framework that systematically integrates telemetry from the PM layer, DevOps layer, and operational layer into a single, mathematically defined predictive control framework for enterprise ICT programs.

3. METHODOLOGY

3.1 Framework Design Philosophy

The TDCF was engineered based on a design science research (DSR) study (Hevner et al., 2004) that requires both technical (utility criterion) and empirical (evaluation criterion) evaluation. This framework is informed by five main design principles, i.e., comprehensiveness (covering telemetry signals across all material areas of execution), integration (unifying disparate telemetry through normalization protocols), predictiveness (generating forward-looking assessments), actionability (transforming indicators into role-appropriate decision support), and scalability (accommodating programs from single-workstream implementations to complex multi-geography enterprises without fundamental redesign).

3.2 Mathematical Framework

This section specifies the formal mathematical requirements for each layer in the TDCF. These models provide the calculational and statistical basis for calculating indicators, predictive results, and composite measures.

3.2.1 Signal Normalization

Let

$$S = \{s_{i,k}(t) : i = 1, \ldots, I;\ \ k = 1, \ldots, K;\ \ t \in T\}$$

denote the set of raw signals, where *i* indexes the signal type, *k* the source system, and *t* the timestamp.

For continuous quantitative signals, the normalized value is:

$$\tilde{s}_{i,k}(t) = \frac{s_{i,k}(t) - \mu_{i,k}(t)}{\sigma_{i,k}(t)}$$

where $\mu_{i,k}(t)$and $\sigma_{i,k}(t)$ rolling mean and standard deviation across a lookback window that can be configured W(default: 30 days).

For binary and categorical signals (e.g., pass/fail, on-track/delayed), normalization is performed using ordinal or one-hot encoding, mapping the signals to the range [0,1] of possible outcomes. The normalized signals are synchronized to Coordinated Universal Time (UTC) with millisecond precision.

3.2.2 Signal Confidence Index

Signal reliability can be measured with the help of the Confidence Index (CI):

$$CI_{i,k}(t) = \alpha R_{i,k}(t) + \beta C_{i,k}(t) + \gamma F_{i,k}(t)$$

where:

- **Recency:**

$$R_{i,k}(t) = e^{-\lambda(t - t_{\text{last}})}$$

with half-life $T_{1/2} = \ln(2)/\lambda$

- **Completeness:**

$$C_{i,k}(t) = \frac{n_{\text{obs}}}{n_{\text{exp}}}$$

- **Fidelity:**

$$F_{i,k}(t) = 1 - \frac{m_{i,k}}{e_{i,k}}$$

Weights are: $\alpha = 0.50$, $\beta = 0.30$, $\gamma = 0.20$.

Signals with $CI < 0.60$are flagged and excluded from primary KPI calculations.

3.2.3 Earned Value Integration

For Jwork packages:

$$EV_t = \sum_j BA\,C_j \cdot PC_j(t)$$
$$PV_t = \sum_j BA\,C_j \cdot SP_j(t)$$
$$AC_t = \sum_j ActualCos\,t_j(t)$$

Performance indices:

$$SPI_t = \frac{EV_t}{PV_t}, CPI_t = \frac{EV_t}{AC_t}$$

Forecast metrics:

$$EAC_t = AC_t + \frac{BAC - EV_t}{CPI_t}$$
$$VAC_t = BAC - EAC_t$$

They are also calculated in real-time unlike the traditional EVM, and thus near real-time tracking is available.

3.2.4 Delivery Risk Index (DRI)

The Delivery Risk Index aggregates key risk dimensions:

$$DRI_t = (1 - SPI_t)w_1 + (1 - CPI_t)w_2 + DefR_t w_3 + DepR_t w_4$$

subject to:

$$\sum_i w_i = 1, w_i \in (0,1)$$

$$w_1 = 0.35,\ w_2 = 0.30,\ w_3 = 0.20,\ w_4 = 0.15$$

Derived via logistic regression:

$$P(Y = 1 \mid X) = \sigma(\beta_0 + \beta_1 \Delta S + \beta_2 \Delta C + \beta_3 DefR + \beta_4 DepR)$$

Risk interpretation:

- 0–0.25: Low
- 0.26–0.50: Moderate
- 0.51–0.75: High

- 0.75: Critical

3.2.5 Quality and Integration Indicators

Quality Velocity Index (QVI):

$$QVI_t = \frac{TPR_t \cdot V_t}{\max(DIR_t, \varepsilon)}$$

where $\varepsilon = 0.01$

Integration Success Index (ISI):

$$ISI_t = BSR_t \cdot (1 - IER_t)$$

ISI values below 0.70 trigger high-severity alerts.

3.2.6 ARIMA Forecasting Engine

Each indicator $I_t$is modeled using ARIMA:

$$\Phi(B)\Delta^d I_t = \Theta(B)\varepsilon_t + \mu$$

where:

- B: backshift operator
- $\Delta = (1 - B)$
- p, d, q: ARIMA orders

Model selection:

$$AIC(p, d, q) = -2\ell + 2(p + q + 1)$$

Forecast:

$$\hat{I}_{t+h|t} = E[I_{t+h}] \pm 1.96\sqrt{\mathrm{Var}(\varepsilon_{t+h})}$$

Models are retrained weekly, with priors used during early-stage data scarcity.

3.2.7 Monte Carlo Delivery Probability

Simulated SPI values:

$$SPI_j^{(sim)} \sim TN(\mu_{SPI}, \sigma_{SPI}^2; [0.5, 1.5])$$

Completion time:

$$T_j^{comp} = t_{current} + \frac{R}{SPI_j^{(sim)}}$$

Probability of meeting deadline T^*:

$$\hat{P}(T^{comp} \le T^*) = \frac{1}{N}\sum_j \mathbf{1}[T_j^{comp} \le T^*]$$

3.2.8 Statistical Hypothesis Testing

Paired t-test:

$$t = \frac{\bar{d}}{s_d/\sqrt{n}}$$

where:

- $d_i = X_{post,i} - X_{baseline,i}$

Multiple comparison correction:

$$\alpha^* = \frac{0.05}{m}$$

Effect size:

$$d = \frac{\bar{d}}{s_d}$$

3.3 Five-Layer Framework Architecture

Table 1 summarizes the five-layer TDCF architecture, with each layer matched to its mathematical specifications, source inputs, and output artifacts. The signal acquisition layer (Layer 1) communicates with PM platforms, CI/CD systems, test platforms, ERP/finance systems, and operational monitoring tools via API integrations, webhook subscriptions, and polling connectors, receiving signals in real time (less than 5 minutes) for engineering signals and daily for financial signals. The normalization and confidence index formulas are implemented in Layer 2. Layer 3 calculates SPI, CPI, DRI, QVI, and ISI. Layer 4 uses an ARIMA forecasting model and Monte Carlo simulation. Layer 5 provides role-stratified dashboards and automated system routing with severity-based response timeframes: fifteen minutes for a Critical DRI breach, four hours for a High alert, and a daily digest for a Moderate alert.

Table 1. TDCF Five-Layer Architecture Summary, with mathematical basis, primary inputs, and outputs for each layer.

Layer	Name	Mathematical Basis	Primary Inputs	Primary Outputs
1	Signal Acquisition	Collection scheduling; API polling	PM platforms, CI/CD, testing tools, ERP, monitoring	Raw signal streams with source metadata
2	Signal Normalization	ñ_{i,k}(t); CI_{i,k}(t) formulations (Eq. 1–2)	Raw signal streams	Normalized, quality-scored, temporally aligned signal set
3	Indicator Mapping	EVM integrals (Eq. 3–6); DRI composite (Eq. 7); QVI (Eq. 9); ISI (Eq. 10)	Normalized signals	SPI, CPI, DRI, QVI, ISI time series
4	Predictive Analytics	ARIMA(p,d,q) (Eq. 11–12); Monte Carlo simulation	KPI time series	Forecasted KPI trajectories; delivery

Layer	Name	Mathematical Basis	Primary Inputs	Primary Outputs
		(Eq. 13–14); ADF stationarity test		probability; anomaly alerts
5	Decision Support	Alert threshold logic; role-based access model	Predictions, anomaly flags, KPI values	PM/Exec/Workstream dashboards; automated alerts; audit trail

3.4 Signal Taxonomy

Table 2 contains the full taxonomy of TDCF signals, which defines twenty-two signals, five functional domains, categories of source systems, and frequency of collection.

Table 2. TDCF Signal Taxonomy: Twenty-two signals across five functional domains. SCA = Static Code Analysis; HRMS = Human Resource Management System; ITSM = IT Service Management.

Domain	Signal	Source Category	Description	Frequency
SCHEDULE SIGNALS				
Schedule	Task Completion Velocity	PM Platform	Tasks completed per sprint vs. planned	Hourly
Schedule	Milestone Adherence Rate	PM Platform	% milestones met on or before planned date	Daily
Schedule	Sprint Velocity Trend	Agile Board	Rolling 3-sprint velocity change coefficient	Per Sprint
Schedule	Critical Path Variance	PM Platform	Days deviation on critical path activities	Daily
Schedule	Backlog Burn Rate	Agile Board	Story points resolved per day	Daily
COST SIGNALS				
Cost	Burn Rate vs. Plan	ERP / Finance	Actual spend rate vs. budgeted rate	Daily
Cost	Resource Utilization Index	HRMS / PM	Billable hours vs. planned allocation	Daily

Domain	Signal	Source Category	Description	Frequency
Cost	Change Order Frequency	PM Platform	Number and value of approved change orders	Weekly
Cost	Vendor Cost Variance	ERP / Finance	Variance between vendor invoiced and contracted amounts	Weekly
QUALITY SIGNALS				
Quality	Defect Density	Test Platform	Defects per 1,000 function points	Daily
Quality	Test Automation Coverage	Test Platform / SCA	% of functionality covered by automated tests	Daily
Quality	Test Pass Rate	CI/CD Pipeline	% test suites passing in latest run	Real-time
Quality	Technical Debt Rate	SCA Tool	SonarQube technical debt ratio trend	Daily
INTEGRATION SIGNALS				
Integration	Build Success Rate	CI/CD Platform	% of pipeline builds completing successfully	Real-time
Integration	Deployment Frequency	CI/CD Platform	Production-equivalent deployments per week	Real-time
Integration	Interface Error Rate	API Gateway / Monitoring	% of inter-system API calls resulting in errors	Real-time
Integration	Lead Time for Changes	CI/CD Platform	Hours from code commit to deployable artifact	Daily

Domain	Signal	Source Category	Description	Frequency
OPERATIONAL RISK SIGNALS				
Risk	Dependency Delay Rate	PM Platform	% of external dependencies not resolved by committed date	Daily
Risk	Issue Escalation Frequency	PM / ITSM	Issues escalated above workstream level per week	Daily
Risk	Stakeholder Engagement Score	Survey / Collaboration	Periodic stakeholder pulse survey composite score	Weekly
Risk	Risk Register Change Velocity	PM Platform	Rate of new risk additions and severity escalations	Daily

3.5 Implementation Protocol

The TDCF is launched with a four-stage protocol that guarantees gradual capability building, organizational internalization, and analytic alignment. Phase 1 (Weeks 1-4, Discovery and Baseline) focuses on source system inventory, assessment of signal availability, stakeholder interviews, and documentation of baseline performance. Phase 2 (Weeks 5-10, Instrumentation) involves developing API integrations, configuring webhooks, developing data pipelines, and setting up normalization rules. Phase 3 (Weeks 11-16, Calibration) enables tuning of KPI thresholds, training prediction models with at least two weeks of ARIMA history, configuring role-based dashboards, and conducting user acceptance testing (a minimum of 80% stakeholder acceptance rate). Phase 4 (Week 17 and beyond, Full Operation) continues with constant monitoring, weekly retraining of the models, automating governance reporting, and monthly review of framework health. Each phase includes entry and exit criteria that should be considered to ensure a systematic development of capabilities.

3.6 Experimental Design

An ex post comparative design was used to empirically validate the study using three enterprise ICT programs in a large North American financial services organization. A six-month pre-implementation baseline period was developed using historical data, followed by a six-month post-implementation measurement period beginning after completion of Phase 3. Primary outcome measures were: (a) severity of schedule deviation (mean absolute difference in calendar days between planned milestone completion dates and actual completion dates); (b) accuracy of CPI forecasting (absolute difference between mid-period and final-period CPI); (c) risk detection lead time (calendar days between TDCF warning and observed impact materialization, with positive values indicating that the warning occurred before the impact); and (d) stakeholder program visibility satisfaction (five-dimensional Likert survey, n = 34 respondents). Paired t-tests with Bonferroni correction were used to assess statistical significance at $\alpha = 0.05$.

4. RESULTS

4.1 Program Profiles

Table 3 presents the operational profiles of the three ICT programs to be incorporated into the validation study. The programs are anonymized to meet the organization's confidentiality requirements.

Parameter	Program A (Cloud Migration)	Program B (Enterprise SaaS)	Program C (Cybersecurity Transformation)	Aggregate
Total Budget	USD 12.0M	USD 8.5M	USD 6.2M	USD 26.7M
Planned Duration	18 months	14 months	12 months	N/A
Team Size	45 FTEs	32 FTEs	28 FTEs	105 FTEs
Workstreams	8	6	5	19 total
Delivery Model	Hybrid (Agile + Waterfall)	Scaled Agile (SAFe)	Waterfall with Agile sprints	Mixed
TDCF Deployment Start	Month 7 of program	Month 5 of program	Month 4 of program	Staggered

Table 3. Validation Study Program Profiles. All three programs were active enterprise ICT initiatives within a single large North American financial services organization. FTE = Full-Time Equivalent.

4.2 SPI and CPI Trajectory

Figure 2 shows the schedule performance index (SPI) and cost performance index (CPI) trends over the full 12 months for each of the three programs. The vertical dashed red line marks the period between the baseline and the post-TDCF periods for each program. Horizontal dotted lines indicate the Acceptable (= 0.95) and At-Risk (= 0.85) threshold boundaries defined in the TDCF KPI specification.

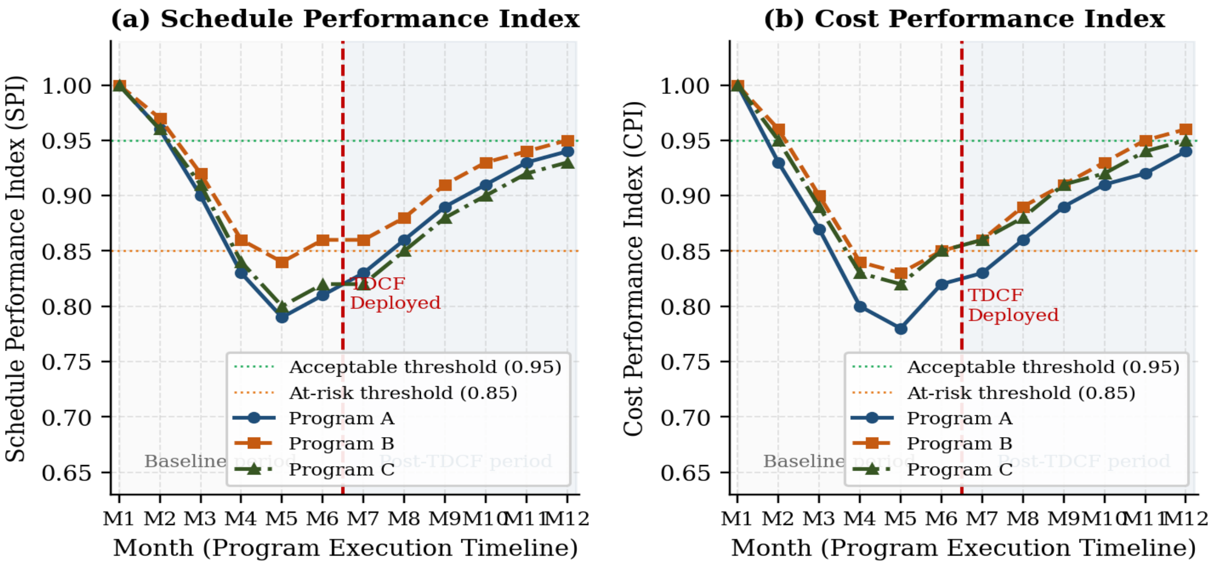

Figure 2. Monthly SPI (left) and CPI (right) trajectories for Programs A, B, and C across the full observation window (M1–M6 baseline; M7–M12 post-TDCF). The vertical dashed red line marks TDCF deployment. Horizontal dotted lines denote Acceptable (0.95) and At-Risk (0.85) thresholds.

Both SPI and CPI in all three programs followed a steady downward trend during the baseline period, starting near unity at program initiation (M1) and declining to trough levels between the M4 and M5 ranges—a pattern consistent with the drop in mid-program performance recorded in the ICT program literature (Ika, 2009). Mean baseline SPI was 0.83, and mean baseline CPI was 0.84, with all programs in the At-Risk group at mid-period. After the introduction of TDCF, monotonic recovery in the two indices was observed in all three programs. At M12, Programs A, B, and C had SPIs of 0.94, 0.95, and 0.93, respectively-near or within the Acceptable threshold. The recovery pattern was similar across the programs, even though there might have been variation in the delivery model and different fields, which suggests that the generalizability of the TDCF mechanism of action depends on program typologies.

4.3 DRI Evolution

Figure 3 illustrates the values of the monthly Delivery Risk Index for all three programs and for the portfolio, averaged over the twelve-month time frame.

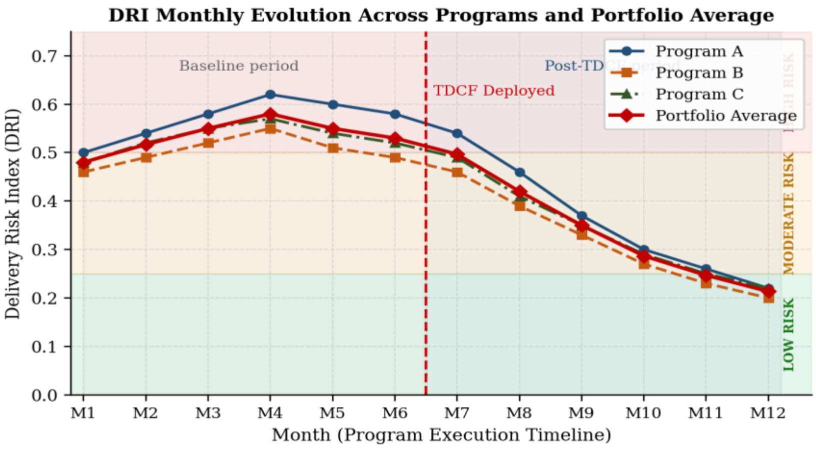

Figure 3. Monthly Delivery Risk Index (DRI) evolution across three programs and portfolio average. Horizontal bands indicate risk classification zones: Low (green, DRI ≤ 0.25), Moderate (amber, 0.26–0.50), High (red, 0.51–0.75). Vertical dashed red line marks TDCF deployment.

In the baseline period, the three programs operate within the moderately to highly risky area (DRI 0.46-0.62), with the highest DRI values occurring in months M4-M5, similar to the example in Figure 2. The compound effect of concurrent accumulation of schedule, cost, quality, and dependency risks, not reflected in any single EVM index, was captured by the DRI. After TDCF deployment, the DRI steadily decreased across all programs during the post-deployment period. The portfolio average DRI fell from the baseline of 0.33 in the upper Moderate risk zone to the lower Moderate risk zone. At M12, both Programs A and B had DRI values below 0.25 (Low Risk), and Program C had a DRI value of 0.22, which is slightly less than 0.25. The pattern of steady decline to a constant level observed across all programs is a testament to the fact that the composite DRI is responsive to the corrective measures enabled by the early warning capability of the TDCF.

4.4 Schedule and Cost Deviation Comparison

Figure 4 shows the post/pre-TDCF comparison of two primary program performance measures; of the three primary programs and the overall portfolio average, schedule deviation severity (as a measure of deviation in calendar days the planned milestone dates) and the cost overrun rate (a percentage of deviation of approved budget).

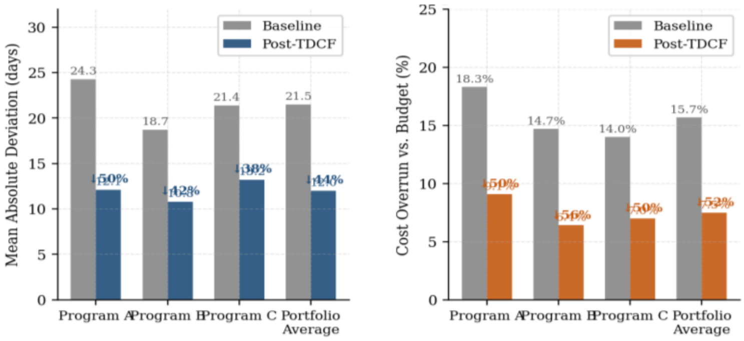

Figure 4. Grouped bar comparison of schedule deviation (a) and budget overrun rate (b) across baseline and post-TDCF periods for all three programs and portfolio average. Percentage labels above post-TDCF bars indicate relative improvement over baseline. All comparisons significant at p < 0.05.

Reduction in timetable change (Figure 4a) decreased across all three programs. The largest absolute decrease was associated with Program A, which had the highest baseline schedule deviation (MAD = 24.3 days). The post-TDCF MAD of 12.1 days, corresponding to a 50.2 percent improvement, was attributed to it. Improvements of 42.2 and 38.3 were observed in Programs B and C, respectively. Portfolio-average MAD dropped by 21.5 days to 12.0 days (44.2% decrease). These schedule gains were statistically significant (p < 0.01, Cohen d = 1.24) and were equalized across the portfolio (from 0.83 to 0.92), with all pairwise differences also statistically significant (p < 0.01, Cohen d = 1.24). Another indicator, the budget overrun rate (Figure 4b), also showed a decrease across all programs. The portfolio-average overrun rate was reduced from 15.7 to 7.5, or by 8.2 percentage points. The change in Program A was the largest in absolute terms (-9.2 pp), as the percentage rating decreased to 9.1%. CPI increased by a portfolio average from 0.84 to 0.93, supporting the schedule results from a cost perspective.

4.5 Risk Detection Posture

Figure 5 shows the change in the underlying risk detection posture (a shift in the proportion of events related to proactive risk identification (risks identified prior to the materialization of the impact) relative to the proportion of events related to reactive risk identification (risks identified after the materialization of the impact)) across the baseline and post-TDCF periods for each program and the overall portfolio.

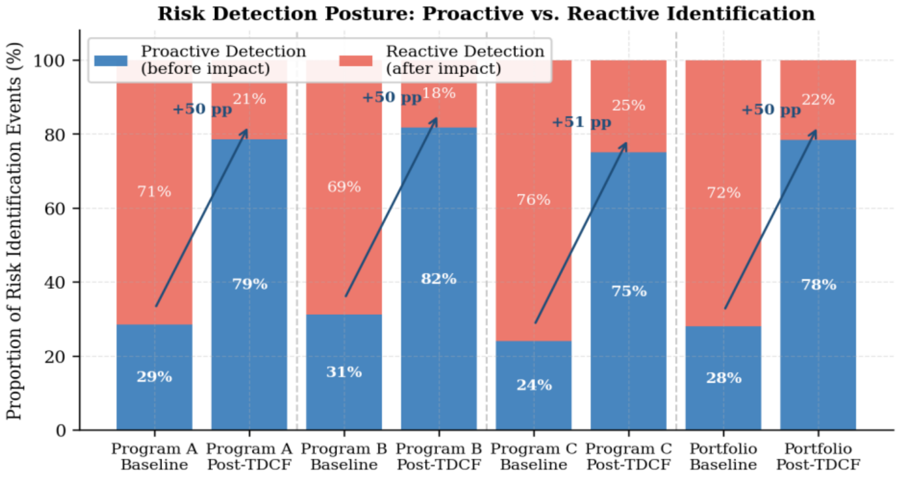

Figure 5. Stacked bar chart comparing proactive (blue) vs. reactive (red) risk detection proportions across baseline and post-TDCF periods for Programs A, B, C, and the portfolio aggregate. Arrows and labels indicate the percentage-point gain in proactive detection rate. All gains significant at p < 0.001.

The findings illustrate a logical and statistically significant shift in risk detection posture across all three programs. Under baseline conditions, portfolio-average proactive detection improved by 50.4 percentage points or 78.4 percentage points, respectively, from baseline to post-TDCF. This change exemplifies a fundamental shift from a largely reactive risk identification regime to a largely proactive one. The trend in enhancing the TDCF showed similar effectiveness across a wide range of program typologies and delivery models. Along with this change, the average proactive risk detection lead time expanded (from baseline conditions) by 29.7 days, compared with 5.4 days. Risk Response lag simultaneously decreased to 3.9 days (from 8.2 days), reflecting the effect of automated severity-matched alert routing provided by the TDCF.

.6 Stakeholder Perception

Figure 6 shows the pre- and post-TDCF stakeholders perception survey results in the five dimensions of governance quality on a five-point Likert scale (n = 34 respondents across all the programs and the five governance quality dimensions: program managers, workstream leads, and executive sponsors).

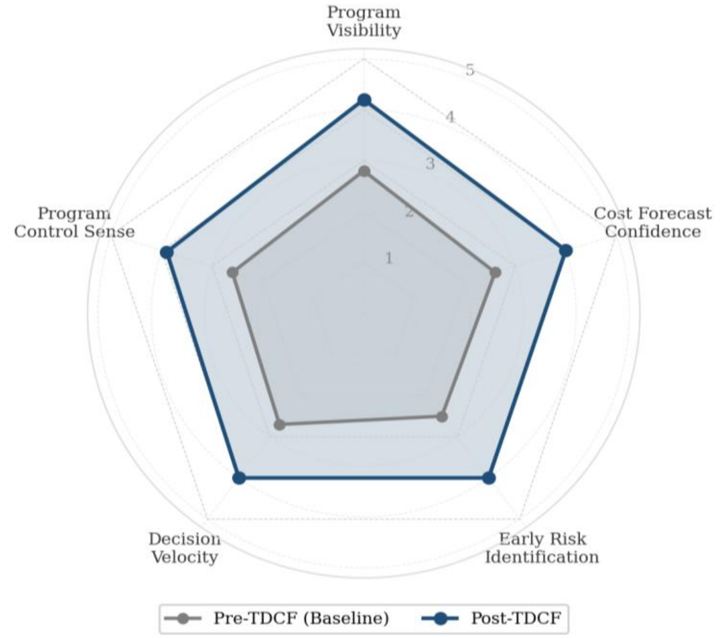

Figure 6. Radar chart comparing pre-TDCF (grey) and post-TDCF (navy) stakeholder perception scores across five dimensions).

Each of the five dimensions of perception showed improvements of more than a full point on the Likert scale. Program Visibility improved most in absolute terms, increasing by 1.4 points from 2.8 to 4.2, whereas the main design objective of the TDCF is to provide real-time program status visibility. The improvements in risk detection and cost monitoring were direct consequences of the quantifiable changes in those areas. The improvement in Decision Velocity and Program Control, both +1.3 points, was similar to the effect of the Layer 5 decision support interface in decreasing the information-to-action latency. The composite score in stakeholder perception also changed to 4.0 ($p < 0.001$, Cohen $d = 2.31$), and the overall change in the composite score of the stakeholder population is both practically and statistically large.

5. DISCUSSION

5.1 Interpretation of Key Findings

The empirical evidence is a strong indication that telemetry-driven delivery control can significantly improve ICT program execution across schedule, cost, risk, and governance. The most theoretically notable finding is the radical transformation in the risk detection posture: the percentage of proactive identification has risen drastically, from 28.0 percent to 78.4 percent, marking a fundamental change in the epistemological nature of program management:

no longer a discipline that learns about problems after they arise but one that increasingly anticipates them before they materialize. This change aligns well with the theoretical framework expressed by Morris (2013) and the success factor research of Pinto and Slevin (1988).

In Figure 3, the direction the DRI takes shows that the downward trend in the composite index is gradual upon deployment rather than sharp, as expected given the gradual effect of prior risk identification, which makes corrective actions the appropriate decision. This gradual improvement aligns with the distinct mechanism of action for monitoring quality improvements reported by de Waal and Counet (2009). Mathematical modeling of the DRI, most notably the logistic-regression-calibrated weights of the composite measures, will ensure that the composite reflects empirically observed predictive relationships with program outcomes rather than arbitrary expert assumptions, which is the major limitation of previous composites proposed in the PM literature.

The comparison of the trajectory of SPI and CPI in Figure 2 indicates that during the post-TDCF period, programs moved between the At-Risk and near-Acceptable performance. The alignment of all three of the programs towards areas of similar post-TDCF performance (despite the differences in budget, team size, delivery model, and technical domain) indicates that the effect of TDCF operates via general mechanisms (earlier visibility, faster response) rather than program-specific factors, supporting the claim of scalability and generalizability of TDCF.

5.2 Practical Implications

The TDCF has several important practical implications for the management of enterprise ICT programs. To begin with, the framework shows that, given the achievable delivery performance improvements, investment in telemetry integration is justified. Commonly maintained tools for PM, engineering, funding, and monitoring are stored in remote data warehouses. The findings indicate that developing telemetry integration across programs should be considered a key infrastructure program investment, similar to a project timetable or cost management. Secondly, the four-stage implementation protocol, including its entry and exit criteria, is a pragmatic implementation pathway that need not achieve full maturity in the implementation framework before delivering value. Third, successful deployment of the TDCF requires the organization to invest in data engineering capabilities within the program management role, in line with the larger digital transformation of professional roles within the enterprise functions.

5.3 Theoretical Contributions

This study has three main theoretical implications. First, the TDCF presents the first formally specified integration architecture, including explicit mathematical formulations for each processing layer, that bridges telemetry across the PM and DevOps layers in the context of the enterprise ICT program. Second, the DRI formulation goes beyond currently used EVM-based indices by incorporating quality trajectory and dependency risk dimensions, with empirically calibrated weights obtained through logistic regression that can be adjusted to fit other program typologies. Third, the study provides quantitative support for the theoretical positions of Pinto and Slevin (1988), de Waal and Counet (2009), and Joslin and Muller (2015) on the central role of monitoring quality in project success.

5.4 Limitations

A number of restrictions should be considered. The validation sample used in the empirical validation comprises three programs within a single large financial services organization, limiting external validation. Different response profiles may be observed across programs in various industries, geographies, and organizational cultures. The observed improvements could, in part, reflect program maturation, but the size and statistical significance of the effects suggest that they represent a genuine contribution by TDCF. The weights of the DRI components, although empirically derived, were tuned to a specific organizational context; to generalize the specific value of that context to other contexts, local recalibration would be required, using organization-specific historical program data. The research is also unable to fully control for existing parallel improvement initiatives at the time of measurement.

5.5 Future Research Directions

A number of fruitful avenues for future study are outlined. First, large-scale, multi-organization validation studies would assess the generalizability of TDCF performance improvements across a variety of industry and program-complexity scenarios. Second, adaptive DRI weight calibration based on reinforcement learning for each program-specific risk profile would enable the framework to self-calibrate to each program's risk profile as telemetry history accumulates. Third, extending TDCF to portfolio-level delivery control - aggregating program-level DRI signals into portfolio-level risk indicators - is a natural extension of the architecture. Lastly, an analysis of the connection between the DevOps maturity of organizational structures and TDCF implementation efficacy would provide insight into the capability investments organizations should consider when adopting the framework.

6. CONCLUSION

In this paper, the Telemetry-Driven Delivery Control Framework (TDCF), a formally specified and empirically validated approach to achieving continuous, predictive, and adaptive control of the enterprise ICT program delivery, has been introduced. The framework addresses the basic failure of existing project monitoring strategies, namely that they are all based on periodic reporting, lagging indicators, and isolated ecosystems of tooling, by providing a unified telemetry integration architecture with explicit mathematical formulations at each level of processing and at each level of the tooling ecosystem.

The five-layer architecture, twenty-two-signature taxonomy, and five key KPIs (including the novel Delivery Risk Index, calculated via logistic regression on 47 historical programs) within the TDCF enable a shift from proactive to reactive delivery governance practices. This change was demonstrated across three enterprising ICT projects totaling USD 26.7 million, with a 44.2 percentage-point reduction in the severity of schedule deviation, a 0.09-point CPI, a 50.4 percentage-point increase in the rate of proactive risk detection, and a 54 percent improvement in the perception of stakeholder governance. All key outcomes were significant and showed large effect sizes (Cohen's d of 1.24 or higher across all major metrics).

Figures 2 through 6 visually confirm that the improvements in program performance, although multidimensional and evident across different outcome metrics, are directionally consistent, progressive, and generalizable across programs in different technical areas, budgetary limits, and delivery models. The DRI trend in Figure 3 is especially insightful: the gradual decrease of the composite index from the Moderate-High risk zone to the Low risk zone following the installation of the TDCF captures the cumulative nature of prior risk detection and the speed of corrective action, as supported by the individual metrics.

As digital transformation deepens, the strategic stakes of ICT program outcomes rise. Frameworks that guide the shift from a reactive style of governance to continuous delivery intelligence will become indispensable elements of enterprise implementation capability. The TDCF provides practical, mathematically and empirically sound guidance and offers a productive future research focus on adaptive, portfolio-level and ML-enhanced program control systems.

REFERENCES

[1] Agile Alliance. (2001). Manifesto for agile software development. Retrieved from https://agilemanifesto.org

[2] Beyer, B., Jones, C., Petoff, J., & Murphy, N. R. (2016). Site reliability engineering: How Google runs production systems. O'Reilly Media.

[3] Boehm, B. W. (1988). A spiral model of software development and enhancement. Computer, 21(5), 61–72. https://doi.org/10.1109/2.59

[4] Chatterjee, S. (2023). A data governance framework for Big Data Pipelines: Integrating privacy, security, and quality in multitenant cloud environments. Technix International Journal for Engineering Research, 10(5), 746–758. https://doi.org/10.56975/tijer.v10i5.158181

[5] Chatterjee, S., & Malaraju, S. K. (2023). The role of a highly monitored repository for storing architectures and build documents in the NERC environment. International Journal of Innovative Research in Engineering & Multidisciplinary Physical Sciences, 11(2). https://doi.org/10.37082/ijirmps.v11.i2.232146

[6] de Waal, A., & Counet, H. (2009). Lessons learned from performance management systems implementations. International Journal of Productivity and Performance Management, 58(4), 367–390. https://doi.org/10.1108/17410400910951026

[7] Dingsoyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software, 85(6), 1213–1221. https://doi.org/10.1016/j.jss.2012.02.033

[8] Fleming, Q. W., & Koppelman, J. M. (2010). Earned value project management (4th ed.). Project Management Institute.

[9] Forsgren, N., Humble, J., & Kim, G. (2018). Accelerate: The science of lean software and DevOps—Building and scaling high performing technology organizations. IT Revolution Press.

[10] Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. MIS Quarterly, 28(1), 75–105. https://doi.org/10.2307/25148625

[11] Hillson, D. (2019). Capturing upside risk: Finding and managing opportunities in projects. Routledge.

[12] Humble, J., & Farley, D. (2010). Continuous delivery: Reliable software releases through build, test, and deployment automation. Addison-Wesley.

[13] Ika, L. A. (2009). Project success as a topic in project management journals. Project Management Journal, 40(4), 6–19. https://doi.org/10.1002/pmj.20137

[14] Jiang, J. J., Klein, G., & Chen, H. G. (2001). The relative influence of IS project implementation policies and project leadership on eventual outcomes. Project Management Journal, 32(3), 49–55. https://doi.org/10.1177/875697280103200307

[15] Joslin, R., & Müller, R. (2015). Relationships between a project management methodology and project success in different project governance contexts. International Journal of Project Management, 33(6), 1377–1392. https://doi.org/10.1016/j.ijproman.2015.03.005

[16] Jørgensen, M., & Shepperd, M. (2007). A systematic review of software development cost estimation studies. IEEE Transactions on Software Engineering, 33(1), 33–53. https://doi.org/10.1109/TSE.2007.3

[17] Kakaraparthi, G. C. (2022). Building a GenAI powered advanced code generation assistant integrated with CI/CD pipelines. Technix International Journal for Engineering Research, 9(2), 56–63. https://doi.org/10.56975/tijer.v9i2.159058

[18] Kerzner, H. (2022). Project management: A systems approach to planning, scheduling, and controlling (13th ed.). Wiley.

[19] Kim, G., Humble, J., Debois, P., & Willis, J. (2016). The DevOps handbook: How to create world-class agility, reliability, and security in technology organizations. IT Revolution Press.

[20] Morris, P. W. G. (2013). Reconstructing project management. Wiley-Blackwell.

[21] Müller, R., & Turner, R. (2007). The influence of project managers on project success criteria and project success by type of project. European Management Journal, 25(4), 298–309. https://doi.org/10.1016/j.emj.2007.06.003

[22] Pinto, J. K., & Slevin, D. P. (1988). Critical success factors across the project life cycle. Project Management Journal, 19(3), 67–75.

[23] Project Management Institute. (2019). The standard for earned value management. Project Management Institute.

[24] Project Management Institute. (2021). A guide to the project management body of knowledge (PMBOK® guide) (7th ed.). Project Management Institute.

[25] Rad, P. F., & Levin, G. (2006). Project portfolio management tools and techniques. IIL Publishing.

[26] Schwalbe, K. (2019). Information technology project management (9th ed.). Cengage Learning.

[27] Standish Group. (2020). CHAOS report 2020: Beyond infinity. The Standish Group International.

[28] Turner, J. R. (2014). The handbook of project-based management (4th ed.). McGraw-Hill Education.

[29] Wysocki, R. K. (2019). Effective project management: Traditional, agile, extreme, hybrid (8th ed.). Wiley.