Beyond Scope, Schedule, and Budget:

A Software Quality Framework for Washington State IT Projects

Executive Summary

Washington State agencies face increasing pressure to deliver reliable, high-performance technology systems within limited budgets and public scrutiny. However, a persistent gap exists between project management practices and the assessment of the quality of the resulting systems.

Traditional metrics such as scope, schedule, and budget measure whether a project was managed well. They do not evaluate if the delivered system will function as intended, perform reliably under real-world conditions, or provide lasting value to the state and its residents.

The consequences are tangible and significant. When government technology fails, it's not just an operational inconvenience. Residents miss benefit payments they rely on, small businesses cannot obtain necessary licenses, and state employees spend hours on manual workarounds instead of assisting constituents. Public trust in government diminishes. These are the real costs of software quality failures in the public sector.

This white paper explores why software quality should play a more central role in Washington State IT project planning and oversight. Drawing on guidance from the Washington State Office of the CIO (OCIO) and WaTech, including the

Software Quality Best Practices Guideline (EA-01-02-G) [1], the IT Investments Approval & Oversight Policy (PM-01) [2], and the Quality Assurance Standard (PM-03) [3], as well as established quality frameworks and observed patterns across government technology initiatives, it argues that:

• Project quality and software quality are different concepts that need separate focus.

• Quality issues found late in a project are far more costly to fix.

• Data interface quality, especially in the context of the OneWA/Workday rollout, is one of the highest-risk and most underestimated quality challenges facing Washington agencies today.

• Accumulation of technical debt during implementation leads to unplanned future projects, which can weaken the long-term benefits the original system was meant to deliver.

• Agencies can greatly reduce technology risks by integrating quality practices throughout the entire implementation process, not just at the end.

> 💡 **Key Insight**
>
> *A government IT system that delivers on time and on budget but fails to perform reliably in production has not succeeded. Software quality is the bridge between project execution and program outcomes, and ultimately between government investment and community benefit.*

Disclaimer: This white paper was prepared by Critical Logic and reflects their views and opinions only. It does not represent the official policy, position, or endorsement of Washington State, the Office of the Chief Information Officer (OCIO), Washington Technology Solutions (WaTech), the Office of Financial Management (OFM), or any other state agency. References to WaTech and OCIO guidelines, the OneWA program, and other state initiatives are included solely for informational purposes. Readers should consult official state guidance and their agency's legal and policy advisors for authoritative advice on compliance requirements.

1. The Problem with Measuring Only What Is Easy to Measure

There are reasons why scope, schedule, and budget dominate government IT project reporting. They are quantifiable, easy to communicate, and align well with appropriations and legislative oversight structures. A project is either on schedule, or it isn't. It either stays within the approved budget or it doesn't. These metrics provide a clean, accountable narrative.

But that narrative is incomplete.

Consider what these metrics do not capture:

- Whether the system correctly implements business rules and program logic
- Whether the system performs adequately under the workloads it will encounter in production
- Whether integrations with other systems — including state enterprise platforms like Workday — behave reliably or produce data inconsistencies
- Whether the system can be maintained, extended, or modified at a reasonable cost after deployment
- Whether users can accomplish their tasks effectively and without encountering errors
- Whether the system actually delivers improved outcomes for the residents and communities it is meant to serve

None of these are captured in a scope tracker, Gantt chart, or budget variance report. Yet all of them directly affect whether an agency's technology investment delivers the intended value.

1.1 The Classic Failure Pattern

The pattern is familiar to anyone who has been involved in government IT for more than a few years. A project team manages the triple constraint diligently. The schedule is tracked weekly. Budget burn is monitored closely. Scope is baselined and change-controlled.

Then the system goes live.

Within weeks or months, a new class of problems emerges, performance is sluggish during peak periods, data from an upstream system arrives malformed and causes downstream errors, a business rule was misunderstood during requirements and is now producing incorrect calculations, or users are struggling with workflows that were never adequately tested in realistic conditions.

By this point, the project is technically closed. The team has been reassigned. The budget has been spent. The agency is now managing a system that is expensive to fix and difficult to modify.

This is not a failure of project management. It is a failure of quality management.

1.2 Why the Metrics Gap Persists

The persistence of the scope, schedule, and budget as the dominant framework for IT oversight is not surprising. These metrics align with how government budgets work: funding is approved for a defined scope, expected to be delivered by a certain date, at a specified cost. Oversight bodies and executive sponsors are naturally inclined to measure progress against those same parameters.

WaTech's IT Investments Approval & Oversight Policy (PM-01) [2] establishes roles and responsibilities for major IT investment planning and monitoring — and explicitly states that the greatest value of quality assurance is realized "when present at all stages of a project, from feasibility through implementation." Yet in practice, agencies often concentrate on quality activities in a testing phase near the end. Closing this gap requires deliberate action at both the agency and enterprise levels.

📋 Washington State Policy Context

WaTech's Software Quality Best Practices Guideline (EA-01-02-G) [1] explicitly requires agencies to follow a "defined software development process that encompasses requirements for gathering, designing, coding, testing, and deployment" — and to conduct "thorough testing throughout the software development lifecycle." Projects that treat quality as a deployment-phase activity, rather than a lifecycle-wide discipline, are directly at odds with this guidance.

2. Software Quality and Community Benefit

Government technology is not built for its own sake. It is built to serve people — the residents of Washington State who depend on government services, the businesses that rely on regulatory and licensing systems, and the communities whose well-being is affected by how effectively public programs function.

This public-purpose dimension of government IT is what makes software quality fundamentally different in government than in the commercial world. When a consumer application has poor user experience, customers switch to a competitor. When a government system has poor user experience or produces incorrect outputs, there is often no alternative. Residents must use the system to access the services they are entitled to.

2.1 The Direct Connection Between Software Quality and Public Outcomes

The connection between software quality and community benefit operates across multiple dimensions:

Quality Dimension	Community Impact When Quality Fails
Functional correctness	Benefits are miscalculated, eligibility determinations are wrong, or payments are missed — directly harming the residents the program is designed to serve
Performance under load	Systems slow or fail during peak demand periods (WA Benefits Exchange, renewals, tax deadlines, emergency registrations), precisely when residents most need reliable access
Data integrity across systems	Inaccurate or incomplete information flows between agencies, leading to incorrect decisions about individuals — from child welfare to professional licensing to tax administration
Usability and accessibility	Residents with limited technical literacy, disabilities, or language barriers are unable to access services independently, increasing inequity and driving up call center costs
Reliability and availability	Outages prevent service delivery at critical moments, eroding public trust and imposing costs on residents who must make repeated attempts or seek in-person assistance
Maintainability	Systems that are expensive or difficult to modify cannot be adapted as program needs, laws, or regulations change — agencies must deliver services through workarounds rather than systems

2.2 The Equity Dimension of Software Quality

Software quality failures in government systems often have a disproportionate impact on the most vulnerable members of the community. Residents who depend on government services for basic needs — food assistance, housing support, healthcare coverage, disability services — are most affected when those systems do not function correctly.

When a benefits system produces incorrect eligibility determinations, the consequences for a family living paycheck to paycheck are far more severe than for families with financial resources to absorb temporary disruptions. When a government portal is inaccessible to users with disabilities or low digital literacy, it effectively excludes people who most need government support.

For Washington State, which has made commitments to equity and inclusive service delivery, software quality is not merely a technical consideration. It is an equity consideration. Building quality into government IT systems is one of the most concrete ways agencies can ensure that their technology investments translate into equitable outcomes for all residents.

2.3 Quality as a Return on Public Investment

Washington State's technology investments are funded by taxpayers and authorized by the Legislature with the expectation that they will improve government performance and service delivery. A system that technically meets its project requirements but fails to deliver the expected operational improvements does not fulfill this mandate.

Measuring return on technology investment requires looking beyond project execution metrics to ask: Did the system actually change how the agency serves the public? Are processes faster, more accurate, and more accessible than before? Are staff freed from manual workarounds to focus on higher-value work? Are residents receiving better outcomes?

These questions cannot be answered by scope, schedule, and budget tracking alone. They require attention to software quality — specifically quality in use — from the earliest stages of project planning through post-deployment measurement.

The Public Accountability Standard

Washington State's OCIO guidance [2] frames IT investments as business-driven initiatives that must demonstrate value to the state. Quality in use — whether the system actually delivers improved outcomes — is the ultimate measure of whether that standard has been met. Every agency deploying technology has an obligation not just to deliver a project, but to deliver a system that serves Washington residents effectively.

3. Understanding Software Quality in a Government Context

Software quality is not a single thing. It is a multidimensional property of a system, encompassing how it is designed, how it is built, how it behaves under different conditions, and how well it serves the people who depend on it.

In a government context, this complexity is amplified. Washington State agencies operate systems that must serve diverse populations, integrate with legacy infrastructure, comply with evolving regulatory requirements, and remain accessible under budget and staffing constraints that limit ongoing maintenance capacity.

3.1 The ISO 25010 Quality Model

WaTech's Software Quality Best Practices Guideline (EA-01-02-G) [1] explicitly aligns with ISO/IEC 25010 and ISO/IEC 25023 as the foundation for software quality standards in Washington State. ISO 25010 defines software product quality in terms of eight characteristics that map directly onto the challenges Washington agencies face:

Quality Characteristic	Relevance to Washington State IT Projects
Functional Suitability	Does the system correctly implement the business rules, program logic, and data processing requirements the agency depends on?
Performance Efficiency	Does the system respond adequately under real production workloads, including peak periods such as benefit renewal cycles or legislative reporting deadlines?
Compatibility	Does the system integrate reliably with existing agency systems, OneWA/Workday, shared state infrastructure, and data exchanges across agency boundaries?
Usability	Can agency staff and, where applicable, members of the public accomplish their tasks effectively without excessive training or encountering confusing errors?
Reliability	Does the system remain available and correct over time, including recovery from failures, without data loss or corruption?
Security	Does the system protect sensitive data, enforce appropriate access controls, and comply with state and federal security requirements?

Quality Characteristic	Relevance to Washington State IT Projects
Maintainability	Can the system be understood, modified, tested, and extended by the agency or its vendors at reasonable cost over its operational lifetime?
Portability	Can the system be adapted to changing infrastructure, vendor environments, or platform requirements without excessive rework?

WaTech's guideline further references ISO/IEC 5055:2021 [1], which provides automated source code quality measures across four business-critical factors — Security, Reliability, Performance Efficiency, and Maintainability — and recommends that agencies use static analysis tools implementing this standard to detect, measure, and track weaknesses throughout the software lifecycle.

3.2 Product Quality vs. Quality in Use

ISO 25010 makes an important distinction that is highly relevant to government IT: the difference between product quality and quality in use.

Product quality refers to the intrinsic properties of the system — whether it is well-designed, well-tested, and well-implemented. Quality in use refers to the outcomes the system enables when it is actually deployed and used — whether it helps users accomplish their goals, whether it is effective in the context of the agency's program mission, and whether it produces the intended outcomes for the state and its residents.

Both dimensions matter, and both require deliberate attention during project planning and execution.

🔍 Distinction to Remember

Project quality asks: Did we deliver what we said we would, on time and on budget? Software quality asks: Does the system actually work well? Quality in use asks: Does the system help the agency achieve its mission and serve residents effectively? All three questions deserve answers — and only the last one measures whether the public investment was worthwhile.

4. The Washington State Policy Context

Washington State has developed a substantial policy framework for IT project quality. Understanding this framework — and applying it in practice — is essential for agencies undertaking significant technology initiatives.

4.1 WaTech Software Quality Best Practices Guideline (EA-01-02-G)

Published by WaTech and adopted by the State CIO on June 28, 2023, the Software Quality Best Practices Guideline [1] establishes the standards and practices Washington State agencies are expected to follow in delivering software. Key provisions relevant to project teams include:

- **Agencies must follow a defined software development process encompassing requirements, design, coding, testing, and deployment, with comprehensive and up-to-date documentation throughout.:** Process and Documentation [1, Section 1]
- **The guideline requires agencies to establish quality goals and thresholds for Security, Reliability, Performance Efficiency, and Maintainability based on ISO/IEC 5055, and to monitor and track quality throughout the system lifecycle using these measures.:** Measurement and Evaluation [1, Section 3]
- **Thorough testing throughout the development lifecycle is required, including unit, integration, system, and user acceptance testing, using appropriate techniques and methodologies aligned to the identified quality characteristics.:** Testing and Quality Assurance [1, Section 4]
- **Agencies must maintain a robust defect tracking system, prioritize and address defects promptly, and establish root cause analysis processes for recurring issues.:** Defect Management [1, Section 5]

For gated projects subject to Section 701 of the Washington State legislative budget, Sections 1–4 of the guideline are given priority review during the gating process, meaning these quality practices are not merely advisory but are evaluated as a condition of project approval and continued funding.

4.2 IT Investments Approval & Oversight Policy (PM-01)

WaTech's IT Investments Approval & Oversight Policy (PM-01) [2] establishes the roles and responsibilities of the OCIO and state agencies in planning and implementing major IT investments. It establishes a lifecycle-based oversight model that begins during feasibility and continues through implementation and closeout.

Critically, PM-01 frames IT investments as "business-driven" initiatives whose success is measured by the value they deliver — not merely by on-time, on-budget delivery. This framing reinforces the argument that software quality is not a technical nicety but a core component of investment performance.

4.3 Quality Assurance Standard (PM-03)

WaTech's Quality Assurance Standard (PM-03) [3] establishes minimum requirements for independent project quality assurance on major IT projects. The standard requires a QA provider to deliver a baseline QA plan within 30 days of engagement, covering methods, criteria, reporting templates, and timelines.

Under PM-03, regular QA reports must assess project performance across scope, schedule, budget, quality, team management, communication, governance, risk, vendor management, training, and deliverables. Notably, quality is treated as a distinct assessment dimension — separate from scope, schedule, and budget — reflecting the state's recognition that these metrics measure different things.

PM-03 also requires a lessons-learned report within 30 days of project completion to support continuous improvement across the state's technology portfolio.

Policy Alignment

Together, EA-01-02-G, PM-01, and PM-03 create a cohesive policy framework for software quality in Washington State. Agencies that align their project quality practices with these documents are not only following best practices — they are meeting explicit state requirements that are evaluated during project gating, oversight reviews, and QA assessments.

5. Where Quality Problems Come From

Understanding where software quality problems originate is essential to addressing them effectively. In government IT projects, quality failures rarely arise from a single cause. They typically reflect a combination of structural, process, and communication factors that compound over the course of the project lifecycle.

5.1 Gaps in the documentation of Full Business Intent

Many quality problems can be traced back to functional descriptions that are incomplete, ambiguous, or inconsistent with how the agency actually operates. This is particularly common in projects involving commercial off-the-shelf software, where the vendor's standard functionality may not align precisely with the agency's business processes.

No matter what methodology is employed, when descriptions are unclear, development teams and system integrators must make assumptions. Those assumptions may be technically reasonable but programmatically incorrect. The result is a system that passes functional testing — because the tests were written to match the implementation — but fails to support program operations as users expect.

WaTech's guideline [1, Section 1] addresses this directly, requiring "comprehensive and up-to-date documentation, including requirements specifications" as a baseline process practice. Addressing this requires rigorous requirements validation, including walkthroughs with subject matter experts, prototyping or demonstrations of key workflows, and explicit traceability from requirements to test cases.

5.2 Data Interface Complexity and the OneWA/Workday Challenge

Data interface quality represents one of the most technically demanding and consequential quality challenges facing Washington State agencies today. As agencies implement new systems — and as the state's OneWA program replaces core financial infrastructure with Workday — the number, complexity, and criticality of data interfaces across the state's technology portfolio is increasing dramatically.

5.2.1 The Scale of the OneWA/Workday Integration Challenge

The OneWA program, led by the Office of Financial Management, is replacing 1960s-era state administrative systems with Workday, a cloud-based ERP platform, across more than 100 Washington State agencies. Phase 1A, focused on replacing the Agency Financial Reporting System (AFRS) as the state's core accounting system, is anticipated to go live between July 2026 and January 2027 [4].

The scale of this undertaking is significant. AFRS currently processes approximately $4.3 billion in payments and 2.9 million payment transactions annually [5]. When Workday goes live, agencies will need to feed financial data into Workday directly or through legacy systems that interface with it. The Workday Financial Data Model (FDM) replaces the existing Chart of Accounts and will serve as the backbone for transactions, reporting, and security across all participating agencies [4].

This creates a complex web of data interface dependencies. Every agency system that currently exchanges data with AFRS or other systems being replaced by Workday must be assessed, redesigned, tested, and validated against the new integration needs. Systems that are not modified in time — or that are modified without adequate interface testing — risk data failures that could affect financial reporting, payroll processing, procurement, and other core administrative functions across the entire state.

> ⚠️ **OneWA Interface Risk**
>
> *The Washington State Auditor's Office identified one of the key risks to the OneWA implementation as the possibility that "the new system might not effectively communicate with legacy systems at state agencies" [5]. Interface failures in this context are not isolated technical issues; they could affect the accuracy of the state's financial statements, the reliability of payments to vendors and employees, and agencies' ability to report on program expenditures. For agencies with systems that interface with Workday, data interface quality must be treated as a first-priority quality risk.*

5.2.2 Common Interface Quality Failure Patterns

Integration and data interface problems often do not surface in unit testing or even in system testing conducted in isolated environments. They emerge when systems are tested together under realistic data conditions and realistic workloads. Common failure patterns in government ERP integrations include:

- Data format mismatches: Fields that exist in legacy systems do not map cleanly to Workday's Financial Data Model, causing records to be rejected, truncated, or silently miscategorized
- Chart of accounts translation errors: The replacement of AFRS Chart of Accounts codes with Workday FDM values creates translation risks wherever agency systems use legacy coding structures
- Timing and sequencing dependencies: Batch processes that assume a particular order of operations may fail when Workday's processing schedule differs from legacy system assumptions
- Authentication and authorization mismatches: Workday's security role model differs fundamentally from legacy system access controls, creating risks of both unauthorized access and blocked legitimate transactions
- Bidirectional data consistency failures: When both Workday and a legacy system hold overlapping data, inconsistencies accumulate over time unless interface logic enforces clear ownership and synchronization rules
- Reporting and reconciliation gaps: Data that was previously reconciled within a single system now crosses a system boundary, requiring new reconciliation processes to detect and resolve discrepancies

5.2.3 What Strong Interface Quality Practice Looks Like

The OneWA program's data conversion guidance [6] provides a useful framework for thinking about interface quality. It identifies data cleansing, validation rule design, mock conversion testing, and iterative

refinement as essential elements of quality data migration. These same principles apply to ongoing operational interfaces:

- Interface inventory and specification: All interfaces must be formally documented, including data element mappings, transformation logic, validation rules, error handling behavior, and volume/frequency characteristics
- Interface-specific test plans: Each interface requires its own test plan covering happy path, error condition, boundary case, and volume scenarios — not just a generic integration test
- Realistic test environments: Interface testing must use test data that reflects the real distribution and variety of production data, including historical edge cases and known data quality issues in legacy systems
- End-to-end reconciliation testing: Testing must verify that data arriving at Workday reconciles correctly with the source system — not just that the interface completes without error
- Parallel operation and rollback planning: For critical interfaces, parallel operation periods and defined rollback procedures are essential quality safeguards

5.3 Technical Debt: The Hidden Budget Threat

Technical debt is one of the most consequential and least discussed quality risks in government IT. It refers to the accumulation of implementation shortcuts, deferred improvements, and suboptimal design decisions that are made — often under schedule pressure — during a project's development and deployment phases.

In government IT, technical debt rarely stays contained. It compounds. And crucially for Washington State agencies operating under biennial budget cycles, it generates unbudgeted future projects that compete with new initiatives for limited capital resources.

5.3.1 How Technical Debt Accumulates

Technical debt accumulates through a variety of mechanisms that are common in government IT projects:

- Configuration workarounds: When a commercial platform like Workday cannot be configured to meet a specific requirement within the project timeline, teams implement workarounds — often manual processes or custom scripts — that are flagged as "temporary" but become permanent fixtures
- Deferred defect remediation: Defects that are discovered during testing but not resolved before go-live — because they are deemed "acceptable" given schedule constraints — accumulate in the production system and grow harder to fix as the system matures
- Inadequate documentation: Systems deployed without adequate technical documentation are harder to maintain, harder to train staff on, and harder to modify — increasing the cost of every future change
- Outdated dependencies: Systems built on components, libraries, or platform versions that are not kept current accumulate security and compatibility debt that eventually requires significant remediation investment

- Interface fragility: Interfaces built to meet immediate functional needs without regard for resilience, error handling, or future volume growth become brittle dependencies that break when anything in the environment changes

5.3.2 Technical Debt Creates Unbudgeted Projects

The most significant financial consequence of technical debt in government IT is not the cost of the debt itself — it is the cost of the unplanned projects it spawns. This is a pattern that Washington State agencies and the Legislature have encountered repeatedly, and it represents a direct threat to the value proposition of major technology investments.

When an agency deploys a system with significant accumulated technical debt, the following sequence becomes almost inevitable:

1. The system goes live with known issues and deferred improvements logged as "future enhancements."
2. The project closes, and the team disbands. Ownership of the deferred items is unclear.
3. The deferred items accumulate in the production system, becoming harder to address as they become embedded in operations.
4. A triggering event — a regulatory change, a security vulnerability, an interface failure, or simply the accumulation of too many operational workarounds — forces a remediation project.
5. The agency must request unplanned funding to address problems that should have been resolved during the original project — funding that was not anticipated in the decision package, was not planned in the biennial budget, and now competes with new initiatives.

This pattern is particularly damaging because it erodes the realized return on the original technology investment. The system that was supposed to modernize operations and deliver efficiencies is instead consuming maintenance and remediation resources that were not budgeted — often for years after the project was declared complete.

> 💰 **The Budget Cycle Trap**
>
> *Washington State's biennial budget cycle makes technical debt especially risky. Remediation needs that emerge mid-biennium cannot be addressed without supplemental requests or reallocation from other priorities. Agencies that deploy technically indebted systems are effectively pre-committing future budget capacity to remediation work they cannot yet see or estimate. The best defense is preventing technical debt from accumulating in the first place — by building quality in from the start, not deferring it.*

5.3.3 Technical Debt and OneWA Interface Sustainability

The intersection of technical debt and interface quality is particularly acute in the context of OneWA. Agencies that implement interfaces to Workday using approaches that are technically functional but not well-designed — fragile transformation logic, undocumented mappings, manual reconciliation steps —

are accumulating interface debt that will require remediation as Workday evolves, as agency systems change, and as data volumes and complexity grow.

Workday is a cloud-based SaaS solution that receives ongoing updates from the vendor. Interface designs that do not account for version compatibility, rely on undocumented Workday behaviors, or use integration patterns inconsistent with Workday's recommended architecture will require rework as the platform evolves — generating recurring, unbudgeted maintenance costs across the agencies involved.

5.4 Insufficient Testing Coverage

One of the most common contributors to quality failures in government IT is insufficient testing coverage. This can take several forms:

- Testing that focuses on happy path scenarios and does not adequately cover error conditions, boundary cases, or unusual but valid data
- Testing conducted in environments that do not reflect production conditions, leading to performance issues that only appear under real workloads
- Testing that covers functionality but not non-functional areas such as performance, security, or accessibility
- User acceptance testing conducted too late in the project lifecycle to allow meaningful remediation of issues discovered

WaTech's guideline [1, Section 4] is explicit: agencies must "conduct thorough testing throughout the software development lifecycle, including unit testing, integration testing, system testing, and user acceptance testing," using "appropriate testing techniques and methodologies to address the identified quality characteristics." Testing late and testing narrowly are direct violations of this guidance.

6. A Framework for Quality-Focused Project Execution

Improving software quality outcomes in Washington State IT projects does not require a fundamental overhaul of project management practice. It requires augmenting existing practice with quality-specific planning, measurement, and governance activities that complement and reinforce the project management framework already in place.

The following framework organizes these activities across four domains aligned with WaTech's quality guidance:

6.1 Process Quality

Process quality refers to the rigor and discipline of the processes used to plan, design, build, and test the system. WaTech's guideline [1, Section 1] establishes the baseline: a defined development process, comprehensive documentation, and consistent application of quality standards across the lifecycle.

Key elements of process quality include:

- A documented delivery methodology that defines how work moves through requirements, design, build, test, and deployment stages
- Defined entry and exit criteria for each phase, so that teams do not move forward with unresolved quality issues
- Defect management processes that ensure issues are tracked, prioritized, and resolved consistent with WaTech PM-03 requirements [3]
- Configuration management practices that maintain the integrity of the system as it evolves, especially critical for interface configurations in complex integrations
- Lessons learned and continuous improvement mechanisms, as required by PM-03 [3]

6.2 Testing Strategy and Coverage

Testing is the primary mechanism by which software quality is verified. WaTech's guideline [1, Section 4] requires thorough testing throughout the lifecycle. A comprehensive testing strategy for a Washington State IT project typically encompasses:

Testing Type	Purpose and Focus
Requirements Validation	Verifies that requirements are complete, unambiguous, testable, and consistent before development begins
Unit / Component Testing	Validates the behavior of individual components or modules in isolation

Testing Type	Purpose and Focus
Interface Testing	Validates data exchange behavior at each interface point, including format correctness, transformation accuracy, error handling, and reconciliation
Integration Testing	Validates end-to-end system behavior when components interact, including complex multi-system scenarios such as Workday integration flows
System Testing	Validates the complete system against functional and non-functional requirements in a production-representative environment
Performance Testing	Verifies response time, throughput, and scalability under expected and peak workloads — required by ISO/IEC 25010 and WaTech guidance [1]
Security Testing	Identifies vulnerabilities; verifies compliance with WaTech security requirements and applicable regulations (CJIS, IRS Pub 1075, HIPAA) [1, Section 6]
Data Integrity Testing	Verifies that data is correctly processed, stored, and transmitted across system boundaries, including Workday interface reconciliation
User Acceptance Testing	Verifies that the system meets user needs and program requirements in the context of actual agency operations
Regression Testing	Verifies that changes do not introduce unintended defects in previously working functionality

6.3 Software Product Quality

WaTech's guideline [1, Section 3] requires agencies to establish measurable quality goals and thresholds in accordance with ISO/IEC 5055. In practice, this means the system should have explicit, measurable quality requirements — not just functional requirements. Examples:

- Response time: "95% of page load requests must complete in under 3 seconds under a concurrent user load of 500 users"
- Availability: "The system must maintain 99.5% availability during business hours, measured monthly."
- Interface reliability: "The Workday financial interface must process nightly batch files with zero rejected records attributable to transformation errors."
- Technical debt threshold: "The system must not proceed to production with more than [X] open Severity 1 or 2 defects, and all identified technical debt items must be documented with remediation timelines."

- Accessibility: “All public-facing interfaces must meet WCAG 2.1 Level AA standards.”

6.4 Quality in Use

Quality in use — the outcomes the system enables in actual agency operations — is the ultimate measure of whether a technology investment has delivered value to Washington residents. Projects can take meaningful steps toward quality in use during the project lifecycle:

- Involving subject matter experts and end users in requirements development to ensure the system addresses real operational needs and reflects how work actually gets done
- Conducting user acceptance testing in conditions that reflect actual agency workflows, including realistic data volumes and operational scenarios, not idealized test cases
- Piloting the system with a subset of users before full rollout to identify usability and operational issues in a controlled setting
- Defining post-deployment success criteria tied explicitly to community and program outcomes before go-live — not just system performance metrics
- Measuring equity impacts: tracking whether the system serves all user populations effectively, including those with limited digital literacy, language needs, or disabilities

7. The Cost of Getting Quality Wrong

The business case for investing in software quality is compelling, and it rests on a well-documented empirical relationship: the cost of detecting and fixing a defect increases dramatically as the project progresses.

7.1 The Relative Cost of Defect Resolution

Research across software development contexts consistently shows that the cost of resolving a defect is lowest when it is caught early in the lifecycle. While precise cost ratios vary by project type and context, the directional pattern is consistent:

Stage of Detection	Relative Cost to Fix
Requirements / Design	1× (baseline)
Development / Build	5–10×
System Testing	10–20×
User Acceptance Testing	20–40×
Production	40–100× or more

For a Washington State agency managing a large technology initiative, this cost differential has direct budget implications. A defect that costs a few hours of developer time to fix during requirements review may require weeks of remediation effort, regression testing, potential re-deployment, and stakeholder communication if it reaches production. Multiply this across the full defect population of a complex system, and the investment required to address quality late in the lifecycle frequently exceeds the cost of building quality in from the beginning.

7.2 The Full Cost Picture: Technical Debt, Unbudgeted Projects, and Erosion of Benefits

Beyond direct defect remediation costs, poor software quality generates a cascade of costs that directly threatens the value proposition of technology investments:

- Unbudgeted remediation projects that consume capital resources intended for new initiatives, as technical debt forces agencies to return to systems that should have been stabilized during the original project
- Program impact costs from incorrect calculations, erroneous data, or failed transactions — including incorrect financial reporting in Workday, miscalculated benefits, or failed procurement processing
- Workaround costs as staff develop manual processes to compensate for system deficiencies, consuming staff time that should be redirected by the system improvement

- Interface maintenance costs such as fragile interface designs require repeated rework when either the agency system or Workday evolves
- Vendor dispute and renegotiation costs when quality failures lead to disagreements about contract performance
- Community harm costs that, while difficult to quantify, are real: residents who do not receive services correctly, businesses delayed by failed licensing systems, public trust damaged by visible system failures
- Opportunity costs when the agency's technical team is absorbed by remediation work rather than delivering planned enhancements or supporting new initiatives

> ⚠️ **The True Cost of Technical Debt in Government**
>
> *Each dollar of technical debt deferred during project implementation does not disappear — it reappears in the next biennium as an unplanned project request, an emergency remediation, or a growing inventory of manual workarounds. The Legislature and agency leadership who approved the original investment expected a return in the form of improved operations and community outcomes. Technical debt silently converts that expected return into future liability. Quality investment during the original project is the only reliable way to prevent this outcome.*

8. Practical Guidance for Washington State Agencies

The following guidance is intended for agency project teams, project sponsors, and IT leadership who plan or oversee technology initiatives. It is organized around actionable steps aligned with WaTech's policy framework.

8.1 During Project Initiation and Planning

- Include a quality management approach in the project charter and plan, defining who is responsible for quality, what standards apply, and how quality will be measured, consistent with WaTech EA-01-02-G [1] and PM-03 [3].
- Develop a testing strategy during planning, not later. The strategy must cover all relevant quality dimensions per WaTech guidance [1, Section 4], including interface testing, performance, security, and data integrity.
- Inventory all data interfaces, including all systems that exchange data with Workday or other enterprise platforms. Each interface is a quality risk that requires its own specification, test plan, and acceptance criteria.
- Define measurable quality requirements for the system, including thresholds for the ISO/IEC 5055 quality factors identified in EA-01-02-G [1, Section 3]: Security, Reliability, Performance Efficiency, and Maintainability.
- Explicitly identify and document technical debt risks. Establish a policy for tracking, prioritizing, and resolving technical debt items — with a clear commitment that items cannot be deferred indefinitely to future biennia.
- Include quality milestones and quality gates in the project schedule. For Section 701 gated projects, ensure Sections 1–4 of EA-01-02-G are demonstrably addressed in the project's quality plan.
- Identify clearly who is responsible for each of the planning and execution deliverables.

8.2 During Project Execution

- Track defect discovery and resolution rates as leading indicators of quality health. A rising open defect count late in the project is a warning sign. This must be done at the project level.
- Conduct interface-specific testing early and repeatedly. Do not wait for system testing to discover interface issues — by then, remediation capacity is extremely limited.
- Use static analysis tools implementing ISO/IEC 5055 as required by EA-01-02-G [1, Section 3] to detect and track code quality weaknesses, and address findings before they become production defects.
- Report on software quality status alongside quality assurance, schedule, and budget status in project governance forums. Quality metrics must be visible to project sponsors and oversight bodies, consistent with PM-03 requirements [3].

- Actively manage technical debt during execution. When shortcuts are taken, log them explicitly with estimated remediation cost and timeline. Do not allow technical debt to accumulate invisibly or without future budget impact.

8.3 During Project Closeout and Transition

- Ensure quality acceptance criteria, including those derived from WaTech's ISO/IEC 5055-based thresholds [1], have been met before approving deployment.
- Formally document all known technical debt items with estimated remediation costs and proposed budget cycle for resolution. Agencies should not accept a system into production without an explicit plan for addressing outstanding debt.
- For Workday interfaces specifically, conduct end-to-end reconciliation testing that validates data accuracy across the full interface chain — not just error-free transmission.
- Define post-deployment monitoring mechanisms, including data quality dashboards, interface reconciliation reports, and community outcome metrics.
- Deliver the lessons-learned report required by PM-03 [3] within 30 days of project completion, with specific attention to quality issues encountered, interface challenges, and technical debt decisions and impact made during execution.

9. The Washington State Software Quality Checklist

To support agencies in evaluating their project's approach to software quality, Critical Logic developed the Washington State Software Quality Checklist for IT Projects. The checklist is designed as a practical self-assessment tool that project teams can use at any stage of the project lifecycle, but particularly during planning and early execution. It is aligned with WaTech's EA-01-02-G, PM-01, and PM-03 requirements.

9.1 What the Checklist Covers

- **Evaluates the rigor of the project's delivery processes, including requirements management, design review, change management, and defect management practices against EA-01-02-G Section 1 requirements:** Process Quality
- **Evaluates whether the project has a comprehensive testing strategy covering functional, interface, integration, performance, security, and user acceptance testing as required by EA-01-02-G Section 4:** Testing Strategy and Coverage
- **Evaluates whether measurable quality requirements and ISO/IEC 5055-based thresholds have been established as called for in EA-01-02-G Section 3:** Software Product Quality
- **Evaluates whether all data interfaces — including Workday integration points — have been inventoried, specified, and included in testing plans with reconciliation criteria.:** Data Interface Quality
- **Evaluates whether the project has a documented approach to identifying, tracking, and planning for the resolution of technical debt, preventing unbudgeted future project obligations.:** Technical Debt Management
- **Evaluates whether the project is taking steps to verify that the system will support agency program outcomes and community benefit, including equity considerations and post-deployment measurement planning.:** Quality in Use

9.2 How to Use the Checklist

The checklist is most valuable when used as a structured conversation tool rather than a compliance exercise. Project teams are encouraged to work through it with both technical and program stakeholders, since quality gaps often arise at the boundary between technical implementation and program requirements.

Areas where the checklist reveals gaps should be treated as risk items requiring a response. Agencies may find it useful to apply the checklist at multiple points: during initial planning, at mid-project, and prior to deployment to confirm quality acceptance criteria have been met. For Section 701 gated projects, the checklist provides a practical preparation tool for the gating review of EA-01-02-G Sections 1–4.

10. Conclusion

Washington State agencies are making significant technology investments — in new systems, modernization efforts, commercial platform deployments, and integrations that span agency boundaries. The success of these investments depends on more than sound project management.

It depends on the quality of the software being delivered.

Scope, schedule, and budget are necessary conditions for project success. They are not sufficient conditions for system success. Agencies that treat software quality as a lifecycle discipline — planning for it early, measuring it consistently, holding it to explicit standards, managing technical debt proactively, and validating data interfaces rigorously — significantly improve their chances of delivering systems that perform reliably, serve residents equitably, and provide lasting value to Washington State.

The OneWA/Workday implementation is the most significant infrastructure change Washington State has undertaken in a generation. The data interfaces that agencies build to connect their systems to Workday will define the reliability of state financial operations for years to come. The technical debt decisions made during these implementations will shape unplanned budget requests for the next decade. Getting quality right now is not optional — it is the difference between an investment that delivers its promised return and one that generates ongoing remediation costs that erode the original benefit.

Washington State has the policy framework, in EA-01-02-G, PM-01, and PM-03, to support a quality-first approach. Critical Logic's Washington State Software Quality Checklist provides a practical starting point for agencies that want to evaluate where they stand and take targeted action to strengthen their quality posture.

> 🚀 **Next Step**
>
> *Download the Washington State Software Quality Checklist for IT Projects from Critical Logic. Use it in your next project planning session to identify quality gaps, assess interface risks, and build a defensible technical debt management plan — before those gaps become the unbudgeted projects of the next biennium.*

The question is not whether software quality matters. It clearly does. The question is whether your agency is treating it with the deliberate, proactive attention that Washington residents deserve.

References

All WaTech and OCIO policy documents referenced below are publicly available at watech.wa.gov and ocio.wa.gov.

[1] WaTech. Software Quality Best Practices Guideline (EA-01-02-G). State CIO Adopted: June 28, 2023. Sunset Review: June 28, 2026. *https://watech.wa.gov/sites/default/files/2024-10/EA-01-02-G%20Software%20Quality%20Best%20Practices.pdf*

[2] WaTech. IT Investments Approval & Oversight Policy (PM-01). Washington Technology Solutions. *https://watech.wa.gov/it-project-oversight-transformation*

[3] WaTech. Minimum Project QA Activities Standard (PM-03-03-S). Washington Technology Solutions. *https://watech.wa.gov/policies/minimum-project-qa-activities*

[4] Office of Financial Management (OFM). Workday Implementation — Phase 1A: Core Financials. Washington State. *https://ofm.wa.gov/it-systems/workday*

[5] Washington State Auditor's Office. One Washington — Strengthening Plans for Reliable Financial Statements. Performance Audit. *https://sao.wa.gov/reports-data/audit-reports/one-washington-strengthening-plans-reliable-financial-statements*

[6] One Washington Program. Data Conversion Overview. Office of Financial Management, August 2020. *https://one.wa.gov/sites/default/files/2022-07/Data_Conversion_Overview.pdf*

[7] ISO/IEC 25010:2011. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. International Organization for Standardization.

[8] ISO/IEC 5055:2021. Information technology — Software measurement — Software quality measurement — Automated source code quality measures. International Organization for Standardization.

About Critical Logic

Critical Logic is a Washington State-based technology consulting firm specializing in software quality assurance, independent verification and validation, and IT project advisory services for government agencies. We work with Washington State agencies under a statewide IT Development contract, DES 16322, to improve the quality outcomes of their technology investments through structured quality planning, testing strategy development, data interface quality assessment, and independent quality review.

About the Washington State Software Quality Checklist

The Washington State Software Quality Checklist for IT Projects was developed by Critical Logic in alignment with WaTech's EA-01-02-G, PM-01, and PM-03 requirements. It is designed as a practical tool for agency project teams, sponsors, and oversight personnel who want to evaluate and strengthen their approach to software quality — including data interface quality and technical debt management in the context of OneWA/Workday implementations.

Contact

For more information about this white paper, the Software Quality Checklist, or Critical Logic's services for Washington State agencies, visit criticallogic.com or contact us through our website CriticalLogic.com.

Disclaimer: This white paper was prepared by Critical Logic and reflects their views and opinions only. It does not represent the official policy, position, or endorsement of Washington State, the Office of the Chief Information Officer (OCIO), Washington Technology Solutions (WaTech), the Office of Financial Management (OFM), or any other state agency. References to WaTech and OCIO guidelines, the OneWA program, and other state initiatives are included solely for informational purposes. Readers should consult official state guidance and their agency's legal and policy advisors for authoritative advice on compliance requirements.